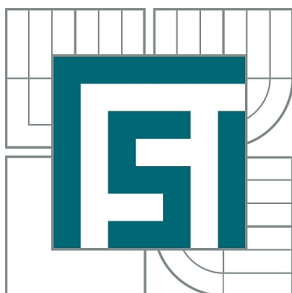


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV AUTOMATIZACE A INFORMATIKY

FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

OPTIMALIZACE ŘÍDICÍHO ALGORITMU POMOCÍ EVOLUČNÍHO ALGORITMU

OPTIMIZATION OF CONTROL ALGORITHM BY EVOLUTIONARY ALGORITHM

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. STANISLAV LANG

VEDOUCÍ PRÁCE
SUPERVISOR

prof. Ing. PAVEL OŠMERA, CSc.

BRNO 2010

Vysoké učení technické v Brně, Fakulta strojního inženýrství

Ústav automatizace a informatiky

Akademický rok: 2009/2010

ZADÁNÍ DIPLOMOVÉ PRÁCE

student(ka): Bc. Stanislav Lang

který/která studuje v **magisterském navazujícím studijním programu**

obor: **Aplikovaná informatika a řízení (3902T001)**

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Optimalizace řídicího algoritmu pomocí evolučního algoritmu

v anglickém jazyce:

Optimalization of control algorithm by evolutionary algorithm

Stručná charakteristika problematiky úkolu:

Optimalizujte řídicí algoritmus pro soustavu druhého řádu s dopravním zpožděním pomocí evolučního algoritmu

Cíle diplomové práce:

Automatický návrh parametrů regulátoru pomocí evolučních algoritmů

Seznam odborné literatury:

J.Balátě: Automatické řízení, BEN 2004

I.Zelinka,Z.Oplatková, M.Šeda, P. Ošmera: Evoluční výpočty a jejich aplikace, BEN, 2008

Vedoucí diplomové práce: prof. Ing. Pavel Ošmera, CSc.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2009/2010.

V Brně, dne

L.S.

Ing. Jan Roupec, Ph.D.
Ředitel ústavu

prof. RNDr. Miroslav Doupovec, CSc.
Děkan fakulty

ABSTRAKT

Práce se věnuje možnostem využití evolučních výpočetních technik v oblasti automatizace. Teoretická část práce popisuje používané techniky v automatizaci a optimalizaci. Praktická část uvedené disciplíny propojuje, výstupem práce je program pro automatický návrh parametrů regulátoru s využitím genetického algoritmu.

ABSTRACT

My thesis deals with possibilities of using evolutionary computation in the field of automation. The theoretical part of the thesis describes the techniques used in the automation and optimization. The practical part of the thesis connects these two disciplines, the output of this work is a program for automatic design of parameters of regulator using a genetic algorithm.

KLÍČOVÁ SLOVA

Evoluční, genetický, optimalizace, parametry, regulace, regulátor, zpoždění.

KEYWORDS

Evolutionary, genetic, optimization, parameters, control, controller, delay.

PODĚKOVÁNÍ

Děkuji prof. Ing. Pavlu Ošmerovi za cenné rady a připomínky k vypracování diplomové práce i za čas mně věnovaný při konzultacích. Také děkuji své rodině za podporu nejen při studiu.

V Brně, dne 27. 5. 2010

Bc. Stanislav Lang

Obsah:

	Zadání závěrečné práce.....	3
	Abstrakt.....	5
	Poděkování.....	7
1	Úvod.....	11
2	Řízení a regulace.....	13
2.1	Základní pojmy.....	13
2.2	Řízení a regulace.....	14
2.3	Technické prostředky regulace.....	16
2.4	Technická realizace elektronických regulátorů.....	17
2.5	Klasické a moderní metody v regulaci.....	19
2.5.1	Klasické regulátory PID.....	19
2.5.2	Regulátory navržené na požadovaný přenos.....	19
2.5.3	Nelineární regulátory.....	19
2.5.4	Fuzzy.....	20
2.5.5	Neuronové sítě.....	20
3	Klasické metody návrhu regulátorů.....	21
3.1	Stanovení přenosu soustavy.....	21
3.2	Identifikace.....	21
3.2.1	Identifikace z přechodové charakteristiky.....	21
3.2.2	Identifikace pomocí korelačních metod.....	22
3.2.3	Identifikace z frekvenční charakteristiky.....	22
3.2.4	Identifikace pomocí spektrální analýzy.....	23
3.2.5	Identifikace pomocí metody nejmenších čtverců.....	23
3.2.6	Identifikace pomocí online metody nejmenších čtverců.....	23
3.3	Volba typu regulátoru a technických prostředků.....	24
3.4	Návrh parametrů regulátoru.....	24
3.4.1	Návrh ve frekvenční oblasti.....	25
3.4.2	Optimální modul.....	25
3.4.3	„Pole placement“.....	25
3.4.4	Algebraická polynomiální teorie.....	26
3.4.5	Integrální kritéria.....	26
3.4.6	Kvadratické integrální kritérium.....	26
3.4.7	Lineární (usměrněné) integrální kritérium.....	27
3.4.8	Integrální kritérium ITAE.....	27
4	Dopravní zpoždění a základní nelinearity.....	29
4.1	Dopravní zpoždění.....	29
4.2	Aproximace Padeho rozvojem.....	30
4.3	Základní nelinearity.....	30
4.3.1	Saturace.....	31
4.3.2	Pásmo necitlivosti.....	31
4.3.3	Kvantování v hodnotě.....	31
4.3.4	Hystereze.....	32
4.4	Možnost využití klasických metod.....	32
5	Evoluční techniky v optimalizaci.....	33
5.1	Pojmy.....	34
5.1.1	Účelová funkce.....	34
5.1.2	Jedinec.....	34
5.1.3	Fitness.....	34
5.1.4	Populace.....	34

5.2	Vybrané metody evolučních technik.....	34
5.2.1	Metoda náhodné procházky.....	34
5.2.2	Horolezecký algoritmus.....	35
5.2.3	Simulované žíhání.....	35
5.2.4	Zakázané prohledávání („Tabu search“).	36
5.2.5	Genetické algoritmy.....	36
5.2.6	Rojení částic.....	37
5.2.7	Optimalizace mravenčí kolonií.....	37
6	Využití evolučních technik při návrhu regulátorů.....	39
6.1	Vymezení rozsahu práce.....	39
6.2	Požadavky na program.....	40
6.2.1	Specifikace soustavy a nelinearit.....	40
6.2.2	Nastavení požadavků na regulační děj (kriteriální funkce).....	40
6.2.3	Vizualizace procesu optimalizace.....	40
6.2.4	Uživatelsky přívětivé prostředí.....	41
6.3	Volba vývojového prostředí.....	41
6.3.1	GUIDE.....	41
6.3.2	Genetic toolbox.....	42
6.4	Výběr evolučního algoritmu.....	42
7	Realizace programu pro optimalizaci regulátorů pomocí genetického algoritmu.	43
7.1	Vytvoření kriteriální funkce.....	43
7.1.1	Simulační model.....	44
7.1.2	Tvary optimalizovaných regulátorů.....	46
7.1.3	Výpočet hodnoty „fitness“.....	49
7.2	Uživatelské rozhraní programu.....	51
7.2.1	Ovládací tlačítka.....	51
7.2.2	Nástroje pro specifikaci optimalizačního problému.....	53
7.2.3	Reprezentace výsledků.....	55
7.2.4	Zprávy pro uživatele.....	56
7.3	Testování.....	57
7.4	Možnosti pokračování v práci.....	60
8	Závěr.....	61
9	Seznam použité literatury.....	63

1 ÚVOD

Práce je věnována problematice optimalizace regulátorů s využitím evolučních algoritmů, spojuje tak oblast optimalizace regulátorů a oblast evolučních výpočetních technik. Uvedené řešení není sice zcela nové, není však ani příliš obvyklé, neboť evoluční techniky se v automatizaci zatím výraznějším způsobem neprosadily.

První část (asi poloviny rozsahu) práce je teoretickým úvodem a zároveň určitou rešerší používaných přístupů v oblasti automatizace a v oblasti evolučních výpočetních technik. První teoretická kapitola je věnována obecně problematice regulace. Druhá kapitola se zabývá používanými metodami návrhu regulátorů pro lineární systémy, následuje kapitola pojednávající o problematice vlivu dopravního zpoždění a nelinearit a omezení klasických možností návrhu regulátorů (klasickými přístupy). Poslední kapitola spadající do teoretické části je věnována evolučním technikám v optimalizaci.

Druhá část práce diskutuje požadavky na systém provádějící automatický návrh regulátorů pomocí evolučních technik. Následuje poslední kapitola popisující vlastní implementaci programu v prostředí MatLab. Popis implementace zahrnuje jednak vytvoření kritériální funkce, ovládání vytvořeného uživatelského rozhraní a diskusi výsledků dosažených činností vytvořeného programu.

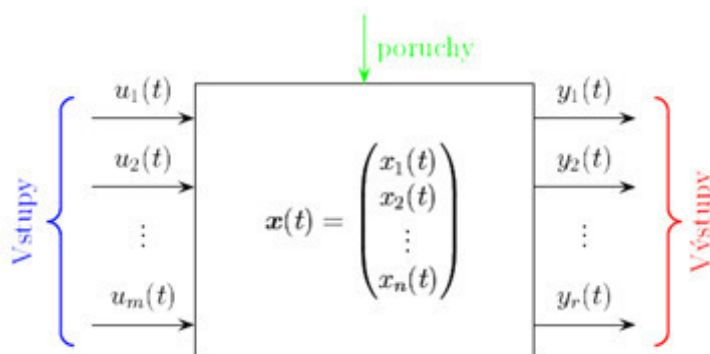
2 ŘÍZENÍ A REGULACE

Problematika řízení proniká do většiny oblastí lidské činnosti, od řízení strojů po řízení lidských zdrojů a ekonomiky. Entitu, která je předmětem našeho zájmu a kterou chceme řídit, nazýváme v teorii řízení obecně „systém“. Do systému vstupují „vstupní veličiny“ ve formě energií, materiálních, finančních či lidských zdrojů a na výstupu soustavy se objevují produkty či energie – „výstupní veličiny“. Cílem činnosti řízení je dosažení požadovaných výstupních veličin působením na systém veličinami vstupními.

2.1 Základní pojmy

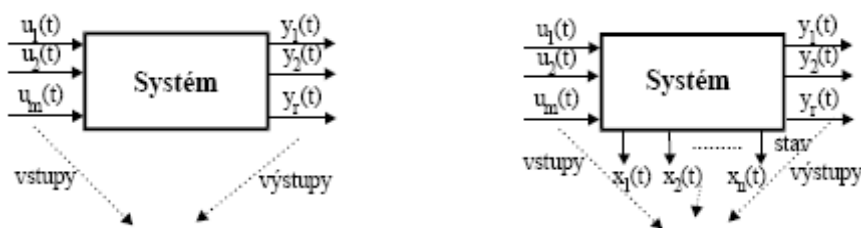
Systém je charakterizován svými statickými a dynamickými vlastnostmi. K vystižení vlastností systému slouží primárně diferenciální rovnice. V zásadě rozlišujeme dva základní typy popisu systému, sice „vnitřní“ a „vnější“.

Vnitřní nebo též stavový popis definuje systém jako množinu vstupů, množinou vnitřních stavů, množinou výstupů a množinou funkcí popisujících vazby vstupů a stavů na stavy a vazby vstupů a stavů na výstupy. Vnitřní popis je jednoznačný, neboť popisuje i vnitřní strukturu systému. Podrobněji se problematikou vnitřního popisu systémů zabývají publikace [2], [4], [6] – [10].



Obr. 1 Obecný systém [8]

Vnější popis pohlíží na systém jako „black box“ tzn. černou krabíčku, kdy nevíme, co se v krabici skrývá, umíme však popsat jak se systém chová, tj. známe vztah mezi vstupem a výstupem a na základě průběhu vstupní veličiny můžeme určit průběh veličin výstupních. Popis zahrnuje pouze funkční závislost výstupů na vstupech a vnitřní stavy neuvažuje, nelze proto postihnout chování systému za nenulových počátečních podmínek (kdy např. některý z vnitřních akumulátorů energie není na nulovém potenciálu). Vnější popis je obecnější, sjednocuje množiny systémů se stejným vstupně/výstupním chováním nehlédě na jejich odlišnou vnitřní strukturu. Vedle popisu diferenciálními rovnicemi, využíváme operátorové či frekvenční přenosy, přechodové, impulzní či frekvenční charakteristiky nebo rozložení pólů a nul systému. Všechny uvedené formy popisu jsou co do informačního obsahu rovnocenné a je možno mezi nimi přecházet.



Obr. 2 Vnější a vnitřní popis systému [6]

Přecházet můžeme i mezi vnitřním a vnějším popisem. Převod stavového popisu na některou z forem popisu vnějšího je vždy jednoznačný, dochází zde ke ztrátě informace o vnitřním provedení systému. Opačný převod je již nutně nejednoznačný, nejčastěji se využívá tzv. paralelního, sériového či přímého programování. Stavový popis však již (zpravidla) neodpovídá vnitřní struktuře.

Požadujeme, aby systém, který řídíme, byl stabilní. Základní stabilita systému z vnějšího pohledu je definována jako „BIBO“ (ohraničený vstup ohraničený výstup), to znamená, že na vstupní signál, který je omezený v amplitudě a čase, soustava odpoví signálem výstupním, který je též omezený v amplitudě a čase. Pro vnitřní popis je toto kritérium přísnější, omezený musí být každý z vnitřních stavů, tj. hodnota žádného stavu nesmí růst nad všechny meze.

Systémy dále dělíme na lineární a nelineární. Základní vlastností lineárních systémů je platnost pravidla „superpozice“, to znamená, že odezva systému na vstupní signál složený ze dvou dílčích signálů je totožná jako součet dílčích odezev systému na dané parciální vstupní signály. Ve stavovém popisu existuje pro lineární systémy jediný „globální“ ustálený stav.

Nelineární systémy pravidlo superpozice nezachovávají a připouští existenci více ustálených stavů. Za ustálený stav považujeme u nelineárních systémů i periodická řešení, která mají ve stavovém prostoru tvar uzavřené křivky. Jednotlivé ustálené stavy mohou být stabilní či labilní. Blíže se problematikou zabývá literatura [2], [9].

V neposlední řadě rozlišujeme systémy na spojité a diskrétní. Termín „spojitý systém“ a „nespojité systém“ je již zažitý, správně bychom měli mluvit o systému pracujícím ve spojitém či diskrétním čase. Spojitý systém je popsán diferenciálními rovnicemi, řešením diferenciálních rovnic jsou spojité funkce (popisující závislost mezi vstupy a výstupy) a hodnoty výstupů, případně i stavů, jsou definovány v každém bodě časové osy.

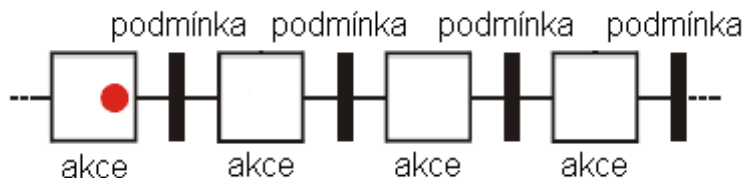
Diskrétní systémy jsou popsány rovnicemi diferenčními a místo Laplaceova obrazu časové oblasti se využívá Z-transformace. Hodnoty výstupů, případně stavů, jsou definovány pouze v časových hodnotách daných periodou vzorkování. Hodnoty mezi okamžiky vzorkování Z-transformace nepostihne. Pokud potřebujeme znát hodnoty mezi okamžiky vzorkování, musíme je doplnit, k čemuž slouží „tvarovací členy“.

2.2 Řízení a regulace

Činnost, jejímž výsledkem je dosažení požadovaného chování systému, nazýváme řízením. Pojem řízení je velmi široký, proto jej členíme na tři základní oblasti:

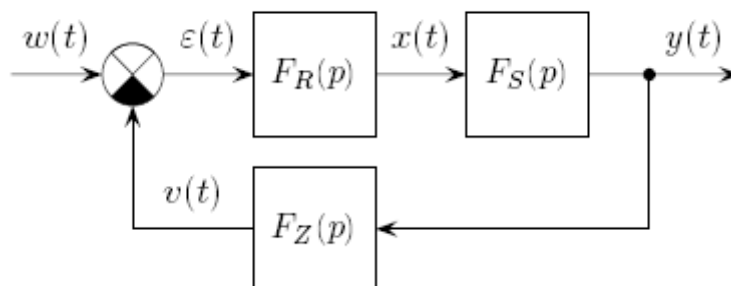
1. Programové řízení
2. (Zpětnovazební) regulace
3. Ovládání (přímovazební regulace)

Programové řízení je sledem operací, které jsou vykonávány sekvenčně. Jedná se o systém diskrétních událostí, příkladem může být automatizovaná výrobní linka, kdy výrobek prochází sérií výrobních operací.



Obr. 3 Schematické znázornění dílčích kroků a podmínek přechodů mezi nimi

Regulace (též zpětnovazební regulace) je formou řízení, kdy řídicí systém má informaci o stavu výstupů systému. Řídicí systém – regulátor sbírá informaci o žádaných hodnotách a skutečných výstupních (případně i stavových) hodnotách a na základě průběhu těchto hodnot provádí operace řízení tzv. „akční zásahy“.



Obr. 4 Zpětnovazební regulační smyčka [8]

Soustředíme-li se na vstupně/výstupní chování systému, tj. vnější popis, postačí, bude-li regulátor pracovat s regulační odchylkou (odchylkami – v případě vícerozměrného systému). Regulační odchylka je rozdílem hodnoty žádané a hodnoty skutečné.

S úlohou zpětnovazební regulace se setkáváme nejčastěji na nejnižší úrovni řízení (řízení pohonů, tepelných soustav...). Pokud pojmem z pohledu vnějšího popisu regulátor a původní systém jako nový celek (systém), můžeme říci, že regulátor mění vlastnosti původního přenosu na nové. Vlastnosti zpětnovazební smyčky popisuje následující operátorový přenos.

$$F_w(p) = \frac{Y(p)}{W(p)} = \frac{F_R(p)F_S(p)}{1 + F_R(p)F_S(p)F_Z(p)}$$

Třetí kategorií řízení je ovládání, kdy působíme na systém prostřednictvím ovládacích prvků tak, abychom dosáhli požadovaného chování, systém sám o sobě nemá informaci o svém stavu. K řízení je třeba lidského faktoru, člověka, který posoudí, zda bylo dosaženo kýženého cíle řízení.



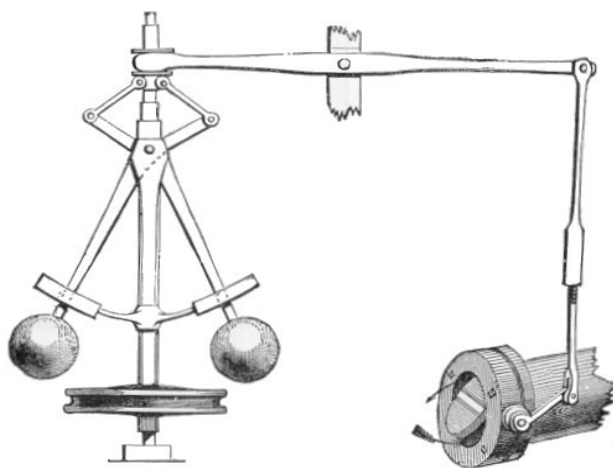
Obr. 5 Ovládání (přímovazební regulace)

Ovládací člen (někdy též regulátor v přímé vazbě) též mění vlastnosti přenosu soustavy, není zde však informace o skutečném stavu.

2.3 Technické prostředky regulace

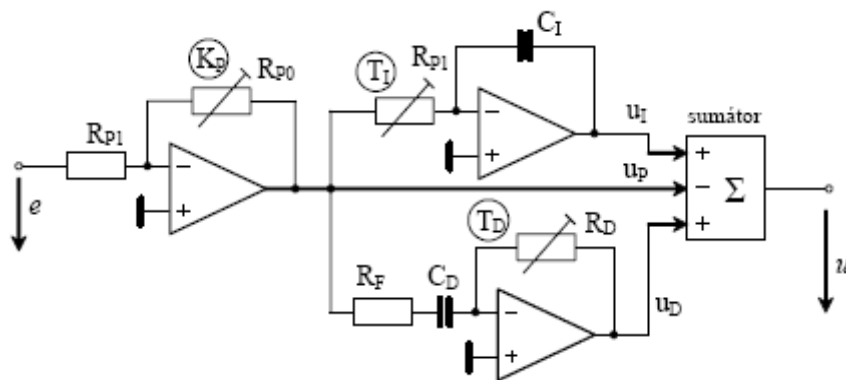
První zmínky o regulaci spadají do období starověku. „Tak např. Už od starodávna používali mlynáři na vodních a větrných mlýnech jednoduché zařízení, které regulovalo přísun zrní mezi mlýnské kameny v závislosti na jejich otáčkách.“ [13]

V novodobém pojetí se obvykle za první regulátor považuje „Wattův odstředivý regulátor“, který měl zajistit konstantní otáčky parního motoru při proměnlivém zatížení.



Obr. 6 Wattův odstředivý regulátor [14]

Vedle mechanických regulátorů se objevily v novodobé historii regulátory elektrické, které pracovaly (stejně jako regulátory mechanické) ve spojitém čase. Ústředním prvkem elektrických regulátorů byly operační zesilovače.

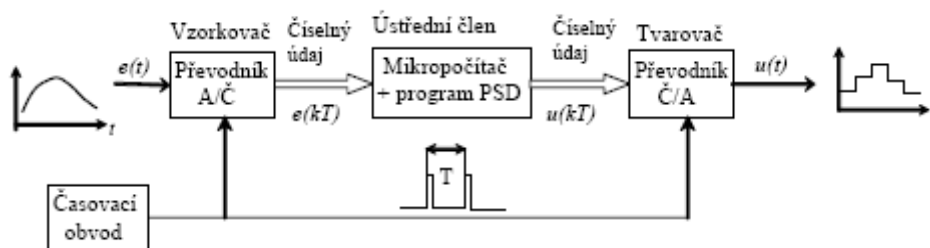


Obr. 7 Realizace PID regulátoru s operačními zesilovači [15]

V posledních desetiletích s prudkým rozvojem výpočetní techniky obsadily pole regulační techniky regulátory elektronické. Elektronické regulátory se skládají z výpočetní jednotky (mikropočítač, počítač,...) a z vstupních a výstupních obvodů. Zatímco analogové regulátory pracovaly spojitě, digitální zařízení může pracovat pouze s hodnotami diskretními (počítač nemůže pojmout hodnoty ve spojitém čase, protože by musel sbírat informace a je zpracovávat v časovém úseku v limitě nulovém, což je nerealizovatelný požadavek).

V diskretním čase je změřena analogová veličina, hodnota je převedena do digitální formy (A/D převodníkem), na základě změřené hodnoty (hodnot) je vypočítán akční zásah, který je předán

výstupním obvodům. Téměř výhradně se na výstupu používá tvarování 0. řádu, tj. klasický D/A převodník, který spočítanou hodnotu „drží“ po dobu celé periody.



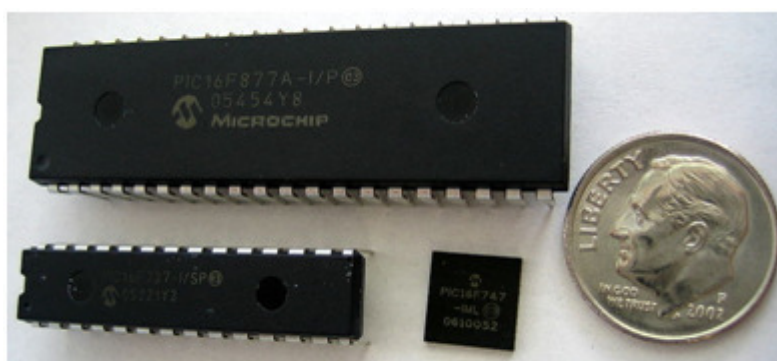
Obr. 8 Číslicový regulátor [15]

Přesto že elektronické regulátory pracují diskrétně, můžeme je v některých případech považovat za spojitě pracující, je-li perioda vzorkování o několik řádů nižší než nejmenší (uvažovaná) časová konstanta systému. V opačném případě je nutno regulátor navrhovat jako diskrétní (perioda vzorkování diskrétních regulátorů se volí často jako třetina nejmenší časové konstanty systému).

2.4 Technická realizace elektronických regulátorů

Základní struktura elektronického regulátoru je na obrázku Obr. 8. Technicky je regulátor tvořen výpočetním zařízením menšího či většího rozsahu, počínaje od mikrokontrolérů přes průmyslové počítače až po decentralizované řídicí systémy.

Mikrokontrolér je nejjednodušší formou výpočetního zařízení, v jediném čipu je integrována procesorová jednotka, sběrnice, paměť a periferie. Výpočetní výkon je relativně nižší (ve srovnání např. s klasickým PC). Jelikož však zpravidla není požadována žádná vizualizace, tento výkon je dostatečný pro zpracování dat. Většinou na mikropočítači běží cyklicky jeden program a zvláštní úlohy jsou vykonávány v obsluze *přerušení*, výjimkou však nejsou ani mikrokontroléry, na kterých je provozován jednoduchý operační systém.



Obr. 9 Mikrokontroléry – ilustrační obrázek [17]

Další možností technické realizace je použití PLC (programovatelný logický automat). Jeho výpočetní výkon je ve srovnání s mikrokontroléry mnohem vyšší, obsahuje větší paměť a větší množství periférií. PLC jsou vyráběny jako celky nebo v modulární formě.



Obr. 10 PLC – ilustrační obrázek [15]

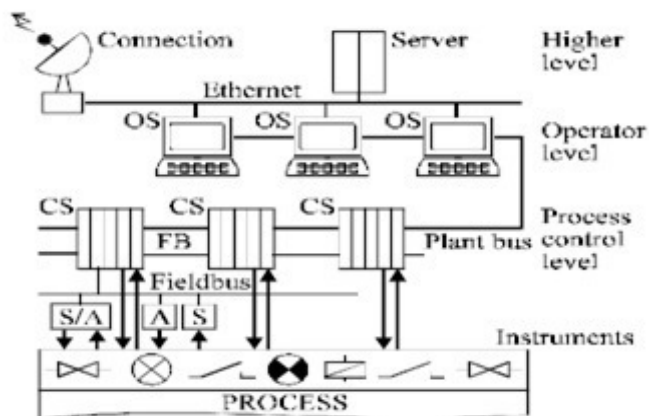
PLC je velmi robustní zařízení určené pro provoz v těžkých průmyslových podmínkách. PLC jsou doplňována o uživatelská rozhraní HMI která se skládají z obrazovky a uživatelských tlačítek, rovněž může být PLC integrováno do nadřazeného systému typu SCADA.

Průmyslové počítače, jsou výkonnými a robustními obdobami klasických PC. Je na nich provozován operační systém (většinou s požadavky na práci v „real time“), což nabízí uživateli nové možnosti, avšak na druhou stranu zvyšuje riziko vzniku možných problémů souvisejících s nedokonalostí operačního systému.



Obr. 11 Industrial PC – ilustrační obrázek [18], [19]

V neposlední řadě je vhodné zmínit decentralizované řídicí systémy, kdy jednotlivé moduly jsou rozmístěny na relativně velkém prostoru a komunikují vzájemně prostřednictvím některé průmyslové sběrnice (průmyslový ethernet, CAN,...)



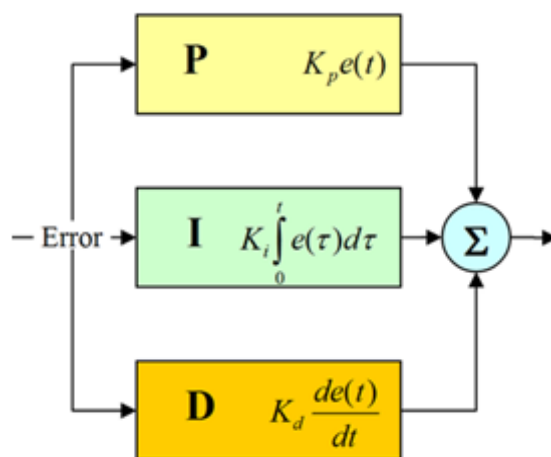
Obr. 12 Schéma DCS 2. generace [20]

2.5 Klasické a moderní metody v regulaci

S příchodem moderní výpočetní techniky se objevily možnosti aplikace složitých řídicích algoritmů, vedle nich však stále přetrvávají původní osvědčené přístupy, které mnohdy nepřinášejí tak široké možnosti regulace, zato jsou jednoduché a robustní.

2.5.1 Klasické regulátory PID

PID představují jednu z nejstarších skupin regulátorů, zahrnující kombinace tří základních složek P, I a D. Velká proporcionální složka „P“ zajišťuje rychlý regulační děj, zároveň však působí velkou chybou při působení poruchy. Integrační složka „I“ se stará o zajištění nulové ustálené odchylky řízení i poruchy (je-li obsažena v regulátoru), působí snížení celkové stability a rychlosti přechodového děje. Derivační složka „D“ naopak přechodový děj zrychluje a má stabilizující účinky, působí však přechodový děj a především akční zásah více kmitavý a zesiluje parazitní šum, proto je třeba v některých případech volit derivační složku malou nebo použít „filtrování derivační složky“.



Obr. 13 Regulátor PID [16]

Vhodnou kombinací složek lze vytvořit poměrně velmi robustní a kvalitní regulátor. PID je nejrozšířenějším regulátorem v oblasti průmyslu, a přestože se objevilo mnoho nových sofistikovaných metod, pro svou jednoduchost a spolehlivost si PID drží svou pozici.

2.5.2 Regulátory navrhované na požadovaný přenos

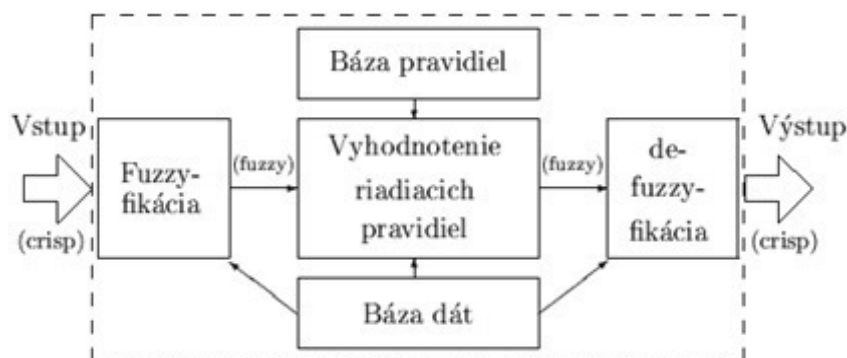
Jedná se o skupinu regulátorů, které lze popsat operátorovým přenosem jako klasický lineární systém (do této skupiny samozřejmě spadají i regulátory PID). Póly a nuly regulátoru jsou rozmístěny tak, aby zpětnovazební systém měl požadované vlastnosti. Stejně jako v případě PID se využívá při návrhu různých metod, je vhodné při návrhu kontrolovat robustnost celkového systému. O metodách návrhu bude zmínka v následující kapitole věnované právě klasickým přístupům návrhu regulátorů.

2.5.3 Nelineární regulátory

Nelineární regulátor již nelze popsat jako lineárními diferenciálními rovnicemi. Akční zásah je spočítán na základě předepsaných matematických vztahů, regulátor je určen pro konkrétní nelineární systém.

2.5.4 Fuzzy

Fuzzy regulátory jsou též nelineární, tvoří však zcela zvláštní skupinu, neboť mechanismus výpočtu velikosti akčního zásahu se zcela liší od předchozích metod. Jak napovídá název, výpočty jsou prováděny s fuzzy lingvistickými hodnotami, nikoli s ostrými číselnými hodnotami. Na vstupu je ostrá hodnota regulační odchylky převedena na fuzzy (fuzzifikace) a po vyhodnocení „pravidel“ je výsledná hodnota stanovena v procesu „defuzzifikace“.

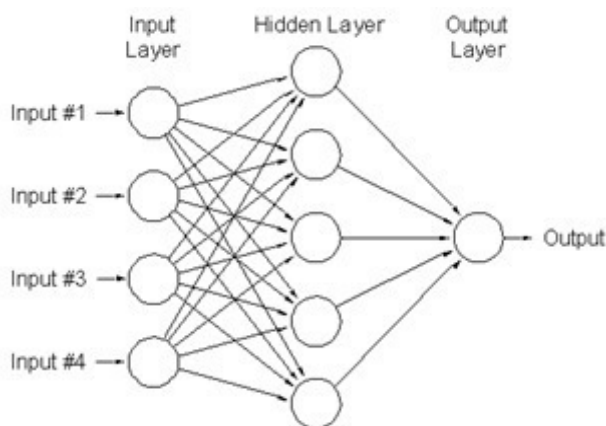


Obr. 14 Fuzzy regulátor [21]

Fuzzy regulátory jsou vhodné i pro nelineární systémy.

2.5.5 Neuronové sítě

Pro úplnost zmiňme ještě možnost využití neuronových sítí. Sít' je schopna se v procesu učení naučit vhodný postup regulace. Nevýhodou neuronové sítě je fakt, že informace je rozptýlena po celé struktuře, a není možné ji zpětně vyčíst a posoudit z analytického pohledu. Ostatně neuronové sítě se používají až tam, kde není možné problém vyřešit klasickými metodami, kde nejsme schopni analyticky stanovit zákonitosti chování řízeného systému.



Obr. 15 Neuronová síť [22]

Prakticky pouze první dvě uvedené skupiny regulátorů lze realizovat analogovými obvody snadným způsobem. V současné době se však setkáme téměř výhradně s řešenými číslicovými. Přesto můžeme mluvit jak o spojitých tak o diskrétních regulátorech. Záleží na poměru vzorkovací periody regulátoru a velikosti časových konstant systému, pokud jsou časové konstanty o mnoho větší (řádově), pak je i číslicový regulátor z pohledu systému spojitý, jestliže je vzorkování vzhledem k časovým konstantám pomalé, je nutno počítat regulátor jako diskrétní.

3 KLASICKÉ METODY NÁVRHU REGULÁTORŮ

Proces vytvoření regulátoru pro řízený systém je otázkou provedení řetězce úkolů:

1. Stanovení přenosu (vlastností) soustavy (řízeného systému)
 - a. Vyjádření na základě fyzikálního popisu
 - b. Identifikace
2. Volba typu regulátoru a technických prostředků
3. Návrh parametrů regulátoru
4. Implementace

3.1 Stanovení přenosu soustavy

Vlastnosti systému můžeme získat z popisu fyzikální realizace systému. Zajisté se jedná (v idealizovaném případě) o nejpřesnější popis soustavy. Není však mnohdy možné postihnout všechny fyzikální vlivy, které se na výsledných vlastnostech systému podílejí a proto je zapotřebí u složitějších systémů odvozený přenos ověřit. Ve složitějších případech není vůbec možné vlastnosti systému stanovit jinak než identifikací, tj. vyšetřováním vlastností soustavy na základě jejího chování.

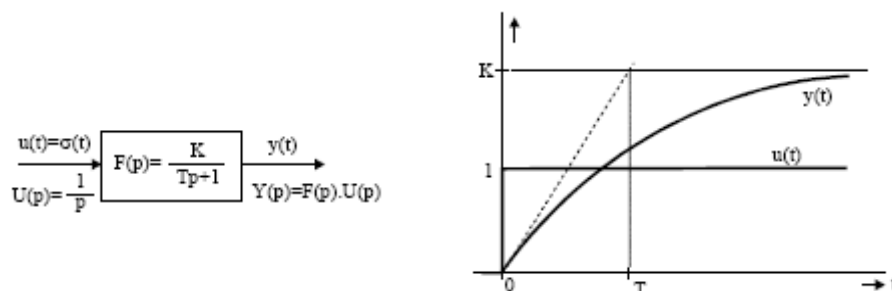
3.2 Identifikace

Proces identifikace je klíčovým pro celý návrh, pokud nebudeme pracovat s modelem, který odpovídá reálné soustavě, stěží dosáhneme kvalitní regulace. Vlastnosti systému vyšetřujeme v časové či frekvenční oblasti, mezi nejpoužívanější metody patří identifikace z:

1. Přechodové charakteristiky
2. Korelačních metod
3. Frekvenční charakteristiky
4. Spektrální analýzy
5. Aplikace metody nejmenších čtverců
6. Aplikace on-line metody nejmenších čtverců

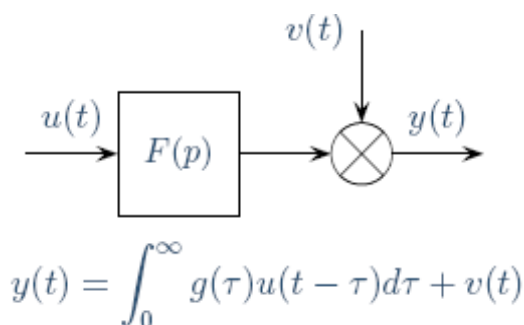
3.2.1 Identifikace z přechodové charakteristiky

Vyšetřujeme odezvu systému na jednotkový skok v časové oblasti. Z přechodové charakteristiky můžeme snadno určit velikost dopravního zpoždění, rovněž určíme, zda se jedná o systém s přenosem prvního či vyššího řádu. V případě 1. řádu jsme schopni časovou konstantu určit velmi přesně, v případě vyšších řádů zpravidla používáme aproximaci charakteristiky přenosem 2. řádu, nebo přenosem n-tého řádu se stejnými časovými konstantami.



Obr. 16 Identifikace systému 1. řádu v časové oblasti [6]

3.2.2 Identifikace pomocí korelačních metod

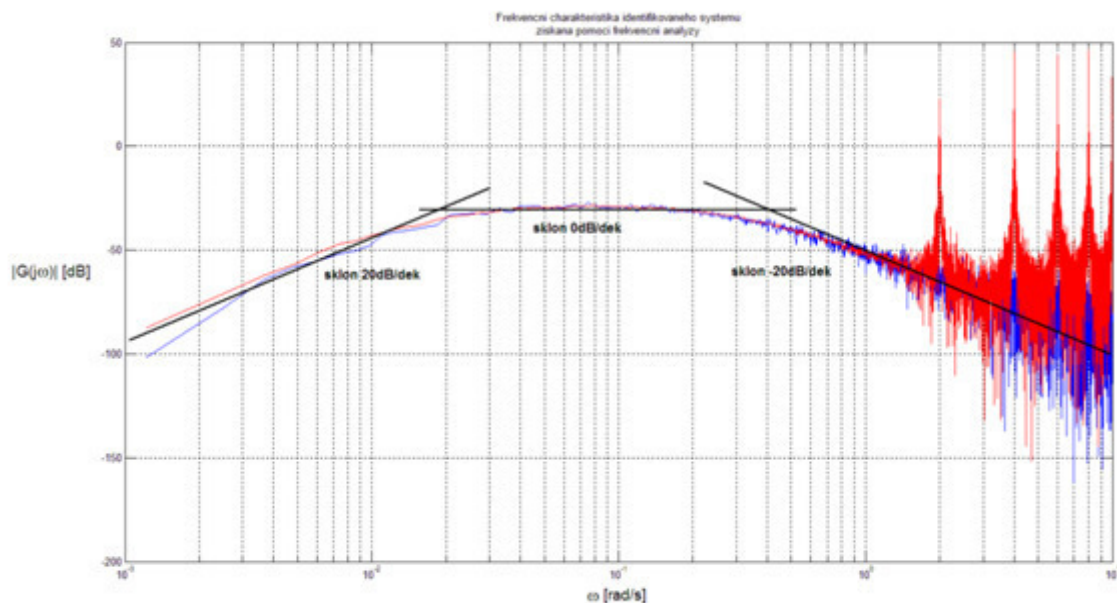


Obr. 17 Výchozí model [23]

Hledáme impulzovou charakteristiku g , jejím Laplaceovým obrazem je pak přenos systému.

3.2.3 Identifikace z frekvenční charakteristiky

Z průběhu amplitudové frekvenční charakteristiky můžeme usuzovat počet a velikost časových konstant, neboť každá časová konstanta působí na frekvenční charakteristice „zlom“. Zlom charakteristiky je ve skutečnosti postupným zakřivením (jako zlom se projeví pouze na aproximačních křivkách ve směru asymptot), jsou-li časové konstanty blízko sebe, může být přesné určení velikostí jednotlivých konstant obtížné.



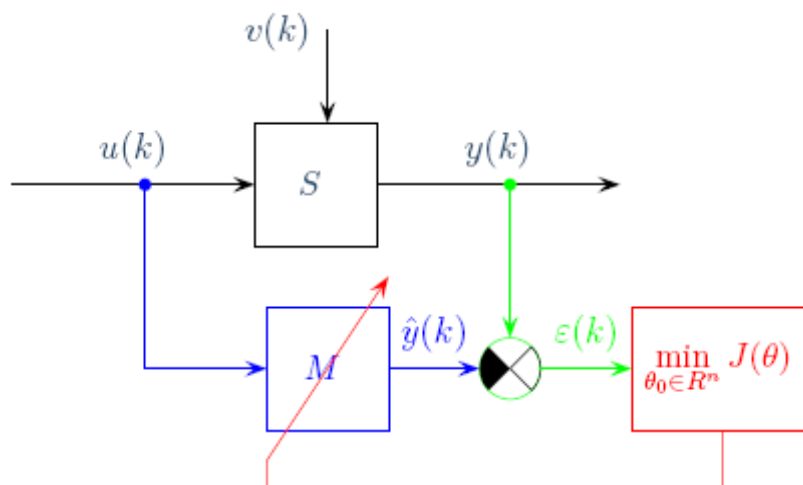
Obr. 18 Amplitudová frekvenční charakteristika

3.2.4 Identifikace pomocí spektrální analýzy

Frekvenční přenos vyjádříme jako podíl Fourierova obrazu korelace vstupního a výstupního signálu ku Fourierovu obrazu autokorelace vstupního signálu.

3.2.5 Identifikace pomocí metody nejmenších čtverců

Metoda hledá konstanty modelu, do něž vstupuje stejný (různorodý) signál jako do reálné soustavy. Výstup z modelu a z reálné soustavy jsou navzájem porovnávány, metoda pak stanoví časové konstanty tak, aby kvadrát odchylek (rozdílů výstupu modelu a soustavy) byl minimální.



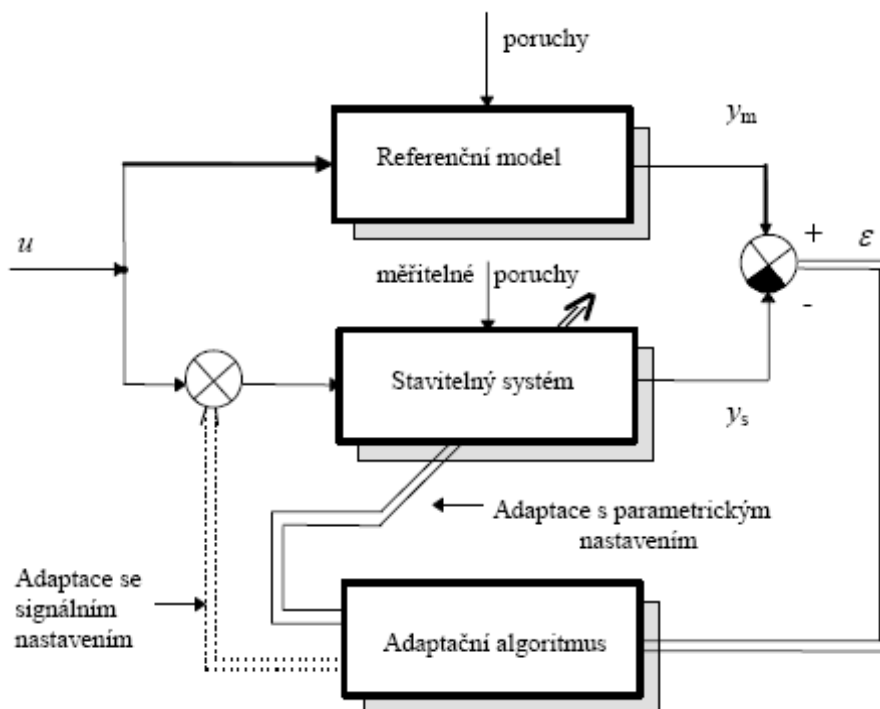
Obr. 24 Principiální schéma metody nejmenších čtverců [24]

Metoda je hojně používaná, neboť nevyžaduje žádná zvláštní měření na soustavě (systému), stačí změřit vstupní a výstupní signál na reálné soustavě. Určení časových konstant uvedenou metodou je poměrně přesné, problém spočívá ve správném určení tvaru modelu (řád, dopravní zpoždění...). Pokročilé metody identifikace využívající metodu nejmenších čtverců automaticky generují modely a vybírají model, s kterým dosahují nejlepších výsledků.

3.2.6 Identifikace pomocí online metody nejmenších čtverců

Existuje modifikace metody nejmenších čtverců, která nevyžaduje statická soubor naměřených dat, ale je schopna pracovat „průběžně“, tedy on-line.

Uvedený postup je vhodný pro monitorování soustavy za provozu. Je tak možno realizovat adaptivní regulátory s referenčním modelem. Dojde-li ke změnám v soustavě, model soustavy je aktualizován a následně jsou modifikovány parametry regulace. Adaptaci lze provést buď přepočítáním parametrů regulátoru (parametrické nastavení) nebo přidáním řídicím signálem (signální nastavení).



Obr. 25 Adaptivní systém s referenčním modelem [11]

3.3 Volba typu regulátoru a technických prostředků

Nepoužíváme-li složité sofistikované regulační algoritmy, které například provádějí adaptaci v závislosti na výsledcích současně prováděné identifikace, a vystačíme-li si s regulátory s víceméně statickými parametry, nebude výběr technických prostředků pro realizaci regulátoru problematický. I na mikrokontrolérech jsme schopni aplikovat například jednoduchý fuzzy-regulátor s lichoběžníkovými (či trojúhelníkovými) funkcemi příslušnosti (gausovské rozložení funkcí příslušností by zřejmě již vyžadovalo větší výpočetní výkon, ale i jednoduché fuzzy-regulátory dosahují kvalitních výsledků).

Otázkou je spíše volba vhodného typu regulátoru. Velmi univerzální a poměrně robustní metodou je použití PID regulátoru, dále jsou zde další regulátory s lineárním přenosem stavěné systému „na míru“. Uvedené varianty jsou vhodné především pro soustavy lineární (alespoň v pracovní oblasti). Pro soustavy s výraznějšími nelinearitami je vhodné přemýšlet o použití fuzzy regulátorů (fuzzy PI+PD). Pro některé speciální aplikace jsou vhodná i méně tradiční řešení (neuronové sítě, atd.).

3.4 Návrh parametrů regulátoru

Nehledě na typ regulátoru (lineární, fuzzy, neuro) je při návrhu regulátoru klíčová otázka požadavků na regulační děj a jejich priorita. V některých aplikacích je naprosto nepřijatelný překmit výstupní veličiny, především v oblasti robotiky, kde řízenou veličinou je například poloha robotického ramene, překmit by mohl znamenat náraz robotického ramene na překážku, což by zřejmě zapříčinilo mechanické poškození. V jiných případech je překmit výstupní veličiny dovozen, ale je požadováno minimální kmitání (tj. požadavek na aperiodický přechodový děj). Dalším hlediskem může být požadavek na rychlost vyregulování výstupní veličiny na žádanou hodnotu. Požadavky na velkou rychlost a minimální překmit jsou ve většině případů protichůdné.

Budeme-li se soustředit na metody návrhu klasických regulátorů, existuje mnoho přístupů, z těch nejznámějších jmenujme tyto metody návrhu regulátoru:

1. Návrh v oblasti frekvenčních charakteristik
2. Optimální modul
3. „Pole placement“
4. Využití algebraické (polynomiální) teorie řízení
5. Integrální kritéria
 - a. Kvadratické
 - b. Lineární
 - c. ITAE

3.4.1 Návrh ve frekvenční oblasti

Tvarování frekvenční charakteristiky je historicky nejstarší přístup k návrhu parametrů regulátoru. Pomocí časových konstant a zesílení regulátoru je ve frekvenční oblasti modifikována frekvenční charakteristika původního systému. Výsledný tvar charakteristiky reprezentuje vlastnosti otevřeného obvodu (tj. sériového zapojení regulátoru a původního systému). Cílem je aby v oblasti tzv. „omega řezu“ (asi po jednu dekádu) protínala amplitudová frekvenční charakteristika osu rostoucích frekvencí (tj. hodnotu 0 dB) pod sklonem -20dB/dekádu. Mimo tuto oblast je vhodné, je-li sklon charakteristiky strmější. Ve výsledné fázi má pak uzavřený obvod až do frekvence „omega řezu“ přenos roven jednotkové velikosti, vyšší frekvence jsou pak tlumeny.

3.4.2 Optimální modul

Metoda vychází z tvaru frekvenčního přenosu soustavy, na základě koeficientů přenosu jsou dle vztahů vypočítány parametry regulátoru tak, že výsledný přenos uzavřeného obvodu nevykazuje žádnou rezonanci, tj. průběh frekvenční charakteristiky je monotónní a přechodový děj v časové oblasti je bez překmitu. Metoda nalezne nastavení, kterým docílíme nejrychlejšího nekmitavého přechodového děje.

3.4.3 „Pole placement“

Návrh probíhá v rovině „P“ (rovině Laplaceovy transformace) případně v rovině „Z“ (rovině Z-transformace). Uživatel může doplňovat do roviny póly a nuly, rozmístění kořenů koresponduje s otevřeným obvodem, lze však na základě známých vlastností systémů usuzovat na chování zpětnovazebního obvodu. Je nutno kontrolovat výsledný návrh (ověřit např. simulací), do našeho zorného pole by se měl dostat především akční zásah, resp. jeho velikost. Při rozmístění nových pólů se nám může podařit systém výrazně „zrychlit“, musíme si však vždy uvědomit, že např. snížení vlivu dvou časových konstant původního systému o řád s sebou nese požadavky na stonásobně vyšší akční zásahy, a to ne každý reálný systém dovolí.

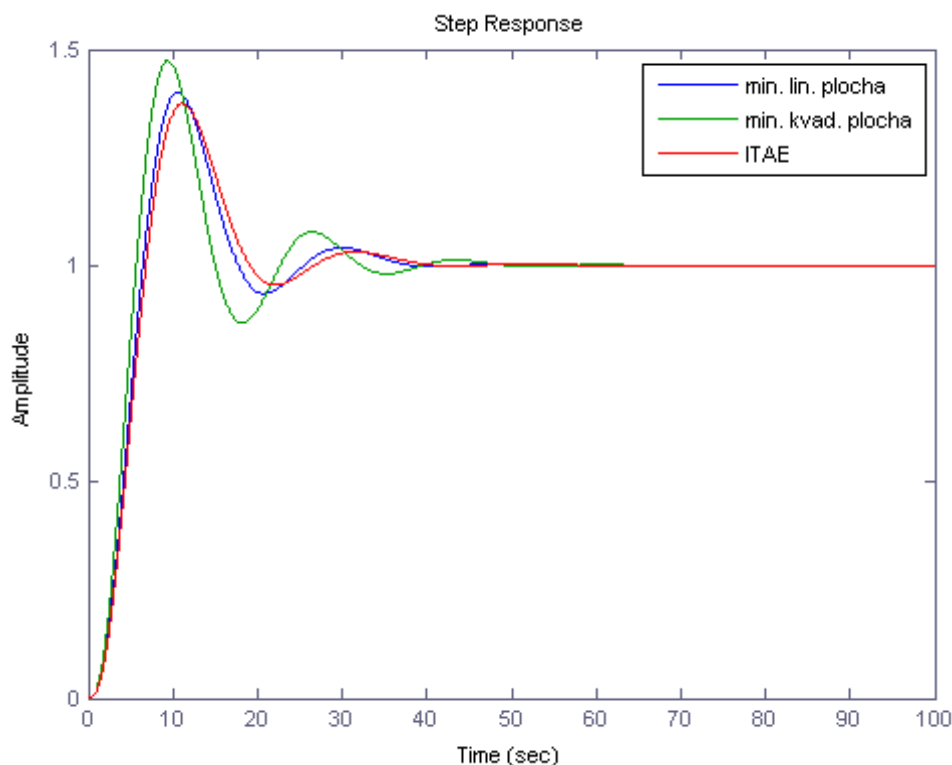
3.4.4 Algebraická polynomiální teorie

Dalším přístupem je využití polynomiální teorie v řízení. Na rozdíl od předchozích metod, kde návrh probíhá většinou ve spojitě oblasti, je pro uvedenou metodu běžné provádět návrh v diskrétní oblasti (Z-transformace). Je možno realizovat nejen regulátory pro časově optimální konečný přechodový děj či robustní regulátory, ale teorie nabízí rovněž vyšetřování a tvarování citlivostní funkce přenosu (tj. vyšetřovat zároveň přenos poruchy). S využitím jednoho regulátoru ovšem nejsme schopni zajistit „ideální řízení“, vždy musíme pamatovat, že požadavky na řízení a požadavky na přenos poruchy jsou již ze své podstaty protichůdné (jelikož přenos řízení a přenos poruchy jsou spolu svázány, součet přenosu řízení a přenosu poruchy působící na výstupu systému je vždy roven „1“).

3.4.5 Integrální kritéria

Integrální kritéria vyhodnocují kvalitu regulačního děje v časové oblasti. Při návrhu je minimalizována integrální plocha regulační odchylky. Metod je několik, výsledky se často liší jen málo, každá z metod klade důraz na jiné ohledy.

Jak ukazuje obrázek, rozdíly jsou malé.



Obr. 26 Soustava 3. řádu s P-regulátorem nastaveným dle integrálních kritérií

3.4.6 Kvadratické integrální kritérium

Metoda minimalizuje kvadratickou plochu pod křivkou regulační odchylky v časové oblasti a lze ji řešit relativně snadno analyticky. Přechodový děj je v porovnání s ostatními metodami nejrychlejší, zato vykazuje největší překymity. Vztah pro výpočet plochy [8] je uveden níže:

$$J_K = \int_0^{\infty} e^2(t) dt$$

3.4.7 Lineární (usměrněné) integrální kritérium

Lineární integrální kritérium minimalizuje (zpravidla „usměrněnou“) lineární plochu pod křivkou regulační odchylky v časové oblasti. V porovnání s kvadratickým kritériem je regulace pomalejší, zato vykazuje menší překmity.

$$J_{UL} = \int_0^{\infty} |e(t) - e(\infty)| dt$$

Vztah pro výpočet lineární plochy [8]

3.4.8 Integrální kritérium ITAE

Důraz je kladen na vyregulování v krátkém čase, hodnotám odchylky je v průběhu času přiřazována stále vyšší váha. Výsledky jsou často podobné jako v případě kritéria lineární plochy.

$$J_{ITAE} = \int_0^{\infty} |e(t) - e(\infty)| t dt$$

Vztah pro výpočet hodnoty kritéria ITAE [8]

Pro modelový příklad byly navrženy regulátory uvedenými způsoby. Na obrázcích je možno porovnat odezvy na skok řízení a skok poruchy.

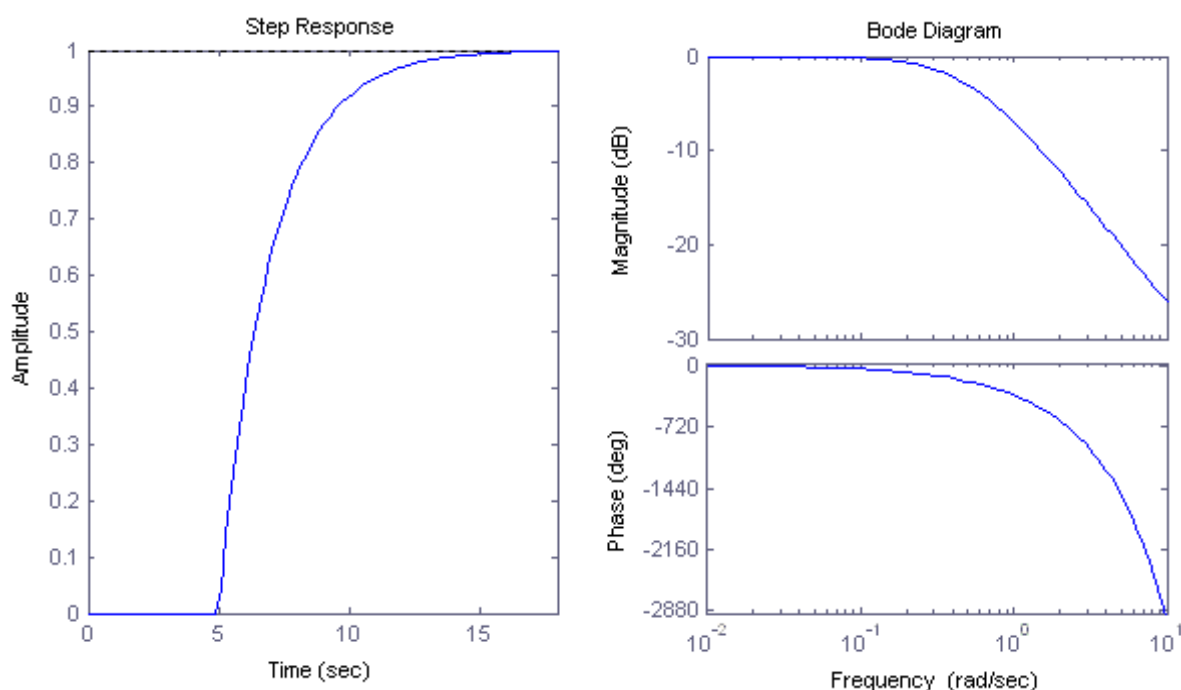
4 DOPRAVNÍ ZPOŽDĚNÍ A ZÁKLADNÍ NELINEARITY

Návrh regulátorů klasickými metodami uvedenými v předchozí kapitole naráží při výskytu dopravního zpoždění v obvodu na problémy s návrhem i robustností výsledného řešení, výskyt nelinearit činí mnohé metody částečně nebo častěji i zcela nepoužitelnými. Jak se s uvedenými nepříznivými vlivy vyrovnáme?

4.1 Dopravní zpoždění

Dopravní zpoždění zdaleka není neobvyklým fenoménem, již podle názvu lze usoudit, že se bude vyskytovat v systémech, kde dochází k fyzické přepravě materiálu. Příkladem může být tepelná elektrárna, kde je uhelná drť přepravována od drtičky ke kotli (permanentně se pohybujícím) pásovým dopravníkem. Další ilustrací je klasická hadice. Napouštíme-li vodu hadicí, pozorujeme časový interval (zpoždění) od chvíle, kdy jsme pustili vodu, do okamžiku, kdy začne voda z hadice vytékat. Čím je hadice delší, tím je efekt markantnější (snadno je jev pozorovatelný i u sprchy). Nutno však poznamenat, že dopravní zpoždění nemusí vždy souviset jenom s přepravou hmotného média, může být způsobeno různými mechanismy v elektronických obvodech (A/D převodníky, atd.).

V časové oblasti si dopravní zpoždění představíme snadno, jedná se o časový úsek mezi změnou průběhu signálu vstupního a signálu výstupního. Jak se projeví dopravní zpoždění ve frekvenčních charakteristikách? Amplitudová frekvenční charakteristika zůstává beze změny, modul dopravního zpoždění je vždy roven jedné. Vliv dopravního zpoždění je patrný ve fázové frekvenční charakteristice, velikost fáze se rostoucí frekvencí stále roste.



Obr. 27 Systém 1. řádu s dopravním zpožděním (časová a frekvenční oblast)

Vlivem dopravního zpoždění se může stát kmitavým, dokonce nestabilním, i systém prvního řádu. Ilustrujme to na příkladu člověka ve sprše, který má pro regulaci teploty k dispozici dva kohoutky, sice na teplou a studenou vodu. Je-li hadice vedoucí k sprchové růžici krátká, regulace bude poměrně snadná, pokud by měla hadice délku několika desítek metrů, regulace by byla již značně obtížná (zvláště za předpokladu, že by se například měnil tlak v potrubí, a poměr mísení by se samovolně měnil).

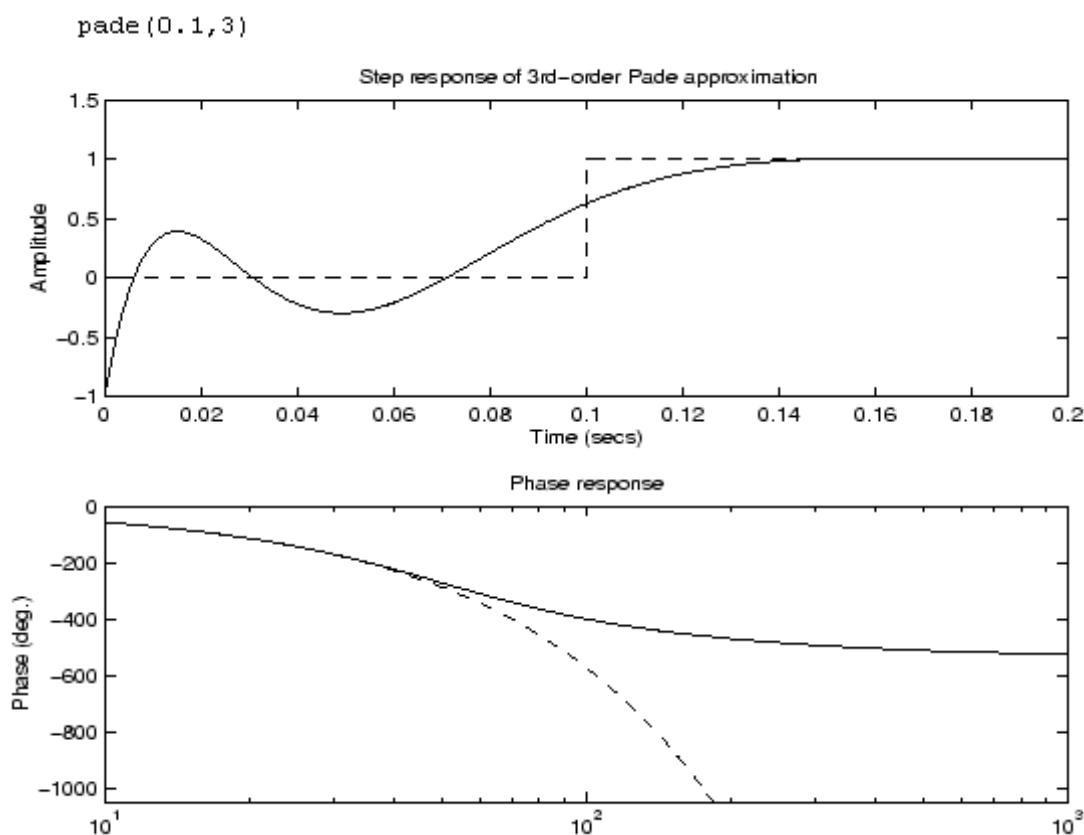
V základním tvaru je operátorový přenos dopravního zpoždění [8]:

$$F(p) = e^{-dp}$$

V uvedeném tvaru nelze s přenosem pracovat (v rámci návrhu regulátorů klasickými metodami), protože se nejedná o poměr dvou polynomů.

4.2 Aproximace Padeho rozvojem

Uvedený výraz (operátorový přenos dopravního zpoždění) lze na poměr dvou polynomů „převést“, resp. provést aproximaci. Nejčastěji je k těmto účelům využit Padeho rozvoj. Polynom čitatele a jmenovatele jsou stejného stupně a řádu, koeficienty polynomů se liší pouze ve znaménkách. Dle požadavků na přesnost lze volit řád polynomů aproximace.



Obr. 28 Aproximace dopravního zpoždění padeho aproximací polynomem 3. řádu
[MatLab 7.8.0 – Help: „pade“]

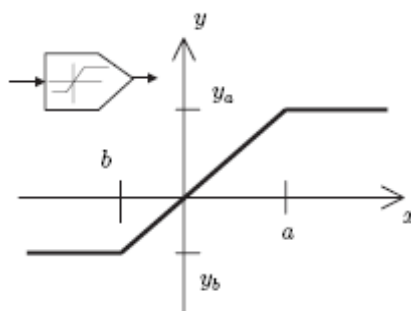
Ve chvíli, kdy získáme tvar dopravního zpoždění jako poměr dvou polynomů, můžeme využít všech výše popsaných klasických metod návrhu regulátorů.

4.3 Základní nelinearity

Zatímco s dopravním zpožděním jsme schopni se relativně elegantním způsobem vypořádat, v případě základních (a velmi častých) nelinearit tomu tak již není. Téměř u všech reálných soustav se setkáme s omezeními linearity, některé systémy jsou na celém svém pracovním rozsahu lineární, některé narážejí například na saturaci, z globálního pohledu však má limity většina soustav.

4.3.1 Saturace

Nejznámější a nejrozšířenější nelinearitou je saturace, neboli nasycení.

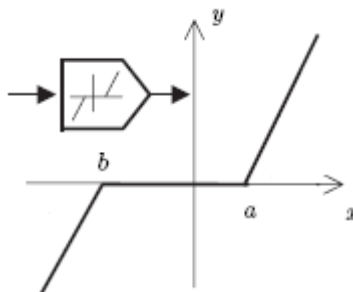


Obr. 29 Nelinearita nasycení [9]

Uvažujme, že akční zásah nikdy nemůže být neomezený (a to ani v amplitudě ani v čase), saturace je omezením v amplitudě. Za předpokladu že se v rámci pracovního rozsahu pohybujeme daleko od hranice nasycení, nemusí nás uvedená skutečnost znepokojovat, v opačném případě je však nutná značná obezřetnost. Pokud bychom totiž navrhli regulátor, který počítá s akčními zásahy přesahujícími mez nasycení, musíme počítat s tím, že regulační děj nebude probíhat podle požadavků, na které byl navržen, v některých případech může být regulátor zcela nefunkční. Je potom potřeba provést návrh předpokládající menší akční zásahy, nebo využít jiné metody návrhu.

4.3.2 Pásmo necitlivosti

Pásmo necitlivosti patří také mezi rozšířené základní nelinearity.

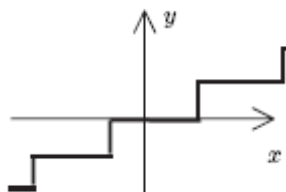


Obr. 30 Nelinearita necitlivosti [9]

Na průběhu existuje tzv. „mrtvá zóna“, kde je nulová citlivost.

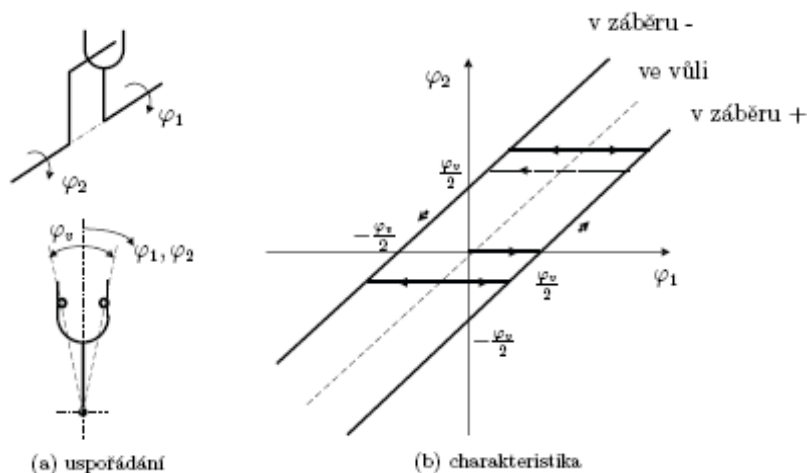
4.3.3 Kvantování v hodnotě

Jedná se o podobný problém jako u pásma necitlivosti, patrný je především u D/A převodníků s malým rozlišením, hodnoty se nemohou měnit spojitě.



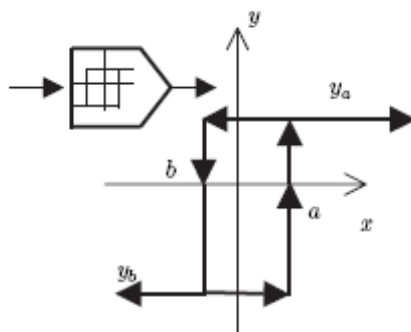
Obr. 31 Nelinearita nespojitosti

4.3.4 Hystereze



Obr. 32 Vůle v převodech [9]

Hystereze je také poměrně častou nelinearitou, setkáme se s ní v mechanických soustavách (např. vůle v převodech), rovněž v elektrotechnice.



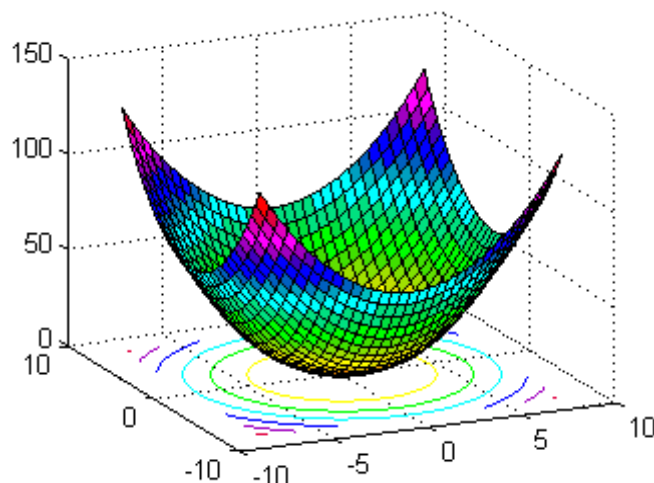
Obr. 33 Relé s hysterezí [9]

4.4 Možnost využití klasických metod

Je nyní otázkou, zde nelze v některých případech najít i pro systém s některou ze základních nelinearit vhodný regulátor ve skupině lineárních. Jelikož již většinou nejsou použitelné klasické metody návrhu, budeme tvar regulátoru hledat jinak. V úvahu přichází pokročilé metody v optimalizaci. Pokud regulátor nalezneme, je nutno dále stanovit, jaké bude mít regulace omezení (jelikož celkový systém není lineární).

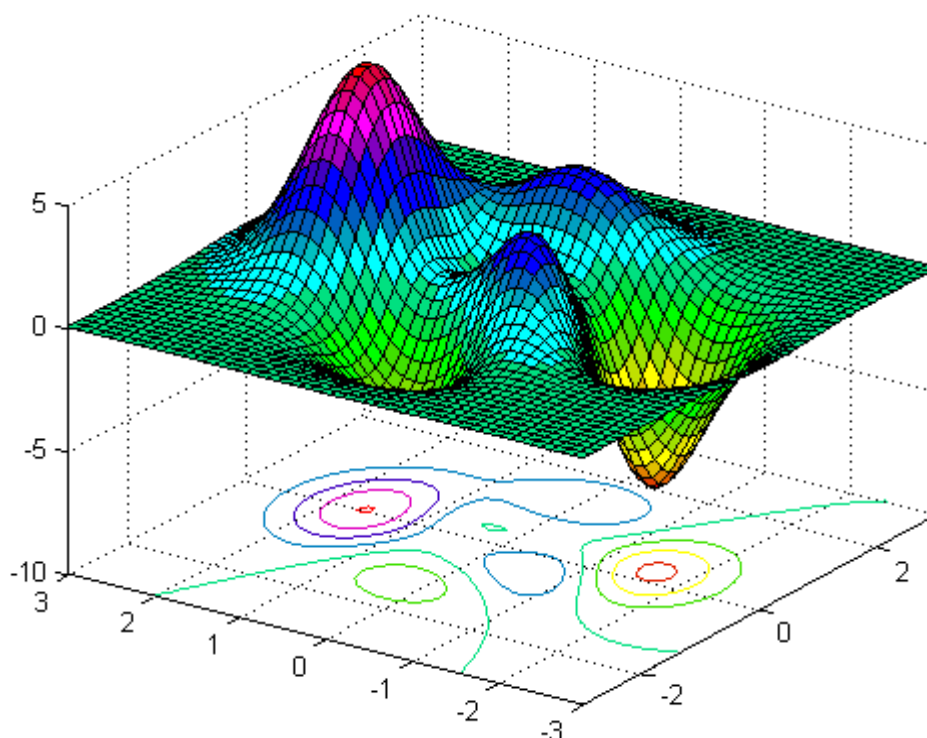
5 EVOLUČNÍ TECHNIKY V OPTIMALIZACI

Hledání optimálního řešení představuje zpravidla hledání globálního extrému účelové funkce. Pokud má funkce jediný extrém, řešení najdeme analyticky či numericky poměrně rychle.



Obr. 35 Ilustrační obrázek funkce dvou proměnných (paraboloid)

Pro velké množství problémů se však na průběhu vyskytuje množství lokálních extrémů, v některých případech i velké množství. Funkce mohou být nespojité, na částech intervalu nedefinované.



Obr. 36 Ilustrační obrázek funkce dvou proměnných [MatLab 7.8.0 - Help: „surf“]

Analytický popis funkcí navíc mnohdy nemáme k dispozici. Nezbyvá tedy než prohledávat

prostor řešení. Uvážíme-li však, že s každým dalším parametrem roste dimenze prostoru řešení, a že i pro jednodimenzionální problém, může být řešení nespočetně mnoho, je zřejmé, že klasické („enumerativní“) metody prohledávání celého prostoru jsou pro nás nepoužitelné, jelikož prohledání prostoru není uskutečnitelné v reálném (někdy ani konečném) čase.

V úvahu proto přicházejí metody, které budou hledat optimum, aniž by procházely všechna řešení.

5.1 Pojmy

Základní pojmy používané v evolučních technikách optimalizace jsou ustálené, nebude však na škodu, když je zde uvedeme.

5.1.1 Účelová funkce

Účelová funkce hodnotí kvalitu („vhodnost“) jednotlivých řešení. Optimální řešení úlohy je pak zpravidla takové, kde hodnota účelové funkce nabývá extrému. Úmyslně používám pojem „extrém“, a nikoli výhradně minimum či maximum, pro úlohy, kdy se snažíme maximalizovat zisk, je hledaným extrémem myšleno maximum, v úloze kde se snažíme o minimalizaci (nákladů, energie, atd.) je hledaným extrémem minimum. Hledání maxima se však snadno převede na hledání minima a naopak, stačí pouze vynásobit účelovou funkci zápornou jednotkou (většinou „-1“).

5.1.2 Jedinec

Pojem *jedinec* znamená konkrétní kombinaci optimalizovaných parametrů.

5.1.3 Fitness

Fitness nebo též *vhodnost* uvádí kvalitu jedince. Fitness je právě hodnota účelové funkce pro konkrétního jedince, tj. pro zvolenou kombinaci parametrů (souřadnice).

5.1.4 Populace

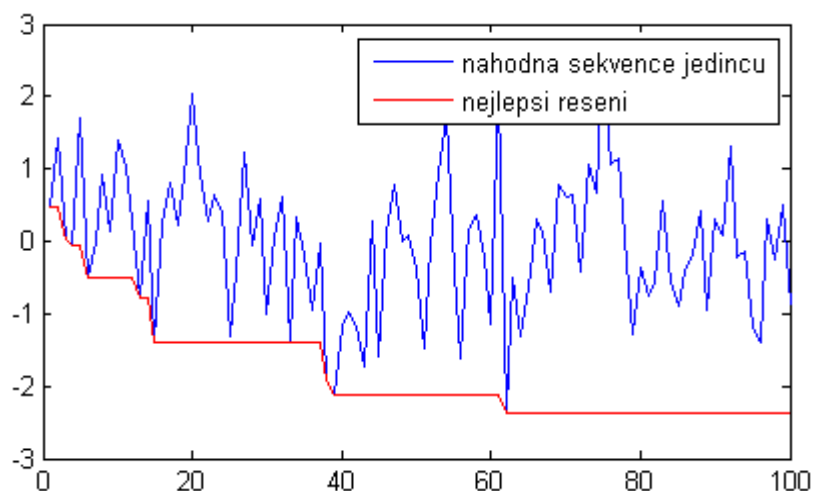
Množina jedinců, se kterými pracujeme.

5.2 Vybrané metody evolučních technik

V následujících podkapitolách budou velmi stručně charakterizovány některé metody evolučních technik, další metody a bližší informace o uvedených metodách jsou k dispozici v [3].

5.2.1 Metoda náhodné procházky

Metoda náhodné procházky nebo též „Random Walk“ [3] vybírá jedince zcela náhodně (jak je patrné již z názvu metody). Metoda uchovává v paměti nejlepšího jedince, dokud nenalezne vhodnějšího, viz obrázek níže.

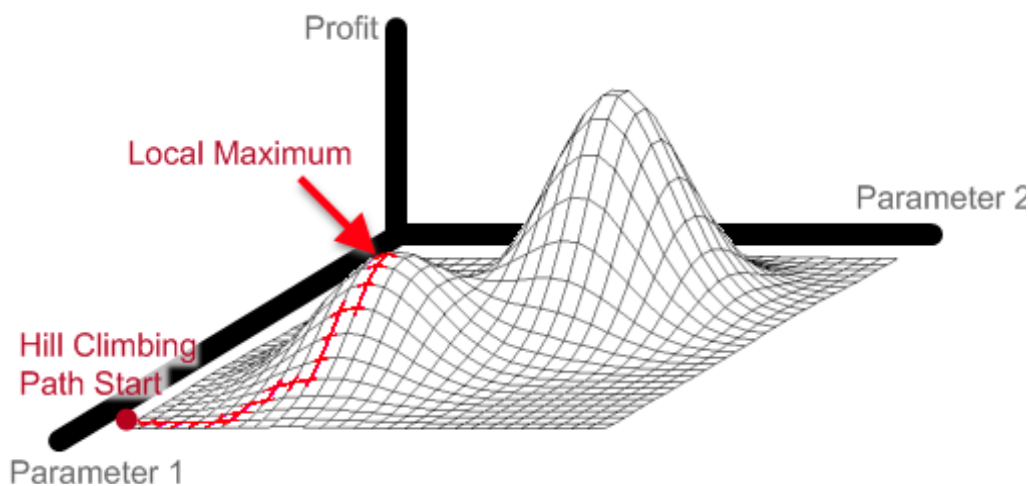


Obr. 37 Prohledávání metodou náhodné procházky

5.2.2 Horolezecký algoritmus

Metoda náhodné procházky je málo efektivní. „Prvním z nápadů bylo zkoumat hodnoty ve směru největšího gradientu.“ [3] Metoda prohledává okolí, za střed tohoto okolí je bráno vždy nejlepší nalezené řešení. Algoritmus však může snadno uváznout v lokálním extrému, pokud oblast „okolí“ nezahrnuje lepší řešení, než je lokální extrém, nebo může dojít k zacyklení.

Mezi modifikace patří horolezecký algoritmus s učením, stochastický horolezecký algoritmus či paralelní horolezecký algoritmus, viz [3]

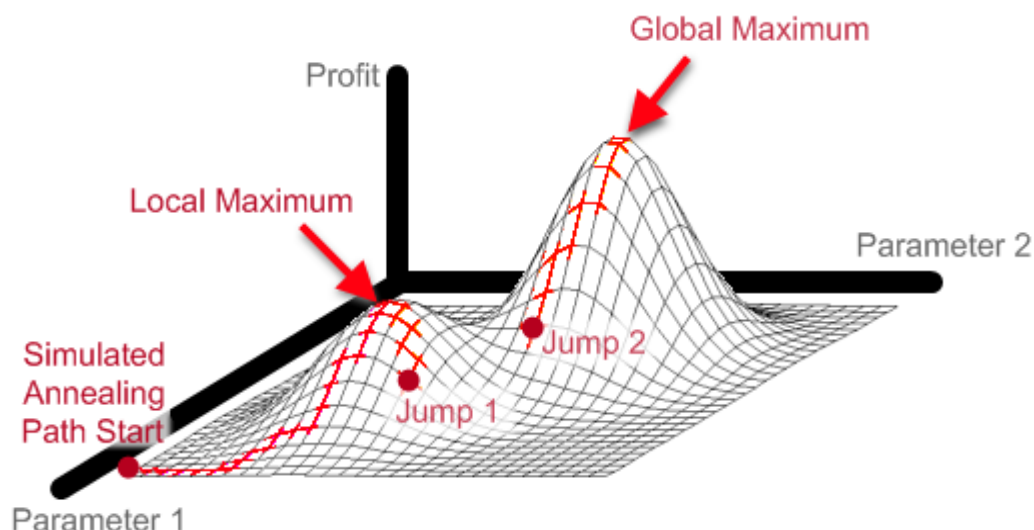


Obr. 38 Horolezecký algoritmus [25]

5.2.3 Simulované žíhání

Algoritmus je inspirován procesem žíhání kovů. Žíhání zlepšuje vlastnosti kovů, neboť dochází ke „stabilizaci volných částic k optimálnímu stavu“ [3], tj. je „opravována“ krystalická mřížka materiálu. Proces připouští přechod i ke stavu s horší energetickou úrovní, s klesající teplotou však pravděpodobnost takového přechodu klesá.

Stejně i numerická metoda připouští přechod k horším řešením s pravděpodobností, která se v průběhu času snižuje. Uvedeným mechanismem snižuje riziko uváznutí v lokálním extrému.



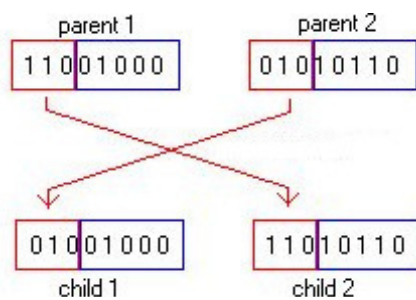
Obr. 39 Simulované žhání [25]

5.2.4 Zakázané prohledávání („Tabu search“)

Principiálně funguje metoda jako horolezecký algoritmus, je však doplněna pamětí typu fronta, v níž jsou udržovány naposledy navštívené pozice. Mechanismus výrazně potlačuje nebezpečí zacyklení (ve srovnání s horolezeckým algoritmem), nicméně nemůže je zcela vyloučit (pokud je cyklus delší než fronta).

5.2.5 Genetické algoritmy

Jsou inspirovány zákony, které se uplatňují v genetice. Z genetiky je přebrána i část názvosloví. Jedinec je definován kombinací parametrů, na kterou se pohlíží jako na genetickou informaci – řetězec DNA. Noví jedinci vznikají křížením původních (rodičovských) jedinců a mutacemi.



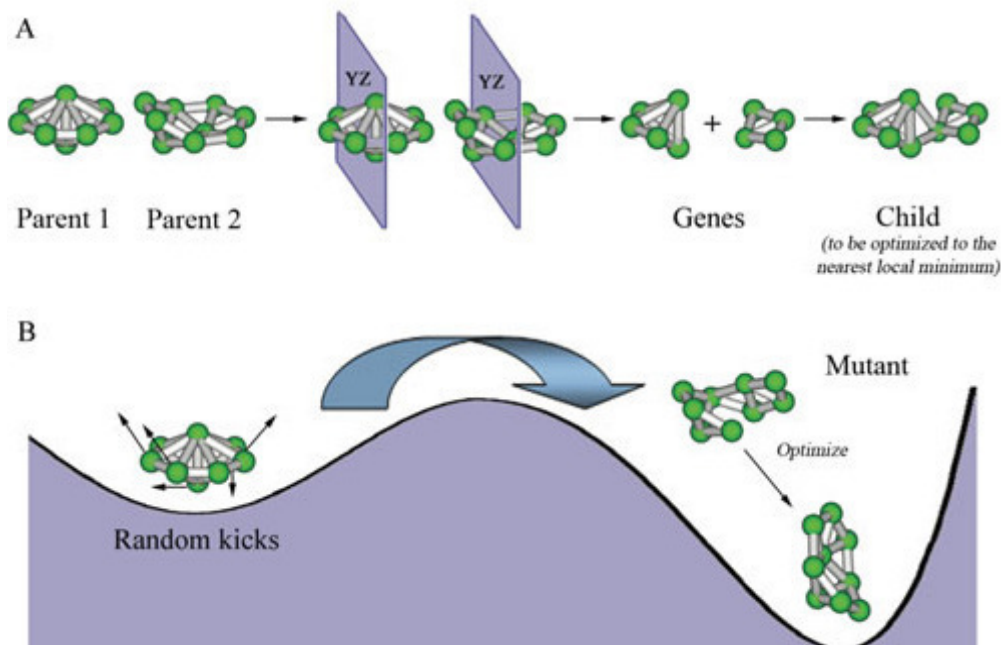
Obr. 40 Křížení [26]

Stejně jako v biologii bakterií ale i vyšších organismů, mají lepší jedinci větší šanci na přežití, tak i uvedená metoda vybírá jedince s nejlepším fitness a tyto pak přednostně vybírá za rodičovské pro další generaci, respektive jedinci s lepším fitness mají větší šanci, že se na procesu generování nové generace budou podílet.

Na rozdíl od výše uvedených metod, které jsou používány spíše pro menší počet optimalizovaných parametrů, jsou genetické algoritmy vhodné i pro úlohy s desítkami optimalizovaných parametrů.

5.2.6 Rojení částic

„*Algoritmus simuluje chování ptačího hejna*“ [3]. Celé hejno (velké množství částic) se pohybuje prostorem, aktuální směr pohybu částice je orientován pozicí nejlepšího momentálně nalezené řešení.

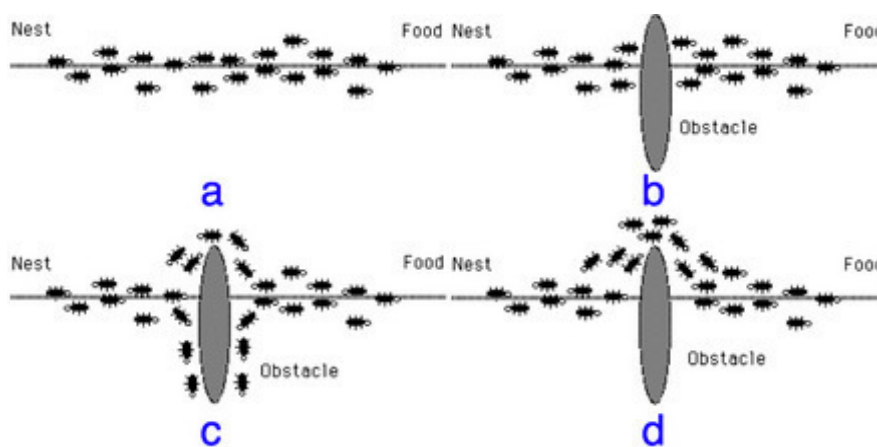


Obr. 41 Optimalizace metodou rojení částic [27]

5.2.7 Optimalizace mravenčí kolonií

Algoritmus je inspirován životem mravenčích kolonií. Mravenci se náhodně pohybují prostorem, pokud mravenec nalezne potravu (vhodné řešení), vrací se do mraveniště a zanechává za sebou pachovou stopu. Pokud na stopu narazí další mravenec, začne také vypouštět feromon. Feromon v průběhu času vyprchá, omezená doba trvání pachové stopy zabraňuje uvíznutí v lokálním extrému.

Metoda je stejně jako rojení částic vhodná i pro velké množství optimalizovaných parametrů.



Obr. 42 Optimalizace cesty za potravou, kterou mravenci provádí (prakticky) [28]

6 VYUŽITÍ EVOLUČNÍCH TECHNIK PŘI NÁVRHU REGULÁTORŮ

Návrh regulátoru je bezesporu optimalizační úlohou. Hledanými parametry jsou tvar a koeficienty (operátorového) přenosu regulátoru. I pro oblast řízení a regulace lze uvažovat o možnostech použití evolučních technik pro design regulačních členů. V současné době není uvedený přístup ještě příliš rozšířený, ale v průběhu času se pravděpodobně dostane do popředí. Pro úlohy, kde si vystačíme s klasickými metodami, nemá smysl hledat jiné řešení, protože klasické metody vedou na exaktní popis (a určí jednoznačné řešení). Na druhou stranu jsou možnosti klasických metod omezené.

Při používání klasických metod designu regulátorů vyvstávají problémy nejen tehdy, vyskytne-li se v systému dopravní zpoždění a nelinearity, ale značně problematická je i kombinace jednotlivých požadavků, neboť jednotlivé metody zpravidla uvažují pouze optimalizaci některých z vlastností výsledného regulačního děje. Ilustračním příkladem může být požadavek pro konkrétní aplikaci, například: maximální dovolený přechodového děje 20%, délka požadavek na minimální kvadratickou plochu. Výpočet regulátoru na uvedené požadavky nelze provést klasickými metodami ani pro lineární systém, alespoň ne bez složitých modifikací kritérií.

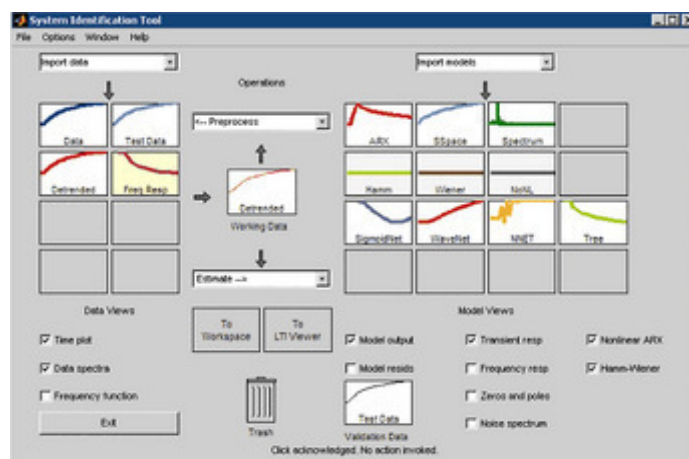
Proč tedy nevyužít evoluční techniky, které požadované nastavení parametrů sice nevyjádří exaktně, ale naleznou je sofistikovaným prohledáváním prostoru možných řešení. Zmíněná myšlenka je zároveň nosnou myšlenkou této práce. Cílem práce je realizace programu provádějícího návrh parametrů regulátorů pro systémy, v nichž se mohou vyskytovat nelinearity, navíc je třeba, aby bylo možné požadavky kombinovat.

6.1 Vymezení rozsahu práce

Představíme-li si řetězec návrhu systému řízení pro řízený systém, můžeme jej rozdělit do dvou základních částí:

1. Identifikace
2. Design regulátoru

Prvnímu bodu je věnována pasáž v teoretickém úvodu, v praktické části se však uvedenou problematikou již zabývat nebudeme, neboť práce je primárně zaměřena na návrhu regulátorů. Oblast identifikace je i v ohledu praktické realizace dobře propracována, uvedme například nástroj prostředí MatLab s názvem „Identification toolbox“, který zahrnuje rozsáhlé možnosti identifikace (na základě nejrozličnějších dat získaných o vlastnostech systému).



Obr. 43 Prostředí Matlab – Identification Toolbox

Druhým celkem je návrh regulátoru a jeho verifikace. Pro návrh regulátoru má být využito evolučních optimalizačních technik. Verifikace programu bude provedena sledováním odezev systému (základní metoda ověření). Verifikace v širším slova smyslu je sama o sobě velmi rozsáhlou oblastí, která přesahuje rozsah této práce, bylo by možné provádět citlivostní analýzu na změny parametrů systému – v reálném systému může docházet ke kolísání některých parametrů, odolnost řídicího algoritmu vůči takovým změnám pak nazýváme robustností.

Pozn.: Evolučními metodami optimalizace budeme hledat parametry regulátorů PID a PSD, které jsou samy o sobě (svou stavbou) poměrně velmi robustní.

6.2 Požadavky na program

Jestliže produktem práce má být program realizující návrh regulátorů, je třeba specifikovat požadavky na aplikaci. Vyjmenujme proto alespoň základní, než se dostaneme ke konkrétní implementaci.

6.2.1 Specifikace soustavy a nelinearit

První požadovanou vlastností programu je vhodný způsob zadávání popisu vlastností systému. Dle zadání práce se jedná o soustavu druhého řádu s dopravním zpožděním. Uvedené požadavky byly ve výsledné implementaci rozšířeny, takže je možnou pracovat se systémem tvořeným soustavou obecného řádu. Kromě dopravního zpoždění je přidána možnost doplnění soustavy o základní nelinearity. Popis parametrů nelinearit je do značné míry intuitivní vzhledem k jejich povaze a budeme se mu blíže věnovat v popisu implementace. Vlastnosti lineární části soustavy (systému) budou zadávány v tvaru přenosu nebo pólů, nul a zesílení systému (ostatní formy popisu jsou na uvedené dvě formy snadno převoditelné).

6.2.2 Nastavení požadavků na regulační děj (kritériální funkce)

Dále je třeba určit požadavky na přechodový děj. Základními hledisky bude rychlost přechodového děje, překmit a charakter (kmitavost) akčního zásahu.

Prioritně posuzovaným bude maximální překmit při přechodovém ději. Program umožní nastavit velikost maximální přípustné hodnoty překmitu.

Pro posuzování rychlosti regulace budou použita integrální kritéria, se kterými jsme se stručně seznámili v teoretickém úvodu.

Třetím podstatným kritériem je charakter akčního zásahu. Jednak není vhodné, aby velikost akčních zásahů v hodnotě převyšovala žádanou hodnotu více než o jeden řád (není zřejmě dobré, ani dobře realizovatelné, abychom regulovali výstupní napětí v řádové úrovni jednotek Voltů akčními zásahy v jednotkách Kilovoltů). V našem případě budeme ale více sledovat kmitavost akčního zásahu, elektronické systémy často snesou i řízení akčním zásahem velmi kmitavého charakteru, ale pro mechanické systémy (např. servomechanismus) je uvedená skutečnost zcela nepřijatelná.

6.2.3 Vizualizace procesu optimalizace

Numerické metody samy o sobě pracují s kritériální funkcí a vracejí optimální nastavení. Primárním úkolem je nalézt optimální řešení, pro uživatele přítomného za výpočetním zařízením však může být zajímavé sledovat průběh optimalizace, vytváří to lepší představu o funkci optimalizačních metod. Je tedy na místě, doplnit systém návrhu o vizualizaci procesu prohledávání.

6.2.4 Uživatelsky přívětivé prostředí

V neposlední řadě je zapotřebí, aby aplikace nabízela uživateli přívětivý interface. Snahou je vytvořit přehledné prostředí, kde by se uživatel snadno orientoval a pokud možno intuitivně pojal prostředky ovládání programu. Program umožní uživateli snadným způsobem zadat parametry soustavy, kritéria optimalizace (tj. požadavky na řízení), nastavit parametry optimalizace a v neposlední řadě sledovat průběh a výsledky procesu optimalizace.

6.3 Volba vývojového prostředí

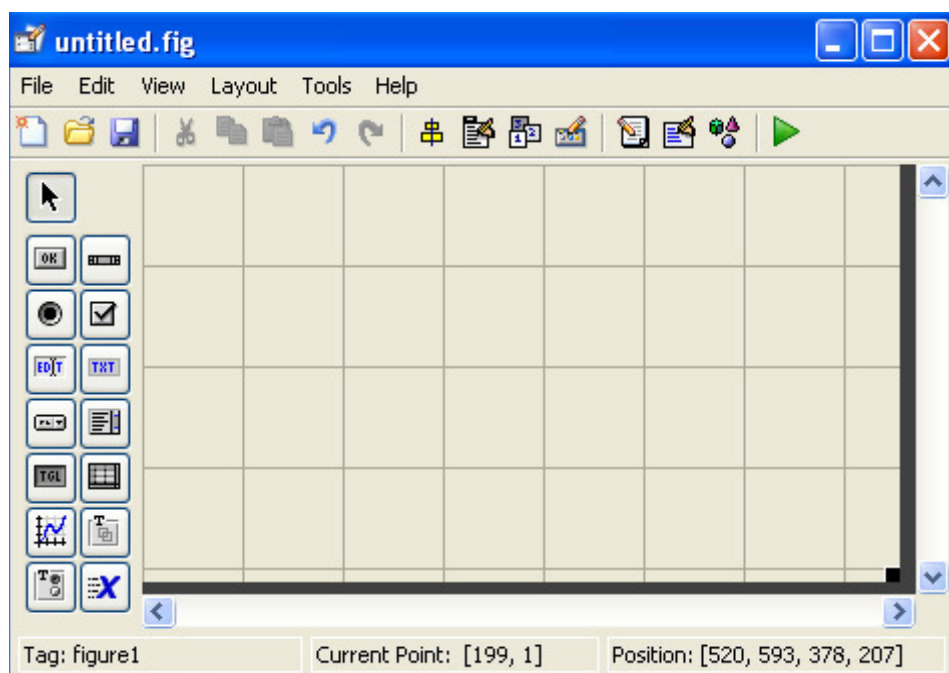
Pro vývoj programu bylo použito matematické prostředí MatLab. Za velikou výhodu lze považovat skutečnost, že MatLab obsahuje simulační prostředí Simulink, kde lze simulovat spojitě i diskrétní systémy, dále MatLab pracuje s vektory a maticemi a pro maticové výpočty používá optimalizované funkce, takže výpočty jsou prováděny velmi efektivně (rychle).

MatLab umí spolupracovat s dalšími programovacími jazyky, například kompiluje programy napsané v jazyce C (v projektu je jedna z částí realizována právě v jazyku C, sice PSD regulátor). Základní programy se píšou do tzv. M-file souborů ve formě skriptů, syntaxe je obdobná jazykům pascalovského typu.

Skripty se spouštějí přímo z prostředí MatLab, existuje však možnost vytvořit samostatně spustitelný program, ten je však nutno doplnit o speciální soubor, obsahující podporu funkcí využívaných aplikací na počítačích kde MatLab není nainstalován.

6.3.1 GUIDE

Pro tvorbu grafické aplikace je využita nadstavba MatLabu „GUIDE“, kde lze vytvářet „klasické“ „oknové“ aplikace. Prostor je relativně intuitivní, uživatel vkládá do grafického rozhraní objekty (tlačítka, textová pole, grafy...), nastavuje jejich vlastnosti a popis jejich funkce dopisuje do souboru M-file, který je při návrhu grafického prostředí automaticky generován a aktualizován.



Obr. 44 Prostředí Matlab – GUIDE

6.3.2 Genetic toolbox

Prostředí MatLab nabízí vedle běžně používaných funkcí možnost rozšíření o tzv. toolboxy (tj. sady nástrojů – funkcí). Pro naši aplikaci bude podstatný „Genetic toolbox“ (jakožto nadstavba optimalizačních funkcí MatLabu), který obsahuje funkce realizující optimalizaci evolučními metodami.

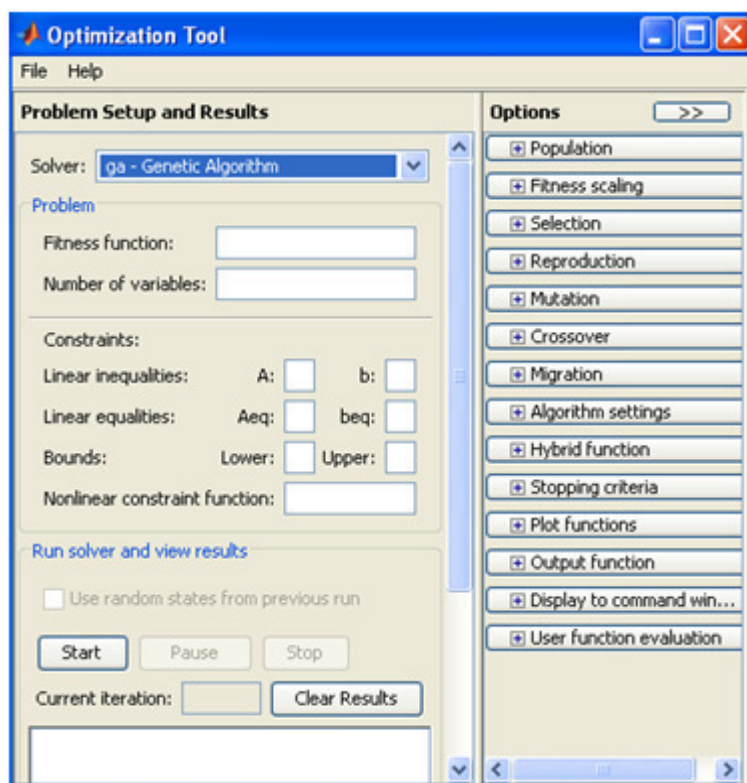
Genetický toolbox obsahuje implementaci evolučních optimalizačních algoritmů jako simulované žíhání, genetický algoritmus a mnoho dalších.

Využití toolboxu nabízí odladěné algoritmy a odbourává tak nutnost jejich programování, čímž vytváří prostor pro intenzivnější soustředění se na jejich aplikaci.

6.4 Výběr evolučního algoritmu

Pro optimalizaci parametrů byl vybrán genetický algoritmus, jenž je efektivní i pro větší množství optimalizovaných parametrů, tím spíše pro naši aplikaci kde pracujeme ani ne s deseti parametry. Genetický algoritmus patří k nejrozšířenějším evolučním algoritmům, dokonce se pojem „genetický algoritmus“ nepřesně používá pro celou množinu algoritmů, které správně popisujeme jako „evoluční algoritmy“.

Genetic toolbox nabízí uživateli poměrně široké možnosti nastavení parametrů samotného genetického algoritmu, např. počet jedinců v populaci, způsob výběru jedince pro křížení, typ a pravděpodobnost mutace a další.



Obr. 45 Prostředí Matlab – Optimtool (optimalizační toolbox rozšířený o evoluční algoritmy)

7 REALIZACE PROGRAMU PRO OPTIMALIZACI REGULÁTORŮ POMOCÍ GENETICKÉHO ALGORITMU

Dle zadání práce měl být vytvořen program realizující návrh regulátoru pro systém 2. řádu s dopravním zpožděním pomocí evolučního algoritmu. Jak bylo zmíněno již v předešlé kapitole, pro optimalizaci byl vybrán genetický algoritmus. Uvedený algoritmus je implementován v nadstavbě prostředí MatLab „Genetic toolbox“ jako funkce, jejímiž vstupními parametry jsou mimo jiné *ukazatel na optimalizovanou funkci* a *počet optimalizovaných parametrů*. Prvním krokem je proto vytvoření funkce reprezentující náš optimalizační problém. Dále je potřeba nastavit volitelné vlastnosti genetického algoritmu (počet jedinců v populaci, typ výběru jedince pro křížení, typ křížení, typ a četnost mutací, atd.). V neposlední řadě je pak zapotřebí reprezentovat výsledky formou přijatelnou pro uživatele.

Realizaci programu lze rozdělit na dva základní kroky:

1. Vytvoření kritériální funkce
2. Optimalizace kritériální funkce
 - a. Nastavení fixních parametrů kritériální funkce
 - b. Nastavení parametrů optimalizačního algoritmu
 - c. Reprezentace výsledků

7.1 Vytvoření kritériální funkce

Chceme optimalizovat parametry regulátoru, proto musíme vytvořit kritériální funkci tak, aby její vstupní parametry měly význam parametrů regulátoru a její výstupní hodnota „fitness“ vypovídala o kvalitě regulace.

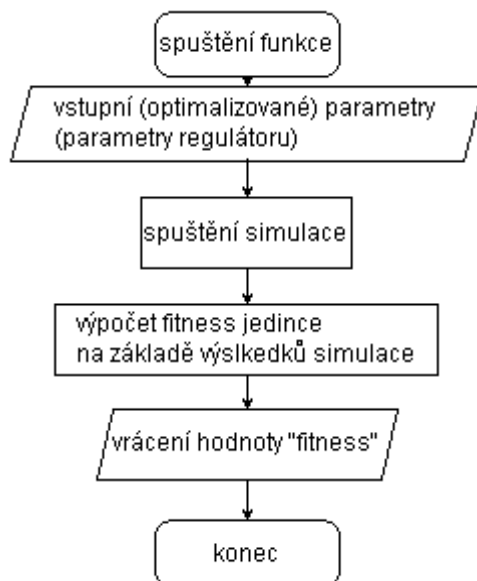
Kritériální funkce je realizována jako funkce prostředí MatLab v souboru typu M-file, pojmenována byla *simulacee*. Volání funkce má tvar:

$$fitness_jedince = simulacee(parametr1,parametr2,...)$$

Z funkce je volána simulace v prostředí *MatLab Simulink*. Simulační model představuje zpětnovazební regulační smyčku obsahující zadaný systémem a regulátor, jehož parametry nejsou konstanty ale proměnné, se kterými byla funkce *simulacee* volána.

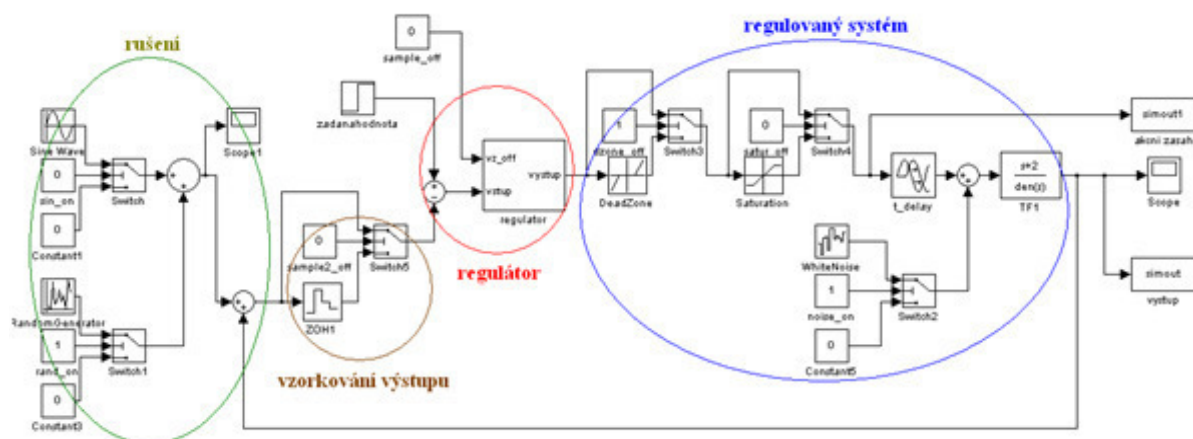
Výsledky simulace jsou exportovány do lokálních proměnných funkce *simulacee* a na základě těchto výsledků je posuzována kvalita regulace – je spočítána hodnota fitness, která je zároveň návratovou hodnotou.

Základní vývojový diagram funkce simulacee je na následujícím obrázku.



Obr. 46 Základní vývojový diagram (realizace) kritériální funkce

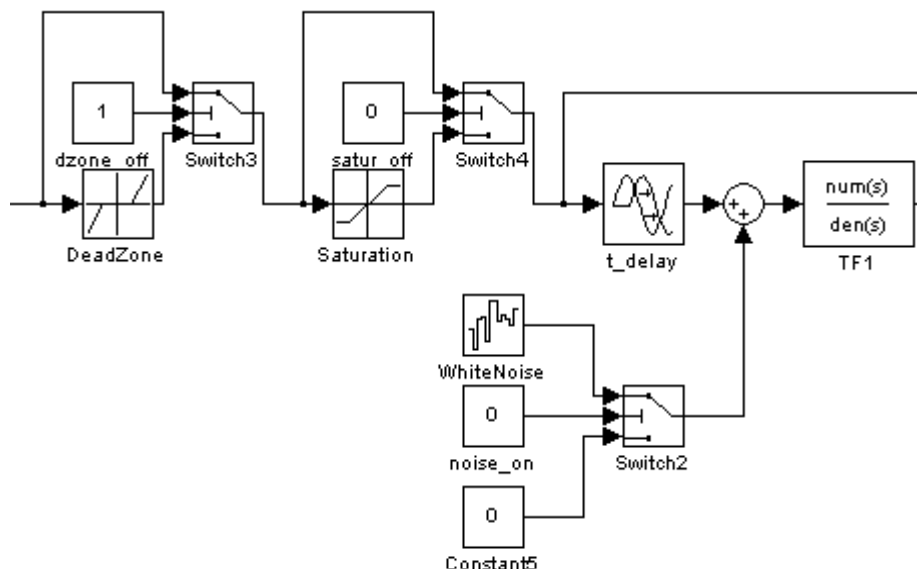
7.1.1 Simulační model



Obr. 47 Simulační model (zvětšený obrázek v příloze 1)

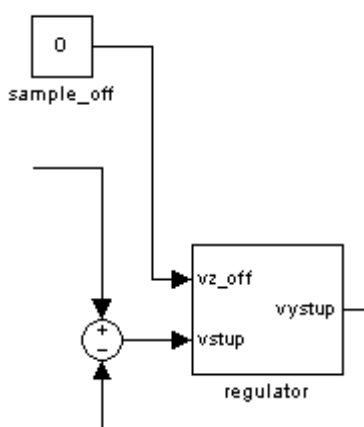
Základem simulačního modelu je zpětnovazební smyčka obsahující regulátor a regulovaný systém.

Regulovaný systém se skládá lineární části přenosu $TF1$, dopravního zpoždění t_{delay} , saturace $Saturation$, pásma necitlivosti $DeadZone$ a šumu $WhiteNoise$ vstupujícího do systému. Parametry všech zmíněných bloků (detail níže) jsou nastaveny staticky před začátkem optimalizace. Lineární část systému je tvořena přenosovou funkcí, koeficienty polynomu čitatele $num(s)$ a jmenovatele $den(s)$ jsou zadány jako dva vektory reálných čísel reprezentujících koeficienty polynomů. Časové zpoždění je zadáno jako nezáporné reálné číslo (zpoždění může být i nulové, tj. žádné). U saturace definujeme horní a dolní mez, u pásma necitlivosti začátek a konec pásma, u šumu jeho výkon, všechny tři uvedené bloky lze přemostit (pro připojení resp. přemostění slouží přepínače $Switch$).



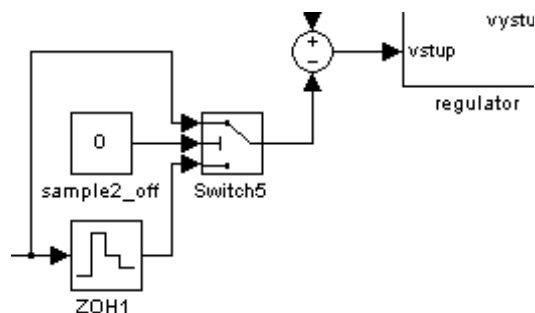
Obr. 48 Detail simulačního modelu – regulovaný systém

Regulátor je do schématu vložen jako subsystém, jeho vnitřnímu zapojení se budeme věnovat později, nyní uvedeme pouze tolik, že tvoří blok se dvěma vstupy a jedním výstupem. První vstup funguje jako přepínač mezi spojitým a diskrétním regulačním algoritmem („0“ – diskrétní regulátor, „1“ – spojitý regulátor). Druhým vstupem je regulační odchylka, výstupem akční zásah.



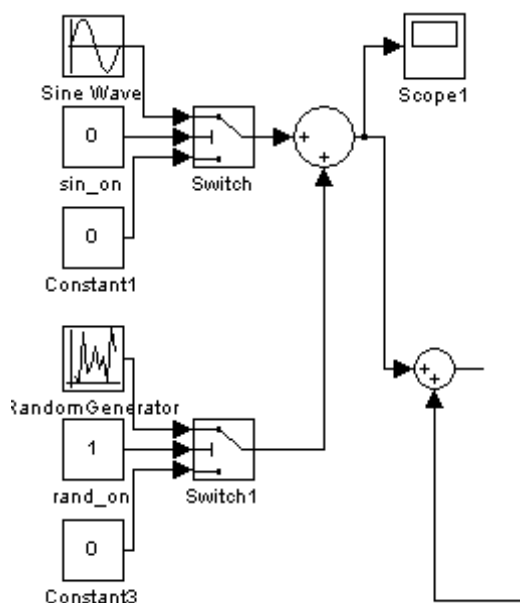
Obr. 49 Detail simulačního modelu – regulátor

Před vstupem do regulátoru je umístěn vzorkovač. Blok reprezentuje skutečnost, že v některých aplikacích měříme výstup regulovaného systému digitálními snímači, které nefungují spojitě. Periodu vzorkování lze v simulačním modelu nastavit, rovněž je možno ji vyřadit.



Obr. 50 Detail simulačního modelu – vzorkování výstupní veličiny

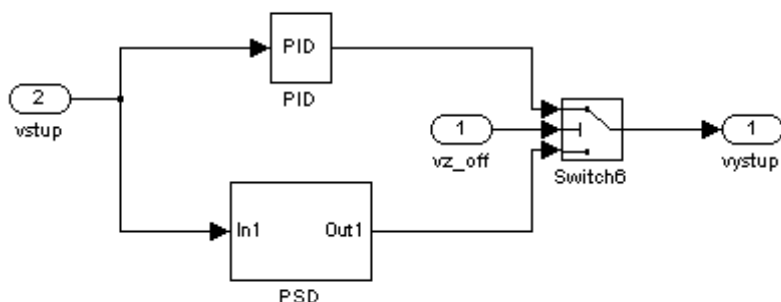
Měření výstupu může být ovlivněno rušením. Abychom mohli pozorovat vliv rušení, jsou k výstupnímu signálu přičteny signály parazitní – sinusový reprezentující síťové rušení a náhodný reprezentující chybu snímání. Amplitudu signálů lze nastavit, rovněž je možno uvedené bloky odpojit.



Obr. 51 Detail simulačního modelu – generátor signálu rušení

7.1.2 Tvary optimalizovaných regulátorů

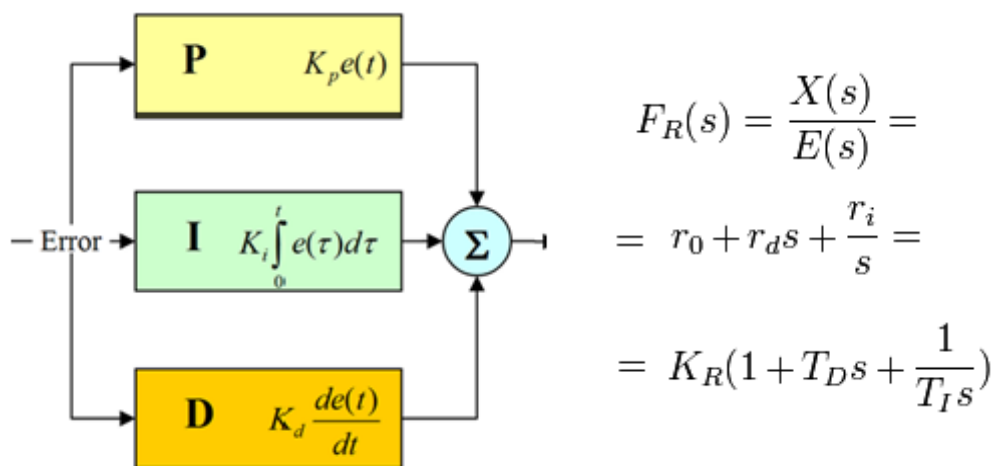
Blok „regulator“ simulačního modelu obsahuje dvě varianty regulačního algoritmu, sice spojitý regulátor PID a jeho diskrétní obdobu PSD. Uživatel může zvolit vždy jednu z variant. Regulátor PID (ve své spojitě i diskrétní době) je velmi rozšířený a charakteristický svou značnou robustností, mimo jiné obsahuje integrační složku, čímž zajistí nulovou ustálenou odchylku. Pokud bychom chtěli řešení rozšířit o hledání regulátorů obecného tvaru, nejen že bychom museli navíc generovat samotnou strukturu, ale bylo by velmi vhodné provádět alespoň základní vyšetření robustnosti, řešení by tak výrazně nabylo na své složitosti. Uvedená varianta však již přesahuje rozsah zadané práce, proto se dále budeme věnovat regulátorům tvaru PID.



Obr. 52 Detail simulačního modelu – vnitřní zapojení bloku „regulator“

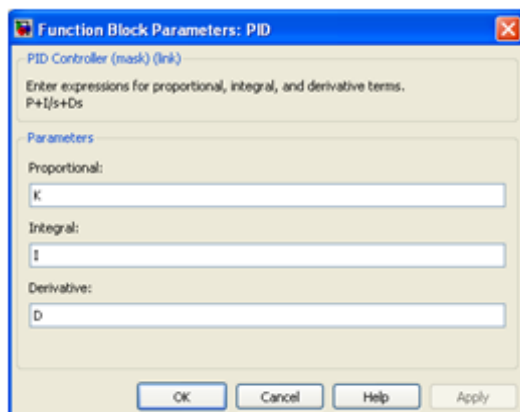
Na obrázku je vidět vnitřní zapojení bloku „regulator“, jak bylo řečeno, obsahuje spojitou a diskrétní verzi PID regulátoru. Uživatel vybírá regulační algoritmus nastavením přepínače *Switch6*.

Spojité PID regulátor najdeme mezi nástroji simulačního prostředí Simulink.



Obr. 53 Regulátor PID a jeho operátorový přenos[16]

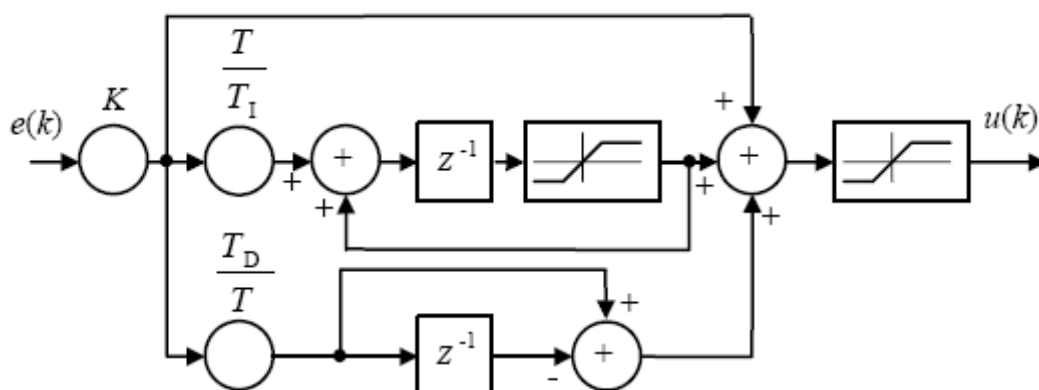
Pro popis bloku využívá Simulink první z forem složkového tvaru.



$$F_R(s) = K + I / s + D * s$$

Obr. 54 Dialogové okno pro manuální nastavení parametrů a vztah popisující přenos bloku

Regulační algoritmus PSD již není knihovní funkcí Simulinku, byl vytvořen jako model stavového schématu PSD regulátoru dle předlohy v [29].

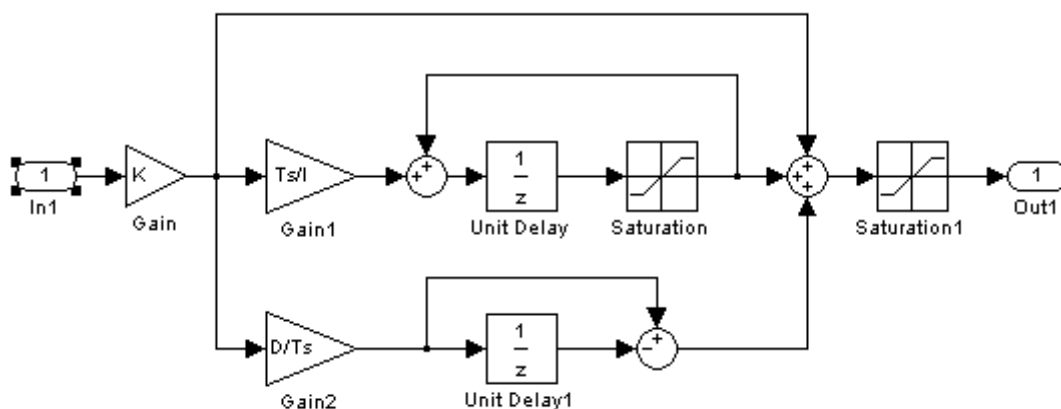


$$F_R(z) = K \left(1 + \frac{Tz^{-1}}{T_I(1 - z^{-1})} + \frac{T_D}{T}(1 - z^{-1}) \right) = \frac{U(z)}{E(z)}$$

$$u(k) = K \left(e(k) + \frac{T}{T_I} \sum_{i=1}^k e(i) + \frac{T_D}{T} (e(k) - e(k-1)) \right)$$

Obr. 55 Stavový diagram polohového PSD regulátoru a vztahy pro přenos a akční zásah [29]

Stavový model odpovídá vztahu pro přenos, je navíc doplněn omezením integrační složky a omezením akčního zásahu. Na následujícím obrázku je pak samotná implementace v Simulinku, jedná se o vnitřní strukturu „subsystemu“ PSD. Meze obou bloků saturace byly nastaveny na -1000 (dolní mez) a 1000 (horní mez).



Obr. 56 Realizace PSD regulátoru v Simulinku

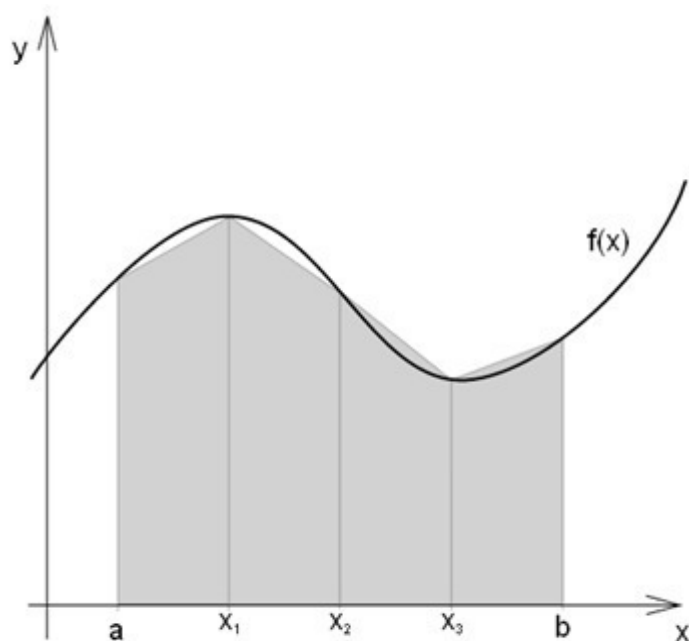
PSD regulátor je popsán druhou formou složkového popisu – jako „polohový regulátor“, konstanty mají pozměněný význam oproti základnímu složkovému tvaru, jsou na něj však převoditelné.

7.1.3 Výpočet hodnoty „fitness“

Na základě výsledků simulace je spočítána hodnota *fitness*, tedy vhodnost jedince reprezentujícího konkrétní kombinaci nastavení parametrů regulátorů.

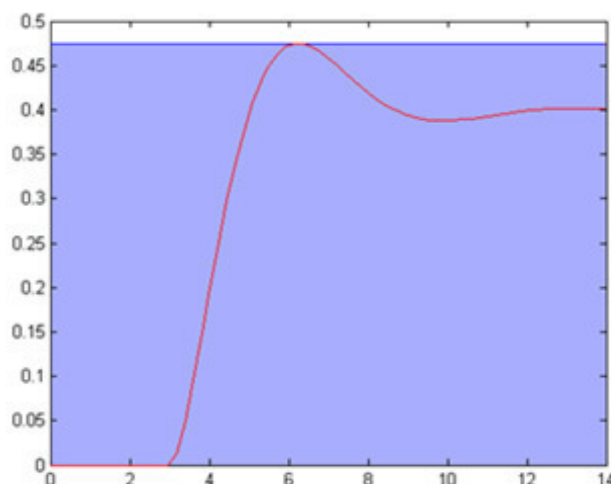
Ještě před spuštěním samotné simulace je ověřeno, zda je kombinace parametrů přípustná (např. není-li délka vzorkovací periody záporná, apod.), pokud jsou parametry nekorektní, simulace spuštěna není a vrací se rovnou velká hodnota *fitness* (tak, aby výrazně převyšovala *fitness* ostatních jedinců).

Pokud jsou parametry korektní a simulace proběhla, průběh výstupu a akčního zásahu byly uloženy do pracovních proměnných (typu pole). Základní hodnotu *fitness* spočítáme podle jednoho (vybraného) integrálního kritéria regulace – integrujeme plochu pod křivkou časového průběhu regulační odchylky, pro integraci je použita lichoběžníková metoda numerické integrace (jemnější krok simulace zvýší přesnost integrování, ale příliš vysoký počet kroků může způsobit naopak nepřesnosti ve výpočtu průběhu na „spojité“ soustavě vlivem nepřesností numerického řešení diferenciálních rovnic, navíc větší počet kroků simulace znamená zvýšení časových nároků na výpočetní zařízení, je proto potřeba brát všechny tyto skutečnosti v úvahu – přednastavená hodnota minimálního počtu kroků je 500 kroků na jeden běh simulace a lze ji modifikovat z uživatelského rozhraní).



Obr. 57 Lichoběžníková metoda numerické integrace

Vedle minimalizace integrální plochy kontrolujeme velikost maximální hodnoty výstupu, chceme tak zajistit, aby výstup nepřesáhl maximální dovolenou hodnotu. Přítomnost překmitu (který přesahuje dovolenou mez) penalizujeme zvýšením hodnoty *fitness*. V naší aplikaci je k hodnotě integrálního kritéria přičteno číslo odpovídající integrálu plochy vymezené časovým intervalem simulace a maximální hodnotou výstupu (hodnota není konstantní a lze tak větší překmity penalizovat více).



Obr. 58 Výpočet velikosti penalizace překmitu

V neposlední řadě požadujeme, aby akční zásah nebyl příliš kmitavý. Kolísání akčního zásahu je dvojího druhu, jednak vysokofrekvenční způsobené reakcí derivační složky šum (rušení), jednak jsou to výrazné (převážně nízkofrekvenční) kmity způsobené kombinací saturace a nevhodného nastavení regulátoru.

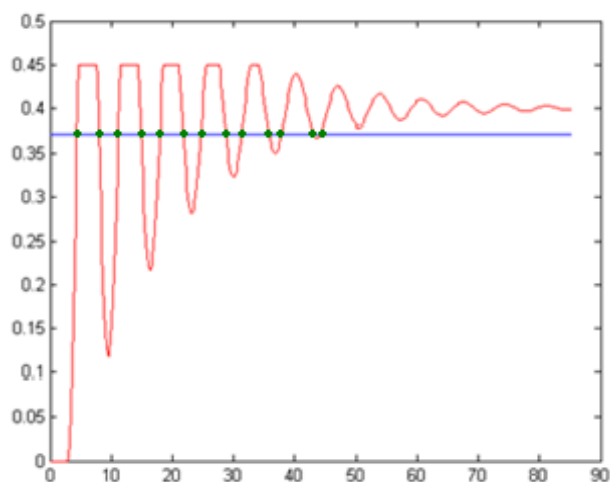
Aby byla postižena míra vysokofrekvenčního kolísání akčního zásahu, je k hodnotě fitness přičtena suma absolutních hodnot druhé difference a hodnoty obdobně spočítané, viz vztahy (kde x je akční zásah). Pomocí proměnné „váha“ lze nastavit míru potlačení kmitání (použita hodnota „1“).

$$fitness = fitness + \left(\sum_k |\Delta_2(k)| + \sum_k |\Delta_{2m}(k)| \right) / k * vaha$$

$$\Delta_2(k) = x(k-1) - 2x(k) + x(k+1)$$

$$\Delta_{2m}(k) = x(k-1) + x(k) - 2x(k+1)$$

Markantnější kolísání penalizujeme mnohem výraznějším způsobem, hodnota fitness je vynásobena počtem průchodů střední hodnotou (ideální stav je jeden průchod).

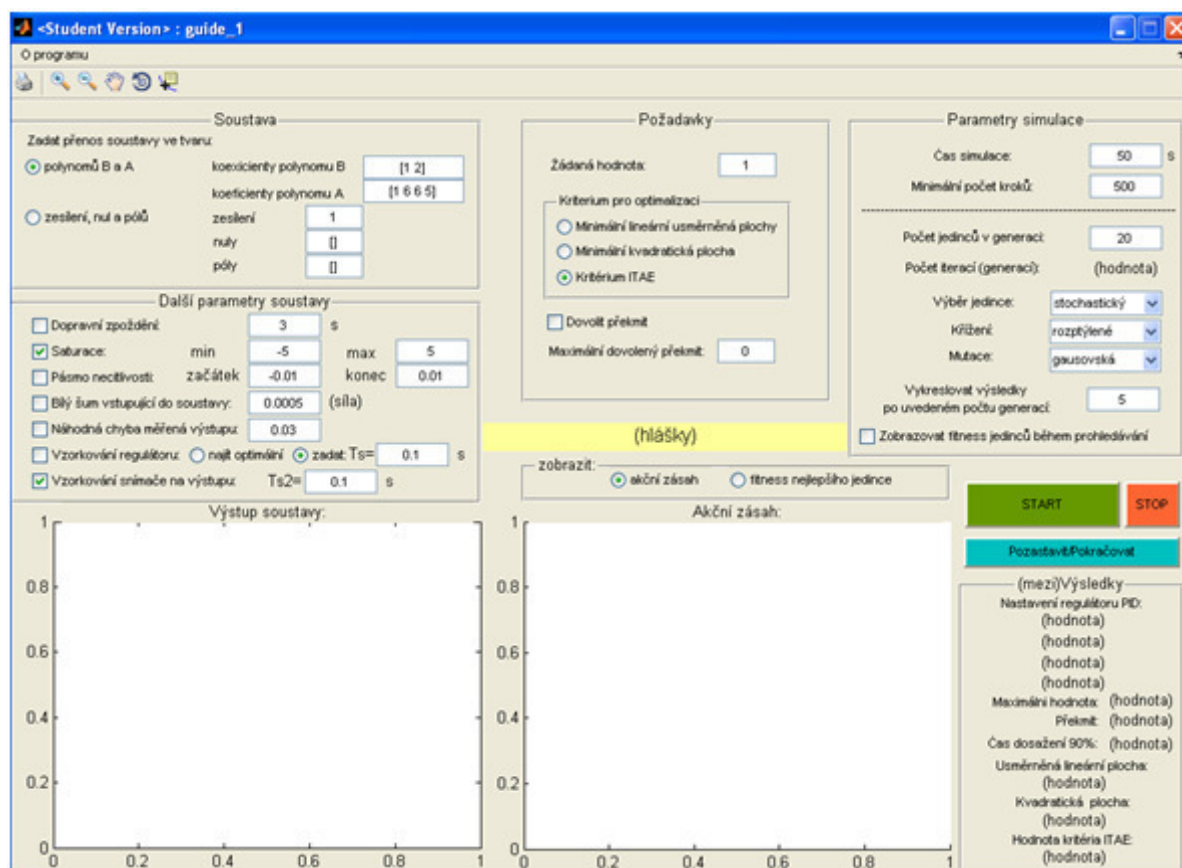


Obr. 59 Počítání průchodů průměrnou hodnotou

7.2 Uživatelské rozhraní programu

Druhou fází vytváření programu byla tvorba uživatelského rozhraní. V předchozí kapitole byl popsán postup vytvoření kritériální funkce reprezentující náš optimalizační problém. Uvedenou kritériální funkci je možné optimalizovat genetickým algoritmem přímo (z textového rozhraní MatLabu), ale uživatel nemá představu o průběhu optimalizace, algoritmus vrátí pouze nejlepší nalezený výsledek. Chce-li uživatel provést změnu v regulovaném systému, nebo chce-li změnit vlastnosti optimalizační funkce, musí vše provádět ručně.

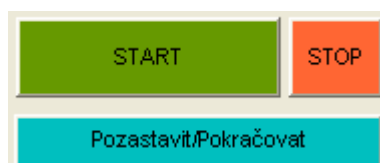
K dosažení potřebné míry komfortu, bylo pro uživatele vytvořeno grafické rozhraní, ze kterého je nejen spouštěna optimalizace, ale jsou zde nástroje pro modifikaci parametrů systému, nástroje pro nastavení požadavků optimalizace a nástroje pro nastavení vlastností samotného optimalizačního (genetického) algoritmu.



Obr. 60 Základní grafické rozvržení aplikace (zvětšený snímek v příloze 2)

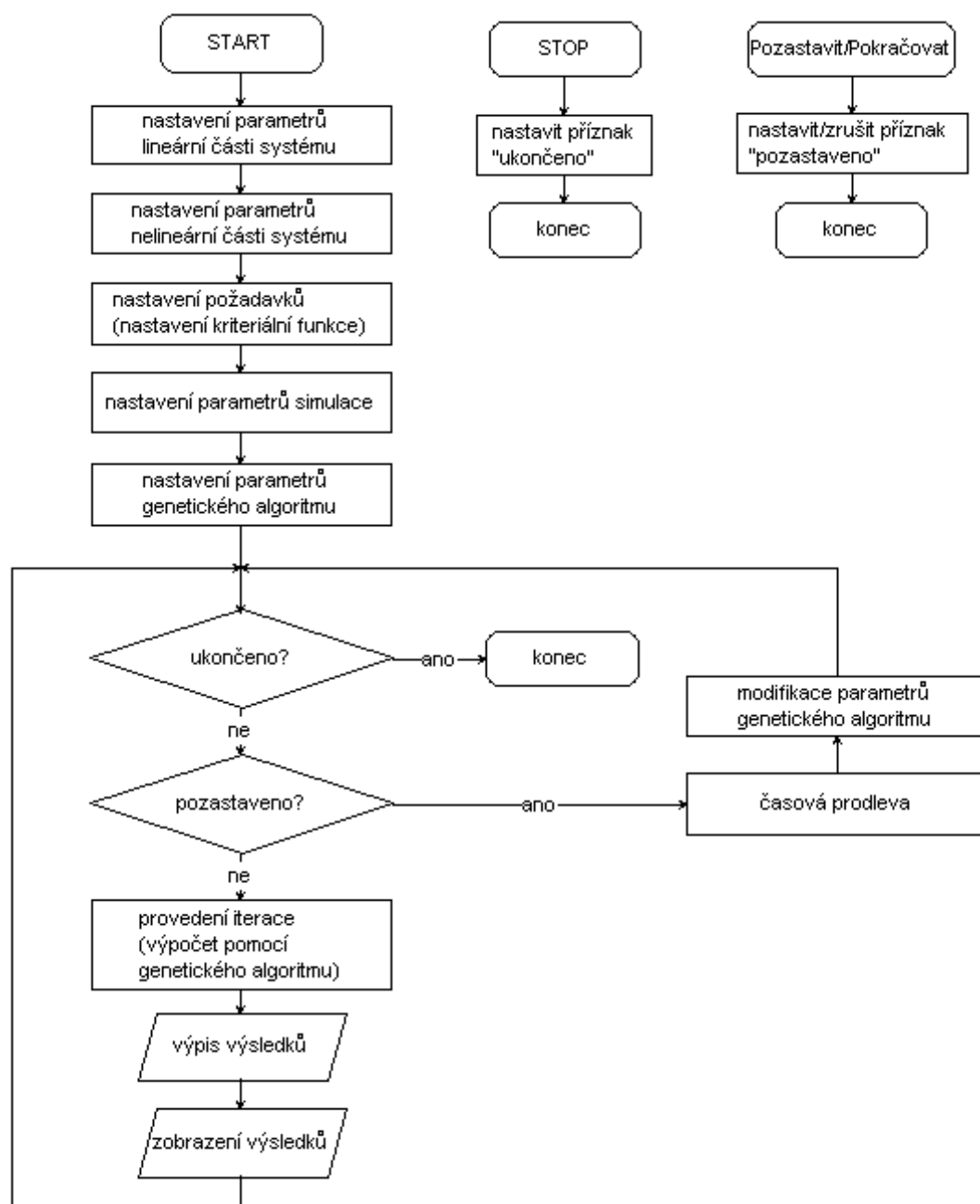
7.2.1 Ovládací tlačítka

Grafické rozhraní je rozčleněno do několika oddílů, v horní polovině okna jsou nástroje pro rozličná nastavení, v dolní polovině jsou pak zobrazovány výsledky (graficky i numericky). K základnímu ovládání slouží tři ovládací tlačítka „START“, „STOP“ a „Pozastavit/Pokračovat“. Názvy tlačítek samy o sobě vypovídají o funkci, „START“ spouští a „STOP“ zastavuje optimalizaci na základě aktuálního nastavení. Při pozastavení optimalizace pomocí „Pozastavit/Pokračovat“ je možno modifikovat vlastnosti optimalizačního algoritmu (ostatní parametry za běhu měnit nelze).



Obr. 61 Ovládací tlačítka

Funkce tlačítek je zachycena ve vývojovém základním diagramu. Vývojový diagram nezahrnuje všechny funkce (kontrola správnosti zadaných parametrů, atd.) aby se nestal příliš komplikovaný a byl pro čtenáře přehledný.



Obr. 62 Základní vývojový diagram funkcí tlačítek

7.2.2 Nástroje pro specifikaci optimalizačního problému

Systém je zadán jako soustava obecného (rovněž druhého) řádu s dopravním zpožděním a základními nelinearitami.

V prvním oddílu s názvem „Soustava“ je popsána dynamika lineární části systému. Dynamiku zadáme buď ve tvaru *přenosu*, nebo ve tvaru *zesílení a rozložení pólů a nul*. Operátorový přenos systému je poměrem Laplaceových obrazů výstupu a vstupu. Přenos je zadán formou dvou vektorů čísel, vektor reprezentuje sestupnou posloupnost koeficientů polynomu (kde neurčitou je operátor „s“). Program automaticky kontroluje správnost zadání. Nejprve ověříme, jsou-li všechny prvky vektoru reálná čísla. Dále je vyšetřena stabilita systému – polynom A je rozložen na kořenové činitele, je-li reálná složka některého z kořenů kladná, program se zastaví a uživateli je oznámeno, že zadáný systém je nestabilní. Zadáný systém musí být kauzální, test kauzality spočívá v porovnání délky vektorů B a A, polynom B musí být nanejvýš stejného stupně jako A, pokud požadavek není splněn, uživatel je upozorněn na skutečnost nekauzálnosti systému a program nepokračuje (nekauzálnost by způsobila zhavarování simulace). Příklady reakce na špatně zadané parametry lineární části systému jsou v příloze 3.

Obr. 63 Nastavení parametrů lineární části systému

Dopravní zpoždění a nelinearity jsou zadávány v druhém oddíle s názvem „další parametry soustavy“. V levém sloupci uživatel určí, zda se nelinearita či rušení v simulaci vůbec projeví, pokud je vlastnost „zatržena“, jsou vyčteny její parametry z jí odpovídajících textových polí, před modifikací simulačního schématu je provedena kontrola správnosti zadaných hodnot (hodnoty musí být reálná čísla, hodnoty musí spadat do povoleného rozsahu – např. dopravní zpoždění je nezáporné, spodní mez saturace je nižší než horní mez, atd.).

Obr. 64 Nastavení dalších parametrů regulovaného systému

V uvedeném oddíle může uživatel zapnout vzorkování snímače výstupu a vzorkování regulátoru. Pokud není zvoleno vzorkování regulátoru, je pro regulaci vybrán regulátor PID. Pokud je zvoleno vzorkování, optimalizujeme parametry regulátoru PSD. Vzorkovací periodu PSD je možno buď zadat ručně, nebo nechat najít optimální (vzorkovací perioda se stane jedním z optimalizovaných parametrů).

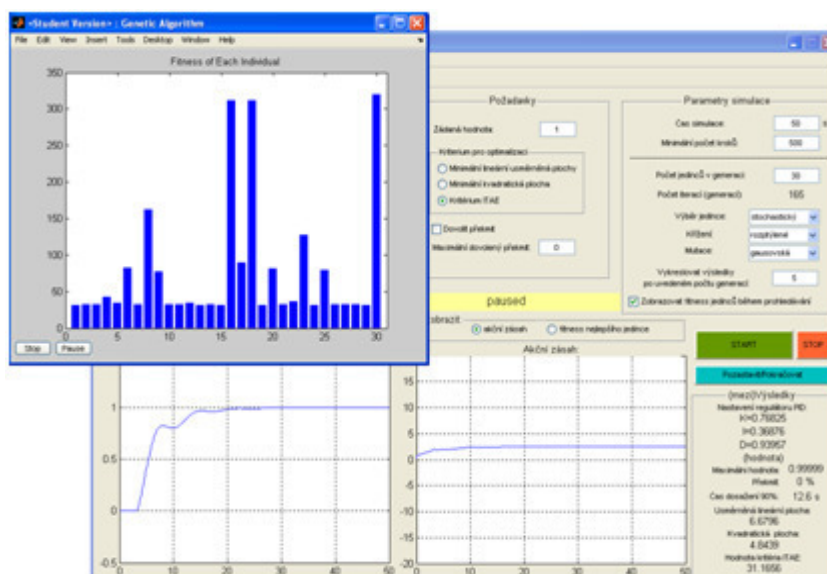
Ve třetím oddíle „Požadavky“ nastavujeme požadavky na regulační děj. Jednak je nastavena žádaná hodnota, dále je zvoleno integrační kritérium, které bude určovat kvalitu regulace. V neposlední řadě je stanoveno, zda je přípustný překmit, a pokud ano, jak velký.

Obr. 65 Nastavení požadavků na regulaci

Poslední oddíl nastavení „Parametry simulace“ slouží k nastavení zbylých parametrů simulace a parametrů optimalizačního algoritmu. V první skupině jsou parametry simulace, jsou to „čas simulace“ – určuje, jak velký časový úsek má být simulován, a „minimální počet kroků“ – hodnota určí jemnost kroků simulace, tedy přesnost. Větší počet kroků simulace zajistí přesnější výpočet integrační plochy, ale zpomalí simulaci.

Obr. 66 Nastavení parametrů simulace a parametrů genetického algoritmu

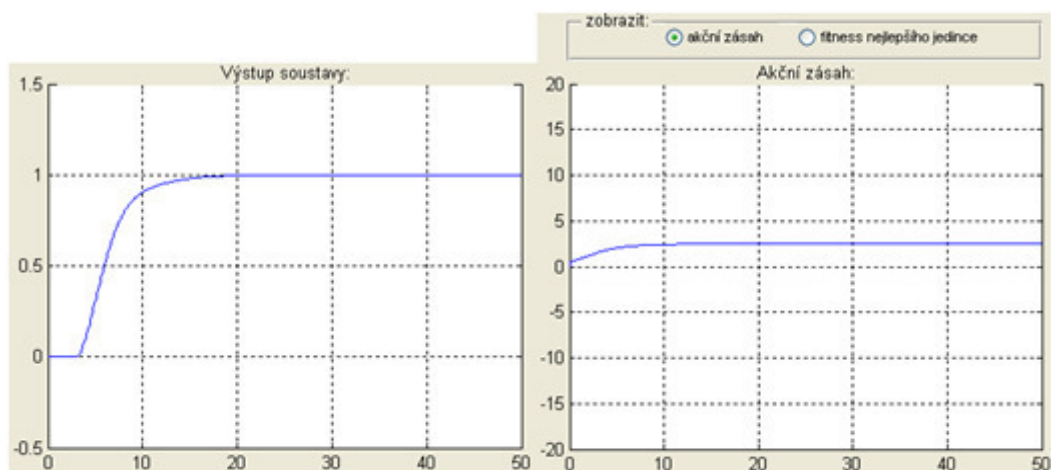
V druhé části oddílu nastavujeme parametry samotného genetického algoritmu – velikost populace, způsob výběru jedince, typ křížení a typ mutace. Můžeme zde sledovat, kolik iterací již proběhlo. Stanovíme, jak často chceme vykreslovat (mezi)výsledky a zda chceme během optimalizace sledovat fitness všech jedinců (následující ilustrační obrázek ukazuje okno s vyobrazením fitness jedinců v populaci).



Obr. 67 Zobrazení fitness všech jedinců v aktuální generaci

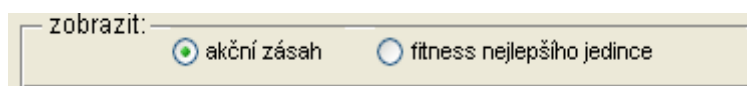
7.2.3 Reprezentace výsledků

Ve zvoleném intervalu jsou vykreslovány (a vypisovány) výsledky. Pro vykreslení byla vytvořena zvláštní funkce, genetický algoritmus zobrazuje pouze vývoj jedinců v populaci, ale vizualizace optimalizace je nadstavbou. Vykreslování charakteristik má velký význam pro uživatele, aby si mohl udělat představu o průběhu optimalizace a o funkci samotného genetického algoritmu. V intervalu zadaném uživatelem je vždy genetický algoritmus přerušen a je uložena aktuální generace, následně je zavolána funkce pro vykreslení, poté je opět zavolán genetický algoritmus s tím, že uložená generace je vložena jako počáteční populace.



Obr. 68 Výpis výsledků

V prvním grafu je vykreslována odezva soustavy (výstup), v druhém grafu je vykreslován akční zásah, nebo vývoj fitness nejlepšího jedince. Mezi dvěma uvedenými možnostmi druhého (sekundárního) grafu lze (i za chodu optimalizace) přepínat.



Obr. 69 Výběr dat pro sekundární graf

V levém horním rohu aplikace se nalézají nástroje pro práci s grafy, uživatel může graf zvětšovat, zmenšovat, posouvat, otáčet a odečítat z něj hodnoty.



Obr. 70 Nástroje pro práci s grafy

V neposlední řadě se v levém dolním rohu aplikace nalézá oddíl s výsledky (mezivýsledky) optimalizace. Zobrazeny jsou nejlepší nalezené parametry regulátoru, maximální hodnota, překmit vyjádřený procentuálně, čas prvního dosažení 90% žádané hodnoty, dále jsou uvedeny hodnoty všech tří integrálních kritérií vypovídající o kvalitě řízení (v časové oblasti).

(mezi)Výsledky

Nastavení regulátoru PID:

(hodnota)

(hodnota)

(hodnota)

(hodnota)

Maximální hodnota: (hodnota)

Překmit: (hodnota)

Čas dosažení 90%: (hodnota)

Usměrněná lineární plocha:

(hodnota)

Kvadratická plocha:

(hodnota)

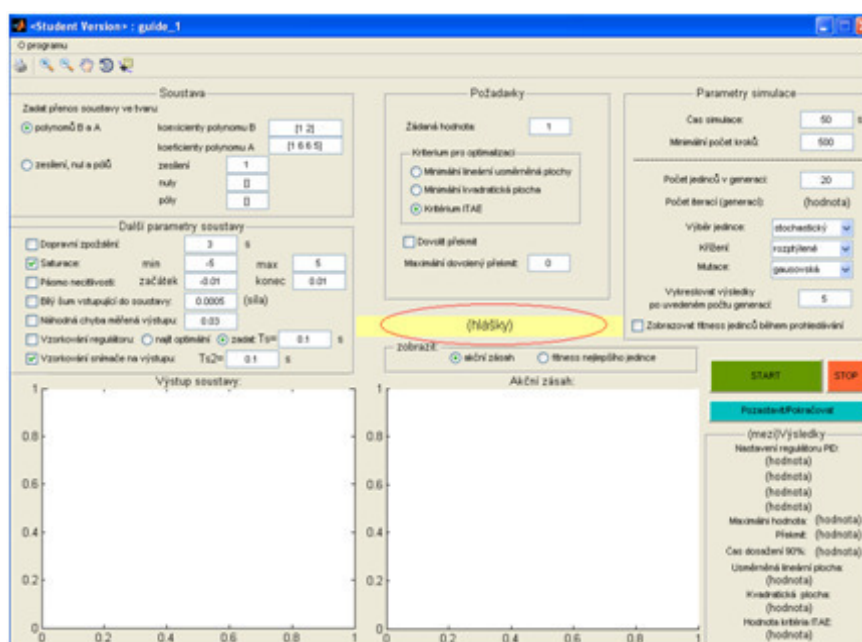
Hodnota kritéria ITAE:

(hodnota)

Obr. 71 Výpis výsledků

7.2.4 Zprávy pro uživatele

Ve středu okna aplikace se nalézá pole určené pro indikaci stavu pro uživatele. Do pole jsou vypisovány informace o chybně zadaných parametrech a o činnosti programu.



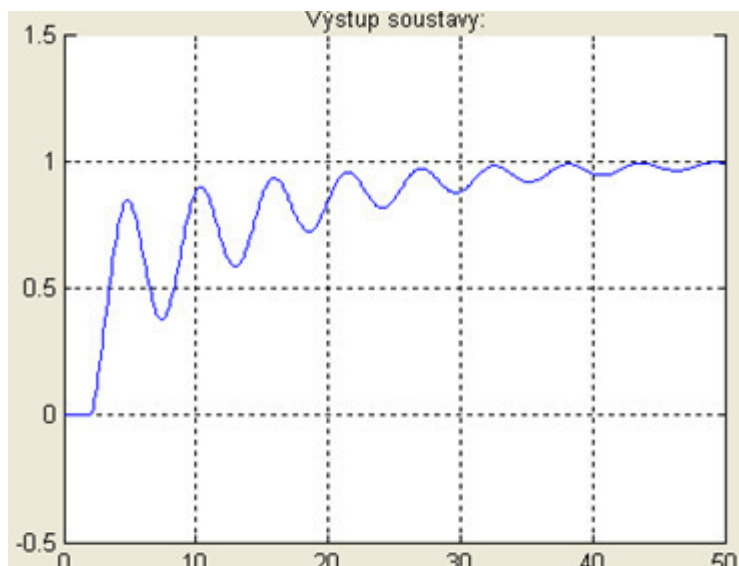
Obr. 72 Pole pro zprávy adresované uživateli

7.3 Testování

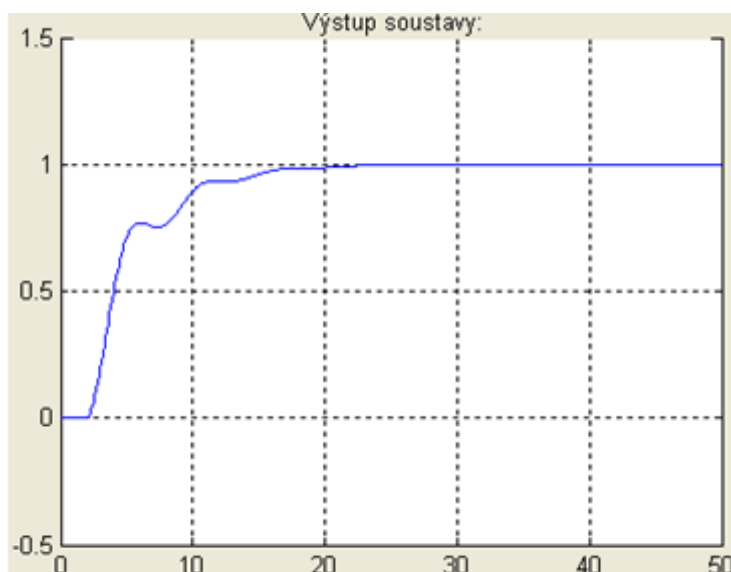
Pokud proběhne úspěšně kontrola parametrů, je spuštěn proces optimalizace. Počáteční populace je vygenerována náhodně, nejen proto je průběh prohledávání vždy jiný (počáteční podmínky jsou odlišné, během prohledávání působí mutace atd.), většinou však dříve či později algoritmus dospěje k podobným výsledkům (v případě PSD se řešení zpravidla liší více).

Rychlost konvergence je ovlivněna (nejen) velikostí populace a nastavením parametrů genetického algoritmu. Regulační děj je typicky v prvních krocích velmi kmitavý a postupně je „usměrňován“. Uvedme ilustrační příklad regulace soustavy s přenosem:

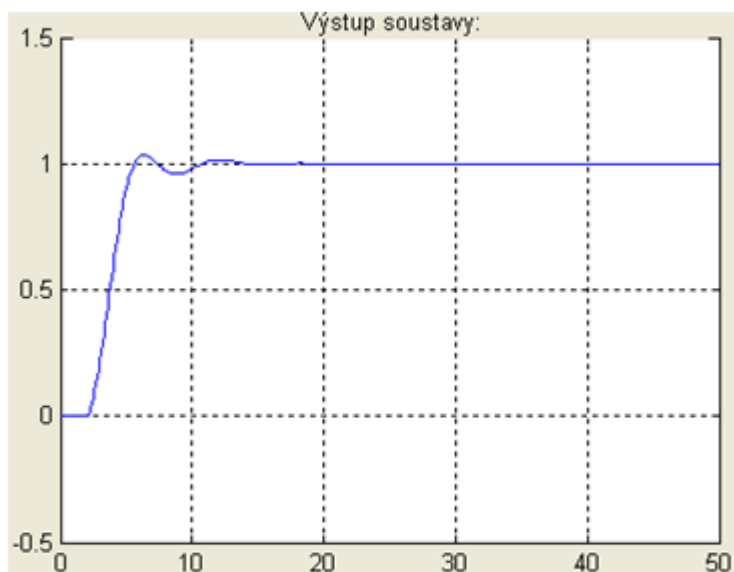
$$F(p) = \frac{s+2}{s^3 + 6s^2 + 6s + 5} * e^{-3s}$$



Obr. 73 Nejlepší nalezené řešení po 2. iteraci



Obr. 74 Nejlepší nalezené řešení po 48. iteraci



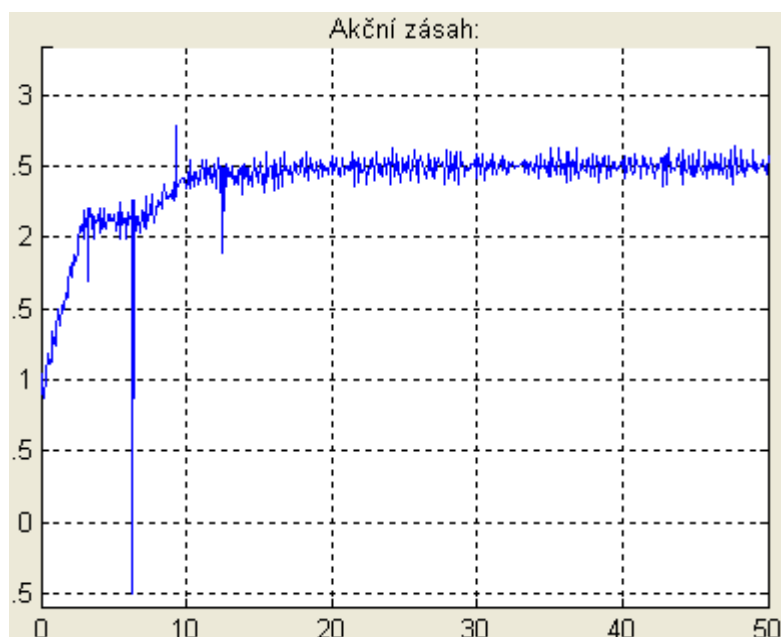
Obr. 75 Nejlepší nalezené řešení po 48. iteraci

Podrobněji je konkrétní příklad ilustrován v příloze 5, kde lze paralelně sledovat i akční zásahy. Pokud nepůsobí rušení (chyba měření výstupu) na vstupu regulátoru, je průběh akčního zásahu „hladký“.



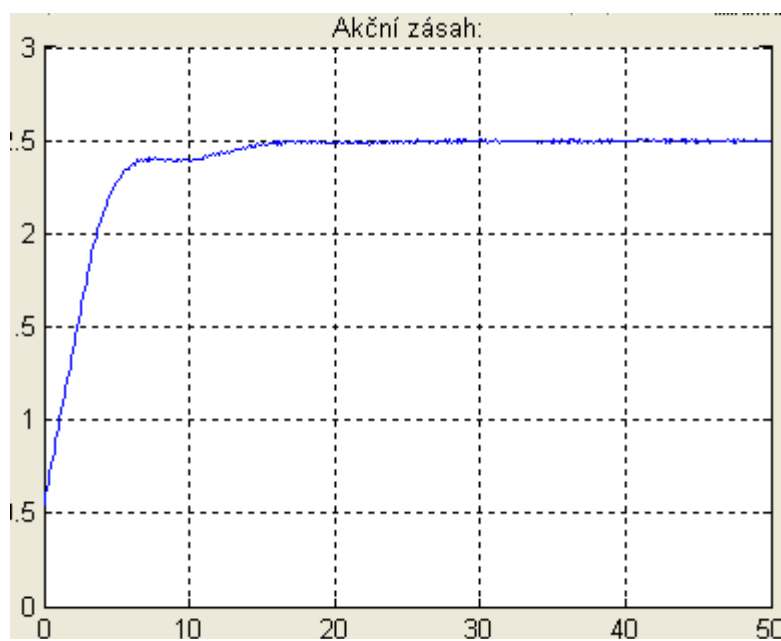
Obr. 76 Akční zásah – varianta bez působení chyby měření výstupu

Pokud se ale na vstupu regulátoru objeví šum, průběh akčního zásahu se výrazně zhorší.



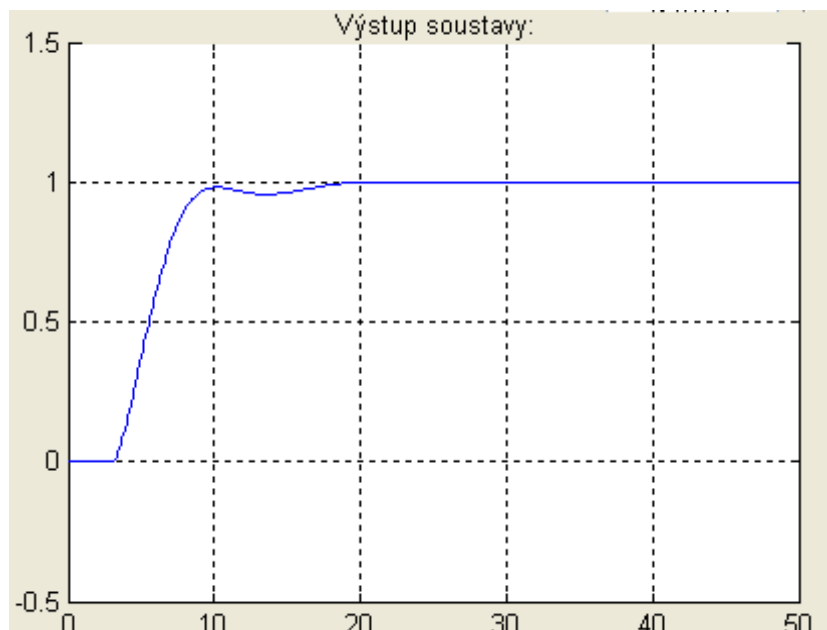
Obr. 77 Akční zásah – varianta s působením chyby měření výstupu

Uvedená míra kmitání je většinou nepřijatelná (obzvláště je-li akčním členem např. servomechanismus), proto je kmitání penalizováno. Algoritmus musí najít lepší řešení, tak že sníží derivační složku.



Obr. 78 Akční zásah – varianta s působením chyby měření výstupu po zmenšení derivační složky (v průběhu iterací)

Snížení derivační složky sníží rychlost a tak i kvalitu řízení (vystiženou integrálními kritérii regulace), výsledek přesto není špatný.



Obr. 79 Výstup – varianta s působením chyby měření výstupu
po zmenšení derivační složky

Optimalizace PSD probíhá obdobně, dosažené výsledky bývají o něco málo horší, značně záleží na zvolené vzorkovací periodě.

7.4 Možnosti pokračování v práci

Kmitání akčního zásahu bychom mohli potlačit také *filtrací akčního zásahu*, filtr by sice zhoršil dynamiku regulátoru, který by však sám o sobě mohl být „rychlejší“, neboť by mohl použít větší derivační složku. Varianta z časových důvodů již vyzkoušena nebyla, ale je podnětem pro případné pokračování.

Dalšími možnostmi rozšíření by bylo vyšetřování robustnosti regulátoru (citlivost na rušení, citlivost na změnu parametrů soustavy, atd.).

8 ZÁVĚR

Cílem diplomového projektu bylo vytvoření programu, který realizuje návrh parametrů regulátoru pomocí evolučních algoritmů. Za vývojový prostředek bylo zvoleno matematické prostředí MatLab rozšířené o nástroj „Genetic toolbox“. Toolbox nabízí metody optimalizace parametrů kritériální funkce evolučními technikami. Za optimalizační funkci byl zvolen genetický algoritmus, který je vzhledem k počtu optimalizovaných parametrů (max. čtyři) vhodný (genetický algoritmus je využíván pro optimalizaci i desítek parametrů). Evoluční algoritmus optimalizuje kritériální funkci, bylo proto nutné převést naši problematiku na optimalizaci funkce.

Prvním z přínosů práce je vytvoření samotné kritériální funkce, která reprezentuje optimalizační problém návrhu regulátoru. Funkce je implementována v souboru M-file, jejími vstupními hodnotami jsou parametry regulátoru, výstupní hodnota fitness je spočítána tak, aby vystihla kvalitu regulace. Jádro kritériální funkce je tvořeno spuštěním a vyhodnocením simulace na modelu, který představuje antiparalelní zapojení systému a optimalizovaného regulátoru. Optimalizovaný regulátor má tvar spojitého PID, nebo diskrétního PSD. U regulátoru typu PSD je možno stanovit periodu vzorkování nebo ji též optimalizovat. Teoreticky by bylo možné realizovat obecný lineární regulátor, uvedená varianta by zahrnovala navíc generování samotného tvaru regulátoru, uvedené řešení by však vyžadovalo kontrolu robustnosti regulátoru. Regulátor PID (ve své spojitě i diskrétní variantě) je sám o sobě velmi robustní, mimo jiné obsahuje integrační složku, čímž zajistí nulovou ustálenou odchylku i při působení poruchy. Pro soustavu druhého řádu s dopravním zpožděním (dle zadání) je PID zcela dostačující, v průběhu testování se ukázalo, že stačí i na složitější systémy. V neposlední řadě je třeba podotknout, že v praxi je PID velmi rozšířený. Při regulaci posuzujeme, zda jsou přípustné překmity, a zda není akční zásah příliš kmitavý, samotnou kvalitu regulačního děje hodnotíme pomocí integrálních kritérií regulace.

Druhým přínosem je vizualizace procesu optimalizace. Samotné evoluční algoritmy během své činnosti zobrazují pouze hodnoty fitness, ze kterých si uživatel jen těžko utvoří představu o regulačním ději. Vizualizace není podstatná pro nalezení samotného výsledku, uživatel si však může vytvořit představu o fungování optimalizační funkce (genetického algoritmu) a posoudit, zda je aktuální nastavení parametrů genetického algoritmu vhodné.

V neposlední řadě bylo zapotřebí vytvořit uživatelské prostředí, ze kterého by bylo možné nastavovat vlastnosti regulovaného systému, požadavky na regulaci a vlastnosti genetického algoritmu. Základní ilustrace vzhledu vytvořeného programu je přiložena v příloze 2. Podrobným popisem prostředí se zabývá kapitola 7. Program obsahuje okna s grafy vypovídajícími o průběhu regulace, uživatelská nastavení jsou podrobena kontrole a o výskytu nesrovnalostí (např. chybný znak, zadaný nekauzální či nestabilní systém pro regulaci, špatná specifikace nelinearity, atd.) je uživatel informován. Vlastnosti regulovaného systému a požadavky na regulaci je nutno stanovit před spuštěním optimalizace, vlastnosti optimalizačního algoritmu však lze měnit i během procesu optimalizace (v pozastaveném stavu).

Program, jehož realizace představuje stěžejní část práce, je přiložen na nosiči CD. Program se volá z prostředí MatLab spuštěním souboru *guide_1.m*.

V závěru poslední kapitoly jsou demonstrovány a diskutovány některé z dosažených výsledků testování programu, další příklady jsou v tištěné příloze.

9 SEZNAM POUŽITÉ LITERATURY

- [1] Poliščuk, R.: Doporučené zásady pro vypracování diplomových/bakalářských prací '07
http://autnt.fme.vutbr.cz/doc/SZZ2008_Instrukce.pdf
- [2] Švarc, Ivan; Šeda, Miloš; Vítečková, Miluše. *Automatické řízení*. 1. vyd. Brno : AKADEMICKÉ NAKLADATELSTVÍ CERM, s.r.o., 2007. 324 s. ISBN 978-80-214-3491-2.
- [3] Zelinka, Ivan; Oplatková, Zuzana; Šeda, Miloš; Ošmera, Pavel; Včelař, František. *Evoluční výpočetní techniky : Principy a aplikace*. Praha : BEN - technická literatura, 2008. 536 s. ISBN 978-80-7300-218-3
- [4] Šťastný, Jiří. *Simulace systémů*. Brno : VUT, FSI, ÚAI, 2002. 98 s.
- [5] Jura, Pavel. *Signály a systémy : Část 1 Spojité signály*. Brno : VUT, FEKT, ÚAMT, 2003. 81 s.
- [6] Jura, Pavel. *Signály a systémy : Část 2 Spojité systémy*. Brno : VUT, FEKT, ÚAMT, 2003. 78 s.
- [7] Jura, Pavel. *Signály a systémy : Část 3 Diskrétní signály a diskrétní systémy*. Brno : VUT, FEKT, ÚAMT, 2003. 89 s.
- [8] Blaha, Petr; Vavřín, Petr. *Řízení a regulace I : Základy regulace lineárních systémů - spojitě a diskrétně*. Brno : VUT, FEKT, ÚAMT, 2005. 214 s.
- [9] Šolc, František; Václavek, Pavel; Vavřín, Petr. *Řízení a regulace II : Analýza a řízení nelineárních systémů*. Brno : VUT, FEKT, ÚAMT, 2006. 228 s.
- [10] Štecha, Jan; Havlena Vladimír. *Teorie dynamických systémů*. Praha : Ediční středisko ČVUT, 2005. 248 s.
- [11] Pivoňka, Petr. *Optimalizace regulátorů*. Brno : VUT, FEKT, ÚAMT, 2005. 112 s.
- [12] Zezulka, František; Bradáč, Zdeněk; Fiedler, Petr; Kučera, Pavel; Štohl, Radek. *Programovatelné automaty*. Brno : VUT, FEKT, ÚAMT, 2003. 79 s.
- [13] Lacko, Bronislav; Beneš, Pavel; Maixner, Ladislav; Šmejkal, Ladislav. *Automatizace a automatizační technika 1*. Praha : Computer Press, 2000. 97 s. ISBN 80-7226-246-7
- [14] Wikipedie. *Wattův odstředivý regulátor* [on-line]. 2004, 21. 8. 2009 [cit. 19.5.2010]. Dostupné z: http://cs.wikipedia.org/wiki/Wattův_odstředivý_regulátor
- [15] Němec, Zdeněk. *Prostředky automatického řízení : elektrické*. 2. vyd. Brno, VUT, FSI, ÚAI, 2008.
- [16] Wikipedie. *PID regulátor* [on-line]. 2009, 17. 5. 2010 [cit. 19.5.2010]. Dostupné z: http://cs.wikipedia.org/wiki/PID_regulátor
- [17] Wikipedie. *Jednočipový počítač* [on-line]. 2007, 15. 3. 2010 [cit. 19.5.2010]. Dostupné z: <http://cs.wikipedia.org/wiki/Mikrokontrolér>
- [18] CKS Global Solutions Ltd. *Industrial PC Products : SB300 Industrial PC module* [on-line]. [cit. 19.5.2010]. Dostupné z: http://www.cksglobal.net/productdetail.php?product_id=SB300
- [19] ferret Australia's Manufacturing and Industrial Directory. *Xycom 4117T Industrial PC from Balmoral Technologies* [on-line]. [cit. 19.5.2010]. Dostupné z: <http://www.ferret.com.au/c/Balmoral-Technologies/Xycom-4117T-Industrial-PC-from-Balmoral-Technologies-n669299>
- [20] Zezulka, František. *Prostředky průmyslové automatizace*. Brno : VUT, FEKT, ÚAMT, 2004. 161 s.
- [21] Marcelm. *Fuzzy Logika* [on-line]. [cit. 19.5.2010]. Dostupné z: <http://marcelm.szm.com/>
- [22] Malcolm Stagg. *Artificial Neural Networks* [on-line]. 2006 [cit. 19.5.2010]. Dostupné z: <http://www.odec.ca/projects/2006/stag6m2/background.html>
- [23] Blaha, Petr. *Modelování a identifikace : Neparametrické metody identifikace* [on-line]. 2009. 69 s. [cit. 19.5.2010]. Dostupné z: <http://sites.google.com/site/modelovaniaidentifikace/přednášky>
- [24] Blaha, Petr. *Modelování a identifikace : Úvodní přednáška* [on-line]. 2009. 69 s. [cit. 19.5.2010]. Dostupné z: <http://sites.google.com/site/modelovaniaidentifikace/přednášky>
- [25] Max Dama on Automated Trading. *Trading Optimization: Simulated Annealing* [on-line]. 2009. [cit. 20.5.2010]. Dostupné z: <http://www.maxdama.com/2008/07/trading-optimization-simulated.html>
- [26] Generation5. *Genetic Algorithms* [on-line]. 1999. 6.2.2005 [cit. 20.5.2010]. Dostupné z: <http://www.generation5.org/content/2005/gaLego.asp>
- [27] Utah State University. *Alexander I. Boldyrev* [on-line]. 2008 [cit. 20.5.2010]. Dostupné z: <http://>

- www.chem.usu.edu/faculty_staff/webpages/alexbold.php>
- [28]Riset Operasi. *Ant Colony Optimization* [on-line]. 21. 1. 2009 [cit. 20.5.2010]. Dostupné z:
<<http://risetoperasi.wordpress.com/2009/01/21/ant-colony-optimization/>>
- [29]Pivoňka, Petr. *Číslicová řídicí technika*. Brno : VUT, FEKT, ÚAMT, 2003. 151 s.