Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Disertační práce

Prediktivní řízení procesů s využitím prvků umělé inteligence

Predictive Control of Processes with Utilization of Artificial Intelligence Elements

Autor: **Ing. Jan Antoš**

Studijní program: P3902 Inženýrská informatika

Studijní obor: 3902V037 Automatické řízení a informatika

Školitel: doc. Ing. Marek Kubalčík, Ph.D.

Zlín, červen 2019

ABSTRAKT

Prediktivní řízení procesů je metoda regulace vhodná pro řízení různých typů systémů, která je založená na myšlence využití predikce budoucího chování systému a její optimalizace. Běžně se pro predikci chování využívá modelu systému, a proto je nutné pro správnou funkci prediktivního řízení provést jeho správný výběr a určit jeho parametry tak, aby byl co nejpřesněji popsán řízený systém. Další výhodou prediktivního řízení je možnost zahrnutí omezení signálů přímo do regulátoru. Cílem této práce je aplikace některých prvků umělé inteligence ve vhodných oblastech prediktivního řízení, zejména využití jednoduchých evolučních algoritmů v rámci optimalizace a neuronových sítí jako nelineárních modelů. Práce popisuje možnosti nasazení těchto prvků. Je prokázáno, že kromě klasických optimalizačních algoritmů je možné použít i jednoduché evoluční algoritmy pro optimalizaci predikce, přičemž výpočetní náročnost může být srovnatelná v závislosti na typu řešeného problému a nastavení. Dále se práce zabývá výběrem vhodných modelových systémů s pomalou dynamikou, jejich odvozením a vytvořením nelineárních modelů v podobě škálovatelných neuronových sítí. Potenciální výhodnost tohoto přístupu pro řízení systémů obtížně popsatelných či pro řízení systémů, jejichž matematicko-fyzikální popis není znám, byla v práci prokázána. Práce se také zabývá možností nasazení nalezených modelů na reálné systémy a stanovením nutných podmínek a požadavků pro jejich aplikaci.

KLÍČOVÁ SLOVA

Prediktivní řízení, MPC, diskrétní řízení, diskrétní dynamické modely, umělá inteligence, neuronové sítě, evoluční algoritmy, nelineární systémy, modelové systémy, nádrže, optimalizace, kvadratické programování, omezení.

ABSTRACT

Predictive control is a method of control process which is suitable for different types of systems. This method is based on the utilization of prediction of the future behaviour of a system and its optimization. A model of this system is mainly used for prediction, hence it is crucial to choose the model properly and set its parameter so that it describes the behaviour of the system as precisely as possible. Another advantage of the predictive control is the possibility to directly apply constraints within a controller. The aim of this work is the application of some elements of artificial intelligence in proper fields of predictive control. It is focused especially on the utilization of simple evolutionary algorithms in the optimization process as well as using neural networks as models of the systems. It has been shown that, besides classical optimization algorithms, it is possible to apply simple evolutionary algorithms with similar computational demands depending of the problem type and accordingly setting the algorithm. The process of choosing proper systems with slow dynamics, their derivations of mathematical formulas and the methodology of model creation in the form of scalable neural networks is discussed further. This approach can be convenient for controlling of systems which are difficult to be mathematically described or for systems whose description is not known at all. The possibility of application of these models to real systems, the definition of necessary conditions and the requirements for their applications are discussed as well.

KEYWORDS

Predictive Control, MPC, Discrete Control, Discrete Dynamical Models, Artificial Intelligence, Neural Network, Evolutionary Algorithms, Models, Tanks, Optimization, Quadratic Programming, Constraints.

PODĚKOVÁNÍ

Rád bych touto cestou poděkoval svému školiteli doc. Ing. Marku Kubalčíkovi, Ph.D. za jeho cenné rady a odborné vedení v průběhu celého doktorského studia. Také bych rád poděkoval celému kolektivu Fakulty aplikované informatiky za jejich cenné připomínky a za předání potřebných vědomostí nutných ke splnění požadavků pro dokončení studia. Rád bych také poděkoval doc. Ing. et Ing. Ivo Kuřitkovi, Ph.D. et Ph.D. za podporu a osobní přístup.

Děkuji také za finanční podporu poskytnutou v rámci Interní grantové soutěže Univerzity Tomáše Bati ve Zlíně (IGA) díky které bylo možné publikovat výsledky na hodnocených konferencích. Jedná se o tyto projekty IGA/FAI/2014/009 a IGA/CebiaTech/2015/026.

V neposlední řadě patří mé díky též všem členům mé rodiny za podporu i trpělivost v průběhu řešení úkolů této disertace.

OBSAH

1. ÚVOD

Prediktivní řízení[1-5] je metoda regulace, jejíž hlavní myšlenkou je využití predikce chování systému pro optimalizaci akčního zásahu. Touto metodou se obvykle řídí spojité systémy s využitím diskrétních modelů[6, 7]. Dynamický spojitý model systému je obvykle vyjádřen pomocí diferenciálních rovnic. Matematické operace s touto formou modelů jsou většinou obtížné. Diskrétní modely využívají k popisu dynamického chování systému rovnice diferenční, které pomocí omezeného počtu vzorků minulých hodnot vstupů a výstupů systému vyjadřují aktuální hodnoty výstupu. Rozšířením tohoto modelu o budoucí hodnoty je pak možno získat predikci budoucího chování systému. Tomuto rozšíření se pak říká prediktor[8]. Tvar prediktoru je závislý na velikosti predikčního horizontu. Budoucí chování systému je poté možné optimalizovat tak, aby bylo dosaženo žádané hodnoty. Optimalizátor a prediktor společně tvoří regulátor[5].

Predikce chování vychází z modelu systému, a proto má tento model popisovat chování systému co nejpřesněji. Zároveň je nutné navrhnout regulátor s co nejmenší výpočetní náročností, zejména pro systémy s velmi rychlou dynamikou. Prediktivní řízení je možné nasadit na různé typy systémů, a to i na ty, které jsou jinými metodami obtížně řiditelné. Jedná se o systémy neminimálně fázové, systémy s dopravním zpožděním či systémy nelineární[1-5]. Model systému se získá identifikací soustavy[9], a to buď matematicko-fyzikální analýzou, nebo identifikací experimentální[10, 11]. Pro systémy, jejichž přesný matematický model je obtížné nebo nemožné získat, je možno využít neuronových sítí[12, 13] jako modelu. Tyto modely pak mohou popisovat širokou škálu systémů s různým typem chování, neboť jejich matematický popis je schopen aproximovat širokou škálu matematických funkcí. Nevýhodou těchto modelů bývá jejich velká výpočetní náročnost, proto jsou vhodné hlavně pro systémy s pomalou dynamikou.

V rámci optimalizačního procesu se obvykle využívá kvadratického programování[14-16], neboť optimalizovaný problém je často kvadratického tvaru. Další výhodou je schopnost zahrnutí omezujících podmínek přímo v optimalizačním procesu. Pro řešení optimalizačního procesu je pak možné využít několik druhů algoritmů, kdy je často využito algoritmů simplexových[17, 18]. Kromě klasických metod je možné využít i některé druhy evolučních algoritmů[19] či algoritmů z nich vycházejících[20, 21]. Pokud jsou tyto algoritmy principiálně jednoduché, je možno s nimi dosáhnout výpočetní náročnosti srovnatelné s náročností klasických metod, přičemž tyto některé evoluční algoritmy je možno použít na celou řadu optimalizačních problémů[22].

2. SOUČASNÝ STAV ŘEŠENÉ PROBLEMATIKY

Pro nasazení prediktivního řízení se často využívá diskrétních dynamických modelů, kdy časté je nasazení modelů stavových[9, 23, 24]. Pro popis systémů je často využíváno lineárních modelů[17, 25-28], neboť matematická práce s těmito modely může být jednodušší než u modelů nelineárních. Mnoho prací se zabývá redukcí výpočetní náročnosti[29-33], zejména v oblasti optimalizátoru. Tato redukce dává smysl pro řízení procesů s velmi rychlou dynamikou (např. systémů elektronických). Existují však také systémy, kde jejich pomalá dynamika umožňuje nasazení výpočetně náročnějších metod.

V reálném řízení bývá často vzorkovací perioda fixní. Maximální kritická doba pro vyřešení optimalizačního problému je tedy dána touto vzorkovací periodou. Současná úroveň hardwaru umožňuje navíc řešit komplexní matematické problémy velmi efektivně např. masivní paralelizací[34-36] a použitím grafických procesorů[37], a to právě v oblasti prediktivního řízení[38-40]. Proto je možné v prediktivním řízení využívat i jinak velmi výpočetně náročných metod a prvků jako jsou prvky umělé inteligence[41, 42].

Prediktivní řízení lze tedy v současnosti úspěšně nasadit i na řízení nelineárních systémů[43, 44], dokonce je to výhodné, ovšem v rámci omezení uvedených výše a v oblastech, kde se jinak aplikace klasických metod optimalizace setkávají s obtížemi. Ve specifickém případě prvků umělé inteligence, např. evolučních algoritmů se pak nabízí využití takových algoritmů, jejichž výpočetní náročnost je srovnatelně nízká v porovnání s klasickými metodami[20, 45, 46]. Proto by měly být tyto algoritmy principiálně co nejjednodušší, a zároveň dostatečně sofistikované pro splnění kritérií kladených na prediktivní řízení, jako je dostatečná přesnost a splnění omezení signálů.

Pro popis nelineárních systémů se často používají modely stavové[44, 47], či částečně linearizované[48-50], popř. je nutné nasazení pokročilých metod řízení, v tomto kontextu zejména adaptivního prediktivního řízení[51-54]. Při využití neuronových sítí jako modelů[55-61] se uplatní jejich schopnosti aproximace široké škály matematických funkcí. Tento prvek umělé inteligence lze tedy využít pro popis různých druhů systémů s nelineární dynamikou, které jsou předmětem této práce.

Jednou z oblastí, kde se prediktivní řízení může vhodně uplatnit, je řízení hladiny nádrží netriviálního typu, tedy reprezentantů nelineárních systémů,

pro které je možné, a za některých podmínek i vhodné, použití prvků umělé inteligence, jak je uvedeno výše.

Regulace průtoků a výšky hladiny v nádržích je klíčová v mnoha nejrůznějších odvětvích. V dosavadní literatuře jsou běžně řešeny problémy řízení hladiny kapaliny v nádržích válcového tvaru (respektive tvaru, při kterém se s výškou hladiny nemění její plocha), kde je převaha klasických metod nepopiratelná – PID (Proporcionálně-Integračně-Derivační) regulátor, adaptivní řízení a prediktivní řízení využívající lineární (nebo linearizované) modely, a problém spíše spočívá v jejich kaskádovém, či složitějším zapojení, a současném zvládnutí efektu hydrostatického tlaku, jakožto zdroje nelinearity [62-66].

Zajímavý příklad přístupu k řešení problému tří vzájemně propojených nádrží představuje konstrukce supervizní struktury, která na základě stavů identifikuje ze skupiny validních modů chodu modelu aktuální mód, který pak řídí přiřazeným lineárně kvadratickým regulátorem založeným na off-line optimalizovaném modelu, přičemž model sám je tvořen rovnicemi vycházejícími ze zjednodušeného principiálního popisu systému, a parametry tohoto modelu jsou odhadnuty pomocí identifikace pseudonáhodným binárním signálem [63].

Příkladem využití prediktivního řízení může být regulace hladiny kapaliny v soustavě tří vzájemně propojených válcových nádrží s dvěma vstupy, dvěma výstupy a třemi stavy s aplikací modelů stavových a vstupně-výstupních, kdy linearizace v okolí operačního bodu byla provedena využitím lineárních členů Taylorova rozvoje [62]. Na systému šesti vzájemně propojených nádrží byly demonstrovány možnosti modelování a využití experimentální identifikace modelu pomocí stupňovité (staircase) vstupně-výstupní charakteristiky pro odvození přenosů prvního a druhého řádu [64], a dále použití rozšířeného prediktivního řízení, opět s využitím linearizace pomocí Taylorova rozvoje k získání šesti přenosů [65].

Obsáhlá je studie srovnávající řízení systému tří vzájemně propojených nádrží pomocí tří různých strategií prediktivního řízení s posuvným horizontem využívajících RBF-ARX model (Radial Basis Function – Auto Regressive eXogenous model) s řízením téhož systému pomocí standardního PID regulátoru a tradičního prediktivního řízení s využitím lineárního ARX modelu [66]. První strategie využívá linearizaci RBF-ARX modelu v okolí aktuálního bodu, ve druhé strategii je pro získání vzdálenější predikce výstupu využito budoucích lokálních charakteristik získaných z předchozích výsledků optimalizace a lokální linearizace RBF-ARX modelu, třetí strategie využívá globální nelineární optimalizace RBF-ARX modelu a jeho nelineární

charakteristiky. Jako nejlepší a také klasické metody překonávající se ukázala třetí strategie, s výjimkou případů, kdy by došlo k takovým omezením výstupu, která mohou enormně zvýšit výpočetní náročnost.

Explicitní využití neuronových sítí v prediktivním řízení hladin kapaliny v systému čtyř vzájemně pojených tanků bylo doposud publikováno ojediněle[67], a to pouze jako demonstrace možnosti dosažení exponenciální stability výstupu v prediktivním řízení pomocí rekurentní neuronové sítě za určitých měkkých podmínek.

Na rozdíl od předchozích případů je u nádrží majících tvar, kde se s výškou hladiny mění její plocha, hlavním zdrojem nelinearity sama geometrie nádoby. Dynamika hladiny v takovém typu nádrže je pak vždy nelineární a vyžaduje nasazení alespoň částečně rozvinutějších metod, než jsou klasické metody využívající lineární modely. V současné literatuře lze nalézt příklady řešení regulace nádrží s poměrně omezeným repertoárem tvarů[68-72], vlastně se jedná o kulovou nádrž a o konickou nádrž zužující se směrem dolů (výpusť je na hrotu kuželu).

Relativně velmi jednoduchý přístup byl popsán pro případ nádrže kulového tvaru[71], a sice použití PI regulátoru s interním modelem (Internal Model Control, IMC) s využitím Skogestadovy metody určení parametrů, který byl srovnán s PI modelem s využitím klasické Ziegler-Nicholasovy metody určení parametrů. Při použití IMC PI regulátoru bylo dosaženo menšího překmitu a kratší doby ustálení ve čtyřech reprezentativních pracovních bodech, než při použití PI (Ziegler-Nicholas) regulátoru.

Jiným přístupem k řízení hladiny v nádrži kulového tvaru [72] bylo využití klasického PID regulátoru, jehož parametry autoři určovali pomocí evolučního algoritmu rojení částic (Kennedy–Eberhart Particle Swarm Optimization, PSO), a pak také řadou klasických metod (Ziegler–Nichols method, Cohen–Coon method, Shinskey tuning method, Maclaurin tuning method, Connel tuning method, Astrom–Hagglund tuning method). Srovnáním kvality regulace ve zvoleném pracovním bodě autoři zjistili jako nejvhodnější metodu pro určení parametrů PSO, s jejichž odhadem PID regulátor nepřekmitl a chybové kritérium (Integral of Absolute Error, IAE) bylo nejnižší. Nevýhodou tohoto přístupu je ovšem platnost výsledku pouze pro okolí pracovního bodu zvoleného pro test.

Nádrže konického tvaru představují další typický nelineární systém vyskytující se v praxi, u kterého se mění plocha hladiny s její měnící se výškou. Srovnání různých regulátorů nasazených na systém nádrže kuželového tvaru reprezentovaný modelem prvního řádu s dopravním

zpožděním identifikovaným po částech v několika pracovních oblastech [69] ukazuje, že regulace pomocí prediktivního řízení dosahuje řádově lepších výsledků v hodnotících parametrech kvality regulace, konkrétně v čase dosažení žádané hodnoty a době ustálení, zatímco v parametru překmitu je velice mírně horší, než DSPI regulátor (Direct Synthesis PI), který překmit nemá vůbec. Lepší výsledek byl pro stejný typ nádrže nalezen při srovnání[70] klasického PI regulátoru a prediktivního řízení s využitím seřizování (tuning) parametrů regulátoru pomocí Shridhar-Cooperovy metody[73], kdy prediktivní řízení nevykazovalo ani překmit.

Dalším příkladem možností regulace hladiny v nádrži konického tvaru[68] je srovnání aplikace prediktivního řízení s využitím linearizovaného modelu a nelineárního prediktivního řízení využívajícího Kálmánova filtru modelu, přičemž nelineární řízení bylo vyhodnoceno jako úspěšnější, a to jak ve smyslu dosahování žádaných hodnot, tak i ve smyslu odolnosti vůči poruchám.

I pro řízení hladiny v nádržích konického tvaru doposud není dostupná seriózní literatura věnovaná použití neuronových sítí, při podrobné rešerši bylo pouze nalezeno krátké srovnání[74] regulace hladiny v systému dvou vzájemně spojených nádrží jednak pomocí PID regulátoru ovládaného naprogramovanou strukturou, rozhodující o použití parametrů regulátoru v příslušné části pracovní oblasti, pro kterou byly nalezeny pomocí Cohen-Coonovy metody, a dále pomocí neuronové sítě natrénované s pomocí back-propagation metody. Autoři se v závěru vyslovili pro lepší funkci neuronové sítě jako regulátoru.

Souhrnem literární rešerše je možné uvést, že možnost nasazení prediktivního řízení výšky hladiny v nádržích byla již ověřena, a společně s použitím různě sofistikovaných přístupů k nelinearitě systémů bylo prediktivní řízení uplatněno především na systémech nádrží, jejichž horizontální průřez se s výškou nemění. Za jistý vrchol lze považovat použití RBF-ARX modelu v prediktivním řízení, což ukazuje směrem k RBF neuronovým sítím. Na druhou stranu však explicitní použití neuronových sítí v této oblasti bylo zaznamenáno v literatuře zcela ojediněle a bez dostatečně silného závěru. Nádrže jiné geometrie, než s neměnnou plochou hladiny, jsou studovány a v literatuře popisovány podstatně méně často, prakticky se jedná pouze o nádrže kulového a jednoduchého kuželového tvaru. Zde je silným zdrojem nelinearity, a s ní spojených obtíží, již sám tvar, zatímco hydrostatický tlak je jen malým přispěvatelem. U nádrží kulového tvaru byly v literatuře nalezeny popisy aplikace regulátorů odvozených od PID s limitovanými úspěchy. Nasazení prediktivního řízení nebylo zaznamenáno, a z prvků umělé inteligence byl využit algoritmus rojení částic pro odhad

parametrů PID regulátoru. U nádrží konického tvaru byla v literatuře prokázána výhodnost prediktivního řízení ve srovnání s řízením pomocí regulátorů odvozených od PID, zejména v případě nelineárního prediktivního řízení. Ve všech relevantních případech se v uvedené literatuře potvrdil lepší výkon sofistikovanějších metod, avšak za cenu výpočetní náročnosti, což je nejpravděpodobnější příčinou doposud nedostatečného množství publikací, věnovaných aplikaci prvků umělé inteligence v prediktivním řízení nelineárních systémů reprezentovaných nádržemi netriviálních tvarů. Obtíže se zvládnutím nelinearity pravděpodobně limitovaly i dosavadní snahy jiných autorů o řešení problematiky spojené s řízením hladiny v nádržích dalších tvarů (mimo kouli a kužel), o kterých se v literatuře nedá najít prakticky ani zmínka. Dalším významným deficitem relevantní oblasti literatury je omezenost všech publikací na konkrétní kazuistiky, vždy se jedná o řízení konkrétních systémů, ať už reálných nebo modelových, takže čtenář může využít publikované výsledky na základě analogie jako inspiraci pro svá vlastní řešení, avšak obecné výsledky schází.

Předmětem výzkumu v této práci bude tedy nasazení prvků umělé inteligence v některých částech prediktivního řízení, kdy nasazení evolučních algoritmů bude provedeno alespoň na dílčím systému, a tento algoritmus bude porovnán s klasickými metodami. Dále budou vytvořeny modelové systémy nádrží s různými geometriemi, které odpovídají zobecněním nejčastějších tvarů reálných systémů vyskytujících se v praxi nebo v přírodě. Pro aplikovatelnost a obecnost je nutné odvození modelů klasickým postupem s non-dimenzionalizací veličin[75-77]. Současně bude pominut efekt hydrostatického tlaku, který je stejný v nádrži jakéhokoliv tvaru a závisí jen a pouze na výšce, a je zpětně implementovatelný do regulátoru při přechodu na reálný systém. Pro tyto modelové systémy budou vytvořeny obecné modely neuronových sítí, které budou využity v prediktoru regulátoru. Takto získané modely a regulátory by tedy měly být škálovatelné a jejich možnosti nasazení v konkrétní aplikaci široké, vždy pro reálnou nádrž daného typu tvaru. Eventuální dopad výpočetní náročnosti při online optimalizaci se vztahuje ke vzorkovací periodě, tj. ve vztahu k dynamice řízeného systému.

3. TEORETICKÁ ČÁST A ZVOLENÉ METODY

3.1 Prediktivní řízení

Prediktivní řízení je metoda regulace, založená na jednoduché myšlence, že lze v rámci aktuální periody řízení provést predikci chování systému, a tuto predikci poté optimalizovat tak, aby bylo dosaženo žádaného stavu systému, a případně splněny další požadavky řízení.

Metoda MPC[1-5] (Model Predictive Control) využívá pro predikci chování systému jeho modelu. Pro dosažení co nejlepšího řízení je tedy nutné, aby vybraný model co nejlépe popisoval chování řízeného systému, a proto je potřeba provést jeho výběr, identifikaci systému a určení parametrů modelu co nejlépe.

Regulátor[8] je pak založen na využití prediktoru a optimalizátoru, kdy prediktor využívá získaného modelu systému pro predikci výstupních a řídicích veličin (případně veličin stavových) a optimalizátor poté provádí optimalizaci predikce tak, aby bylo dosaženo žádané hodnoty a dalších požadovaných vlastností pro řízení systému.

Hlavní výhodou prediktivního řízení je jeho schopnost vypořádat se s omezeními[4, 14-16] akčních, řízených i stavových veličin. Další výhodou je jeho aplikovatelnost na systémy, které jsou jinými metodami obtížně řiditelné. Jedná se zejména o mnohorozměrové systémy, systémy neminimálně fázové či systémy s dopravním zpožděním. Nevýhodami jsou větší výpočetní náročnost a nutnost návrhu vhodného prediktoru a optimalizátoru.

3.1.1 Modely

Jak již bylo zmíněno v úvodu teoretické části, je vhodná volba modelu pro návrh regulátoru klíčová. V další části se budeme zabývat pouze modely matematickými, zejména modely vstupně-výstupními[3, 17, 26]. Pro získání modelu mohou být použity dva přístupy: matematicko-fyzikální analýza a experimentální identifikace.

Při matematicko-fyzikální analýze je chování systému popsáno pomocí rovnic založených na fyzikálních a chemických zákonech, zejména bilanci hmoty a energie. Využitím tohoto přístupu je poté získán matematický popis systému ve formě algebraických (statické systémy) či diferenciálních rovnic (dynamické systémy). Takový model poté může popisovat chování systému velmi přesně i v nestandardních situacích, neboť jeho koeficienty mají fyzikální význam. Nevýhodou pak může být práce s těmito modely v rámci

simulace, kdy nalezení analytického řešení může být obtížné, a proto se tyto výpočty často provádějí numerickými metodami[78, 79] (např. metody Runge-Kutta, ode45, atd.).

Při experimentálním přístupu jsou využívány informace o zkoumaném systému získané během experimentu, zejména naměřené hodnoty vstupních a výstupních signálů. Vnitřní struktura systému se nebere v úvahu a systém je považován za „černou skříňku“. Na základě naměřených vstupních a výstupních signálů je zkoumána vstupně-výstupní relace a je určen příslušný matematický model, ovšem koeficienty modelu nemají fyzikální význam, a práce s těmito modely v rámci simulace může být jednodušší. Pokud model kombinuje data a alespoň částečný fyzikální model, je považován za tzv. „grey box“,[28] čistě teoretický model je pak tzv. „white box“.[26, 80]

3.1.2 Lineární modely

V další části budou uvedeny některé lineární modely[17, 26, 81-83], které se pro prediktivní řízení systémů používají. Lineární systémy jsou lineární v tom smyslu, že v rámci své pracovní oblasti nemění své ustálené vlastnosti. Budeme se také zabývat pouze modely diskrétními, neboť tyto se obvykle používají pro popis spojitých systémů v rámci prediktivního řízení.

Impulsní funkce

Impulsní funkce systému popisuje reakci diskrétního systému na jednotkový (Diracův) impuls. Výstupní hodnota diskrétního systému je určena následujícím vzorcem:

$$y(k) = h(k) * u(k) = \sum_{i=1}^{N} h_i u(k-i) = H(z^{-1})u(k) \tag{3.1}$$

kde $H(z^{-1}) = h_1 z^{-1} + h_2 z^{-2} + \cdots + h_N z^{-N}$.

Nevýhodou tohoto modelu je, že může popisovat pouze stabilní procesy a hodnota *N* musí být vysoká. Tento model, označený jako FIR (Finite Impulse Response), se často používá pro popis filtrů.

Přechodová funkce

Přechodová funkce vyjadřuje odezvu systému na jednotkový (Heavisideův) skok. Výstup systému se dá vyjádřit ve vztahu k přírůstkům akčního zásahu:

$$\begin{gathered} y(k) = \Delta g(k) * u(k) \\ y(k) = \sum_{i=1}^{N} g_i \Delta u(k-i) = G(z^{-1})(1-z^{-1})u(k) \end{gathered} \tag{3.2}$$

Nevýhody tohoto modelu jsou stejné, jako u modelu impulsní odezvy.

Přenos

Model systému je popsán podílem polynomů. Přenos systému lze zapsat ve tvaru:

$$G(z^{-1}) = \frac{Y(z^{-1})}{U(z^{-1})} = \frac{B(z^{-1})}{A(z^{-1})} \tag{3.3}$$

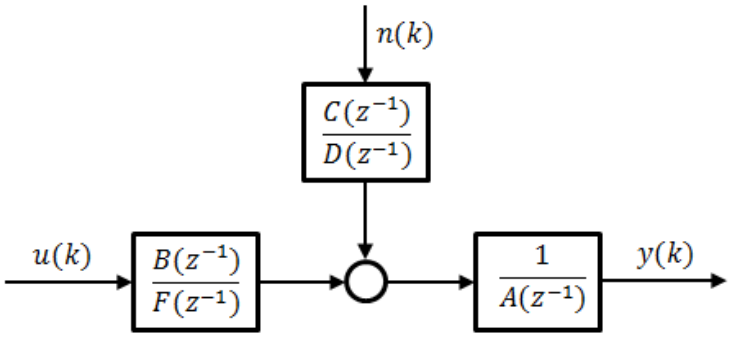
Výstup systému tedy lze zapsat jako:

$$y(k) = \frac{B(z^{-1})}{A(z^{-1})} u(k) \tag{3.4}$$

Tento systém popisuje i nestabilní procesy. Je však třeba znát polynomy $A(z^{-1})$ a $B(z^{-1})$.

Obecné lineární modely

Lineární modely jsou modely popsány polynomy $A(z^{-1})$, $B(z^{-1})$, $C(z^{-1})$, $D(z^{-1})$ a $F(z^{-1})$ a zahrnují kromě vstupu také poruchu systému. Obecné uspořádání těchto modelů je zobrazeno na Obr. 3.1.



Obr. 3.1: Obecný lineární model, podle ref. [26, 28]

kde $n(k)$ je bílý šum.

Pro ARMAX[26] (AutoRegressive-Moving-Average model with eXogenous input) model platí $D(z^{-1}) = F(z^{-1}) = 1$. Odezvu systému tedy můžeme zapsat jako

$$y(k) = \frac{B(z^{-1})}{A(z^{-1})} u(k) + \frac{C(z^{-1})}{A(z^{-1})} n(k) \tag{3.5}$$

Rovnice se dá zapsat také v následujícím tvaru (pro přehlednost nebude v dalším zápise uveden operátor z^{-1})

$$Ay(k) = Bu(k) + Cn(k) \tag{3.6}$$

CARIMA (Controlled AutoRegressive Integrated Moving Average)

Vyjdeme-li z ARMAX modelu a zahrneme-li do této rovnice $\Delta = 1 - z^{-1}$ (ΔC zapíšeme opět jako C), dostaneme rovnici CARIMA modelu[3, 26, 84].

$$\Delta Ay(k) = B\Delta u(k) + Cn(k) \tag{3.7}$$

Pro výpočet prediktoru však není vhodné uvádět rovnici v tomto tvaru, neboť $u(k)$ je neznámá veličina, kterou je nutné predikovat. Je proto vhodné použít přímo $u(k-1)$ a polynom B zapsat s nenulovým absolutním koeficientem.

$$Ay(k) = Bu(k-1) + \frac{C}{\Delta}n(k) \tag{3.8}$$

3.1.3 Prediktor

Prediktor je nástroj pro předpovídání budoucího chování systému. Délka intervalu predikcí je dána velikostí predikčního horizontu.[1, 3, 4, 85] Uvažujme 2 typy horizontů, kdy výstupní horizont daný intervalem $\langle N_1, N_2 \rangle$ určuje velikost predikce výstupní veličiny y a řídicí horizont N_u určuje velikost vektoru vstupní veličiny u. Je také nutné ještě podotknout, že prediktor slouží pouze k predikci, kdy optimalizace veličin se provádí pomocí optimalizátoru (viz 3.1.4).

Pro ilustraci budeme předpokládat model 2. řádu popsaný diferenční rovnicí

$$y(k) = a_1 y(k-1) + a_2 y(k-2) + b_1 u(k-1) + b_2 u(k-2) \tag{3.9}$$

kde $y(k)$ je výstup modelu, $y(k-1)$ a $y(k-2)$ jsou minulé hodnoty výstupu, $u(k-1)$ a $u(k-2)$ jsou minulé hodnoty vstupu a (a_1, a_2, b_1, b_2) jsou parametry modelu získané experimentální identifikací.

Označíme-li vektor parametrů modelu $\boldsymbol{\Theta}_m$, lze rovnici přepsat následujícím způsobem

$$y(k) = f_m(y(k-1), y(k-2), u(k-1), u(k-2), \boldsymbol{\Theta}_m) \tag{3.10}$$

kde f_m je matematická funkce modelu.

Dále předpokládejme velikost výstupního horizontu roven třem. Rozšířením modelu o budoucí hodnoty je možné získat rovnice prediktoru.

$$\begin{aligned} y(k) &= f_m(y(k-1), y(k-2), u(k-1), u(k-2), \boldsymbol{\Theta}_m) \\ y(k+1) &= f_m(y(k), y(k-1), u(k), u(k-1), \boldsymbol{\Theta}_m) \\ y(k+2) &= f_m(y(k+1), y(k), u(k+1), u(k), \boldsymbol{\Theta}_m) \end{aligned} \tag{3.11}$$

Zavedením vektorů budoucích a minulých hodnot,

$$\begin{aligned} \overleftarrow{\boldsymbol{u}} &= \big(u(k-1), u(k-2)\big) & \vec{\boldsymbol{u}} &= \big(u(k), u(k+1)\big) \\ \overleftarrow{\boldsymbol{y}} &= \big(y(k-1), y(k-2)\big) & \vec{\boldsymbol{y}} &= \big(y(k), y(k+1), y(k+2)\big) \end{aligned} \tag{3.12}$$

kde $\overleftarrow{\boldsymbol{u}}$ a $\overleftarrow{\boldsymbol{y}}$ jsou minulé hodnoty vstupu a výstupu, $\vec{\boldsymbol{u}}$ a $\vec{\boldsymbol{y}}$ jsou hodnoty budoucí (predikované), lze rovnici prediktoru zapsat ve tvaru

$$\vec{\boldsymbol{y}} = f_p\big(\vec{\boldsymbol{u}}, \overleftarrow{\boldsymbol{y}}, \overleftarrow{\boldsymbol{u}}, \boldsymbol{\Theta}_p\big) \tag{3.13}$$

kde f_p je rovnice prediktoru a $\boldsymbol{\Theta}_p$ jsou parametry prediktoru.

3.1.4 Optimalizátor

Optimalizátor slouží k nalezení sub-optimálního řešení v rámci aktuální řídicí periody. Pro tyto účely se využívá prediktoru. V rámci optimalizace je definována účelová funkce[4] obvykle kvadratického charakteru

$$J = (\hat{\boldsymbol{y}} - \vec{\boldsymbol{w}})^T(\hat{\boldsymbol{y}} - \vec{\boldsymbol{w}}) + \lambda \tilde{\boldsymbol{u}}^T \tilde{\boldsymbol{u}} \tag{3.14}$$

kde $\hat{\boldsymbol{y}}$ je výstup prediktoru, $\vec{\boldsymbol{w}}$ je žádaná hodnota v rámci výstupního horizontu, $\tilde{\boldsymbol{u}}$ je vstup prediktoru a λ volný váhový parametr.

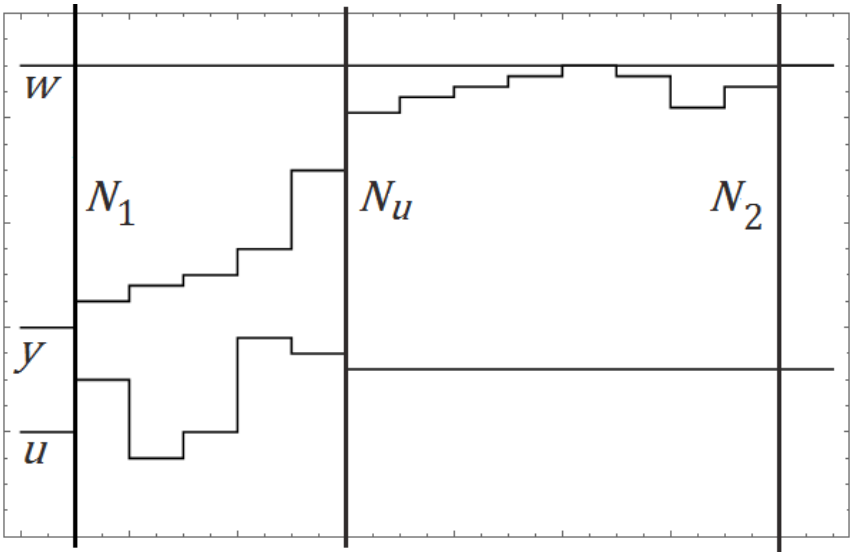
Pro účely aplikace na reálný systém je možné v prediktivním řízení aplikovat ještě různé typy omezení[4, 14, 16] jako např. omezení přírůstků akčního zásahu, omezení hodnot akčního zásahu, omezení výstupů či omezení stavů. Budeme-li předpokládat, že omezení jsou intervalového typu, je možné omezení zapsat jako soustavu nerovnic

$$f_c(\tilde{\boldsymbol{u}}) \leq 0 \tag{3.15}$$

kde f_c je zvolená funkce omezení.

Definice omezení poté klade na optimalizátor další požadavky, kdy je třeba pečlivě vybírat postup optimalizace tak, aby byly splněny požadavky na řízení procesu.

Celý problém optimalizace se řeší pro aktuální periodu řízení. V rámci celého cyklu řízení je aplikován princip posuvného horizontu[2, 86, 87] (Obr. 3.2).



Obr. 3.2: Princip posuvného horizontu, volně podle ref. [2, 86, 87]

Velikost predikce výstupní veličiny je dána výstupním horizontem $\langle N_1, N_2 \rangle$ (možno zahrnout dopravní zpoždění). Z tohoto vyplývá, že žádaná hodnota $\vec{\boldsymbol{w}}$ musí být v rámci tohoto horizontu známá, či alespoň předvídatelná. Velikosti řídicího a výstupního horizontu mohou být odlišné. V takovém případě je zbytek hodnot akčního zásahu nastaven na hodnotu poslední.

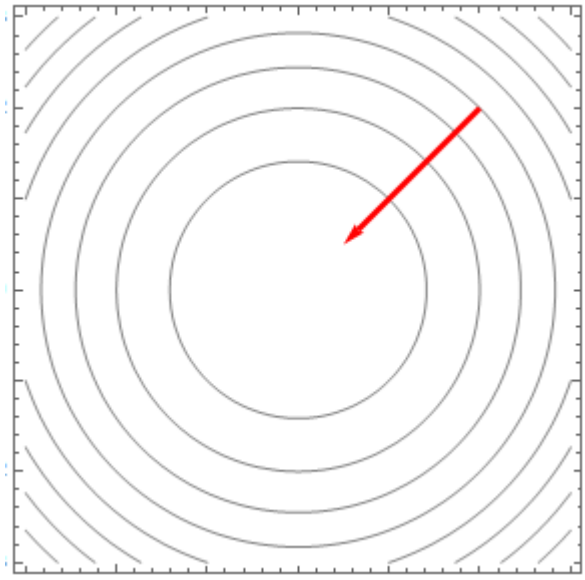
Po provedení sub-optimalizace je obvykle použita pouze první hodnota akčního zásahu platná pro aktuální periodu řízení. V následující periodě jsou horizonty posunuty a celý proces predikce a optimalizace se opakuje.

3.1.5 Metody optimalizace

Jak již bylo výše zmíněno, kvalita regulace je závislá na výsledku optimalizace. Existuje několik optimalizačních algoritmů a přístupů. Problém optimalizace může být vyřešen analyticky či numericky. Analytické řešení je však často neznámé či obtížně dosažitelné. Proto se využívají numerické metody[22, 79], které sice nemusí poskytnout řešení optimální, nicméně i sub-optimální řešení může být natolik dobré, aby bylo dosaženo potřebné kvality regulace.

Jednou ze základních tříd algoritmů jsou algoritmy gradientní[22], které se hodí pro nasazení na funkce unimodální (jeden extrém). Základní myšlenkou

gradientního algoritmu (Obr. 3.3) je fakt, že lokální extrém se nachází ve směru gradientu[1].



Obr. 3.3: Princip gradientního algoritmu, volně podle ref.[22]

Gradientní algoritmus postupuje v jednotlivých iteracích a je možné jej zapsat následujícím způsobem

$$\boldsymbol{u}(n) = \boldsymbol{u}(n-1) + \varepsilon\, \boldsymbol{\nabla} \tag{3.16}$$

kde $\boldsymbol{u}(n-1)$ je předchozí řešení algoritmu, $\boldsymbol{u}(n)$ je nové řešení, $\boldsymbol{\nabla}$ vyjadřuje gradient funkce a ε je velikost kroku (ovlivňuje rychlost a kvalitu optimalizace).

Jednou z vlastností prediktivního řízení je možnost zahrnout omezení v rámci optimalizace. V případě lineárního problému je možné zahrnout tato omezení použitím lineárního programování[16-18, 22]. Jednou z metod lineárního programování je simplexový algoritmus.

Simplexový algoritmus počítá s aplikací omezení, které v rámci řešené hyperfunkce vymezí oblast povolených řešení ve tvaru polytopu. Jelikož řešení jsou obvykle popsána pomocí nerovnic, je hranice tohoto polytopu dána jejich rovností. Optimální řešení se pak nachází v některém z vrcholů tohoto polytopu. Stačí se tedy pouze pohybovat po povrchu tohoto polytopu (resp. jeho vrcholech) a nalézt řešení, které má nejlepší hodnotu účelové funkce. Výhodou tohoto algoritmu je pak to, že prohledává pouze omezené množství řešení úloh.

[1] Pro hledání minima se postupuje ve směru negativního gradientu.

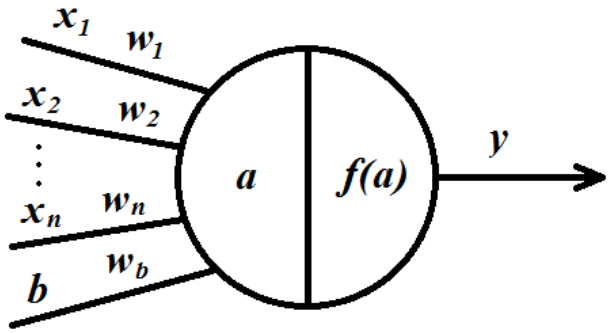
Pro řešení nelineárních problému je pak nutné použít některou z metod nelineárního programování[16, 88]. V prediktivním řízení se pak používá kvadratické programování[14-16], které využívá komplementární algoritmy, založené na principu simplexových algoritmů.

3.2 Umělá inteligence

Pojmem umělá inteligence[13, 42, 89] se označují systémy nebo stroje, které vykazují určitý stupeň inteligentního chování. Často jsou tyto systémy inspirovány přírodními procesy. Obor umělé inteligence (UI) je však velmi široký a proto se tato práce bude zabývat pouze dvěma oblastem UI a to neuronovým sítím[13, 89-91] (NN) a evolučním algoritmům[19-21, 92]. Jedním z důvodů proč se zabývat možností nasazení prvků UI v prediktivním řízení je fakt, že tyto systémy mohou řešit problémy složité či obecné. Jedním z hlavních limitů nasazení UI pak může být často vyšší výpočetní náročnost v porovnání s klasickými metodami. Proto budou zvoleny takové oblasti, které jsou pro toto nasazení vhodné.

3.2.1 Neuronové sítě

Neuronové sítě jsou nástrojem (prvkem) umělé inteligence, jejichž uplatnění lze nalézt v mnoha různých oblastech jako je např. strojový překlad, autonomní systémy či zpracování multimediálních dat. Tyto sítě jsou inspirovány přírodními neuronovými sítěmi, tedy mozky, a jejich hlavní výhodou je jejich schopnost učení se. Neuronové sítě se původně používaly pro klasifikaci (třídění), ovšem lze je uplatnit i na problémy typu aproximace, predikce či rozpoznávání vzorů. Základem neuronové sítě je elementární jednotka zvaná neuron (Obr. 3.4).



Obr. 3.4: Schéma neuronu, volně podle ref.[13, 89]

Neuron obvykle obsahuje množinu vstupů $\boldsymbol{x}$ a jejich příslušných vah $\boldsymbol{w}$. Někdy také mohou obsahovat další vstup b (případně i jeho váhu w_b), kterému se říká práh. Tento práh má v přírodních sítích obvykle excitační či tlumící funkci.

Kombinací vstupů a vah vzniká aktivační funkce a.

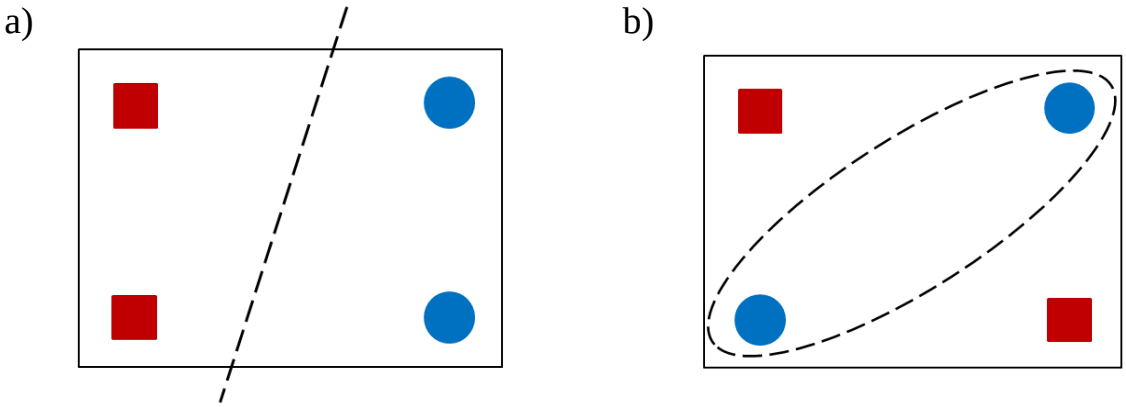
$$a = \sum_{i=1}^{n} x_i\, w_i \quad \text{případně} \quad a = \sum_{i=1}^{n} x_i\, w_i + b\, w_b \tag{3.17}$$

Výstup neuronu y je pak dán přenosovou funkcí $f(a)$, která může být různého typu, v závislosti na zvoleném typu sítě a řešeném problému. Jelikož různé prameny používají různá značení, budeme dále používat pro libovolný vstup symbol u. Rovnice aktivační funkce poté tedy bude nabývat následujícího tvaru.

$$a = \sum_{i=1}^{n} u_i\, w_i \tag{3.18}$$

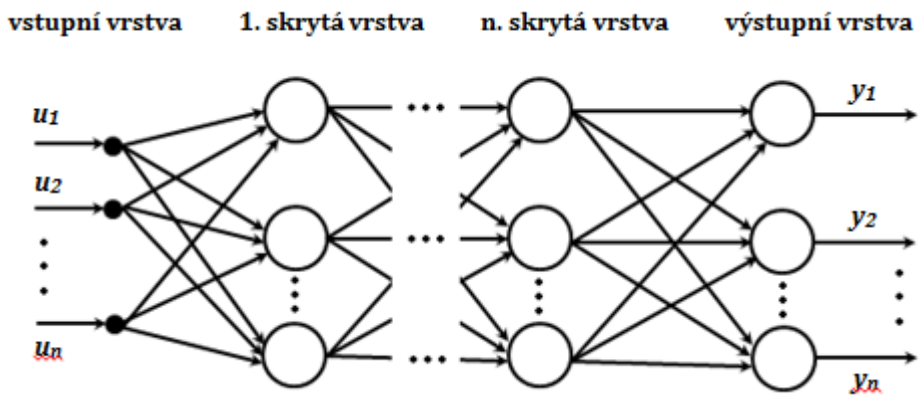
Neuronová síť je pak množina těchto neuronů, které jsou vzájemně propojené v jeden funkční celek. Existují v podstatě dvě třídy neuronových sítí: dopředné a rekurentní[13, 89]. Dopředné sítě jsou takové, kde je množina výstupů zavislá pouze na vstupech sítě. Rekurentní poté obsahují i zpětné vazby, tz. že výstupy neuronové sítě nezávisejí pouze na vstupech, ale i na výstupech sítě.

Jedna z prvních neuronových sítí byla síť Perceptron[13, 89], která se používala pro klasifikaci. Jako přenosovou funkci používá funkci binární (v diskrétní verzi). Tato síť byla tvořena pouze jedním neuronem (Obr. 3.4), nevýhodou této sítě bylo, že uměla řešit pouze lineárně separovatelné problémy (Obr. 3.5).



Obr. 3.5: a) lineárně separabilní problém, b) nelineárně separabilní problém, volně podle ref. [13]

Po jisté době bylo zjištěno, že problém separability se u této sítě dá vyřešit přidáním více neuronů. Tato síť (Obr. 3.6) se nazývá **Multi-Layer Perceptron**[13, 89] (MLP).



Obr. 3.6: Multi-Layer Perceptron, volně podle ref. [89]

Jedná se o dopřednou, vícevrstvou síť, jejíž přenosové funkce jsou (u spojité verze) voleny saturačního typu jako logistická sigmoida či hyperbolický tangens. Při předpokladu jedné skryté vrstvy, a za předpokladu lineární výstupních přenosových funkcí lze síť MLP zapsat následovně

$$\hat{y}_k = \sum_{i=0}^{M} w_i \, \Phi_i \left(\sum_{j=0}^{p} w_{ij} u_j \right) \tag{3.19}$$

kde u_j jsou vstupy, w_{ij} váhy skryté vrstvy, Φ_i zvolená přenosová funkce skryté vrstvy, w_i jsou váhy výstupní vrstvy (dále jen výstupní váhy) a y_k. je výstup k-tého neuronu.

Výhodou této sítě může být relativně dobrá přesnost, nevýhodou pak možná vyšší výpočetní náročnost.

Dalším typem sítě, která může být vhodná pro nasazení v prediktivním řízení, je síť **Radial Basis Function**[89] (RBF). Používají se dva typy funkcí Gaussovská

$$f(x) = e^{-\lambda x^2} \tag{3.20}$$

a Inverzní multi-kvadratická.

$$f(x) = \frac{1}{\sqrt{x^2 + \lambda^2}} \tag{3.21}$$

Skalár x i-tého neuronu skryté vrstvy, je určen jako střední vzdálenost vstupního vektoru od vektoru středů

$$x_i = \|\boldsymbol{u} - \boldsymbol{c_i}\| = \sqrt{(\boldsymbol{u} - \boldsymbol{c_i})^{\mathrm{T}} \boldsymbol{\Sigma_i} (\boldsymbol{u} - \boldsymbol{c_i})} \tag{3.22}$$

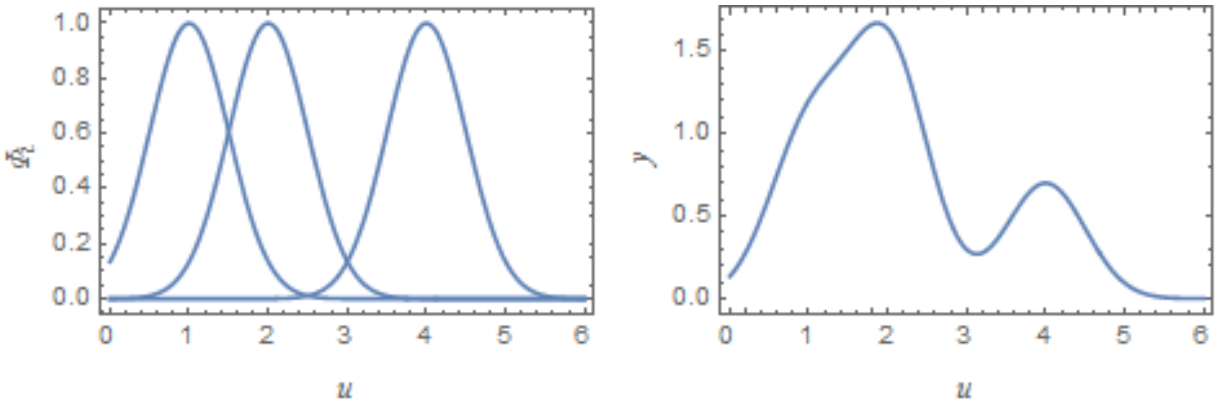
kde $\boldsymbol{u}$ je vektor vstupů, $\boldsymbol{c_i}$ je vektor středů a $\boldsymbol{\Sigma_i}$ je norma.

Struktura neuronové sítě radiální báze je podobná struktuře sítě MLP. Při použití jedné skryté vrstvy s neurony radiální báze a výstupním neuronem s lineární přenosovou funkcí je výstup sítě dán lineární váhovanou kombinací výstupních funkcí neuronů skryté vrstvy. Síť se dá poté zapsat ve tvaru

$$\hat{y}_k = \sum_{i=0}^{M} w_i \, \Phi_i\left(\|\boldsymbol{u} - \boldsymbol{c}_i\|_{\boldsymbol{\Sigma}_i}\right) \tag{3.23}$$

kde Φ_i je funkce radiální báze.

Jednou z výhod této sítě je dobrá interpretovatelnost vlivu[89] nastavení jednotlivých parametrů na výstup této sítě (Obr. 3.7). Pro lepší názornost budeme předpokládat pouze jednorozměrný vstup (a tedy i vektor středů).

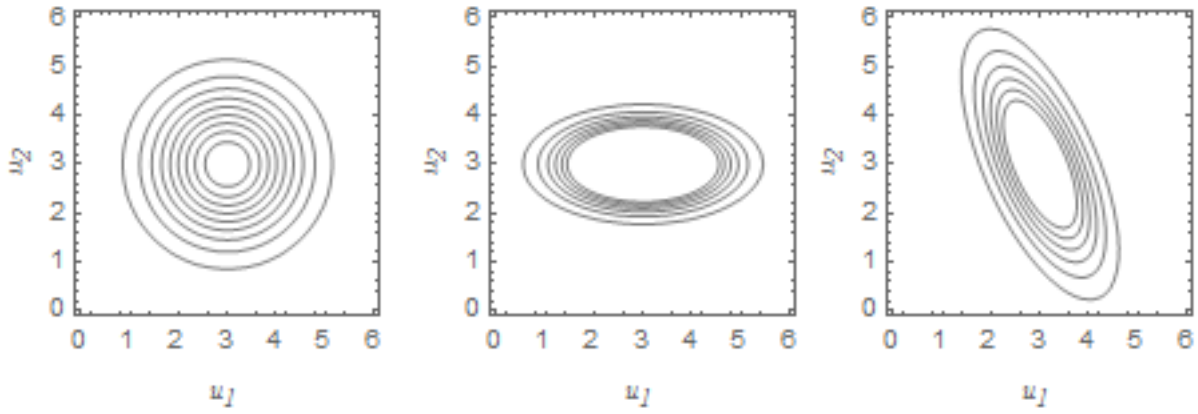


Obr. 3.7: Kombinace přenosových funkcí radiální báze a jejich vliv na výstup sítě, vytvořeno volně podle ref.[89]

Jak je patrné z předchozího obrázku, tak každý neuron skryté vrstvy je jasně definován svým středem, rozptylem (normou) a jeho výstupní váhou. Kombinací těchto neuronů lze tedy provést celkem dobrou aproximaci dat (při splnění určitých podmínek).

Rozptyl jednotlivých neuronů je v N-dimenzionálím prostoru definován normou $\boldsymbol{\Sigma}$. Pro lepší ilustraci je na obrázku zobrazena síť se dvěma vstupy (Obr. 3.8). Jak je patrné, tak i vhodným nastavením normy lze příznivě ovlivnit kvalitu výstupní funkce (aproximace) a zároveň zachovat komplexitu sítě i při relativně nižším počtu neuronů skryté vrstvy. Vždy je však třeba mít na paměti, že každý další parametr zvyšuje dimenzionalitu optimalizačního problému při učení neuronové sítě (viz kapitola 3.2.3).

a) **Euklidovská** (***Σ*** - jednotková) b) **diagonální** (***Σ*** - diagonální) c) **Mahalonobisova** (***Σ*** - obecná)



Obr. 3.8: Vliv normy na tvar přenosové funkce neuronu radiální báze, vytvořeno volně podle ref.[89]

3.2.2 Neuronové sítě: učení

Existují celkem dva typy učení: učení s učitelem a učení bez učitele[13]. Učení s učitelem je typ učení, kde k příslušným vstupům známe i požadované výstupy, kterých chceme dosáhnout. Naproti tomu u učení bez učitele není předem jasné, jaké mají výstupy být.

Učení neuronové sítě obvykle probíhá ve dvou fázích: trénování a testování[13]. Pro tyto účely se používá trénovací a testovací množina. Trénovací množina má velký vliv na kvalitu naučení sítě. Proto by měla obsahovat takové vzory, které jsou dostatečně reprezentativní pro danou třídu.

Jedním z přístupů učení je přírůstková metoda[13]. Principem této metody je průchod trénovací množinou a porovnání výstupu sítě s výstupem předpokládaným. Toto porovnání se zobrazuje chybovou funkcí[13] (někdy také energetickou), přičemž cílem je dosáhnout minimální chyby (energie). Jeden průchod trénovací množinou se nazývá jedna epocha[13]. Váhy se poté mírně upraví

$$w_i(n+1) = w_i(n) + \varepsilon \tag{3.24}$$

kde $w_i(n)$ je původní váha, $w_i(n+1)$ je nová váha a ε je nový přírůstek vah. Tento přírůstek může výrazně ovlivňovat rychlost i kvalitu učení proto je třeba jej volit rozumně.

Obecnějším přístupem je pak metoda **Backpropagation**[13, 89], kde je chyba počítána nejdříve na výstupu sítě, a poté propagována zpět směrem ke vstupu sítě.

3.2.3 Neuronové sítě: učení RBF

RBF síť (3.23) obsahuje množinu parametrů, které lze rozdělit na parametry nelineární: středy a váhy a parametry lineární: výstupní váhy. Určení parametrů sítě je optimalizační problém, kdy je třeba nalézt parametry modelu tak, aby energetická funkce sítě byla minimální. Dimenzionalita optimalizačního problému je v tomto případě taková, kolik je parametrů modelu. Protože nelineární optimalizace[89] může být náročná a často není nalezení optimálního řešení nutné, je často vhodné použít některý postup, který vede k řešení sub-optimálnímu.

Určení středů sítě je typ učení bez učitele, protože není předem dané, jakých hodnot mají středy nabývat. Jelikož vstupem přenosových funkcí radiální báze je skalár x, který vyjadřuje střední vzdálenost vstupů od středů, je možné tyto středy vybrat ze vstupních dat. Základní metodou pro výběr těchto středů je metoda k-means[89], jejíž základní myšlenkou je rozdělení vstupních dat na k oblastí, jejichž data vykazují podobné vlastnosti. Jednotlivé středy jsou poté přiřazeny k nejbližší k-oblasti.

Jednotlivé parametry sítě jsou poté upravovány tak dlouho, dokud není dosaženo ukončovací podmínky.

3.2.4 Evoluční algoritmy

Evoluční algoritmy[19] (EA) je oblast umělé inteligence zabývající se optimalizací[19, 20]. Výhodou těchto algoritmů může být fakt, že mohou řešit komplexní funkce jako např. funkce multimodální, multidimenzionální, multikriteriální či funkce s jistými patologiemi[21]. Nevýhodou pak může být vyšší výpočetní náročnost, v rámci zvoleného algoritmu, jeho nastavení a typu řešeného problému. Proto je nutné velmi pečlivě zvážit vhodnost nasazení těchto algoritmů i jejich výběr.

Evoluční algoritmy jsou inspirovány přírodním procesem evoluce[21] a přirozeného výběru. Základní myšlenkou tohoto procesu je fakt, že lze v rámci určité populace jedinců nalézt jedince, kteří jsou lepší než ostatní, a jejich vzájemnou kombinací a modifikací pak lze sestavit populaci novou[2]. Opakováním tohoto postupu lze pak nalézt řešení nejvíce vyhovující.

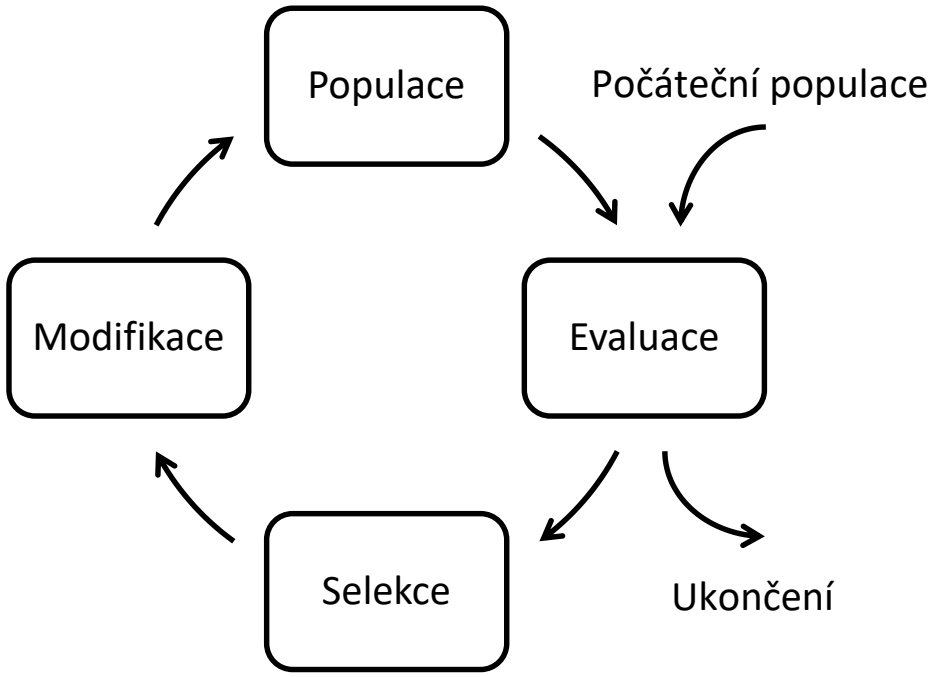
Před pokračováním popisu je vhodné si dále ujasnit některé pojmy. Jedincem se označuje vektor nezávislých proměnných, v našem případě $[u_1, u_2, \dots, u_n]$. Populace označuje množinu těchto jedinců, doplněnou případně i o hodnotu vhodnosti jedince. Vhodnost jedince označuje jeho

[2] Pod pojem evoluční algoritmy zahrnujeme i algoritmy, které pracují i pouze s jedinou populací, kterou postupně modifikují jako např. algoritmy migrační.

kvalitu v rámci řešení optimalizačního problému, kdy kvalita se vyjadřuje buď pomocí cost function (často minimální hodnota je nejlepší) nebo pomocí fitness function (v rozsahu 0-1, kdy 1 je nejlepší) [19, 79, 92].

Evoluční vývoj probíhá v cyklech (zobrazeno na Obr. 3.9).

- *Populace* – sestavení nové populace jedinců,
- *Evaluace* – ohodnocení kvality jedinců populace,
- *Selekce* – výběr nejkvalitnějších jedinců,
- *Modifikace* – úprava či vzájemná kombinace jedinců (např. křížení a mutace).



Obr. 3.9: Evoluční vývoj, nakresleno volně podle ref.[92-94]

Určení počáteční populace jedinců může být náhodné, případně může být určeno pomocí určitých podmínek a předpokladů. Podmínkou ukončení může být např. dosažení určité hodnoty chybové funkce, dosažení minimálního rozdílu zlepšení, či dosažení maximálního počtu iterací (cyklů). Předpokladem je ovšem konvergence algoritmu.

Existuje několik různých druhů algoritmů, přičemž vhodnost jejich nasazení závisí na několika faktorech, jako je např. tvar účelové funkce, omezující podmínky či požadovaná výpočetní náročnost. Obecně lze považovat evoluční algoritmy za výpočetně náročnější oproti klasickým metodám. Jednotlivé algoritmy budou tedy vybrány a aplikovány pouze

tehdy, když to bude vhodné (např. na učení NN) a tedy pouze takové, jejichž výpočetní náročnost je ještě akceptovatelná v rámci prediktivního řízení. Z tohoto důvodu je jejich nasazení možné pouze u procesů s pomalejší dynamikou.

Dále bude uveden stručný (neúplný) přehled algoritmů[92] a jejich stručný popis. Je však třeba mít na paměti, že ne všechny jsou vhodné pro nasazení v prediktivním řízení, kdy jako nevhodné se jeví zejména ty algoritmy, které jsou komplexní a velmi výpočetně náročné.

- *Random walk* – jedná se nejméně sofistikovaný algoritmus, kdy jsou všechny populace generovány náhodně.
- *Hill climbing* – patří do rodiny gradientních algoritmů, vhodný pro unimodální účelové funkce. Základním principem je generování populace vždy kolem nejlepšího jedince. To by mělo zajistit sestup/šplhání ve směru gradientu až k nalezení nejlepšího řešení.
- *Tabu search* – vylepšená verze algoritmu Hill climbing doplněná o krátkodobou a dlouhodobou paměť transformací, což má předejít zacyklení algoritmu.
- *Simulated annealing* – inspirovaná procesem žíhání tuhého tělesa. Základní myšlenkou je umožnit na začátku evolučního procesu pohyb jedinců v širším okolí, přičemž ke konci už je oblast hledání výrazně zúžena. To má pomoci proti uváznutí v lokálních extrémech.
- *Genetické algoritmy* – založená na hledání nejlepšího genu. Modifikace jedinců probíhá pomoci genetické mutace a křížení.
- *Rojení částic* – migrační algoritmus založený na principu chování ptačího hejna. Pohyb jedince je ovlivněn pohybem ostatních jedinců v rámci sledovaného okolí.
- *Ant colony* – založen na principu hledání potravy v mravenčí kolonii. Vhodné pro řešení optimalizačního problému typu obchodní cestující.
- *SOMA* – migrační algoritmus založený na spolupráci skupiny jedinců. Jednotliví jedinci migrují po hyperploše účelové funkce ve směru nejlepšího jedince v populaci. Existuje několik verzí algoritmu.
- *Diferenciální evoluce* – je zde jistá podobnost s genetickými algoritmy. Modifikace jedinců zde probíhá pomocí diferenciální mutace.

Jako vhodné se jeví algoritmy typu *Hill climbing* a jeho modifikace jako *Tabu search* a *Simulated annealing*. Jako velmi nevhodné se jeví algoritmy, které pracují s velkými generacemi či s velkým počtem cyklů jako *SOMA* a *diferenciální evoluce*. Tyto algoritmy jsou vhodnější na řešení patologických funkcí, nicméně mohou být velmi výpočetně náročné. Ostatní algoritmy jsou spíše vhodné pro řešení jiných typů problémů, kdy např. algoritmus *Ant colony* je možné využít pro řešení problému typu obchodní cestující.

4. CÍLE DISERTAČNÍ PRÁCE

Hlavním cílem disertační práce je aplikovat prvky umělé inteligence ve vhodné oblasti prediktivního řízení. Konkrétně se jedná především o neuronové sítě s případným využitím evolučních algoritmů. Jako vhodnou oblast pro jejich nasazení lze považovat prediktivní řízení systémů, které nejsou (ani snadno, ani úplně) popsatelné matematickým modelem, případně obtížně řiditelné pomocí, byť i nelineárních, klasických metod (např. systémy s modely nelineárními, časově variantními, neminimálně fázovými).

Tento hlavní cíl se z hlediska praktické realizace rozpadá do následujících dílčích cílů:

1. Předběžná analýza a ověření použitých prvků umělé inteligence
2. Volba systému pro prediktivní řízení a vytvoření modelového systému
3. Volba a vytvoření modelu
4. Identifikace soustavy a nalezení parametrů modelu
5. Vytvoření prediktoru
6. Optimalizace řízení
7. Návrh přenosu získaných výsledků na reálný systém

4.1 Popis dílčích cílů a způsob jejich realizace

1. Předběžná analýza a ověření použití prvků umělé inteligence

S ohledem na záměr použití prvků umělé inteligence v prediktivním řízení, je jako první bod řešení problému nutno zařadit předběžné ověření a prokázání schůdnosti aplikace zvolených prvků řešení alespoň na dílčím příkladu. Jako první úkol se naskýtá analýza tvaru účelové funkce a vlivu omezení na dosažitelnou oblast. Dalším předběžným úkolem je prokázání možností a limitů nasazení evolučních algoritmů v optimalizační části regulátoru ve srovnání s klasickým algoritmem kvadratického programování. Třetím předběžným úkolem je pak ověření možnosti nasazení neuronových sítí jako modelu systému na dobře známém fyzikálním modelovém systému – svého druhu standardu.

2. Volba systému pro prediktivní řízení a vytvoření modelového systému

Jako vhodný testovací systém pro prediktivní řízení se jeví průtočná nádrž na kapalinu, ve které se reguluje výška hladiny. S ohledem na potřebnou obecnost hledaného řešení a abstrakci od konkrétního fyzikálního modelu z předběžné studie bude zvoleno šest základních geometrií.

Nejjednodušším modelovým systémem (z teoretického i praktického hlediska) je vertikální válcová nádrž. S ohledem na zobecnění modelového systému byly uvažovány další tvary nádrží. Jednalo se o nádrž ve tvaru vertikálně postaveného kuželu. V prvním takovém vhodném případě je dno nádrže tvořeno vrcholem kuželu, a tedy, pokud se nádrž plní konstantním přítokem, rychlost zvedání hladiny se zpomaluje, obdobně, pokud z takové nádrže kapalina konstantním tokem vytéká, pokles hladiny se zrychluje. Obrácenou situaci lze získat v případě kuželové nádrže, kde dno je tvořeno podstavou. V takové nádrži se při rovnoměrném plnění vzestup hladiny zrychluje, a při vypouštění zpomaluje. Kombinací obojího jsou situace s nádrží, která se ve střední výšce zužuje, tedy „přesýpací hodiny“ nebo naopak rozšiřuje – dvoj-kužel nebo koule. V realitě (přírodní i technické) pak můžeme obdobné chování a geometrickou podobnost nalézt v různých rezervoárech kapalin (v průmyslu), vodních a jiných nádržích, odkalištích, poldrech, přehradách, kanálech (umělých i jeskynních systémech) a podobně. Systém s katastrofickým chováním (například nádrž se sifonem) je záměrně pominut. Ve vztahu k fyzikální realitě, kdy by byl nejspíše k regulaci využit výtokový ventil, je efekt hydrostatického tlaku na rychlost výtoku kapaliny z nádrže pominut tím, že akčním zásahem je přímo (od)tok kapaliny z nádrže.

V případě modelových systémů je samozřejmě vhodné získat tento model matematicko-fyzikální analýzou, neboť v tomto případě mají jednotlivé parametry i popis jednoznačný fyzikální (geometrický) význam. Tohoto bude využito pro simulaci modelového systému v prostředí SIMULINK pomocí S-Funkce. Výhodou simulace je možnost provádět virtuální experimenty opakovaně a rychleji než v případě reálného systému nebo v případě, kdy není reálný systém pro experiment či analýzu dostupný.

Pro dosažení škálovatelnosti (tj. v nejjednodušším případě použitelnosti výsledků bez ohledu na měřítko modelu, popřípadě stanovením postupu, jak výsledek získaný na modelu jistého měřítka přenést na model či systém jiného měřítka) získaných výsledků, a tedy i pro přenos na reálný systém, bude nutné modely realizovat pomocí vhodně zvolených bezrozměrných veličin (proměnných). Non-dimenzionalizace vede ke zjednodušení diferenciálních (diferenčních) rovnic odstraněním jednotek fyzikálních veličin vhodnou substitucí proměnných. Velmi účelné je pracovat s veličinami vyjádřenými

relativně vzhledem k určité vhodné jednotce, např. volbou charakteristických hodnot veličin popisujících systém.

Přínos volby systému vyplývá z přenositelnosti zkušeností a poznatků získaných na modelových systémech na reálné systémy, jejichž důležitost je zjevná.

3. Volba a vytvoření modelu

Volba správného modelu závisí na charakteru systému, jenž má model popisovat. Z předchozího bodu vyplývá, že systém nádrže obecného tvaru bude vykazovat nelineární vlastnosti.

Vzhledem k tomu, že reálný systém může být obecného nepravidelného tvaru, vlastně se to přímo očekává, jeví se jako vhodné nasazení modelu neuronové sítě, která má schopnost učení se a nevyžaduje dostupnost matematického popisu modelu. Z neuronových sítí připadají v úvahu 2 typy: multi-layer perceptron a sítě založené na funkcích radiální báze (RBF). Vzhledem k lepší interpretovatelnosti vlivu parametrů na výstupní funkci se jako vhodný kandidát na model jeví druhý případ.

4. Identifikace soustavy a nalezení parametrů modelu

Identifikace soustavy záleží na volbě typu modelu. V případě neuronových sítí probíhá obvykle identifikace modelu pomocí trénovací množiny. Vzhledem k charakteru modelového systému bude zvolena typická vstupně-výstupní sada, která bude získána simulací modelového systému v prostředí SIMULINK. V případě RBF sítě pak existuje několik strategií učení (jako např. náhodné umístění středů, seskupování středů, strategie založené na mřížce či volba na základě podskupiny dat a nelineární optimalizace). Pro učení sítě bude zvolen vhodný přístup a vhodné optimalizační algoritmy a techniky. Vzhledem ke kumulační povaze systému je testování pomocí testovacích množin získaných simulací modelového systému nevhodné. Testování proběhne pomocí testu kvality regulace, což je na druhou stranu výhodné, neboť již v průběhu učení lze vybírat nejlepší kombinace parametrů, optimalizačních algoritmů a technik. Současně tento přístup znamená, že se po vytvoření regulátoru (tj. vytvoření prediktoru a použití optimalizátoru) ve zpětnovazebné smyčce řešení vrací k úkolu 4, a sice k modifikaci trénovací množiny.

5. Vytvoření prediktoru

V případě nasazení RBF sítě jako modelu, lze prediktor vytvořit nejméně dvěma způsoby. Tím prvním je samotné rozšíření modelu přidáním budoucích vstupů a výstupů. Počet vstupů a výstupů je pak dán velikostí řídicího a výstupního horizontu. Výhodou může být zachování relativně nízké výpočetní náročnosti, nevýhodou pak nutnost identifikace prediktoru pomocí trénovacích množin nebo změna struktury (a znovu identifikace) v případě změny velikosti horizontů.

Druhým přístupem je rekurzivní přístup, kdy rovnici modelu rozšíříme o budoucí hodnoty. Výhodou může být snadná změna velikosti horizontů a možnost zachovat původní model (bez nutnosti identifikace prediktoru). Nevýhodou pak může být zvýšená výpočetní náročnost. Vzhledem k pomalé dynamice systému nádrže se jako vhodný jeví druhý přístup.

6. Optimalizace řízení

Po vytvoření prediktoru je nutné zvolit správný optimalizátor. Tvar účelové funkce bude použit jako v řešení prvního dílčího cíle práce, a vliv požadavků řízení vyplývá z omezení veličin. Vhodný optimalizační algoritmus bude určen empiricky zkoušením dostupných reprezentativních typů algoritmů. Poté budou provedeny experimenty s modelovými systémy a vyhodnocena úspěšnost řízení.

7. Návrh přenosu získaných výsledků na reálný systém

Po realizaci předchozích bodů by měla být síť trénována na datech z reálného systému. Podmínkou je možnost získat dostatečné množství dat. Jelikož experimentální práce (realizace fyzikálního systému) překračuje rámec této disertační práce, budou stanovena pravidla a proveden návrh, jak ověřit nebo porovnat řízení reálného systému dosavadními metodami a pomocí nejlepší sítě (sítí). Pokud by byla k dispozici data z vhodného reálného systému, lze toto ověření provést virtuálně.

5. EXPERIMENTÁLNÍ A METODICKÁ ČÁST

Pro dosažení dílčích cílů práce nebylo zapotřebí žádných reálných experimentů.

Byl využit následující software za pomoci využití následující literatury, která má na tyto software vazby. [4, 5, 17, 26, 28, 78, 79, 90]

- Software Matlab/Simulink ve verzi R2012b
- Software Wolfram Mathematica v aktuálních verzích po dobu řešení práce.

1. Úkol: Předběžná analýza a ověření použití prvků umělé inteligence

Pro řešení předběžných úkolů analýzy účelové funkce byl využit vlastní kód v SW Wolfram Mathematica, pro ověření nasazení evolučních algoritmů byla využita funkce quadprog() a dva vlastní kódy s algoritmem typu „hill climbing“ a „evolučně-gradientním“ algoritmem v SW Matlab, a výsledky byly zpracovány v SW Wolfram Mathematica. Pro ověření nasazení neuronové sítě byl vytvořen vlastní kód neuronové sítě v SW Matlab a pro učení sítě byla využita nelineární optimalizace NMinimize() v SW Mathematica.

2. Úkol: Volba systému pro prediktivní řízení a vytvoření modelového systému

Průtočná nádrž na kapalinu byla jako vhodný testovací systém pro prediktivní řízení pojednána pomocí abstrakce a zredukována na základní geometrické tvary – modelové systémy – reprezentující zvolené situace, fyzikální faktory byly taktéž maximálně redukovány s ohledem na potřebnou obecnost hledaného řešení. Vytvoření modelových systémů proběhlo nejprve návrhem na papíru, poté byly vytvořeny jako S-funkce v SW Simulink, ke kterým byly přidány korektivní S-funkce řešící problémy singularit, saturací a podobně, tak aby simulace v SW Simulink poskytovala realistický model v rámci daných omezení a zobecnění.

3. Úkol: Volba a vytvoření modelu

Byl zvolen typ neuronové sítě RBF, typ MLP byl vyloučen. Oproti původnímu záměru vlastní konstrukce „*de novo*“ neuronové sítě typu RBF, bylo s výhodou využito funkce newrb() v SW Matlab. Byla zvolena struktura modelu, počet vstupů a výstupů. Vnitřní struktura modelu je předmětem pozdějších variací, zatímco počet vstupů (4) a výstupů (1) je fixní.

4. Úkol: Identifikace soustavy a nalezení parametrů modelu

Pro identifikaci soustavy a určení parametrů modelu bylo nutno provést časové přeškálování modelových systémů a určení vzorkovací periody. Dále byla provedena úprava modelových systémů (formálně matematicky i v kódu) tak, aby vstupní signál byla akumulace namísto odtoku. Toto další zobecnění je možné při předpokládané znalosti přítoku a systém se tím stává nezávislým na hodnotě přítoku. Nevýhodou je pak ztráta aktuální absolutní informace (integrálního charakteru).

Vzhledem k nevhodnosti využití testovací množiny pro kumulační systém byla trénovací množina testována pomocí kvality regulace pilotních experimentů, a postupně vylepšována ve zpětné vazbě po provedení optimalizace (úkol 6). Počet cyklů zpětné vazby byl <10. Velikost trénovací množiny je cca 12 000 vzorků.

Trénovací množina byla (v každém cyklu zpětné vazby) použita k nalezení parametrů modelu určeného funkcí newrb(), což probíhalo vždy ve dvou fázích. Nejprve byla měněna hodnota spread (rozptyl) až došlo ke konvergenci hodnoty energetické funkce. Onset tohoto ustálení potom určuje minimální efektivní počet neuronů. Poté byl omezen maximální počet neuronů na tuto efektivní hodnotu a byl natrénován RBF model a uložen do souboru.

5. Úkol: Vytvoření prediktoru

Prediktor byl vytvořen pomocí rekurzivního přístupu a byl realizován jako procedura v rámci S-Funkce regulátoru v simulačním prostředí Simulink, Regulátor si soubor s naučenou sítí načte a předává prediktoru jako parametr.

6. Úkol: Optimalizace řízení

Obecně kvadratický tvar účelové funkce byl použit jako v řešení prvního dílčího cíle práce, avšak zde bylo využito funkce fmincon() v SW Matlab, která umoňuje nelineární optimalizaci s omezením veličin. Byly zkoušeny následující optimalizační algoritmy: interior-point, sqp-legacy, active-set a trust-region-reflective, a byl vybrán nejlepší. V průběhu práce byly taktéž upravovány parametry účelové funkce a funkce omezení. Po odladění regulátoru byly s pomocí nejlepšího optimalizačního algoritmu provedeny experimenty s modelovými systémy a vyhodnocena úspěšnost řízení. Pro dosažení shody ustálených hodnot výstupu s žádanou veličinou byl zvolen přístup korekce žádané hodnoty pomocí korekční funkce získané kalibrací žádané hodnoty proti změřenému výstupu. Následně byla korekční funkce

zavedena do regulátoru a provedeny kontrolní simulace, které potvrdily funkčnost navrženého řešení.

7. Úkol: Návrh přenosu získaných výsledků na reálný systém

Data z reálného systému nebyla k dispozici, avšak i přesto bylo možné stanovit postupy a pravidla pro přenos získaných poznatků na reálný systém. Jednak byla využita škálovatelnost modelu, dále byly využity zkušenosti s chováním použitých modelů a procesů, a taktéž bylo postupováno ve směru konkretizace a zpětné extrapolace k fyzikálním vlivům, od nichž se na počátku při volbě modelových systémů abstrahovalo.

6. VÝSLEDKY A DISKUSE

6.1 Předběžná analýza a ověření použití prvků umělé inteligence

6.1.1 Analýza účelové funkce

Prvotní experimenty byly provedeny na modelu popisujícím tepelnou soustavu, která byla inspirována reálným systémem pece.[95] Tento systém byl vybrán pro jeho jednoduchost a snadnou práci v rámci výpočtů a simulací. Tento systém je možné popsat přenosem (rovnice 3.3). Pro účely prediktivního řízení budeme uvažovat CARIMA model[3] (viz kapitola 3.1.1). Diferenční rovnici modelu lze získat např. příkazem c2d (continuous-to-discrete) v matlabu. Velikost periody lze pak získat odhadem, např. experimentálním změřením odezvy systémy na jednotkový skok.

Odvození prediktoru a optimalizátoru CARIMA modelu

Po získání modelu systému, lze přistoupit k odvození prediktoru[8]. Tento pak bude získán metodou rozšíření diferenciálních rovnic o budoucí hodnoty (kapitola 3.1.3). Dosazením do rovnice (3.11) a rozdělením na známé a neznámé členy dostaneme rovnici,

$$\begin{pmatrix} \hat{y}(k+1) \\ \hat{y}(k+2) \\ \hat{y}(k+3) \\ \vdots \end{pmatrix} = \begin{pmatrix} g_0 & 0 & \cdots \\ g_1 & g_0 & \cdots \\ g_2 & g_1 & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix} \begin{pmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \end{pmatrix} + \begin{pmatrix} x_{y11} & x_{y12} & \cdots & x_{y1(n+1)} \\ x_{y21} & x_{y22} & \cdots & x_{y2(n+1)} \\ x_{y31} & x_{y32} & \cdots & x_{y3(n+1)} \\ \vdots & \vdots & \ddots & \vdots \end{pmatrix} \begin{pmatrix} y(k) \\ y(k-1) \\ \vdots \\ y(k-n) \end{pmatrix} + \begin{pmatrix} x_{u11} & x_{u12} & \cdots & x_{u1m} \\ x_{u21} & x_{u22} & \cdots & x_{u2m} \\ x_{u31} & x_{u32} & \cdots & x_{u3m} \\ \vdots & \vdots & \ddots & \vdots \end{pmatrix} \begin{pmatrix} \Delta u(k-1) \\ \Delta u(k-2) \\ \vdots \\ \Delta u(k-m) \end{pmatrix} \tag{6.1}$$

kterou lze pak zapsat v maticové formě.

$$\vec{\boldsymbol{y}} = \boldsymbol{G}\Delta\vec{\boldsymbol{u}} + \boldsymbol{X_y}\overleftarrow{\boldsymbol{y}} + \boldsymbol{X_u}\overleftarrow{\boldsymbol{u}}$$
$$\vec{\boldsymbol{y}} = \boldsymbol{G}\Delta\vec{\boldsymbol{u}} + \boldsymbol{X}\begin{pmatrix} \overleftarrow{\boldsymbol{y}} \\ \overleftarrow{\boldsymbol{u}} \end{pmatrix} \tag{6.2}$$

Matice $\boldsymbol{G}$ pak obsahuje hodnoty přechodové funkce ve tvaru,

$$\boldsymbol{G} = \begin{pmatrix} g_0 & 0 & 0 & \cdots & 0 \\ g_1 & g_0 & 0 & \cdots & 0 \\ g_2 & g_1 & g_0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{N_2-1} & g_{N_2-2} & g_{N_2-3} & \cdots & g_0 \end{pmatrix} \tag{6.3}$$

které je možné získat např. podílem $\boldsymbol{B}/\Delta\boldsymbol{A}$. Velikost matice je pak dána příslušnými horizonty. Matice $\boldsymbol{X}$, stejně jako matice $\boldsymbol{G}$, je získána separací příslušných koeficientů z rovnice prediktoru.

Rovnici prediktoru lze poté zapsat v maticové formě

$$\hat{\boldsymbol{y}} = \boldsymbol{G}\tilde{\boldsymbol{u}} + \boldsymbol{y_0} \tag{6.4}$$

kde $\hat{\boldsymbol{y}}$ je výstup prediktoru, $\tilde{\boldsymbol{u}}$ je vstup prediktoru a $\boldsymbol{y_0}$ je volná odezva systému.

Dále je potřeba odvodit rovnici účelové funkce. Dosazením (6.4) do (3.14) a jejich odvozením dostaneme

$$\begin{gathered} J = (\boldsymbol{G}\tilde{\boldsymbol{u}} + \boldsymbol{y_0} - \boldsymbol{w})^T(\boldsymbol{G}\tilde{\boldsymbol{u}} + \boldsymbol{y_0} - \boldsymbol{w}) + \lambda\tilde{\boldsymbol{u}}^T\tilde{\boldsymbol{u}} \\ J = c_0 + 2\boldsymbol{g}^T\tilde{\boldsymbol{u}} + \tilde{\boldsymbol{u}}^T\boldsymbol{H}\tilde{\boldsymbol{u}} \end{gathered} \tag{6.5}$$

kde $\boldsymbol{g}$ je gradient účelové funkce a $\boldsymbol{H}$ je Hessova matice, které je možné získat jako

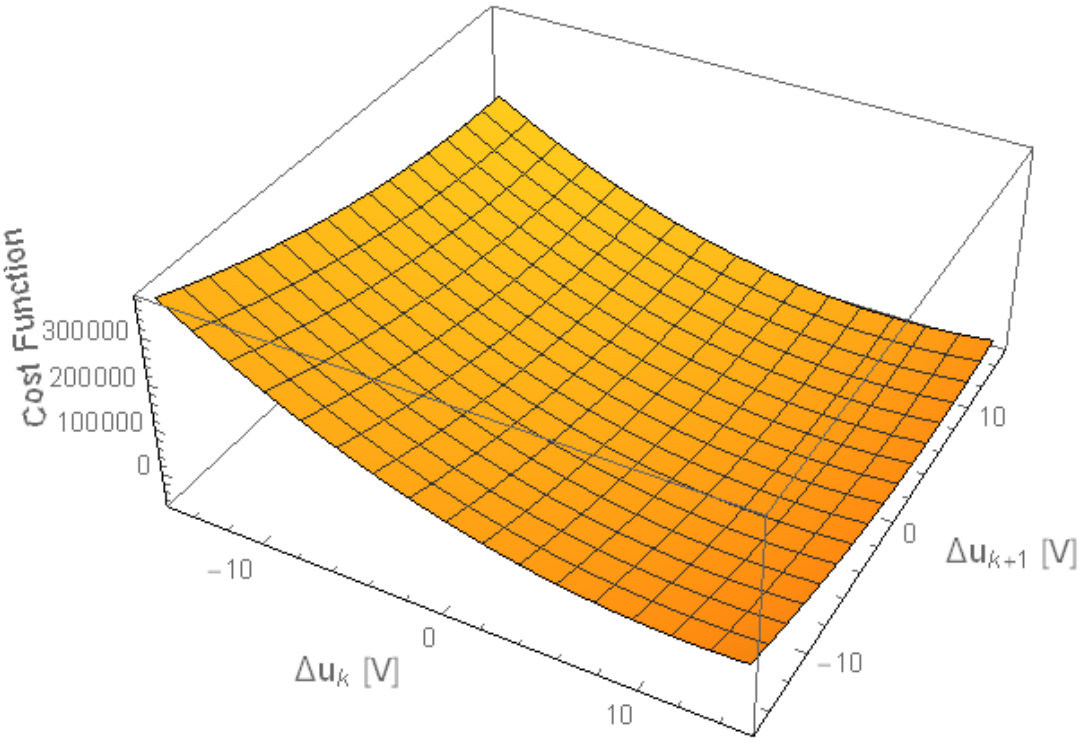
$$\begin{gathered} \boldsymbol{g}^T = \boldsymbol{G}^T(\boldsymbol{y_0} - \boldsymbol{w}) \\ \boldsymbol{H} = \boldsymbol{G}^T\boldsymbol{G} + \boldsymbol{\lambda I} \end{gathered} \tag{6.6}$$

Známe-li analytický tvar, je možné derivací účelové funkce získat analytické minimum, které se nachází v bodě

$$\begin{gathered} \tilde{\boldsymbol{u}} = -\boldsymbol{H}^{-1}\boldsymbol{g} \\ \tilde{\boldsymbol{u}} = -(\boldsymbol{G}^T\boldsymbol{G} + \lambda\boldsymbol{I})^{-1}\boldsymbol{G}^T(\boldsymbol{y_0} - \boldsymbol{w}) \\ \tilde{\boldsymbol{u}} = \boldsymbol{K}(\boldsymbol{w} - \boldsymbol{y_0}) \end{gathered} \tag{6.7}$$

kde $\boldsymbol{K} = (\boldsymbol{G}^T\boldsymbol{G} + \boldsymbol{\lambda I})^{-1}\boldsymbol{G}^T$.

Po výpočtech a jejich aplikaci lze provést simulaci prediktivního řízení a analyzovat tvar účelové funkce pro zvolenou periodu řízení (Obr. 6.1).



Obr. 6.1: Účelová funkce optimalizátoru pro CARIMA model, vlastní zdroj[95].

Jak je patrné z obrázku výsledná účelová funkce je konvexní a unimodální. Bez aplikace omezení tedy existuje pouze jedno optimální řešení, které lze získat analytickým řešením. Z numerických metod pak mohou být vhodné metody gradientní.

Analýza omezení

Předchozí případ účelové funkce a jejího řešení předpokládal ideální stav, kdy jednotlivé veličiny nejsou ničím omezené. Pro aplikaci prediktivního řízení na systémy reálné je ale nutné tato omezení brát v úvahu. Existuje několik typů omezení, pro další odvození však budeme předpokládat pouze omezení přírůstků vstupu (akčního zásahu), omezení hodnoty vstupu a omezení hodnoty výstupu. Všechna omezení pak budou zadána jako spojitý interval, takže omezené veličiny mohou nabývat spojitých hodnot mezi minimální a maximální hodnotou. Omezení mohou být proto zapsána ve formě nerovnic

$$\begin{aligned} \Delta\boldsymbol{u}_{\boldsymbol{min}} \leq \Delta\boldsymbol{u} \leq \Delta\boldsymbol{u}_{\boldsymbol{max}} \\ \boldsymbol{u}_{\boldsymbol{min}} \leq \boldsymbol{u} \leq \boldsymbol{u}_{\boldsymbol{max}} \\ \boldsymbol{y}_{\boldsymbol{min}} \leq \boldsymbol{y} \leq \boldsymbol{y}_{\boldsymbol{max}} \end{aligned} \tag{6.8}$$

kde vektory $\Delta\boldsymbol{u}_{\boldsymbol{min}}$, $\Delta\boldsymbol{u}_{\boldsymbol{max}}$, $\boldsymbol{u}_{\boldsymbol{min}}$, $\boldsymbol{u}_{\boldsymbol{max}}$, $\boldsymbol{y}_{\boldsymbol{min}}$, $\boldsymbol{y}_{\boldsymbol{max}}$ udávají minimální a maximální povolené hodnoty přírůstků vstupu, vstupu a výstupu.

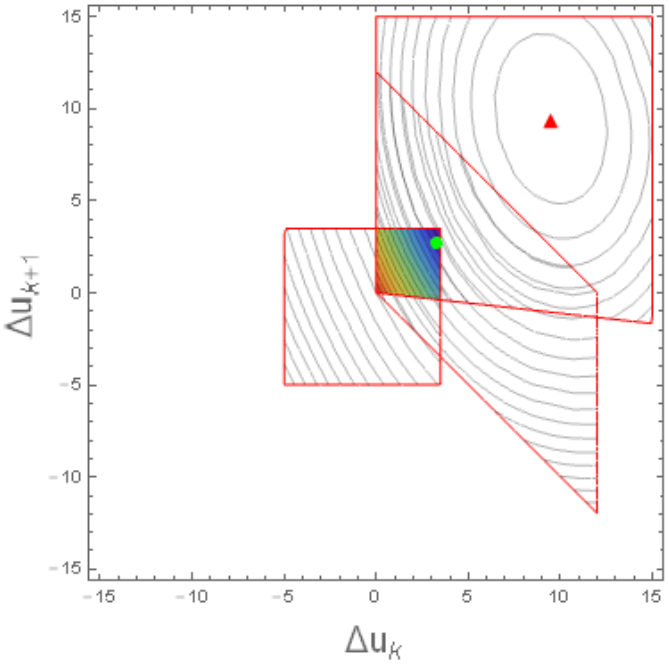
Tyto nerovnice lze poté přepsat do maticové formy

$$\begin{pmatrix} \boldsymbol{I} \\ \boldsymbol{-I} \\ \boldsymbol{T} \\ \boldsymbol{-T} \\ \boldsymbol{G} \\ \boldsymbol{-G} \end{pmatrix} \boldsymbol{\Delta u} \geq \begin{pmatrix} \boldsymbol{1\Delta u_{min}} \\ \boldsymbol{-1\Delta u_{max}} \\ \boldsymbol{1u_{min} - 1u(k-1)} \\ \boldsymbol{-1u_{max} + 1u(k-1)} \\ \boldsymbol{1y_{min} - y_0} \\ \boldsymbol{-1y_{max} + y_0} \end{pmatrix} \tag{6.9}$$

$$\boldsymbol{A\Delta u \geq b}$$

kde $\boldsymbol{I}$ je jednotková matice o rozměrech $N_u \times N_u$, $\boldsymbol{T}$ je dolní trojúhelníková matice s jedničkami na nenulových pozicích a $\boldsymbol{G}$ je matice obsahující koeficienty přechodové charakteristiky.

Po odvození jednotlivých tvarů a jejich aplikací při simulaci systému pak lze zobrazit vliv omezení (Obr. 6.2) na tvar účelové funkce. Pro lepší ilustraci budeme předpokládat řídicí horizont $N_u = 2$.



Obr. 6.2: Účelová funkce – dosažitelná oblast, vlastní zdroj[95].

Jak je patrné z obrázku tak aplikací omezení je v rámci účelové funkce vymezena oblast povolených řešení, kterou budeme dále označovat termínem dosažitelná oblast. Tato oblast je dána průnikem povolených řešení v rámci jednotlivých typů omezení a dohromady tvoří na $(N_u + 1)$ dimenzionální hyperploše účelové funkce N_u-dimenzionální polytop.

V takovém případě mohou nastat dvě situace, kdy je analytické řešení uvnitř dosažitelné oblasti, nebo jak je zobrazeno výše, nachází se mimo tuto oblast. V takovém případě je pak nutné použít některou z numerických metod.

6.1.2 Ověření možnosti nasazení evolučních algoritmů

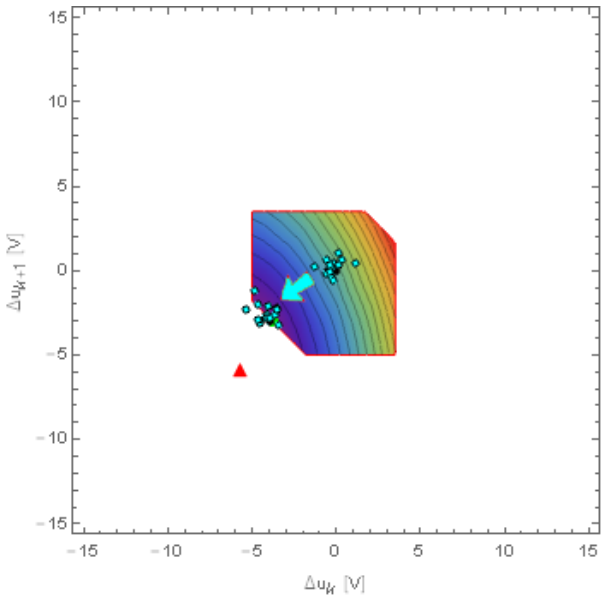
U stejného modelu jako v předchozí kapitole provedeme ověření možnosti nasazení evolučního algoritmu jako optimalizátoru.[96] Tento modelový systém je vybrán z důvodu snadné porovnatelnosti. Jako možný kandidát pro řešení optimalizační problému byl vybrán algoritmus Hill climbing (kapitola 3.2.4), který by měl být dostatečně jednoduchý pro dosažení co možná nejmenší výpočetní náročnosti. Opět si připomeneme názvosloví pro evoluční algoritmy.

- *Jedinec* - konkrétní hodnota nezávislé proměnné (vstupu),
- *Populace* - množina jedinců,
- *Leader* - jedinec s nejmenší hodnotou závislé proměnné (výstupu)

Princip algoritmu lze popsat následovně

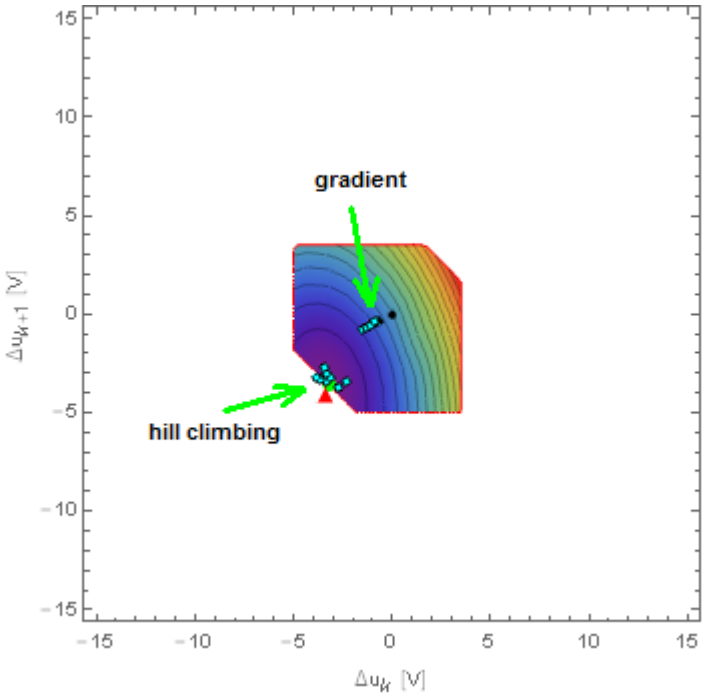
1. Určení počátečního jedince (zároveň leadera).
2. Vygenerování nové populace v okolí aktuálního leadera.
3. Vyhodnocení populace a určení nového leadera.
4. Opakování od kroku 2 dokud není splněna podmínka zastavení.

V případě omezení je ale nutné vyřešit několik problémů, např. určení prvního jedince, nastavení parametrů algoritmu (velikost okolí, velikost populace, podmínka zastavení), chování mimo a na hranici dosažitelné oblasti (soft/hard constraints). Výsledný algoritmus je možné zobrazit následovně.



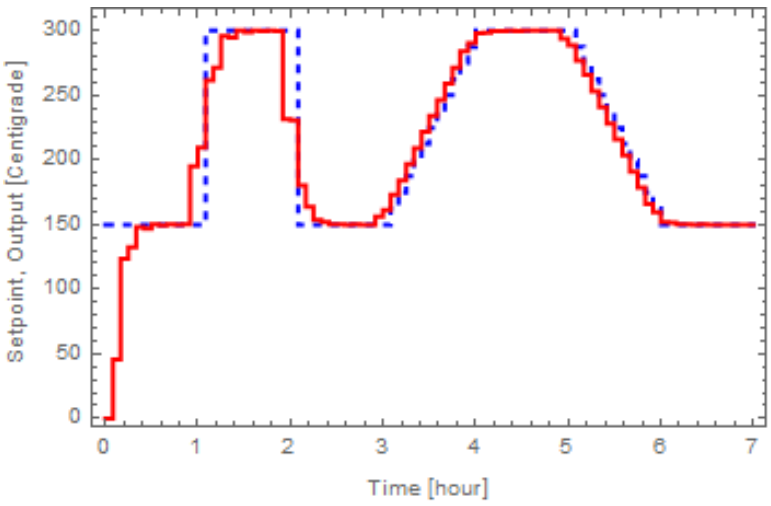
Obr. 6.3: Hill climbing v prediktivním řízení, vlastní zdroj[96].

Dále bylo navrženo několik modifikací, které by mohly vést k lepším výsledkům z hlediska výpočetní náročnosti. Protože je gradient (i Hessián) znám, je možné využít tohoto gradientu a postupovat ve směru nejlepšího řešení. Po dosažení hranice je pak opět využita základní verze Hill climbing. Algoritmus lze pak graficky znázornit, viz Obr. 6.4.



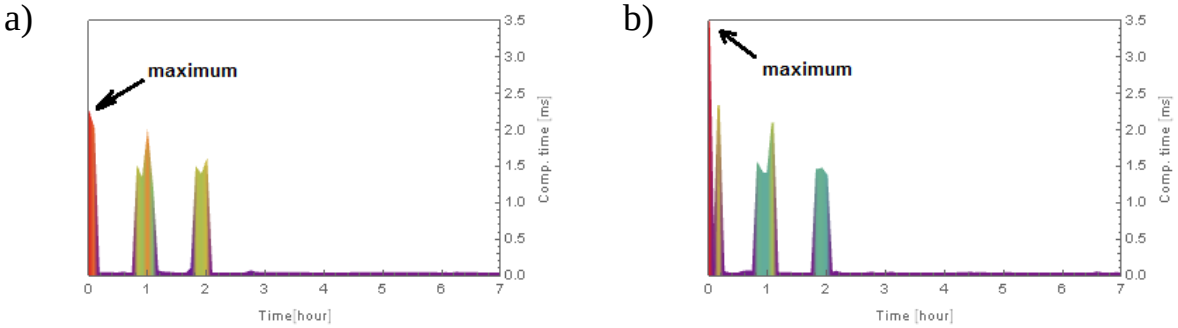
Obr. 6.4: Evolučně-gradientní algoritmus, vlastní zdroj[96]

Výsledek řízení pomocí Evolučně-gradientního (EG) algoritmu je zobrazen na Obr. 6.5.



Obr. 6.5: Řízení evolučně-gradientním algoritmem, vlastní zdroj[96]

Řízení pomocí EG algoritmu a pomocí vestavěné funkce quadprog (QP) v Matlabu je pak porovnáno z hlediska výpočetního času, potřebného pro optimalizaci v daný časový okamžik, viz Obr. 6.6.



Obr. 6.6: Časový průběh výpočetního času optimalizace: a) EG algoritmus, b) QP algoritmus, vlastní zdroj[96]

Z časových průběhů je patrné, že maximální výpočetní čas EG algoritmu byl v tomto případě o něco málo lepší, než maximální výpočetní čas algoritmu kvadratického programování. Dále je patrné, že největší výpočetní náročnosti je dosaženo v oblastech s velkou změnou žádané hodnoty (největší odchylky od výstupní veličiny). V ostatních případech je použito analytické řešení.

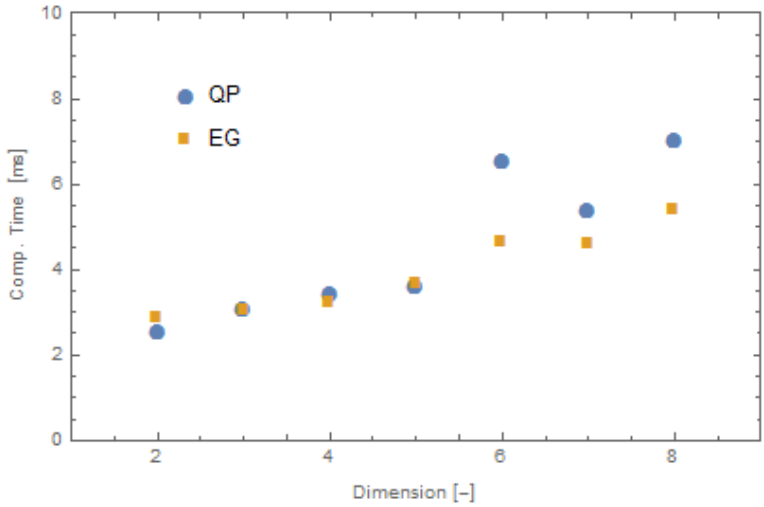
I přes snahu vytvořit porovnatelné podmínky, můžou být tyto výsledky závislé např. na nastavení algoritmů, HW a SW konfiguraci a jiných

aspektech. Bylo proto provedeno několik opakování a výsledky porovnání jsou uvedeny v následující tabulce Tab. 1 (menší číslo znamená lepší).

Jak je vidět z tabulky, všechny hodnoty jsou podobné, odlišují se až v jednotkách procent. Algoritmy byly také porovnány v závislosti na nastavení řídicího horizontu, a tedy dimenzionality řešení optimalizačního problému, viz Obr. 6.7.

Tab. 1: Porovnání algoritmů optimalizátoru pro řízení s CARIMA modelem, vlastní zdroj[96]

	Výpočetní. čas [ms]	**SSE**
EG algoritmus	2,10	234,67
HC algoritmus	2,28	238,29
QP algoritmus	2,01	232,00



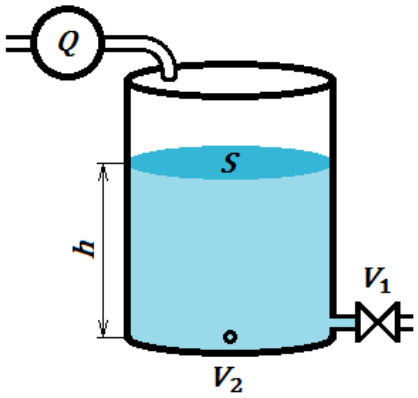
Obr. 6.7: Porovnání výpočetní náročnosti pro různou dimenzionalitu optimalizačního problému, vlastní zdroj[96]

Uvedené výsledky měly ukázat, že existují i evoluční algoritmy, které mohou být srovnatelné s klasickými metodami. Vzorkovací perioda řízení je obvykle navíc pevně stanovena, a z tohoto hlediska je tedy nutné hledat pouze takové algoritmy, které jsou schopné v rámci vzorkovací periody vyřešit optimalizační problém. Z výše uvedených důvodů byla tedy možnost nasazení evolučních algoritmů v prediktivním řízení potvrzena.

6.1.3 Ověření možnosti nasazení RBF jako modelu

Stejně jako u evolučních algoritmů, je i pro použití neuronových sítí v prediktivním řízení nutné provést ověření možnosti nasazení tohoto prvku. Pro tyto účely bude použit modelový systém, který je svými vlastnostmi podobný modelovým systémům uvedených v cílech práce.

Nechť existuje nelineární systém tvořený nádrží, čerpadlem, ventilem a výpustním ventilem, viz Obr. 6.8.



Obr. 6.8: Nelineární systém

kde Q je objemový průtok vstupního čerpadla, h je výška hladiny v nádrži, S je průřez nádrže, V_1 reprezentuje výstupní ventil a V_2 simuluje odpouštění kapaliny.

Pro účely práce je tento systém simulován v prostředí SIMULINK. Abstraktní model systému lze popsat diferenciální rovnicí, kterou lze odvodit matematicko-fyzikální analýzou systému (při splnění zákonů zachování). Změna objemu kapaliny v nádrži je pak rovna součtu jednotlivých průtoků.

$$S\frac{dh(t)}{dt} = Q(t) - q_1(t) - q_2(t) \tag{6.10}$$

V rámci prostředí SIMULINK je tento systém reprezentován S-Funkcí. Průtok je dán poměrem tlaku před a za ventilem, přičemž tlak je závislý na výšce hladiny. Při uvážení konstrukce systému (hladina za ventilem je rovna nule), lze jednotlivé průtoky vyjádřit následovně

$$q_1(t) = k_1\sqrt{h(t)}, \quad q_2(t) = k_2\sqrt{h(t)} \tag{6.11}$$

kde k jsou koeficienty jednotlivých ventilů.

Po substituci lze reálný systém zapsat následující diferenciální rovnicí.

$$S\frac{dh(t)}{dt} = Q(t) - k_1\sqrt{h(t)} - k_2\sqrt{h(t)}$$
$$S\frac{dh(t)}{dt} = Q(t) - (k_1 + k_2)\sqrt{h(t)} \tag{6.12}$$

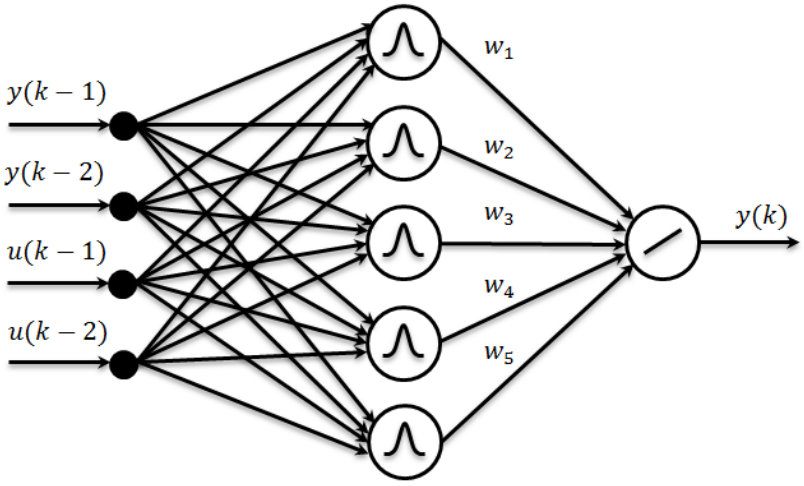
Volba modelu

Z rovnic je patrná nelinearita systému, takže pro účely řízení je možné použít některý z nelineárních typů modelu. Pro účely možností nasazení NN byl aplikován model RBF sítě. V zájmu zachování jednoduchosti je vynechána strukturní optimalizace a pro snížení stupňů volnosti jsou některé parametry voleny fixně (např. počet neuronů). Ostatní parametry jsou pak určeny sub-optimalizací.

Při předpokladu systému 2. řádu a tvaru běžných diskrétních modelů, lze systém zapsat jako funkci minulých hodnot (a parametrů).

$$y(k) = f(y(k-1), y(k-2), u(k-1), u(k-2), \boldsymbol{\Theta}(k)) \tag{6.13}$$

Toto lze popsat modelem RBF sítě se 4 vstupy a 1 výstupem. Počet neuronů skryté vrstvy je prvním stupněm volnosti, předpokládejme ale, že pro popis systému bude stačit 5 skrytých neuronů (možno určit strukturní optimalizací). Model pak lze znázornit schématem na Obr. 6.9, kde skryté neurony mají přenosovou funkci radiální báze, výstupní neuron pak funkci lineární. Jak je patrné z kapitoly 3.2.1, každý neuron je popsán svým středem $\boldsymbol{c}_i$, svou normou $\boldsymbol{\Sigma}_i$ a svou výstupní vahou w_i. Počet parametrů však určuje dimenzionalitu účelové funkce, proto byla zavedena stejná konstantní norma pro všechny neurony, čímž byl ubrán další stupeň volnosti. Vektor parametrů modelu $\boldsymbol{\Theta}(k)$ byl tedy redukován na vektor výstupních vah a středů, které byly určeny sub-optimalizací (viz dále).



Obr. 6.9: Model soustavy RBF sítě, vlastní zdroj.

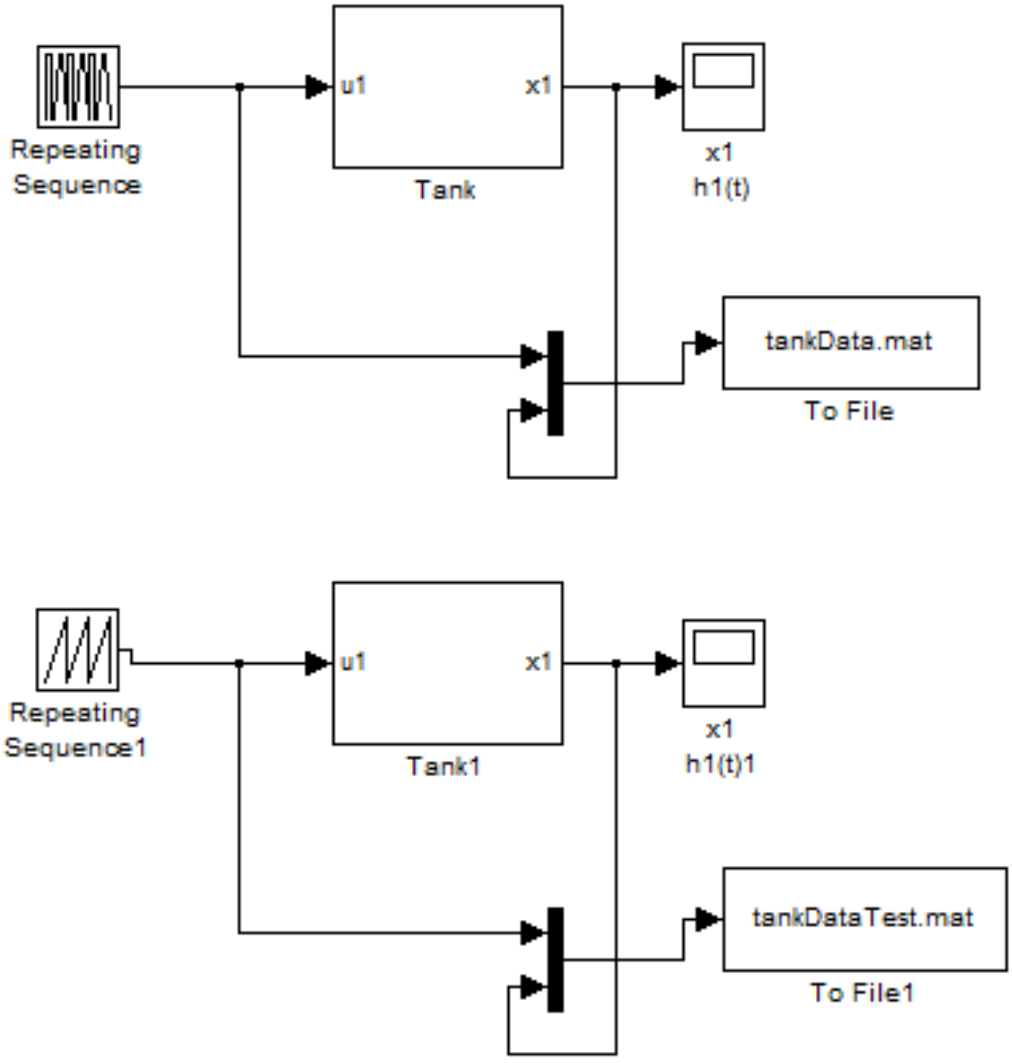
Identifikace a verifikace modelu

Pro identifikaci modelu RBF je nutné síť dobře naučit. Pro tyto účely byla vytvořena trénovací množina a sada testovacích množin. K tomu lze využít simulaci reálného systému v prostředí SIMULINK a aplikaci typických vstupních signálů ve zvolené pracovní oblasti. Výstupem systému je pak sledovaná výška hladiny a vstupem je objemový průtok čerpadla. Schémata tvorby trénovací a testovací množiny v SIMULINK jsou uvedena na Obr. 6.10.

Pro identifikaci modelu je použita pouze trénovací množina. Nejprve je však nutné definovat účelovou funkci.

$$CF = (\boldsymbol{y}_s - \boldsymbol{y}_m)^2 \tag{6.14}$$

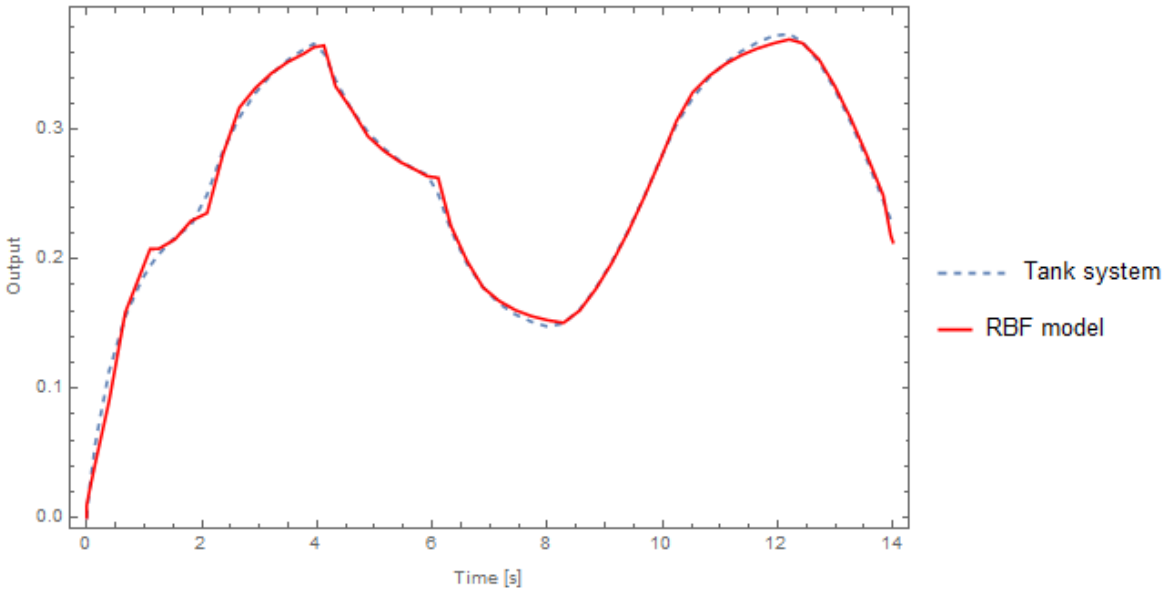
kde $\boldsymbol{y}_s$ je výstup reálného systému, $\boldsymbol{y}_m$ je výstup modelu a CF je účelová funkce optimalizačního problému učení.



Obr. 6.10: Tvorba trénovací a testovací množiny, vlastní zdroj.

Existuje několik strategií pro trénování RBF sítě [89], v tomto řešeném případě však byla strategie vytvořena kombinací zvolených postupů do konkrétní následující podoby (možno provést i více iterací):

1. Výběr středů náhodným výběrem vstupních dat
2. Optimalizace vah na základě středů (lineární vrstva)
3. Optimalizace středů na základě určených vah (nelineární vrstva)

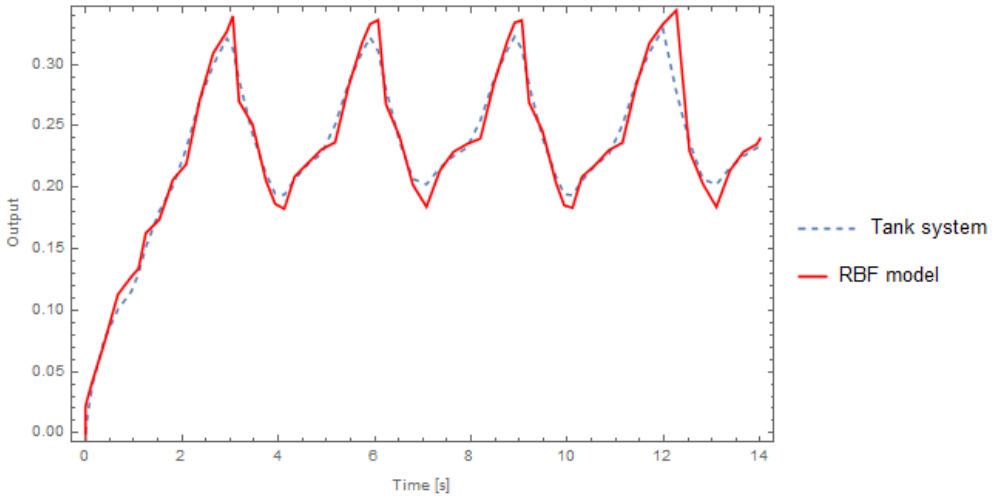


Obr. 6.11: Trénování RBF sítě, vlastní zdroj.

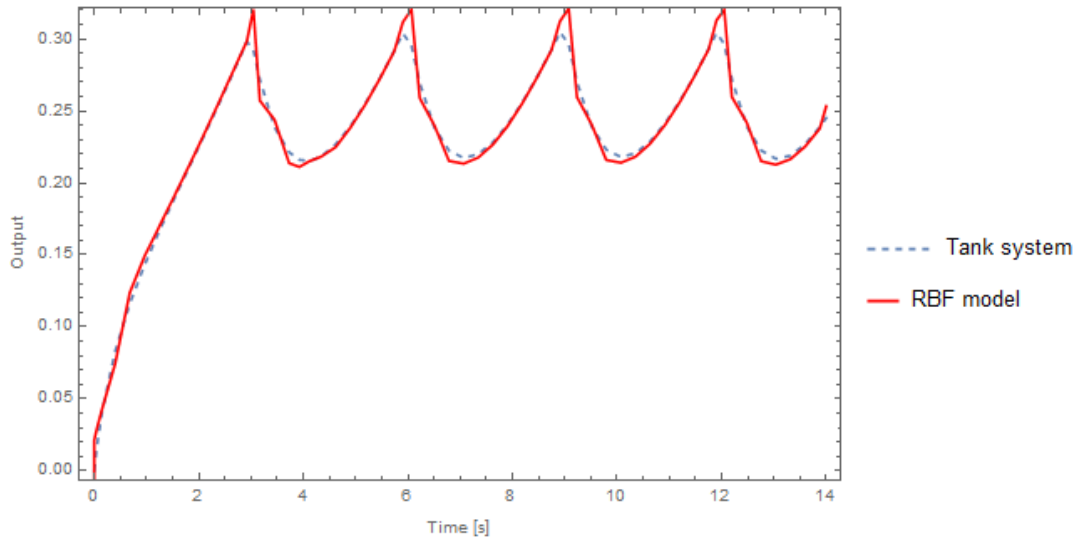
Trénování je úspěšné, pokud se výstup RBF sítě a reálného systému shoduje, viz Obr. 6.11 (vstupní signál není zobrazen). Dobré naučení sítě je klíčové pro správnou tvorbu modelu, proto by trénovací množina měla obsahovat typické vstupní signály (vzory), jedině tak je totiž možné vytvořit model, který dostatečně přesně popisuje reálný systém.

Učení neuronové sítě obvykle sestává ze dvou kroků: trénování a testování. Proto byla na vstup reálného systému aplikována sada testovacích vstupů, pomocí nichž lze otestovat úspěšnost vytvoření modelu. Po aplikaci testovacích množin jsou reálný systém (S-Funkce) a RBF síť opět porovnány (Obr. 6.12).

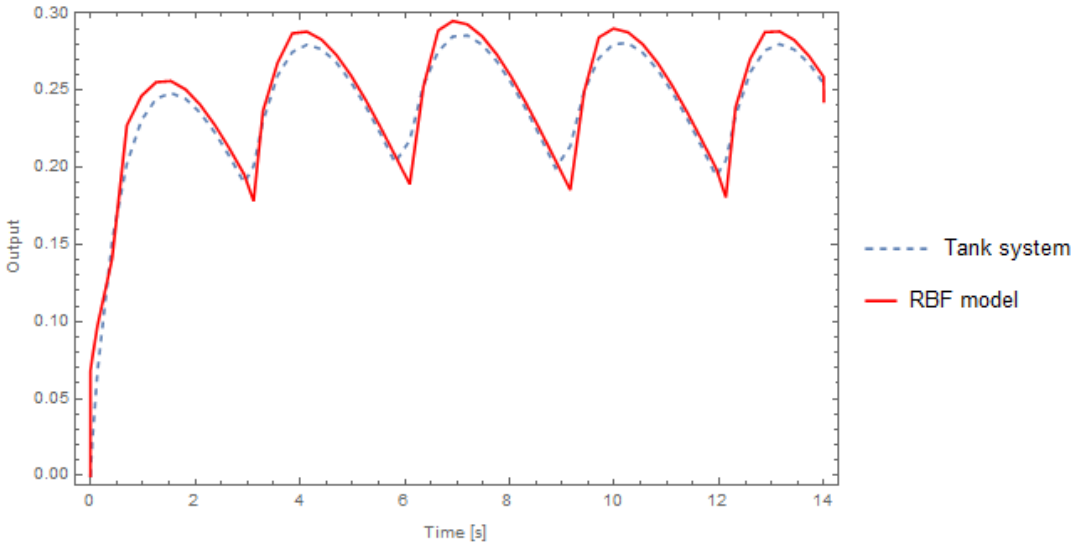
a)



b)



c)



Obr. 6.12: Testování RBF sítě, vlastní zdroj.

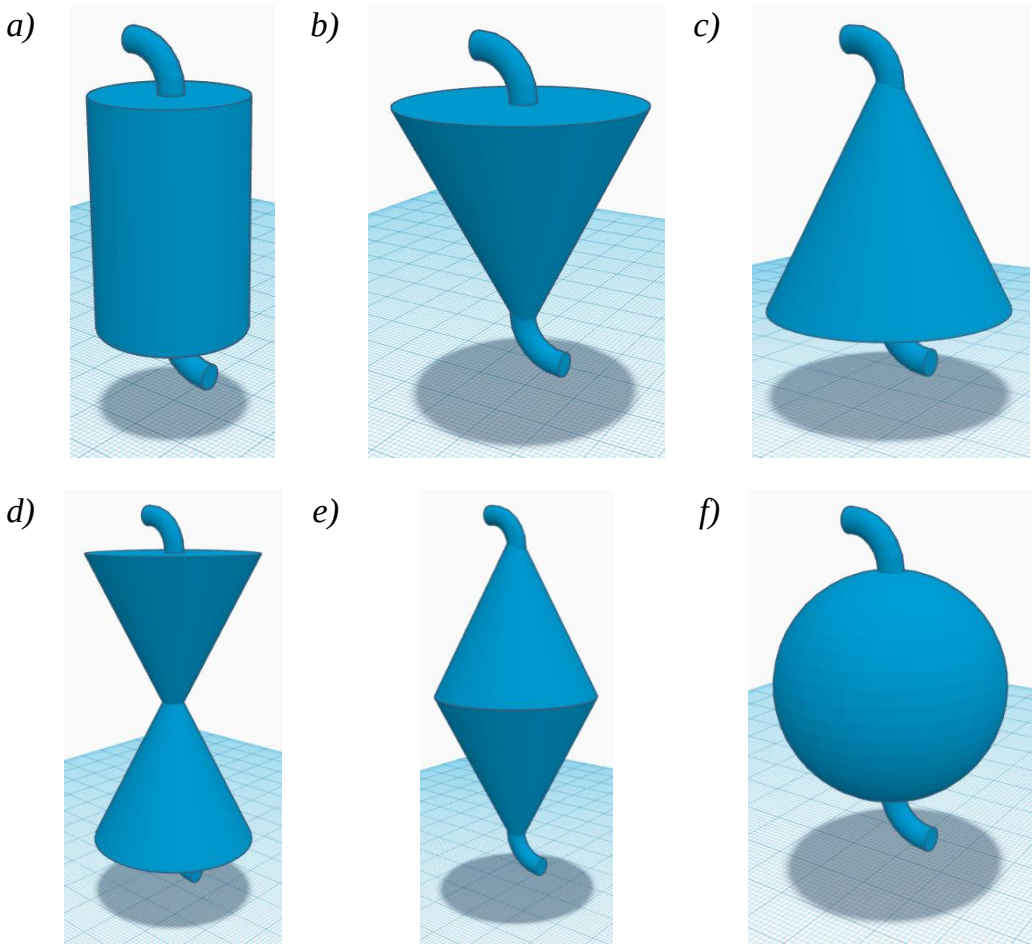
Jak je z grafů v Obr. 6.12 patrné, RBF síť celkem dobře odpovídá chování reálného systému ve zvolené pracovní oblasti ($R^2 \approx 0.96$). Možnost nasazení neuronové sítě jako modelu byla tedy ověřena.

6.2 Volba systému pro prediktivní řízení a vytvoření modelového systému

6.2.1 Modelové systémy – geometrické abstrakce

Modelové systémy budou sloužit k simulaci a ke tvorbě trénovacích a testovacích množin. Z důvodu škálovatelnosti bude přikročeno k non-dimenzionalizaci všech veličin. V první fázi nebudou systémy omezené, a otázka omezení a saturace bude diskutována později.

Bylo vybráno celkem 6 systémů s dobře definovanou geometrií (Obr. 6.13).

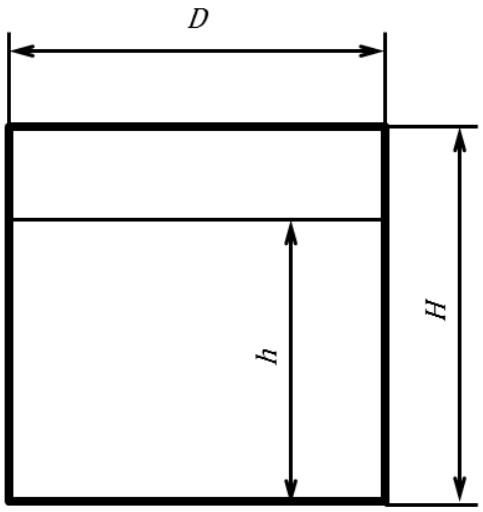


Obr. 6.13: Modelové systémy: a) válec, b) kužel, c) komolý kužel, d) dvojkužel s úzkých hrdlem, e) dvojkužel a f) koule, vlastní zdroj.

Vybrané systémy jsou osově souměrné, kdy rovnice objemu pro model obecného tvaru může být získána rotací funkce tvaru a integrací. Pro získání diferenciální rovnice je tedy možné tyto rovnice derivovat, čímž je získán jeden element integrace. U všech modelů bude využito chain-rule, kdy derivace vnitřní funkce odpovídá požadovanému podílu diferenciálů.

Modelový systém: válec

Nejjednodušším modelem, který bude uvažován, je válec výšky H o průměru D, viz Obr. 6.14.



Obr. 6.14: Modelový systém: válec, vlastní zdroj.

Pro další odvození budeme uvažovat výšku $H = 1$ a velikost průměru jako násobek této výšky.

$$D = k_D H \quad \Rightarrow \quad D\big|_{H=1} = k_D \tag{6.15}$$

Aktuální objem kapaliny ve válci je závislý na aktuální výšce h. Pro zjednodušení dalšího zápisu budou všechny časově závislé veličiny označeny (t) pouze při prvním výskytu.

$$V_h(t) = \pi R^2 h(t) = \frac{\pi}{4} {k_D}^2 h \tag{6.16}$$

Změna objemu nádrže q_t je pak popsána následující diferenciální rovnicí.

$$q_t(t) = \frac{dV_h}{dt} = \frac{\pi}{4} {k_D}^2 \frac{dh}{dt} \tag{6.17}$$

Rovnici je možno upravit do vhodnějšího tvaru.

$$\frac{dh}{dt} = \frac{q_t}{\frac{\pi}{4} {k_D}^2} \tag{6.18}$$

Pro nondimenzionalizaci času se bude dále průtok uvažovat v jednotkách [objem nádrže/jednotka času]. Budeme tedy uvažovat, že maximální

průtok q_{max} je takový, kdy se celý objem ($h = H = 1$) nádrže napustí/vypustí za jednu jednotku času ($\Delta t_{max} = 1$).

$$q_{max}\Big|_{h=H} = \frac{V_{max}}{\Delta t_{max}} = \frac{\pi}{4} {k_D}^2 \frac{H}{\Delta t_{max}}$$
$$q_{max}\Big|_{H=1 \wedge \Delta t_{max}=1} = \frac{\pi}{4} {k_D}^2 \tag{6.19}$$

Pro pozdější potřeby simulace zavedeme do rovnice přepočet času jako

$$k_q = \frac{1}{\Delta t_{max}} \Rightarrow q_{max} = k_q \frac{\pi}{4} {k_D}^2 \tag{6.20}$$

Pro další účely odvození budou přítok q_{in} i odtok q_{out} uvažovány v intervalu $\langle 0,1 \rangle$ a tedy zavedením do rovnice (6.18) je získána finální podoba diferenciální rovnice modelového systému.

$$\frac{dh}{dt} = \frac{k_q(q_{in}(t) - q_{out}(t)) \frac{\pi}{4} {k_D}^2}{\frac{\pi}{4} {k_D}^2} = \frac{k_q(q_{in} - q_{out})}{1} \tag{6.21}$$

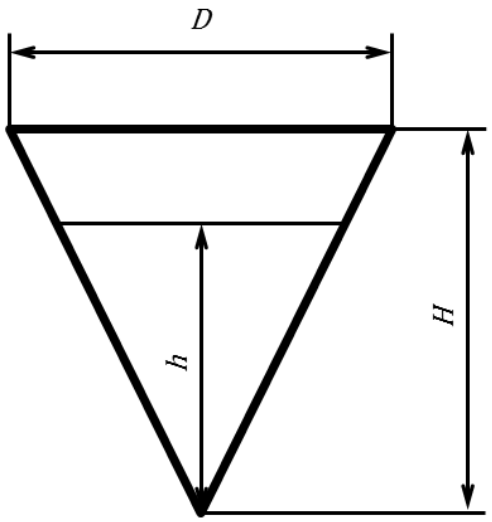
Z rovnice je patrné, že změna výšky hladiny (objemu) je nezávislá na průměru nádrže. Pro dosažení přenositelnosti je dále pominut efekt hydrostatického tlaku, přičemž řízen bude samotný odtok z nádrže $q_{out} = u$

$$\frac{dh}{dt} = \frac{k_q(q_{in} - u)}{1} = \frac{k_q a(t)}{1} \tag{6.22}$$

kde a je akumulace kapaliny v nádrži.

Modelový systém: kužel

Pro získání diferenciální rovnice modelového systému ve tvaru kuželu nejprve definujeme jeho geometrické vlastnosti (Obr. 6.15).



Obr. 6.15: Modelový systém: kužel, vlastní zdroj.

Opět je maximální výška $H = 1$ a aktuální objem je závislý na aktuální výšce h a aktuálním průměru d.

$$V_h = \frac{\pi}{3} r^2 h = \frac{\pi}{3} \left(\frac{d(t)}{2} \right)^2 h = \frac{\pi}{12} d^2 h \tag{6.23}$$

Závislost aktuálního průměru d na výšce hladiny h je lineární.

$$d = k_d h \tag{6.24}$$

Po dosazení do rovnice (6.23) je aktuální objem ve tvaru.

$$V_h = \frac{\pi}{12} (k_d h)^2 h = \frac{\pi {k_d}^2}{12} h^3 \tag{6.25}$$

Změna objemu nádrže je získána derivací.

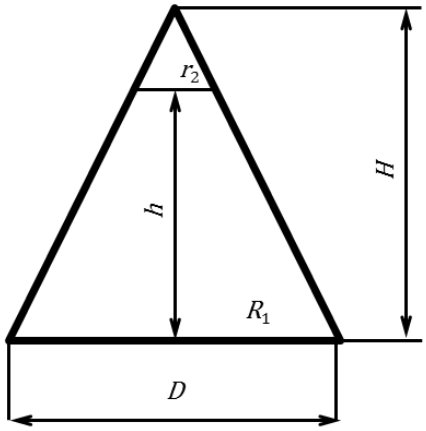
$$q_t = \frac{dV_h}{dt} = \frac{\pi k_d}{4} h^2 \frac{dh}{dt} \quad \Rightarrow \quad \frac{dh}{dt} = \frac{q_t}{\frac{\pi k_d}{4} h^2} \tag{6.26}$$

Po nondimenzionalizaci času je získána finální podoba rovnice ($q_{out} = u$).

$$\frac{dh}{dt} = \frac{k_q (q_{in} - u)}{3h^2} \tag{6.27}$$

Modelový systém: komolý kužel

Mohlo by se zdát, že tento model (Obr. 6.16) je pouze otočení předchozího modelu o 180 stupňů. Nicméně pro odvození je nutné si uvědomit, že při jiné než maximální výšce je aktuální objem dán rovnicí objemu komolého kuželu.



Obr. 6.16: Modelový systém: komolý kužel, vlastní zdroj.

$$V_h = \frac{\pi}{3}\left({R_1}^2 + R_1 r_2(t) + (r_2(t))^2\right)h \tag{6.28}$$

Z geometrie modelu je patrné, že poloměr r_2 je závislý na výšce hladiny. Z podobnosti trojúhelníků a při uvažování $H = 1$, lze odvodit tuto závislost.

$$\frac{R_1}{1} = \frac{r_2}{1-h} \quad \Rightarrow \quad r_2 = R_1 - R_1 h \tag{6.29}$$

Po dosazení do (6.28) a zjednodušení, lze získat rovnici objemu závislou pouze na poloměru podstavy.

$$V_h = \frac{\pi}{3}\left(3{R_1}^2 - 3{R_1}^2 h + {R_1}^2 h^2\right)h \tag{6.30}$$

Pro dosažení stejného postupu jako u předchozích modelů nahradíme poloměr průměrem (resp. přepočtovou konstantou $k_D = D$) a upravíme pro účely derivace.

$$V_h = \frac{\pi}{4}{k_D}^2 h - \frac{\pi}{4}{k_D}^2 h^2 + \frac{\pi}{12}{k_D}^2 h^3 \tag{6.31}$$

Jeden element integrace pak získáme derivací.

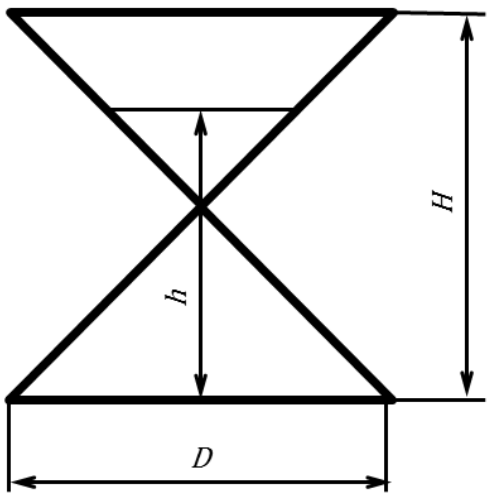
$$q_t = \frac{dV_h}{dt} = \frac{\pi}{4}{k_D}^2\frac{dh}{dt} - \frac{\pi}{2}{k_D}^2 h\frac{dh}{dt} + \frac{\pi}{4}{k_D}^2 h^2\frac{dh}{dt}$$
$$\Rightarrow \quad \frac{dh}{dt} = \frac{q_t}{\pi{k_D}^2\left(\frac{1}{4} - \frac{1}{2}h + \frac{1}{4}h^2\right)} \tag{6.32}$$

Po dosazení maximálního průtoku je opět získána diferenciální rovnice modelu.

$$\frac{dh}{dt} = \frac{k_q(q_{in} - u)\frac{\pi}{12}{k_D}^2}{\pi{k_D}^2\left(\frac{1}{4} - \frac{1}{2}h + \frac{1}{4}h^2\right)} = \frac{k_q(q_{in} - u)}{3 - 6h + 3h^2} \tag{6.33}$$

Modelový systém: dvojkužel s úzkým hrdlem

Pro tento systém (Obr. 6.17) připomínající tvarem přesýpací hodiny, jako pro předchozí, platí, že maximální výška hladiny je rovna 1, což znamená, že pro využití již odvozených modelů je nutné výšku h přemapovat. Dále budeme předpokládat, že rychlost proudění ve střední části není omezená, čímž vyloučíme nepříznivé stavy pro simulaci a modelování, tj. podobnost s přesýpacími hodinami je čistě geometrická, nikoliv však funkční, ve spodní části nádrže není volná hladina, dokud je kapalina v horní části nádrže.



Obr. 6.17: Modelový systém: dvojkužel s úzkým hrdlem, vlastní zdroj.

Tento model se skládá ze dvou částí, kdy první polovinu výšky tvoří model komolého kuželu a druhou pak model kuželu. Nejprve vyřešíme případ kdy $h\epsilon\langle 0; 0{,}5)$. Pro využití rovnice (6.33) je nutné výšku přemapovat tak, aby byl interval v původní rovnici $h_{map}\epsilon\langle 0; 1)$

$$\frac{dh}{dt} = \frac{k_q(q_{in} - u)}{3 - 6h_{map} + 3{h_{map}}^2} = \frac{k_q(q_{in} - u)}{3 - 6(2h) + 3(2h)^2}$$
$$\frac{dh}{dt} = \frac{k_q(q_{in} - u)}{3 - 12h + 12h^2} \tag{6.34}$$

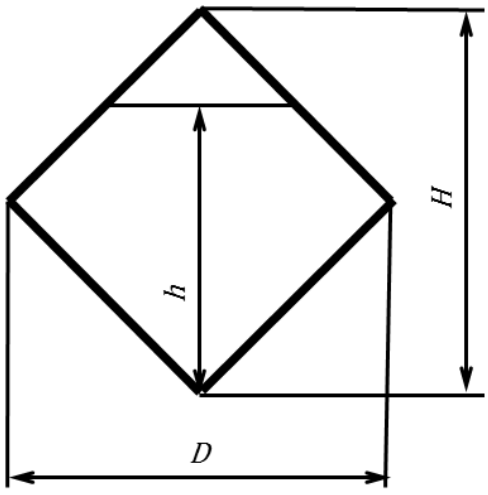
Poté vyřešíme případ $h\epsilon\langle 0{,}5; 1\rangle$

$$\frac{dh}{dt} = \frac{k_q(q_{in} - u)}{3\left(2\left(h - \frac{1}{2}\right)\right)^2} = \frac{k_q(q_{in} - u)}{3(2h - 1)^2}$$
$$\frac{dh}{dt} = \frac{k_q(q_{in} - u)}{3 - 12h + 12h^2} \tag{6.35}$$

Jak je patrné, tak při přemapování nabyly rovnice (6.34) a (6.35) stejného tvaru, a tudíž je možné použít pro popis pouze jednu rovnici modelu.

Modelový systém: dvojkužel

Stejně jako u předchozího modelu je nutné provést přemapování výšky h, u dvojkuželu (Obr. 6.18) však odpadá problém s úzkým hrdlem.



Obr. 6.18: Modelový systém: dvojkužel, vlastní zdroj.

Pro případ $h\epsilon\langle 0;\, 0{,}5)$.

$$\frac{dh}{dt} = \frac{k_q(q_{in} - u)}{3(2h)^2} = \frac{k_q(q_{in} - u)}{12h^2} \tag{6.36}$$

Pro případ $h\epsilon\langle 0{,}5;\, 1\rangle$

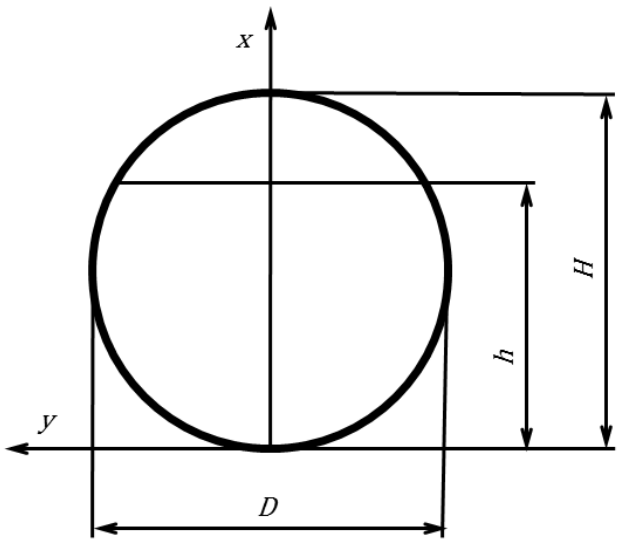
$$\frac{dh}{dt} = \frac{k_q(q_{in} - u)}{3 - 6\left(2\left(h - \frac{1}{2}\right)\right) + 3\left(2\left(h - \frac{1}{2}\right)\right)^2} \tag{6.37}$$

$$\frac{dh}{dt} = \frac{k_q(q_{in} - u)}{12 - 24h + 12h^2}$$

V tomto případě se již obě rovnice liší, a je tedy nutné testovat, v které části nádrže se hladina právě nachází.

Modelový systém: koule

Pro odvození modelového systému koule (Obr. 6.19) je nutné si uvědomit, že aktuální objem není dán vztahem pro výpočet objemu koule, ale její části.



Obr. 6.19: Modelový systém: koule, vlastní zdroj.

Pro výpočet objemu této části můžeme využít rotace funkce a integrace,

$$V_h = \pi \int_0^h f^2(x)dx \tag{6.38}$$

kde $f(x)$ je funkce tvaru. Pokud $f = y$ a $H = 1$, pak je funkce dána jako

$$\left(x - \frac{1}{2}\right)^2 + y^2 = \left(\frac{1}{2}\right)^2 \quad \Rightarrow \quad f^2(x) = x - x^2 \tag{6.39}$$

Po dosazení (6.39) do (6.38) získáme rovnici aktuálního objemu části koule.

$$V_h = \pi \int_0^h (x - x^2)dx = \pi \frac{h^2}{2} - \pi \frac{h^3}{3} \tag{6.40}$$

Změna objemu je opět získána derivací.

$$\begin{gathered} q_t = \frac{dV_h}{dt} = \pi h \frac{dh}{dt} - \pi h^2 \frac{dh}{dt} = \frac{dh}{dt}(\pi h - \pi h^2) \\ \Rightarrow \quad \frac{dh}{dt} = \frac{q_t}{\pi(h - h^2)} \end{gathered} \tag{6.41}$$

A po dosazení maximálního toku.

$$\frac{dh}{dt} = \frac{k_q(q_{in} - u)}{6(h - h^2)} \tag{6.42}$$

6.2.2 Modelové systémy: ověření správnosti

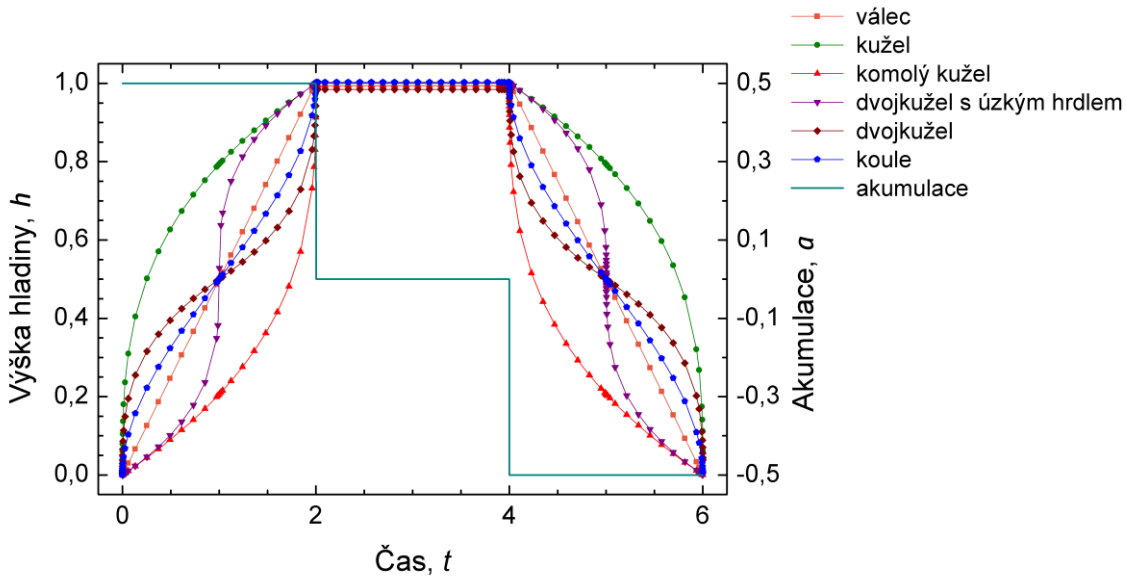
Po odvození modelových systémů je třeba ověřit, že získaný matematický popis odpovídá navrženým geometrickým vlastnostem a zároveň je správně provedena nondimenzionalizace veličin. Pro testování byl zvolen konstantní přítok $q_{in} = 0{,}5$ a přepočtový faktor $k_q = 1$. Odtok byl nastaven na hodnoty

$$q_{out} = \begin{cases} 0 & \text{pro } t \in \langle 0; 2) \\ 0{,}5 & \text{pro } t \in \langle 2; 4) \\ 1 & \text{pro } t \in \langle 4{,}6 \rangle \end{cases} \tag{6.43}$$

Počáteční hladina byla nastavena na $h(0) = 0$, pokud model vyžadoval nenulové počáteční podmínky, byla nastavena na $h(0) = 0{,}0000001$. Hodnota akumulace a (viz (6.22)), která bude použita v grafu, tedy bude

$$a = \begin{cases} 0{,}5 & \text{pro } t \in \langle 0; 2) \\ 0 & \text{pro } t \in \langle 2; 4) \\ -0{,}5 & \text{pro } t \in \langle 4{,}6 \rangle \end{cases} \tag{6.44}$$

Dle definice nondimenzializace času (viz kapitola Modelový systém: kužel) se předpokládá, že při hodnotě akumulace $a = 1$, by se celý objem nádrže měl napustit za jednu jednotku času. Při zvoleném časovém průběhu akumulace lze tedy předpokládat, že při nulových počátečních podmínkách se nádrž nejdříve zcela napustí (za dvě jednotky času), poté se hladina ustálí (vyrovnání toků) a nakonec se nádrž zcela vypustí.

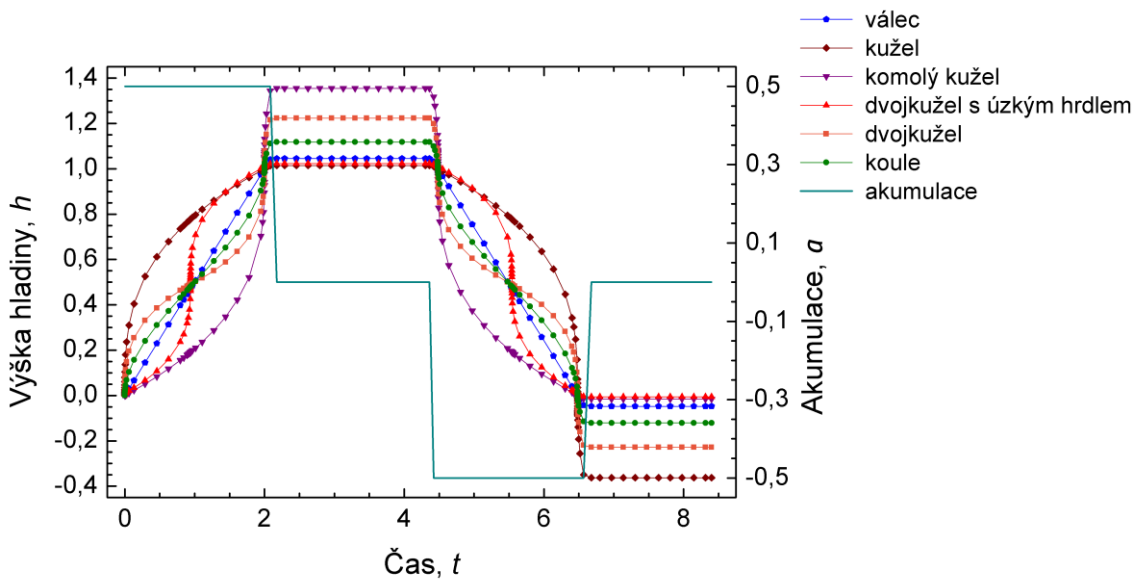


Obr. 6.20: Modelové systémy – ověření správnosti, vlastní zdroj.

Simulace s modelovými systémy potvrdila předpokládaný časový průběh, viz Obr. 6.20. Tvar časových průběhů také odpovídá očekávání, a tedy správnost popisu modelových systémů byla potvrzena. Drobné nepřesnosti např. v dosažení maximální hladiny mohou být způsobeny např. numerickým charakterem výpočtu při simulaci, či nenulovými poč. podmínkami.

Dále byly provedeny simulace s prodlouženými časovými intervaly pro studium chování mimo definované meze. Jak je vidět na Obr. 6.21, výstupní veličina modelových systémů se pohybuje i mimo definovaný interval $\langle 0; 1 \rangle$, kde může vykazovat odlišné vlastnosti od vlastností očekávaných.

Je třeba také uvést, že nastavení jednotlivých hodnot signálů a simulace bylo zvoleno za účelem ověření správnosti modelů, a tudíž není třeba uvažovat jiné situace jako je např. nenulový poč. stav či odlišné nastavení přítoku q_{in} a odtoku q_{out}. Pro účely generování trénovacích a testovacích množin bude ale nutné toto později vzít v úvahu.

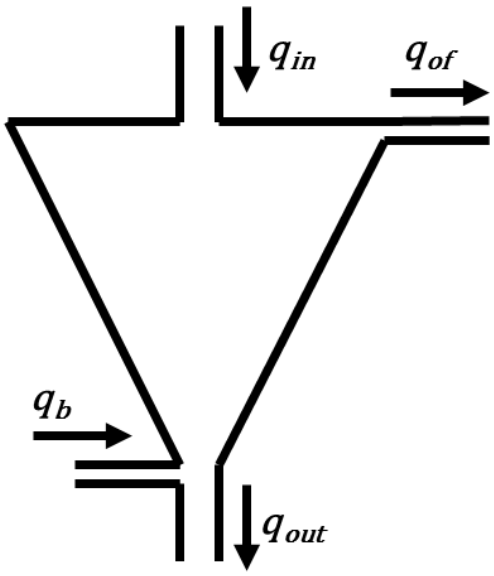


Obr. 6.21: Modelové systémy – nesaturované, vlastní zdroj.

6.2.3 Modelové systémy: saturace, singularity

Získané modely nejsou omezeny v oboru hodnot výstupní veličiny. Pro účely simulace by ale mohlo být vhodné získat modely saturované. Přímé omezení výstupní veličiny je v případě S-Funkce (level 1) obtížné, a může vykazovat několik negativních efektů. Jedním z možných řešení je úprava jednotlivých modelových systémů zavedením funkce přepadu a dna, což vlastně odpovídá i fyzikální realitě. Pokud dojde k naplnění skutečné nádoby, pak veškerý přítok, který nestačí odtékat naplno otevřenou výpustí, odtéká přes přepad, a hladina se od hrany přepadu navyšuje už pouze o aktuální výšku přelivu. To pro daný průtok nelze určit bez znalosti délky hrany přepadu (resp. smáčeného obvodu přepadového koryta). Proto je přepad modelován tak, jako by měl nekonečně velkou propustnost (tj. nekonečně dlouhou hranu, nebo nekonečně velkou rychlost pohybu kapaliny), tedy že odvádí všechen přebytečný přítok a dosažená výška hladiny už se dál nemění, dokud kapalina přetéká přepadem. Naproti tomu, pokud dojde k úplnému vypuštění reálné nádrže, kterou však stále protéká nějaká kapalina, je výška hladiny protékající nádrží dána geometrií smáčeného průřezu výpustě a objemovým tokem kapaliny. Jinými slovy řečeno, i nádrž, ve které se žádná kapalina nezadržuje při zcela otevřené výpusti, nemůže být zcela prázdná, pokud jí nějaká kapalina protéká. Proto je dosažení nulové výšky hladiny v nádrži ošetřeno obdobně jako přepad, avšak v tomto případě vše, co do nádrže přiteče, z ní také odteče, ovšem s nekonečně velkou rychlostí kapaliny, neboť výpusť nemá geometrické rozměry. Současně je však nutno omezit i dosažení záporné výšky hladiny, resp. udržet nulovou hladinu. Uvedené

řešení saturace, tedy singularit, je demonstrováno na následujícím příkladu modelového systému kuželové nádoby (Obr. 6.22).



Obr. 6.22: Modelový systém: saturace, vlastní zdroj.

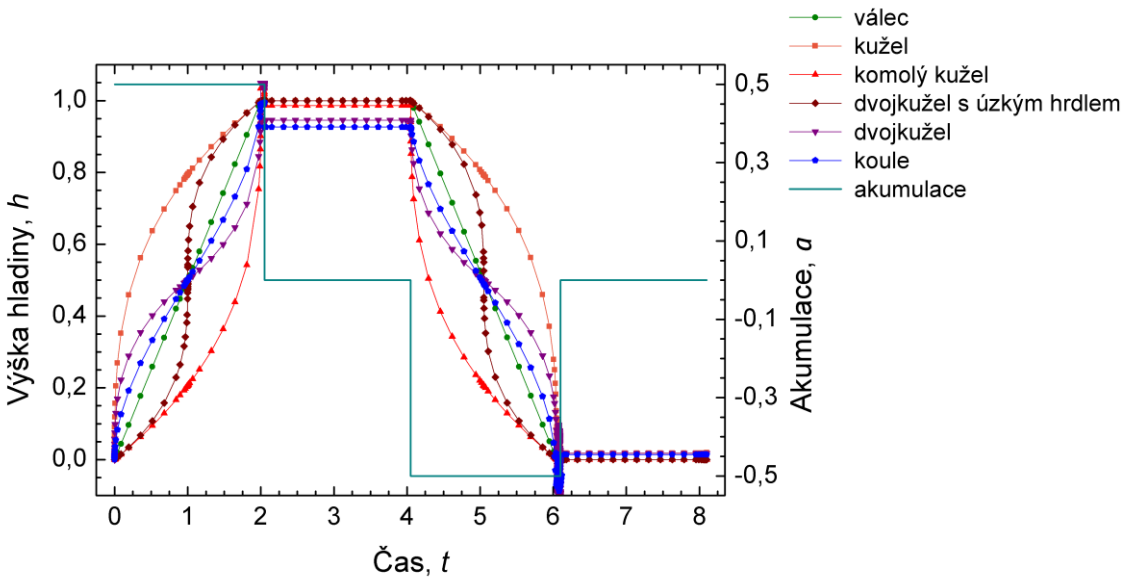
Čitatel modelu lze tedy rozšířit o přepad q_{of} (overflow) a q_b dno (bottom).

$$\frac{dh}{dt} = \frac{k_q\left(+q_{in} - q_{of} + q_b - q_{out}\right)}{3h^2}$$
$$\frac{dh}{dt} = \frac{k_q\left((q_{in} - q_{of}) - (q_{out} - q_b)\right)}{3h^2} \tag{6.45}$$

Přepad i dno lze poté řídit tak, aby funkce byla omezená.

$$[q_{of} \quad q_b] = \begin{cases} [0 \quad q_{out}] & \text{pro } h < 0 \\ [0 \quad 0] & \text{pro } 0 \leq h \leq 1 \\ [q_{in} \quad 0] & \text{pro } h > 1 \end{cases} \tag{6.46}$$

Podobným způsobem lze přizpůsobit i matematický popis ostatních modelových systémů.



Obr. 6.23: Modelové systémy - saturované, vlastní zdroj.

Jak je patrné z grafu v Obr. 6.23, omezení modelových systémů funkcí přepadu a dna jsou účinná. Nicméně, i tak signály mohou trpět jistými dalšími anomáliemi jako je kmitání v oblasti limitních hodnot výstupní funkce, či ustálení signálu na aktuální hodnotě překmitu (nebo podkmitu). Toto může být způsobeno jednak tím, jak funguje samotná S-Funkce, jednak geometrickým charakterem jednotlivých modelů. Byl zjištěn výskyt i dalších anomálií, způsobený chováním diskrétního výpočtu v blízkosti singularit, zejména v oblasti s neomezenou rychlostí průtoku (tvar typu špička a tvar kulového vrchlíku). Tomuto se dá předejít několika způsoby jako úpravou matematického popisu, úpravou dynamiky signálů (např. „zaoblením“ hran), testování/sledování stavů či omezení pracovní oblasti na „bezpečnou zónu“. Vzhledem k charakteru systému a k vlastnostem prediktivního řízení se jeví poslední způsob jako nejvhodnější.

Další singularitu nacházíme v modelovém systému dvojkužele s úzkým hrdlem právě v místě nejužšího místa, kde se stýkají dva vrcholy proti sobě stojících kuželů, a kapalina v tomto místě musí mít nekonečnou rychlost. Pro matematický popis tohoto modelu stačí jedna funkce (viz kapitola 6.2.1). Diskontinuita u modelového systému dvojkužele je řešena funkcí if, kdy spodní a horní část nádoby je popsána každá svojí částí funkce. Tyto i všechny další situace, kdy jmenovatel ve zlomku nabude nulové hodnoty, jsou ošetřeny jeho testováním v každém cyklu výpočtu a nahrazením případně se vyskytnuvší nulové hodnoty hodnotou velmi malou nenulovou, odhadnutou na základě zkušenosti s provozem modelu. Toto řešení je nejjednodušší a nevyžaduje určení kořenů polynomu.

6.2.4 Modelové systémy: aplikovatelnost

Při simulacích s modelovými systémy bylo získáno několik zkušeností s charakteristickými vlastnostmi navržených modelů, které dobře odpovídají geometrickým vlastnostem, nicméně díky nutným přijatým zjednodušením mohou v některých případech zdánlivě nebo i reálně vykazovat vlastnosti, jež nejsou zcela v souladu s fyzikální realitou. Pokud si však jsme těchto limitů vědomi, modelové systémy tím nijak neztrácejí na hodnotě.

Je nutné si uvědomit, že odvozené modely byly získány po nondimenzionalizaci všech veličin. Pro aplikovatelnost na systémy reálné je tedy třeba reálné signály dostat do této formy (více v kapitole 6.7). Vedlejším (zdánlivým) efektem nondimenzionalizace je také ztráta obvyklého intuitivního pohledu na konzistenci jednotek fyzikálních veličin. Například součet výšky a jejího čtverce $h + h^2$ by nedával fyzikální smysl, dokud si neuvědomíme, že před prvním členem h je skryta rozměrová konstanta, která u druhého (kvadratického) členu není.

Dále byl vysloven předpoklad, že hydrostatický tlak je pominut a řízen je samotný odtok q_{out}, nikoliv výpustní ventil (výpusť, stavidlo). Po analýze chování geometrických modelů je však patrné, že v blízkosti singularit se rychlost proudění blíží neomezeným hodnotám, což není fyzikálně realizovatelné. Jedním z možných řešení by mohlo být zavedení hydrostatického tlaku nahrazením $q_{out} = k_{vout}\sqrt{h}$, kde k_{vout} je koeficient ventilu, a řízením tohoto ventilu odtoku. Nicméně v tomto případě se ztrácí výhoda nondimenzionalizace a škálovatelnosti, protože výsledné chování je parametrově závislé. Druhým přístupem je omezení pracovní oblasti tak, aby se výstupní veličina pokud možno nepohybovala v blízkosti singularit. Tento přístup může také omezit některé anomálie dané numerickým charakterem výpočtu při simulacích. Singularitě se nelze ovšem zcela vyhnout v případě dvojkuželu s úzkým hrdlem, neboť ta se nachází uprostřed výšky nádoby (h = 0,5). Konečně posledním předpokladem použitelnosti extrémně zjednodušeného modelu výpusti (ventilu odtoku) je obvyklá monotónnost (ne-li přímo linearita) jeho (její) charakteristiky v reálném světě.

Dále je nutné si uvědomit, že provedené simulace, o kterých pojednává tato kapitola, měly ukázat, že odvozené modely vykazují předpokládané geometrické a časové vlastnosti, a mohou tedy posloužit k získání trénovacích a testovacích množin dat. Nicméně reálné systémy bývají provozovány v bezpečných mezích a dynamika jednotlivých signálů bývá zpravidla daleko menší, než v provedených simulacích.

6.3 Volba a vytvoření modelu

Při volbě modelu je nutné zvážit, jaký typ systému má model popisovat a jaké jsou požadavky na řízení systému. Jak již bylo zmíněno v teoretické části, tak model dynamického systému může být popsán diferenciální rovnicí, která popisuje chování systému. Takový model, na rozdíl od statického, má tedy „paměť“, kdy diferenciální rovnice popisuje změnu výstupní veličiny na základě aktuálního stavu systému. Takový model nemusí být ovšem pro řízení vhodný, neboť analytické řešení může být obtížné. Proto se obvykle tento systém převádí pomocí Laplaceovy transformace[83] na funkce komplexní proměnné s, kdy místo diferenciální rovnice pracujeme s algebraickými výrazy (většinou polynomy).

V reálném řízení však obvykle nejsou signály spojitého charakteru a dochází zde ke vzorkování. Spojitý systém lze poté vyjádřit pomocí diskrétního modelu (Z-transformace[83]), čemuž v časové oblasti odpovídají rovnice diferenční. Tyto rovnice jsou závislé na zvolené periodě a obvykle jsou popsány jako vážená kombinace hodnot v „okolí“ aktuální periody (v závislosti na řádu systému). Tyto váhy (koeficienty modelu) poté mohou být konstantní, či mohou být průběžně upravovány (adaptivní řízení[10, 52, 53, 55, 97]).

6.3.1 Model systému popsaný neuronovou sítí

Jak je z předchozí kapitoly patrné, modelové systémy vykazují jistý typ nelineárního chování. Volba správného modelu je pro řízení důležitá, neboť model by měl co nejpřesněji popisovat chování systému. Z tohoto hlediska mohou být nejvhodnější modely získány matematicko-fyzikální analýzou (viz kapitola 6.1.3), neboť pak mají koeficienty modelu fyzikální význam. Ne vždy je však možné takový model získat, případně nemusí být pro daný typ řízení vhodný.

V našem případě byly odvozeny modelové systémy, které se svým geometrickým charakterem mohou přibližovat systémům reálným. Předpokládá se ale, že modely popsané neuronovými sítěmi mohou být nasazeny na systémy obtížně popsatelné, jako jsou např. nádrže obecného tvaru, či na systémy, u nichž není znám jejich matematický popis, či je obtížně získatelný. Podmínkou nasazení je však dostatečné množství trénovacích dat (viz cíl č. 7), které dostatečně přesně reprezentují typické průběhy veličin.

Z matematického hlediska jsou neuronové sítě, stejně jako ostatní modely, popsány svými vstupy, výstupy (př. stavy) a svými parametry. Matematicky se dá neuronová síť zapsat jako

$$\boldsymbol{y} = f_{NN}(\boldsymbol{u}, \boldsymbol{\Theta_{NN}}) \tag{6.47}$$

kde $\boldsymbol{y}$ je výstupní vektor sítě, $\boldsymbol{u}$ je vstupní vektor sítě a $\boldsymbol{\Theta_{NN}}$ jsou parametry modelu. Struktura a přenosové funkce jsou popsány funkcí f_{NN}.

Jak již bylo několikrát zmíněno, je nutné, aby model co nejpřesněji popisoval chování systému. Tento požadavek však může být v konfliktu s omezením výpočetní náročnosti, která by měla být taková, aby vzhledem k charakteru řízení a jeho požadavkům (jako je např. vzorkovací perioda, výpočetní náročnost optimalizátoru a omezení) byla únosná.

Při výběru neuronové sítě byli zvažováni dva kandidáti, kteří mohou být použiti v prediktivním řízení. Neuronová síť **multi-layer perceptron** je vícevrstvá dopředná síť s přenosovými funkcemi saturačního typu. Více vrstev může teoreticky přispět k lepší přesnosti, avšak vztah mezi nastavením jednotlivých parametrů a dosažení požadované výstupní funkce nemusí být zcela patrný. Počet parametrů tohoto modelu je pak dán počtem neuronů a volných parametrů.

6.3.2 RBF model systému

Výhodou RBF modelu pak může jasná interpretovatelnost vlivu nastavení parametrů na tvar výstupní funkce (viz teorie, kapitoly 3.2.1 až 3.2.3). Při návrhu struktury sítě je nejprve nutné určit počet vstupů a výstupů. Jelikož se bude řídit vždy jedna nádrž, která má pouze jednu výstupní (stavovou) veličinu a to výšku hladiny, bude počet výstupů roven jedné. Pro popis dynamického systému je nutné zařadit do modelu paměť. U lineárního diskrétního modelu je paměť implementována v podobě váženého součtu minulých hodnot (vstupních i výstupních). Např. pro diskrétní systém 2. řádu může být model popsán následující rovnicí

$$y(k) = a_1 y(k-1) + a_2 y(k-2) + b_1 u(k-1) + b_2 u(k-2) \tag{6.48}$$

kde $y(k)$ je aktuální hodnota výstupu, $y(k-1)$ a $y(k-2)$ jsou minulé hodnoty výstupní veličiny, $u(k-1)$ a $u(k-2)$ jsou minulé hodnoty vstupního signálu[3] a a_1, a_2, b_1, b_2 jsou parametry modelu.

[3] Pro dosažení integračního charakteru modelu je možné použít místo absolutních hodnot vstupů jejich přírůstky.

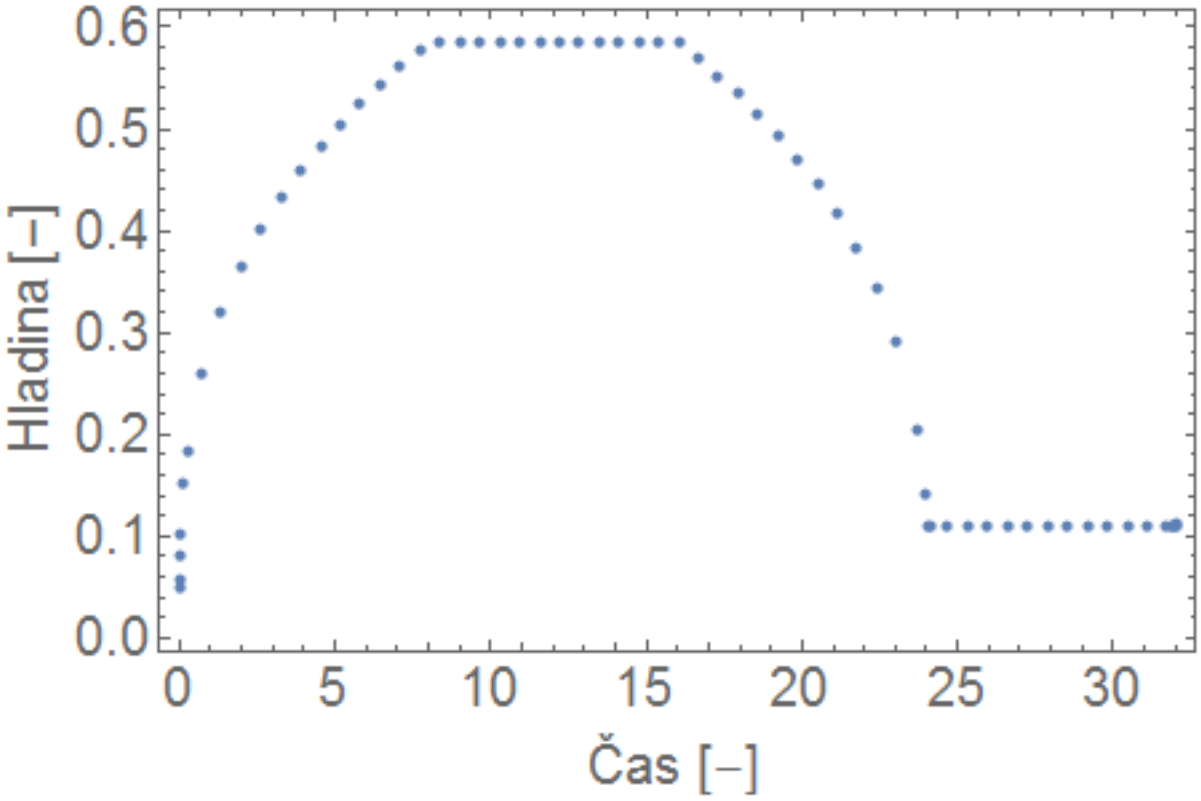
Zařazením minulých hodnot výstupní veličiny na vstupy neuronové sítě se z dopředné sítě stane síť rekurentní, čímž síť získá paměť stavu. Z charakteru modelových systémů je patrné, že model systému by měl být alespoň 2. řádu. Počet vstupů neuronové sítě bude tedy roven čtyřem.

Jelikož neuronová síť RBF obsahuje obvykle jednu skrytou vrstvu, je nutné určit počet neuronů skryté vrstvy. Určení parametrů modelu je optimalizační problém, ve kterém se tyto parametry hledají tak, aby byla výstupní odchylka minimální. Toto je problém aproximace, kdy je třeba pro naměřená (trénovací) data nalézt funkční předpis a jeho parametry. Rovnice RBF modelu je pak popsána jako součet funkcí Gaussovského typu. Počet členů této rovnice (neuronů ve skryté vrstvě) by tedy měl být alespoň takový, jako je řád systému a zároveň takový, aby poloha středů jednotlivých neuronů byla schopna pokrýt nelinearitu výstupní funkce.

6.4 Identifikace soustavy a nalezení parametrů modelu

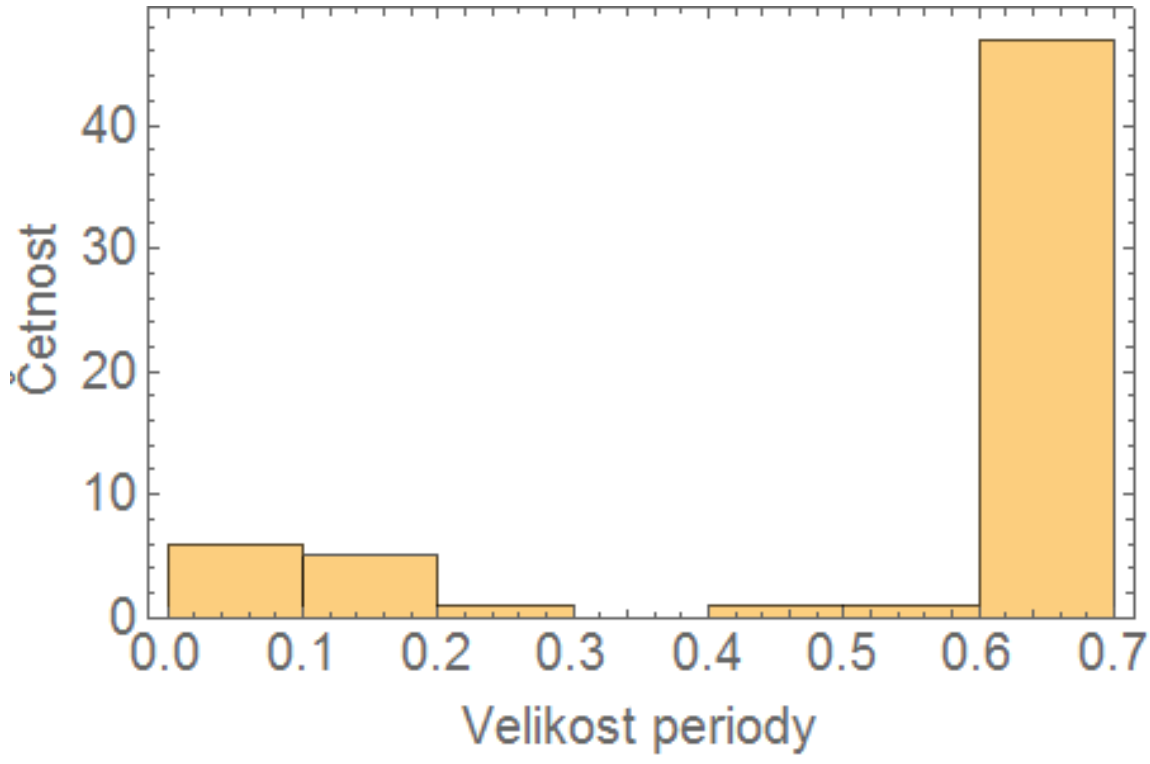
Pro identifikaci soustavy bude použit experimentální přístup, kdy experimentu bude podroben spojitý systém, reprezentovaný spojitou S-Funkcí v Simulinku. Pro účely snadnější simulace byl modelový systém časově přeškálován nastavením parametru $k_q = 0{,}075$. Dále je pro zachování škálovatelnosti předpokládána znalost okamžitého přítoku q_{in}, a proto je možné řídit systém přímo pomocí akumulace a. Autor si uvědomuje, že tento předpoklad nemusí být vždy splněn, nicméně v opačném případě by mohla být ztracena výhoda škálovatelnosti a aplikovatelnosti. Po zavedení tohoto opatření je nutné změnit také model systému, neboť omezení systému zavedením funkcí přepadu a dna (6.2.3) je uzpůsobeno pro omezení přítoku a odtoku. Proto je kladná akumulace považována za přítok a záporná za odtok.

Jelikož model popisující spojitý systém má diskrétní charakter, je potřeba určit vzorkovací periodu. To může být provedeno např. odhadem, změřením přechodové charakteristiky a určením hodnoty periody pomocí času ustálení či jinými přístupy, kdy je třeba uvažovat i další vlivy jako např. vliv nastavení periody na velikost časové predikce či výpočetní náročnost. Jelikož jsou v simulaci použity numerické metody derivace, mají i tyto metody omezený počet kroků. Velikost periody je v tomto případně proměnlivá a je automaticky určena podle aktuální situace. Toto je možné zobrazit pomocí časového průběhu výstupu spojitého systému, viz Obr. 6.24.



Obr. 6.24: Časový průběh výstupu spojitého systému s proměnlivou vzorkovací periodou, vlastní zdroj.

Perioda vzorkování se poté odhadne pomocí četnosti výskytu (Obr. 6.25).



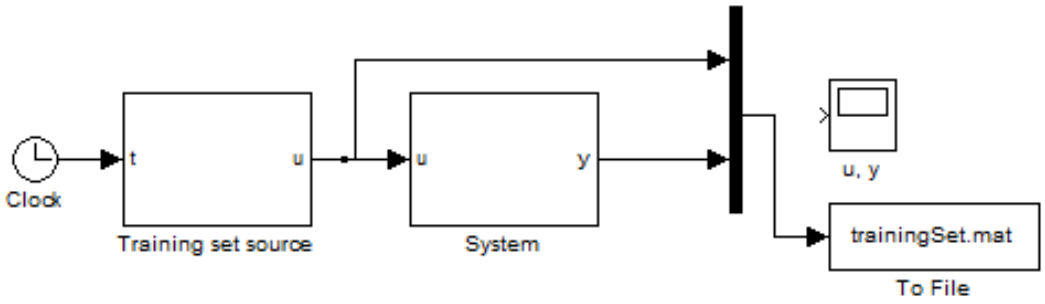
Obr. 6.25: Četnost výskytu velikosti periody časového průběhu spojitého systému, vlastní zdroj.

Perioda vzorkování by měla být taková, aby respektovala dynamiku systému. Zároveň by však měla být taková, aby byla časová velikost predikce dostatečná při zachování co nejmenší výpočetní náročnosti. Po zvážení těchto požadavků byla velikost periody nastavena na $\Delta t = 0{,}1$.

Mohlo by se zdát, že zvolením vzorkovací periody došlo opět k parametrové závislosti, avšak naměřená data mohou být dle potřeby převzorkovány tak, aby bylo možno použít nalezený model systému.

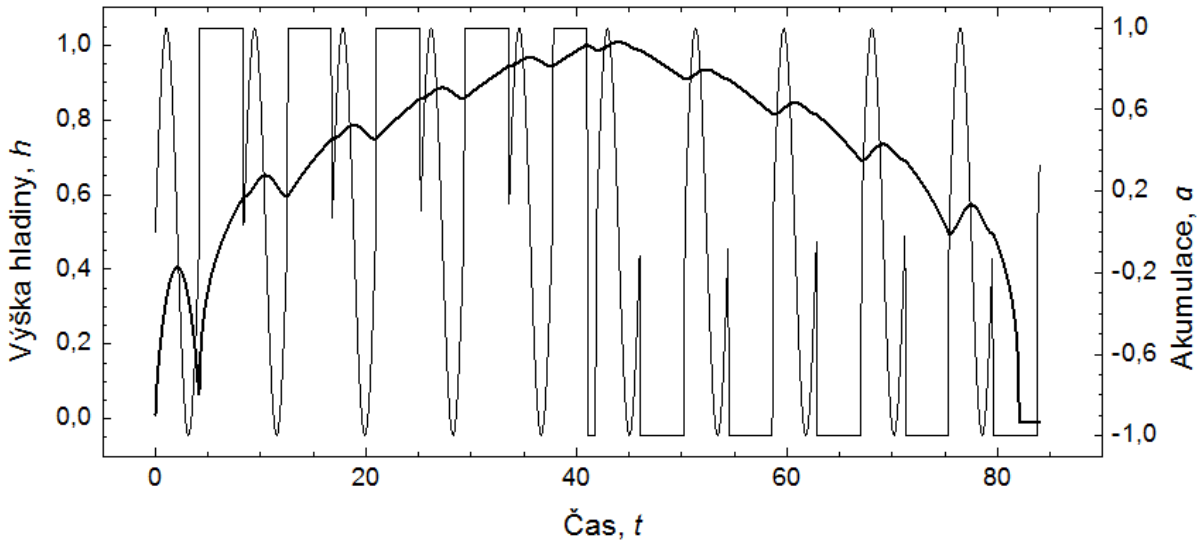
6.4.1 Trénovací množina

Pro účely další práce je nutné vytvořit trénovací (příp. testovací) množinu, viz diagram v Obr. 6.26. Ze své podstaty jsou tyto množiny vždy neúplné, neboť není možné provést nekonečné množství experimentů.



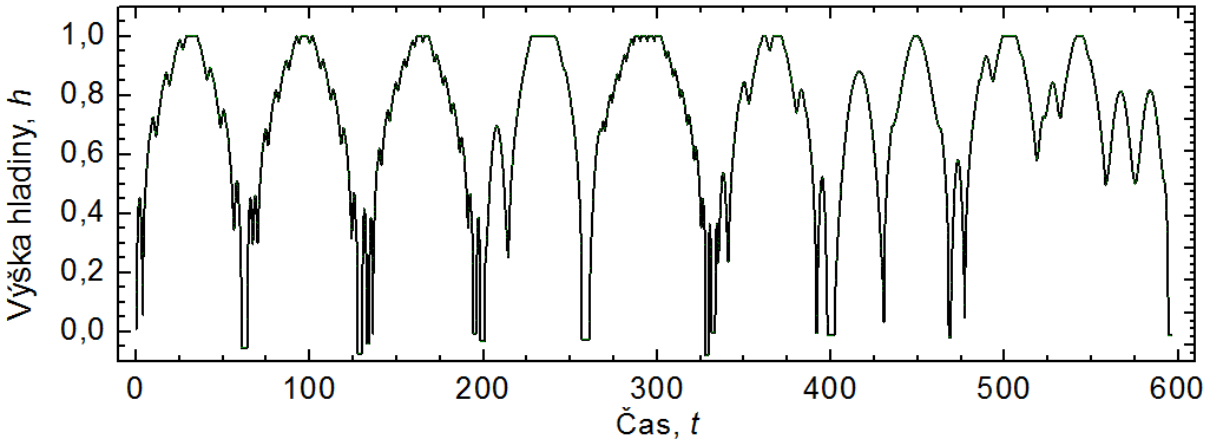
Obr. 6.26: Generování trénovací množiny, vlastní zdroj.

Nejjednodušší možností pro vytvoření trénovací množiny je použití náhodného vstupního signálu. Tento přístup však nemusí zajistit úplné pokrytí pracovní oblasti. Tvar vstupního signálu byl zvolen tak, aby byla zjištěna reakce systému na změny v různých úrovních pracovní oblasti (Obr. 6.27).



Obr. 6.27: Jeden cyklus trénovací množiny RBF NN, vlastní zdroj.

Výsledná trénovací množina (část zobrazena na Obr. 6.28) pak obsahuje množinu jednotlivých cyklů napuštění a vypuštění nádrže s různě rychlými změnami signálu. Maximální velikost trénovací množiny je omezena velikostí dostupné přidělené paměti.



Obr. 6.28: Část trénovací množiny pro RBF model systému (zobrazeno bez vstupu), vlastní zdroj.

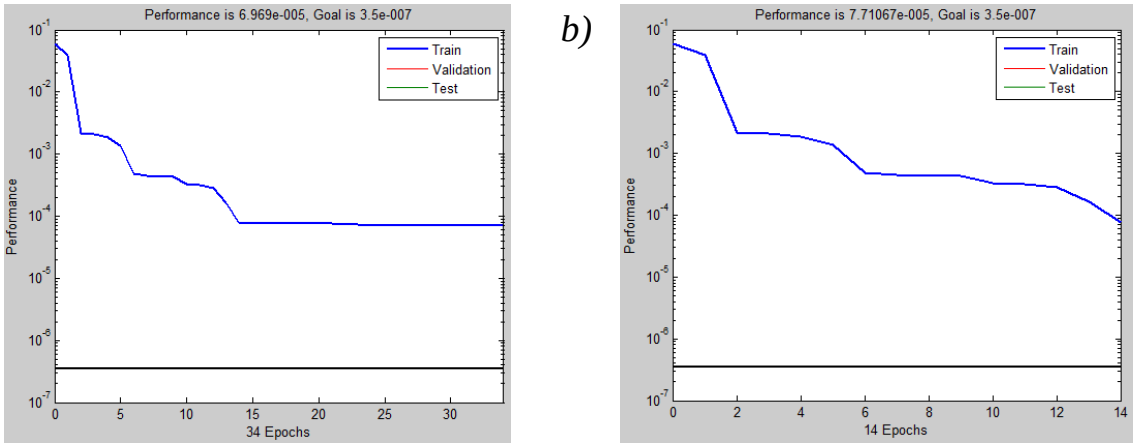
6.4.1 Trénování RBF modelu

Tato trénovací množina (či její část) je poté použita pro trénování neuronové sítě radiální báze. Cílem trénování je dosáhnout co nejmenší energetické funkce, zároveň je však třeba mít na paměti výpočetní náročnost (a s tím související optimalizaci) a proto je nutné volit parametry neuronové sítě (jako počet neuronů) tak, aby byly tyto požadavky splněny.

Pro trénování neuronové sítě byla vzolena funkce newrb(P,T,goal,spread,MN,DF), kde P je množina vstupů (viz Obr. 6.9), T je množina požadovaných výstupů, goal je požadovaná cílová hodnota energetické funkce, spread ovlivňuje rozptyl (normu) neuronů, MN je maximální množství neuronů a DF je počet neuronů pro zobrazení jednoho kroku energetické funkce. Maximální počet neuronů může být až do velikosti trénovací množiny (v tomto případě je každý prvek trénovací množiny popsán právě jedním neuronem), toto však není pro účely tvorby modelu vhodné.

Trénování je pak provedeno v několika fázích (Obr. 6.29). Jelikož energetická funkce obvykle konverguje k určité hodnotě (*a*), je nejprve nastaven větší maximální počet neuronů a experimentálně měněna hodnota spread. Ta by měla být nastavena tak, aby pokrytí výstupní funkce bylo dostatečně přesné a zároveň hladké. Poté je odečten minimální počet neuronů pro dosažení dostatečně dobré hodnoty energetické funkce a následně je provedeno nové trénování s nastaveným počtem neuronů (*b*).

a) b)

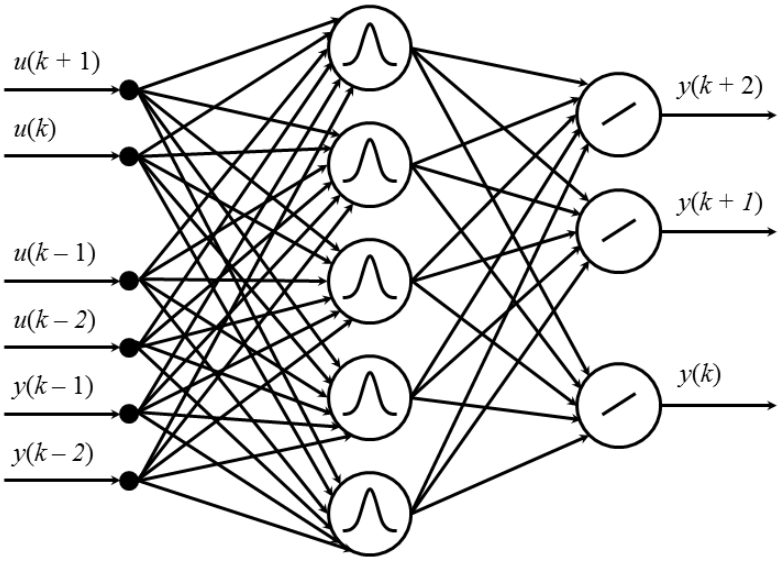


Obr. 6.29:Trénování RBF modelu: a) nalezení počtu neuronů, b) trénování, vlastní zdroj.

Stejný postup je poté aplikován na ostatní modelové systémy, kdy každý tento systém je pak popsán svým RBF modelem. Po natrénování je síť možné otestovat. Jelikož je však systém kumulačního charakteru, není možné testovat shodu sítě se systémem na dlouhé časové úseky (motýlí efekt[92]). Proto byl tento krok vypuštěn a kvalita sítě může posouzena podle kvality regulace. Naučená síť byla vždy uložena jako objekt do souboru, aby bylo možné přistoupit k ní z různých bloků v rámci simulačního prostředí Simulink.

6.5 Vytvoření prediktoru

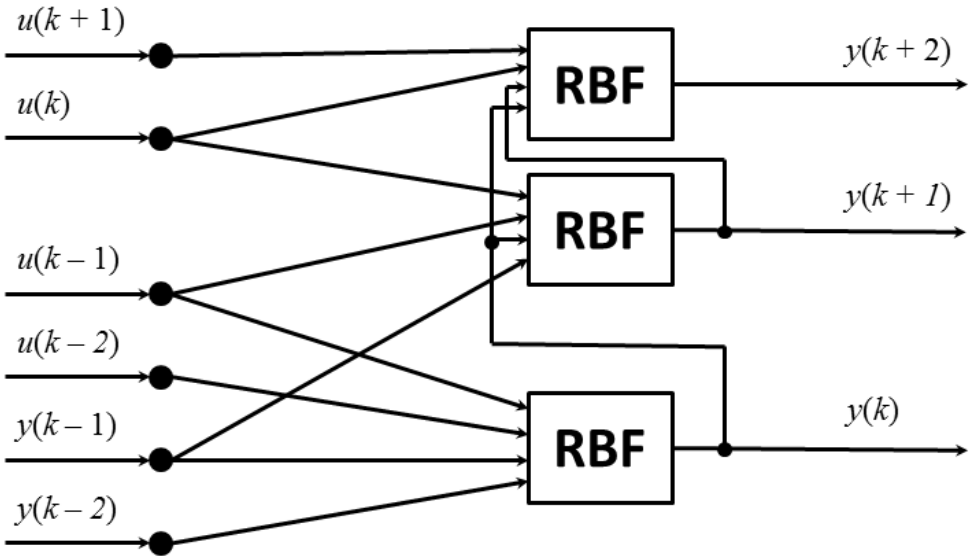
K tvorbě prediktoru lze využít nejméně dvou přístupů. Prvním je rozšíření samotného modelu[98] systému. Základní myšlenkou tohoto přístupu je využití trénovacího procesu pro přímé vytvoření prediktoru (Obr. 6.30).



Obr. 6.30: RBF prediktor pomocí rozšíření modelu, vlastní zdroj.

Výhodou tohoto přístupu by mohlo být zmenšení výpočetní náročnosti predikce, a tedy i optimalizace, neboť počet neuronů by mohl zůstat relativně malý v porovnání s druhým přístupem (viz níže). Nevýhodou by pak mohl být složitější proces trénování z důvodu zahrnutí nejenom volné, ale také vynucené odezvy. Další nevýhodou je pak nutnost změny struktury a nové přetrénování prediktoru pokaždé, když jsou změněny horizonty, neboť počet vstupů a výstupů je poté na těchto horizontech závislý.

Druhým přístupem je pak rekurzivní přístup (Obr. 6.31), kdy je model systému považován za funkci, kterou lze opakovaně volat jako proceduru.



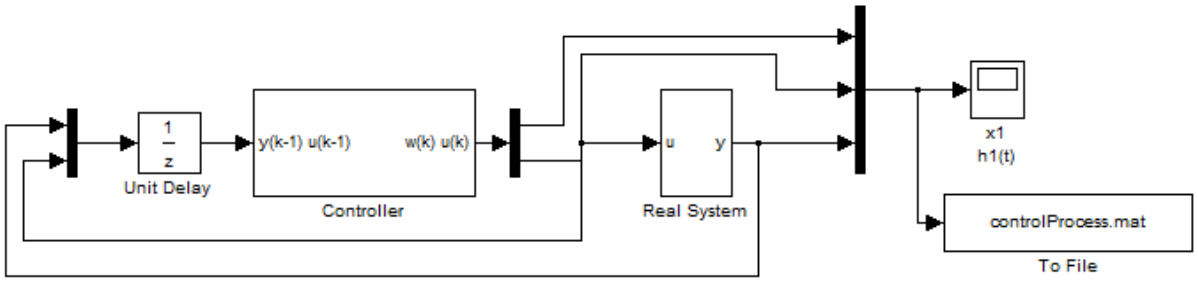
Obr. 6.31:RBF prediktor - rekurzivní, vlastní zdroj.

Výhodou tohoto přístupu je snadná změna prediktivního horizontu. Nevýhodou může být zvýšená výpočetní náročnost predikce (a optimalizace) z důvodu opakování struktury neuronové sítě. U dostatečně pomalých procesů však může být perioda optimalizace nastavena dostatečně velká, aby bylo možno tento přístup použít.

Prediktor byl tedy realizován rekurzivně jako procedura v rámci S-Funkce regulátoru.

6.6 Optimalizace řízení

Regulátor se skládá ze dvou částí – prediktoru a optimalizátoru. Optimalizátor využívá prediktoru k predikci chování systému. Velikost horizontu byla zvolena následovně $\{N_1, N_2, N_u\} = \{1, 3, 2\}$, pro dosažení nízké výpočetní náročnosti. Proces regulace byl simulován v prostředí Simulink, viz Obr. 6.32.



Obr. 6.32: Simulace prediktivního řízení modelových systémů, vlastní zdroj.

Optimalizátor využívá fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options), to je funkci, kde význam jednotlivých argumentů je následující:

- *fun* – účelová funkce, která má být optimalizována
- *x0* – počáteční jedinec (seed)
- *A, b* – matice lineárních nerovnic omezení (6.9)
- *Aeq, beq* – matice lineárních rovnic omezení
- *lb, ub* – spodní a horní hranice proměnné
- *nonlcon* – nelinární omezení
- *options* – nastavení nelineární optimalizace

Účelová funkce byla upravena tak, aby bylo dosaženo lepší kvality regulace zavedením dalšího váhového parametru

$$fun = \lambda_e(\hat{\boldsymbol{y}} - \boldsymbol{w})^T(\hat{\boldsymbol{y}} - \boldsymbol{w}) + \lambda_{\tilde{u}}\tilde{\boldsymbol{u}}^T\tilde{\boldsymbol{u}} \tag{6.49}$$

kde λ_e je váhový koeficient regulační odchylky a $\lambda_{\tilde{u}}$ je váhový koeficient změny akčního zásahu.

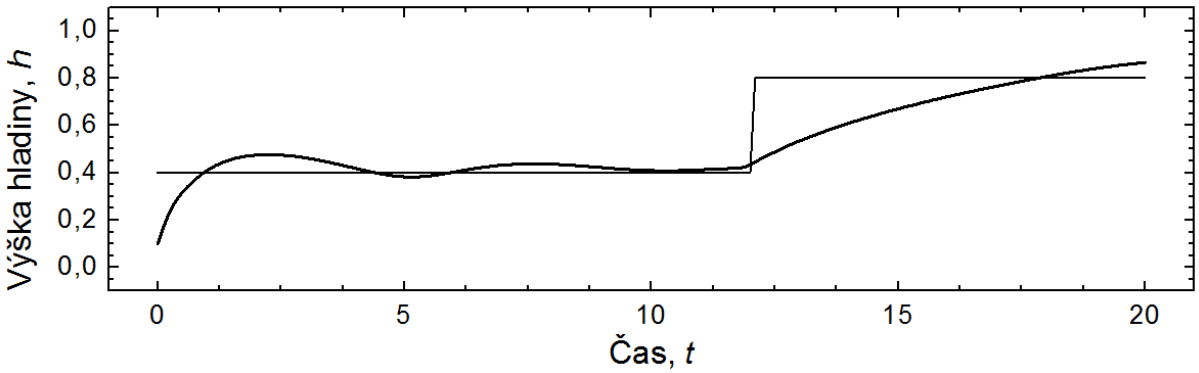
Do *nonlcon* byla zahrnuta nelineární omezení tak, aby uvnitř dosažitelné oblasti byl tento výraz menší než nula a hodnota výrazu určovala kvalitu splnění omezení tak, že s větší vzdáleností od této oblasti absolutní hodnota výrazu roste.

V rámci parametru *options* byl vybrán optimalizační algoritmus, kde na výběr jsou možnosti *interior-point*, *sqp-legacy*, *active-set* a *trust-region-reflective*. Experimentálně se osvědčily poslední 2 možnosti. První možností je použití algoritmu *active-set*, což je metoda kvadratického programování, která využívá odhadu Hessiánu pomocí algoritmu BFGS (Broyden–Fletcher–Goldfarb–Shanno). BFGS je kvazi-Newtonovská metoda, která spadá do třídy Hill Climbing technik. Druhou možností je pak použití algoritmu *trust-region-reflective*, který je založen na myšlence aproximace účelové funkce

v místě aktuálního jedince funkcí jednodušší, která dobře odráží charakter funkce v jeho okolí. V rámci aproximované funkce (*trust-region*) je pak provedena suboptimalizace.

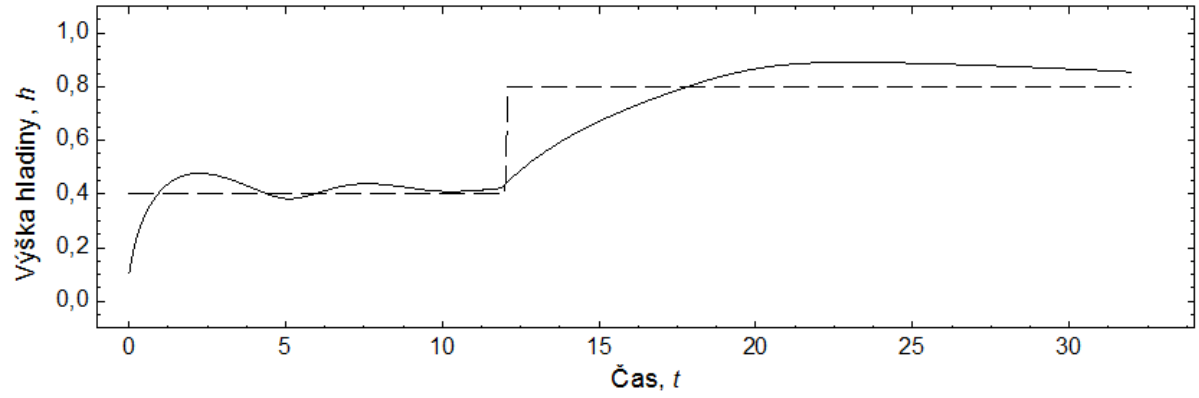
6.6.1 Řízení modelových systémů

Pilotní experimenty byly z důvodu vyšší výpočetní náročnosti provedeny pro kratší časové úseky než simulace pozdější. Bylo provedeno několik úprav, jak v oblasti trénovací množiny, tak v oblasti regulátoru.



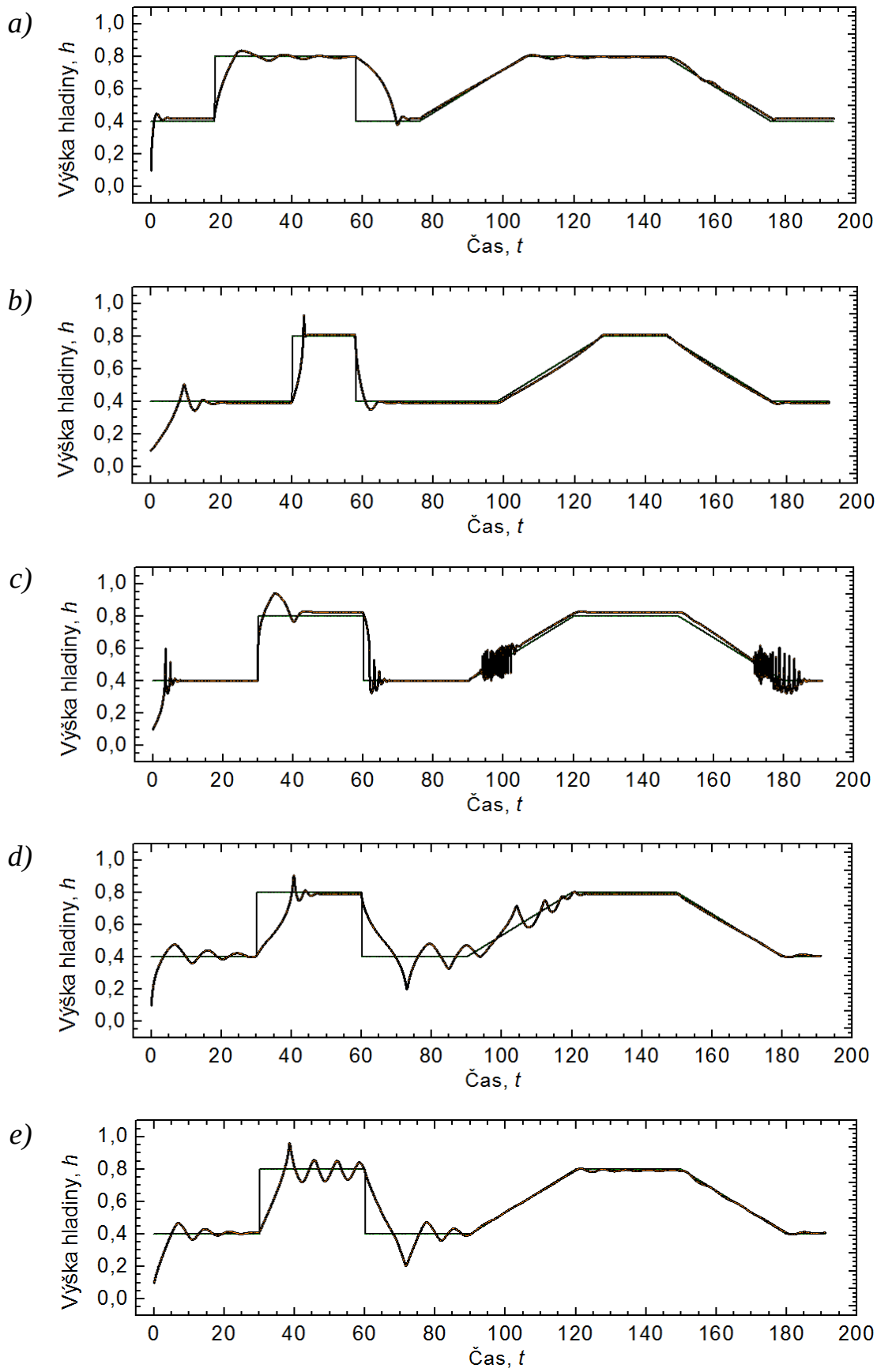
Obr. 6.33: Jeden z pilotních experimentů měření časového průběhu regulace, vlastní zdroj.

Zprvu se zdálo (Obr. 6.33), že regulační proces není úspěšný. Nicméně po bližším zkoumání bylo zjištěno, že regulace odpovídá charakteru modelového systému (kuželu). Jak je patrné z obrázku Obr. 6.34, tak modelový systém má různou dynamiku v různých bodech pracovní oblasti. Plnění/vypouštění v oblasti s nižší hladinou je výrazně rychlejší než plnění/vypouštění s hladinou v horní části oblasti. Toto je způsobeno kuželových tvarem a různou plochou horizontálního průřezu v různých místech nádrže. Po zvážení těchto aspektů byla simulační doba prodloužena.



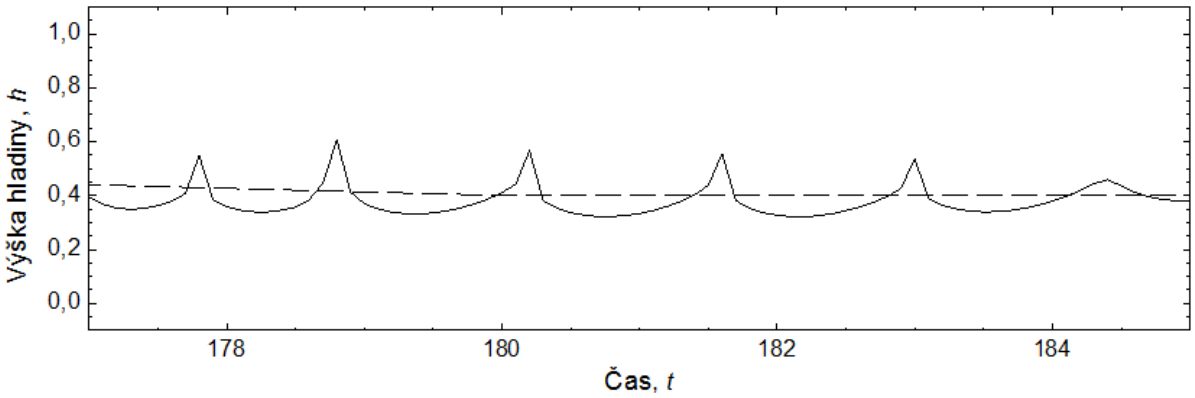
Obr. 6.34: Pilotní experiment – dynamika systému v různých úrovních, vlastní zdroj.

Po odladění optimalizačního procesu byla provedena simulace pro pět modelových systémů, výsledky jsou zobrazeny na Obr. 6.35.



Obr. 6.35: Časový průběh řízení výšky hladiny pro a) kužel, b) komolý kužel, c) dvojkužel s úzkým hrdlem, d) dvojkužel a e) kouli, vlastní zdroj.

Obr. 6.35 ukazuje časový průběh řízení výšky hladiny pro modelové systémy. Modelový systém válec byl záměrně vynechán, kvůli své lineární povaze. Optimalizátor využíval algoritmu *trust-region-reflective*. Na časových průbězích je velmi patrná změna dynamiky systému v různých úrovních pracovní oblasti. Zajímavý je také průběh (*c*) pro řízení modelového systému s úzkým hrdlem, kde je velmi patrná střední oblast obsahující singularitu. Přestože se může zdát, že u střední části není řízení moc dobré, je nutné si uvědomit, že jde o matematický modelový systém, kde rychlost proudění tekutiny ve střední části systému může být neomezená. Vzhledem k tomuto faktu je řízení ve střední části relativně dobré, jak je zřejmé z detailního zobrazení na Obr. 6.36.



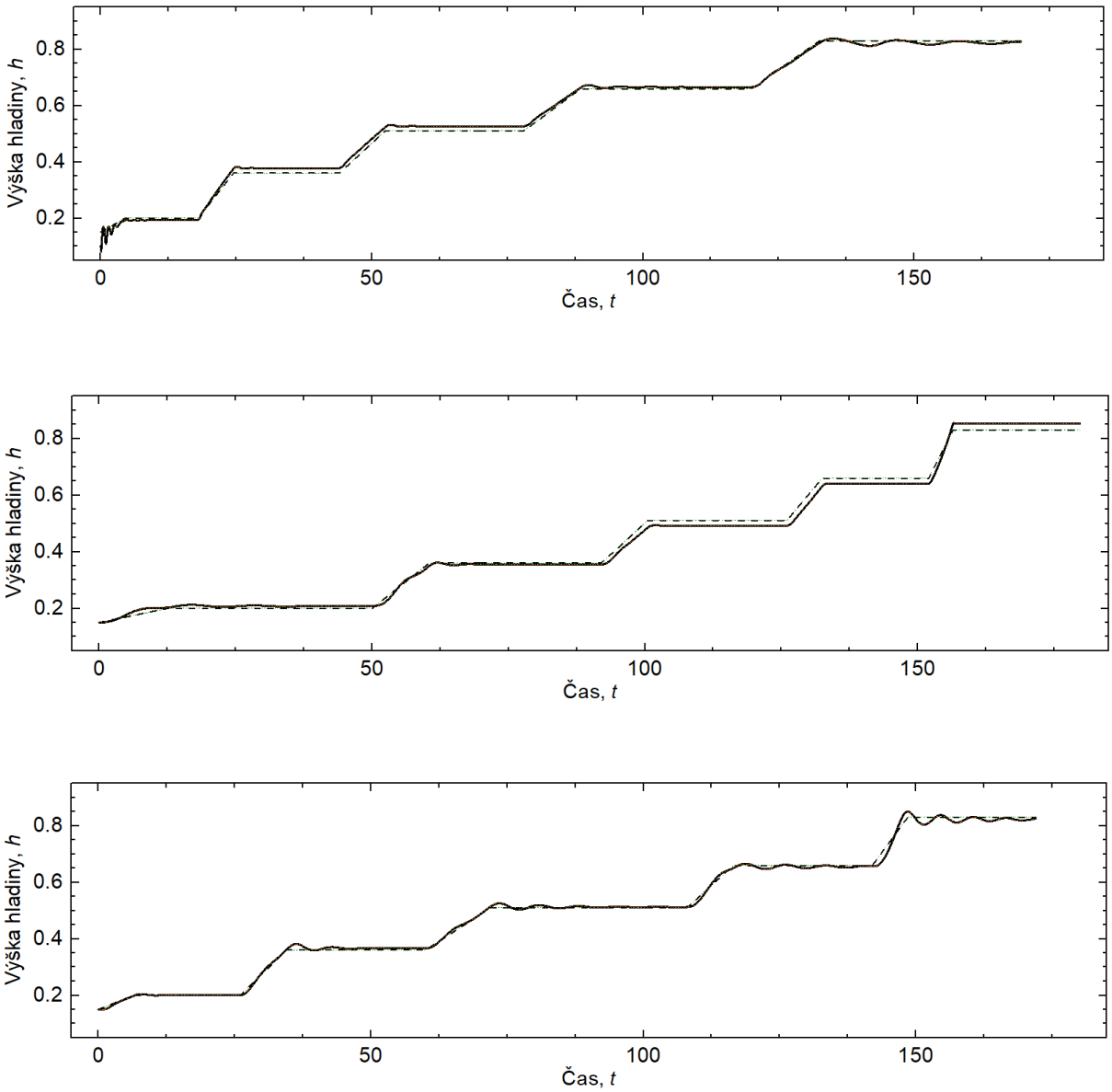
Obr. 6.36: Řízení modelového systému s úzkým hrdlem v oblasti singularity, vlastní zdroj.

Průběhy *d*) a *e*) jsou podle předpokladů podobné, neboť se podobají i tvarově. Je možné si povšimnout kmitavého chování v okolí rychlých změn žádané hodnoty. Pokud se navíc stav systému nachází blízko hranic/singularit, může docházet k poměrně velkým výkyvům. Tomuto jevu lze předejít poměrně snadno úpravou žádané hodnoty tak, aby její změna byla pozvolnější. Poté je možné dosáhnout relativně dobrého řízení.

Všechny systémy vykazují drobné regulační odchylky od ustáleného stavu. To může být způsobeno tím, že se optimalizace řízení provádí pomocí modelu systému, který však nemusí popisovat reálný systém úplně přesně. Dalším důvodem může být nastavení regulátoru (a jeho schopnosti optimalizace), a také fakt, že v simulinku nelze pro komplexní modely jako je neuronová síť zapojit aktuální hodnotu výstupu přímo na vstup, neboť toto vede k algebraické smyčce, kterou lze řešit pouze u systémů snadno popsatelných. Překonat problém trvalé regulační odchylky by bylo možné např. dalším zpřesněním modelu tj. použitím více neuronů s nižší hodnotou rozptylu, úpravou trénovací množiny (např. zahrnutím získaných dat do množiny a

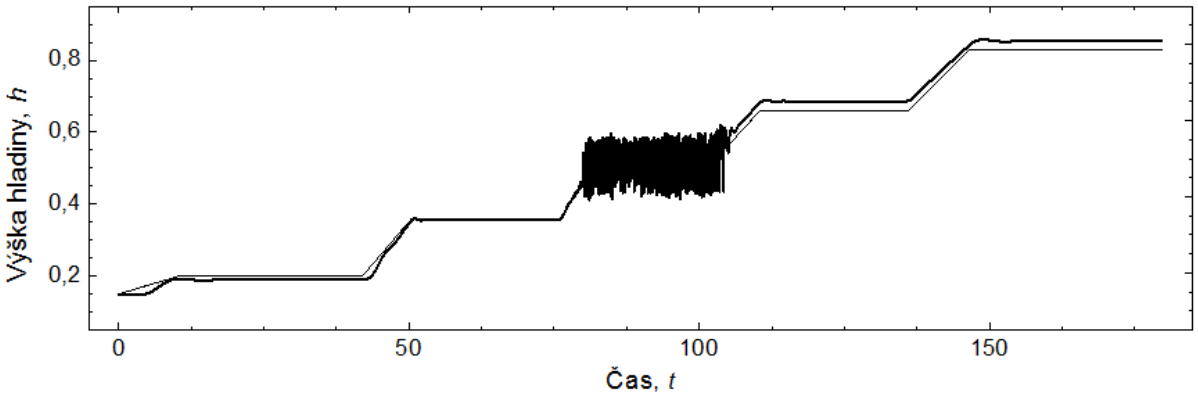
doučením sítě), úpravou regulátoru, korekcí vstupní, výstupní či žádané veličiny, či zavedením adaptivního přístupu.

Pro účely simulace byl zvolen přístup korekce žádané hodnoty. Hlavní myšlenkou je využití již získaného modelu, pro který bude provedena kalibrace na ustálené hodnoty (Obr. 6.37). Žádaná hodnota tedy byla zkonstruována tak, aby byly změřeny ustálené hodnoty regulace v různých místech pracovní oblasti. Tvar žádané hodnoty byl navíc upraven tak, aby se předešlo velkým výkyvům (jak bylo zmíněno výše).



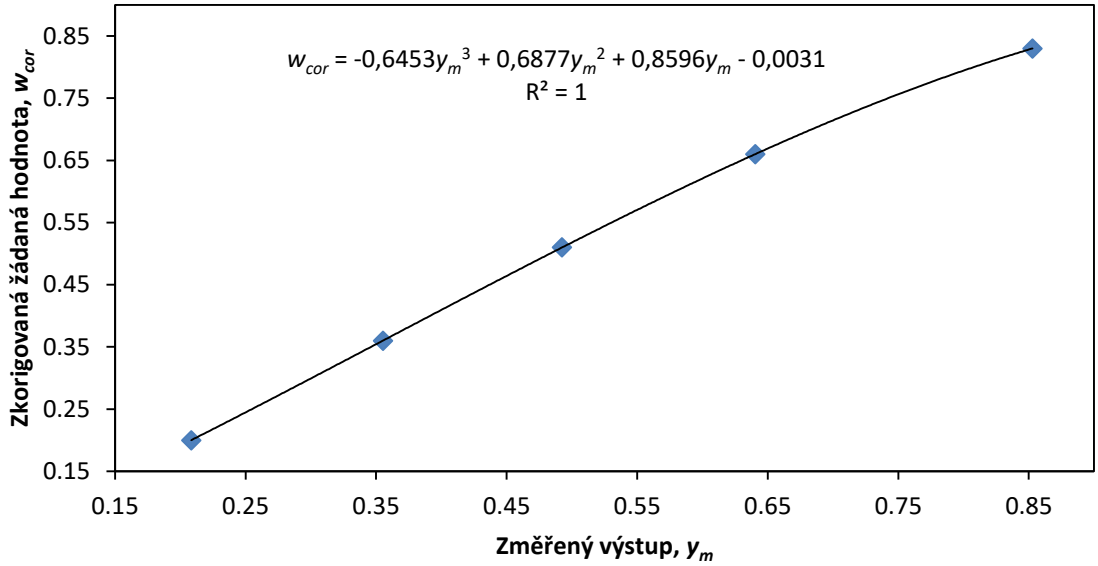
Obr. 6.37: Měření ustálených hodnot řízení – kužel (nahoře), komolý kužel (uprostřed) a koule (dole), vlastní zdroj.

U modelového systému s úzkým hrdlem (Obr. 6.38) nelze hodnotu v oblasti singularity použít.



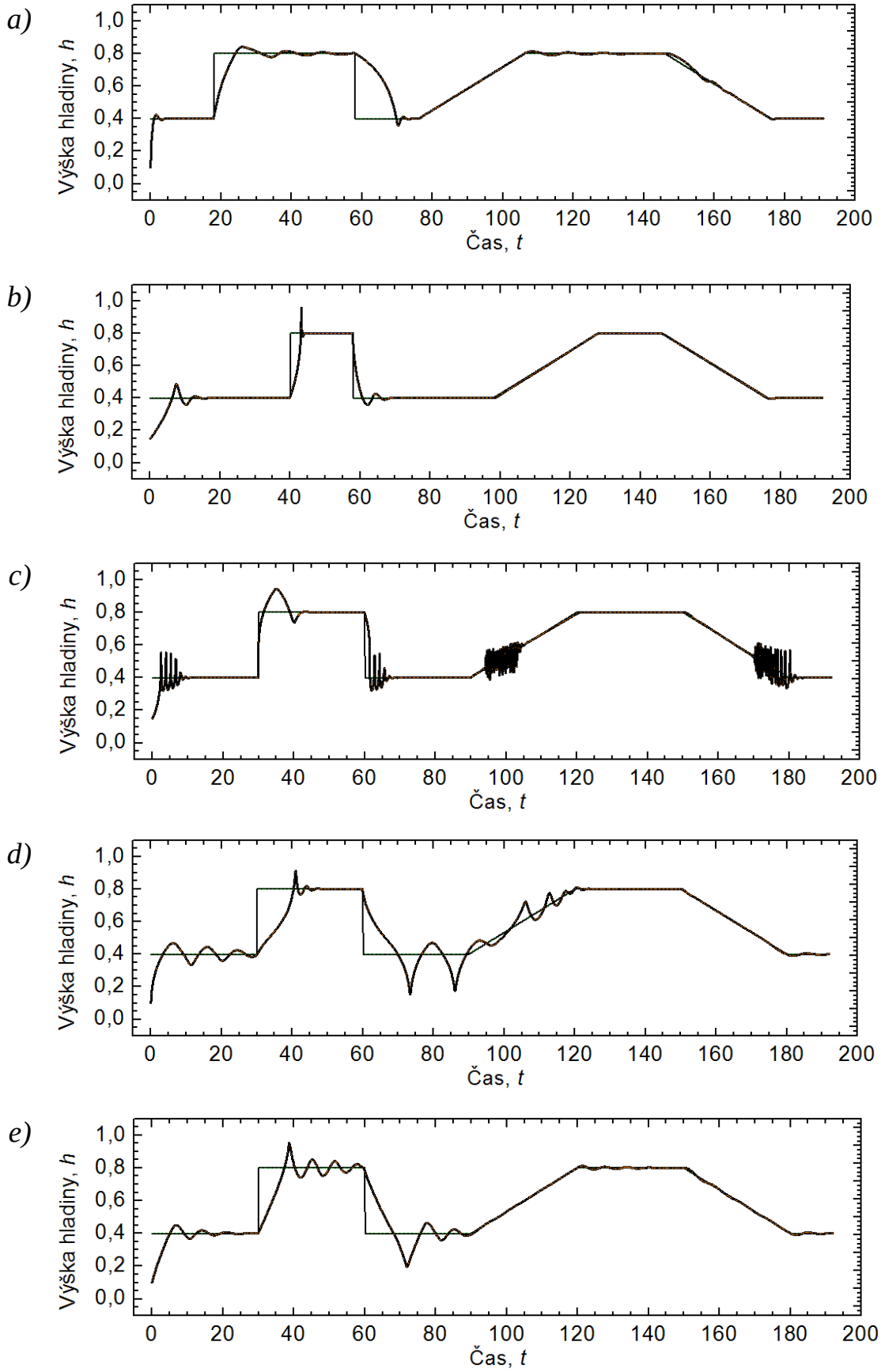
Obr. 6.38: Měření ustálených hodnot řízení – systém s úzkým hrdlem, vlastní zdroj.

Po změření ustálených hodnot lze vytvořit korekční funkci žádané hodnoty (Obr. 6.39).



Obr. 6.39: Určení korekční funkce žádané hodnoty - příklad pro model komolého kuželu, vlastní zdroj.

Takto jsou vytvořeny korekční křivky pro všechny modelové systémy. Výhodou použití korekčních křivek (či tabulek) může být fakt, že je možné tyto křivky postupně upravovat zahrnutím nových měření řízených ustálených hodnot regulačního procesu. Dále je možné využít jednodušší verze RBF modelů (méně neuronů) bez nutnosti přetrénování modelu. Všechny simulace byly opět zopakovány, tentokrát s upravenou žádanou hodnotou (Obr. 6.40). Z časových průběhů lze pozorovat, že ustálené hodnoty výstupů modelových systémů nyní odpovídají požadované hodnotě.



Obr. 6.40: Časový průběh řízení výšky hladiny pro modelové systémy s upravenou žádanou hodnotou, vlastní zdroj.

6.7 Návrh přenosu získaných výsledků na reálný systém

Získané modely jsou aplikovatelné na reálné systémy pouze za určitých podmínek. Odvozené modely jsou zcela platné pouze při splnění uvedených předpokladů. Diferenciální rovnice popisující systémy obsahují veličiny přítoku a odtoku, kdy přítok minus odtok je považován za akumulaci, přičemž je zanedbán hydrostatický tlak, a fyzikální limity proudění kapalin vůbec. Předpokládá se tedy, že tyto veličiny lze přímo řídit, či alespoň měřit, což lze v praxi provést např. pomocí regulátorů, čerpadel či měřičů průtoku. Dále je třeba si uvědomit, že odvozené modelové systémy byly non-dimenzionalizovány. Proto je nutné reálné rozsahy přemapovat tak, aby byly v rozsazích, platných pro modelové systémy.

Pro použití naučených neuronových sítí je nejdříve nutné změřit dobu naplnění/vypuštění nádrže. Neuronové sítě byly trénovány na časově přeškálovaných modelových systémech s koeficientem $k_q = 0{,}075$. Čas naplnění/vypuštění modelového systému při maximálním přítoku/odtoku je tedy převrácená hodnota této konstanty.

$$k_q = \frac{1}{\Delta t_{max}} \quad \Rightarrow \quad \Delta t_{max} = \frac{1}{0{,}075} = 13,\overline{3} \tag{6.50}$$

Vzorkovací perioda v simulacích byla zvolena $\Delta t = 0{,}1$. Vzájemným poměrem těchto hodnot je možné získat poměr hodnot reálných. V případě jiné reálné vzorkovací periody, než odpovídá poměru 1:133, je nutné převzorkovat na frekvenci odpovídající tomuto poměru. Alternativně je možné provést přetrénování modelu pro novou periodu, což je ovšem mnohem náročnější postup, a jeho nasazení by bylo praktické, pouze pokud je reálná vzorkovací frekvence nižší, než je třeba.

Při aplikaci modelu na reálné systémy bude potlačen výskyt scénářů selhání spojených s výskytem singularit. Na druhou stranu, situace spojené se saturací jsou v praxi řešeny reálně přepadem, nebo uzavřením výpusti.

Uvedené modely lze použít pro systémy, které jsou tvarově příbuzné modelovým systémům, jejichž výběr ovšem pokrývá významné množství případů. Nejlépe by bylo vždy vhodné modely natrénovat přímo z dat reálného systému. K tomu je však zapotřebí velkého množství experimentů, a to v celém rozsahu pracovní oblasti, popřípadě disponovat dlouhou historickou datovou řadou, z níž by bylo možné trénovací množinu vybrat či doplnit. Vzhledem k nedostupnosti reálných dat je aplikace těchto modelů na reálné systémy nad rámec této disertační práce. Přesto lze oprávněně formulovat následující pravidla odhadu:

1. Doba naplnění/vypuštění nádrže je pokryta škálováním času kompletně.
2. Objem nádrže je pokryt škálováním kompletně.
3. Korekční křivka žádané hodnoty je pokryta škálováním kompletně.
4. Model platný pro jedno těleso (nádrž) vždy platí i pro všechna podobná tělesa (nádrže).
5. Efekt škálování je ovšem širší, než předchozí bod, neboť z rovnic vypadl přepočtový faktor nádrže (tj. poměr průměru a výšky), a tedy na tomto poměru při daném tvaru rovnice rychlosti změny výšky hladiny nezáleží, což platí tehdy, pokud je objem nádrže lineárně závislý na tomto poměru. Jinými slovy řečeno, každý objem je udán součinem tří rozměrů tělesa v na sebe navzájem kolmých směrech (např. $a.b.c$). Pokud jeden závisí na druhých dvou, tak přepočtový faktor nádrže není pro přenos modelu do praxe v rámci stejného typu geometrie důležitý.
6. Pokud je tvar zcela nezávislý ve všech třech rozměrech, uplatní se fakt, že ze všech poměrů mezi výškou a průměrem byl u použitých modelových systémů volen ten, který uzavírá relativně největší objem; je to poměr 1:1, který představuje nejmenší extrém z obou stran, a dá se se tudíž považovat za dobrý střední odhad, což platí i pro tvary složitější než pláště válce, kuželů, koule a ostatních centrosymetrických a rotačně symetrických těles, pokud si vybereme tvarově správný – odpovídající – typ modelu.
7. V praxi bude také zapotřebí znát a použít přepočtovou funkci mezi reálnou akční veličinou (např. výškou vytažení stavidla, úhlem otočené kohoutem, atd.) a akční veličinou modelovou, tj. musí být známa inverzní charakteristika daného zařízení (výpustě).
8. Efekt hydrostatického tlaku, je stejný v nádrži jakéhokoliv tvaru a závisí jen a pouze na výšce, a je zpětně implementovatelný do regulátoru, až by byly známy konkrétní rozměry nádrže, pokud již nebude implicitně zahrnut v přepočtové funkci regulačního zařízení.

Jako jedna z hlavních nevýhod prvků umělé inteligence se uvádí velká výpočetní náročnost způsobená použitím mnoha jednoduchých elementů, jako jsou neurony v případě neuronových sítí, či jedinci v případě evolučních algoritmů. Kritickou hodnotou je reálná doba vzorkovací periody, tedy 1/133 doby, za kterou se nádrž napustí/vypustí při maximální/minimální hodnotě akumulace. V případě nutnosti by jednou z dalších cest optimalizace regulačního procesu mohla být paralelizace nezávislých prvků umělé

inteligence, což by snížilo výpočetní náročnost a umožnilo nasazení i na procesy s rychlejší dynamikou. Při dostatečném počtu jader/vláken by pak v ideálním případě nemuselo záležet na velikosti množiny (počtu neuronů či jedinců) a výpočetní náročnost modelu by se pak zredukovala pouze na počet průchodů (vrstev neuronové sítě, iterací evolučního algoritmu). Toho může být dosaženo použitím paralelního výpočtu při simulaci modelu, případně využitím jader grafického procesoru.

7. ZÁVĚR A PŘÍNOS PRÁCE PRO VĚDU A PRAXI

V experimentální části práce byl vytvořen algoritmus využívající principů klasických i evolučních metod a porovnán s klasickou metodou kvadratického programování. Bylo ukázáno, že může být dosaženo podobné výpočetní náročnosti (v závislosti na nastavení) jako u metody QuadProg(). Později bylo dokonce zjištěno, že výpočetní náročnost by mohla být dále snížena pomocí paralelizace, nicméně řešení této otázky je již nad rámec této disertační práce. Algoritmus přesto obstál i v neparalelizované verzi.

Dále byly v rámci disertační práce odvozeny matematické tvary modelových systémů (nádrží), jejichž tvar je podobný tvarům systémů vyskytujících se v praxi a v přírodě. Uvedené modelové systémy byly navíc non-dimenzionalizovány, což umožňuje jejich škálovatelnost a nasazení na různé typy systémů s různými rozsahy při splnění podmínek pro jejich aplikovatelnost. Dále byly vytvořeny modely systémů založené na neuronové síti RBF. Byly vytvořeny trénovací množiny a parametry těchto modelů byly určeny pomocí trénování na modelových systémech. Dále byly určeny jejich kalibrační křivky pro ustálené hodnoty výstupu. Jelikož původní modelové systémy byly non-dimenzionalizovány a škálovatelné, mají také vytvořené modely i kalibrační křivky tyto vlastnosti. Při splnění podmínek aplikovatelnosti je tedy možné tyto modely nasadit na řízení systémů reálných. Vzhledem k nedostupnosti dat a nemožnosti experimentu s reálným systémem např. vodní nádrže, jsou v poslední kapitole disertační práce analyzovány možnosti přenosu získaných modelů do praxe a formulovány zásady, jak postupovat, čímž byl splněn poslední z cílů práce.

8. ZÁVĚREČNÉ SHRNUTÍ

Tato práce se zabývá možnostmi nasazení některých prvků umělé inteligence ve specifických částech prediktivního řízení. Jedná se především o využití neuronových sítí jako modelu systému a evolučních algoritmů v rámci optimalizace řízení.

Práce je rozdělena do sedmi hlavních kapitol, kdy v první kapitole jsou nastíněné základní principy prediktivního řízení a ve druhé kapitole je provedena rešerše současného stavu problematiky nelineárních systémů ve vztahu k výpočetní komplexitě, prediktivnímu řízení a aplikaci prvků umělé inteligence, přičemž hlavní část je věnována oblasti řízení hladiny kapaliny v nádržích, která byla vybrána jako vhodná pro řešení daného zadání. Ze souhrnu rešerše vyplynula obecná formulace úkolů práce.

Následuje třetí kapitola, která poskytuje teoretický rámec, kde v rámci prediktivního řízení jsou popsány jednotlivé jeho prvky, jako jsou obvykle používané modely, tvorba prediktoru a optimalizátoru a základní metody optimalizace. V druhé části teoretického rámce jsou popsány prvky umělé inteligence zejména struktura a vytvoření neuronové sítě Perceptron, Multi-layer Perceptron a síť založená na funkcích typu radiální báze RBF. Dále je pak stručně popsán princip evolučních algoritmů a jejich stručný přehled.

Čtvrtá kapitola popisuje hlavní cíl práce a jeho dílčí cíle, kdy hlavním cílem práce je aplikace některých prvků umělé inteligence ve vhodných oblastech prediktivního řízení. Dílčí cíle pak popisují jednotlivé kroky, které jsou nutné pro splnění cíle hlavního. Těmito dílčími cíli jsou předběžná analýza a ověření použití prvků umělé inteligence, volba vhodných modelových systémů, volba a vytvoření modelu, identifikace soustavy a nalezení parametrů modelu, vytvoření prediktoru a optimalizátoru a návrh přenosu poznatků na řízení systémů reálných.

Pátá kapitola se zabývá experimentální a metodickou částí, kde jsou popsané použité SW nástroje, algoritmy a funkce. Dále je popsán metodický postup řešení jednotlivých dílčích cílů.

Nejobsáhlejší šestá kapitola se zabývá splněním jednotlivých cílů práce. Nejprve je provedena předběžná analýza a ověření použití prvků umělé inteligence. V rámci tohoto dílčího cíle byla provedena analýza účelové funkce a vlivu omezení na dosažitelnou oblast, ověření možnosti nasazení evolučního algoritmu Hill Climbing a algoritmu evolučně-gradientního v rámci optimalizátoru a ověřením možnosti nasazení neuronové sítě jako modelu a její verifikace v rámci zvolené pracovní oblasti. Druhý dílčí cíl se zabývá odvozením tvarů diferenciálních rovnic modelových systémů a jejich

úpravou pro účely simulace (saturace, singularity, aplikovatelnost). Dále je popsána volba a vytvoření RBF modelu systému. Čtvrtá část se zabývá určením parametrů modelu, tj. volbou periody, tvorbou trénovací množiny a samotným trénováním neuronové sítě. Další podkapitola popisuje výběr a tvorbu prediktoru a jejich možnosti nasazení. Šestá podkapitola popisuje tvorbu regulátoru a prezentuje výsledky experimentů řízení pro všechny modelové systémy (kromě systému typu válec) a úpravu řízení pomocí korekčních křivek. Poslední podkapitola se zabývá návrhem přenosu získaných poznatků pro řízení systémů reálných a podmínkami, které je nutné splnit pro možnost nasazení těchto modelů a přístupů.

Sedmou kapitolu představuje věcný závěr provedené studie a souhrn přínosů práce pro vědu a praxi.

Hlavním cílem disertační práce byla aplikace některých prvků umělé inteligence ve vhodných oblastech prediktivního řízení. Cíle práce bylo dosaženo.

SEZNAM POUŽITÉ LITERATURY

[1] RAWLINGS, James B. a David Q. MAYNE. *Model predictive control: theory and design.* Madison, Wis.: Nob Hill Pub., 2009. 533 s. ISBN 9780975937709.

[2] KWON, W. H. a S. HAN. *Receding horizon control: model predictive control for state models.* Berlin ; London: Springer, 2005. 380 s. ISBN 9781846280245.

[3] CAMACHO, E. F. a C. BORDONS. *Model predictive control.* London ; New York: Springer, 2003. 405 s. ISBN 1852336943.

[4] WANG, Liuping. *Model predictive control system design and implementation using MATLAB.* London: Springer, 2009. 375 s. ISBN 9781848823303.

[5] KHALED, Nassim a Bibin PATTEL. *Practical design and application of model predictive control: MPC for MATLAB® and Simulink® users.* Kidlington, Oxford: Butterworth-Heinemann, an imprint of Elsevier, 2018. ISBN 9780128139196.

[6] KLUEVER, Craig A. *Dynamic systems: modeling, simulation, and control.* Hoboken, NJ: John Wiley and Sons, Inc., 2015. ISBN 9781118289457.

[7] VÍTEČEK, Antonín a Miluše VÍTEČKOVÁ. *Optimální systémy řízení.* 1. vyd. Ostrava: Vysoká škola báňská - Technická univerzita Ostrava, Strojní fakulta, 1999. 158 s. ISBN 8070787368.

[8] ROSSITER, John. *Model-based Predictive Control: A Practical Approach.* CRC Press, 2003.

[9] MIKLEŠ JÁN, Fikar M. *Modelovanie, identifikácia a riadenie procesov 2. Identifikácia a optimálne riadenie.* STU Press, Bratislava, 2004. 260 s. ISBN 80-227-2134-4.

[10] BOBÁL, Vladimír a Univerzita Tomáše Bati ve Zlíně Fakulta aplikované informatiky. *Adaptivní a prediktivní řízení.* Vyd 1. Zlín: Univerzita Tomáše Bati ve Zlíně, 2008. 134 s. ISBN 978-80-7318-662-3.

[11] BOBÁL, Vladimír a Univerzita Tomáše Bati ve Zlíně Fakulta aplikované informatiky. *Identifikace systémů.* Vyd 1. Zlín: Univerzita Tomáše Bati ve Zlíně, 2009. 128 s. ISBN 978-80-7318-888-7.

[12] DU, K. -. a M. N. S. SWAMY. *Neural networks in a softcomputing framework.* London: Springer, 2006. 566 s. ISBN 1846283027.

[13] Kröse Ben, van der Smagt Patrick. *An introduction to Neural Networks.* Eight Edition. The University of Amsterdam, 1996.

[14] Lee Gue Myung, Tam N.N., Yen Nquyen Dong. *Quadratic Programming and Affine Variational Inequalities: a Qualitative Study.* Springer US, 2005. ISBN 978-0-387-24278-1.

[15] Dostál Zdeněk. *Optimal Quadratic Programming Algorithms: with Applications to Variational Inequalities.* Springer US, 2009. 284 s. ISBN 978-0-387-84806-8.

[16] Luenberger David G., Ye Yinyu. *Linear and Nonlinear Programming.* Springer US, 2008. 546 s.

[17] PLOSKAS, Nikolaos a Nikolaos SAMARAS. *Linear programming using MATLAB®.* Cham: Springer, 2017. ISBN 9783319659190.

[18] HU, T. C. a Andrew B. KAHNG. *Linear and integer programming made easy.* Switzerland: Springer, 2016. ISBN 9783319240015.

[19] Back Thomas, B. FOGEL DAVID a Michalewicz Zbigniew. *Handbook of Evolutionary Computation.* Oxford University Press, 1997. 988 s. ISBN 0750303927.

[20] MIRJALILI, Seyedali. *Evolutionary algorithms and neural networks : theory and applications.* . ISBN 9783319930244; 1860-949X.

[21] SIMON, Dan. *Evolutionary optimization algorithms: biologically-Inspired and population-based approaches to computer intelligence.* Hoboken, New Jersey: John Wiley & Sons Inc., 2013. ISBN 9780470937419.

[22] KOCHENDERFER, Mykel J. *Algorithms for optimization.* . Tim A. WHEELER. . ISBN 9780262039420.

[23] RAU, M. a D. SCHRODER. *Model predictive control with nonlinear state space models.* 2002. 136-141 s. ISBN 0-7803-7479-7.

[24] STREJC, Vladimír. *Stavová teorie lineárního diskrétního řízení.* 1 vyd. Praha: Academia, 1978. 374 s.

[25] CAMACHO, Eduardo a Carlos BORDONS. *Model Predictive Control.* Springer-Verlag, London, 2004. ISBN 978-0-85729-398-5.

[26] TANGIRALA, Arun K. *Principles of system identification : theory and practice.* 1st. . ISBN 9781439896020.

[27] LJUNG, Lennart. *System identification: theory for the user.* Upper Saddle River, N.J.: Prentice Hall, 1999. 609 s. ISBN 0136566952.

[28] Anonymous. *Selecting a Model Structure in the System Identification Process.* 2018 Dostupné z: http://www.ni.com/product-documentation/4028/en/.

[29] ZHENG, A. Reducing on-line computational demands in model predictive control by approximating QP constraints. *Journal of Process Control*. 1999, vol. 9, no. 4, s. 279-290. ISSN 0959-1524.

[30] WAN, Z. Y. a M. V. KOTHARE. An efficient off-line formulation of robust model predictive control using linear matrix inequalities. *Automatica*. 2003, vol. 39, no. 5, s. 837-846. ISSN 0005-1098.

[31] SALTIK, M. Bahadir et al. An outlook on robust model predictive control algorithms: Reflections on performance and computational aspects. *Journal of Process Control*. 2018, vol. 61, s. 77-102. ISSN 0959-1524.

[32] RAMIREZ, D. R., T. ALAMO a E. F. CAMACHO. Computational burden reduction in min-max MPC. *Journal of the Franklin Institute-Engineering and Applied Mathematics*. 2011, vol. 348, no. 9, s. 2430-2447. ISSN 0016-0032.

[33] ORAVEC, J. a M. BAKOSOVA. Robust MPC Based on Nominal System Optimization and Weighted Control Input Saturation. *2015 54th Ieee Conference on Decision and Control (Cdc)*. 2015, s. 6239-6244. ISSN 0743-1546.

[34] MUEHLENBEIN, Heinz. Evolutionary Computation: Centralized, Parallel or Collaborative. *Computational Intelligence: Collaboration, Fusion and Emergence*. 2009, vol. 1, s. 561-595. ISSN 1868-4394.

[35] PARRILLA, M., J. ARANDA a S. DORMIDO-CANTO. Parallel evolutionary computation: Application of an EA to controller design. *Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach, Pt 2, Proceedings*. 2005, vol. 3562, s. 153-162. ISSN 0302-9743.

[36] MURTHY, V. K. a E. V. KRISHNAMURTHY. *Parallel programming paradigm: Application to evolutionary computations.* 1995. 299 s. ISBN 0780327594.

[37] TSUTSUI, Shigeyoshi a Noriyuki FUJIMOTO. *An Analytical Study of GPU Computation for Solving QAPs by Parallel Evolutionary Computation with Independent Run.* 2010. ISBN 978-1-4244-8126-2.

[38] SADRIEH, Arash a Parisa A. BAHRI. Application of Graphic Processing Unit in Model Predictive Control. *21st European Symposium on Computer Aided Process Engineering*. 2011, vol. 29, s. 492-496. ISSN 1570-7946.

[39] PHUNG, Duc-Kien et al. Model Predictive Control for Autonomous Navigation Using Embedded Graphics Processing Unit. *Ifac Papersonline*. 2017, vol. 50, no. 1, s. 11883-11888. ISSN 2405-8963.

[40] HYATT, Phillip a Marc D. KILLPACK. *Real-Time Evolutionary Model Predictive Control Using a Graphics Processing Unit.* , 2017. 576 s. ISBN 978-1-5386-4678-6.

[41] MAZINAN, A. H. a A. R. KHALAJI. A comparative study on applications of artificial intelligence-based multiple models predictive control schemes to a class of industrial complicated systems. *Energy Systems-Optimization Modeling Simulation and Economic Aspects*. 2016, vol. 7, no. 2, s. 237-269. ISSN 1868-3967.

[42] RUSSELL, Stuart J. a Peter NORVIG. *Artificial intelligence: a modern approach.* Upper Saddle River, N.J.: Prentice Hall, 2010. 1132 s. ISBN 0132071487.

[43] GRÜNE, Lars a Jürgen PANNEK. *Nonlinear model predictive control: theory and algorithms.* Second. Cham: Springer, 2016. ISBN 9783319460246.

[44] LEE, Jay H. Model Predictive Control: Review of the Three Decades of Development. *International Journal of Control Automation and Systems*. 2011, vol. 9, no. 3, s. 415-424. ISSN 1598-6446.

[45] ZIMMER, Andrea et al. Evolutionary algorithm enhancement for model predictive control and real-time decision support. *Environmental Modelling & Software*. 2015, vol. 69, s. 330-341. ISSN 1364-8152.

[46] YAN, Jingyu, Qing LING a Wei CHEN. *Nonlinear Model Predictive Control Based on Evolutionary Algorithms: Framework, Theory, and Application.* . AJ Costa GIORDANO GE. , 2009. 205 s. ISBN 978-1-60456-883-7.

[47] CANNON, M. Efficient nonlinear model predictive control algorithms. *Annual Reviews in Control*. 2004, vol. 28, no. 2, s. 229-237. ISSN 1367-5788.

[48] NANDOLA, Naresh N. a Sharad BHARTIYA. A multiple model approach for predictive control of nonlinear hybrid systems. *Journal of Process Control*. 2008, vol. 18, no. 2, s. 131-148. ISSN 0959-1524.

[49] BINDLISH, R. a J. B. RAWLINGS. Target linearization and model predictive control of polymerization processes. *AIChE Journal*. 2003, vol. 49, no. 11, s. 2885-2899. ISSN 0001-1541.

[50] LEE, J. H. a N. L. RICKER. Extended Kalman Filter Based Nonlinear Model-Predictive Control. *Industrial & Engineering Chemistry Research*. 1994, vol. 33, no. 6, s. 1530-1541. ISSN 0888-5885.

[51] WANG, Xin, Longxiang GUO a Yunyi JIA. Road Condition Based Adaptive Model Predictive Control for Autonomous Vehicles. *Proceedings of the Asme 11th Annual Dynamic Systems and Control Conference, 2018, Vol 3*. 2018, s. V003T37A005. ISSN 2151-1853.

[52] TANASKOVIC, Marko, Lorenzo FAGIANO a Vojislav GLIGOROVSKI. *Adaptive Model Predictive Control for Constrained Time Varying Systems.* , 2018. 1703 s. ISBN 978-3-9524-2698-2.

[53] TESFAY, Mehari et al. Adaptive-model predictive control of electronic expansion valves with adjustable setpoint for evaporator superheat minimization. *Building and Environment*. 2018, vol. 133, s. 151-160. ISSN 0360-1323.

[54] GRIFFITH, Devin W., Lorenz T. BIEGLER a Sachin C. PATWARDHAN. Robustly stable adaptive horizon nonlinear model predictive control. *Journal of Process Control*. 2018, vol. 70, s. 109-122. ISSN 0959-1524.

[55] SUÁREZ, G. I. et al. *Adaptive neural model predictive control for the grape juice concentration process.* , 2010 [cit. 12 September 2018]. 57-62 s. Cited By :5.

[56] KUMAR, Steven Spielberg Pon et al. A Deep Learning Architecture for Predictive Control. *Ifac Papersonline*. 2018, vol. 51, no. 18, s. 512-517. ISSN 2405-8963.

[57] PICHE, S. et al. Neural network based Model Predictive Control. *Advances in Neural Information Processing Systems 12*. 2000, vol. 12, s. 1029-1035. ISSN 1049-5258.

[58] WONG, Wee Chin et al. Recurrent Neural Network-Based Model Predictive Control for Continuous Pharmaceutical Manufacturing. *Mathematics*. 2018, vol. 6, no. 11, s. 242. ISSN 2227-7390.

[59] DEEPA, S. N. a I. BARANILINGESAN. Optimized deep learning neural network predictive controller for continuous stirred tank reactor. *Computers & Electrical Engineering*. 2018, vol. 71, s. 782-797. ISSN 0045-7906.

[60] HOU, Guolian et al. *Multi-model Predictive Control based on Neural Network and Its Application in Power Plant.* NEW YORK; 345 E 47TH ST, NEW YORK, NY 10017 USA: IEEE, 2014. 4383 s. ISBN 978-1-4799-5825-2.

[61] HAN, H. -G, X. -L WU a J. -F QIAO. Real-time model predictive control using a self-organizing neural network. *IEEE Transactions on Neural Networks and Learning Systems*. 2013, vol. 24, no. 9, s. 1425-1436 [cit. 12 September 2018].

[62] KUBALCÍK, Marek a Vladimír BOBÁL. *Predictive control of three-tank-system utilizing both state-space and input-output models.* Regensburg, Germany: European Council for Modelling and Simulation (ECMS), 2016. 348-353 s. ISBN 9780-993244025.

[63] SATHISHKUMAR, K., V. KIRUBAKARAN a T. K. RADHAKRISHNAN. Real Time Modeling and Control of Three Tank Hybrid System. *Chemical Product and Process Modeling*. 2018, vol. 13, no. 1, s. 20170016. ISSN 1934-2659.

[64] MAXIM, Anca, Clara IONESCU a Robin DE KEYSER. Modelling and identification of a coupled sextuple water tank system. *Proceeding of 2016 Ieee International Conference on Automation, Quality and Testing, Robotics (Aqtr)*. 2016, s. 393-398. ISSN 1844-7872.

[65] COPOT, Dana et al. Multivariable control of sextuple tank system with non-minimum phase dynamics. *Proceeding of 2016 Ieee International Conference on Automation, Quality and Testing, Robotics (Aqtr)*. 2016, s. 399-404. ISSN 1844-7872.

[66] ZHOU, Feng et al. RBF-ARX model-based MPC strategies with application to a water tank system. *Journal of Process Control*. 2015, vol. 34, s. 97-116. ISSN 0959-1524.

[67] XU, Jing et al. Recurrent neural network for solving model predictive control problem in application of four-tank benchmark. *Neurocomputing*. 2016, vol. 190, s. 172-178. ISSN 0925-2312.

[68] KLAUCO, Martin, L'ubos CIRKA a Juraj KUKLA. Non-linear model predictive control of conically shaped liquid storage tanks. *Acta Chimica Slovaca*. 2018, vol. 11, no. 2, s. 141-146. ISSN 1337-978X.

[69] KALA, H., P. ARAVIND a M. VALLUVAN. *Comparative Analysis of Different Controller for a Nonlinear Level Control Process.* , 2013. 729 s. ISBN 978-1-4673-5758-6.

[70] VENKATESAN, N. a N. ANANTHARAMAN. *Controller design based on model predictive control for a nonlinear process.* , 2012 [cit. 6 June 2019]. Cited By :2.

[71] NITHYA, S. et al. Model based controller desigh for a sphrical tank process in real time. *International Journal of Simulation: Systems, Science and Technology*. 2008, vol. 9, no. 4, s. 25-31 [cit. 6 June 2019].

[72] MERCY, D. a S. M. GIRIRAJKUMAR. Modeling and analysis of a real time spherical tank process for sewage treatment plant. *Applied Mathematics and Information Sciences*. 2017, vol. 11, no. 5, s. 1491-1498 [cit. 10 June 2019].

[73] SHRIDHAR, R. a D. J. COOPER. A tuning strategy for unconstrained multivariable model predictive control. *Industrial and Engineering Chemistry Research*. 1998, vol. 37, no. 10, s. 4003-4016 [cit. 10 June 2019].

[74] MANOHAR, Gunaselvi et al. *Neural Network Based Level Control in Two Tank Conical Interacting System.* . Manoharan, S Arunkumar, E Ananthi, K Ganesh,RM. , 2013. 196 s. ISBN 978-1-4673-4601-6.

[75] MÍKA, Vladimír. *Základy chemického inženýrství.* 2 nezm vyd. Praha: Státní nakladatelství technické literatury, 1981. 870 s.

[76] KUNEŠ, Josef a Josef KUNEŠ. *Dimensionless physical quantities in science and engineering*. London ; Waltham, MA: Elsevier, 2012. 441 s. ISBN 9780124160132.

[77] JERRARD, H. G. a D. B. MCNEILL. *A dictionary of scientific units, including dimensionless numbers and scales.* 2d. London: Chapman & Hall, 1964. 204 s.

[78] HAHN, Brian D. a Daniel T. VALENTINE. *Essential MATLAB for engineers and scientists.* 5th. Waltham, Mass. ; Oxford: Acad. Press, 2013. 408 s. ISBN 0123943981.

[79] LINDFIELD, G. R. a J. E. T. PENNY. *Numerical methods.* Academic Press, 2012. [electronic resource]: using MATLAB / George Lindfield, John Penny.; 1 online resource; MATLAB. ISBN 9780-123869425.

[80] DUUN-HENRIKSEN, A. K. et al. Model identification using stochastic differential equation grey-box models in diabetes. *Journal of Diabetes Science and Technology*. 2013, vol. 7, no. 2, s. 431-440 [cit. 6 June 2019].

[81] CORRIOU, Jean-Pierre. *Process Control: Theory and Applications.* 2018. Cham: Springer International Publishing : Imprint : Springer, 2018. ISBN 9783319611433.

[82] ŠULC BOHUMIL, Vítečková M. *Teorie a praxe návrhu regulačních obvodů.* Praha: Vydavatelství ČVUT, 2004. ISBN 80-01-03007-5.

[83] Balátě Jaroslav. *Automatické řízení.* Praha: BEN, 2003. ISBN 80-7300-020-2.

[84] GENG, G. a G. M. GEARY. Experimental comparisons between generalised predictive control algorithms using CARMA and CARIMA models. *Proceedings of the 1996 Ieee Iecon - 22nd International Conference on Industrial Electronics, Control, and Instrumentation, Vols 1-3*. 1996, s. 108-113. ISSN 1553-572X.

[85] HUANG, Sunan, Kok K. TAN a Tong H. LEE. *Applied predictive control.* London ; New York: Springer, 2002. 264 s. ISBN 1852333383.

[86] CLARKE, D. W., C. MOHTADI a P. S. TUFFS. Generalized predictive control-Part I. The basic algorithm. *Automatica*. 1987, vol. 23, no. 2, s. 137-148 [cit. 11 September 2018].

[87] CLARKE, D. W., C. MOHTADI a P. S. TUFFS. Generalized Predictive Control-Part II Extensions and interpretations. *Automatica*. 1987, vol. 23, no. 2, s. 149-160 [cit. 11 September 2018].

[88] BERTSEKAS, Dimitri P. *Nonlinear programming.* Third. Belmont, Mass.: Athena Scientific, 2016. 861 s. ISBN 9781886529052.

[89] Nelles Oliver. *Nonlinear System Identification: from Classical Approaches to Neural Networks and Fuzzy Models.* Springer-Verla Berlin Heidelberg, 2001. 786 s.

[90] Hudson Beale, MarkHagan, Martin. T.Demuth, Howard B. *Neural network toolbox™ getting started guide.* MathWorks, 2016.

[91] *Neural networks for modelling and control of dynamic systems : a practitioner's handbook.* . Magnus NØRGAARD. London: Springer, 2000. ISBN 1852332271.

[92] ZELINKA, Ivan. *Evolutionary algorithms and chaotic systems.* Berlin ; Heidelberg: Springer-Verlag, 2010. 521 s. [electronic resource] / Ivan Zelinka ... [et al.] (Eds.). New York : Springer, 2010. v. 267. ISBN 9783-642107078.

[93] HOLLAND, John H. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence.* Ann Arbor: University of Michigan Press, 1975. 183 s. ISBN 0472084607.

[94] RIBEIRO, J. L., P. C. TRELEAVEN a C. ALIPPI. Genetic-Algorithm Programming Environments. *Computer*. 1994, vol. 27, no. 6, s. 28-29. ISSN 0018-9162.

[95] ANTOŠ JAN, Kubalčík Marek. Analysis of some aspects of optimization problem in predictive control. *International Journal of Mathematical Models and Methods in Applied Sciences*. 2014, vol. 8, s. 8. ISSN 1998-0140.

[96] ANTOS, J. a M. KUBALCIK. Combination of evolutionary and gradient optimization techniques in model predictive control. *International Journal of Mathematical Models and Methods in Applied Sciences*. 2016, vol. 10, s. 34 [cit. 25 September 2018].

[97] RICKER, N. L. *Adaptive optimal control: The thinking man's GPC: R. R. bitmead, M. gevers and V. wertz. ,* 1993. 798-800 s. ID: 271426. ISBN 0005-1098.

[98] TAN, G., H. HAO a Y. WANG. *Real time turning flow estimation based on model predictive control. ,* 2011 [cit. 12 September 2018]. 356-360 s.

SEZNAM OBRÁZKŮ A TABULEK

SEZNAM POUŽITÝCH SYMBOLŮ

$h(k)$	k-tá hodnota impulsní funkce
$g(k)$	k-tá hodnota přechodové funkce
$H(z^{-1})$	impulsní funkce (z-oblast)
$G(z^{-1})$	přenos (z-oblast)
$U(z^{-1}), Y(z^{-1})$	obrazy funkcí vstupu a výstupu
A, B, C, D, F	polynomy lineárních modelů
Δ	přírůstek
n	bílý šum
u	vstup/akční zásah
y	výstup
a_1, a_2, b_1, b_2	koeficienty modelu
$\boldsymbol{\Theta}$	parametry modelu
f	matematická funkce
$\overleftarrow{\boldsymbol{u}}, \overleftarrow{\boldsymbol{y}}$	minulé hodnoty vstupu a výstupu
$\vec{\boldsymbol{u}}, \vec{\boldsymbol{y}}, \vec{\boldsymbol{w}}$	budoucí hodnoty vstupu, výstupu a žádané hodnoty
$\widetilde{\boldsymbol{u}}, \widehat{\boldsymbol{y}}$	vstup a výstup prediktoru
$\langle N_1, N_2 \rangle$	spodní a horní mez výstupního horizontu
N_u	řídicí horizont
J, λ	účelová funkce, váhový parametr
$\boldsymbol{\varepsilon}, \boldsymbol{\nabla}$	velikost kroku, obecný gradient funkce
x, w, a	vstup, váha a aktivační funkce neuronové sítě
Φ, b	přenosová funkce, práh neuronové sítě
x_i, c_i, Σ_i	střední vzdálenost středů od vstupů, střed a norma neuronu
$\boldsymbol{G}, \boldsymbol{X}$	matice prediktoru
$\boldsymbol{y_0}, \boldsymbol{g}, \boldsymbol{H}$	volná odezva, gradient účelové funkce, Hessova matice

$\langle \Delta \boldsymbol{u_{min}}, \Delta \boldsymbol{u_{max}} \rangle$	omezení změny akčního zásahu
$\langle \boldsymbol{u_{min}}, \boldsymbol{u_{max}} \rangle$	omezení akčního zásahu
$\langle \boldsymbol{y_{min}}, \boldsymbol{y_{max}} \rangle$	omezení výstupu
$\boldsymbol{I}, \boldsymbol{T}$	jednotková a dolní trojúhelníková matice
Q, h, S, V_1, V_2	objemový průtok, výška hladiny, průřez nádrže, výstupní ventily
CF	účelová funkce (Cost-Function)
y_m, y_s	výstup modelu, výstup systému
D, R, H, k_D, t_{max}	průměr, poloměr, výška, přepočtový faktor a doba napuštění/vypuštění nádrže (konstanty)
V_h, q_t, h, d, r, k_d	objem, změna objemu, výška nádrže, průměr nádrže, poloměr nádrže a přepočtový faktor (proměnné)
q_{max}, V_{max}, k_q, t	maximální průtok, maximální objem, přepočtový faktor (časový), čas
q_{in}, q_{out}, a	přítok, odtok, akumulace
q_b, q_{of}	funkce dna (bottom), přepadu (overflow)
fun	optimalizovaná funkce nelineární optimalizace

SEZNAM ZKRATEK

ARMAX	AutoRegressive-Moving-Average model with eXogenous input
ARX	AutoRegressive eXogenous model
BFGS	Broyden-Fletcher-Goldfarb-Shanno
CARIMA	Controlled AutoRegressive Integrated Moving Average
CF	Cost Function
DSPI	Direct Synthesis Proportional Integral
EA	Evoluční Algoritmus
EG	Evolučně-Gradientní algoritmus
FIR	Finite Impuls Response
HC	Hill Climbing algoritmus
HW	HardWare
IAE	Integral of Absolute Error
IMC	Internal Model Control regulátor
MLP	Multi-Layer Perceptron
MPC	Model Predictive Control
NN	Neural Network
PID	Proporcionálně-Integračně-Derivační regulátor
PSO	Particle Swarm Optimization
QP	QuadProg
RBF	Radial Basis Function
SOMA	Self-Organizing Migrating Algorithm
SSE	Sum of Squared Errors
SW	SoftWare
UI	Umělá Inteligence

PUBLIKAČNÍ AKTIVITY AUTORA

D_2 – Článek ve sborníku mezinárodní konference neevidované v databázi ISI Proceedings společnosti Thomson Reuters – světový jazyk

ANTOŠ JAN, Kubalčík M. *Optimization in Predictive Control Algorithm. in: Latest Trends in Circuits, Systems, Signal Processing and Automatic Control*. Salerno, Italy: WSEAS Press, 2014, 6 s. ISSN 1790-5117. ISBN 978-960-474-374-2. Dostupné z: http://www.wseas.us/e-library/conferences/2014/Salerno/CISSPA/CISSPA-19.pdf

$D_{nehodn1}$ – Články ve sborníku mezinárodní konference neevidované v databází WoS nebo SCOPUS

ANTOŠ JAN; Kubalčík M. *Optimization in Model Predictive Control Using Evolutionary-Gradient Algorithm. in: The 3rd International Conference on Applied, Numerical and Computational Mathematics*. Sliema, Malta: WSEAS, 2015, 35-41 s. ISSN 2227-4588. ISBN 978-1-61804-328-3.BC - Teorie a systémy řízení

D_{sjr} – Články ve sborníku konference evidované v databázi SCOPUS

ANTOS, J. a M. KUBALCIK. *Alternative approach to optimization in model predictive control using hill climbing algorithm* [online]. , 2015 [cit. 25 September 2018]. 856-864 s.

J_{sc} - Články v recenzovaných časopisech indexovaných v databázi SCOPUS

ANTOŠ JAN, Kubalčík Marek. Analysis of some aspects of optimization problem in predictive control. *International Journal of Mathematical Models and Methods in Applied Sciences* [online]. 2014, vol. 8, s. 8. ISSN 1998-0140. Dostupné z: http://www.naun.org/main/NAUN/ijmmas/2014/a422001-049.pdf.

ANTOS, J. a M. KUBALCIK. Combination of evolutionary and gradient optimization techniques in model predictive control. *International Journal of Mathematical Models and Methods in Applied Sciences* [online]. 2016, vol. 10, s. 34 [cit. 25 September 2018]

Ostatní publikační aktivity

SKODA, David; URBANEK, Pavel; SEVCIK, Jakub; MUNSTER, Lukas; NADAZDY, Vojtech; CULLEN, David A; BAZANT, Pavel; ANTOS, Jan; KURITKA, Ivo. Colloidal Cobalt-doped ZnO Nanoparticles by Microwave-assisted Synthesis and Their Utilization in Thin Composite Layers with MEH-PPV as an Electroluminescent Material for Polymer Light Emitting Diodes. Organic Electronics. 2018, vol. 59, no. C s. 337-348. ISSN:1566-1199.

SKODA, David, Pavel URBANEK, Jakub SEVCIK, et al. Colloidal cobalt-doped ZnO nanoparticles by microwave-assisted synthesis and their utilization in thin composite layers with MEH-PPV as an electroluminescent material for polymer light emitting diodes. Organic Electronics. 2018, 59, 337-348. DOI: 10.1016/j.orgel.2018.05.037. ISSN 15661199.

DATTA, Sanjoy; ANTOS, Jan; STOCEK, Radek. Characterisation of Ground Tyre Rubber by Using Combination of FT-IR Numerical Parameter and DTG Analysis to Determine the Composition of Ternary Rubber Blend. Polymer Testing. 2017, vol. 59 s. 308-315. ISSN:0142-9418.

DATTA, Sanjoy; ANTOŠ, Jan; STOČEK, Radek. Smart Numerical Method for Calculation of Simple General Infrared Parameter Identifying Binary Rubber Blends. Polymer Testing. 2017, vol. 57 s. 192-202. ISSN:0142-9418.

DATTA, Sanjoy, Jan ANTOŠ a Radek STOČEK. A novel algorithm. LION, Alexander a Michael JOHLITZ, ed. Constitutive Models for Rubber X. CRC Press, 2017, 2017-8-15, s. 213-217. DOI: 10.1201/9781315223278-31. ISBN 9781315223278.

MASAŘ, Milan; MACHOVSKÝ, Michal; URBÁNEK, Michal; URBÁNEK, Pavel; ANTOŠ, Jan; KUŘITKA, Ivo. On-line Measurement of Photocatalytic Activity of Powdered Samples. 8th International Conference on Nanomaterials - Research & Application (NANOCON 2016). 2016, 275-279 s.

ŠEVČÍK, Jakub; URBÁNEK, Pavel; ŠULY, Pavol; URBÁNEK, Michal; MAŠLÍK, Jan; ANTOŠ, Jan; KUŘITKA, Ivo. Preparation and Characterization of Nanostructured Thin Films Applicable in Polymer Light Emitting Devices. 8th International Conference on Nanomaterials - Research & Application (NANOCON 2016). 2016, 817-821 s.

ODBORNÝ ŽIVOTOPIS

Osobní informace

Jméno, příjmení	**Jan Antoš**
Adresa	Rymice 112, 769 01 Holešov
Telefon	+420 603 915 687
E-mail	antos@utb.cz
Státní občanství	Česká republika
Datum narození	05. 12. 1988
Pohlaví	Muž

Vzdělání, odborná příprava

Datum	2013 až současnost
Aktuální stav	doktorský student
Organizace	Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky, Nad Stráněmi 4511, 760 05 Zlín

Datum	2011 – 2013
Dosažený stupeň	Inženýr / Ing.
Organizace	Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky, Nad Stráněmi 4511, 760 05 Zlín

Datum	2008 – 2011
Dosažený stupeň	Bakalář / Bc.
Organizace	Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky, Nad Stráněmi 4511, 760 05 Zlín

Datum	2004 – 2008
Organizace	Střední škola informatiky, elektrotechniky a řemesel, Školní 1610, 756 61 Rožnov pod Radhoštěm

Osobní dovednosti

Mateřský jazyk	Čeština
Další jazyky	Angličtina – C1
Počítačové dovednosti	Matlab, Mathematica, Labview, C#, MS Office

Ve Zlíně dne 02. 06. 2019 Ing. Jan Antoš

PŘÍLOHA A – STRUKTURA PŘILOŽENÉHO CD

Disertacni_prace.pdf	Disertační práce
struktura_CD.pdf	Soubor s popisem struktury CD
/zdrojove_soubory	Adresář se zdrojovými kódy a daty