Univerza v Ljubljani

Fakulteta za elektrotehniko

Goran Andonovski

Samorazvijajoči se sistemi pri spremljanju in vodenju procesov (Evolving systems in process monitoring and control)

DOKTORSKA DISERTACIJA

Ljubljana, december 2018

Univerza v Ljubljani

Fakulteta za elektrotehniko

Goran Andonovski

Samorazvijajoči se sistemi pri spremljanju in vodenju procesov (Evolving systems in process monitoring and control)

DOKTORSKA DISERTACIJA

Mentor: prof. dr. Igor Škrjanc

Somentor: izr. prof. dr. Gregor Klančar

Ljubljana, december 2018

Zahvala

Rad bi se zahvalil vsem, ki so na kakršenkoli način pripomogli k nastajanju doktorske disertacije. Seveda je doktorska disertacija moje avtorsko delo, ki pa ne bi nastalo v takšni obliki, kot je, brez sodelovanja in podpore vseh, ki so vsak na svoj način vpleteni v to zgodbo.

Najprej bi se zahvalil mentorju prof. dr. Igorju Škrjancu in somentorju izr. prof. dr. Gregorju Klančarju za vse nasvete, pripombe, komentarje in za usmerjanje v času mojega raziskovalnega dela. Hvala obema tudi za izjemno podporo. Zahvalil bi se tudi celotnima kolektivoma laboratorijev LMSV in LAMS za prijetno vzdušje ter spodbudo in podporo v času nastajanja doktorske disertacije. V času študija sem spoznal in sodeloval tudi z nekaterimi kolegi raziskovalci iz tujine (Brazilije, Španije, Anglije in Čila), ki so mi pomagali najti rešitve za nekatere skupne izzive.

Ključen del sestavljanke predstavlja tudi moja družina. Najprej bi se zahvalil staršema, ki sta mi omogočila celoten študij v Sloveniji in me nenehno podpirala ter spodbujala v času študija. Tudi moj brat Zoran ima posebno mesto v celotni zgodbi, ki mi je vedno stal ob strani. Na koncu pa bi se zahvalil tudi soprogi Maji in sinu Damjanu, ki sta v moje življenje vnesla novo in posebno dimenzijo ter mi tako razširila obzorje razmišljanja.

Povzetek

V pričujoči doktorski disertaciji smo raziskovali različna področja uporabe samorazvijajočih se sistemov, kot so vodenje, identifikacija in spremljanje procesov. Samorazvijajoči se sistemi, ki jih obravnavamo v disertaciji, temeljijo na posplošenem mehkem modelu AnYa (po priimkih avtorjev Angelov in Yager), ki se razlikuje od klasičnih mehkih modelov (Mamdani in Takagi-Sugeno) v načinu definiranja strukture modela. Namesto vnaprej definiranih mehkih pravil (Gaussovo, trikotno, lingvistično itn.), model AnYa na podlagi sprotno sprejetih podatkov tvori mehka pravila v obliki oblakov podatkov (ang. *data clouds*). Z vsakim novim prejetim podatkom se struktura in parametri modela prilagajajo novim spremenjenim stanjem v procesu. To nam omogoča implementacijo različnih algoritmov sprotnega vodenja, identifikacije ali spremljanja dinamičnih procesov. Poleg implementacije teh algoritmov smo v doktorski disertaciji obravnavali tudi mehanizme samorazvijanja modelov za dodajanje novih in odstranjevanje nepomembnih oblakov. Prav tako smo iskali načine, kako preprečiti dodajanje oblakov na osnovi osamelcev.

V disertaciji smo predstavili robusten samorazvijajoči se adaptivni mehki regulator (ang. *robust evolving cloud-based controller*, RECCo). Regulator je sestavljen iz dveh glavnih delov: samorazvijajoča se struktura modela (z mehanizmom za dodajanje novih oblakov na podlagi lokalne gostote podatkov) in sprotna adaptacija parametrov lokalnih regulatorjev (na podlagi gradienta kriterijske funkcije). Mehanizem samorazvijanja modela skrbi za zaznavanje nelinearnih področij v procesu, kar pomeni, da se parametri lokalnih regulatorjev prilagajajo delovni točki procesa. Z normiranjem podatkovnega prostora smo dosegli enostavnejše nastavljanje začetnih parametrov regulatorja. Podali smo tudi smernice, kako nastaviti oziroma izračunati začetne vrednosti parametrov regulatorja. V doktorski disertaciji smo prikazali nekaj primerov uporabe RECCo-regulatorja na simuliranih in realnih napravah. Vodenje na simuliranih procesih smo izvedli na modelu toplotnega izmenjevalnika in na modelu distribuiranega sistema sončnih

kolektorjev. Uporaba vodenja na realnih napravah pa je bila izvedena pri regulaciji temperature na toplotnem izmenjevalniku in regulaciji nivoja na sistemu dveh povezanih tankov.

Mehanizem samorazvijanja na osnovi oblakov podatkov smo vpeljali tudi v mehki prediktivno funkcijski regulator (ang. *fuzzy cloud-based predictive functional controller*, FCPFC). Za delovanje tega regulatorja potrebujemo model procesa, ki ga želimo voditi. S tem namenom smo združili samorazvijajoči se model z rekurzivno metodo najmanjših kvadratov. Na ta način lahko identificiramo dinamičen model procesa, ki je potem del prediktivnega regulatorja. Model uporabimo za predikcijo reguliranega signala na vnaprej določenem horizontu in nato določimo še regulirni signal, ki minimizira razliko med izhodnim in referenčnim/želenim signalom. Takšen pristop je primeren za regulacijo nelinearnih dinamičnih procesov. Delovanje predlaganega prediktivnega regulatorja FCPFC smo preizkusili na modelu reaktorja z neprekinjenim mešanjem (ang. *continuous stirred tank reaktor*, CSTR). Dobljene rezultate smo primerjali z RECCo-regulatorjem.

V nadaljevanju smo predlagali in preizkusili samorazvijajoči se model na osnovi oblakov za identifikacijo dinamičnih sistemov. V tem primeru smo raziskali različne mehanizme dodajanja in odstranjevanja oblakov in njihov vpliv na učinkovitost celotne metode. Predlagano metodo smo preizkusili na dveh različnih primerih. Prvi primer je model kemičnega procesa Tennessee Eastman, ki ima zelo kompleksno strukturo in dinamiko. Iz tega modela smo pridobili simulirane podatke ter poskušali pridobiti modele kazalnikov proizvodnje učinkovitosti. Rezultate smo primerjali z metodo eFuMo ter z nevronskimi mrežami. Drugi primer je bil realen sistem hladilne postaje, ki obratuje v enem od podjetij v Sloveniji. Pridobljene podatke smo prav tako uporabili za identifikacijo dinamičnih modelov nekaj ključnih kazalnikov proizvodnje. Te modele smo naknadno uporabili za nadzorovano in prediktivno preklapljanje hladilnih agregatov, ki so ključni elementi celotnega sistema. Izkazalo se je, da z uporabo modelov lahko preprečimo nepotrebna preklapljanja agregatov in s tem omogočimo boljšo učinkovitost celotnega sistema.

V zadnjem delu smo samorazvijajoči se model uporabili kot orodje za spremljanje procesov. Z uporabo mehanizmov za dodajanje novih oblakov lahko

definiramo področje procesa, ki opisuje normalno stanje delovanja, in področje, ki označuje napako na procesu. Nato lahko z izračunom lokalnih gostot za vsak podatek posebej določimo ali predstavlja napako oziroma normalno delovanje. Predlagali smo tudi izračun delnih lokalnih gostot z upoštevanjem najbolj vplivnih komponent. Delovanje metode smo preizkusili na področju zaznavanja napak na sistemu Tennessee Eastman. Rezultate smo primerjali z nekaj znanimi metodami za zaznavanje napak na procesih, kot so PCA (ang. *principal component analysis*), ICA (ang. *independent component analysis*) in FDA (ang. *fisher discriminate analysis*). Rezultati metode so primerljivi in dosegajo podobno natančnost, kot že uveljavljene metode na tem področju.

Na koncu smo samorazvijajoči se model razdelili na več hierarhičnih nivojev z namenom zaznavanja manevrov pri voznikih osebnih avtomobilov, kot so prehitevanje, zaviranje, ustavljanje in podobno. Metoda uporablja le osnovne senzorje (in ne naprednih, kot so kamere, laserji itd.), ki so del standardne opreme osebnih avtomobilov. Izkazalo se je, da predlagani hierarhični koncept od spodaj navzgor (od manj do bolj kompleksnih akcij) v kombinaciji s samorazvijajočim se modelom uspešno zaznava in ločuje med različnimi manevri pri voznikih.

Abstract

In the doctoral thesis the usage of evolving systems on different fields was investigated, namely the field of control, system identification and process monitoring. The evolving systems in the current thesis are based on the simplified fuzzy model AnYa which differs from the classical ones (such as Mamdani and Takagi-Sugeno) in the way how the model structure is defined. The AnYa model uses data clouds in the antecedent part of the fuzzy system instead of predefined fuzzy rules (Gaussian, triangular, trapezoid etc.) in the classical models. Therefore, the structure of the AnYa could be adapted with each data point received in online manner. This leads to easy implementation of advanced on-line methods for control, identification or monitoring of dynamic processes. Beside this we have investigated different evolving mechanisms for adding new rules/clouds and for removing less important ones.

Firstly we have presented a robust evolving cloud-based controller RECCo for nonlinear processes. The controller consists of two main parts: evolving mechanism (adding new clouds according to the local density of the data) and adaptive law of the local controllers' parameters (based on gradient descent method). The evolving part takes care of the controlled process nonlinearity and the adaptive law adjusts to the current operating point. We proposed a normalization of the data space which simplifies the process of setting the initial parameters. Also some instructions for setting/calculating the initial parameters are given. In the dissertation we show some practical examples of using the controller on simulated and real processes. For the simulated examples we used a model of heat exchanger and a model of distributed solar collector field. Moreover a real plant of heat exchanger and two tank plant were used for temperature and level control, respectively.

Based on the evolving model we also introduce a fuzzy cloud-based predictive functional controller (FCPFC). This controller requires a model of the controlled

process. With this purpose we joined the evolving mechanisms with the recursive weighted least square method. Actually we developed a tool for identification of dynamic process model. The model is further used for the controlled signal prediction in a certain horizon and then the control signal is chosen to minimize the error between the predicted and the desired value of the signal. This approach is suitable for controlling nonlinear dynamic processes. The efficiency of the proposed controller FCPFC was tested on Continuous Stirred Tank Reactor, CSTR. The results were also compared with the RECCo controller.

In the second part of the doctoral thesis we proposed an evolving method for dynamic process identification. Actually the recursive identification method was already tackled in the previous part when FCPFC controller was proposed. In this part we additionally investigated different mechanisms for adding and removing data clouds (fuzzy rules) and their impact on the overall identification method efficiency. The proposed method was tested on two different examples. The first example was Tennessee Eastman process which has really complex structure with dynamic behavior. We acquired the necessary data from the model and then we identified the models of the production Performance Indicators (pPI). The obtained results were compared to the established methods eFuMo and neural networks. The second example was a real process of water chiller plant (WCP). The data were acquired directly from the plant and were used for model identification of the key indicators. The models were further used for process monitoring and predictive operating with the chillers, which are the key elements of the whole system. It has been shown that using the acquired models we can prevent unnecessary switching of the chillers which leads to more efficient operation of the system.

In the last part of the thesis we investigated the usage of the evolving system for process monitoring and fault detection purposes. Using the evolving model we can define the part of data space which describe normal process operation and faults. According to the local density we can easily determine if each data sample is a fault or not. Moreover we proposed a partial data density calculation which takes into account only the most influential components. The efficiency of the method was tested on the Tennessee Eastman process. The results were compared with well established methods on the field such as: PCA (principal

component analysis), ICA (independent component analysis) and FDA (fisher discriminate analysis). With the comparison analysis of the results we show that the proposed evolving method is comparable with the established ones.

At the end we developed a hierarchical evolving model for car-driver behavior detection. The method uses the basic sensors (no cameras, radars etc.) in the car which are part of standard equipment of a modern car. We show that with the proposed hierarchical concept and the evolving model we can efficiently detect different maneuvers (such as overtaking, stopping, breaking, etc.) and can differentiate between them.

Vsebina

Seznam slik

Seznam tabel

1. Uvod

Današnji industrijski procesi so zelo kompleksni sistemi z velikim številom merjenih (in nemerjenih) parametrov, ki so nam na voljo v realnem času. Količina in hitrost obdelave teh signalov s trenutno tehnologijo ne predstavlja večjega problema. S tem se povečujejo zahteve po hitrih in sprotnih (ang. *on-line*) algoritmov modeliranja, spremljanja in vodenja takih procesov na podlagi merjenih podatkov. Algoritmi, ki nenehno spremljajo proces in se mu prilagajajo, lahko zagotovijo, da so nastavljeni na trenutno, dejansko stanje procesa. Spremenljivi pogoji, neznani zunanji vplivi in nova (neodkrita) stanja na sistemu lahko sprožijo novo dinamično obnašanje sistema, kar nam prej ni bilo znano. V takih primerih bi statični konvencionalni sistemi odpovedali. S tem namenom se je pred petnajstimi leti začelo razvijati področje samorazvijajočih se (mehkih) inteligentnih sistemov [13], ki se prilagajajo trenutnim razmeram na procesu.

Samorazvijajoči se mehki sistemi (ang. *evolving fuzzy systems*, EFS) posodabljajo svojo strukturo in parametre ter na ta način vključujejo novo znanje na podlagi sprotno sprejetih podatkov. V tem smislu lahko sisteme EFS štejemo kot pomemben korak k računalniški inteligenci, saj se modeli nenehno in avtomatično učijo in prilagajajo spremembam stanj, zunanjih pogojev in odkritjem novih nelinearnih področij procesa.

Vsi zgoraj našteti izzivi se lahko preslikajo tudi na področje vodenja v primeru, ko želimo voditi nelinearen in časovno spremenljiv proces. Vodenje takega procesa s standardnim (statičnim) pristopom ponuja zadovoljive rezultate le v okolici lokalnega modela in ni primeren za celotno operacijsko območje procesa. Kombinacija samorazvijajoče se strukture mehkega modela s sprotno adaptacijo parametrov lokalnih regulatorjev ponuja celovito rešitev za tovrstne težave.

Prav tako lahko na področju identifikacije sistemov uporabimo samorazvijajoči se model za detekcijo nelinearnih področij sistema. Načela samorazvijanja v kombinaciji z modelom NARX (ang. *nonlinear autoregressive exogenous model*) in z rekurzivno metodo uteženih najmanjših kvadratov za oceno parametrov tvori orodje za identifikacijo nelinearnih, dinamičnih in časovno spremenljivih

procesov. V disertaciji je poudarek na različnih mehanizmih samorazvijanja in kombinacijah med njimi.

Inteligentne (napredne) metode spremljanja procesov igrajo pomembno vlogo tudi pri zaznavanju in diagnosticiranju napak (ang. *fault detection and diagnostic*). Klasične metode spremljanja procesov namreč temeljijo le na spremljanju posameznih (merljivih) spremenljivk, in sicer ali so znotraj predhodno opredeljenih meja. Spremljanje procesov lahko znatno izboljšamo, če upoštevamo več merjenih spremenljivk, ki so nam na voljo v kombinaciji z naprednimi, inteligentnimi metodami. Tudi na tem področju imajo samorazvijajoče se sisteme velik potencial. Z njimi lahko zaznavamo napake na procesu ter lezenje parametrov procesa [166].

Doktorsko disertacijo lahko razdelimo v štiri glavne sklope. V prvem sklopu so podane osnove samorazvijajočega se sistema (poglavje 2), predstavljen pa je tudi robustni samorazvijajoči se adaptivni mehki regulator za nelinearne sisteme (poglavje 3). Poleg tega je v prvem sklopu podan tudi prediktivni funkcijski regulator, katerega identifikacijski del se prekriva z drugim sklopom disertacije. Drugi sklop obravnava identifikacijo dinamičnih sistemov, prav tako z uporabo načela in mehanizme samorazvijanja (poglavje 4). Tretji in četrti sklop sta dokaj sorodna in obravnavata spremljanje procesov. V tretjem sklopu je podan sistem za zaznavanje napak na industrijskih procesih (poglavje 5). Četrti sklop pa obravnava področje spremljanja manevrov pri voznikih na podlagi hierarhične strukture v kombinaciji s samorazvijajočim se modelom (poglavje 6).

2. Samorazvijajoči se sistemi

Z razvojem informacijskih tehnologij (t.i. informacijska revolucija) so se izrazito povečali tok in hitrost prenašanja podatkov kot tudi količina shranjenih podatkov. Velika industrijska podjetja v Evropi že razvijajo in ustvarjajo nov standard, ki so ga poimenovali Industrija 4.0. Spremembe, ki jih prinaša novi standard, so na področju organizacije proizvodnih procesov. Ideja je, da se poslovni in procesni ukrepi določajo na podlagi podatkov. Pravzaprav ideja o obdelavi in analizi shranjenih in sprotno zajetih podatkov ni nova, vendar se sedaj intenzivneje razvijajo celovite rešitve. Z analizo podatkov želimo pridobiti nove koristne informacije o sistemu oziroma procesu, ki ga upravljamo. Pogosto se zgodi, da s podatkovno analizo izluščimo informacije, o katerih prej nismo niti razmišljali in jih tudi nismo pričakovali.

Pri obdelavi podatkov se srečujemo ne le z veliko količino podatkov, temveč tudi z dinamiko podatkovnih struktur, ki se lahko interpretira kot tok podatkov (ang. *data streams*). S tem namenom so bile v preteklosti razvite metode, ki omogočajo sprotno (ang. *on-line*) kot tudi nesprotno (ang. *off-line*) obdelavo ter analizo podatkov. Obstaja več različnih tovrstnih metod, kot so adaptivne metode na področju vodenja (ang. *adaptive systems*) [29], metode strojnega učenja (ang. *machine learning*) in statističnega učenja (ang. *statistical learning*) [108] na področju modeliranja procesov ter klasične identifikacijske metode [165]. Področje, ki bo obravnavano v tem poglavju in na splošno v doktorski disertaciji, obravnava samorazvijajoče se inteligentne sisteme (ang. *evolving intelligent systems*) [20]. V literaturi se srečujemo tudi samo z izrazom samorazvijajoči se sistemi (ang. *evolving systems*), kar bo uporabljeno tudi v pričujoči doktorski disertaciji. Treba je poudariti, da v strokovni angleški literaturi pride pogosto do medsebojne zamenjave pomenov *evolving systems* in *evolutionary systems*. Pravzaprav gre za dva konceptualno različna pristopa (podpoglavje 1.6 v [17]).

Glede na hitro povečevanje količine in kompleksnosti shranjenih podatkovnih struktur se tudi v industrijskih okoljih (področje vodenja, modeliranja itd.) povečujejo zahteve po sistemih, ki svoje delovanje (inteligentno) samodejno pri-

lagajajo pridobljenim informacijam o procesu [96]. S takšnim pristopom lahko zagotovimo, da se sistemi lahko hitro prilagodijo trenutnim spremembam v delovanju procesa (npr. sprememba delovnih pogojev, zunanjih vplivov itn.) [220]. To je glavna prednost pred metodami, ki temeljijo na konvencionalnih modelih, ki so nastavljeni oziroma naučeni predhodno (*off-line* način) in se med uporabo ne spreminjajo.

Področje samorazvijajočih se sistemov ponuja rešitve in pristope za izvedbo novih ali za nadgradnjo obstoječih metod pri reševanju dinamičnih problemov, ki se spreminjajo s časom. V literaturi lahko najdemo veliko primerov uporabe teh sistemov, kot na primer na področju aktivnega učenja [168], adaptivnega vodenja [55], bioinformatike [137], pametnih senzorjev [197], sprotnega nadzora [174] itd. Področje samorazvijajočih se sistemov se aktivno razvija v zadnjih dveh desetletjih, kar je rezultiralo v obsežno literaturo [13, 17, 20, 21, 166].

Poleg možnosti nadgradnje obstoječih metod samorazvijajoči se sistemi ponujajo tudi možnost razvoja popolnoma novih algoritmov z visoko stopnjo samostojnega (strojnega) učenja (ang. *autonomous machine learning*). V glavnem se področje samorazvijajočih se sistemov razvija hitro in tako povečuje svojo bazo znanja in razumevanja kompleksnih povezav in odvisnosti v realnih aplikacijah. Iščejo se novi mehanizmi, kako iz podatkov samodejno izluščiti nove kompleksne povezave, in sicer v čim krajšem času in na čimbolj optimalen način. Tovrstno učenje se nenehno izvaja, vse dokler prihajajo novi podatki, in tako odkrivamo nove zakonitosti znotraj obravnavanih sistemov.

Pogosto, vendar ne nujno, uporabljajo samorazvijajoči se sistemi mehke (ang. *fuzzy rule-based*), nevromehke (ang. *neuro-fuzzy*) ali nevronske (ang. *neural-network*) modele v kombinaciji s strojnim učenjem. Vsekakor je koncept, ki ga ponujajo samorazvijajoči se sistemi, uporaben tudi v povezavi z drugimi sistemi, kot so odločitvena drevesa (ang. *decision trees*), mešani Gaussovi modeli (ang. *Gaussian mixture models*) [169], modeli na osnovi Gaussovih procesov [147] itn. V nadaljevanju bomo opisali osnovne mehke modele in njihovo uporabo s perspektive samorazvijajočih se sistemov.

2.1. Mehki sistemi

Mehki modeli so dobro znani kot orodje za univerzalno aproksimacijo procesov [148], [253]. Avtor mehkih sistemov je prof. Lotfi A. Zadeh, ki je prvi predstavil mehko logiko [265, 266], in sicer kot razširitev klasične (Boolove) logike. Klasična logika logični spremenljivki pripisuje le dve vrednosti (1 za *pravilno* in 0 za *napačno*), mehka logika pa dovoljuje katerokoli vrednost na intervalu [0, 1]. Mehki sistemi so predstavljeni v obliki pravil **ČE-POTEM** (ang. **IF-THEN**), kjer je vsako pravilo sestavljeno iz dveh delov, iz pogojnega (ang. *antecedent*) in posledičnega (ang. *consequent*) dela.

2.1.1 Model Mamdani

Na osnovi teorije, ki jo je podal prof. Zadeh, se je razvilo nekaj različnih tipov mehkih modelov. Nov mehki model je prvi predstavil Mamdani (model Mamdani) [177]. V svojem delu je pravzaprav predstavil mehki regulator na industrijskem procesu parnega motorja. Mehka logika je bila uporabljena za pretvorbo lingvističnih spremenljivk (pravila določena s strani operaterja) v strategijo avtomatskega vodenja. Oblika modela Mamdani je predstavljena z naslednjo enačbo:

$$\mathcal{R}^i: \quad \textbf{IF } \big((x_1 \text{ is } \mu_1^i) \textbf{ AND } \ldots \textbf{ AND } (x_p \text{ is } \mu_p^i)\big) \textbf{ THEN } (y^i(\mathbf{x}) \text{ is } \Phi^i) \quad (2.1)$$

kjer gre indeks $i = 1, \ldots, M$ preko vseh pravil mehkega sistema (M je število vseh pravil). Pogojni del je sestavljen iz p vhodnih x_j ter p lingvističnih μ_j^i spremenljivk ($j = 1, \ldots, p$). Posledični del je sestavljen iz p izhodnih spremenljivk y^i in mehke množice Φ^i. Izhod sistema se ponavadi izračuna z metodo gravitacijskega središča (ang. *center of gravity, COG*) ali s povprečjem maksimumov (ang. *mean of maximum, MOM*).

Eden od samorazvijajočih se modelov, ki uporablja modela Mamdani za identifikacijo nelinearnih sistemov, je SOFMLS [217]. V SOFMLS-u je dodajanje novega mehkega pravila (roja) odvisno od razdalje med trenutnim vzorcem in najbližjem rojem. Če je ta razdalja manjša od vnaprej določenega radija r, je trenutni podatek dodan k najbližjemu roju; v nasprotnem definiramo novo pravilo (roj). Metoda SOFMLS vsebuje tudi pogoje za brisanje najmanj aktivnih rojev (temelji na podlagi števila podatkov, ki pripadajo določenemu roju). Seveda je dodajanje in brisanje rojev strukturna sprememba modela, ki spada pod

pogojni (**IF**) del modela. Parametri posledičnega dela v primeru SOFMLS se posodabljajo na osnovi modificirane metode najmanjših kvadratov.

Na podlagi prednosti dveh predhodnih metod SELM [250] in ECSFS [251] so avtorji v [252] ustvarili novo metodo SSEM (ang. *simplified structure evolving method*). Metoda SSEM razdeli prostor problema na več področij in na podlagi največjega lokalnega pogreška razdeli določeno področje na dve podpodročij (dodajanje novih pravil). Parametri posledičnega dela se posodabljajo na osnovi rekurzivne metode najmanjših kvadratov.

Metoda predstavljena v [254] razdeli vhodni in izhodni prostor na ločene roje. Dodajanje novega roja temelji na minimalni razdalji med trenutnim podatkom in roji, ki so del vhodnega ter izhodnega prostora. Če je podatek precej oddaljen od najbližjega roja v vhodnem in v izhodnem prostoru, potem je dodan nov roj. Parametri posledičnega dela se posodabljajo z uporabo rekurzivne metode Levenberg–Marquardt [81].

Samorazvijajoči se sistem ET2FIS, predstavljen v [244], je pravzaprav nevromehki model, ki uporablja načela modela Mamdani. Dodajanje novih pravil se izvede v primeru, če je najmanjša stopnja pripadnosti podatka obstoječim pravilom manjša od prednastavljenega praga. Model ET2FIS vsebuje tudi mehanizme za združevanje prekrivajočih se kot tudi za brisanje neaktivnih pravil.

2.1.2 Model Takagi-Sugeno

Drugi tip mehkega sistema je model Takagi-Sugeno (T-S), ki je bil predstavljen v [238]. Avtorji so na osnovi mehke logike, ki jo je postavil prof. Zadeh, predstavili matematično orodje za gradnjo mehkih modelov. Za razliko od modela Mamdani je model T-S uporabljen nekoliko širše na področju samorazvijajočih se sistemov [166]. Seveda obstaja več razlogov za to. Najprej so bili zaradi svoje uporabnosti mehki modeli T-S deležni velike pozornosti na različnih področjih, kot na primer na področju vodenja [32,92,93,133,178,195], identifikacije [2,187], spremljanja procesov oziroma detekcije napak [5, 7, 223], obdelave slik [125, 191, 208] in drugje. Nadalje so T-S modeli znani kot univerzalni aproksimator z zadostno stopnjo natančnosti. Struktura T-S modela se razlikuje od modela Mamdani v posledičnem (**THEN**) delu, ki namesto lingvistične mehke množice Φ (2.1)

uporablja poljubno (linearno) funkcijo[1]:

$$\begin{aligned}\mathcal{R}^i : \textbf{IF } (\mathbf{x}(1) \text{ is } \boldsymbol{\mu}^i(1)) \textbf{ AND } \ldots \textbf{ AND } (\mathbf{x}(p) \text{ is } \boldsymbol{\mu}^i(p)) \\ \textbf{THEN } y^i(\mathbf{x}) = f(\mathbf{x}(1), \ldots, \mathbf{x}(p))\end{aligned} \tag{2.2}$$

kjer je vhodni vektor $\mathbf{x} = [\mathbf{x}(1), \ldots, \mathbf{x}(p)]^T$ p-dimenzionalen in je $\boldsymbol{\mu}^i$ mehka (lingvistična) množica. Kot smo že omenili, je posledični del sestavljen iz poljubne funkcije, večinoma pa v obliki linearne kombinacije:

$$y^i(\mathbf{x}) = f(\mathbf{x}(1), \ldots, \mathbf{x}(p)) = \omega_0^i + \omega_1^i \mathbf{x}(1) + \cdots + \omega_p^i \mathbf{x}(p) \tag{2.3}$$

kjer so $\omega_0^i, \ldots \omega_p^i$ uteži funkcije, ki jih moramo določiti. Izhod celotnega sistema y je sestavljen kot linearna kombinacija p lokalnih modelov y^i. Model Takagi-Sugeno lahko torej interpretiramo tudi kot skupek odsekoma lokalno linearnih modelov z glajenjem med njimi.

Obstaja tudi takoimenovani generalizirani model Takagi-Sugeno kot orodje za lažjo implementacijo samorazvijajočih se sistemov. Najprej so ga predstavili avtorji v [154, 156] in, potem je bil tudi nadgrajen v [197, 198]. Osnovni namen generaliziranega modela T-S je predstavitev mehkih pravil (množic) z večdimenzionalno normalno porazdelitvijo. To predvsem olajša način modeliranja lokalnih korelacij med vhodnimi in izhodnimi spremenljivkami.

Eden prvih samorazvijajočih se sistemov na podlagi modela Takagi-Sugeno je eTS (ang. *evolving Takagi-Sugeno*) [26]. Ta metoda vsebuje le mehanizem za dodajanje novih mehkih pravil na podlagi potenciala, ki je definiran kot mera oddaljenosti med trenutnim podatkom in vsemi podatki v prostoru. Če je potencial trenutnega podatka večji, kot ga imajo vsi obstoječi centri, se ta podatek definira kot center novega roja. V [26] so predstavili dva načina posodabljanja parametrov posledičnega dela, in sicer lokalnega (ang. *weighted recursive least-square*, wRLS) in globalnega (ang. *recursive least-square*, RLS). V [19] je bil predstavljen enostavnejši model eTS, ki so ga poimenovali simpl_eTS. V tem modelu so avtorji dodali tudi mehanizem za brisanje nereprezentativnih pravil. V primeru, da neko pravilo vsebuje manj kot 1% podatkov celotne populacije, potem se to pravilo izbriše. Posledični del je ostal isti kot pri eTS [26].

[1]Spremenljivke napisane s krepko pisavo (npr. $\mathbf{x}$) označujejo vektorje/matrike, medtem ko spremenljivke z navadno pisavo označujejo skalarje (npr. ω_0^i). Z oklepaji označujemo posamezne elemente vektorjev oziroma matrik (npr. $\mathbf{x}(1)$ označuje prvi element vektorja $\mathbf{x}$).

V [16] je bil predstavljen mehki model eTS+, ki predstavlja skupek vseh dotedanjih variacij predhodnih modelov [15, 19, 26]. Model eTS+ vsebuje tudi nove parametre za spremljanje kakovosti obstoječih pravil oziroma rojev (starost, podporna množica, uporabnost, področje vpliva in lokalna gostota). Z istimi parametri so definirani tudi mehanizmi za dodajanje in brisanje pravil.

Glavna ideja metode SAFIS [215] je bila ustvariti mehki model z manjšim številom pravil in pri tem obdržati ter izboljšati učinkovitost metode v primerjavi z obstoječimi metodami. Metoda SAFIS vsebuje dva mehanizma samorazvijanja strukture modela, in sicer za dodajanje in brisanje rojev. Oba mehanizma temeljita na kriteriju vpliva (ang. *influence criterion*) ki podaja doprinos trenutnega podatka k skupnemu modelu. V obeh primerih je potrebno nastaviti zgornjo mejo za dodajanje novih in spodnjo mejo za brisanje obstoječih pravil. Parametri posledičnega dela se posodabljajo le za pravilo, kateremu trenutni podatek pripada z največjo pripadnostjo. Posodabljanje se vrši z uporabo razširjenega Kalmanovega filtra (ang. *extended Kalman filter*, EKF).

Model SEIT2FNN [128] predstavlja šestnivojsko nevromehko mrežo, ki uporablja teorijo modela T-S tipa 2. Glede na to, da je v primeru mehkega modela tipa 2 aktivacijska funkcija interval, SEIT2FNN izračuna povprečje tega intervala. Če je maksimalna vrednost teh povprečij (za vsa pravila) manjša od predhodno nastavljenega praga, potem se doda novo pravilo. Posledični del se posodablja z rekurzivno metodo najmanjših kvadratov (RLS).

V [129] je predstavljen model AHLTNM, ki uporablja adaptivno nastavljanje vrednosti praga, ki določa, ali je treba združevati ali deliti obstoječe roje. Mehka pravila v pogojnem delu so predstavljena s hiperkvadrati (center in dolžina stranic). V primeru, da je pogrešek modela večji od adaptivnega praga, potem se pravilo z največjim volumnom razdeli na dve novi pravili; v nasprotnem se pravilo z največjim volumnom združi z najbližjim pravilom. Metoda AHLTNM uporablja rekurzivno metodo najmanjših kvadratov za posodabljanje prametrov linearne funkcije v posledičnem delu mehkega modela.

Avtorji v [163] so predstavili koncept samorazvijajočega se sistema z uporabo participativnega učenja (ang. *evolving participatory learning*, ePL). Metoda ePL spreminja svojo T-S strukturo na podlagi dveh parametrov (enega za dodajanje in drugega za odstranjevanje pravil). Oba parametra imata vsak svoj prag, oba

pa sta predhodno nastavljena. Tako kot pri večini prejšnjih metod, so tudi v primeru ePL parametri posledičnega dela posodobljeni na osnovi rekurzivne metode najmanjših kvadratov.

V [86] so avtorji predstavili samorazvijajoči se sistem eFuMo (ang. *evolving fuzzy model*). Model eFuMo temelji na rekurzivnem mehkem rojenju z uporabo rekurzivnega c-means in algoritma Gustafson-Kessel [89, 90]. Sprotno (rekurzivno) rojenje je nadgrajeno z mehanizmi za dodajanje, odstranjevanje, združevanje in deljenje rojev. Parametri posledičnega dela mehkega sistema se posodabljajo z metodo uteženih najmanjših kvadratov s pozabljanjem.

Model FLEXFIS je bil najprej predstavljen v [171], nato pa je v [167] sledila njegova nadgradnja v FLEXFIS(++). Pravzaprav so avtorji v [167] predstavili celo družino metod (ang. *FLEXFIS family*) za reševanje različnih problemov (rojenje, klasifikacija in regresija). V svoj koncept samorazvijanja so dodali še mehanizme za detekcijo lezenja parametrov in predlagali postopek za odpravljanje tega fenomena. FLEXFIS(++) vsebuje tudi mehanizme za zmanjševanje kompleksnosti modela (združevanje prekrivajočih se pravil).

Na podlagi modela SAFIS [215] so avtorji najprej v [216] in potem še v [214] predstavili razširjeno verzijo modela (ang. *extended SAFIS*, ESAFIS). Metoda SAFIS/ESAFIS ima strukturo nevromehkega modela in vključuje mehanizme za dodajanje in odstranjevanje mehkih pravil.

Do sedaj smo obravnavali samo dva tipa mehkih modelov (model Takagi-Sugeno in model Mamdani). Seveda obstaja tudi veliko različnih izpeljank mehkih modelov, ki pa niso tako zanimive s stališča samorazvijajočih se sistemov. V nadaljevanju bo predstavljen nov neparametričen (vektorski) mehki model, ki je prilagojen in načrtan za samorazvijajoče se sisteme ter je že uveljavljen na tem področju.

2.1.3 Model AnYa

Poleg tradicionalna mehka modela, Mamdani [177] in Takagi-Sugeno [238], so avtorji v [23–25] predstavili nov, neparametričen mehki model, ki so ga poimenovali AnYa (po priimkih avtorjev Angelov in Yager). Razlikuje se v tem, da je pogojni (**IF**) del mehkega modela neposredno sestavljen na podlagi prejetih podatkov. V

Tabela 2.1: Primerjava med različnimi mehkimi modeli [24].

	POGOJNI DEL (**IF**)	POSLEDIČNI DEL (**THEN**)	OSTRENJE
Mamdani [177]	Mehke množice (skalarne, lingvistične, parametrične)	Mehke množice (skalarne, parametrične)	Središče gravitacije
T-S [238]		Funkcija (pogosto linearna)	Utežena vsota
AnYa [24]	Podatkovni oblaki (neparametrični)	Katerokoli od zgoraj navedenih metod	

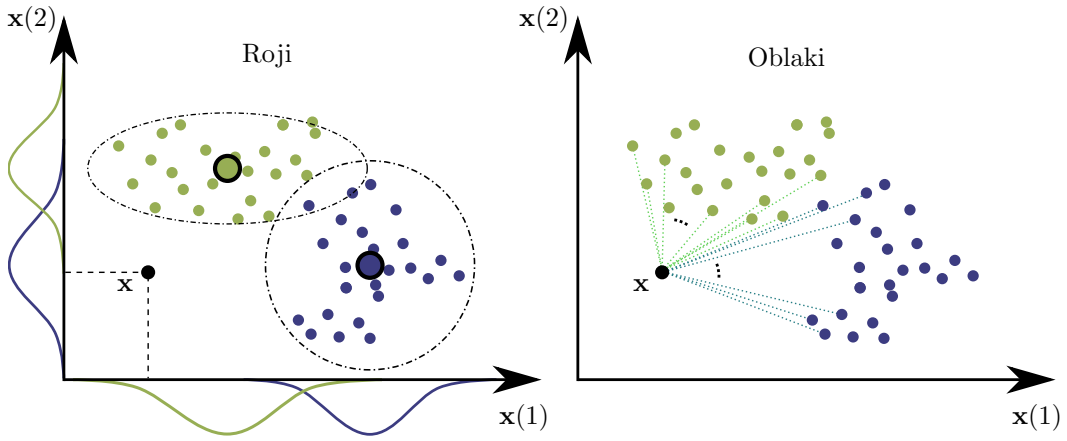
tabeli 2.1 so predstavljene ključne razlike med novim (AnYa) in klasičnimi mehkimi modeli (model Mamdani in model T-S). Pogojni del mehkega modela AnYa igra ključno vlogo pri samorazvijajočih se sistemih. Ta upošteva vse pretekle podatke, tako da nam ni poterbno vnaprej razmišljati o naravi in obliki podatkov, kar je pravilo pri drugih mehkih sistemih. Prav tako ne rabimo določevati tipa pripadnostne funkcije (npr. trapezna, trikotna, Gaussova funkcija ipd.), saj v primeru mehkega modela AnYa pripadnostne funkcije temeljijo na lokalni gostoti podatkov. Podatki, ki ležijo skupaj v podatkovnem prostoru in imajo podobne lastnosti, tvorijo oblake podatkov[2].

Oblaki podatkov so na prvi pogled podobni rojem, a se konceptualno precej razlikujejo od njih. Ena ključnih razlik je ta, da oblaki niso omejeni (nimajo natančno določene meje) in nimajo definirane oblike (glej sliko 2.1). Razlika je tudi v načinu, kako so predstavljeni pretekli podatki. V primeru rojev je vsako pravilo predstavljeno s centrom in področjem vpliva, medtem ko oblaki upoštevajo vse pretekle podatke (podpoglavje 4.3 v [17]).

Struktura mehkega modela AnYa je predstavljena z naslednjo enačbo:

$$\mathcal{R}^i: \quad \textbf{IF} \quad (\mathbf{x} \sim X^i) \quad \textbf{THEN} \quad (y^i = f^i(\mathbf{x})), \qquad i = 1, \ldots, c \tag{2.4}$$

[2]V doktorski disertaciji bomo uporabljali pojem *oblaki podatkov* za označitev mehkih pravil, kar je pomensko različno kot *podatki v oblaku* za hranjenje informacijskih podatkov v računalniški terminologiji.



Slika 2.1: Grafična primerjava med roji (levo) in oblaki podatkov (desno) v 2D prostoru.

kjer je število mehkih pravil $\mathcal{R}^i$ enako številu obstoječih podatkovnih oblakov $i = 1, \ldots, c$. Pogojni (**IF**) del modela je definiran z matematičnim operatorjem $\sim$, ki preberemo kot *je povezan z* (ang. *is associated with*) in označuje podobnost med podatkom $\mathbf{x} = [x(1), x(2), \ldots, x(n)]^T$ in oblakom podatkov X^i. Posledični (**THEN**) del je definiran s poljubno lokalno (npr. linearno) funkcijo $y^i = f^i(\mathbf{x})$. V disertaciji bo sistem AnYa uporabljen na dva načina, in sicer kot mehki regulator in kot mehki model. V prvem primeru bo posledični del definiran z lokalnim regulatorjem u^i, medtem ko bo v drugem primeru definiran z modelom s posplošenim pogreškom ARX (ang. *autoregressive exogenous model*).

Omenili smo že, da pripadnostna funkcija temelji na lokalni gostoti[3] γ_k^i med trenutnim podatkom $\mathbf{x}_k$ in obstoječimi oblaki X^i:

$$\gamma_k^i = \mathcal{K}\left(\sum_{j=1}^{M^i} d_{kj}^i\right), \qquad i = 1, \ldots, c \tag{2.5}$$

kjer je $\mathcal{K}$ poljubno matematično jedro, d_{kj}^i je razdalja med podatkom $\mathbf{x}_k$ in vsemi podatki $\mathbf{x}_{kj}^i$ $(j = 1, \ldots, M^i)$, ki so del enega oblaka (pravila), M^i je število podatkov, ki pripadajo i-temu oblaku, c pa je število vseh oblakov. Če vzamemo Cauchy-jevo jedro, lahko lokalno gostoto zapišemo kot sledi:

$$\gamma_k^i = \frac{1}{1 + \frac{1}{M^i}\sum_{j=1}^{M^i}(d_{kj}^i)^2}, \qquad i = 1, \ldots, c \tag{2.6}$$

[3]V disertaciji bomo uporabljali indeks k za časovni trenutek, medtem ko bomo z i označevali pripadnost k i-temu oblaku (pravilu).

Za izračun lokalne gostote lahko izberemo katerokoli razdaljo, npr. Evklidsko [24,78,229], Mahalanobisovo [41,42], kosinusno [230] itd. V našem primeru bomo uporabili Evklidsko razdaljo $d_{kj}^i = \|\mathbf{x}_k - \mathbf{x}_{kj}^i\|$, ker ni večje razlike v rezultatih v primerjavi z drugimi razdaljami [41].

Evklidska razdalja

Če vzamemo enačbo (2.6) in upoštevamo Evklidsko razdaljo $(d_{kj}^i)^2 = (\mathbf{x}_k - \mathbf{x}_{kj}^i)^T(\mathbf{x}_k - \mathbf{x}_{kj}^i) = \|\mathbf{x}_k - \mathbf{x}_{kj}^i\|^2$, lahko izpeljemo rekurzivno obliko enačbe (2.6) za izračun lokalne gostote (podpoglavje 2.5 v [17]). Postopek izpeljave je prikazan v nadaljevanju. Kot že omenjeno, če vzamemo enačbo (2.6) in upoštevamo $d_{kj}^i = \|\mathbf{x}_k - \mathbf{x}_{kj}^i\|$, dobimo:

$$\gamma_k^i = \frac{1}{1 + \frac{1}{M^i}\sum_{j=1}^{M^i} \|\mathbf{x}_k - \mathbf{x}_{kj}^i\|^2} \tag{2.7}$$

kjer je $\|\cdot\|$ Evklidska norma ($\|\mathbf{h}\| = \sqrt{h_1^2 + h_2^2 + \cdots + h_n^2}$).

V nadaljevanju lahko drugi izraz pod ulomkovo črto v enačbi (2.7) razvijemo v naslednjo obliko (upoštevamo pravilo kvadrata razlike dvočlenika):

$$\gamma_k^i = \frac{1}{1 + \mathbf{x}_k \cdot \mathbf{x}_k - 2\mathbf{x}_k \cdot \left(\frac{1}{M^i}\sum_{j=1}^{M^i} \mathbf{x}_{kj}^i\right) + \frac{1}{M^i}\sum_{j=1}^{M^i} \left(\mathbf{x}_{kj}^i \cdot \mathbf{x}_{kj}^i\right)} \tag{2.8}$$

kjer pika med dvema spremenljivkama označuje skalarni produkt dveh vektorjev. S prištevanjem in odštevanjem lahko izraz pod ulomkovo črto razširimo kot sledi:

$$\gamma_k^i = \frac{1}{1 + \left[\mathbf{x}_k \cdot \mathbf{x}_k - 2\mathbf{x}_k \cdot \left(\frac{1}{M^i}\sum_{j=1}^{M^i} \mathbf{x}_{kj}^i\right) + \left(\frac{1}{M^i}\sum_{j=1}^{M^i} \mathbf{x}_{kj}^i\right)^2\right] - \cdots} \cdots$$
$$\cdots \frac{}{- \left(\frac{1}{M^i}\sum_{j=1}^{M^i} \mathbf{x}_{kj}^i\right)^2 + \frac{1}{M^i}\sum_{j=1}^{M^i} \left(\mathbf{x}_{kj}^i \cdot \mathbf{x}_{kj}^i\right)} \tag{2.9}$$

Izraz v oglatem oklepaju pod ulomkovo črto lahko združimo in skrajšano zapišemo kot kvadrat razlike. Ostale člene obdržimo in dobimo:

$$\gamma_k^i = \frac{1}{1 + \|\mathbf{x}_k - \frac{1}{M^i}\sum_{j=1}^{M^i} \mathbf{x}_{kj}^i\|^2 + \frac{1}{M^i}\sum_{j=1}^{M^i} \left(\mathbf{x}_{kj}^i \cdot \mathbf{x}_{kj}^i\right) - \left(\frac{1}{M^i}\sum_{j=1}^{M^i} \mathbf{x}_{kj}^i\right)^2} \tag{2.10}$$

Na koncu lahko z uporabo enačb (2.12) in (2.13) zapišemo rekurzivno obliko za izračun lokalne gostote med podatkom $\mathbf{x}_k$ in i-tim oblakom:

$$\gamma_k^i = \frac{1}{1 + \|\mathbf{x}_k - \boldsymbol{\mu}_k^i\|^2 + \sigma_k^i - \|\boldsymbol{\mu}_k^i\|^2} \tag{2.11}$$

$$\boldsymbol{\mu}_k^i = \frac{1}{M^i} \sum_{j=1}^{M^i} \mathbf{x}_{kj}^i \tag{2.12}$$

$$\sigma_k^i = \frac{1}{M^i} \sum_{j=1}^{M^i} \left(\mathbf{x}_{kj}^i \cdot \mathbf{x}_{kj}^i\right) \tag{2.13}$$

Enačbi (2.12) in (2.13) lahko zelo preprosto zapišemo tudi v rekurzivni obliki, in sicer:

$$\boldsymbol{\mu}_k^i = \frac{M^i - 1}{M^i} \boldsymbol{\mu}_{k-1}^i + \frac{1}{M^i} \mathbf{x}_k, \qquad \boldsymbol{\mu}_1^i = \mathbf{x}_1 \tag{2.14}$$

$$\sigma_k^i = \frac{M^i - 1}{M^i} \sigma_{k-1}^i + \frac{1}{M^i} \|\mathbf{x}_k\|^2, \qquad \sigma_1^i = \|\mathbf{x}_1\|^2 \tag{2.15}$$

Enačbe (2.11), (2.14) in (2.15) so ključnega pomena za sprotno posodabljanje samorazvijajočega se modela. Te enačbe bodo večkrat omenjene in uporabljene v naslednjih poglavjih doktorske disertacije.

Mahalanobisova razdalja

Avtorji so v [42] prvič uporabili Mahalanobisovo razdaljo za izračun lokalne gostote. Glavna ideja je, da bi upoštevali tudi obliko podatkov/oblakov v samem izračunu gostote. Seveda to vnese dodatno kompleksnost v celoten postopek računanja (izračun kovariančne matrike podatkov, ki tvorijo oblake). V primeru, če vzamemo enačbo (2.6) in upoštevamo Mahalanobisovo razdaljo:

$$d_{kj}^i = (\mathbf{x}_k - \mathbf{x}_{kj}^i)^T (\mathbf{A}^i)^{-1} (\mathbf{x}_k - \mathbf{x}_{kj}^i) \tag{2.16}$$

kjer je $\mathbf{A}^i$ kovariančna matrika i-tega oblaka, dobimo enačbo za izračun lokalne gostote:

$$\gamma_k^i = \frac{1}{1 + \rho \frac{\sum_{j=1}^{M^i} (\mathbf{x}_k - \mathbf{x}_{kj}^i)^T (\mathbf{A}^i)^{-1} (\mathbf{x}_k - \mathbf{x}_{kj}^i)}{M^i}} \tag{2.17}$$

kjer ρ označuje volumen oblaka (t.i. področje vpliva). V [42] je podana tudi rekurzivna oblika enačbe (2.17), ki se glasi:

$$\gamma_k^i = \frac{1}{1 + \rho \left[(\mathbf{x}_k - \boldsymbol{\mu}_k^i)^T (\mathbf{A}^i)^{-1} (\mathbf{x}_k - \boldsymbol{\mu}_k^i) + \frac{(M^i - 1)q}{M^i} \right]} \tag{2.18}$$

kjer je q dimenzija vhodnega vektorja $\mathbf{x}_k$ in $\boldsymbol{\mu}_k^i$ srednja vrednost oblaka, izračunana z enačbo (2.14).

Glavna prednost uporabe Mahalanobisove razdalje je ta, da upošteva elipsoidno obliko podatkov, ki pripadajo določenemu oblaku. Čeprav je ta ideja precej upravičena, prinaša tudi nekaj dodatnih problemov. V primeru, če je oblak sestavljen iz majhnega števila podatkov, ki ležijo precej skupaj, potem je tudi kovariančna matrika $\mathbf{A}^i$ majhna. Drugače povedano: tudi če bo kakšen podatek relativno blizu oblaka bo lokalna gostota še vedno precej majhna saj je volumen oblaka majhen. Da bi preprečili tak pojav, so avtorji v [41] razširili enačbo (2.18) in normalizirali inverzno vrednost kovariančne matrike, kot sledi:

$$\gamma_k^i = \frac{1}{1+\rho\left[\frac{(\mathbf{x}_k-\boldsymbol{\mu}_k^i)^T(\mathbf{A}^i)^{-1}(\mathbf{x}_k-\boldsymbol{\mu}_k^i)}{\sqrt{\det((\mathbf{A}^i)^{-1})}}+\frac{(M^i-1)q}{M^i}\right]} \tag{2.19}$$

Izračun rekurzivne lokalne gostote z (2.18) in (2.19) zahteva izračun inverzne vrednosti kovariančne matrike ob vsakem koraku. Da bi se izognili temu, lahko uporabimo Woodburyjevo matrično identiteto [111] in tako dobimo naslednji psevdo algoritem [41]:

$$M^i \leftarrow M^i + 1 \tag{2.20}$$

$$\mathbf{d} \leftarrow \mathbf{x} - \boldsymbol{\mu}^i \tag{2.21}$$

$$\boldsymbol{\mu} \leftarrow \boldsymbol{\mu}^i + \frac{1}{M^i}\mathbf{d} \tag{2.22}$$

$$\bar{\mathbf{S}}^i \leftarrow \bar{\mathbf{S}}^i - \bar{\mathbf{S}}^i\mathbf{d}\left[1+(\mathbf{x}-\boldsymbol{\mu}^i)^T\bar{\mathbf{S}}^i\mathbf{d}\right](\mathbf{x}-\boldsymbol{\mu}^i)^T\bar{\mathbf{S}}^i \tag{2.23}$$

$$(\mathbf{A}^i)^{-1} \leftarrow (M^i-1)\bar{\mathbf{S}}^i \tag{2.24}$$

kjer z vpeljavo novega stanja $\bar{\mathbf{S}}^i$ posredno izračunamo inverzno vrednost kovariančne matrike $(\mathbf{A}^i)^{-1}$. Pred začetkom postopka (ob dodajanju novega oblaka oziroma pravila) moramo inicializirati stanja $\bar{\mathbf{S}}^i$ (ponavadi z veliko diagonalno matriko).

3. Vodenje nelinearnih procesov s samorazvijajočim se adaptivnim regulatorjem

3.1. Uvod in pregled literature

Vodenje nelinearnih, dinamičnih in kompleksnih sistemov je dandanes še vedno aktivno raziskovalno področje. Ne glede na spreminjajoče se zunanje vplive, spremenljivo procesno dinamiko in večanja kompleksnosti procesov v industriji ostajajo zahteve po visoki kakovosti in varnosti delovanja vseh procesov enake. Z lokalno linearno aproksimacijo procesov lahko poenostavimo problem načrtovanja algoritma vodenja. Tovrsten pristop ponuja enostavne in zadovoljive rezultate le v okolici lokalnega modela in pod statičnimi pogoji ter ni primeren za celotno operacijsko območjo nelinearnega procesa.

V literaturi lahko najdemo veliko različnih primerov in pristopov, kako rešiti problem nelinearnosti procesa. V [61] je predstavljena samonastavitvena metoda PID-regulatorja za nelinearne probleme, ki temelji na Ljapunovi teoriji. Nekoliko drugačen pristop je predstavljen v [132], kjer je sprotno učenje uporabljeno za napovedovanje dinamike procesa in adaptacijo parametrov PID-regulatorja, in temelji, tako kot v prejšnji metodi, na teoriji Ljapunova. Poleg tradicionalnih adaptivnih metod obstajajo tudi bolj napredne metode, ki uporabljajo znanje o procesih. Na primer: v [212] so avtorji predstavili sprotno adaptacijo parametrov regulatorja na osnovi nevronskih mrež (ang. *neural networks*), medtem ko so v [9] z genetskim algoritmom (ang. *genetic algorithm*) poiskali optimalne parametre regulatorja. Tudi optimizacija z roji delcev (ang. *particle swarm optimization*) daje zadovoljive rezultate pri nastavljanju parametrov PID-regulatorja [182].

Vse zgoraj naštete metode so namenjene sprotnemu nastavljanju parametrov klasičnega PID-regulatorja. Številni raziskovalci so se posvetili tudi nadaljnji raziskavi in razširitvi koncepta PID-regulatorja. S tem namenom so razvili PID-regulator z delnim redom (ang. *fractional order PID*, FrPID), ki deluje bolje kot klasični PID-regulator [196], vendar pa zahteva določitev še dveh dodatnih parametrov. Podobno kot pri klasični verziji obstajajo tudi v tem primeru opti-

mizacijske rešitve pri nastavljanju parametrov. V [173, 239] so avtorji predstavili rešitev optimalnega FrPID-regulatorja na osnovi genetskega algoritma, medtem ko je v [183] predstavljen koncept optimizacije na osnovi rojev delcev.

Mehka logika (podpoglavje 2.1) predstavlja alternativen pristop, ki je bil razvit z namenom reševanja problema nelinearnosti procesov z razdelitvijo območja delovanja na več manjših linearnih področij. Zahvaljujoč njeni prilagodljivosti in zmogljivosti modeliranja nelinearnih procesov je bila mehka logika uporabljena v različnih aplikacijah [35, 40, 48, 49, 206] Kot smo že omenili v prejšnjem poglavju, je mehko logiko prvič predstavil prof. Zadeh [266] in pozneje je nastalo več različic oziroma izpeljank te teorije, npr. Takagi-Sugeno, Mamdani in podobno. Eden prvih prispevkov na področju vodenja, vključujoč mehko logiko, je [176], ki obravnava problem vodenja dinamičnega modela parnega motorja.

Na splošno lahko metode vodenja, ki temeljijo na teoriji mehke logike, razdelimo v 6 skupin [93]:

1. *konvencionalno mehko vodenje (na podlagi modela Mamdani)*,
2. *mehko PID vodenje*,
3. *nevromehko vodenje*,
4. *mehko vodenje v drsnem režimu*,
5. *adaptivno mehko vodenje*,
6. *vodenje na podlagi modela Takagi-Sugeno*.

Potrebno je omeniti, da pogostokrat ni natančne definicije med posameznimi pristopi in te se namreč velikokrat tudi prekrivajo. Na primer: konvencionalni mehki regulator je lahko adaptiven, mehki PID-regulator je lahko nastavljen z uporabo nevromehkega sistema ali nevromehki regulator je lahko adaptiven itn. Vsi ti pristopi so bili aplicirani na različnih akademskih in industrijskih problemih. Mehki regulatorji so dosegli velik uspeh na več različnih področjih: telekomunikacije [66, 130, 273]; energetski sistemi [1, 94, 146]; mehatronika/robotika [35, 39, 49, 104, 141]; kemijska industrija [65, 87, 257]; zdravstvo in številna druga področja.

Poleg zgoraj naštetih mehkih metod vodenja so bili z iznajdbo samorazvijajočih se sistemov (Poglavje 2) razviti tudi mehki regulatorji, ki vključujejo samorazvijajnje [169]. Ena prvih objav na tem področju je predstavljena v [14], kjer predlagana rešitev obravnava problem nastavljanja mehkih pravil s sprotnim učenjem na podlagi podatkov. Rešitev je bila uspešno preizkušena na primeru klimatske naprave. V [79] so predstavili metodo, ki s sprotnim učenjem na podlagi podatkov zgradi (dodaja in briše) pravila mehkega modela Takagi-Sugeno. Avtorji v [43, 47, 85, 268, 269] so predstavili različne tipe samorazvijajočih se regulatorjev, ki temeljijo na metodi eFuMo [86], ki služi kot orodje za razdelitev nelinearnega področja procesa. Omenimo tudi delo v [200], kjer je predstavljeno vodenje aerodinamičnega sistema z dvojnim rotorjem na osnovi samorazvijajočega se modela Takagi-Sugeno. Podobno delo je predstavljeno tudi v [228], kjer so avtorji uporabili načela sprotnega učenja nevromehkega modela. Z istim pristopom in metodologijo kot v [200] so avtorji v [201] poskušali rešiti problem vodenja protetične roke. Nekoliko drugačen koncept samorazvijajočega se regulatorja je bil predstavljen v [155], kjer je mehki model zasnovan na podlagi granul. Granule predstavljajo mehka področja in delijo globalni problem nelinearnosti procesa na manjša lokalna linearna področja. Struktura samorazvijajočega se mehkega [56] in nevromehkega [55] modela se razvija od začetka (ang. *from scratch*) oziroma brez predhodnega znanja o lastnosti procesa. V obeh primerih je pri izračunu izhoda regulatorja uporabljen produkt za konjunkcijo mehkih pravil in uteženo povprečje. V [193] so avtorji predstavili regulacijski sistem s samorazvijajočo se strukturo na osnovi Gaussovih procesov. Regulator je namenjen vodenju dinamičnih sistemov in predpostavlja nekaj osnovnega znanja o procesu.

Z iznajdbo neparametričnega mehkega modela AnYa (glej razdelek 2.1.3) so se začele raziskave, ki so poskušale združiti uporabnost in enostavnost modela AnYa z znanjem s področja vodenja. Najprej so v [22] so predstavili samorazvijajoči se regulator, ki svojo strukturo razvija (dodaja nova mehka pravila) na podlagi lokalne in globalne gostote sprotno sprejetih podatkov. V posledičnem delu mehkega modela so predstavili adaptivni mehanizem za sprotno adaptacijo parametrov dveh tipov regulatorjev, in sicer PID in MRC (ang. *model reference control*). Svojo rešitev so avtorji preizkusili na simulacijskem primeru nelinearnega hidravličnega procesa. S praktično enakim pristopom kot v prejšnjem primeru so avtorji v [78] predstavili praktično implementacijo regulatorja na realnem

sistemu dveh tankov. Regulator je uspešno sledil spremembi reference, vendar je pokazal zelo slabe rezultate v ustaljenem stanju. Sledil je razvoj novega regulatorja [229] (na podlagi [22]), kjer dodajanje novih mehkih pravil (oblakov) ne temelji na globalni gostoti podatkov, temveč so avtorji vpeljali prag (ang. *threshold*), na podlagi katerega se dodajajo novi oblaki. Regulator je bil preizkušen na dveh simulacijskih nelinearnih primerih. V obeh primerih z enakimi začetnimi nastavitvami je regulator deloval uspešno (z minimalnim pogreškom) pri sledenju referenci in v ustaljenem stanju.

V nadaljevanju bomo predstavili samorazvijajoči se robusten regulator, ki temelji na mehkem sistemu AnYa. Ideja je, da bi z osnovnih znanj o procesu (območje vhodnega in izhodnega signala, ocenjena/želena časovna konstanta in čas vzorčenja) nastavili potrebne začetne parametre regulatorja, kar predvsem olajša delo operaterja.

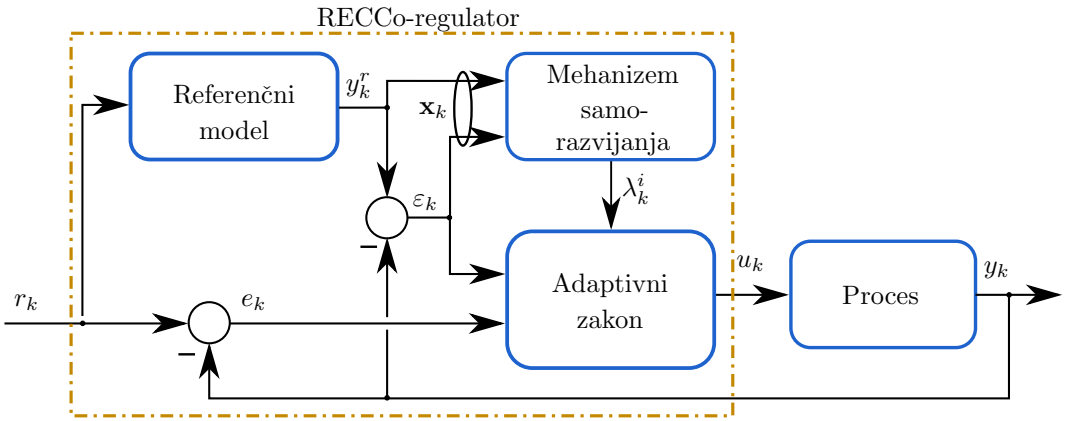
3.2. Robusten samorazvijajoči se adaptivni regulator RECCo

V tem podpoglavju bomo razložili splošno strukturo in mehanizme delovanja RECCo-regulatorja (ang. *robust evolving cloud-based controller*). Algoritem RECCo je bil predstavljen v [10] in vsebuje tri sklope: *referenčni model*, *mehanizem samorazvijanja* in *adaptivni zakon*. Vsak sklop je prikazan na sliki 3.1 in bo podrobno opisan v nadaljevanju. Teoretično se lahko regulator nastavi samodejno, vendar nam vsako dodatno znanje o procesu lahko bistveno olajša postopek nastavljanja začetnih parametrov in na ta način izboljšamo delovanje regulatorja. Z vsakim naslednjim prejetim podatkom algoritem samodejno razvija svojo strukturo in adaptira parametre regulatorja s ciljem zmanjšanja pogreška med želeno in dejansko vrednostjo reguliranega signala. Kot je razvidno iz slike 3.1, regulacijski algoritem RECCo predstavlja pravzaprav indirektni adaptivni regulator [116].

RECCo-regulator temelji na mehkem sistemu AnYa (razdelek 2.1.3) in ima naslednjo obliko:

$$\mathcal{R}^i: \quad \textbf{IF} \quad (\mathbf{x} \sim X^i) \quad \textbf{THEN} \quad (u^i), \qquad i = 1, \ldots, c \tag{3.1}$$

kjer $\mathcal{R}^i$ označuje i-to mehko pravilo in trenutno število mehkih pravil c je enako številu obstoječih podatkovnih oblakov X^i. Pogojni (**IF**) del sistema je definiran



Slika 3.1: Regulacijska shema samorazvijajočega se RECCo-regulatorja.

z matematičnim operatorjem $\sim$, ki ga preberemo *je povezan z* (ang. *is associated with*) in označuje podobnost med podatkom $\mathbf{x} = [\mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(n)]^T$ in oblakom podatkov X^i. Posledični (**THEN**) del je definiran z lokalnim regulatorjem u^i.

Stopnja aktivacije posameznega pravila je izražena z normalizirano lokalno gostoto λ_k^i, izračunano med trenutnim podatkom $\mathbf{x}$ in določenim oblakom X^i:

$$\lambda_k^i = \frac{\gamma_k^i}{\sum_{j=1}^{c} \gamma_k^j}, \qquad i = 1, \ldots, c \tag{3.2}$$

kjer je γ_k^i lokalna gostota med trenutnim podatkom in i-tim oblakom. Lokalna gostota je ključnega pomena pri mehanizmu samorazvijanja, ki bo obravnavan v razdelku 3.2.2.

3.2.1 Referenčni model

Izbira primernega referenčnega modela je ključnega pomena za pravilno delovanje adaptivnega algoritma. Splošna priporočila za izbiro referenčnega modela so, da mora biti časovna konstanta modela enaka oziroma nekoliko krajša kot časovna konstanta procesa [118]. Prav tako mora biti red referenčnega modela manjši ali enak redu modela procesa [139]. Poleg tega je priporočljivo, da je začetna vrednost referenčnega modela enaka izhodni vrednosti procesa ($y_0^r = y_0$).

Referenčni model definira želeno trajektorijo y_k^r in dinamiko, za katero hočemo, da ji regulirani signal y_k sledi. V tem primeru, na podlagi napotkov

v [139], smo se odločili za model prvega reda, ki ima naslednjo obliko:

$$y_{k+1}^r = a_r y_k^r + (1 - a_r) r_k, \qquad 0 < a_r < 1 \tag{3.3}$$

kjer je r_k referenčni signal; parameter a_r predstavlja pol modela in ga lahko aproksimiramo z $(1 - \frac{T_s}{\tau})$, T_s je čas vzorčenja in τ je želena časovna konstanta zaprtozančnega sistema. Cilj našega regulacijskega algoritma je, da bi regulirani signal y_k čim bolje sledil referenčnemu signalu y_k^r oziroma si želimo čim manjši sledilni pogrešek med njima:

$$\varepsilon_k = y_k^r - y_k \tag{3.4}$$

Na tem mestu je treba omeniti, da RECCo-regulator ni omejen le na tip referenčnega modela (3.3), ampak se lahko glede na dinamiko in lastnosti procesa uporabi tudi drugačen referenčni model.

3.2.2 Mehanizem samorazvijanja

V tem razdelku bomo razložili drugi korak RECCo-regulatorja, in sicer mehanizem samorazvijanja modela, ki definira pravila dodajanja novih mehkih pravil ter mehanizme zaznavanja osamelcev (ang. *outlier*). Mehanizem samorazvijanja v tem primeru sestoji le iz mehanizma za dodajanje novih oblakov podatkov oziroma mehkih pravil[1]. Kot bomo videli v naslednjih poglavjih, je možno dodati tudi mehanizme za odstranjevanje oblakov. V primeru vodenja smo se odločili samo za »konzervativne« mehanizme dodajanja novih oblakov, ker nam olajša problem dodajanja novih delnih regulatorjev (to bo obravnavano v naslednjem razdelku).

Mehanizem za dodajanje oblakov temelji na lokalni gostoti γ_k^i, ki se izračuna med trenutno sprejetim podatkom $\mathbf{x}_k$ in vsemi obstoječimi oblaki X^i, $i = 1, \ldots, c$. Po definiciji lokalna gostota upošteva vse podatke, ki pripadajo določenemu oblaku, in je izračunana z uporabo poljubnega jedra $\mathcal{K}$:

$$\gamma_k^i = \mathcal{K}\left(\sum_{j=1}^{M^i} d_{kj}^i\right) \tag{3.5}$$

kjer je M^i število podatkov, ki pripadajo i-temu oblaku in d_{kj}^i razdalja med trenutnim $\mathbf{x}_k$ in j-tim podatkom iz oblaka. V našem primeru smo izbrali Cauchyjevo

[1] V disertaciji termin mehka pravila predstavlja oblake podatkov in obratno.

jedro, kot je bilo predlagano v [24], ter Evklidsko mero za izračun razdalje med dvema podatkoma. Na podlagi izpeljave, ki je bila izvedena v razdelku 2.1.3, dobimo rekurzivno obliko za izračun lokalne gostote:

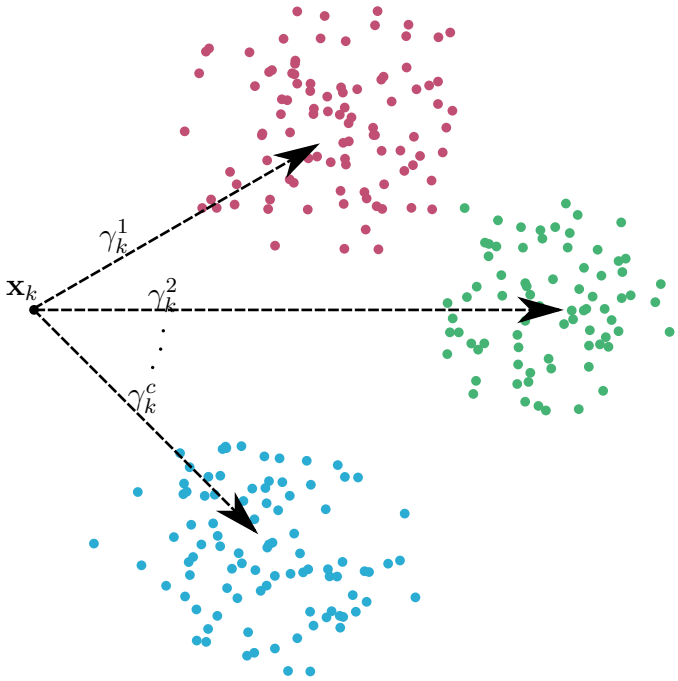
$$\gamma_k^i = \frac{1}{1 + \|\mathbf{x}_k - \boldsymbol{\mu}_k^i\|^2 + \sigma_k^i - \|\boldsymbol{\mu}_k^i\|^2}, \qquad i = 1, \ldots, c$$

$$\boldsymbol{\mu}_k^i = \frac{M^i - 1}{M^i}\boldsymbol{\mu}_{k-1}^i + \frac{1}{M^i}\mathbf{x}_k, \qquad \boldsymbol{\mu}_1^i = \mathbf{x}_1$$

$$\sigma_k^i = \frac{M^i - 1}{M^i}\sigma_{k-1}^i + \frac{1}{M^i}\|\mathbf{x}_k\|^2, \qquad \sigma_1^i = \|\mathbf{x}_1\|^2$$

kjer je μ_k^i srednja vrednost in σ_k^i povprečna vsota kvadrata dolžin podatkov, ki pripadajo i-temu oblaku.

Kot smo že omenili, **mehanizem samorazvijanja** modela temelji na dodajanju novih mehkih pravil (oblakov podatkov). Velja opomniti, da se prvi oblak X^1 inicializira s prvim prejetim podatkom $\mathbf{x}_1$. Mehanizem dodajanja novih pravil temelji na toku podatkov (ang. *data stream*), kar pomeni, da z vsakim sprejetim podatkom preverjamo, če so izpolnjeni določeni pogoji. Za vsak prejet podatek izračunamo c lokalnih gostot $\gamma_k^i, i = 1, \ldots, c$ (med trenutnim podatkom in obstoječimi oblaki) ter poiščemo maksimalno vrednost med njimi. Če je maksimalna gostota $\max_i(\gamma_k^i)$ manjša kot predhodno nastavljen prag γ_{max}, potem je prvi pogoj za dodajanje novega oblaka izpolnjen. Da bi hkrati preprečili »eksplozijo« oblakov, dodamo dodaten konzervativen pogoj, da mora od zadnjega dodanega oblaka miniti n_{add} vzorcev. Če sta oba pogoja izpolnjena, potem se nov oblak (pravilo) inicializira s trenutnim vzorcem. Če pa trenutni vzorec ne izpolnjuje pogojev dodajanja, potem se ta vzorec (podatek) razvrsti k oblaku, ki ima največjo lokalno gostoto. Teoretično ima lahko določen vzorec enako lokalno gostoto z več kot z enim samim oblakom. V takem primeru pripada ta podatek starejšemu oblaku (tistemu, ki je bil dodan prej). Na koncu dodamo še eno omejitev (če vnaprej poznamo strukturo procesa), ki določa maksimalno število oblakov c_{max}, ki jih lahko dodamo. Na sliki 3.2 je prikazan 2D grafičen primer povezovanja trenutnega podatka $\mathbf{x}_k$ z obstoječimi oblaki na podlagi lokalne gostote.



Slika 3.2: Prikaz trenutnega podatka $\mathbf{x}_k$ in obstoječimi oblaki podatkov.

3.2.3 Adaptivni zakon

Mehanizmu samorazvijanja strukture iz prejšnjega razdelka sledi adaptacija parametrov (adaptivni zakon). Lahko rečemo, da je mehanizem samorazvijanja del pogojnega (**IF**) dela mehkega sistema AnYa (3.1), medtem ko je adaptivni zakon del posledičnega (**THEN**) dela.

V posledičnem delu celotnega regulacijskega algoritma RECCo smo vsakemu izmed oblakov iz pogojnega dela dodelili PID-R tip regulatorja. Število lokalnih regulatorjev je enako številu obstoječih oblakov c, ki se zaradi samorazvijajoče se narave tudi spreminja. Diskretni regulator PID-R ima naslednjo obliko:

$$u_k^i = P_k^i \varepsilon_k + I_k^i \Sigma_k^\varepsilon + D_k^i \Delta_k^\varepsilon + R_k^i, \qquad i = 1, \ldots, c \tag{3.6}$$

kjer so P_k^i, I_k^i, in D_k^i parametri regulatorja in R_k^i kompenzator delovne točke i-tega lokalnega modela, ε_k je sledilni pogrešek, Σ_k^ε je vsota sledilnega pogreška (ima značaj integracije), Δ_k^ε pa diferenca sledilnega pogreška:

$$\Sigma_k^\varepsilon = \sum_{\kappa=0}^{k-1} \varepsilon_\kappa = \Sigma_{k-1}^\varepsilon + \varepsilon_{k-1} \tag{3.7}$$

$$\Delta_k^\varepsilon = \varepsilon_k - \varepsilon_{k-1} \tag{3.8}$$

Dodatno je enačba (3.7) pogojena z zaščito pred integralskim pobegom:

$$\Sigma_k^{\varepsilon} = \begin{cases} \Sigma_{k-1}^{\varepsilon} + \varepsilon_{k-1}, & u_{min} < u_k < u_{max} \\ \Sigma_{k-1}^{\varepsilon}, & u_k <= u_{min} \textbf{ or } u_k => u_{max} \end{cases} \tag{3.9}$$

Vektor parametrov posameznega (delnega) regulatorja je definiran z $\boldsymbol{\theta}_k^i = [P_k^i, I_k^i, D_k^i, R_k^i]^T$, parametri prvega oblaka pa so inicilaizirani z $\boldsymbol{\theta}_0^1 = [0, 0, 0, 0]^T$, medtem ko so parametri vseh nadaljnjih regulatorjev nastavljeni, kot sledi:

$$\boldsymbol{\theta}_0^c = \frac{1}{c-1} \sum_{j=1}^{c-1} \boldsymbol{\theta}_k^j \tag{3.10}$$

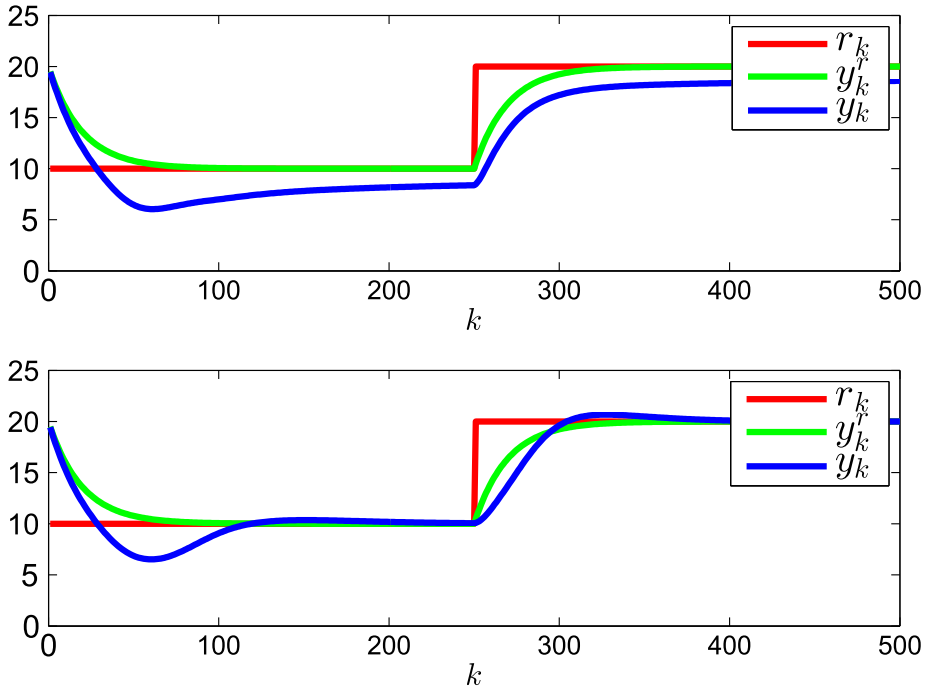
V primeru, da trenutni podatek ne izpolnjuje pogojev za dodajanje novega oblaka in se razvrsti k enemu od obstoječih oblakov, se parametri regulatorja tega oblaka adaptirajo na podlagi naslednjega zakona:

$$\boldsymbol{\theta}_k^i = \boldsymbol{\theta}_{k-1}^i + \Delta\boldsymbol{\theta}_k^i \tag{3.11}$$

kjer $\Delta\boldsymbol{\theta}_k^i$ predstavlja korak adaptacije, ki je definiran za vsak parameter posebej, in sicer kot sledi:

$$\begin{aligned} \Delta P_k^i &= \alpha_P \, G_{sign} \lambda_k^i \frac{|e_k \varepsilon_k|}{1 + r_k^2} \\ \Delta I_k^i &= \alpha_I \, G_{sign} \lambda_k^i \frac{|e_k \Delta_k^{\varepsilon}|}{1 + r_k^2} \\ \Delta D_k^i &= \alpha_D \, G_{sign} \lambda_k^i \frac{|e_k \Delta_k^{\varepsilon}|}{1 + r_k^2} \\ \Delta R_k^i &= \alpha_R \, G_{sign} \lambda_k^i \frac{\varepsilon_k}{1 + r_k^2} \end{aligned} \tag{3.12}$$

kjer so $\alpha_P, \alpha_I, \alpha_D$ in α_R koeficienti ojačenja adaptacije, $G_{sign} = \pm 1$ je znan predznak procesa, λ_k^i normirana lokalna gostota (3.2), ε_k sledilni pogrešek (3.4), $e_k = r_k - y_k$ pa je izhodni pogrešek. Absolutna vrednost v enačbi (3.12) se uporablja le za obdobje prvih nekaj časovnih konstant (npr. pet časovnih konstant), potem pa se adaptacija nadaljuje brez upoštevanja absolutnih vrednosti. Problem na začetku adaptacije nastane, ker so začetni parametri regulatorja inicializirani z ničlami in hkrati navzdol omejeni z nič. Če sta produkta $e_k \varepsilon_k$ in $e_k \Delta_k$ na začetku negativna, pomeni, da bo adaptacija (3.12) (brez absolutnih vrednosti) želela parametre dodatno zmanjševati, kar pa ni mogoče. Zato z absolutnimi vrednostmi



Slika 3.3: Prikaz vpliva izračuna brez (zgornja slika) in z (spodnja slika) absolutnimi vrednosti v enačbi (3.12). Prikaz reference r_k (rdeča črta), izhoda referenčnega modela y_k^r (zelena črta) in izhoda procesa y_k (modra črta).

»prisilimo« adaptacijo v pozitivno smer. Na sliki 3.3 je prikazana razlika odzivov v primerih z in brez upoštevanja absolutnih vrednosti v (3.12). Bolj podrobna analiza vpeljave absolutnih vrednosti v adaptivnem zakonu je bila izvedena v [11].

Privzeta vrednost koeficientov adaptacije $\alpha_P, \alpha_I, \alpha_D$ in α_R je odvisna od območja regulirnega signala $\Delta u = u_{max} - u_{min}$. Iz izkušenj smo ocenili, da jo podaja naslednja relacija:

$$\alpha_{new} = \Delta u \cdot 0.5\,\% \tag{3.13}$$

Primer: če je razpon regulirnega signala od $u_{min} = 0$ do $u_{max} = 100$, potem je nova vrednost koeficientov enaka $\alpha_{new} = 0.5$.

Predstavljeni adaptivni zakon v enačbi (3.12) kot kriterijsko funkcijo uporablja normiran produkt med sledilnim in izhodnim pogreškom. Normiranje se v praksi izvede s kvadratom referenčnega signala $(1 + r_k^2)$. Več o adaptivnih zakonih z

normiranjem je razloženo v [118], kjer je prikazana in dokazana tudi stabilnost takih pristopov na osnovi Lyapunove teorije.

Izhod regulatorja se izračuna kot utežena vsota vseh lokalnih regulatorjev PID-R (ostrenje z uteženim povprečenjem):

$$u_k = u_{min} + \sum_{i=1}^{c} \lambda_k^i u^i = u_{min} + \frac{\sum_{i=1}^{c} \gamma_k^i u^i}{\sum_{i=1}^{c} \gamma_k^i} \tag{3.14}$$

kjer je u_{min} minimalna vrednost regulirnega signala.

3.2.4 Zaščitni mehanizmi adaptacije

V tem razdelku bomo posvetili večjo pozornost adaptivnemu zakonu in z uvedbo nekaterih zaščitnih mehanizmov poskušali povečati robustnost zaprtozančnega sistema (slika 3.1). Teoretično (in tudi praktično) lahko z nadzorovano adaptacijo izboljšamo stabilnost, učinkovitost in robustnost regulacijskega sistema [45, 46, 118, 213]. S ciljem zmanjševanja vpliva parazitne dinamike procesa in motenj v sistemu ter odpravo slabe integracije adaptivnega zakona smo v algoritmu RECCo uvedli dodatne zaščitne mehanizme, ki bodo predstavljeni v nadaljevanju.

Mrtva cona v adaptivnem zakonu

Adaptacija parametrov v zaprtozančnih sistemih predstavlja potencialno nevarnost pri zagotavljanju stabilnosti sistema. Adaptacija je vodena na podlagi obeh pogreškov (sledilnega ε_k in izhodnega e_k, glej (3.12)), ki sta sestavljena v glavnem iz koristne komponente, a vsebujeta tudi škodljivo komponento (zaradi različnih motenj ali parazitne/ne-modelirane dinamike procesa). V praksi je vpliv koristne komponente veliko večji kot vpliv škodljive in zato je smiselno omejiti (ustaviti) adaptacijo parametrov ob majhnih spremembah pogreška. To je tudi posledica drugih zaščitnih mehanizmov adaptacije razloženih v tem razdelku. Glavna ideja mrtve cone v adaptivnem zakonu je ustaviti adaptacijo parametrov, če je vrednost pogreška manjša od določene vrednosti [194]:

$$\Delta\boldsymbol{\theta}_k^i = \begin{cases} \Delta\boldsymbol{\theta}_k^i, & |\varepsilon_k| \geq d_{dead} \\ 0, & |\varepsilon_k| < d_{dead} \end{cases} \tag{3.15}$$

Vrednost parametra d_{dead} izberemo nekoliko večjo, kot so ocenjene motnje v procesu, s tem izboljšamo učinkovitost adaptivnega zakona. Prevelika vrednost parametra povzroči krajši čas adaptacije (hitrejša konvergenca parametrov), vendar na račun prevelikega pogreška v ustaljenem stanju, medtem ko lahko (pre)majhna vrednost parametra d_{dead} povzroči lezenje parametrov.

Projekcija parametrov

Naraven način, kako preprečiti lezenje parametrov, je projekcija le-teh v znanem omejenem prostoru [149]. V primeru procesa s pozitivnim ojačenjem morajo biti parametri navzdol omejeni z 0. Zgornjo mejo parametrov je nekoliko težje določiti brez dodatnega znanja o procesu. V našem primeru smo se odločili, da parametre navzgor pustimo neomejene. Izkazalo se je, kar bo razvidno iz rezultatov v nadaljevanju, da taka odločitev ni napačna, ker parametri konvergirajo. Adaptivni zakon (3.12) smo razširili v bolj splošno obliko, kot sledi:

$$\boldsymbol{\theta}_k^i = \begin{cases} \boldsymbol{\theta}_{k-1}^i + \Delta\boldsymbol{\theta}_k^i, & \underline{\boldsymbol{\theta}} \leq \boldsymbol{\theta}_{k-1}^i + \Delta\boldsymbol{\theta}_k^i \leq \overline{\boldsymbol{\theta}} \\ \underline{\boldsymbol{\theta}}, & \boldsymbol{\theta}_{k-1}^i + \Delta\boldsymbol{\theta}_k^i < \underline{\boldsymbol{\theta}} \\ \overline{\boldsymbol{\theta}}, & \boldsymbol{\theta}_{k-1}^i + \Delta\boldsymbol{\theta}_k^i > \overline{\boldsymbol{\theta}} \end{cases} \tag{3.16}$$

kjer je $\underline{\boldsymbol{\theta}}$ minimalna in $\overline{\boldsymbol{\theta}}$ maksimalna vrednost določenega parametra. V našem primeru smo izbrali $\underline{\boldsymbol{\theta}} = 0$ in $\overline{\boldsymbol{\theta}} = \infty$ za P_k, I_k, in D_k parametre regulatorja, kompenzator delovne točke R_k pa nima nobene omejitve. Kot smo že omenili, nam vsako dodatno znanje o procesu ali parametrih procesa lahko olajša postopek nastavljanja začetnih parametrov. V primeru, da bi *a priori* poznali ožje področje znotraj matematičnega prostora parametrov $\mathbb{R}^n$, lahko meje projekcije definiramo v tem ožjem področju. Taka informacija bi bistveno pospešila proces adaptacije parametrov in bi zato hitreje prišli v bližino pravih vrednosti $\boldsymbol{\theta}^*$.

Adaptivni zakon s puščanjem

Uporaba adaptivnega zakona s puščanjem (ang. *leakage*) je že uveljaven pristop in v literaturi zasledimo različne tipe adaptivnih zakonov s puščanjem, kot so sigma-modifikacija [117], e_1-modifikacija [186] in preklopna σ-modifikacija [119].

Z uvedbo puščanja v adaptivnem zakonu (3.11) dobimo:

$$\boldsymbol{\theta}^i = (1 - \sigma_L)\boldsymbol{\theta}^i_{k-1} + \Delta\boldsymbol{\theta}^i_k \tag{3.17}$$

kjer σ_L definira obseg puščanja, ki je ponavadi majhna vrednost ranga 10^{-5} – 10^{-6}.

Prekinitev adaptacije

Poleg zgoraj naštetih mehanizmov, ki so posledica neznanih oziroma nezaželenih situacij, smo dodali še mehanizem za prekinitev adaptacije v primeru, če aktuator (regulirni signal) doseže svojo maksimalno u_{max} ali minimalno u_{min} vrednost. V tem primeru lahko pride do navzkrižja interesov med prehitro želeno dinamiko in zmožnostjo fizičnega aktuatorja. V primeru, da regulirni signal doseže svoj minimum oziroma maksimum ustavimo adaptacijo na naslednji način:

$$\Delta\boldsymbol{\theta}^i_k = \begin{cases} \Delta\boldsymbol{\theta}^i_k, & u_{min} \leq u_k \leq u_{max} \\ 0, & \text{sicer} \end{cases} \tag{3.18}$$

3.2.5 Normiranje vhodno-izhodnega prostora

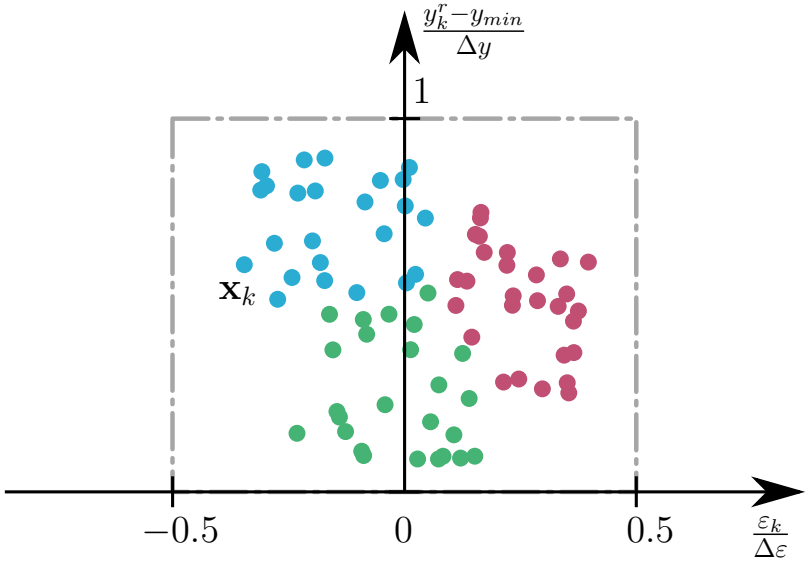
Do sedaj nismo omenjali, kako je definiran podatek $\mathbf{x}_k$ in katere komponente vsebuje, zato v nadaljevanju to podrobneje opišemo. V prejšnjih različicah algoritma RECCo [229] je bil vhodni podatek definiran v dvodimenzionalnem prostoru kot $\mathbf{x}_k = [\varepsilon_k, y^r_k]^T$. Ta izbira vhodnega podatka močno vpliva na parameter γ_{max}, ki ga je potrebno določiti pred vsakim eksperimentom.

Naša ideja je definirati omejen prostor podatkov (slika 3.4), kjer bomo dobivali podatke ne glede na to, kakšen je razpon signalov $y_{min}, y_{max}, u_{min}$ in u_{max}. To dosežemo z normiranjem prostora:

$$\mathbf{x} = \left[\begin{array}{cc} \frac{\varepsilon_k}{\Delta\varepsilon}, & \frac{y^r_k - y_{min}}{\Delta y} \end{array} \right]^T \tag{3.19}$$

kjer je $\Delta y = y_{max} - y_{min}$ in $\Delta\varepsilon = \frac{\Delta y}{2}$. V tem primeru operater izbere le območje delovanja procesa $[y_{min}, y_{max}]$.

V eksperimentalnem delu tega podpoglavja bomo pokazali tudi vpliv izbire vhodno-izhodnega prostora.



Slika 3.4: Prikaz normiranega vhodno-izhodnega prostora.

3.3. Nastavitev RECCo-regulatorja v praksi

V tem podpoglavju bomo pokazali, kako praktično nastavimo RECCo-regulator, njegove parametre in začetne vrednosti. V naslednjih poglavjih bomo predstavili rezultate vodenja na simulacijskih modelih kot tudi na realnih napravah. Potrebno je poudariti, da začetna nastavitev RECCo-regulatorja velja za vse eksperimente, ki jih bomo pokazali. Celoten regulacijski algoritem RECCo, ki smo ga podrobno opisali v prejšnjih poglavjih, je povzet v algoritmu 1, kjer so v obliki psevdokode prikazani vsi njegovi koraki. Kot smo že omenili, regulator RECCo se razvija od začetka (ang. *from scratch*), torej brez začetnih oblakov oziroma pravil. Prvi oblak se inicializira s prvim sprejetim podatkom in parametri regulatorja se nastavijo z vrednostjo nič. Potem se začne proces samorazvijanja strukture in adaptacije parametrov.

Za začetno nastavitev RECCo-regulatorja ne potrebujemo matematičnih enačb modela procesa ali kakršnihkoli drugih predpostavk (npr. o Gaussovi porazdelitvi podatkov). Potrebujemo le osnovne informacije o procesu, ki ga želimo voditi, kot so delovno območje reguliranega signala $[y_{min}, y_{max}]$, območje regulirnega signala $[u_{min}, u_{max}]$, ocenjena oziroma želena časovna konstanta τ zaprtozančnega sistema in čas vzorčenja T_s. Nekateri nastavitveni parametri algoritma se izračunajo na podlagi omenjenih parametrov procesa, drugi pa imajo vnaprej privzete vrednosti. V vseh eksperimentih, ki jih bomo predstavili v nadaljevanju,

bomo določili omenjene nastavitve in zagnali algoritem RECCo.

Kot smo pokazali na sliki 3.1, je RECCo-regulator sestavljen iz treh sklopov. Tako bomo razdelili tudi nastavitvene parametre. V prvo skupino (razdelek 3.2.1) spada pol referenčnega modela a_r (3.3), ki ga izračunamo na osnovi časa vzorčenja in želene časovne konstante zaprtozančnega sistema:

- $a_r \approx 1 - T_s/\tau$.

Drugi sklop parametrov se nanaša na samorazvijajoči se del RECCo-regulatorja (razdelek 3.2.2):

- $\gamma_{max} = 0.93$ – prag lokalne gostote,
- $c_{max} = 20$ – maksimalno število mehkih oblakov (pravil),
- $n_{add} = 20$ – minimalno število korakov med dodajanjem novih oblakov.

Sledijo parametri adaptivnega zakona (razdelka 3.2.3 in 3.2.4):

- $\alpha_P, \alpha_I, \alpha_D$ in α_R – ojačenja adaptacije (3.13),
- $d_{dead} = \Delta y/100$ – mrtva cona predstavlja 1 % območja regulirane veličine,
- $\left[\underline{\boldsymbol{\theta}}, \overline{\boldsymbol{\theta}}\right] = [0, \infty]$ – projekcija parametrov,
- $\sigma_L = 10^{-6}$ – puščanje adaptivnega zakona.

Potem ko so vsi parametri regulatorja določeni, zgradimo referenčni signal r_k tako, da definiramo stopničaste spremembe znotraj območja reguliranega signala $[y_{min},\ y_{max}]$.

3.4. Eksperimenti na simulacijskih modelih

3.4.1 Primerjava med RECCo-regulatorjem in klasičnim PID-regulatorjem na modelu toplotnega izmenjevalnika

V tem razdelku bomo prikazali rezultate primerjave med klasičnim PID-regulatorjem in samorazvijajočim se RECCo-regulatorjem. Najprej bomo na kratko opisali model toplotnega izmenjevalnika, saj bomo v naslednjih poglavjih

Algoritem 1 Psevdokoda RECCo-regulatorja.

Inicializacija (parametri procesa): τ, T_s, u_{min}, u_{max}, y_{min}, y_{max}.
Inicializacija (parametri samorazvijajočega se sistema): $\gamma_{max} = 0.93$, $c = 0$, $c_{max} = 20$, $n_{add} = 20$.
Inicializacija (parametri adaptivnega zakona): α_R, α_P, α_I, α_D, d_{dead}, $\underline{\boldsymbol{\theta}}$, $\overline{\boldsymbol{\theta}}$, σ_L.
repeat
    Meritev: y_k.
    Izračunaj: y_k^r z uporabo (3.3). ▷ *Referenčni model*
    Izračunaj: e_k, ε_k, Σ_k^{ε}, Δ_k^{ε}.
    Izračunaj: $\mathbf{x}_k$ z uporabo (3.19).
    if $c = 0$ **then** ▷ *Začetek algoritma samorazvijanja*
        Povečaj indeks: c,
        Shrani: k_{add}^c,
        Inicializacija: $\boldsymbol{\mu}_0^1$, σ_0^1, $\boldsymbol{\theta}_0^1$.
    else
        Izračunaj: γ_k^i, λ_k^i, za $i = 1, \ldots, c$
        if $(\gamma_{max} > max_i(\gamma_k^i)$ **and** $k > (k_{add} + n_{add})$ **and** $c_{max} > c)$ **then**
            Povečaj indeks: c,
            Shrani: k_{add}^c,
            Inicializacija: $\boldsymbol{\mu}_0^c$, σ_0^c, $\boldsymbol{\theta}_0^c$.
        else
            Povežemo podatek $\mathbf{x}_k$ z i-tem oblaku $(max_i\gamma_k^i)$.
            Posodobimo parametre $\boldsymbol{\mu}_k^i$, σ_k^i i-tega oblaka $(max_i\gamma_k^i)$.
        end if
    end if ▷ *Konec algoritma samorazvijanja*
    Adaptacija parametrov regulatorja PID-R z uporabo (3.12).
    Izračun **regulirnega** signala z uporabo (3.14).
    Preverimo kriterije za povečanje robustnosti (3.15), (3.16), (3.17), (3.18).
until Konec toka podatkov.

podrobneje opisali realni proces izmenjevalnika. Model toplotnega izmenjevalnika v tem razdelku je prevzet iz literature [232]. Proces je sestavljen iz dveh ločenih tokokrogov (topla in hladna veja). Prvi (topli) tokokrog ima konstantno vhodno

temperaturo medija T_{ec} in motorno gnan ventil, ki skrbi za ustrezen pretok vode F_c, ki predstavlja naš regulirni signal u_k. Izhod iz izmenjevalnika je povezan z rezervoarjem, kjer se medij ponovno segreva z grelcem na nastavljeno konstantno temperaturo T_{ec}. Drugi (hladni) tokokrog ima neznano hladnejšo vhodno temperaturo medija T_{ep} s konstantnim pretokom F_p. Izhodna temperatura medija T_{sp} predstavlja regulirano veličino našega procesa. Proces je opisan z naslednjo diferencialno enačbo [232]:

$$\tau_2(T_{sp})\dot{T}_{sp} + T_{sp} = \delta T_{ep} + (1-\delta)T_{ec} \tag{3.20}$$

kjer se koeficient δ izračuna z naslednjim izrazom:

$$\delta = \frac{1 + k_c(\frac{1}{F_c})^m}{1 + k_c((\frac{1}{F_c})^m + (\frac{1}{F_p})^m)} \tag{3.21}$$

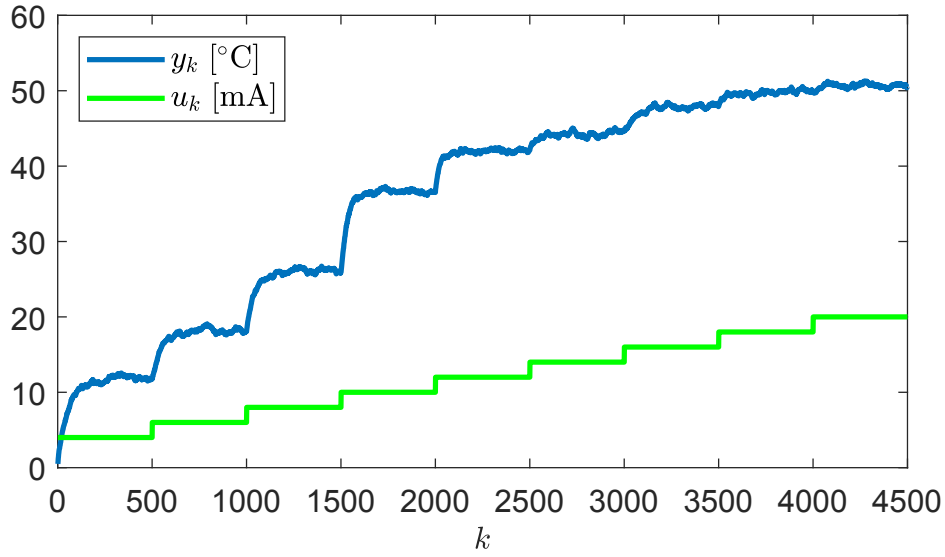
kjer sta k_c in m neznani konstanti in τ_2 neznana funkcija. V [232] je bil predstavljen mehki model procesa toplotnega izmenjevalnika, ki smo ga uporabili za testiranje regulatorjev.

Odprtozančen odziv modela toplotnega izmenjevalnika je prikazan na sliki 3.5. Že iz samega odziva lahko opazimo, da je model izrazito nelinearen. Iz karakteristike lahko tudi kvantitativno ocenimo karakteristične parametre modela za vsako delovno točko. V tabeli 3.1 so prikazani ojačenja K_P, časovne konstante τ_P in mrtvi časi $T_{dead,P}$ za vsako delovno točko posebej. Razen mrtvega časa, ki se ne spreminja, se časovne konstante in ojačenja zelo razlikujejo v različnih delovnih točkah. To predstavlja velik izziv pri načrtovanju klasičnih PID-regulatorjev.

Tabela 3.1: Simulacija. Karakteristični parametri modela procesa toplotnega izmenjevalnika: ojačenje K_P, časovna konstanta τ_P in mrtvi čas $T_{dead,P}$.

u_k [mA]	6	8	10	12	14	16	18	20
K_P	3.38	4.17	4.93	2.71	1.21	1.73	1.10	0.68
τ_P [s]	48	50	27	19	18	47	44	29
$T_{dead,P}$ [s]	4	4	4	4	4	4	4	4

V nadaljevanju bomo najprej predstavili postopek nastavitve in rezultate RECCo-regulatorja in nato bomo rezultate primerjali z različnimi nastavitvenimi metodami za PID-regulatorje. Kot smo opisali v prejšnjem podpoglavju 3.3, je



Slika 3.5: Simulacija. Odprtozančni odziv modela toplotnega izmenjevalnika.

nekaj parametrov algoritma RECCo nastavljenih vnaprej, druge pa izračunamo na podlagi lastnosti procesa. Parametri procesa toplotnega izmenjevalnika in začetni parametri RECCo-regulatorja so podani v tabeli 3.2.

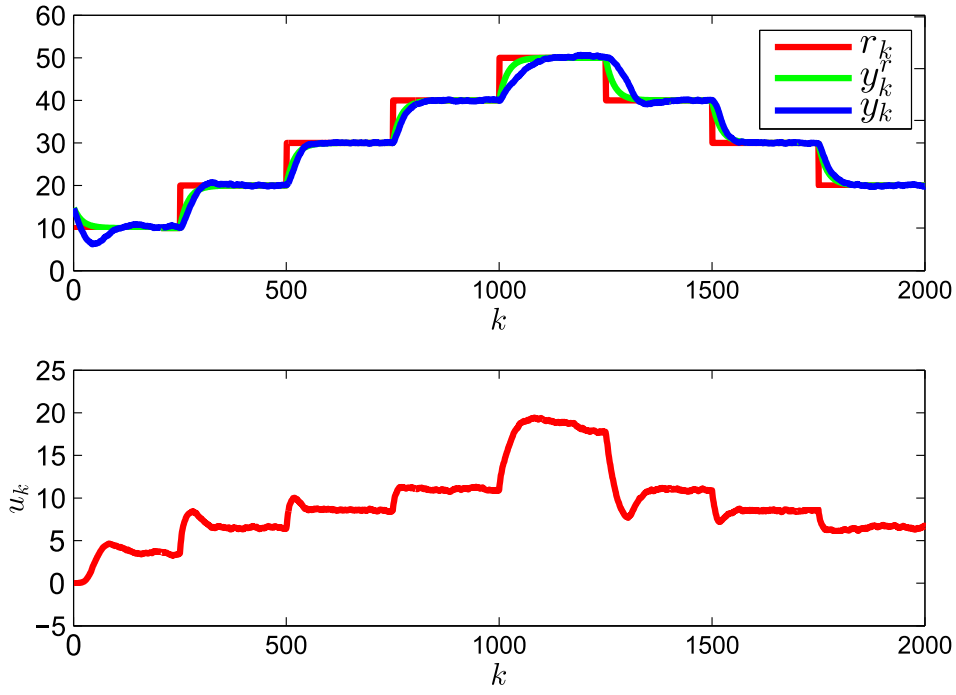
Tabela 3.2: Parametri procesa in nastavitveni parametri RECCo-regulatorja.

Parametri procesa					
u_{min}	u_{max}	y_{min}	y_{max}	τ	T_s
0	20	0	60	35 s	2 s

Parametri RECCo								
a_r	γ_{max}	c_{max}	n_{add}	α_{new}	d_{dead}	$\underline{\boldsymbol{\theta}}$	$\overline{\boldsymbol{\theta}}$	σ_L
0.95	0.93	20	20	0.1	0.6	0	∞	10^{-6}

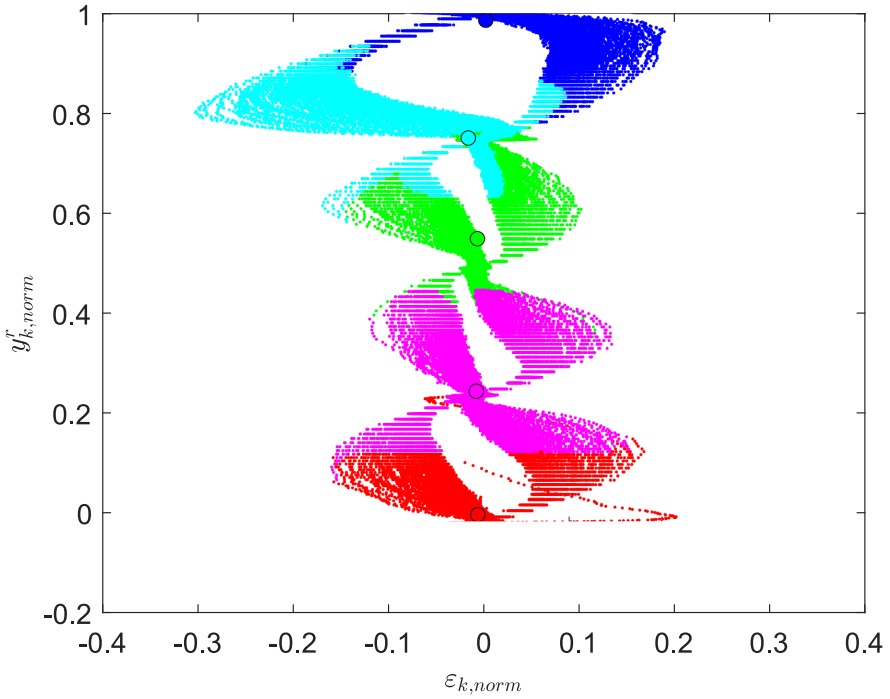
Učinkovitost sprotnega učenja iz podatkov in lastnost samorazvijanja RECCo-regulatorja bosta prikazana na naslednjih nekaj slikah. Najprej smo ustvarili želeni referenčni signal r_k, ki je sestavljen iz 5 stopničastih sprememb (navzgor iz y_{min} proti y_{max} in obratno). Na sliki 3.6 je prikazana začetna faza vodenja algoritma RECCo, kjer so prikazani referenčni signal r_k, izhod referenčnega modela y_k^r ter regulirni u_k in regulirani signal y_k. Na samem začetku eksperimenta (s prvim prejetim podatkom) se inicializira prvi oblak in pripadajoči parametri

regulatorja PID-R se postavijo na nič. Kljub takšni nastavitvi RECCo-regulator hitro prilagodi strukturo in adaptira parametre regulatorja, da doseže čim manjši pogrešek v ustaljenem stanju. Po večkratni ponovitvi sekvence, prikazane na sliki 3.6, se mehka struktura razvije in ustvari 5 oblakov (slika 3.7a). Hkrati se odvija postopek adaptacije parametrov vseh delnih regulatorjev PID-R, katerih vrednosti so prikazane na sliki 3.7b. Barve prikazanih parametrov na sliki 3.7b se ujemajo z barvami pripadajočega oblaka na sliki 3.7a. Na sliki 3.8 je prikazan rezultat vodenja v končni fazi učenja, kjer vidimo, da v primerjavi s sliko 3.6 regulirani signal y_k lepo sledi izhodu referenčnega modela y_k^r. Za boljši prikaz učinkovitosti učenja in prilagajanja smo na sliki 3.9 prikazali sledilni pogrešek skozi celotni postopek učenja in vodenja procesa. Vidimo, da je na začetku eksperimenta ta pogrešek največji in nato upada s časom. Kot enega od kriterijev za ustavitev učenja lahko zato izkoristimo sledilni pogrešek [118].

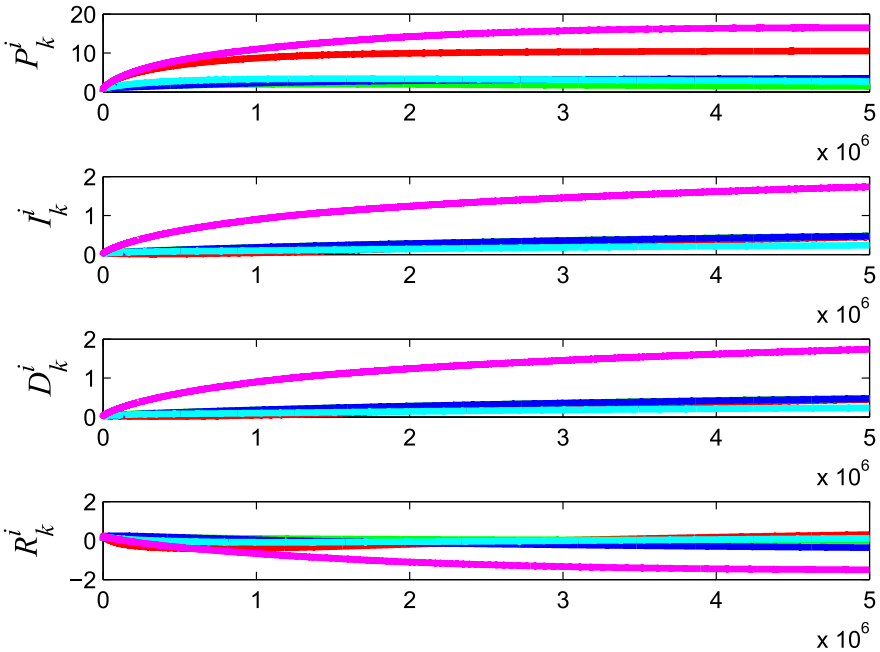


Slika 3.6: Simulacija – začetek učenja. Prikaz reference r_k, izhoda referenčnega modela y_k^r in izhoda procesa y_k (zgornja slika) ter regulirnega signala u_k (spodnja slika).

V nadaljevanju bomo kvantitativno primerjali rezultate RECCo-regulatorja

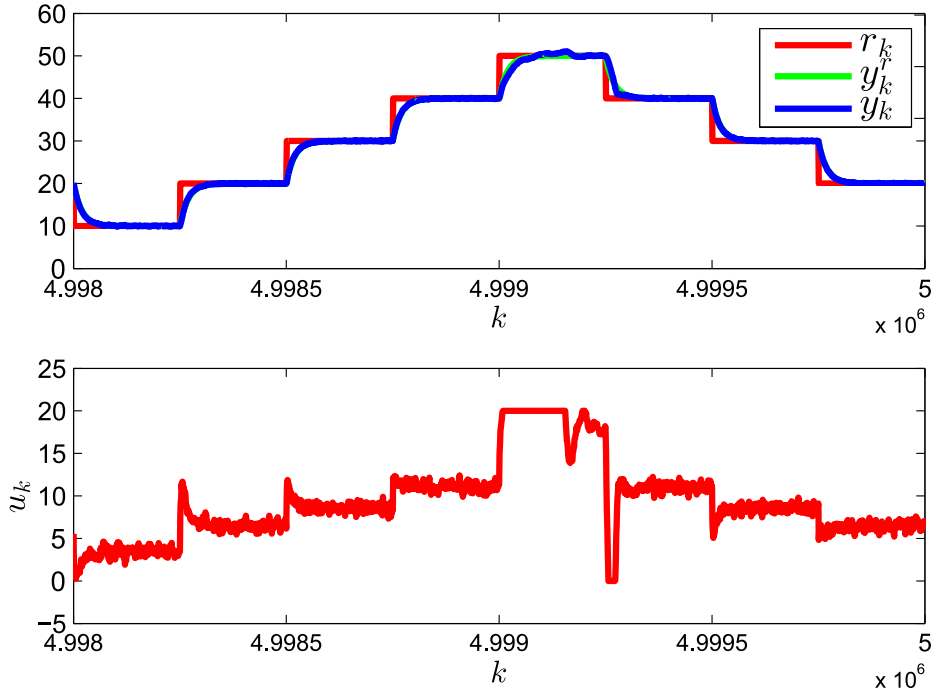


(a) Prikaz oblakov dodanih z samorazvijajočim se RECCo-regulatorjem, kjer je ε_k sledilni pogrešek in y^r_k izhod referenčnega modela.



(b) Prikaz adaptacije parametrov regulatorja PID-R.

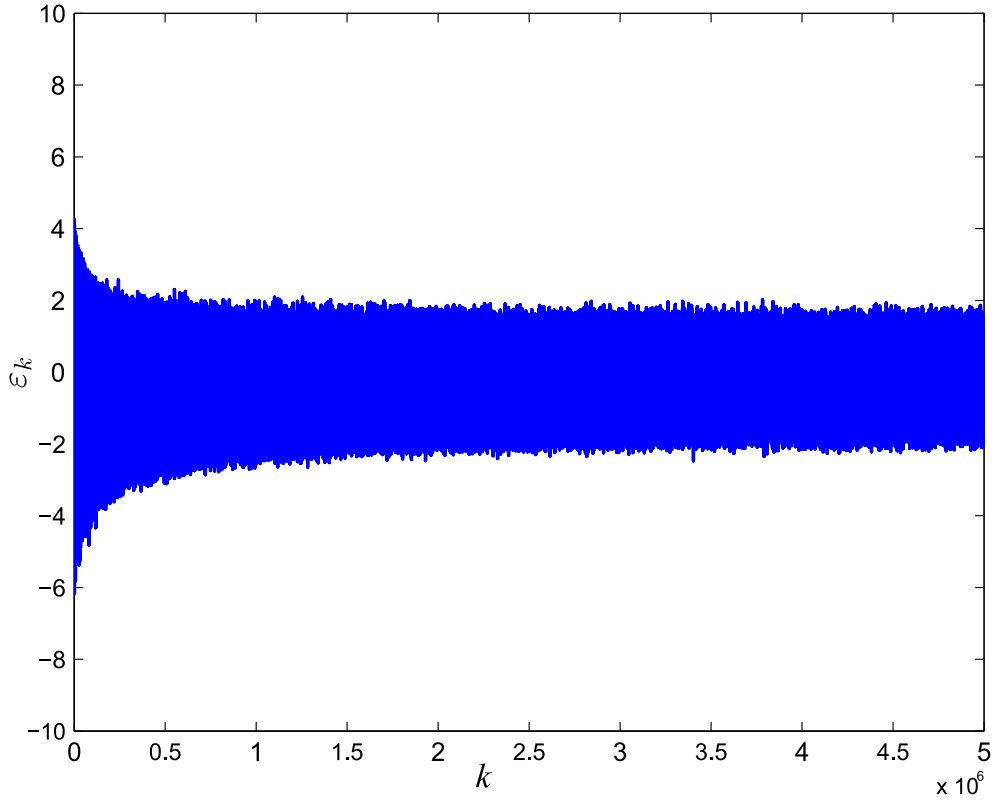
Slika 3.7: Simulacija. Prikaz razvijanja strukture mehkega modela in adaptacija parametrov regulatorja.



Slika 3.8: Simulacija - konec učenja. Prikaz reference r_k, izhoda referenčnega modela y_k^r in izhoda procesa y_k (zgornja slika) ter regulirnega signala u_k (spodnja slika).

s klasičnimi nastavitvenimi pravili za določanje parametrov PID-regulatorjev. PID-regulator velja za najbolj razširjen pristop v industriji. V [82] je bilo objavljeno poročilo, da je v industriji PID-regulator uporabljen v več kot 90 % vseh regulacijskih procesih. Zato želimo primerjati učinkovitost predlagane metode s klasičnimi pristopi. Uporabili bomo tri različne metode: Ziegler–Nicholsovo metodo [277] (PID_{ZN}), metodo Cohen–Coon [73] (PID_{CC}) in metodo s premikanjem polov [28] (PID_{PP}). Za določitev parametrov PID-regulatorja potrebuje vsaka od teh metod ojačenje procesa, časovno konstanto in čas vzorčenja. Glede na to, da želimo z PID-regulatorjem pokriti celotno delovno območje procesa, moramo izračunati povprečne vrednosti (iz tabele 3.1) ojačenja $\hat{K}_P = 2.49$ in časovne konstante $\hat{\tau}_P = 35.25$, medtem ko je čas vzorčenja $T_s = 2\,\mathrm{s}$.

Za kvantitativno primerjanje rezultatov med različna regulacijska sistema, PID in RECCo, smo uporabili več kriterijev, kot so: čas vzpona (od 10 % do 90 % kočne vrednosti reference r_k), maksimalni prevzpon in čas umiritve (e_k v usta-



Slika 3.9: Simulacija. Prikaz sledilnega pogreška ε_k.

ljenem stanju manjši od 0.25 °C). Poleg tega smo primerjali še vsoto absolutnih vrednosti (ang. *sum of the absolute input differences*, f_{SAdU}) in vsoto kvadratnih vrednosti (ang. *sum of the squared input differences*, f_{SSdU}) sprememb regulirnih signalov.

$$f_{SAdU} = \sum_k |\Delta u_k| \tag{3.22}$$

$$f_{SSdU} = \sum_k \Delta u_k^2 \tag{3.23}$$

kjer je $\Delta u_k = u_k - u_{k-1}$ sprememba regulirnega signala.

Poleg teh dveh kriterijev, (3.22) in (3.23), smo uporabili še vsoto absolutnih f_{SAE} in kvadratnih f_{SSE} vrednosti izhodnega pogreška ($e_k = r_k - y_k$):

$$f_{SAE} = T_s \sum_k |e_k| \tag{3.24}$$

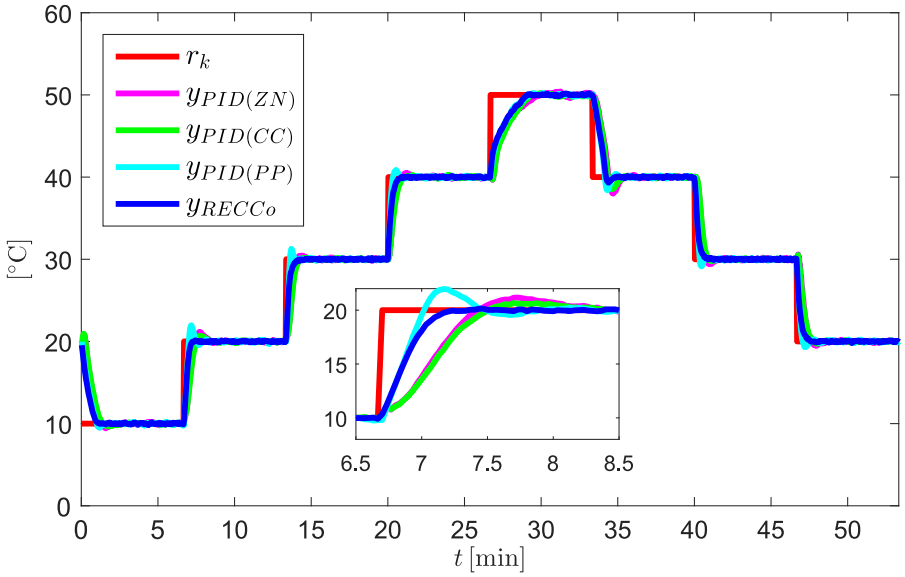
$$f_{SSE} = T_s \sum_k e_k^2 \tag{3.25}$$

Iz tabele 3.3 je razvidno, da RECCo-regulator v primerjavi z PID-regulatorjem daje boljše rezultate. Potrebno je izpostaviti, da ima regulator PID_{PP} hitrejši čas vzpona na račun večjega maksimalnega prevzpona in daljšega časa umiritve. V tabeli 3.4 so prikazane vrednosti kriterijev (3.22), (3.23), (3.24) in (3.25), kjer lahko vidimo, da daje RECCo-regulator boljše rezultate.

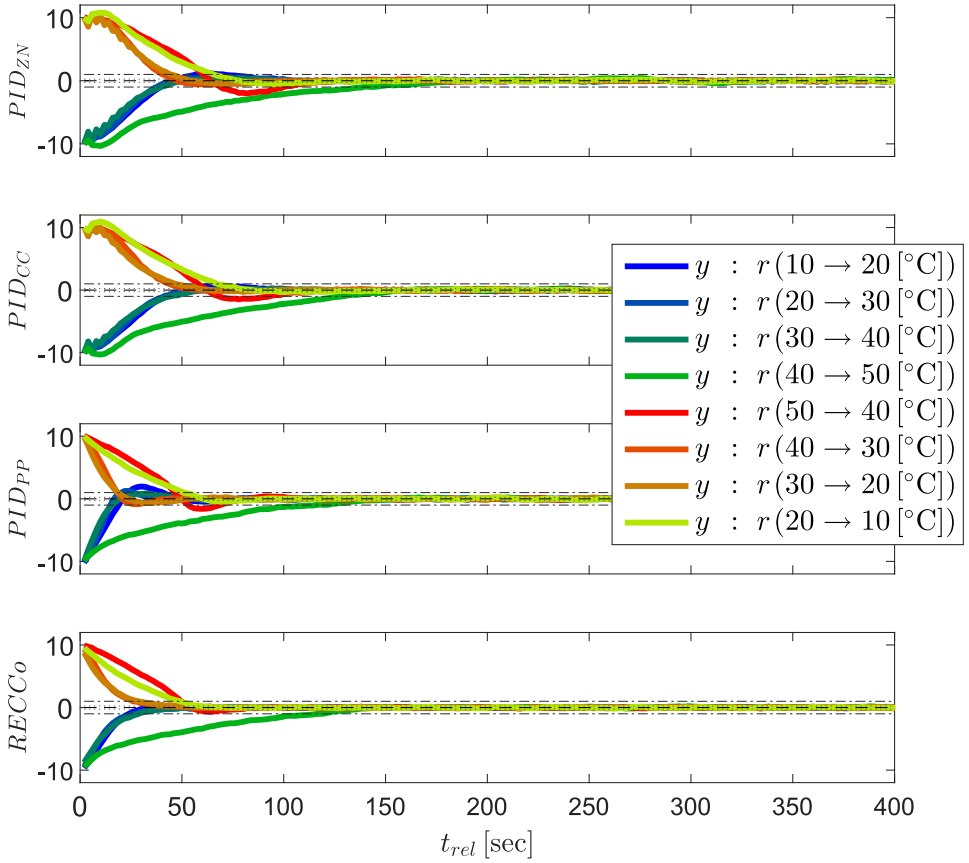
Tabela 3.3: Simulacija. Primerjava zmogljivosti regulatorjev v več delovnih točkah.

r_k [°C]	10	20	30	40	50	40	30	20
				Čas vzpona [s]				
PID_{ZN}	44	36	34	36	122	42	26	42
PID_{CC}	46	40	38	42	112	42	28	44
PID_{PP}	46	14	12	12	110	38	14	16
$RECCo$	42	20	24	28	104	36	22	28
				Maksimalni prevzpon [%]				
PID_{ZN}	5.8	11.7	4.5	5.0	4.2	20.1	5.9	5.1
PID_{CC}	3.5	6.6	1.6	2.9	2.2	14.8	2.5	3.0
PID_{PP}	5.1	19.6	12.7	8.6	2.3	16.0	8.6	7.1
$RECCo$	1.9	1.4	1.6	1.3	2.0	6.8	1.3	2.0
				Čas umiritve [s]				
PID_{ZN}	116	98	78	94	382	168	78	100
PID_{CC}	122	96	54	242	154	114	68	82
PID_{PP}	150	238	34	48	144	102	38	74
$RECCo$	60	30	42	42	138	80	42	48

Na sliki 3.10 so prikazani odzive regulatorjev v več delovnih točkah. Na sliki 3.10a je prikazana celotna sekvenca referenčnega signala r_k z rdečo barvo (stopničaste spremembe navzgor in navzdol) in odzivi vseh regulatorjev ($y_{PID(ZN)}$ – vijolična, $y_{PID(CC)}$ – zelena, $y_{PID(PP)}$ – svetlo modra, y_{RECCo} – modra barva). Na sliki 3.10b pa so prikazani odzivi za vsako delovno točko, ustrezno premaknjeni na isto časovno skalo.



(a) Primerjava odzivov regulacijskih sistemov v eni sekvenci in povečan prikaz odzivov za prvo stopničasto spremembo.



(b) Združeni prikaz odzivov vseh delovnih točkah (ustrezno premaknjeni).

Slika 3.10: Simulacija. Primerjava odzivov regulacijskih sistemov.

Tabela 3.4: Simulacija. Primerjava zmogljivosti regulatorjev za različne kriterijske funkcije.

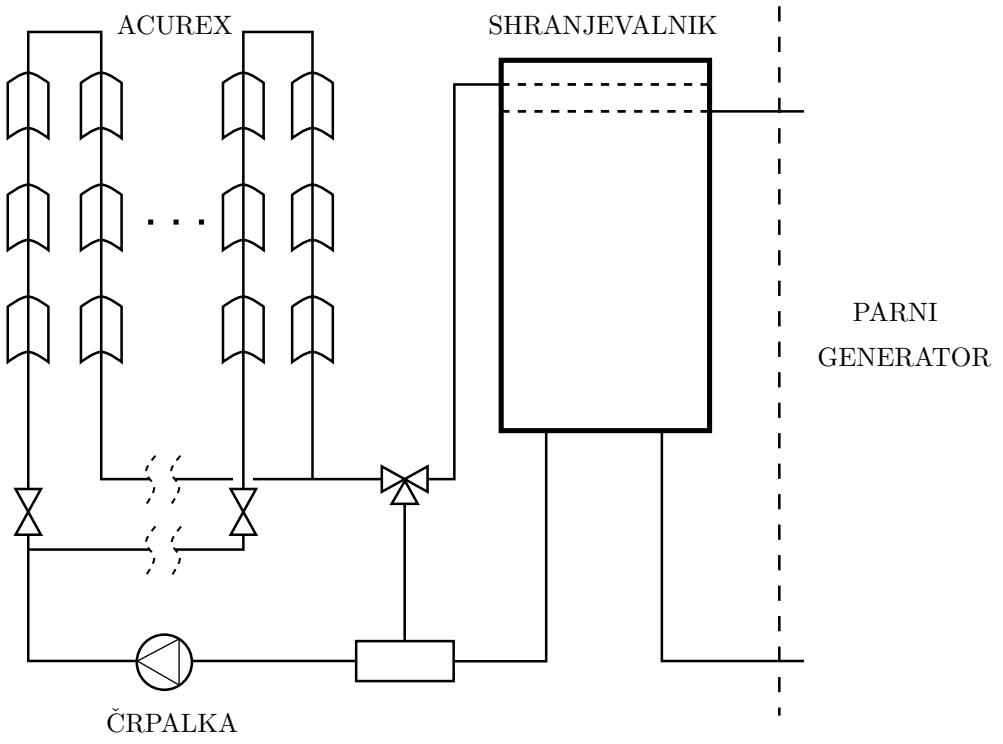
	f_{SAdU} [A]	f_{SSdU} [A^2]	f_{SAE} [°C]	f_{SSE} [$°C^2$]
PID_{ZN}	2.93	18.52	1419	25430
PID_{CC}	2.87	18.64	1121	20277
PID_{PP}	1.77	9.28	327	6110
$RECCo$	1.59	6.48	309	5129

3.4.2 Model sončnega kolektorja

Porazdeljene sončne kolektorje (ang. *distributed solar collector field*, DSCF) se uporablja za absorpcijo sončnega sevanja in shranjevanje toplotne energije v zaprtem toplovodnem sistemu. Medij, ki prenaša toploto (ang. *heat transfer fluid*, HTF), je lahko sanitarna voda ali posebno termalno olje. Ta medij se potem uporablja za prenos toplote (do toplotnega shranjevalnika) z namenom proizvodnje električne energije (npr. parna turbina). Glavni cilj regulacije v sistemu DSCF je regulirati in vzdrževati želeno vrednost temperature medija HTF, in sicer kljub spremenljivim in nepredvidljivim razmeram okolice.

V okviru doktorske disertacije je bila izvedena študija na modelu sončnega kolektorja, zgrajenega na podlagi realnega kolektorskega sistema ACUREX [51]. Sistem ACUREX (slika 3.11) se nahaja v Almeriji v Španiji in je sestavljen iz 480 medsebojno povezanih sončnih kolektorjev, ki so organizirani v desetih vzporednih zankah. Vsaka zanka je dolga 172 metrov, od katerih je 142 metrov aktivnih (sprejemajo sončne žarke) in 30 metrov je pasivnih. Uporabljen HTF medij je termalno olje Therminol®55, ki se lahko segreje maksimalno do 300 °C. Termalno olje se s črpalko požene iz dna shranjevalnika (ang. *storage tank*), nato preko sončnih kolektorjev, kjer se pobere toplota, se vrne nazaj na vrh shranjevalnika. Toplota v shranjevalniku se uporabi za uparjanje vode, ki potem žene parni generator za proizvodnjo električne energije.

Upoštevajoč določene predpostavke, lahko obnašanje kolektorskega sistema modeliramo na podlagi parcialnih diferencialnih enačb za opis energijskega rav-



Slika 3.11: Shematski prikaz sončnega kolektorja ACUREX [51].

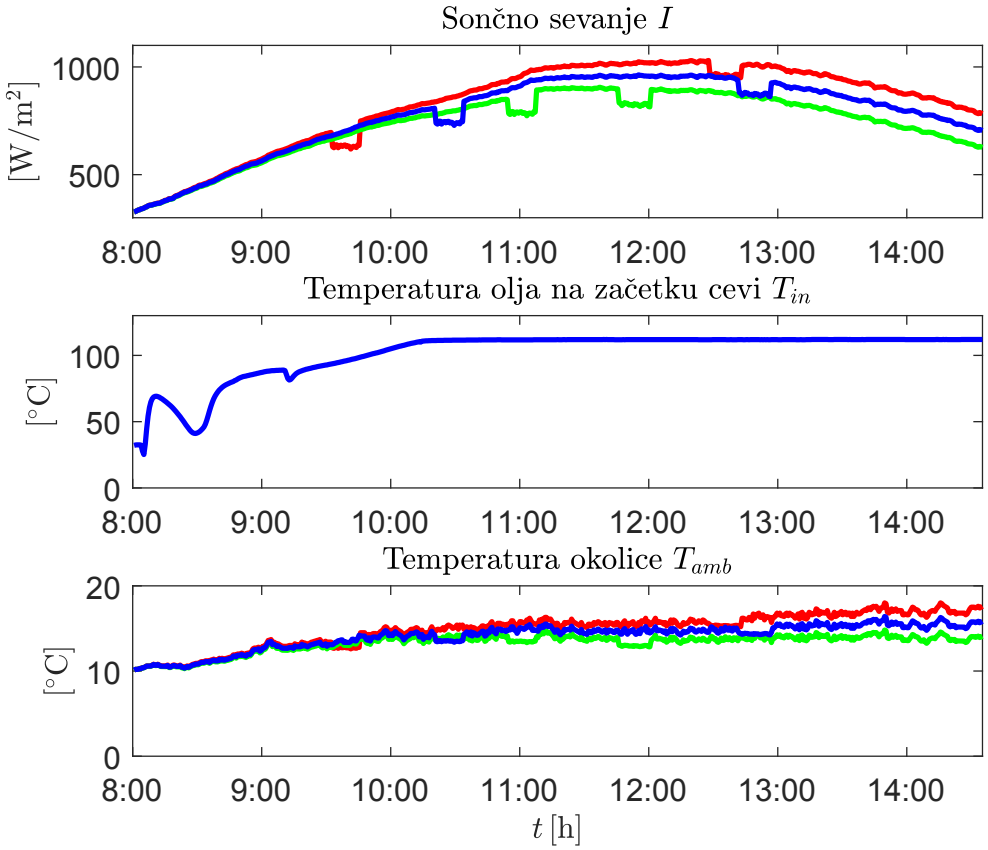
novesja [52]:

$$\begin{aligned}\rho_m c_m A_m \frac{\partial T_m}{\partial t}(t,l) &= \eta_{col} G I(t) - P_{rc} - D_f \pi H_t \left[T_m(t,l) - T_f(t,l)\right] \\ \rho_f c_f A_f \frac{\partial T_f}{\partial t}(t,l) + \rho_f c_f q(t) \frac{\partial T_f}{\partial l}(t,l) &= D_f \pi H_t \left[T_m(t,l) - T_f(t,l)\right]\end{aligned} \tag{3.26}$$

kjer je indeks m oznaka za kovinske cevi in indeks f oznaka za HTF medij, T je temperatura, ρ gostota medija, c specifična toplota, A površina prečnega prereza, t čas, l dolžina, η_{col} globalna učinkovitost kolektorja, G zaslonka kolektorja, I neposredno sončno sevanje, P_{rc} konvekcijske toplotne izgube, D premer cevi, H_t konvekcijski koeficient prenosa toplote in q volumski pretok olja.

Pri izvedbi vodenja je treba vedeti, da regulirna veličina predstavlja volumski pretok olja q, medtem ko je regulirana veličina temperatura olja na koncu cevi ($T_{out}(t) = T_f(t,L)$, kjer je L celotna dolžina cevi). Glavni problem ali izziv v sistemu DSCF so zunanje motnje (lahko jih merimo) oziroma nepredvidljiv osnovni vir energije, sončno sevanje I, temperatura olja na začetku cevi $T_f(t,0) =$

T_{in} in temperatura okolice T_{amb}. Na sliki 3.12 je prikazan nabor signalov, ki so bili pridobljeni na lokaciji v Španiji, kjer je inštaliran sistem ACUREX. Tem signalom smo dodali še nekaj naključnih motenj v obliki stopničastih sprememb in dodaten odmik (približna simulacija oblačnih obdobij v dnevu). Signali so bili posneti v časovni dolžini 6,5 ur (od 8h do 14:30h).



Slika 3.12: Motnje na sistemu DSCF.

V praksi se je izkazalo, da se lahko s krmilnikom, ki upošteva motnje (ang. *feedforward controller*), znatno izboljšamo vodenje zaprtozančnega sistema [51]. Vhod v direktni krmilnik (krajše za krmilnik z upoštevanjem motnje) je referenčna temperatura ($u_k = T_{ref_{ff}}$), medtem ko je izhod volumski pretok olja q, ki se izračuna na podlagi merljivih motenj (I, T_{in} in T_{amb}), sončne entalpije ΔH in povprečne temperature T_m.

V doktorski disertaciji [57] je bil razvit krmilnik z upoštevanjem motnje, kot

sledi:

$$\begin{aligned} X_1(t) &= \frac{R \cdot I(t) \cdot \cos(\Theta_i) \cdot A}{\rho_f(t) \cdot \Delta H(t)} \\ X_2(t) &= \frac{T_m(t) + T_{amb}(t)}{\rho_f(t) \cdot \Delta H(t)} \end{aligned} \tag{3.27}$$

kjer je R odbojnost, Θ_i vpadni kot in A učinkovita površina. Povprečna temperatura T_m in sončna entalpija sta izračunani kot sledi:

$$T_m = \frac{T_{ref_{ff}} + T_{in}}{2} \tag{3.28}$$

$$\Delta H = 1.882 \cdot 10^{-3} \cdot (T_{ref_{ff}}^2 - T_{in}^2) + 0.793 \cdot (T_{ref_{ff}} - T_{in}) \tag{3.29}$$

Upoštevajoč zgornji enačbi je izhod krmilnika z upoštevanjem motnje (masni pretok olja) podan z naslednjim izrazom [57]:

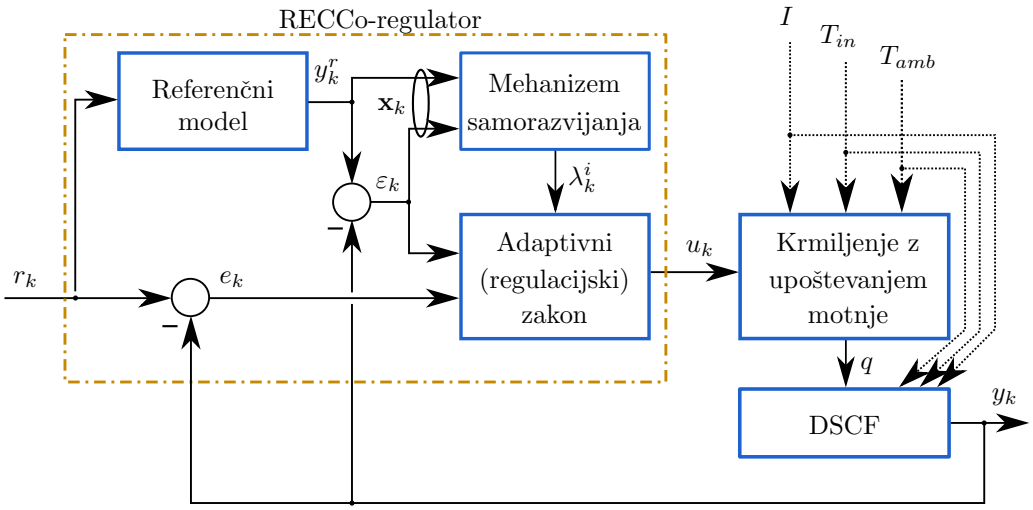
$$q = 0.697 X_1(t) - 1.19 X_2(t) + 0.05 \tag{3.30}$$

Celoten regulacijski algoritem, ki smo ga uporabili za vodenje sistema DSCF, je prikazan na sliki 3.13. Vidimo, da je zaporedno z RECCo-regulatorjem vezan še krmilnik z upoštevanjem motnje. Samorazvijajoči se RECCo-regulator izračuna referenčno temperaturo $T_{ref_{ff}}$ in potem direktni krmilnik določi ustrezen pretok olja q.

Osnovni problem pri vodenju sistema DSCF je ta, da nimamo direktnega vpliva na vir energije (sončno sevanje I) in ga ne moremo manipulirati. Poleg tega vsaka motnja v sistemu močno vpliva na razmere v samem procesu DSCF. Velik del tega vpliva odpravimo z direktnim krmilnikom, za ostalo pa poskrbi samorazvijajoči se in adaptivni RECCo-regulator. Potrebno je omeniti, da smo dodali še eno omejitev, in sicer da je regulacija aktivna samo, če je sevanje sonca večje kot $I_{limit} = 500\,\mathrm{W/m^2}$.

V tabeli 3.5 so prikazani parametri procesa DSCF in začetni parametri RECCo-regulatorja. Tudi v tem primeru smo upoštevali enačbe za izračun parametrov, ki so bile podane v podpoglavju 3.3.

Glede na to, da želimo pokriti večji del delovnega območja procesa DSCF, smo referenčni signal r_k definirali v štirih delovnih točkah (200 °C, 190 °C, 180 °C in 170 °C). V osnovi želimo v času enega dneva vzdrževati temperaturo samo v eni



Slika 3.13: Regulacijska shema samorazvijajočega se RECCo-regulatorja z vzporednim krmilnikom z upoštevanjem motnje.

Tabela 3.5: Parametri procesa DSCF in nastavitveni parametri RECCo-regulatorja.

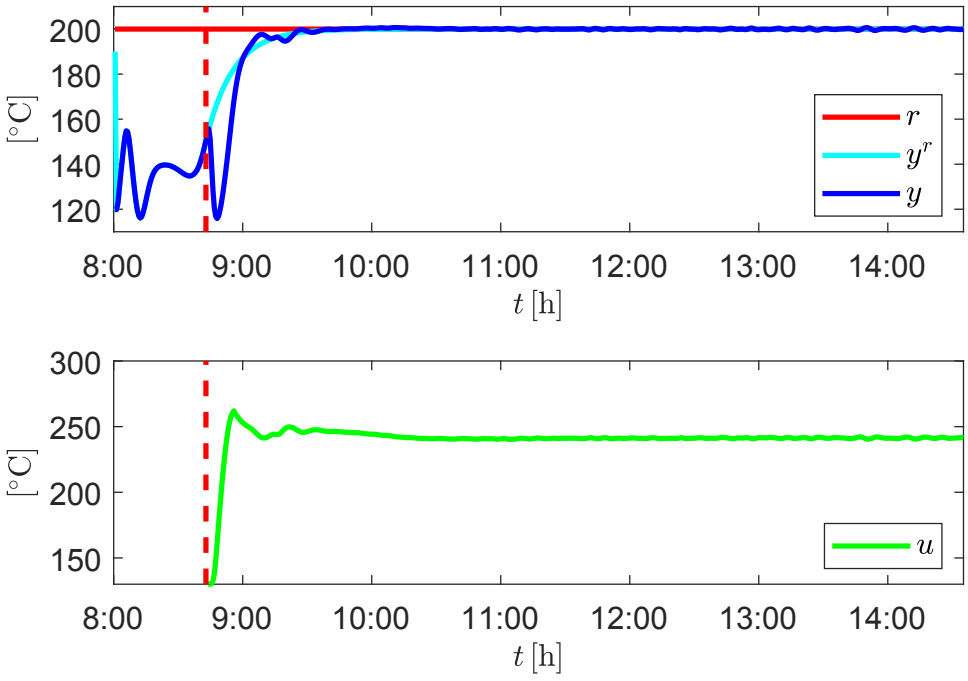
Parametri procesa DSCF					
u_{min}	u_{max}	y_{min}	y_{max}	τ	T_s
130	300	150	210	780 s	39 s

Parametri RECCo								
a_r	γ_{max}	c_{max}	n_{add}	α_{new}	d_{dead}	$\underline{\boldsymbol{\theta}}$	$\overline{\boldsymbol{\theta}}$	σ_L
0.95	0.93	20	20	1.2	0.7	0	∞	10^{-6}

delovni točki in zato smo eksperiment razdelili tako, da je vsak dan delovna točka druga. Prvi dan začnemo z $r_k = 200\,°\mathrm{C}$, naslednji dan nastavimo $r_k = 190\,°\mathrm{C}$ itn. Na sliki 3.14 so prikazani signali v začetni fazi učenja/adaptacije (referenca r_k, izhod referenčnega modela y_k^r, regulirani signal y_k, regulirni signal u_k). Poleg tega je označena tudi meja (rdeča črtkana črta), kadar je sončno sevanje večje od $500\,\mathrm{W/m^2}$. Levo od črte je $I < 500\,\mathrm{W/m^2}$ (vodenje ni aktivno) in desno od črte je $I > 500\,\mathrm{W/m^2}$ (vodenje je aktivno). Moramo poudariti, da razen parametrov, ki smo jih opisali v tabeli 3.5, nimamo nobenih dodatnih znanj o procesu, ki ga vodimo. Če opazujemo signale na sliki 3.14, in sicer tik za rdečo črtkano črto, lahko opazimo, da regulirani signal y_k najprej zaniha v nasprotno smer, kot je

izhod referenčnega signala y_k^r. Adaptacija parametrov PID-R začne z vrednostjo nič in še ni dosegla želenega učinka, vendar v zelo kratkem času ujame dinamiko referenčnega modela. Opazimo tudi, da je pogrešek v ustaljenem stanju zelo majhen (zanemarljiv).

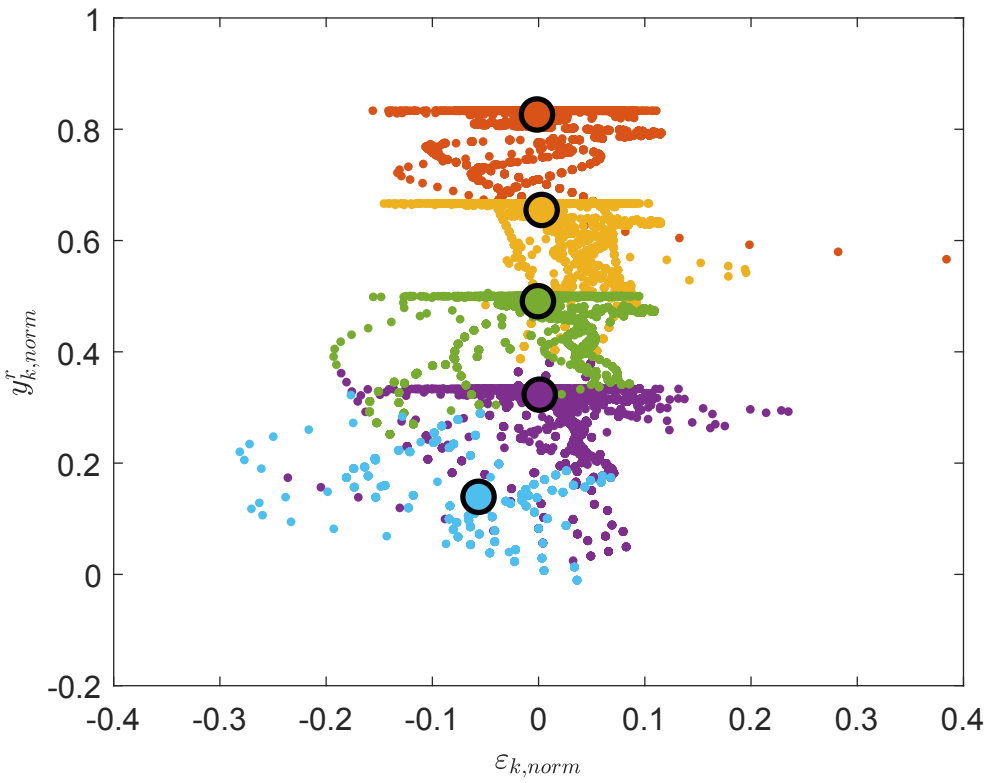
Po večkratni ponovitvi reference r_k za vsako delovno točko se mehka struktura regulatorja prilagaja in ustvari pet oblakov (slika 3.15). Na sliki 3.16 so prikazani signali v končni fazi učenja za vsako delovno točko posebej. Rdeča črtkana črta na sliki 3.16 označuje, ko sončno sevanje preseže mejno vrednost $I_{limit} = 500\,\mathrm{W/m^2}$ oziroma trenutek, ko je vodenje sistema DSCF aktivno.



Slika 3.14: Začetna faza učenja. Prikaz reference r, izhoda referenčnega modela y^r in izhoda procesa y (zgornja slika) ter regulirnega signala u (spodnja slika). Vertikalna rdeča črtkana črta označuje mejo, ko sončno sevanje preseže mejno vrednost $I_{limit} = 500\,\mathrm{W/m^2}$.

3.5. Eksperimenti na realnih napravah

V tem podpoglavju bomo prikazali rezultate vodenja realnih naprav z RECCo-regulatorjem. Najprej sledijo rezultati eksperimenta na modelni napravi toplo-



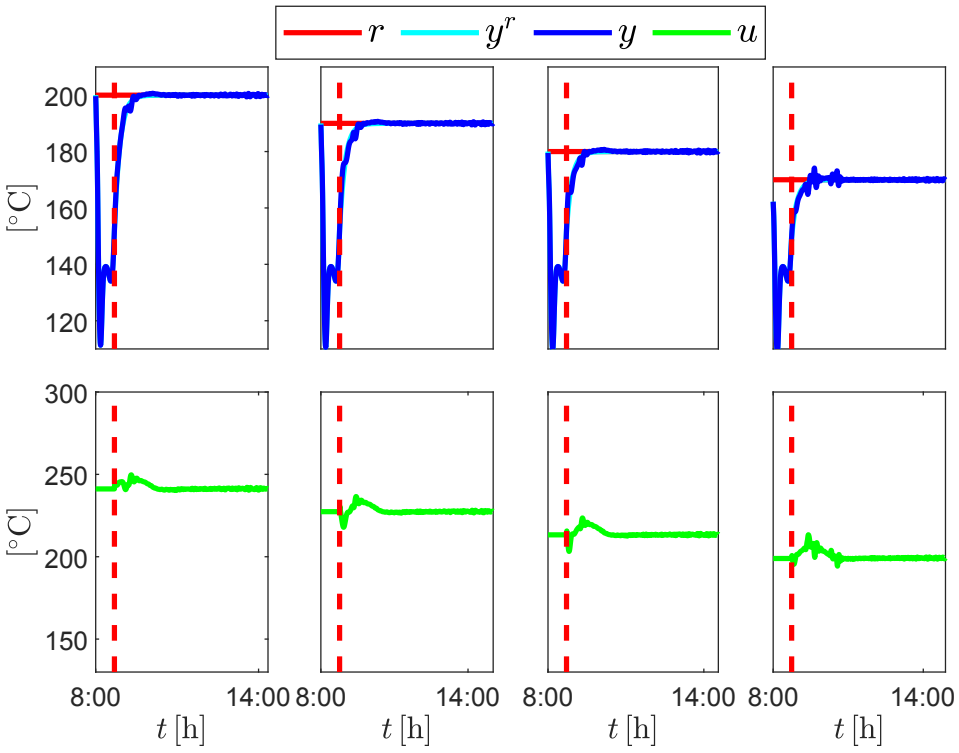
Slika 3.15: Prikaz oblakov dodanih z samorazvijajočim se RECCo-regulatorjem, kjer je ε_k sledilni pogrešek in y^r_k izhod r eferenčnega modela.

tnega izmenjevalnika, potem še sistem dveh povezanih tankov. Želimo pokazati, da lahko z enakim pristopom, kot je bil razložen v podpoglavju 3.3, vodimo tudi realne procese.

3.5.1 Vodenje toplotnega izmenjevalnika

Toplotni izmenjevalnik je naprava, ki omogoča prehod toplote iz prve tekočine na drugo tekočino (kapljevino ali plin), ne da bi se ti dve tekočini mešali ali prišli v neposredni stik. Bistveno načelo izmenjevalnika toplote je, da prenaša toploto brez prenosa tekočine, ki nosi toploto. Obstajajo različni tipi prenosnih izmenjevalnikov: ploščni (kot je v našem primeru) in v obliki školjke oziroma cevi ter razne kombinacije.

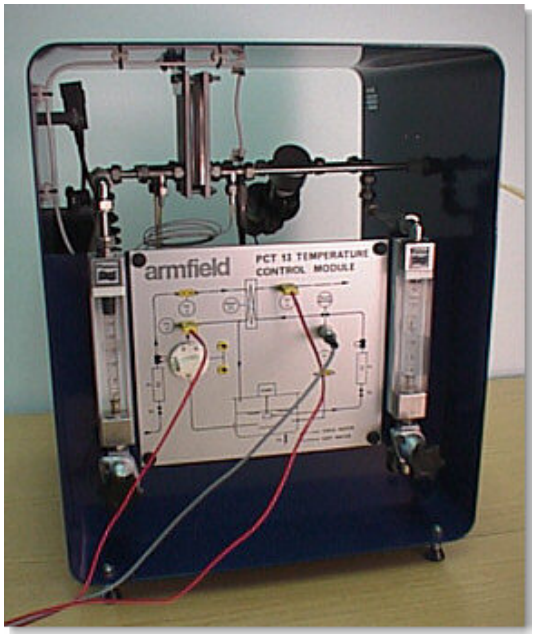
V našem primeru smo uporabili modelno napravo ploščnega toplotnega izmenjevalnika, ki je prikazan na sliki 3.17a. Pripadajoči shematski načrt naprave je



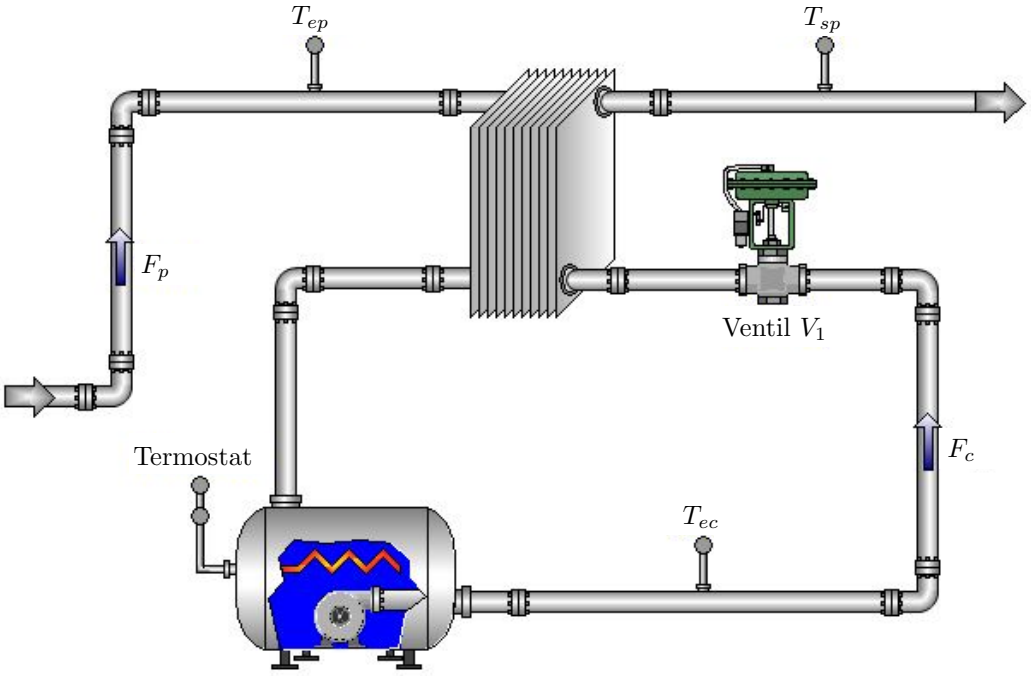
Slika 3.16: Končna faza učenja. Prikaz reference r_k, izhoda referenčnega modela y_k^r in izhoda procesa y_k (prva vrstica) ter regulirnega signala u_k (druga vrstica). Vertikalna rdeča črtkana črta označuje mejo, ko sončno sevanje preseže mejno vrednost $I_{limit} = 500\,\mathrm{W/m^2}$.

prikazan na sliki 3.17b. Proces je sestavljen iz dveh ločenih tokokrogov (topla in hladna veja). Prvi (topli) tokokrog ima v rezervoarju konstantno vhodno temperaturo medija T_{ec}, ki se uravnava s termostatom, ki ima histerezo nastavljeno na približno $\pm 2\,^{\circ}\mathrm{C}$, črpalko s konstantnimi obrati ter motorno gnan ventil V_1, ki skrbi za ustrezen pretok vode F_c. Odprtost ventila V_1 predstavlja naš regulirni signal u_k. Izhod iz izmenjevalnika se vrača nazaj v rezervoar, kjer se medij spet segreva z grelcem. Drugi (hladni) tokokrog ima neznano vhodno temperaturo medija T_{ep} s konstantnim pretokom F_p. Izhodna temperatura medija T_{sp} predstavlja regulirano veličino našega procesa.

Realno napravo smo povezali preko vmesnika za zajemanje podatkov z računalnikom. Regulacijski algoritem RECCo se je izvajal na računalniku v oko-



(a) Modelna naprava toplotnega izmenjevalnika Armfield PCT13.

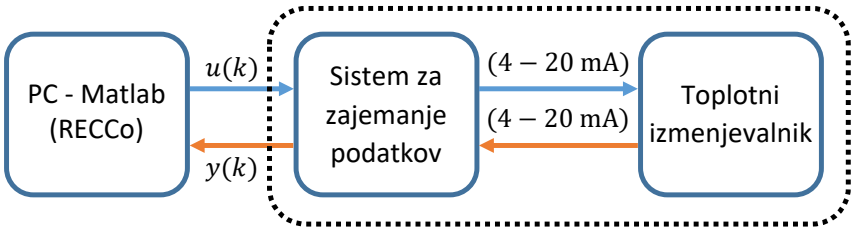


(b) Shematski prikaz procesa toplotnega izmenjevalnika.

Slika 3.17: Prikaz realne naprave toplotnega izmenjevalnika.

lju Matlab® in je dobival informacijo o trenutni vrednosti regulirane veličine y_k ter prilagajal regulirni signal u_k na podlagi izračunov. Na sliki 3.18 je prikazana

shema povezave med računalnikom (algoritmom) in realnim procesom.



Slika 3.18: Prikaz povezave med računalnikom (RECCo-regulator) z realnim sistemom toplotnega izmenjevalnika preko sistema za zajemanje podatkov.

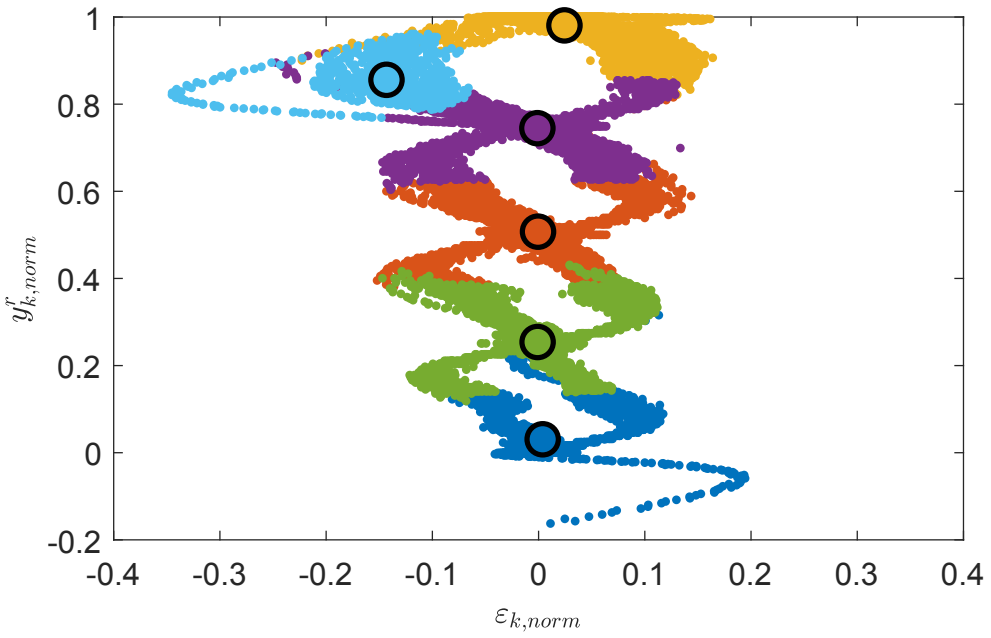
Tako kot v prejšnjih (simulacijskih) eksperimentih smo tudi v tem primeru sestavili referenčni signal r_k na celotnem področju izhodnega signala procesa. Pravzaprav smo definirali stopničaste spremembe navzgor in navzdol, ki pokrijejo različne delovne točke procesa. Glavni namen tega eksperimenta je pokazati, da se lahko algoritem RECCo sprotno nauči in razvije strukturo oblakov, ki pokrijejo delovne točke, ter sprotno prilagodi parametre vsakega posameznega regulatorja. Nastavitvene parametre algoritma RECCo smo določili na podlagi navodil, opisanih v podpoglavju 3.3 in so prikazani v tabeli 3.6. Parametri regulatorja so izračunani na podlagi parametrov realnega procesa. Fizični signali aktuatorjev so tokovno krmiljeni v območju od 4 mA do 20 mA. Tudi signali senzorjev so v enakem tokovnem območju. Za lažjo predstavo in razumevanje smo vrednost temperaturnega senzorja T_{SP} pretvorili iz tokovnega območja v območje od 25 °C do 50 °C.

Tabela 3.6: Parametri realnega procesa toplotnega izmenjevalnika in nastavitveni parametri RECCo-regulatorja.

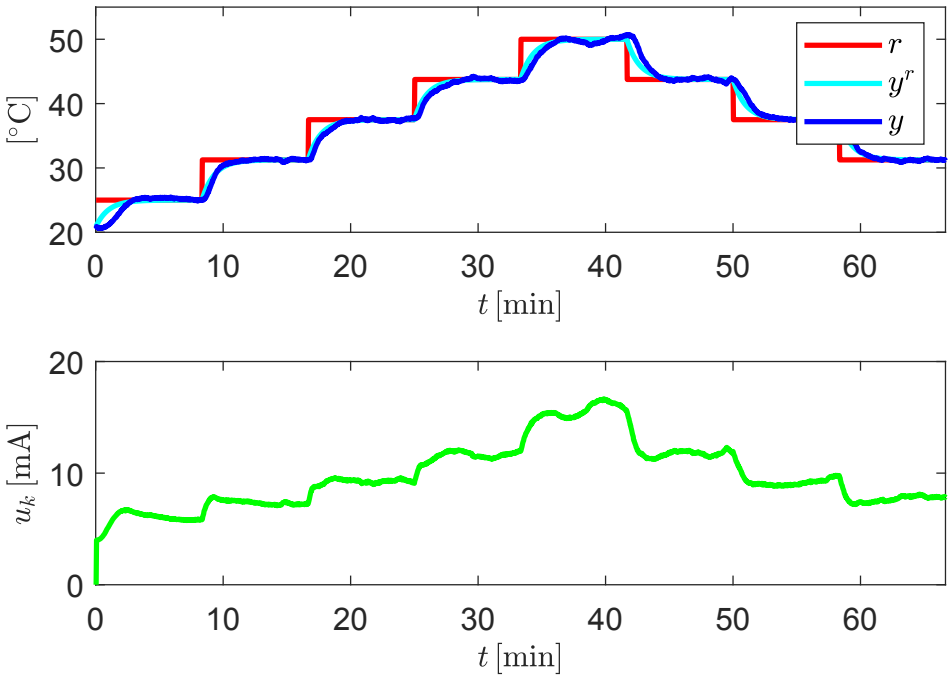
Parametri realnega procesa					
u_{min}	u_{max}	y_{min}	y_{max}	τ	T_s
4 mA	20 mA	25	50	40 s	2 s

Parametri RECCo								
a_r	γ_{max}	c_{max}	n_{add}	α_{new}	d_{dead}	$\underline{\boldsymbol{\theta}}$	$\overline{\boldsymbol{\theta}}$	σ_L
0.95	0.93	20	20	0.1	0.25 °C	0	∞	10^{-6}

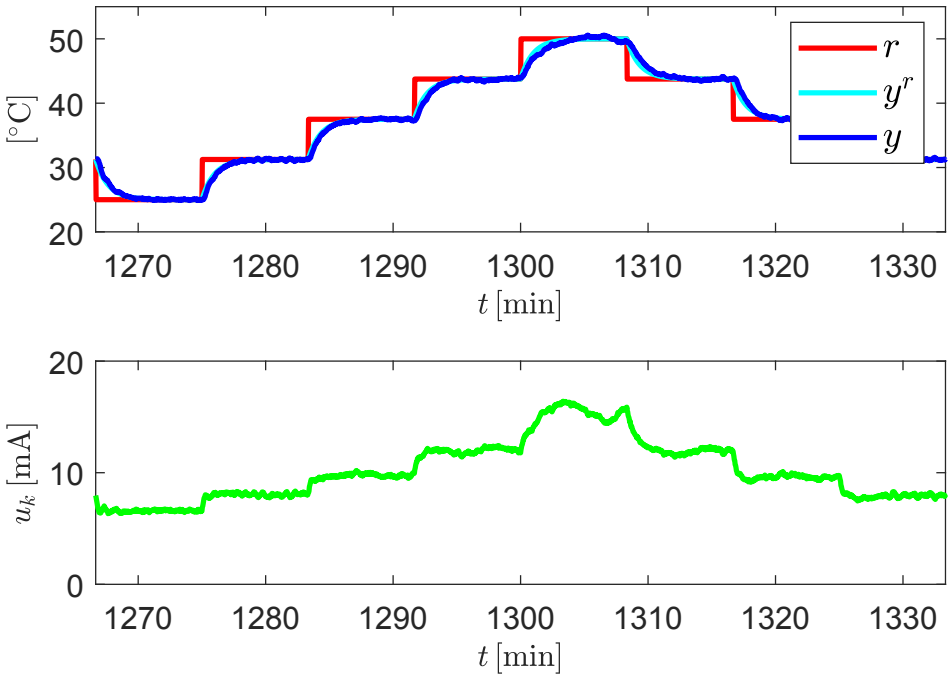
Potem, ko smo nastavili vse parametre RECCo-regulatorja in vzpostavili povezavo med računalnikom in napravo, smo zagnali eksperiment. Na začetku eksperimenta je prostor oblakov prazen (nimamo predhodno definiranih oblakov oziroma mehkih pravil), parametri regulatorja pa so enaki nič. V času samorazvijanja in adaptacije na podlagi mehanizma samorazvijanja je bilo dodanih 6 oblakov (slika 3.19). S prvim prejetim podatkom inicializiramo prvi oblak ter začnemo s postopkom sprotnega razvijanja strukture ter adaptacijo parametrov. Na sliki 3.20a je prikazana začetna faza adaptacije/učenja, kjer se vidi, da algoritem RECCo zelo hitro prilagodi strukturo in parametre prvega regulatorja tako, da izhod procesa y_k začne slediti izhodu referenčnega modela y_k^r. Največje odstopanje je v najvišji delovni točki ($r_k = 50\,°\mathrm{C}$), kjer je razlika med temperaturo tople vode T_{ec} in regulirano veličino T_{sp} najmanjša. Po nekaj ponovitvah referenčnega stopničastega signala r_k se struktura RECCo-regulatorja razvije ter prilagodi parametre, da se izhod procesa čim bolje prilega referenci. Na sliki 3.20b je prikazan rezultat vodenja po dvajsetih ponovitvah reference, kjer se vidi precej izboljšano sledenje v primerjavi s sliko 3.20a.



Slika 3.19: Prikaz oblakov dodanih z samorazvijajočim se RECCo-regulatorjem, kjer je ε_k sledilni pogrešek in y_k^r izhod referenčnega modela.



(a) Začetna faza eksperimenta.



(b) Končna faza eksperimenta.

Slika 3.20: Prikaz rezultatov vodenja toplotnega izmenjevalnika. Prikaz reference r (rdeča), izhoda referenčnega modela y^r (svetlo modra) in izhoda procesa y (modra) ter regulirnega signala u (zelena).

3.5.2 Vodenje sistema dveh povezanih tankov

V tem razdelku bomo preskusili delovanje RECCo-regulatorja na realnem sistemu dveh povezanih tankov. V tem razdelku uporabljen proces predstavlja pilotna industrijska naprava, razvita s strani podjetja DeLorenzo [80], ki se fizično nahaja v Laboratoriju za avtomatiko pri federalnem inštitutu v zvezni državi Rio Grande de Norte v Braziliji. Ta sistem je že bil uporabljen v različnih raziskavah na področju vodenja, zaznavanja napak, identifikacije itn. Sestavljen je iz več industrijskih senzorjev, aktuatorjev in voden z industrijskim programirljivim logičnim krmilnikom PLK. Nekaj avtorjev je že predstavilo različne tehnike vodenja. Samorazvijajoči se regulator, ki temelji na mehkem sistemu AnYa, je bil predstavljen v [78]. Razlika z algoritmom RECCo je v načinu samorazvijanja strukture ter v adaptivnem zakonu. Pred tem so avtorji v [76] predstavili večstopenjski hierarhični mehki regulator, ki zahteva minimalno nastavitev začetnih parametrov. Cilj tega regulatorja je zajeti in obvladovati nelinearno naravo procesa. Sledilo je nekaj raziskav na področju zaznavanja napak in diagnosticiranja. V [74] je bila predlagana rešitev za sprotno zaznavanje napak in anomalij v procesu, ki temelji na konceptu rekurzivnega ocenjevanja gostote (ang. *recursive density estimation*, RDE). Ta koncept so avtorji v [75, 199] nadgradili in so predlagali sistem za klasifikacijo oziroma diagnostiko napak. Področje zaznavanja napak je bilo del raziskave v [38], kjer je bil predlagan koncept avtomatskega zaznavanja anomalij v sistemu na podlagi tipičnosti in ekscentričnosti podatkov (ang. *data typicality and eccentricity*). Nazadnje je bila predstavljena metoda samorazvijanja modela za klasifikacijo podatkov [77] in je izkazala zelo obetavne rezultate.

Kot je razvidno iz zgoraj omenjenih raziskav, omogoča realni proces dveh povezanih tankov zelo širok nabor eksperimentov na precej različnih področjih. Na sliki 3.21a je prikazana slika realnega procesa, medtem ko je na sliki 3.21b prikazana pripadajoča shema. Komunikacija med krmilnikom PLK oziroma oddaljenim klientom in samim procesom je izvedena z vmesnikom OPC (ang. *open platform communications*) [112].

Proces dveh povezanih tankov vsebuje naslednje elemente: senzorje za temperaturo, tlak, pretok in nivo; pretvornike, ki pretvorijo fizične signale v električne za procesiranje na krmilniku PLK; programska oprema SCADA (ang. *supervisory control and data acquisition*) za konfiguracijo in nadzor procesa; dva ventila V_1

in V_2; črpalko s frekvenčnim pretvornikom; grelnik in toplotni izmenjevalnik.

Na sliki 3.21b je prikazana shema procesa, kjer je razvidna povezava med obema tankoma. Pretok vode iz tanka T_2 v tank T_1 je omogočen s črpalko, pretok iz T_1 v T_2 pa je odvisen od gravitacije preko ventila V_1. Oba ventila sta uporabljena za regulacijo pretoka v posameznem delu sistema. V našem primeru je cilj eksperimenta regulirati nivo vode v tanku T_1 ($y_k = L_{T1}$) z manipulacijo hitrosti vrtenja črpalke ($u_k = V_{pump}$) preko frekvenčnega pretvornika. Oba ventila, V_1 in V_2, sta odprta na 100 %. Poleg tega smo testirali tudi občutljivost regulatorja na motnje (zapirali smo ventila V_1 in V_2). Rezultati eksperimentov bodo predstavljeni v nadaljevanju.

Tako kot v prejšnjih primerih smo nastavitvene parametre RECCo-regulatorja (podpoglavje 3.3) ocenili na podlagi parametrov procesa. Parametri procesa in regulatorja so prikazani v tabeli 3.7.

Tabela 3.7: Parametri realnega sistema dveh povezanih tankov in nastavitveni parametri RECCo-regulatorja.

Parametri realnega procesa					
u_{min}	u_{max}	y_{min}	y_{max}	τ	T_s
0 %	100 %	0 %	100 %	40 s	1 s

Parametri RECCo								
a_r	γ_{max}	c_{max}	n_{add}	α_{new}	d_{dead}	$\underline{\boldsymbol{\theta}}$	$\overline{\boldsymbol{\theta}}$	σ_L
0.95	0.93	20	20	0.5	0.5 %	0	∞	10^{-6}

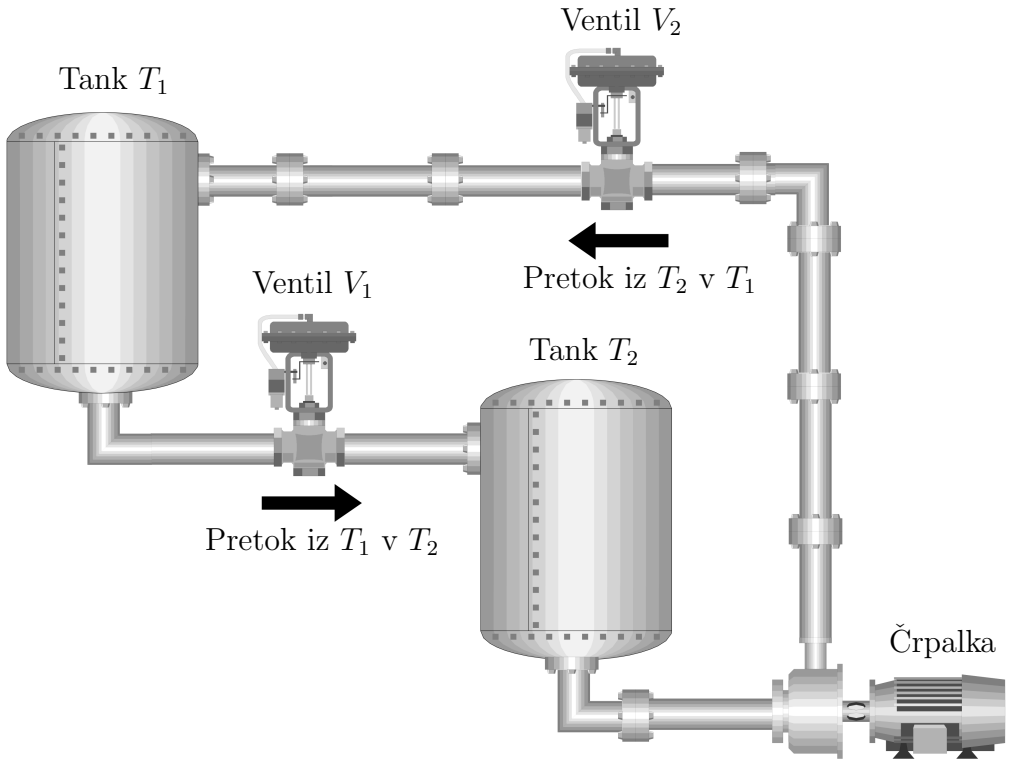
Vodenje sistema dveh povezanih tankov je bilo realizirano z algoritmom RECCo tako, kot je prikazano na sliki 3.1. Mehanizem samorazvijanja poskrbi za razdelitev prostora na lokalno linearna področja in adaptivni zakon sprotno prilagodi parametre lokalnega regulatorja za optimalno delovanje v določenem področju.

Na začetku eksperimenta (slika 3.22a) regulator precej dobro izniči pogrešek v ustaljenem stanju z ocenjenim časom izravnave približno 2 min in z maksimalnim prevzponom približno 20 %. Z nadaljnjo adaptacijo parametrov in z dodajanjem novih oblakov se struktura in parametri regulatorja prilagodijo vodenemu procesu

in na ta način se izboljša vodenje. V končni fazi eksperimenta (slika 3.22b) je maksimalni prevzpon skoraj nič in čas izravnave je manj kot 1 min. Širina ene stopnice je 4 min.

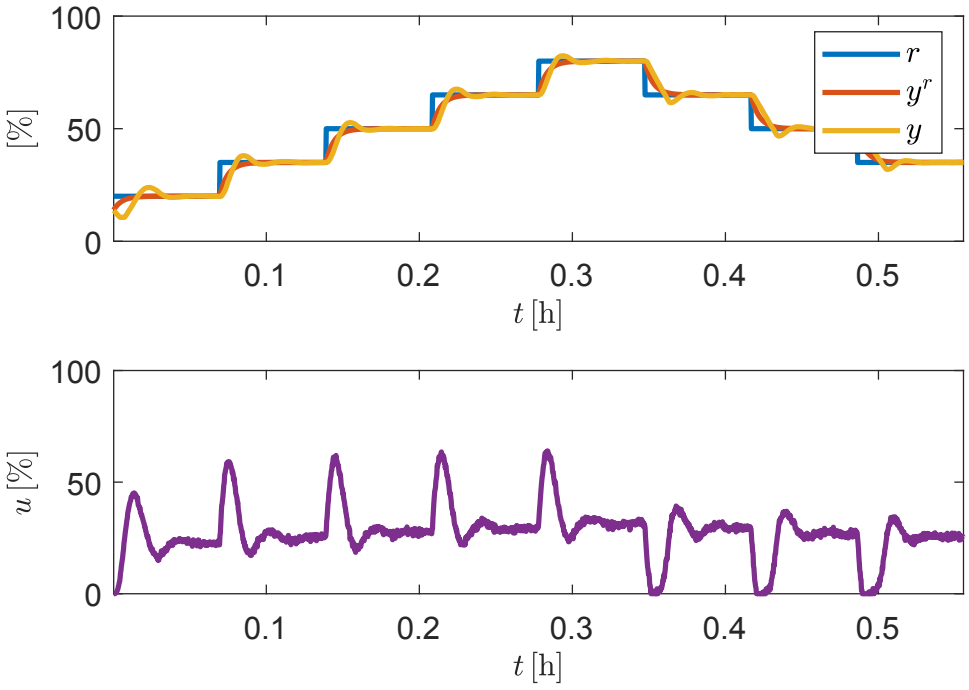
Na sliki 3.23 so prikazani dodani oblaki s pripadajočimi parametri regulatorjev. V tem eksperimentu so bili dodani 4 oblaki (slika 3.23a). Adaptacija in konvergenca parametrov vseh štirih regulatorjev sta prikazani na sliki 3.23b. Da bi pokazali učinek učenja, smo z enačbo (3.24) izračunali vsoto absolutnih vrednosti SAE_κ sledilnega pogreška ε_k za vsak cikel κ posebej (slika 3.24). Iz slike je razvidno, da vrednost SAE_κ na začetku strmo upada, kar je rezultat dodajanja novih oblakov in sprotne adaptacije parametrov. Kot smo že omenili zgoraj, smo na procesu dveh povezanih tankov testirali tudi odzivnost regulacijskega algoritma na zunanje motnje. S tem namenom smo v štirih časovnih trenutkih za nekaj časa zaprli dovodni ventil V_2 iz 100 % na 60 % (zgornja vrstica na sliki 3.25). Prvi dve motnji sta bili na začetku, drugi dve pa proti koncu eksperimenta. Kot pričakovano, se je regulirni signal u_k (kar predstavlja hitrost vrtenja črpalke) ustrezno povečal (srednja vrstica na sliki 3.25), da bi se s povečanim tlakom, ki ga ustvari črpalka, povečal tudi pretok vode iz tanka T_2 v vank T_1. Iz spodnje vrstice na sliki 3.25 lahko opazimo, da motnja zelo malo vplivala na vrednost reguliranega signala.

(a) Slika realnega procesa dveh povezanih tankov.

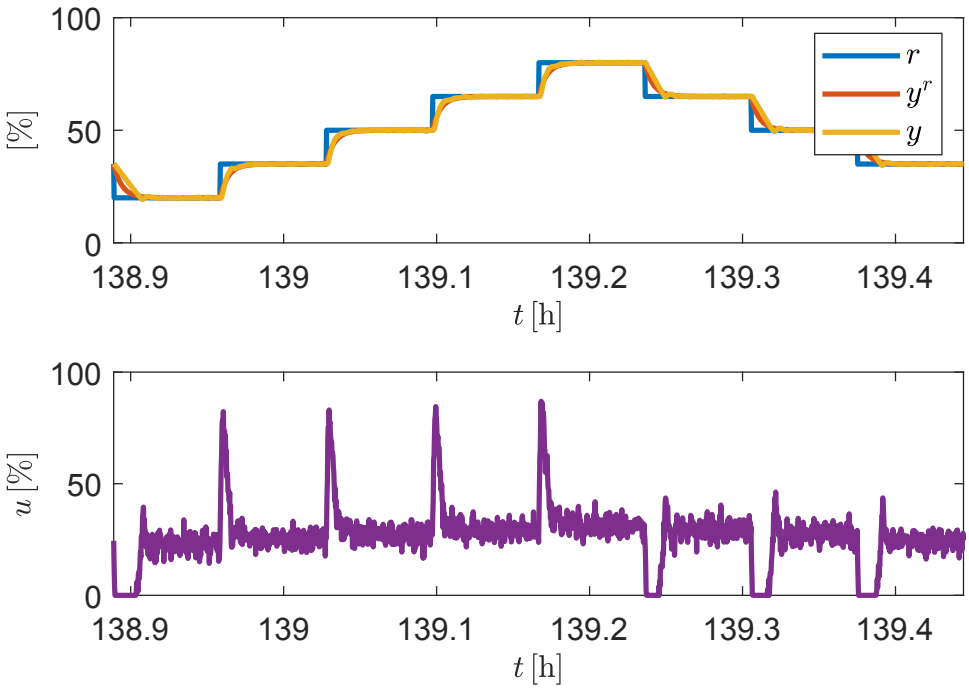


(b) Shematski prikaz procesa dveh povezanih tankov.

Slika 3.21: Prikaz realne naprave dveh povezanih tankov [78].

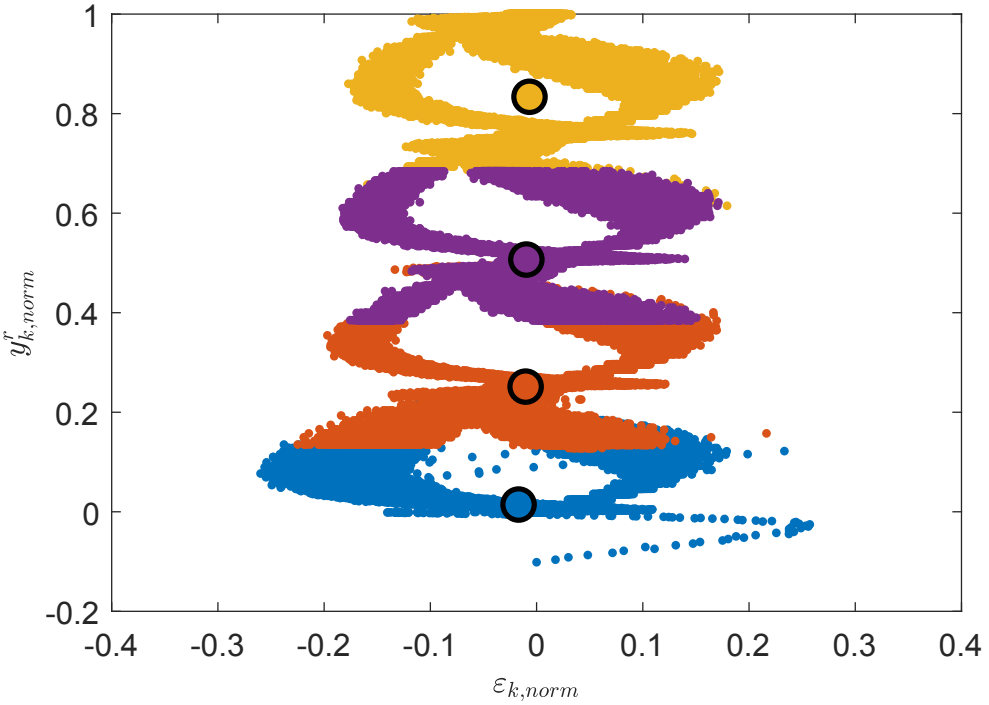


(a) Začetna faza eksperimenta.

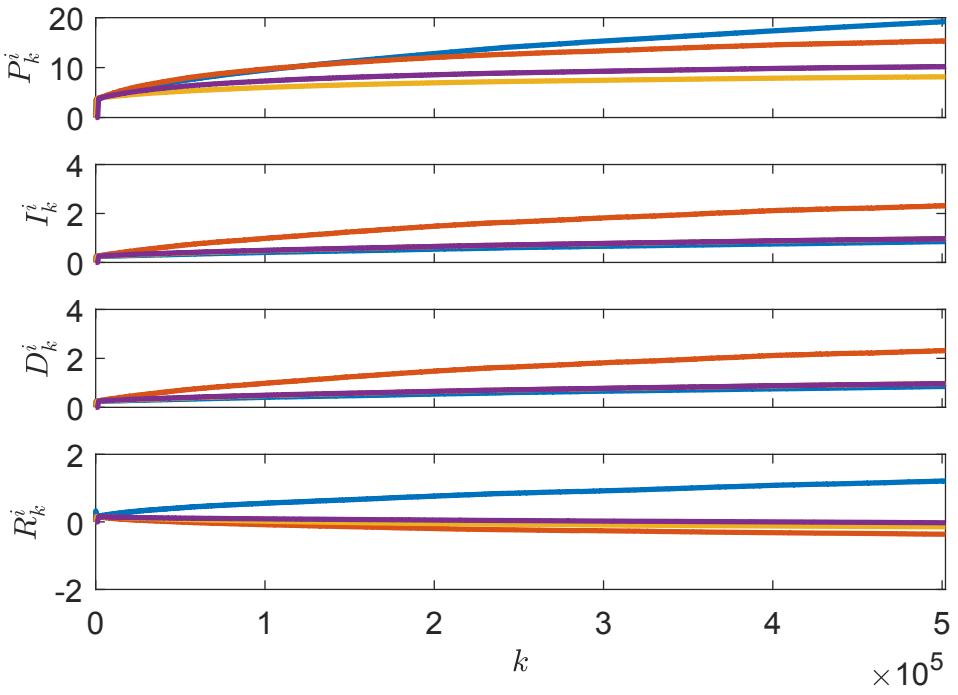


(b) Končna faza eksperimenta.

Slika 3.22: Prikaz rezultatov vodenja procesa dveh povezanih tankov. Prikaz reference r_k (modra), izhoda referenčnega modela y_k^r (rdeča) in izhoda procesa y_k (oranžna) ter regulirnega signala u_k (vijolična).

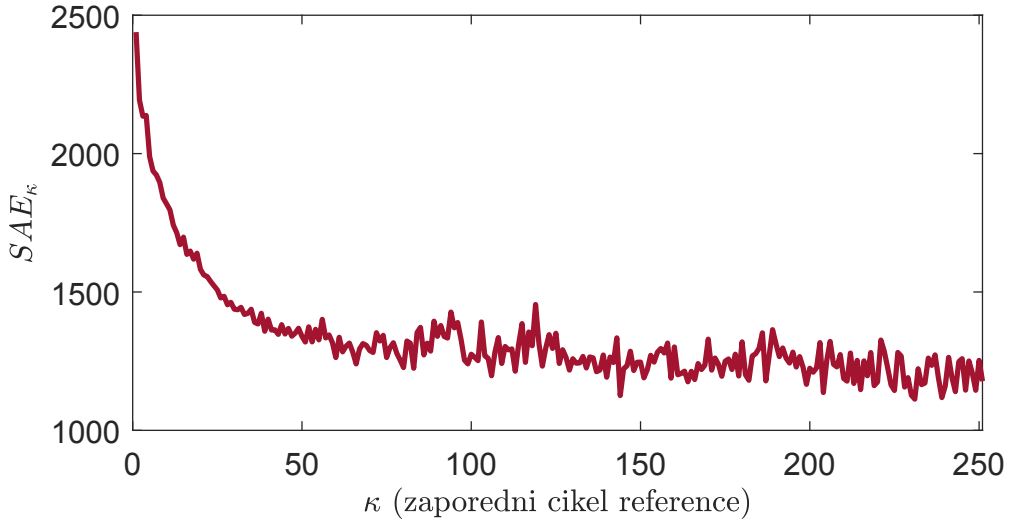


(a) Prikaz oblakov dodanih z samorazvijajočim se RECCo-regulatorjem, kjer je ε_k sledilni pogrešek in y^r_k izhod referenčnega modela.

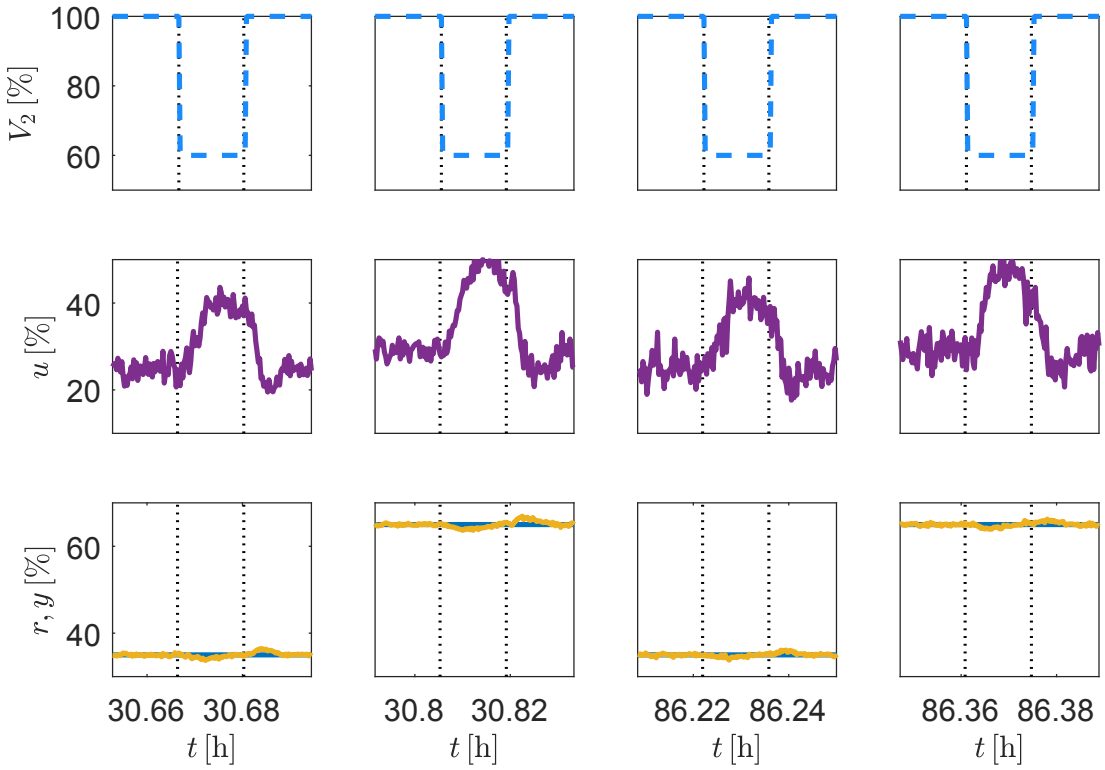


(b) Prikaz adaptacije parametrov regulatorja za vsak oblak podatkov.

Slika 3.23: Prikaz rezultatov (mehanizma samorazvijanja in adaptivnega zakona) vodenja procesa dveh povezanih tankov.



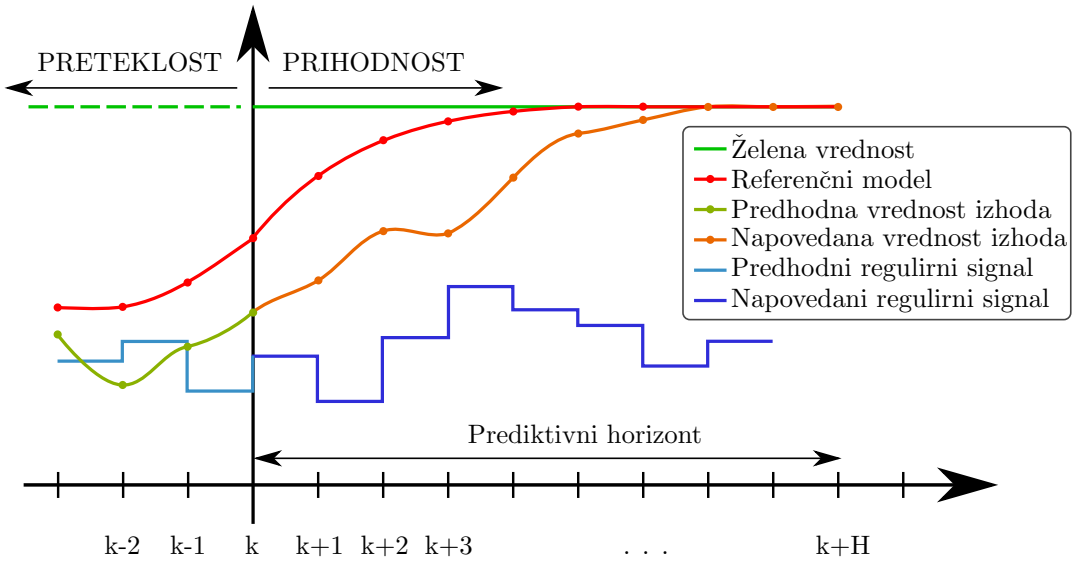
Slika 3.24: Prikaz vsote absolutne vrednosti sledilnega pogreška ε_k za vsak cikel



Slika 3.25: Prikaz vpliva motenj na delovanje RECCo-regulatorja. Zgornja vrstica prikazuje motnjo – zapiranje ventila V_2 ob štirih različnih časovnih trenutkih. Druga vrstica prikazuje spremembo regulirnega signala u, spodnja vrstica pa prikazuje odziv reguliranega signala y.

3.6. Mehki prediktivno funkcijski regulator na podlagi oblakov

Mehki prediktivni funkcijski regulator (ang. *fuzzy predictive functional controller*, FPFC) uporablja mehki model reguliranega procesa za predikcijo izhodnega signala na določenem horizontu v prihodnosti. Regulator FPFC lahko uvrstimo v skupino prediktivnih regulatorjev na podlagi modela (ang. *model predictive control*, MPC). Osnovni princip vodenja z MPC regulatorjem je predstavljen na sliki 3.26. Na podlagi znanja iz preteklosti se zgradi dinamičen model procesa. Z uporabo tega modela se izvede predikcija reguliranega signala (oranžna črta) in določi se takšen regulirni signal (modra črta), da minimiziramo razliko med izhodnim (oranžna črta) in referenčnim (rdeča črta) signalom v vnaprej določenem prediktivnem horizontu $k + H$. Prediktivno funkcijski regulator (ang. *predictive*



Slika 3.26: Prikaz načela prediktivnega vodenja.

functional controller, PFC) je bil predstavljen v [209] in je v osnovi namenjen vodenju linearnih procesov. Za regulacijo nelinearnih procesov pa je bila razvita rešitev na podlagi mehkih modelov, ki poskrbijo za razdelitev prostora procesa na linearna področja [88, 134, 159, 232].

V tem podpoglavju bomo predstavili mehki prediktivno funkcijski regulator na podlagi oblakov (ang. *fuzzy cloud-based predictive functional control*, FCPFC). Metoda za identifikacijo modela procesa kombinira teorijo mehkih oblakov [24] (mehki sistem AnYa) in mehanizem samorazvijanja (glej razdelek 3.2.2) za deli-

tev nelinearnega prostora na linearna področja ter rekurzivno metodo uteženih najmanjših kvadratov za izračun parametrov lokalno linearnih modelov. V nadaljevanju bomo najprej predstavili postopek identifikacije modela procesa, ki ga želimo regulirati, nato pa prediktivno funkcijski regulator za vodenje na osnovi modela.

3.6.1 Identifikacija mehkega modela na podlagi oblakov

V tem razdelku bomo na kratko razložili načelo identifikacije mehkega modela procesa na podlagi oblakov. Bolj podrobno o identifikaciji bo razloženo v poglavju 4. Mehki model, ki ga želimo identificirati, ima naslednjo obliko:

$$\mathcal{R}^i: \quad \textbf{IF } (\mathbf{x} \sim X^i) \textbf{ THEN } (y^i = f^i(\mathbf{x})) \tag{3.31}$$

To enačbo smo že srečali v poglavju 2 in nam je že dobro znana. Vhodni vektor je definiran z zakasnjenimi vhodi in izhodi procesa $\mathbf{x}_k = [y_{k-1},\ \ldots,\ y_{k-n_a},\ u_{k-1},\ \ldots,\ u_{k-n_b}]$, kjer sta n_a in n_b števili zakasnjenih izhodov in vhodov. V pogojnem (**IF**) delu je definirana pripadnost trenutnega podatka obstoječim oblakom, kar definira skupni samorazvijajoči se model. Posledični (**THEN**) del modela je definiran z NARX (ang. *nonlinear autoregressive exogenous model*) modelom. Delni lokalni model je definiran kot:

$$f_k^i = {\boldsymbol{\theta}_k^i}^T \boldsymbol{\psi}_k \tag{3.32}$$

kjer je $\boldsymbol{\psi}_k = [\mathbf{x}_k,\ 1]^T$ regresijski vektor in $\boldsymbol{\theta}_k^i$ vektor neznanih parametrov lokalnega modela $\boldsymbol{\theta}_k^i = \left[a_1^i, \ldots, a_{n_a}^i, b_1^i, \ldots, b_{n_b}^i, r^i\right]^T$. Ocenjeni izhod modela v trenutku k se izračuna kot uteženo povprečje vseh lokalnih modelov:

$$\hat{y}_k = \sum_{j=1}^{c} \beta_k^j {\boldsymbol{\theta}_k^j}^T \boldsymbol{\psi}_k = \boldsymbol{\beta}_k^T \boldsymbol{\Theta}_k^T \boldsymbol{\psi}_k \tag{3.33}$$

kjer je β_k^j pripadnost vhodnega podatka $\mathbf{x}_k$ k j-temu oblaku. Desni del enačbe (3.33) je zapis v matrični obliki z naslednjimi vektorji/matrikami:

$$\begin{aligned} \boldsymbol{\beta}_k &= \left[\beta_k^1, \beta_k^2, \ldots, \beta_k^c\right]^T \\ \boldsymbol{\Theta}_k &= \left[\boldsymbol{\theta}_k^1, \boldsymbol{\theta}_k^2, \ldots, \boldsymbol{\theta}_k^c\right] \\ \boldsymbol{\psi}_k &= [\mathbf{x}_k,\ 1]^T \end{aligned} \tag{3.34}$$

Razen regresijskega vektorja $\boldsymbol{\psi}$ je potebno elemente vektorja $\boldsymbol{\beta}$ in matrike $\boldsymbol{\Theta}$ oceniti oziroma identificirati. Vrednosti elementov pripadnostne funkcije $\boldsymbol{\beta}$ se določijo z izračunom lokalne gostote (2.11) in tako dobimo:

$$\beta_k^i = \frac{(\gamma_k^i)^\eta}{\sum_{j=1}^{c}(\gamma_k^j)^\eta}, \qquad i = 1, \ldots, c \tag{3.35}$$

kjer parameter η definira mehkost preklapljanja (faktor mehkosti) med mehkimi pravili [269].

Nenazadnje, kot smo že omenili zgoraj, bomo elemente matrike $\boldsymbol{\Theta}$ identificirali s uporabo rekurzivne metode uteženih najmanjših kvadratov (ang. *recursive Weighted Least Squares*, rWLS) [187]:

$$\begin{aligned}
\boldsymbol{\psi}_k &= [\mathbf{x}_k \quad 1]^T \\
\boldsymbol{P}_k^i &= \frac{1}{\lambda_r}\left(\boldsymbol{P}_{k-1}^i - \frac{\beta_k^i \boldsymbol{P}_{k-1}^i \boldsymbol{\psi}_k \boldsymbol{\psi}_k^T \boldsymbol{P}_{k-1}^i}{\lambda_r + \beta_k^i \boldsymbol{\psi}_k^T \boldsymbol{P}_{k-1}^i \boldsymbol{\psi}_k}\right) \\
\boldsymbol{\theta}_k^i &= \boldsymbol{\theta}_{k-1}^i + \boldsymbol{P}_{k-1}^i \boldsymbol{\psi}_k \beta_k^i \left(y_k - \boldsymbol{\psi}_k^T \boldsymbol{\theta}_{k-1}^i\right), \qquad i = 1, \ldots, c
\end{aligned} \tag{3.36}$$

kjer matrika $\boldsymbol{P}_k^i$ predstavlja ocenjeno vrednost inverzne Hessejeve matrike. Začetna nastavljena vrednost matrike $\boldsymbol{P}_0^i$ zelo vpliva na vrednost vektorja parametrov $\boldsymbol{\theta}_0$, ki je lahko daleč od optimalne vrednosti $\boldsymbol{\theta}_{opt}$. Začetna vrednost $\boldsymbol{P}_0^i$ je ponavadi diagonalna matrika $\boldsymbol{P}_0^i = \alpha \boldsymbol{I}$, kjer je $\alpha >> 1$ (običajno med 100 in 1000) [187]. Ko imamo opravka z nelinearnimi in časovno spremenljivimi procesi, želimo preprečiti, da bi se matrika $\boldsymbol{P}^i$ zelo zmanjšala. Majhna vrednost matrike $\boldsymbol{P}^i$ pomeni, da se adaptacija parametrov $\boldsymbol{\theta}^i$ upočasni [187]. To lahko preprečimo z uvedbo faktorja pozabljanja λ_r. Vrednost faktorja λ_r je ponavadi nastavljena med 0.9 in 1, kjer $\lambda_r = 1$ pomeni, da ni pozabljanja. Pravzaprav nastavitev parametra λ_r pomeni kompromis med neobčutljivostjo na motnje (velik λ_r) in hitrim prilagajanjem spremembam v procesu (majhen λ_r).

3.6.2 Prediktivno funkcijski regulator

V tem razdelku bomo razložili delovanje prediktivno funkcijskega regulatorja (PFC), ki uporablja ocenjeno izhodno vrednost modela (3.33) za napovedovanje v prihodnosti. PFC je zasnovan tako, da minimizira kriterijsko funkcijo [232]:

$$J(u,k) = \sum_{j=N_1}^{N2} (y_{m,k+j} - y_{r,k+j})^2 + \lambda \sum_{j=1}^{N_u} u_{k+j}^2 \tag{3.37}$$

kjer je y_m izhod modela procesa, y^r referenčna trajektorija in u regulirni signal; indeks $k+j$ pomeni, da opazujemo j-koračno predikcijo teh signalov; N_1 in N_2 definirata horizont izhoda in N_u horizont regulirnega signala; λ je utež, ki določi relativen vpliv med regulirano in regulirno veličino.

PFC algoritem predpostavlja, da dinamiko procesa lahko lokalno aproksimiramo z modelom prvega reda. Zato predpišemo želeno (referenčno) trajektorijo odziva procesa z naslednjim zapisom:

$$y^r_{k+1} = a_r y^r_k + (1 - a_r) r_k, \qquad 0 < a_r < 1 \tag{3.38}$$

kjer parameter a_r definira dinamiko referenčnega modela (tako kot v enačbi (3.3)) in r_k je referenčni signal.

Prav tako lahko za model procesa izberemo dinamiko prvega reda, kar pomeni, da regresijski vektor $\mathbf{x}_k$ določimo z $n_a = 1$ in $n_b = 1$. Parametre tega modela bomo določili z postopkom, ki je bil razložen v prejšnjem razdelku. Tako lahko i-ti lokalni model (3.32) zapišemo v naslednji obliki:

$$y^i_{m,k+1} = a^i_m y_{m,k} + b^i_m u_k + r^i_m, \qquad i = 1, \ldots, c \tag{3.39}$$

kjer so a^i_m, b^i_m in r^i_m neznani parametri vektorja $\boldsymbol{\theta}^i_k$ in jih bomo določili z uporabo najmanjših kvadratov iz prejšnjega razdelka. Parametre vseh lokalnih modelov ($i = 1, \ldots, c$) lahko strnjeno zapišemo v vektorski obliki:

$$\begin{aligned} \boldsymbol{a}_m &= \left[a^1_m, a^2_m, \ldots, a^c_m\right]^T \\ \boldsymbol{b}_m &= \left[b^1_m, b^2_m, \ldots, b^c_m\right]^T \\ \boldsymbol{r}_m &= \left[r^1_m, r^2_m, \ldots, r^c_m\right]^T \end{aligned} \tag{3.40}$$

in potem lahko izračunamo še globalne linearne parametre modela:

$$\begin{aligned} \hat{a}_m &= \boldsymbol{\beta}^T \boldsymbol{a}_m = \sum_{j=1}^{c} \beta^j a^j_m \\ \hat{b}_m &= \boldsymbol{\beta}^T \boldsymbol{b}_m = \sum_{j=1}^{c} \beta^j b^j_m \\ \hat{r}_m &= \boldsymbol{\beta}^T \boldsymbol{r}_m = \sum_{j=1}^{c} \beta^j r^j_m \end{aligned} \tag{3.41}$$

in dobimo skupen izhod modela reguliranega procesa:

$$y_{m,k+1} = \hat{a}_m y_{m,k} + \hat{b}_m u_k + \hat{r}_m \tag{3.42}$$

Upoštevajoč enačbe (3.38), (3.42) in kriterijsko funkcijo (3.37) ($N_1 = N_2 = H$ in utežni faktor $\lambda = 0$) v [232] je analitično izpeljan regulirni signal:

$$u_k = \frac{(1 - a_r^H)(r_k - y_{p,k})}{\frac{\hat{b}_m}{1 - \hat{a}_m}(1 - \hat{a}_m^H)} + \frac{y_{m,k}}{\frac{\hat{b}_m}{1 - \hat{a}_m}} - \frac{\hat{r}_m}{\hat{b}_m} \tag{3.43}$$

kjer H predstavlja predikcijski horizont in je določen v intervalu

$$N \leq H \leq \frac{\tau}{2T_s} \tag{3.44}$$

kjer je N red procesa, τ časovna konstanta in T_s čas vzorčenja.

3.6.3 Vodenje reaktorja CSTR

V tem razdelku bomo predstavili proces reaktorja z neprekinjenim mešanjem[2] (ang. *continuous stirred tank reactor*, CSTR) in primerjali rezultate vodenja med FCPFC in RECCo-regulatorjem. Glavna karakteristika procesa CSTR je, da ima izrazito nelinearno obnašanje in tako tudi veliko negotovost parametrov [249]. Kemični procesi se različno obnašajo v različnih delovnih točkah, nekateri od njih lahko celo preidejo v oscilatorno delovanje, kar še toliko bolj oteži načrtovanje regulacijskih algoritmov.

V literaturi lahko najdemo različne pristope za vodenje procesa CSTR, od klasičnih in enostavnih PID-regulatorjev do bolj naprednih, podatkovno gnanih, prediktivnih in adaptivnih metod vodenja. Optimalno nastavljen PID-regulator za vodenje procesa CSTR je predstavljen v [83,175], medtem ko so avtorji v [150] predstavili robusten PID-regulator na podlagi Kharitonovega teorema. Tako kot smo že predhodno omenili, obstajajo tudi bolj napredne metode, ki upoštevajo nelinearnost procesa in poskušajo na različne načine obravnavati ta problem. Na primer, avtorji v [33, 127] so uporabili metodo z razvrščanjem ojačenj (ang. *gain schaduling*), medtem ko so v [8,102,145,218] predstavljene nekatere različice mehkih regulatorjev. Kombinacija mehkih modelov in nevronskih mrež (nevromehki regulator) je predstavljena v [226,262]. Tudi modelno prediktivno vodenje je bilo že obravnavano za vodenje procesa CSTR [44, 203, 231, 260], kar je tudi iztočnica za pričujoče poglavje. Avtorji v [62, 126] so podali tudi komparativne analize vodenja procesa CSTR med različnimi naprednimi metodami.

[2] V nadaljevanju bomo uporabljali samo izraz reaktor.

Reaktor CSTR predstavlja pomembno enoto v raznih kemičnih procesih. V našem primeru je proces definiran z naslednjimi enačbami [221]:

$$\begin{aligned}\frac{dC_A}{dt} &= \frac{q}{V}(C_{A0} - C_A) - k_0 \exp\left(-\frac{E}{RT}\right) C_A \\ \frac{dT}{dt} &= \frac{q}{V}(T_f - T) - k_0 \frac{\Delta H}{\rho C_p} \exp\left(-\frac{E}{RT}\right) C_A \\ &\quad + \frac{UA}{V\rho C_p}(T_c - T)\end{aligned} \tag{3.45}$$

kjer so vse spremenljivke in konstante definirane v tabeli 3.8 [222]. Cilj vodenja je nadzorovati temperaturo reaktorja ($y = T$) s temperaturo vstopne hladilne tekočine ($u = T_c$). Na sliki 3.27 je prikazan odprtozančni odziv procesa, iz katerega je razvidna nelinearna dinamika procesa. Področje signala $T_c = 300 \div 310\,\mathrm{K}$ predstavlja največji izziv pri vodenju.

Tabela 3.8: Nominalne vrednosti parametrov procesa CSTR [222].

Veličina	**Oznaka**	**Vrednost**	**Enota**
Pretok	q	100	[l/min]
Koncentracija hranila	C_{A0}	1	[mol/l]
Temperatura hranila	T_0	350	[K]
Volumen reaktorja	V	100	[l]
Gostota tekočine	ρ	1×10^3	[g/l]
Specifična toplota	C_p	0.239	[J/g K]
Entalpija reakcije	ΔH	-5×10^4	[J/mol]
Energija aktivacije	E/R	8750	[K]
Konstanta hitrosti reakcije	k_0	7.2×10^{10}	[min^{-1}]
Prenos toplote	UA	5×10^4	[J/min K]
Temperatura vstopne hladilne tekočine	T_c	300	[K]
Merjena koncentracija izdelka	C_A	0.5	[mol/l]
Temperatura reaktorja	T	350	[K]

V nadaljevanju sledi primerjava med rezultati vodenja s regulatorjem FCPFC in algoritmom RECCo. S tem namenom bomo sorodne nastavitvene parametre izbrali tako, da bodo imeli enake vrednosti. Za oceno uspešnosti vodenja bomo poleg grafičnih prikazov uporabili tudi kvantitativna merila: povprečje kvadratov

pogreška (MSE) in vsoto absolutnih diferenc vhoda ($SAiD$):

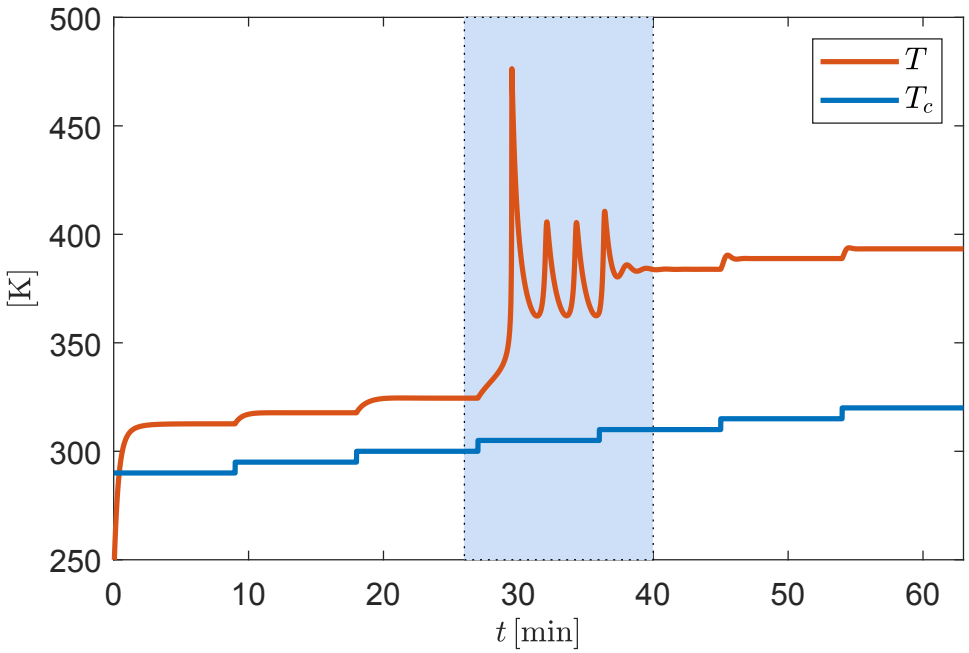
$$MSE = \frac{1}{N}\sum_{k=1}^{N} e_k^2 \tag{3.46}$$

$$SAiD = \sum_{k=1}^{N} |\Delta u_k| \tag{3.47}$$

kjer je $\Delta u_k = u_k - u_{k-1}$ diferenca vhodnega signala in N število podatkovnih vzorcev. Poleg tega bomo za oceno uspešnosti vodenja uporabili tudi čas vzpona (med 10 % in 90 %), čas umiritve (pogrešek znotraj območja ±2 %) in maksimalni prevzpon.

Nastavitveni parametri

Kot smo že omenili, je mehanizem samorazvijanja pri obeh regulatorjih (FCPFC in RECCo) nastavljen z enakimi parametri: $\gamma_{max} = 0.93$, $n_{add} = 20$, in $c_{max} = 20$. Prav tako je faktor mehkosti enak pri obeh primerih in smo ga določili eksperimentalno oziroma s poskušanjem $\eta = 10$. Vhodni podatek v (3.31) je definiran



Slika 3.27: Odprtozančni odziv procesa CSTR, kjer je T_c vhodni in T izhodni signal procesa.

kot $\mathbf{x}_k = [y_{k-1},\ u_{k-1}]^T$, kar nam definira regresijski vektor $\boldsymbol{\psi}_k = [y_{k-1},\ u_{k-1},\ 1]^T$ (3.32). Glede na čas vzorčenja $T_s = 0.01\,\text{min}$ in na želeno zaprtozančno časovno konstanto $\tau = 0.1\,\text{min}$ smo izračunali parameter referenčnega modela v (3.38) z uporabo približka $a_r \approx 1 - T_s/\tau = 0.9$. Sledijo še parametri, ki so specifični za vsako metodo. Za metodo FCPFC smo izbrali 4-koračni predikcijski horizont $H = 4$, kjer smo upoštevali (3.44). Pri RECCo-regulatorju smo določili ojačenje adaptacije $\alpha_P = \alpha_I = \alpha_D = \alpha_R = 4$ in mrtvo cono $d_{dead} = 2$. Območje vhodnega signala je od 250 do 350 K in izhodnega signala od 300 do 390 K.

Rezultati

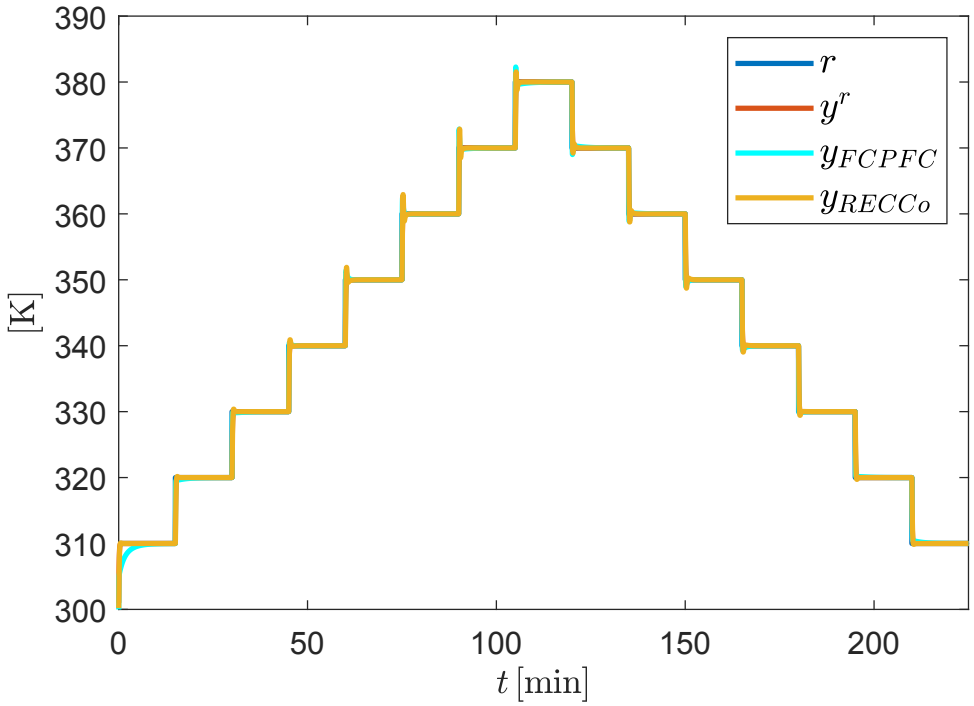
Delovanje regulatorjev FCPFC in RECCo smo testirali v osmih delovnih točkah od 310 K do 380 K. Na sliki 3.28a so prikazani referenca r, izhod referenčnega modela y^r in izhoda procesa obeh regulatorjev y_{FCPFC} in y_{RECCo}. Slika je nekoliko nepregledna saj težko razlikujemo odziva procesa med regulatorjema. Primarni cilj te slike je pokazati obliko referenčnega signala in vse delovne točke. Zato smo odzive procesa obeh regulatorjev za vse delovne točke ustrezno preslikali v en graf (slika 3.28b). S primerjavo obeh grafov na sliki 3.28b lahko ocenimo, da ima regulator FCPFC krajši dvižni čas na račun daljšega časa umiritve. Odzivi obeh regulatorjev pa imajo približno enak maksimalni prevzpon. Vrednosti vseh parametrov za vsako delovno točko so prikazani v tabeli 3.9. Puščice v prvem stolpcu poleg vrednosti delovne točke pomenijo smer premika referenčnega signala. Če je puščica usmerjena navzgor, pomeni, da je bila prejšnja delovna točka nižja od trenutne in obratno. Poleg teh kazalnikov kakovosti regulacije smo uporabili tudi oceno porabljene »energije« (3.47) in vsoto pogreška (3.46). Rezultati teh so prikazani v tabeli 3.10. Iz tabele je razvidno, da ima vodenje z regulatorjem PCPFC manjši kumulativni pogrešek MSE, vendar na račun porabljene energije vhodnega signala. Splošno lahko zaključimo, da oba regulatorja uspešno opravljata vodenje procesa CSTR s to razliko, da je FCPFC bolj učinkovit pri odpravljanju pogreška, medtem ko je RECCo bolj varčen glede porabljene »energije« (tabela 3.10).

Tabela 3.9: Primerjava med regulatorjema FCPFC in RECCo.

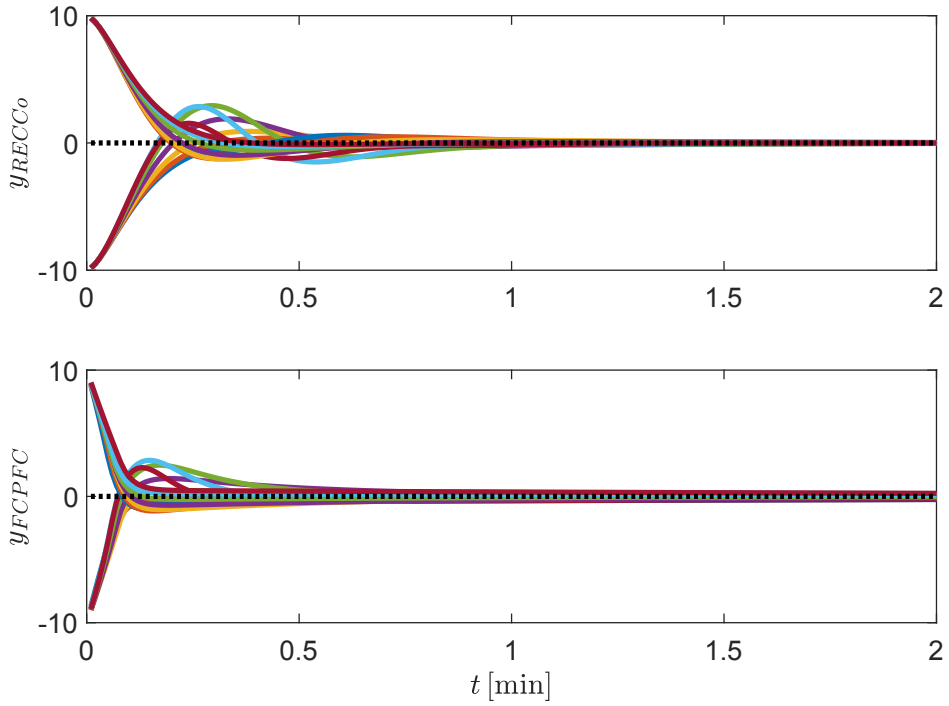
	Dvižni čas [min]		Čas umiritve [min]		Maksimalni prevzpon [%]	
r_k [K]	RECCo	FCPFC	RECCo	FCPFC	RECCo	FCPFC
↑ 320	0,21	0,08	0,33	1,08	0,05	0,00
↑ 330	0,19	0,08	0,65	0,17	0,12	0,00
↑ 340	0,17	0,07	0,65	0,72	0,27	0,13
↑ 350	0,14	0,06	1,09	1,05	0,54	0,40
↑ 360	0,12	0,05	1,02	0,70	0,81	0,69
↑ 370	0,12	0,05	0,87	1,31	0,76	0,77
↑ 380	0,12	0,05	0,90	2,90	0,41	0,60
↓ 370	0,14	0,05	0,90	2,73	0,21	0,27
↓ 360	0,13	0,06	1,06	0,57	0,34	0,32
↓ 350	0,14	0,06	1,09	1,03	0,36	0,30
↓ 340	0,15	0,07	0,66	1,38	0,28	0,21
↓ 330	0,17	0,07	0,69	0,49	0,17	0,07
↓ 320	0,19	0,09	0,61	0,25	0,09	0,00
↓ 310	0,21	0,11	0,34	0,38	0,04	0,00

Tabela 3.10: Primerjava učinkovitosti vodenja med regulatorjema FCPFC in RECCo.

	MSE [K^2]	SAiD [K]
FCPFC	0.28	1985
RECCo	0.54	1066



(a) Prikaz reference r, izhoda referenčnega modela y^r in izhoda procesa y.



(b) Prikaz zamaknjenih odzivov procesa (y_{RECCo} in y_{FCPFC}) za vsako delovno točko.

Slika 3.28: Prikaz rezultatov vodenja sistema CSTR z RECCo in FCPFC-regulatorjem.

3.7. Zaključek

V tem poglavju smo predstavili samorazvijajoči se adaptivni mehki regulator RECCo. Regulacijski algoritem sprotno prilagaja svojo strukturo z dodajanjem novih oblakov (mehkih pravil) ter adaptira parametre delnih regulatorjev PID-R. En regulator pripada enemu oblaku. Skupni izhod regulatorja se v vsakem trenutku izračuna na podlagi normaliziranih lokalnih gostot med trenutnim podatkom in obstoječimi oblaki. Predlagali smo postopek nastavljanja začetnih parametrov regulatorja za različne procese. Vsi nastavitveni parametri regulatorja se določijo z osnovnem znanju o procesu, ki ga reguliramo (delovno območje regulirnega in reguliranega signala, želena časovna konstanta zaprtozančnega procesa in čas vzorčenja).

Delovanje RECCo-regulatorja smo najprej preizkusili na simulacijskih primerih ter primerjali učinkovitost vodenja s klasičnim PID-regulatorjem (z uporabo različnih nastavitvenih pravil). Pokazalo se je, da za procese z nelinearno karakteristiko RECCo-regulator izkazuje boljše rezultate (primerjali smo maksimalni prevzpon, dvižni čas in čas izravnave). Z enakimi nastavitvami začetnih parametrov, kot smo jih uporabili za simulacijske modele, smo preizkusili delovanje RECCo-regulatorja na dveh realnih napravah: na toplotnem izmenjevalniku in sistemu dveh povezanih tankov. V prvem primeru reguliramo temperaturo vode, medtem ko v drugem nivo vode v rezervoarju. V obeh primerih regulator uspešno ujame predpisano dinamiko vodenja in regulirana veličina sledi izhodu referenčnega modela. Na realnem procesu dveh povezanih tankov smo testirali tudi vpliv zunanje motnje na delovanje RECCo-regulatorja. Pokazali smo, da regulator zelo hitro izniči vpliv motnje brez opaznega efekta na regulirano veličino.

V tem poglavju smo pokazali tudi, da lahko algoritem RECCo uporabimo skupaj z vzporednim krmilnikom z upoštevanjem motnje (ang. *feedforward controller*). Ta princip vodenja smo preizkusili na modelu sončnega kolektorja, kjer je osnovni problem v tem, da nimamo neposrednega vpliva na vira energije (sončno sevanje), lahko pa reguliramo pretok medija. V tem primeru smo pokazali, da predlagana rešitev uspešno rešuje problem zunanjih motenj.

Nenazadnje smo predlagali tudi kombinacijo samorazvijajočega se modela na podlagi mehkih oblakov za identifikacijo modela procesa in prediktivno funkcij-

ski regulator za izvedbo vodenja (podpoglavje 3.6). S predikcijskim pristopom lahko precej skrajšamo dvižni čas odziva na stopnico v primerjavi z RECCo-regulatorjem. Je pa res, da prediktivni pristop zahteva identifikacijo modela procesa, kar včasih ni tako trivialen problem.

4. Samorazvijajoča se mehka identifikacija nelinearnih dinamičnih procesov

4.1. Uvod in pregled literature

Modeli realnih sistemov so bistvenega pomena na skoraj vseh področjih, kot so npr. tehnično področje (elektrotehnika, mehatronika itn.), kot tudi netehnično področje (ekonomija, biologija, medicina, kemija itn.). Pridobljeni (matematični) modeli so lahko uporabni za različne namene. Z njimi napovedujemo obnašanje sistema v prihodnosti (predikcija) ali pa simuliramo njegovo obnašanje za daljše časovno obdobje in pod različnimi pogoji, ki si jih težko privoščimo na realnem sistemu. S tem lahko pridobimo nova znanja o sistemu in bolje razumemo, kako sistem pravzaprav deluje. S uporabo modelov lahko tudi optimiziramo obnašanje sistema in to znanje prenesemo na realni sistem. Poleg tega večina naprednih regulacijskih metod temelji na modelu procesa. Model se lahko uporabi za neposredno načrtovanje regulacijskega algoritma ali pa za linearizacijo procesa v delovni točki. Potrebno je poudariti, da je pri načrtovanju vodenja bolj pomembno ustrezno delovanje regulacijskega algoritma, kot pa sama natančnost modela procesa.

Področje identifikacije sistemov je precej široko in pomembno za industrijo in znanost, kar je rezultiralo v objavi številnih člankov in knjig [123, 165, 187]. Metode identifikacije sistemov lahko razdelimo v več kategorij, in sicer na podlagi naslednjih kriterijev [123]:

- vrsta matematičnega modela (parametrične in neparametrične metode),
- uporabljeni testni signali (deterministične, stohastične, psevdo-stohastične metode),
- izračun pogreška med modelom in procesom (vhodni, izhodni in generalizirani pogrešek),
- izvedba eksperimenta in vrednotenje (sprotne in nesprotne metode),

- algoritem za obdelavo podatkov (nerekurzivne, rekurzivne in metode v realnem času).

Identifikacijo mehkih sistemov lahko razdelimo v dve skupini: sprotna identifikacija [16, 19, 26, 31, 124, 135, 136, 138, 158, 167, 170, 171, 215, 234] in nesprotna indentifikacija [72, 124, 211]. Veliko teh metod je že bilo razloženih in omenjenih v poglavju 2, kjer je bil poudarek na samorazvijajočih se modelih. Na splošno se pri nesprotnem identificiranju modelov (ang. *off-line*) parametri in struktura modela posodobijo šele, ko so vsi (merjeni) podatki na voljo. Najprej se določi struktura modela in nato se določijo še parametri lokalnih modelov. Eden od načinov določanja strukture modela (mehka pravila) je lahko poizkušanje (ang. *trial-and-arror approach*). Bolj splošen način strukturne identifikacije je razdelitev vsake vhodne spremenljivke na mehka področja, kar rezultira v mehki mreži. Tak pristop uporablja metoda ANFIS [124]. Skoraj vse metode delijo vhodno-izhodni prostor problema na mehka področja z različnimi metodami rojenja (ang. *clustering*). Nekaj metod je naštetih v nadaljevanju: mehko rojenje s c-središči (ang. *fuzzy c-means*, FCM) [37], metoda Gustafson-Kessel (GK) [103], LOLIMOT (ang. *local linear model trees*) [188], SUHICLUST (ang. *supervised hierarchical clustering*) [106], *mountain clustering* [261] itn. Pri metodah sprotne (ang. *on-line*) identifikacije modelov se struktura in parametri posodabljajo z vsakim sprejetim podatkom (z vsako iteracijo algoritma). Nekaj *off-line* metod ima tudi svojo sprotno/rekurzivno različico, kot na primer: rekurzivna GK (rGK) [89] in rekurzivna FCM (rFCM) [90].

V nadaljevanju bomo predstavili algoritem na osnovi samorazvijajočega se modela za identifikacijo mehkega modela procesa, ki je bila objavljena v [12]. Metoda sprotno identificira strukturo mehkega modela v obliki oblakov podatkov in parametrov lokalnih modelov, ki so posodobljeni z metodo uteženih najmanjših kvadratov.

4.2. Identifikacija samorazvijajočega se mehkega modela

O samem postopku identifikacije modela smo že govorili v podpoglavju 3.6. Celoten postopek in vse enačbe bodo v enakem smislu uporabljene tudi v tem poglavju, s tem da bomo na tem mestu dodali nov zaščitni mehanizem ob dodajanju novih oblakov in bomo preučili vpliv brisanja obstoječih oblakov na podlagi

različnih kriterijev. Celoten postopek za identifikacijo modela procesa je predstavljen na sliki 4.1.

Kot je razvidno s slike 4.1, vsebuje mehanizem samorazvijanja dva koraka, in sicer enega za dodajanje novih in drugega za odstranjevanje obstoječih oblakov. Oba bosta razložena v nadaljevanju.

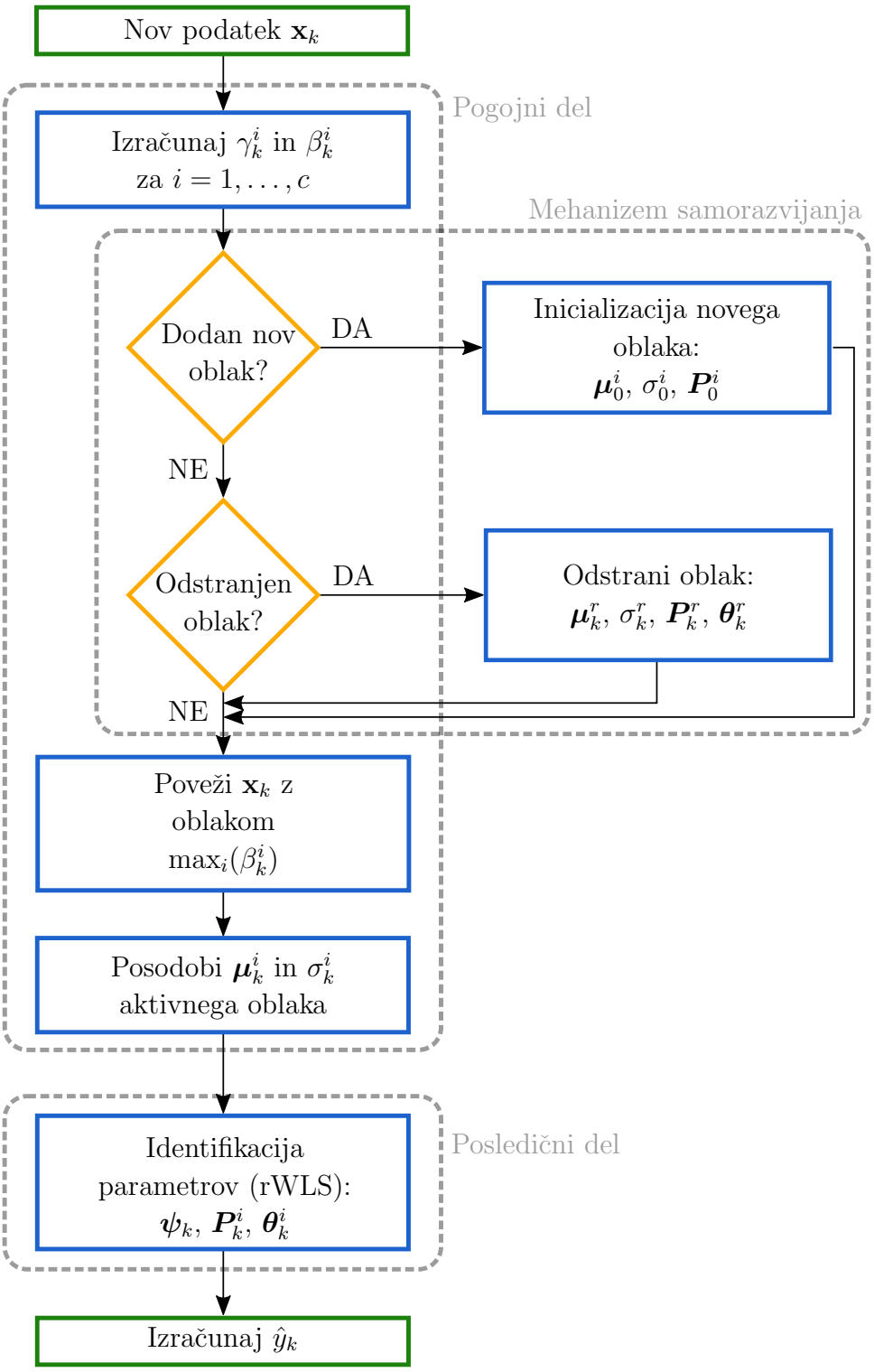
4.2.1 Dodajanje novih oblakov

Mehanizem dodajanja novih oblakov je bil že razložen v razdelku 3.2.2. V tem razdelku bomo na hitro ponovili vse kriterije ter dodali novega, ki preprečuje dodajanje novih oblakov zaradi osamelcev. Nov oblak je definiran v primeru, če so izpolnjeni vsi kriteriji, našteti v nadaljevanju:

1. Prvi kriterij se nanaša na *bližino* oziroma *gostoto* trenutnega podatka $\mathbf{x}_k$ do obstoječih oblakov. V primeru, da je podatek relativno daleč od vseh oblakov, pomeni, da je zaznano novo področje procesa. Prvi kriterij je definiran kot sledi:
$$C_1^{add} = (\max_i(\gamma_k^i) < \gamma_{max})\,;$$
kjer je $\gamma_{max} \in [0,\ 1]$. Vrednost parametra γ_{max} lahko zavzame katerokoli vrednost definicijskega območja. V praksi je ponavadi ta vrednost med 0,75 in 0,95. Večje vrednosti parametra omogočijo dodajanje številnih novih oblakov, medtem ko manjše vrednosti dodajo le nekaj oblakov.

2. Drugi kriterij prepreči dodajanje novih oblakov enega za drugim. Pravzaprav ta mehanizem za nekaj korakov n_{add} zamrzne celotni mehanizem dodajanja novih oblakov ob trenutku k_{add}, ko smo ravnokar dodali nov oblak:
$$C_2^{add} = (k > k_{add} + n_{add})\,;$$
kjer je n_{add} parameter, ki ga določi operater. Ta kriterij omogoča novim oblakom, da pridobijo nekaj znanja o sistemu oziroma dovolj podatkov za verodostojno oceno kovariančne matrike algoritma rWLS.

3. Tretji kriterij je povezan z maksimalnim številom oblakov c_{max}, ki jih lahko dodamo:
$$C_3^{add} = (c < c_{max})\,;$$



Slika 4.1: Shematski prikaz samorazvijajočega se modela za identifikacijo dinamičnih procesov.

kjer je c število obstoječih oblakov. Pravzaprav je ta kriterij uporaben le v primeru, ko vnaprej poznamo število oblakov, ki jih želimo identificirati. V večini naših primerov pa imamo opravka s tako imenovanim modelom »črne škatle« (ang. *black-box model*), kjer nimamo veliko informacij o procesu.

4. Zadnji kriterij dopolnjuje prvega in preprečuje dodajanja novih oblakov na podlagi osamelcev (ang. *outliers*). Princip je prikazan na sliki 4.2. Na začetku grafa vidimo, da trije vzorci izpolnjujejo prvi kriterij, a to so pravzaprav osamelci, ker je trend modre krivulje ($\max_i(\gamma_k^i)$) še vedno večji od parametra γ_{max}. Zato definiramo parameter τ_{add}, ki označuje število zaporednih vzorcev, ki morajo izpolnjevati prvi kriterij:

$$C_4^{add} = (k_{C_1} > \tau_{add})\,;$$

kjer je k_{C_1} število zaporednih vzorcev, ki že izpolnjujejo prvi pogoj C_1^{add}. Vrednost parametra τ_{add} smo izbrali enako, kot je dimenzija regresijskega vektorja. V večini primerov se to izkaže kot pravilna izbira [87, 105].

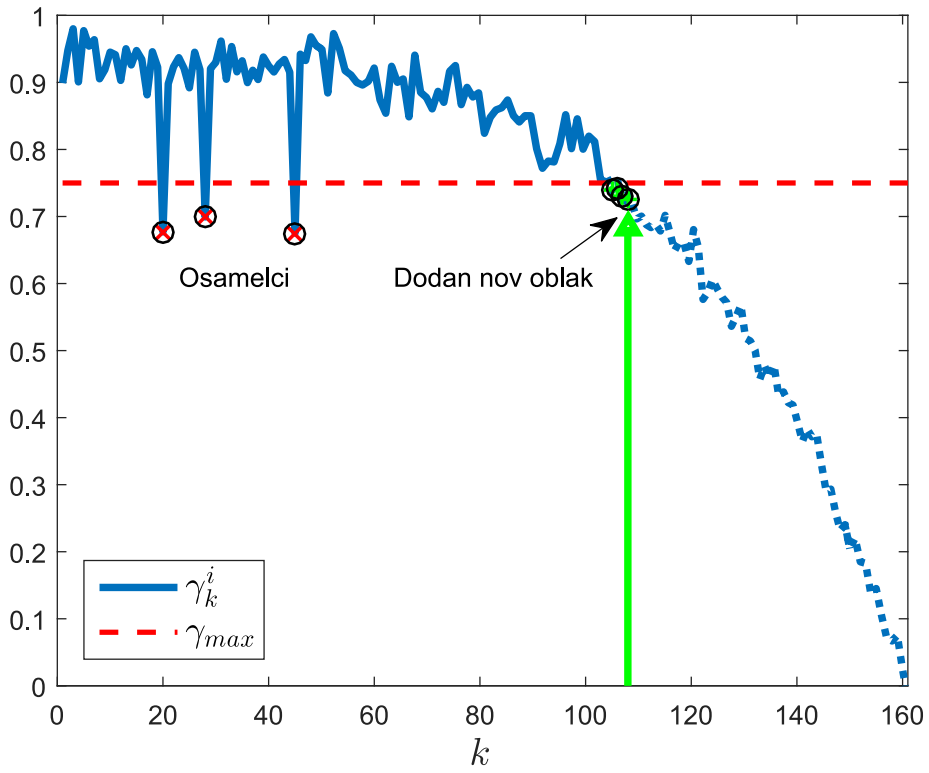
Kot smo že omenili, morajo biti vsi navedeni kriteriji izpolnjeni ($C_1^{add} \wedge C_2^{add} \wedge C_3^{add} \wedge C_4^{add}$), da lahko dodamo nov oblak. S tem namenom moramo nastaviti tri parametre γ_{max}, n_{add} in c_{max}, medtem ko smo τ_{add} določili z dimenzijo regresijskega vektorja.

4.2.2 Odstranjevanje oblakov

Poleg vseh kriterijev za preprečevanje dodajanja novih oblakov na podlagi osamelcev nekateri oblaki še vedno premalo prispevajo k skupnemu izhodu modela [87]. S tem namenom lahko definiramo mehanizme za odstranjevanje *manj aktivnih* ali *manj informativnih* oblakov. *Aktivnost* oblaka je definirana z relativnim številom podatkov, ki pripadajo temu oblaku od trenutka njegovega nastanka. Koncept brisanja na podlagi aktivnosti je definiran z naslednjim kriterijem:

$$C_1^{rem} = \left(\frac{M^i}{k - k^i} < \zeta\frac{1}{c}\right), \qquad i = 1, \ldots, c$$

kjer je c število obstoječih oblakov, M^i število podatkov, ki pripadajo i-temu oblaku, k^i časovni trenutek dodajanja i-tega oblaka in $\zeta \in [0,\ 1]$ parameter, ki ga moramo določiti. Če je vrednost $\zeta = 0$, potem je odstranjevanje oblakov



Slika 4.2: Prikaz mehanizma za preprečevanje dodajanja novih oblakov na podlagi osamelcev. Rdeča črtkana linija je vrednost parametra γ_{max} in modra črta predstavlja vrednost maksimalne gostote $\max(\gamma_k^i)$.

onemogočeno, medtem ko vrednost parametra $\zeta = 1$ pomeni, da z vsakim novim dodanim oblakom izbrišemo enega obstoječega. Izbira parametra $\zeta = 1$ ni čisto uporabna rešitev, zato je v praksi definicijsko območje parametra definirano z intervalom $[0,\ 1)$. V literaturi je vrednost tega parametra ponavadi nastavljena na 0,1 [87].

Pri odstranjevanju oblakov moramo biti precej konzervativni in pazljivi, ker ne želimo odstraniti oblaka, ki vsebuje koristne informacije o našem procesu. Z uporabo samo prvega kriterija C_1^{rem} se potencialno to lahko zgodi, ker se gleda samo relativno število podatkov, ki jih oblak vsebuje, ne pa tudi informacija, ki jo prinaša. Zato smo poleg brisanja manj aktivnih oblakov dodali še en mehanizem, ki odstanjuje »manj informativne« oblake. Informativnost oblakov je definirana s številom vzorcev $\tilde{M}^i$, ki imajo najmanjšo gostoto ($\min_i(\gamma_k^i)$). To pomeni, da za

vsak vzorec izračunamo gostoto γ_k^i z vsemi oblaki ($i = 1, \ldots, c$) in se ta vzorec dodeli oblaku z najmanjšo gostoto $\min_i(\gamma_k^i)$. Tako vsak oblak vsebuje M^i vzorcev, ki pripadajo temu oblaku z največjo gostoto $\max_i(\gamma_k^i)$ kot tudi $\tilde{M}^i$ vzorcev, ki pripadajo oblaku z najmanjšo gostoto $\min_i(\gamma_k^i)$. Drugi pogoj za odstranjevanje oblakov je potem definiran z naslednjim kriterijem:

$$C_2^{rem} = \left(1 - \frac{\tilde{M}^i}{k - k^i} < \frac{\zeta}{2}\right), \qquad i = 1, \ldots, c$$

kjer je ζ isti parameter kot pri prvem kriteriju C_1^{rem} in ima isto vrednost (0,1). Celoten mehanizem odstranjevanja oblakov upošteva oba pogoja $C_1^{rem} \wedge C_2^{rem}$.

Še enkrat omenimo, da je celoten algoritem identifikacije modela procesa predstavljen grafično na sliki 4.1, kjer so razvidni vsi koraki pogojnega, posledičnega in samorazvijajočega se dela.

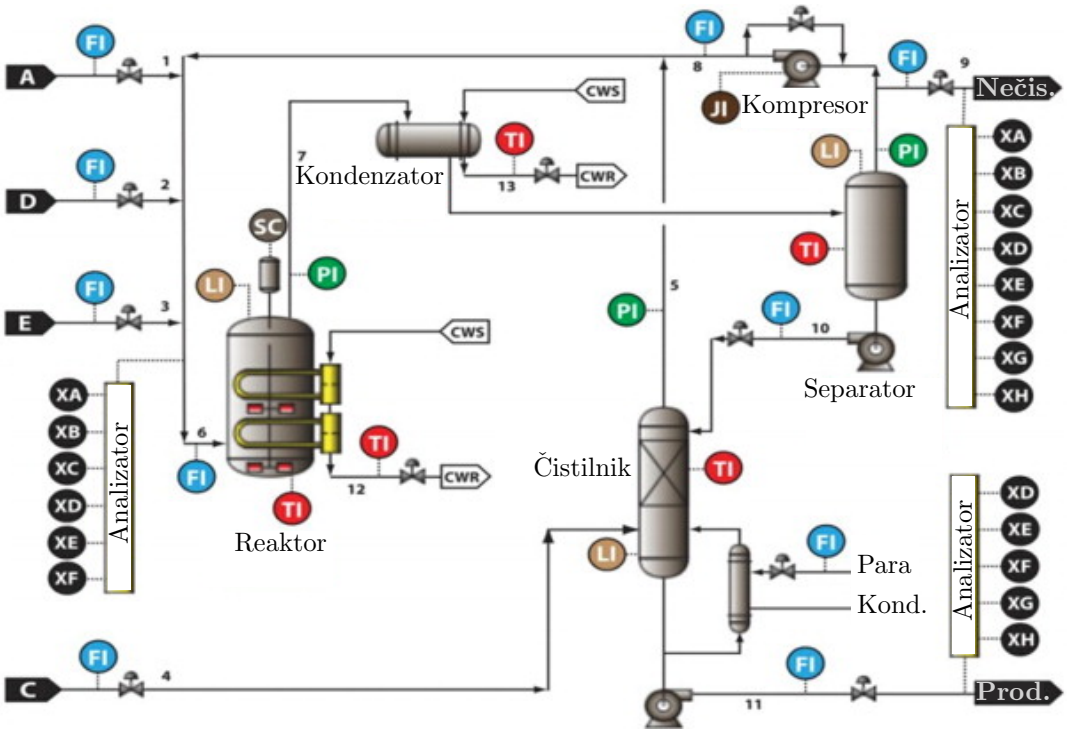
V nadaljevanju bomo preko dveh primerov (enega simulacijskega in drugega na podlagi realnih podatkov) predstavili delovanje in efektivnost algoritma samorazvijanja za identifikacijo modelov na podlagi mehkih oblakov. Prvi primer je model procesa Tenneessee Eastman in drugi realen proces hladilne postaje.

4.3. Proces Tennessee Eastman

Proces Tennessee Eastman (TE) je bil predstavljen v [91] in je realističen ter reprezentativen model kemijskega produkcijskega procesa. Uporabljen je bil v številnih raziskavah na različnih področjih (vodenja, identifikacije, zaznavanje napak itn. [27, 69, 151, 224, 263, 276]). Glavne značilnosti procesa TE so njegova kompleksnost, nelinearnost in odprtozančna nestabilnost. Prikazan je na sliki 4.3. Sestavljen je iz 41 merljivih in 12 manipulativnih spremenljivk, kar omogoča izvajanje različnih eksperimentov in študij. Proces TE je sestavljen iz petih glavnih enot: mešalni *reaktor*, *hladilnik*, *separator* pare in tekočine, povratni *kompresor* ter *čistilnik* produkta. Enote so med seboj povezane z 12 vodi. Pet vhodnih plinov (reagentov), označenih z A, B, C, D in E, se dozira v mešalni reaktor in reagira do tekočih produktov. Reagenti zapustijo vstopni reaktor kot hlapne snovi, ki potem z hladilnikom kondenzirajo in se v separatorju delijo na tekočino in paro. Nekondenzirani plini se vračajo v reaktor, medtem ko se ostali dozirajo v čistilnik produkta. Na koncu procesa dobimo dva produkta (G in H), ki zapustita čistilnik, in en stranski produkt (F) kot nečistoča. Ireverzibilne in

eksotermne kemične reakcije so podane z naslednjimi reakcijami:

$$A(g) + C(g) + D(g) \rightarrow G(liq), \qquad \text{Produkt 1},$$
$$A(g) + C(g) + E(g) \rightarrow H(liq), \qquad \text{Produkt 2},$$
$$A(g) + E(g) \rightarrow F(liq), \qquad \text{Stranski produkt},$$
$$3D(g) \rightarrow 2F(liq), \qquad \text{Stranski produkt}.$$



Slika 4.3: Shematski prikaz procesa Tennessee Eastman [276].

4.3.1 Opis problema

Za demonstracijo koncepta identifikacije na podlagi samorazvijajočega se mehkega modela smo uporabili kazalnike proizvodne učinkovitosti (ang. *performance production indicators*, pPI) procesa TE. Kazalniki so bili definirani v [99, 100] kot *Strošek*, *Produktivnost* in *Kakovost*. Poleg tega so avtorji v [100] naredili obsežno analizo in določili najbolj vplivne komponente pri izračunu kazalnikov pPI. V tabeli 4.1 so prikazane vse krmilne spremenljivke procesa TE.

Tabela 4.1: Krmilne spremenljivke procesa TE [100].

Oznaka	Opis spremenljivke
F_p	Proizvodni indeks
R_2	Nivo čistilnika produkta
R_3	Nivo separatorja pare in tekočine
R_4	Nivo reaktorja
R_5	Pritisk v reaktorju
R_7	$\%C$ v odvodni cevi za nečistoče
R_8	Pretok povratnega voda
R_9	Temperatura v reaktorju
r_2	Pretočno razmerje vodov D in E

Za vsak kazalnik uspešnosti pPI so avtorji v [100] določili najbolj vplivne komponente in posamezen regresijski vektor je določen kot sledi:

$$\begin{aligned} Strošek : \mathbf{x}_{f,k} = [&y_{k-1}, y_{k-2}, F_{p,k-1}, F_{p,k-2}, F_{p,k-4}, R_{4,k-1}, R_{7,k-1}, R_{7,k-5}, \\ &R_{9,k-1}, R_{9,k-5}, r_{2,k-1}, r_{2,k-4}, r_{2,k-5}]^T \\ Produktivnost : \mathbf{x}_{f,k} = [&y_{k-1}, y_{k-2}, F_{p,k-1}, F_{p,k-5}, R_{4,k-1}, R_{4,k-4}, R_{7,k-1}, r_{2,k-5}]^T \\ Kakovost : \mathbf{x}_{f,k} = [&y_{k-1}, y_{k-2}, F_{p,k-1}, R_{4,k-1}, r_{2,k-5}]^T \end{aligned}$$

Glavni cilj v tem podpoglavju je identificirati kazalnike pPI (*Strošek*, *Produktivnost* in *Kakovost*) z uporabo samorazvijajočega se modela, predstavljenega v podpoglavju 4.2 kot tudi v 3.6.1.

4.3.2 Rezultati

Za namen preizkušanja in evaluacije smo pridobili tri sete vhodno-izhodnih simuliranih podatkov na modelu TE. Prva skupina podatkov je poimenovana TRAIN in smo jo uporabili za pridobivanje mehkih modelov vsakega kazalnika pPI. To fazo imenujemo tudi faza učenja. Ostala dva seta podatkov (poimenovana VAL1 in VAL2) sta uporabljena za vrednotenje pridobljenih modelov. V fazi vrednotenja smo uporabili dva pristopa, in sicer sprotno (ang. *on-line*) in nesprotno

učenje (ang. *off-line*). Sprotno učenje pomeni, da so tudi v času vrednotenja aktivni mehanizmi dodajanja, odvzemanja oblakov in prilagajanja parametrov $(\boldsymbol{\mu}^i, \sigma^i, \boldsymbol{P}^i, \boldsymbol{\theta}^i)$, ob nesprotnem vrednotenju pa uporabimo modele, ki so bili naučeni le iz podatkov TRAIN, in jih ne spreminjamo. Pridobljene modele iz faze učenja smo ovrednotili tako, da smo primerjali 4-koračne predikcije med modelom in realnim procesom nad podatki VAL1 in VAL2. Ocenjeni izhod modela v prihodnosti smo označili z $\hat{y}(k+H)$ in ga primerjali z dejanskim izhodom $y(k+H)$ oziroma smo izračunali vsoto kvadratov pogreška med njima z naslednjo enačbe:

$$MSE = \frac{1}{N-H} \sum_{k=1}^{N-H} (\hat{y}(k+H) - y(k+H))^2 \tag{4.1}$$

kjer je N število vzorcev v podatkovnem setu.

Rezultate, ki smo jih dobili s predlaganim samorazvijajočim se modelom, smo primerjali z obstoječima metodama: eFuMo [86] in umetne nevronske mreže (ang. *artifitial neural network*, ANN) [189]. Rezultati obeh teh metod so del tehničnega poročila [101]. Želimo in je potrebno poudariti, da so modeli, pridobljeni z ANN, optimalni in določeni za vsak pPI posebej. Parametri modelov ANN so bili optimirani z uporabo metode Levenberg-Marquardt [187], medtem ko je bilo krajšanje strukture nevronske mreže izvedeno z metodo optimalnega »možganskega prereza« (ang. *optimal brain surgery*) [107]. V nasprotju s tem je bila metoda eFuMo nastavljena s privzetimi parametri, ki so enaki za vsak pPI. Parametri uporabljeni v tem primeru so prikazani v tabeli 4.2 in so enaki pri pridobivanju modelov vseh kazalnikov (*Strošek*, *Produktivnost* in *Kakovost*).

Za bolj pravično primerjavo med predlagano metodo na podlagi mehkih oblakov (podpoglavje 4.2) in metodo eFuMo (razložena tudi v razdelku 2.1.2) smo izbrali enake vrednosti parametrov, ki imajo enak pomen. V tabeli 4.2 so prikazane vrednosti nastavitvenih parametrov obeh metod. Kot smo že omenili, je metoda ANN optimirana za vsak model posebej in nima sorodnih parametrov, kot jih imata ostali dve metodi.

V fazi vrednotenja delovanja predlaganega pristopa smo analizirali tudi učinkovitost mehanizmov za dodajanje novih kot tudi za odstranjevanje obstoječih oblakov. S tem namenom smo izvedli štiri različne eksperimente: 1. OLD – vsebuje le kriterije za dodajanje novih oblakov (C_1^{add}, C_2^{add} in C_3^{add}); 2. OLD+ADD – vsebuje vse kriterije za dodajanje (C_1^{add}, C_2^{add}, C_3^{add} in C_4^{add});

Tabela 4.2: Uporabljeni nastavitveni parametri metode eFuMo [101].

	γ_c	γ_v	λ_r	τ	c_{max}	N_{out}	k_n	ζ	γ_{max}
eFuMo	0.9999	0.9999	0.9999	80	20	50	3	0.1	/
Predlagana metoda	/	/	0.9999	80	20	/	/	0.1	0.85

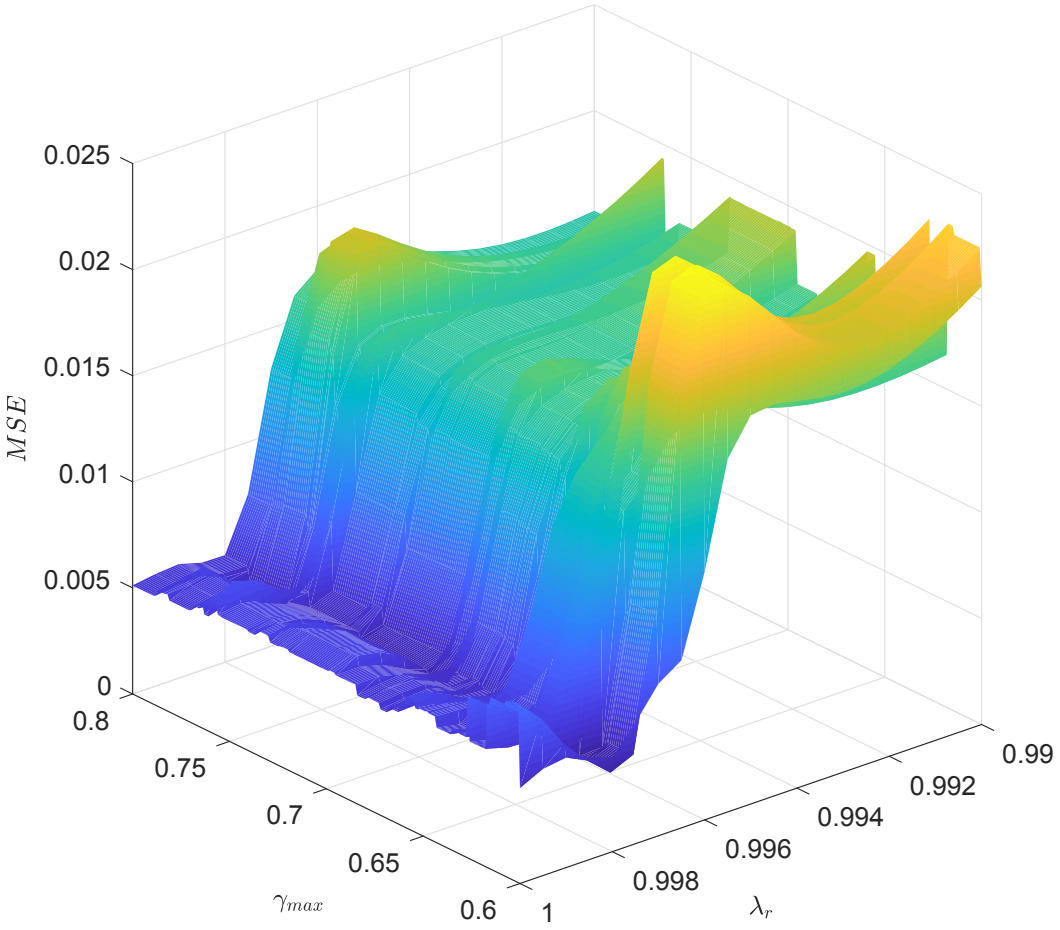
3. OLD+REM – vsebuje še mehanizem za odstranjevanje obstoječih oblakov;
4. NEW – vsebuje vse kriterije za dodajanje in odstranjevanje oblakov.

Glavni cilj pri tem eksperimentu je pokazati, kako nova mehanizma (ADD in REM) za dodajanje in odstranjevanje oblakov vplivata na učinkovitost celotnega identifikacijskega algoritma.

Za vse zgoraj razložene in izvedene eksperimente smo rezultate (povprečje vsote kvadratov pogreška MSE) prikazali v tabeli 4.3 za vsak kazalnik (*Strošek*, *Produktivnost* in *Kakovost*) ločeno. Iz podatkov v tabeli 4.3 lahko razberemo sledeče. Najprej, da vrednotenje s sprotnim učenjem (*on-line*) pričakovano daje boljše rezultate, ker se parametri modelov prilagajajo trenutnemu stanju signalov. Poleg tega lahko tudi primerjamo rezultate predlagane metode med različnimi mehanizmi dodajanja in odstranjevanja oblakov. Hitro lahko opazimo, da včasih dodatni mehanizmi prispevajo k boljšemu rezultatu, včasih pa žal tudi poslabšajo učinkovitost algoritma. Razlog leži v nastavljanju posameznih parametrov predlagane metode. Če želimo za določen problem dobiti najboljšo možno rešitev, moramo poiskati optimalne parametre. Na primer, na sliki 4.4 je prikazana odvisnost rezultata MSE od vrednosti parametrov γ_{max} in λ_r.

V primeru prvih dveh kazalnikov *Strošek* in *Produktivnost* predlagana metoda daje boljše rezultate kot eFuMo, čeprav je poterbno nastaviti manj parametrov in je kompleksnost algoritma manjša. Po drugi strani pa eFuMo daje boljše rezultate v primeru tretjega kazalnika *Kakovost*.

Iz tabele 4.3 je razvidno, da je metoda ANN uspešnejša od obeh, eFuMo in predlagane metode, vendar je to posledica dejstva, da so parametri modelov ANN optimirani za vsak kazalnik posebej. Po drugi strani je slabost metode ANN ta, da ima fiksno in nespremenljivo strukturo in se ne more prilagajati novim razmeram. Princip samorazvijanja ima to prednost, da se struktura razvija samodejno in na podlagi sprotnih podatkov, s privzetimi parametri pa dobimo precej pri-



Slika 4.4: Prikaz vpliva parametrov γ_{max} in λ_r na vrednost MSE.

merljive rezultate kot z metodo ANN. V nadaljevanju bomo predstavili problem identifikacije modela hladilne postaje. Uporabili bomo enako samorazvijajočo se metodo, kot v tem podpoglavju.

4.4. Hladilna postaja

Pomemben del skoraj vsake (moderne) tovarne predstavlja hladilna postaja (ang. *watter chiller plant*). Glavni namen hladilne postaje (HP) je oskrba procesov in podsistemov v tovarni s hladno tekočino (običajno voda ali pa mešanica glikola in vode). Eden največjih odjemalnikov tega hlada so sistemi za uravnavanje temperature, vlažnosti in menjavanja zraka v zaprtih prostorih (HVAC[1], ang. *heating, ventilation and air conditioning*). V sistemih HVAC se hladilna tekočina uporablja za hlajenje dovodnega zraka preko toplotnih izmenjalnikov. Poleg tega uporabljajo hladilno tekočino iz HP tudi drugi tehnološki postopki v tovarni (npr. reaktorji, bio-reaktorji itn.). Skratka, hladilna postaja je sestavljena iz nekaj ključnih podsistemov. Hladilni agregat (HA, ang. *water chiller*) je osnovni gradnik hladilne postaje, ki vodo na dovodu preko uparjalno-kondenzacijskega postopka ohladi na vnaprej določeno temperaturo. Kot stranski produkt hlajenja se pojavi odvečna toplota, ki jo je treba odvesti. Za optimalno delovanje HA skrbijo hladilni stolpi (HS, ang. *cooling tower*), ki preko ločenega tokokroga odvajajo odvečno temperaturo v okolico. Vzdrževanje HA v optimalni točki je pomemben del procesa, saj na ta način izboljšamo delovanje in efektivnost termodinamičnih procesov v samem HA.

V literaturi smo zasledili nekaj primerov vodenja, nadzora in optimizacije hladilnih postaj. V [6] je predstavljeno adaptivno vodenje hladilnega agregata na podlagi modela procesa. Avtorji v [271,272] so predstavili različna pristopa za optimizacijo delovanja več hladilih agregatov skupaj. Cilj optimizacije je zmanjšati porabo energije v hladilni postaji. V [271] je najprej predstavljen primer optimizacije na osnovi tabu področij (ang. *tabu search*, TS), medtem ko so v [272] avtorji predstavili primer optimizacije z uporabo genetskih algoritmov (ang. *genetic algorithms*, GA). Primer prediktivnega vodenja na podlagi modela je predstavljen v [172] (ang. *model predictive control*, MPC). Posebnost tega primera je ta, da

[1]V doktorski disertaciji bo uporabljena okrajšava HVAC za sisteme za uravnavanje temperature in vlažnosti ter menjavanje zraka v zaprtih prostorih.

Tabela 4.3: Prikaz in primerjava vrednosti MSE (4.1) za 4-koračno predikcijo izhoda za vse tri kazalnike pPI.

Strošek

TRAIN	VAL1				VAL2			
	OLD	OLD+ADD	OLD+REM	NEW	OLD	OLD+ADD	OLD+REM	NEW
Predlagana metoda (on-line)	4.9654	4.8923	4.9654	4.8923	7.1088	7.0964	7.1088	7.0964
Predlagana metoda (off-line)	5.1641	5.1478	5.1641	5.1478	7.4398	7.4436	7.4398	7.4436
eFuMo (off-line)		6.7610				3.7930		
Nevronska mreža		3.5050				5.8610		
Produktivnost								
Predlagana metoda (on-line)	0.1761	0.1788	0.1761	0.1788	0.1642	0.1659	0.1642	0.1659
Predlagana metoda (off-line)	0.1964	0.2006	0.1964	0.2006	0.1639	0.1664	0.1639	0.1664
eFuMo (off-line)		0.2654				0.3282		
Nevronska mreža		0.1393				0.1285		
Kakovost								
Predlagana metoda (on-line)	0.1123	0.1387	0.1203	0.1466	0.0687	0.0687	0.0684	0.0682
Predlagana metoda (off-line)	0.2310	0.2302	0.2853	0.2729	0.0683	0.0682	0.0694	0.0688
eFuMo (off-line)		0.0715				0.0881		
Nevronska mreža		0.0596				0.0599		

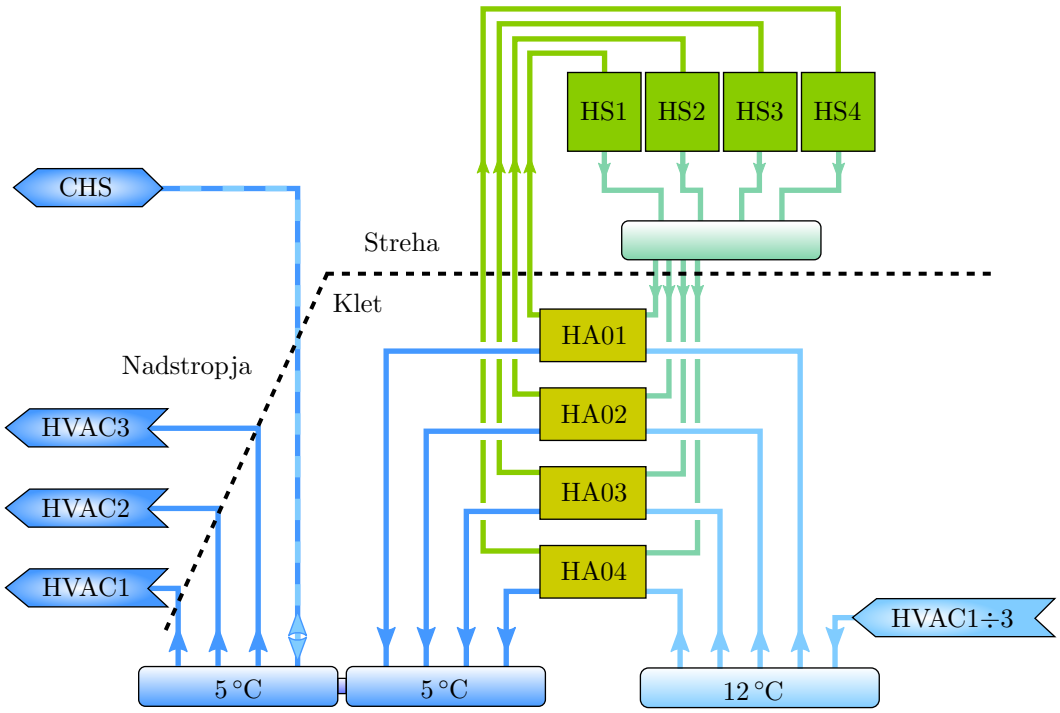
hladilna postaja poleg osnovnih gradnikov (HA in HS) vsebuje tudi rezervoar, ki služi za hranjenje odvečne hladilne energije. Cilj vodenja je minimizacija porabe energije ob zagotavljanju zahtev porabnikov. V [204] je predstavljen nevromehki model, pridobljen na podlagi podatkov iz realnega sistema. Na osnovi modela so razvili metodo za nadzor in predikcijo porabe energije in s tem optimirali delovanje celotnega procesa. Nenazadnje je bil v [184] predstavljen postopek optimizacije vodenja HP z več hladilnimi agregati z metodo ESC (ang. extremum seeking control).

Na sliki 4.5 je prikazan konkreten primer procesa hladilne postaje, ki bo razložen v nadaljevanju. V našem primeru je hladilna postaja sestavljena iz štirih vzporedno povezanih hladilnih agregatov (HA01 ÷ 04), vsak s svojim pripadajočim hladilnim stolpom (HS01 ÷ 04). Hladilna energija, ki jo proizvedejo hladilni agregati, se distribuira do odjemalnikov preko skupnega hladilnega mosta. V konkretnem primeru imamo tri velike odjemalnike (HVAC01 ÷ 03), naša HP pa je povezana s centralnim hladilnim sistemom (CHS). CHS je pravzaprav večje stikališče, kjer se več ločenih hladilnih postaj v tovarni poveže skupaj v skupni sistem. Namen tega je medsebojna izmenjava hladilne energije v primeru, ko določena HP proizvaja več energije, kot jo lahko porabijo njeni neposredni odjemalniki. Na ta način lahko celoten sistem CHS skrbi za optimalno delovanje vseh hladilnih postaj v tovarni.

4.4.1 Opis problema

Za hladilno postajo, prikazano na sliki 4.5, so na ravni vodenja že realizirani mehanizmi za vklop in izklop posameznih HA glede na vnaprej določene kriterije: trenutna poraba moči (P_{HVAC}) vseh porabnikov (HVAC01 ÷ 03), trenutna proizvodnja moči (P_{HP}) hladilne postaje in število obratovalnih ur posameznega agregata. Poleg tega ima vsak hladilni agregat (HA01 ÷ 04) svojo prioriteto delovanja (od 1 do 4), ki se spremeni enkrat mesečno oziroma na podlagi števila vklopov/izklopov posameznega HA, ki prav tako vpliva na spremembo prioritete. S tem se zagotovi enakomerno obratovanje vseh agregatov.

Ko so pogoji za vklop nedelujočega oziroma izklop delujočega agregata izpolnjeni, se začne postopek vklopa/izklopa HA. Ponavadi procedura vklopa/izklopa traja do 15 min (oziroma nekoliko dlje). V tem času se vzpostavijo potrebne



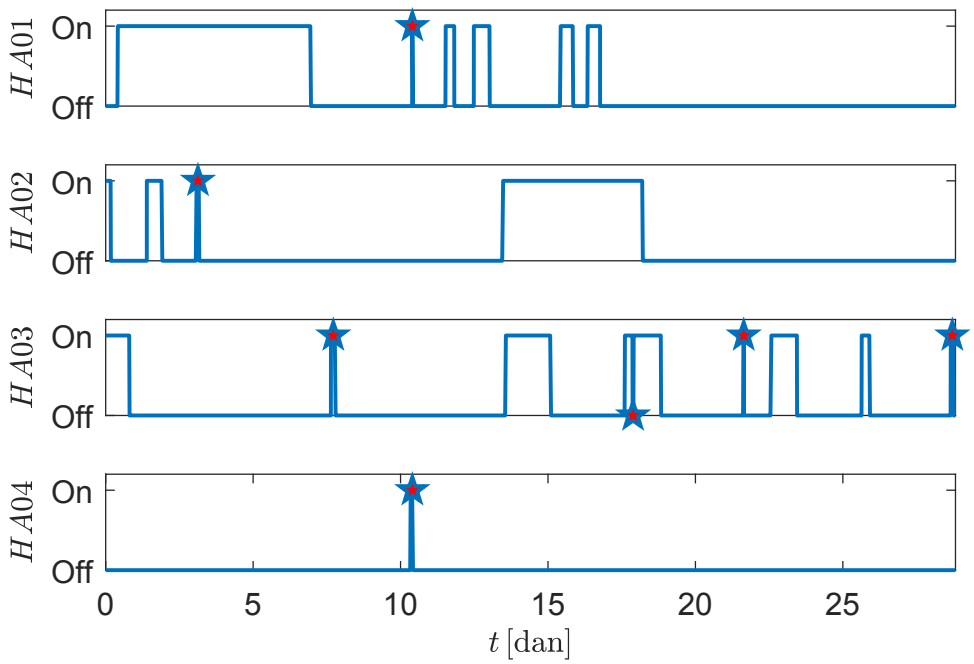
Slika 4.5: Shematski prikaz hladilne postaje (HA – hladilni agregat, HS – hladilni stolp, CHS – centralno hladilni sistem).

razmere (pretok, temperaturna razlika itn.) in šele nato lahko HA prispeva k celotnemu omrežju, zato želimo preprečiti kratkotrajne vklope in izklope HA. Na sliki 4.6 je prikazana sekvenca vklopov/izklopov vseh HA v obdobju 28 dni in z zvezdico so označeni nezaželeni kratkotrajni preklopi.

4.4.2 Rezultati

Da bi rešili problem, ki smo ga predstavili v prejšnjem razdelku, želimo pridobiti modele dveh ključnih spremenljivk (P_{HVAC} in P_{HP}). Na osnovi modelov bi napovedali vrednost v prihodnosti in s tem preprečili nepotrebno preklapljanje HA. Potrebne podatke smo pridobili iz podatkovne baze podjetja, kjer sistem dejansko obratuje. Na sliki 4.7 so prikazani podatki v obdobju 58 dni. Prvo polovico podatkov (vse do rdeče črtkane črte) smo uporabili za identifikacijo, medtem ko je druga polovica namenjena testiranju in vrednotenju pridobljenih modelov. Čas vzorčenja signalov je $T_s = 15\,\text{min}$.

Vektorja regresorjev (3.31) za pridobivanje modelov $\hat{P}_{HVAC}$ in $\hat{P}_{HP}$ z uporabo samorazvijajočim se mehkim modelom na osnovi oblakov smo izbrali na sledeč



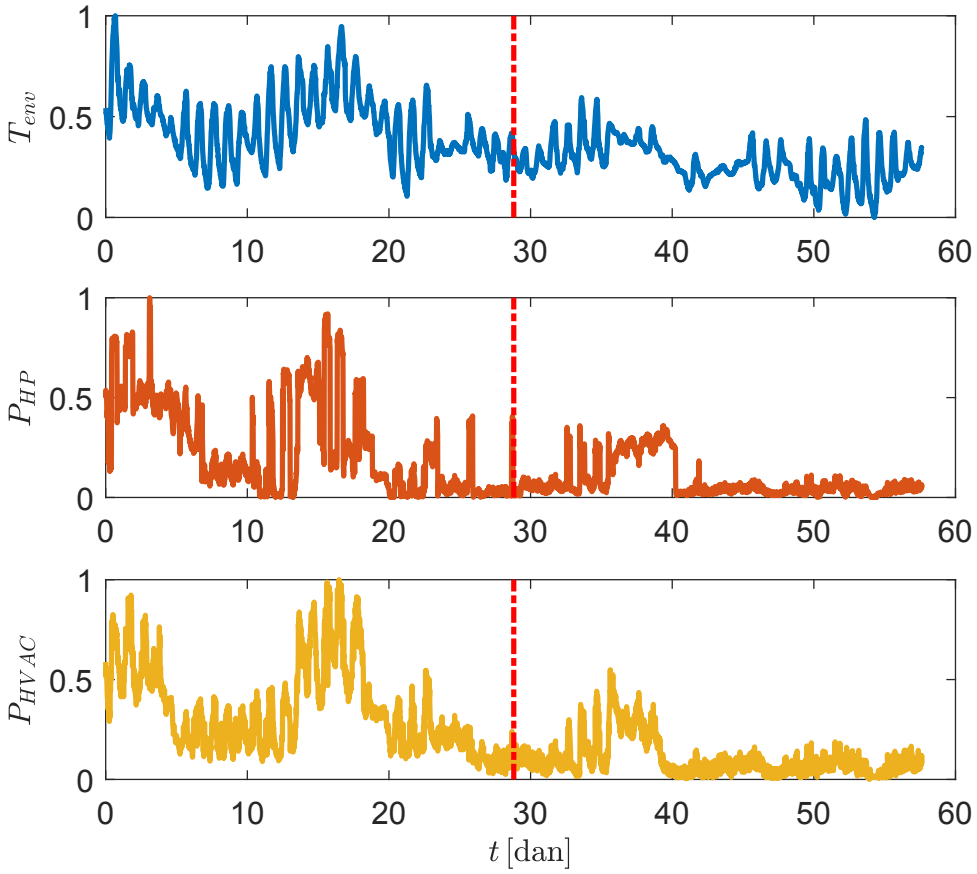
Slika 4.6: Prikaz stanj delovanja hladilnih agregatov. Zvezdice prikazujejo časovne trenutke nezaželenih kratkotrajnih preklopov.

način:

$$\hat{P}_{HVAC}:$$
$$\mathbf{x}_k = [P_{HVAC,k-1},\ P_{HVAC,k-2},\ T_{env,k-1},\ T_{env,k-2},\ T_{env,k-3}]^T$$
$$\hat{P}_{HP}:$$
$$\mathbf{x}_k = [P_{WCP,k-1},\ P_{WCP,k-2},\ T_{env,k-1},\ T_{env,k-2},\ T_{env,k-3}]^T$$

kjer smo v obeh primerih izbrali modele z enako strukturo $n_a = 2$ in $n_b = 3$. Ostali parametri identifikacijske metode so nastavljeni tako, kot je bilo razloženo v razdelku 4.3.2.

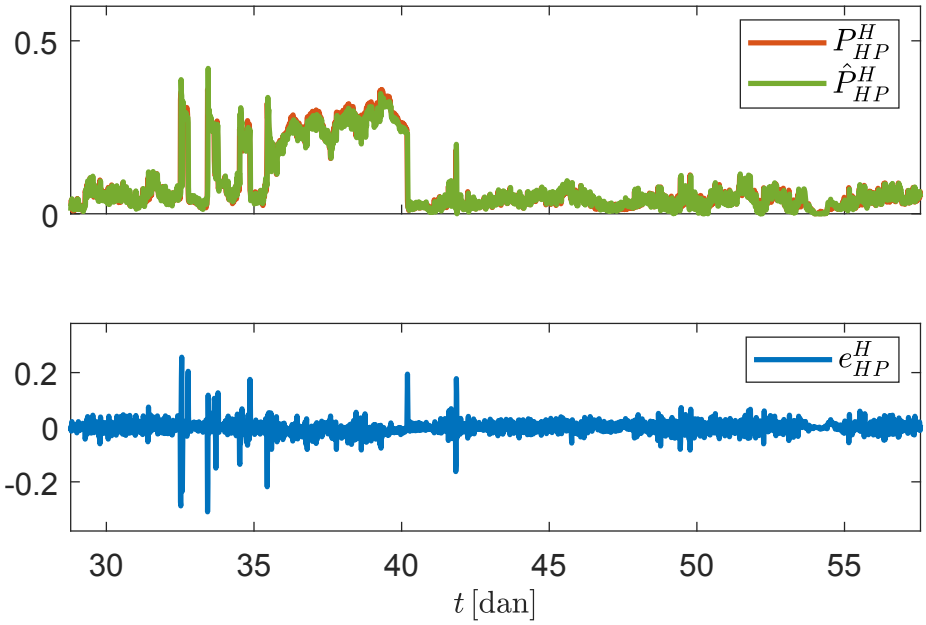
Z uporabo predlaganega identifikacijskega algoritma (slika 4.1 in razdelek 3.6.1) ter nastavitvenih parametrov iz tabele 4.2 smo pridobili modele za vsak izhod $\hat{P}_{HVAC}$ in $\hat{P}_{HP}$. Na slikah 4.8 in 4.9 je predstavljena primerjava med realnim in napovedanim izhodom za dva koraka naprej. Predstavljena sta dejanski in napovedani izhod ter razliko med njima. Izračunali smo tudi povprečje kvadratov pogreška MSE, ki iznaša $0.64 \cdot 10^{-3}$ za e^H_{HP} in $0.95 \cdot 10^{-3}$ za e^H_{HVAC}. Iz



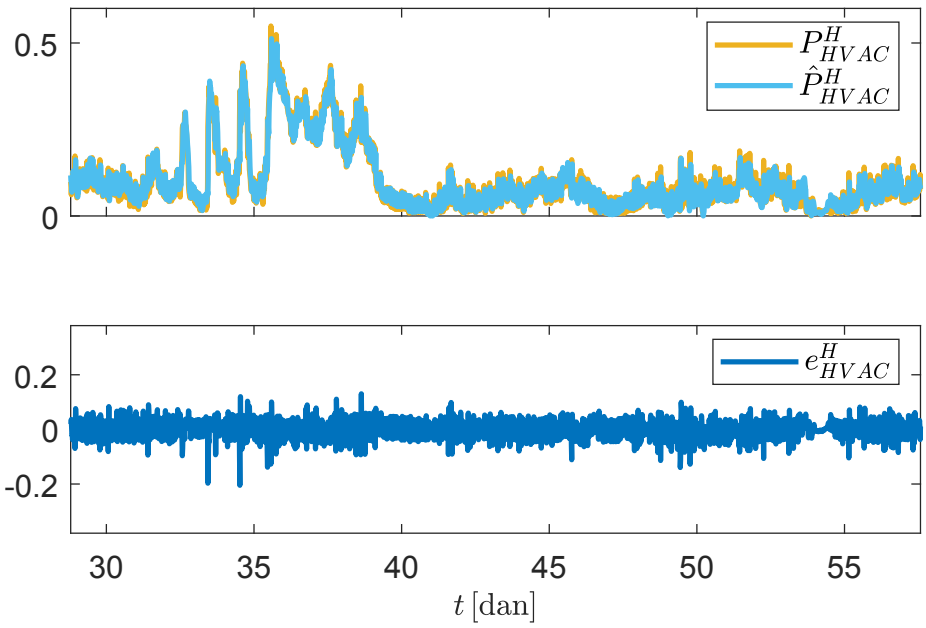
Slika 4.7: Prikaz dejanskih vrednosti zunanje temperature (zgornja slika), skupna poraba moči P_{HVAC} (srednja slika) in proizvodna moč P_{HP} (spodnja slika).

slik in numerične analize lahko sklepamo, da sta pridobljena modela $\hat{P}_{HVAC}$ in $\hat{P}_{HP}$ dober približek dejanskih izhodov.

Pridobivanje modelov predstavlja prvi korak pri optimizaciji in reševanju zgoraj opisanega problema. Pridobljena modela služita za napoved/oceno porabe in proizvodnje hladilne moči, ki neposredno vpliva na preklapljanje HA. Z 2-koračno napovedjo (kar predstavlja 30 min naprej) lahko rešimo zgoraj opisani problem (glej razdelek 4.4.1). Na sliki 4.10 je prikazan primer, kako lahko z uporabo napovedanih signalov preprečimo nezaželen izklop hladilnega agregata $HA03$. Najprej je na sliki 4.10a prikazana sekvenca delovanja agregata $HA03$ z označenimi neželenimi preklopi. Na sliki 4.10b sta prikazana oba napovedana izhoda $\hat{P}_{HP}$ in

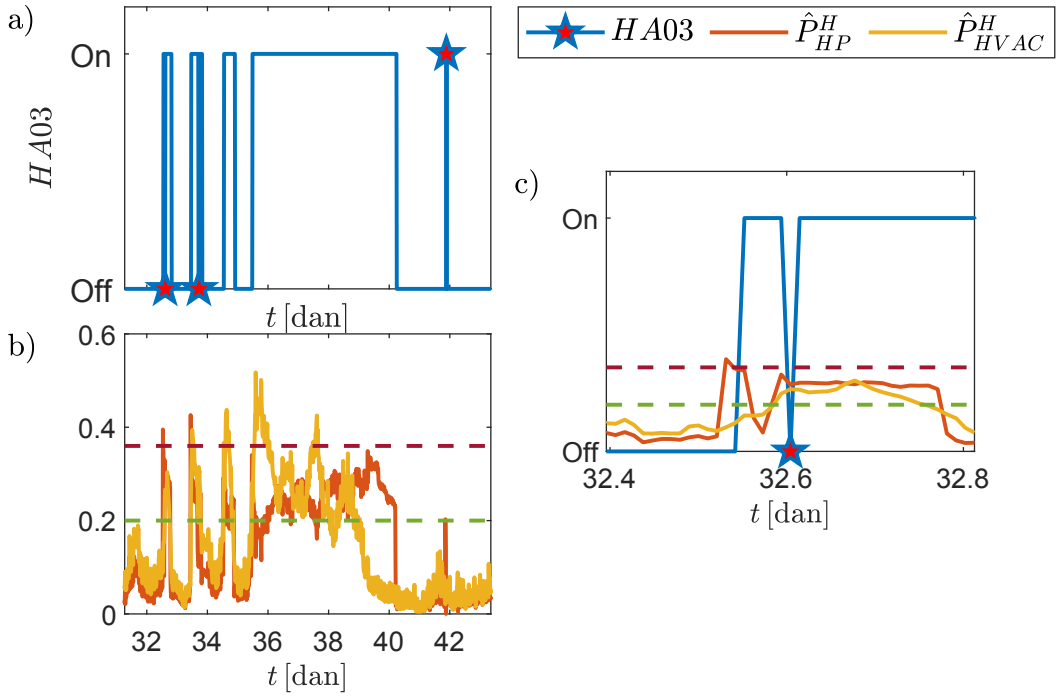


Slika 4.8: Prikaz primerjave med dejanskim P^H_{HP} in napovedanim $\hat{P}^H_{HP}$ izhodom (zgornja slika) in pogrešek med njima e^H_{HP} (spodnja slika).



Slika 4.9: Prikaz primerjave med dejanskim P^H_{HVAC} in napovedanim $\hat{P}^H_{HVAC}$ izhodom (zgornja slika) in pogrešek med njima e^H_{HVAC} (spodnja slika).

$\hat{P}_{HVAC}$. Na isti sliki sta prikazani tudi meji za vklop (rdeča črtkana črta) in izklop (zelena črtkana črta) HA. Na sliki 4.10c sta obe prejšnji sliki združeni in prikazan je podrobnejši pogled v trenutku 32.6. S slike je razvidno, da v tem trenutku pride do kratkega izklopa $HA03$, pri čemer napovedana vrednost ni manjša od spodnje meje (zelena črtkana črta), ki je pogoj za izklop HA. V tem primeru bi lahko preprečili neželen izklop, saj povzroča nepotrebno izgubo energije.



Slika 4.10: Prikaz nadzora hladilne postaje. a) Prikaz stanj delovanja $HA03$. b) Prikaz 2-koračne napovedi porabljene in hladilne moči ($\hat{P}^H_{HVAC}$ in $\hat{P}^H_{HP}$) z mejami za vklop/izklop. c) Združen in bolj podroben prikaz obeh slik a) in b).

4.5. Zaključek

Glavni namen tega poglavja je bil predstaviti samorazvijajoči se postopek za identifikacijo nelinearnih dinamičnih procesov. Osredotočili smo se na raziskavo novih mehanizmov dodajanja in odstranjevanja oblakov. Ti mehanizmi preprečujejo dodajanje novih oblakov na podlagi osamelcev in odstranijo obstoječe, ki so neaktivni ali »nosijo« malo informacije. Seveda je vse to s ciljem izboljšave uspešnosti celotnega identifikacijskega algoritma.

Predlagani algoritem smo najprej preizkusili na simuliranih podatkih kemičnega procesa Tennessee Eastman. Proces ima precej kompleksno strukturo z velikim številom vhodov in izhodov. Na podlagi podatkov smo pridobili tri modele za izbrane kazalnike proizvodne učinkovitosti (*Strošek*, *Produktivnost* in *Kakovost*). Za vse tri modele smo preizkusili različne kombinacije mehanizmov samorazvijanja in rezultate primerjali z dobro uveljavljeno metodo eFuMo in z nevronskimi mrežami. Izkazalo se je, da lahko z novimi mehanizmi dodajanja in odstranjevanja oblakov izboljšamo učinkovitost identifikacijskega algoritma. Je pa tudi res, da to vedno ne drži, ker včasih dobimo tudi slabše rezultate. Za vsak primer je treba optimizirati delovanje algoritma in poiskati najboljšo kombinacijo mehanizmov samorazvijanja. To si lahko privoščimo le v primeru, ko imamo na voljo vse podatke in je za to potrebna nesprotna (off-line) analiza/optimizacija leteh. V našem primeru pa želimo mehanizme generalizirati in jih narediti čimbolj splošne. Dobljene rezultate smo primerjali z drugimi metodami in lahko rečemo, da je predlagana rešitev konkurenčna tudi v primeru, če uporabimo samo prednastavljene parametre (brez dodatne optimizacije parametrov) metode.

Drugi primer, kjer smo preizkusili delovanje predlagane metode, je optimizacija učinkovitosti delovanja hladilne postaje. Zaznali smo problem nezaželenega preklapljanja hladilnih agregatov za kratek čas. To povzroči kar nekaj energijske izgube za celoten sistem. S predlaganim identifikacijskim algoritmom lahko zgradimo modele ključnih spremenljivk, ki vplivajo na vklope/izklope hladilnih agregatov. Z uporabo modelov lahko napovemo stanje teh izhodov v prihodnosti in pokazali smo, da lahko s tem preprečimo nepotreben preklop stanja.

5. Zaznavanje napak s samorazvijajočim se modelom

5.1. Uvod in pregled literature

Sodobni industrijski sistemi so tipično zelo kompleksni in zgrajeni iz različnih dinamičnih procesov. Pogosto obratujejo v različnih obratovalnih pogojih in pod spremenljivimi zunanjimi vplivi. Ne glede na to, ali gre za zelo velike in kompleksne procese ali pa za majhne in enostavne, so zahteve v industriji za varnejše in zanesljivejše delovanju vedno večje. Pri tem (inteligentne) metode spremljanja procesov (ang. *process monitoring*) igrajo pomembno vlogo pri zaznavanju in diagnostiki napak (ang. *fault detection and diagnostic*). Klasične metode spremljanja procesov temeljijo le na spremljanju posameznih (merljivih) spremenljivk, če so znotraj predhodno opredeljenih meja (ang. *limit checking*). Vendar spremljanje procesov lahko znatno izboljšamo, če upoštevamo več merjenih spremenljivk, ki so nam na voljo, in sicer v kombinaciji z naprednimi in inteligentnimi metodam.

V zadnjih nekaj desetletjih so bili raziskani različni teoretični in eksperimentalni pristopi z namenom zaznavanja in diagnosticiranja (izolacije) napak [98,122]. Pravzaprav lahko ločimo ta dva pristopa v dve kategoriji. Na področju zaznavanja napak spadajo metode, ki lahko odkrijejo, če se je pojavila napaka v sistemu, in določajo čas, v katerem se je napaka pojavila. Metode za diagnostiki napak ugotavljajo vrsto, velikost in čas zaznave napak in sledijo postopku zaznavanja napak. V literaturi se zadnja kategorija imenuje tudi izolacija napak (ang. *fault isolation*). Večkrat so metode za zaznavanje in diagnostiko napak (ang. *fault dedection and diagnosis*, FDD) združene v eno področje, ki na celovit način pokriva zgoraj opisani problem.

Naslednje pomembno področje, ki spada v okvir spremljanja procesov, je tudi upravljanje z napakami (ang. *fault management*). To pomeni, da poskušamo preprečiti nezaželene ustavitve procesa s predčasnim odkrivanjem napak. Na ta način lahko preprečimo, da sploh pride do napake ali do neželenega stanja procesa. V primerih, da se napakam ne moremo izogniti, pridejo v poštev metode, ki poskušajo procesu pomagati in zmanjšati vpliv napak (ang. *fault-tolerant*

systems). Na ta način lahko zagotovimo delovanje brez zastojev. Ponavadi se ta problem reši z redundantnim elementom v procesu, ki pa po drugi strani poveča kompleksnost procesa [121, 235] kot tudi stroške vzdrževanja.

Kot smo že omenili, je ena najbolj pogosto uporabljenih metod spremljanja procesov preverjanje limitnih vrednosti (poglavje 7 v [121]). Prednost te metode je ta, da je enostavna, zanesljiva in lahka za implementacijo. Problem nastane, ko imamo opravka s kompleksnim in velikim sistemom. Na primer, če imamo veliko število spremenljivk, ki jih nadziramo z uporabo metode limitnih vrednosti, potem se lahko zgodi, da dobimo več alarmov hkrati (ang. *alarm-shower*). V tem primeru operater ne more določiti najbolj pomembne napake in jih težko odkrije (diagnosticira). Potemtakem igrajo napredne metode pomembno vlogo v reševanju tovrstnih problemov.

Na splošno morajo napredne metode spremljanja procesov izpolnjevati naslednje pogoje [121]:

(i) zgodnje zaznavanje napak;

(ii) diagnostika napak v določenem delu procesa (napaka senzorja, aktuatorja itn.);

(iii) zaznavanje napak v zaprti zanki;

(iv) spremljanje procesa v času prehodnega pojava.

Metode procesnega nadzora in zaznavanje napak lahko razdelimo v tri večje skupine [245–247, 264]:

1. metode na podlagi matematičnih (fizikalnih) modelov procesa,
2. statistične metode in
3. metode na podlagi podatkov (ang. *data-based methods*).

V prvo skupino spadajo metode, ki temeljijo na matematičnem modelu in zahtevajo *a pripri* znanje o fizikalnem ozadju procesa (v obliki diferencialnih ali diferenčnih enačb, v prostoru stanj itn.). Te metode so bile uspešno uporabljene na različnih industrijskih aplikacijah [53, 95, 98, 109, 113, 120, 143, 247]. Poleg

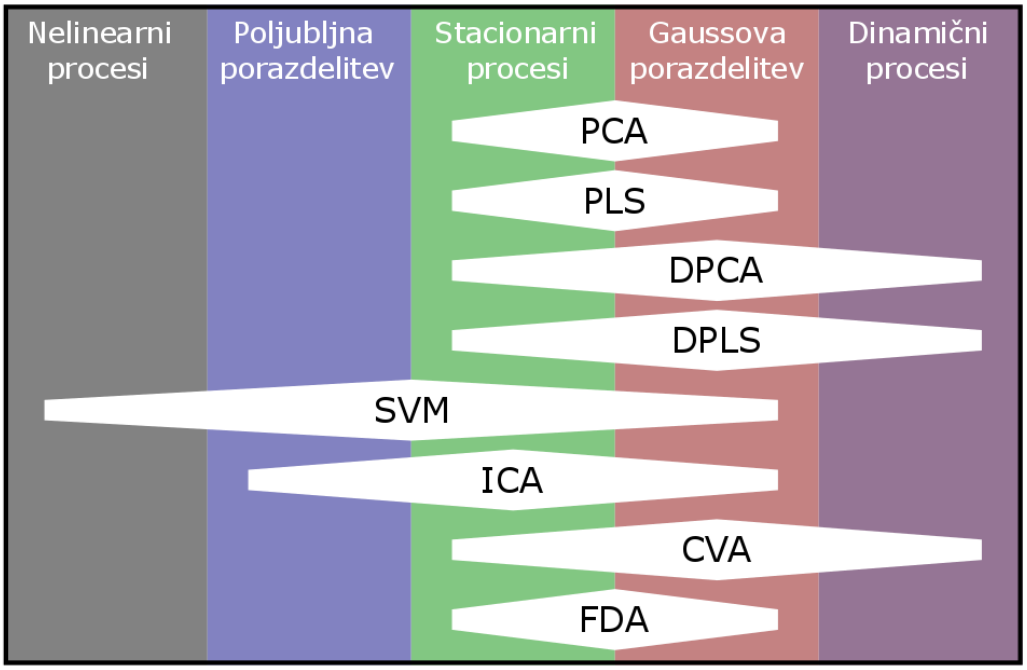
njihove široke uporabnosti imajo te metode tudi svoje slabosti. Omejene so na linearne modele in le v nekaterih primerih na specifične nelinearne modele (z uporabo linearne aproksimacije). Poleg tega pa ne upoštevajo vpliva motenj, lezenja parametrov, uporabljajo le *a priori* znanje o napakah itn.

Druga skupina metod temelji na statističnih analizah podatkov procesa [202]. Te metode uporabljajo shranjene podatke za pridobitev znanja o procesu v normalnem delovanju in potem lahko zaznajo odstopanja od normalnega stanja. Najbolj osnovni metodi v tej skupini sta analiza glavnih komponent (PCA, ang. *principal component analysis*) [64, 97, 142, 144, 162] in metoda delnih najmanjših kvadratov (PLS, ang. *partial least squares*) [64, 160, 256, 275]. V zadnjem času je metoda neodvisnih komponent (ICA, ang. *independent component analysis*) [243, 274] prejela veliko pozornosti in izkazala praktično uporabnost v številnih aplikacijah. Statistične metode so skratka uporabne v primeru, ko industrijski proces obratuje v stacionarnem stanju, in niso primerne za kompleksne dinamične procese s spremenljivimi pogoji. Nekaj teh statističnih metod je prikazanih na sliki 5.1. Opazimo, da so te metode v glavnem namenjene za stacionarne procese z Gaussovo porazdelitvijo. Izjema sta metodi DPCA (ang. *dynamic principal component analysys*) in DPLS (ang. *dynamic partial least squares*), ki predstavljajo razširitev zgoraj omenjenih metod za dinamične procese. Tudi metoda CVA (ang. *canonical variate analysis*) je namenjena zaznavanju napak na stacionarnih kot tudi na dinamičnih procesih.

Na tem mestu je potrebno omeniti metode, ki uporabljajo tok podatkov, tako kot tudi v nadaljevanju predstavljena metoda. Pravzaprav bomo večjo pozornost posvetili metodam s samorazvijajočo se strukturo in adaptivnimi parametri. V [223] so avtorji predstavili pristop na podlagi residualov (ostankov), kjer je originalni signal transformiran v prostor modela z identifikacijo večdimenzionalnih povezav. Arhitektura modela je sestavljena iz lokalno linearnih modelov, kjer so »blage« nelinearnosti predstavljene z generičnimi modeli Box-Cox in bolj kompleksna področja z modeli Takagi-Sugeno. Metoda za zaznavanje napak na podlagi samorazvijajočega se deljenja prostora na Gaussove roje je predstavljena v [157]. Zaznavanje napak je izvedeno z adaptivnim postopkom in vsaka zaznana napaka se vključi v klasifikator. Ko se naslednjič pojavi ista napaka, jo sistem zazna samodejno. V [38] je predstavljena metoda za zaznavanje napak, ki temelji na

načelu TEDA (ang. *typicality and eccentricity data analytics*) [18]. TEDA je zasnovana na Čebiševi neenakosti in predstavlja mero za zaznavanje anomalij v podatkih.

Predlagana metoda za zaznavanje napak, ki bo razložena v nadaljevanju, temelji na samorazvijajočem se mehkem sistemu AnYa [24]. Predhodno podobno delo je bilo predstavljeno v [75] in temelji na rekurzivnem izračunu gostote podatkov (ang. *recursive density estimation*, RDE). Slabost te metode je ta, da uporablja časovne pragove za zaznavanje napak in normalnega delovanja. To je lahko zelo odvisno od časovne konstante procesa.



Slika 5.1: Prikaz nekaj uveljavljenih statističnih metod za spremljanje procesov [264].

5.2. Spremljanje procesov in zaznavanje napak s samorazvijajočim se modelom

Metoda za zaznavanje napak, ki bo razložena v nadaljevanju, je zasnovana na osnovi samorazvijajočim se modelom in temelji na delnem izračunu lokalnih gostot (ang. *partial density calculation*). Osnova za to metodo so enačbe (2.4), (2.11), (2.14) in (2.15). Enačba (2.4) predstavlja mehki model AnYa, kjer pogojni del

vsebuje mehanizme za samorazvijanja strukture, funkcija posledičnega dela pa je pravzaprav klasifikacija podatkov z namenom zaznavanja napak. Ostale tri enačbe (2.11), (2.14) in (2.15) razvijemo po komponentah in dobimo naslednje[1]:

$$\begin{aligned}
\|\mathbf{x}_k - \boldsymbol{\mu}_k^i\|^2 &= (\mathbf{x}_k(1) - \boldsymbol{\mu}_k^i(1))^2 + \cdots + (\mathbf{x}_k(m) - \boldsymbol{\mu}_k^i(m))^2 \\
\sigma_k^i &= \frac{1}{M^i}\sum_{j=1}^{M^i}\|\mathbf{x}_j^i\|^2 = \frac{1}{M^i}\sum_{j=1}^{M^i}\left[(\mathbf{x}_j^i(1))^2 + \cdots + (\mathbf{x}_j^i(m))^2\right] = \\
&= \frac{1}{M^i}\sum_{j=1}^{M^i}(\mathbf{x}_j^i(1))^2 + \cdots + \frac{1}{M^i}\sum_{j=1}^{M^i}(\mathbf{x}_j^i(m))^2 = \\
&= \rho_k^i(1) + \cdots + \rho_k^i(m) \\
\|\boldsymbol{\mu}_k^i\|^2 &= \boldsymbol{\mu}_k^i(1)^2 + \cdots + \boldsymbol{\mu}_k^i(m)^2
\end{aligned}$$

kjer je m dimenzija vektorjev $\mathbf{x}_k$ in $\boldsymbol{\mu}$, in ρ_k^i povprečje kvadratov po komponentah vhodnega vektorja $\mathbf{x}_k$. Enačbo (2.11) lahko potem zapišemo z novo obliko:

$$\begin{aligned}
\gamma_k^i =& \frac{1}{1 + [(\mathbf{x}_k(1) - \boldsymbol{\mu}_k^i(1))^2 + \cdots + (\mathbf{x}_k(m) - \boldsymbol{\mu}_k^i(m))^2]} \cdots \\
\cdots& \frac{}{+ [\rho_k^i(1) + \cdots + \rho_k^i(m)] + [\boldsymbol{\mu}_k^i(1)^2 + \cdots + \boldsymbol{\mu}_k^i(m)^2]} = \\
=& \frac{1}{1 + [(\mathbf{x}_k(1) - \boldsymbol{\mu}_k^i(1))^2 + \rho_k^i(1) - \boldsymbol{\mu}_k^i(1)^2] +} \cdots \\
\cdots& \frac{}{+ [(\mathbf{x}_k(m) - \boldsymbol{\mu}_k^i(m))^2 + \rho_k^i(m) - \boldsymbol{\mu}_k^i(m)^2]} = \\
=& \frac{1}{1 + p_k^i(1) + \cdots + p_k^i(m)} = \frac{1}{1 + \sum_{j=1}^m p_k^i(j)} \\
& i = 1, \ldots, c
\end{aligned} \tag{5.1}$$

kjer je končni izračun gostote odvisen od vsote prispevkov posameznih komponent $p_k^i(j), \quad j = 1, \ldots, m$. Vrednost posamezne komponente $p_k^i(j)$ je odvisna od $\mathbf{x}_k$, $\boldsymbol{\mu}_k^i$ in ρ_k^i, kjer se ρ_k^i izračuna z naslednjim izrazom:

$$\rho_k^i = \frac{M^i - 1}{M^i}\rho_{k-1}^i + \frac{1}{M^i}\mathbf{x}_k^2, \quad \rho_0^i = \mathbf{x}_k^2 \tag{5.2}$$

kjer je $\mathbf{x}_k^2 = [x(1)^2, x(2)^2, \ldots, x(m)^2]$.

[1]Z oklepaji označujemo posamezno komponento vektorja, npr. $\mathbf{x}_k(1)$ označuje prvi element vektorja $\mathbf{x}_k$ v k-tem koraku.

Končna oblika delne (modificirane) lokalne gostote je definirana z naslednjim izrazom:

$$\gamma^i_{\delta k} = \frac{1}{1 + \sum_{j=1}^{m} \mathbf{q}(j) \cdot p^i_k(j)} = \frac{1}{1 + \mathbf{q}^T \mathbf{p}^i_k}, \quad i = 1, \ldots, c \tag{5.3}$$

kjer je $\mathbf{q}$ vektor binarnih vrednosti $\mathbf{q}(j) \in \{0,\ 1\},\ j = 1, \ldots m$, indeks δ pa označuje število izbranih komponent pri izračunu delne lokalne gostote. Če želimo, da določena komponenta vpliva na izračun (5.3), potem je vrednost spremenljivke enaka $q(j) = 1$, sicer je enaka $q(j) = 0$. Število vplivnih komponent δ je predhodno definiran parameter. S tem upoštevamo samo najbolj vplivne komponente pri izračunu lokalne gostote.

Mehanizem samorazvijanja za zaznavanje novih stanj (dodajanje novih oblakov) je enak, kot je bilo razloženo v razdelku 3.2.2 s tem, da se lokalna gostota izračuna z enačbo (5.3).

5.2.1 Zaznavanje napak

Zaznavanje napak temelji na samorazvijajoči se metodi iz prejšnjega podpoglavja in je sestavljeno iz dveh faz: *učenje* in *vrednotenje*. Najprej na podlagi podatkov in s samorazvijajočim se modelom (glej algoritem 2) zaznamo vse oblake, ki predstavljajo normalno delovanje procesa in področje napak. V drugi fazi imamo podatke, ki opisujejo normalno delovanje in vsebujejo napake. V tej fazi se preveri uspešnost zaznavanja napak. Obe fazi sta razloženi v nadaljevanju.

Faza učenja

Namen faze učenja je pridobiti informacije o procesu v normalnem stanju delovanja (brez napake) in stanju, ko je prisotna napaka. Za ta namen potrebujemo dva seta podatkov, ki opisujeta ti dve stanji procesa. Uporabimo isti postopek samorazvijanja (algoritem 2) za odkrivanje oblakov brez ($X^i \in \{\boldsymbol{\mu}^i_k, \sigma^i_k, \boldsymbol{\rho}^i_k, F = 0\}$) in z napako ($X^i \in \{\boldsymbol{\mu}^i_k, \sigma^i_k, \boldsymbol{\rho}^i_k, F = 1\}$). Zapis $F = 0$ označuje pripadnost oblaka na področju brez napake in $F = 1$ je za oblake, ki opisujejo napako.

Algoritem 2 Psevdo koda mehanizma samorazvijanja.

Inicializacija: γ_{max} in δ.
repeat
  Vhod: $\mathbf{x}_k = [y_{1k}, \ldots, y_{n_a k}, u_{1k}, \ldots, u_{n_b k}]$
  if k==1 **then**
    Inicializacija: $c = 1$
    Inicializacija: $\boldsymbol{\mu}_1^i = \mathbf{x}_1$, $\sigma_1^i = \|\mathbf{x}_1\|^2$, in $\boldsymbol{\rho}_1^i = \mathbf{x}_1^2$.
    Definiraj: $X_1^i \in \{\boldsymbol{\mu}_1^i, \sigma_1^i, \boldsymbol{\rho}_1^i, F \in [0,1]\}$
  else
    Izračunaj: $\gamma_{\delta k}^i, \quad i = 1, \ldots, c.$
    if $\max_i(\gamma_{\delta k}^i) < \gamma_{max}$ **then**
      Zaznan je nov oblak.
      Povečaj: c.
      Inicializacija: $\boldsymbol{\mu}_1^i = \mathbf{x}_k$, $\sigma_1^i = \|\mathbf{x}_k\|^2$, in $\boldsymbol{\rho}_1^i = \mathbf{x}_k^2$.
      Definiraj: $X_k^i \in \{\boldsymbol{\mu}_1^i, \sigma_1^i, \boldsymbol{\rho}_1^i, F \in [0,1]\}$
    else
      Poveži podatek $\mathbf{x}_k$ z X^i na osnovi $(max_i \gamma_{\delta k}^i)$
      Posodobimo parametre $\boldsymbol{\mu}_k^i$, σ_k^i, in $\boldsymbol{\rho}_k^i$ izbranega oblaka X^i
    end if
  end if
until Konec toka podatkov.

Faza vrednotenja

V fazi vrednotenja bomo na mešanih podatkih, ki vsebujejo normalno delovanje in napako, preverili učinkovitost predlagane metode. Znanje (oblake) pridobljeno v prejšnji fazi bomo uporabili za detekcijo napak. Za vsak podatek $\mathbf{x}_k$ iz testnega seta podatkov izračunamo maksimalno delno gostoto oblakov brez napake $\max_i \gamma_{\delta k}^i(X_{F=0}^i)$ in z napako $\max_i \gamma_{\delta k}^i(X_{F=1}^i)$. Z naslednjo funkcijo, ki je pravzaprav del posledičnega dela enačbe (2.4), lahko vsak podatek razvrstimo kot normalno delovanje ali napako:

$$Napaka = \begin{cases} 1, & \max_i \gamma_{\delta k}^i(X_{F=0}^i) < \max_i \gamma_{\delta k}^i(X_{F=1}^i) \\ 0, & \text{sicer} \end{cases} \tag{5.4}$$

kjer 1 pomeni, da smo odkrili napako, medtem ko 0 pomeni normalno obratovanje procesa.

5.3. Proces Tennessee Eastman

Proces Tennessee Eastman smo že razložili v podpoglavju 4.3. Na tem mestu bomo uporabili isti model procesa za preverjanje učinkovitosti samorazvijajoče se metode za zaznavanje napak (razdelek 5.2.1). Proces TE ima veliko število merjenih in manipulativnih spremenljivk, kar odpira številne možnosti za preverjanje različnih napak in metod na procesu [67, 190, 205, 207, 259].

5.3.1 Opis problema

V [91] so avtorji definirali šest različnih načinov obratovanja procesa na podlagi različnih razmerij produkta in/ali produktnih mešanic, zahtevanih s strani tržišča ali pa zaradi nekaterih omejitev na procesu. Prav tako so avtorji predlagali 20 procesnih motenj/napak, ki posnemajo realno obnašanje procesa TE. Proces TE je zelo nestabilen sistem in brez nizkonivojske regulacije [34,210] zelo hitro doseže kritično mejo obratovanja in sistem se nemudoma ustavi.

V tem eksperimentu smo uporabili simulirane podatke procesa TE [70] in izbrali tri različne napake iz nabora napak definiranih v [91]. Tabela 5.1 prikazuje opis in tip napak, uporabljenih za preverjanje predlagane metode. Dobljene rezultate bomo primerjali z rezultati drugih metod, predstavljenih v [263], kot so FDA [71, 110] (ang. *fisher discriminate analysis*), PCA [121, 246] (ang. *principal component analysis*), in ICA [131, 153, 161] (ang. *independent component analysis*).

Tabela 5.1: Opis napak na procesu TE [263].

Št. napake	**Procesna spremenljivka**	**Tip napake**
$IDV(1)$	Pretočno razmerje A/C, reagent B konstanten	Stopnica
$IDV(2)$	reagent B, Pretočno razmerje A/C konstantno	Stopnica
$IDV(16)$	Neznana	Neznana

5.3.2 Rezultati

V tem razdelku bomo prikazali rezultate uspešnosti zaznavanja napak s samorazvijajočim se modelom na podlagi delnih lokalnih gostot. Z namenom primerjave z drugimi metodami bomo uporabili metriko (kot je bila predlagana v [263]) FDR (ang. *fault detection rate*):

$$FDR = \frac{TP}{vsi\ podatki\ (f = 1)} 100\,[\%] \tag{5.5}$$

kjer TP (ang. *true positives*) označuje število pravilno detektiranih napak in $f = 1$ vse podatke, ki vsebujejo napako. Z namenom optimizacije parametrov predlagane metode smo uporabili še splošnejšo metriko ACC (ang. *accuracy*), ki upošteva tudi pravilno razvrščene podatke pri normalnem obratovanju procesa:

$$ACC = \frac{TP + TN}{vsi\ podatki} 100\,[\%] \tag{5.6}$$

kjer TN označuje število pravilno detektiranih podatkov normalnega obratovanja procesa.

Na sliki 5.2 je prikazana odvisnost uspešnosti algoritma (vrednost ACC) pri različnih nastavitvah parametrov γ_{max} in δ za obravnavane napake IDV(1), IDV(2) in IDV(16). Na sliki 5.2 temnordeča barva označuje področja z največjo uspešnostjo zaznavanja napak ($ACC \approx 100\,\%$), modra barva pa označuje področja s slabšo uspešnostjo ($ACC \approx 0\,\%$). Lahko razberemo, da je za isto uspešnost pri zmanjševanjem števila komponent δ potrebno povečevati parameter γ_{max}. Vidimo tudi, da je odvisnost uspešnosti algoritma od parametrov podobna pri vseh napakah. Optimalne vrednosti parametrov za prvi dve napaki so enake, in sicer $\delta = 12$ ter $\gamma_{max} = 0.96$. Za tretjo napako IDV(16) pa so optimalne vrednosti parametrov enake $\delta = 27$ in $\gamma_{max} = 0.3$. V nadaljevanju bo predlagana metoda z optimalnimi parametri označena z $CB_{optimal}$. Privzeti parametri (ang. *default parameters*) metode ($CB_{default}$) so nastavljeni z naslednjimi vrednostmi $\delta = m = 33$ in $\gamma_{max} = 0.75$, kjer so upoštevane vse komponente.

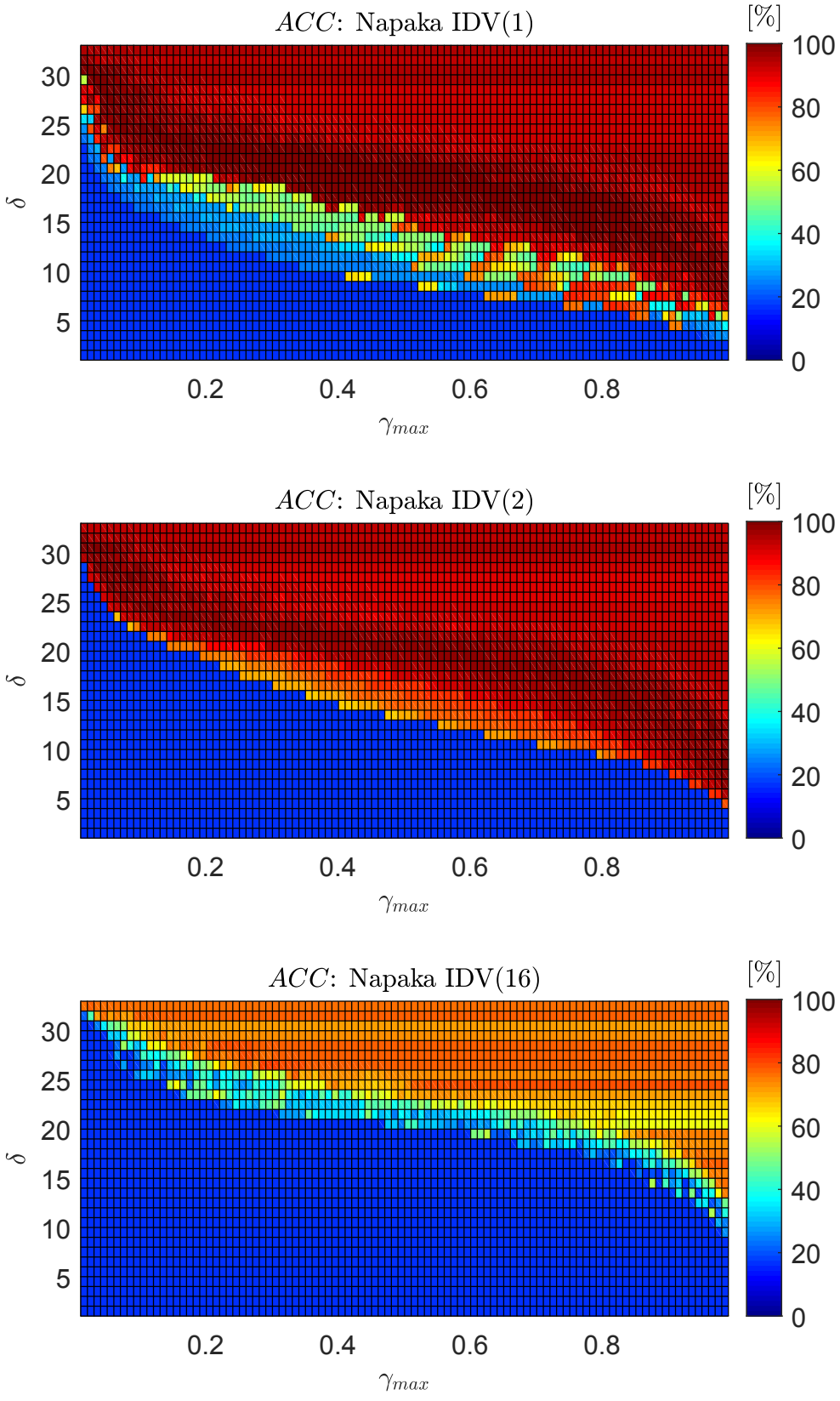
Na sliki 5.3 so prikazani primeri zaznavanja napak za vse tri napake: IDV(1), IDV(2) in IDV(16). Na prvem, tretjem in petem grafu na sliki 5.3 so prikazani signali maksimalnih gostot $\max_i \gamma^i_{\delta k}(X^i_{F=0})$ (modra barva) in $\max_i \gamma^i_{\delta k}(X^i_{F=1})$ (rdeča barva) za vsako napako. Ob upoštevanju enačbe (5.4) so pa na drugem,

četrtem in šestem grafu prikazani signali zaznavanja napak $Napaka$ (oranžna barva). Vrednost 1 pomeni, da smo zaznali napako, medtem ko vrednost 0 pomeni, da nismo zaznali nobene napake. Iz slike 5.3 je razvidno, da smo v primeru napak IDV(1) in IDV(2) zelo uspešno zaznali trenutke, ko je dejanska napaka prisotna. V primeru tretje napake IDV(16) pa je iz slike 5.3 razvidno, da večkrat napake ne zaznamo, kot tudi, da zaznamo napake, ko je proces v normalnem delovanju.

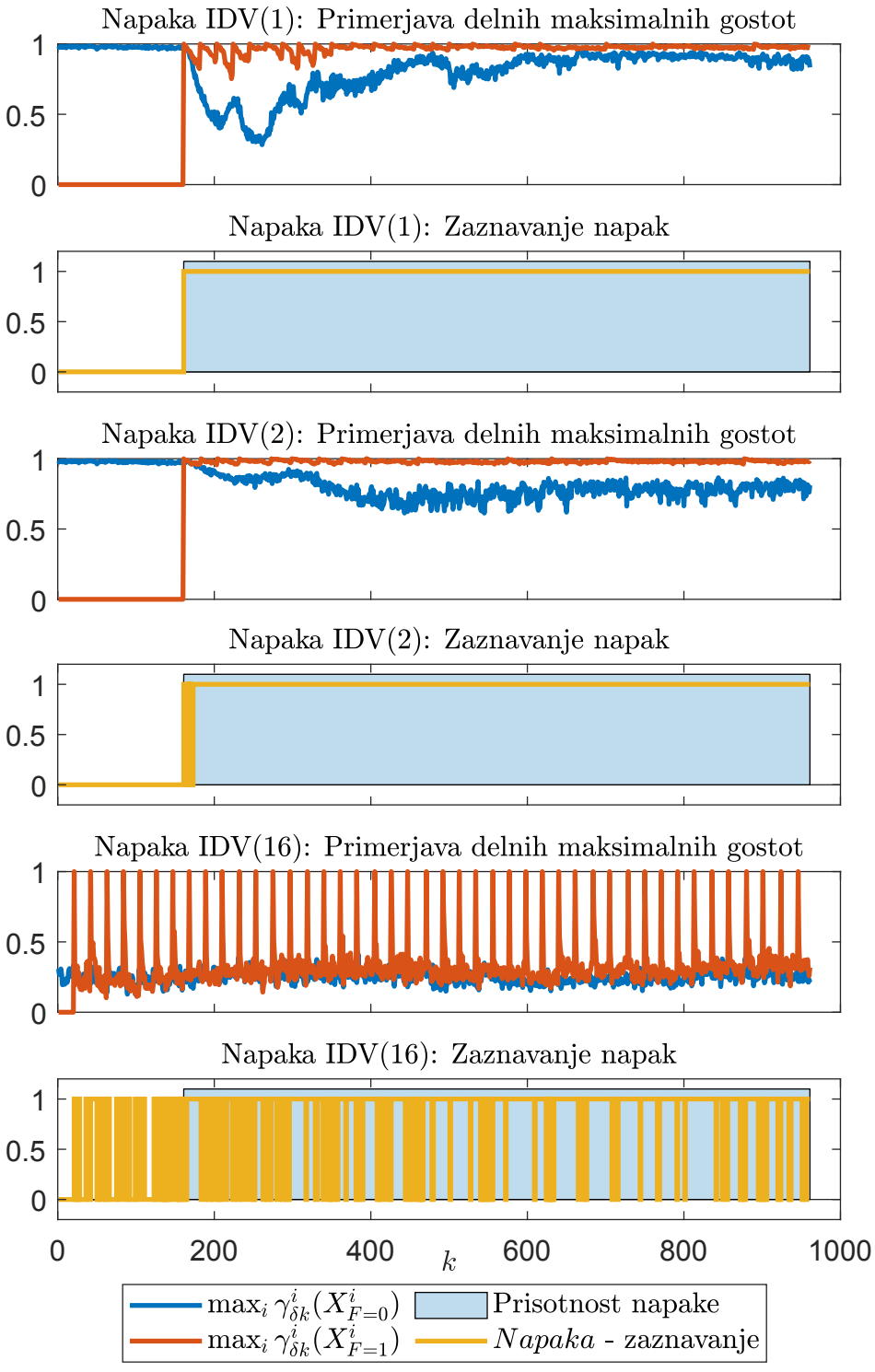
Nazadnje smo tudi kvantitativno primerjali (z metriko FDR) rezultate predlagane metode z drugimi metodami (glej tabelo 5.2). Iz tabele je razvidno, da ima v primeru napake IDV(1) predlagana metoda $CB_{optimal}$ 100 % razpoznavanje te napake in za napako IDV(2) ima najboljši rezultat. V primeru neznane napake IDV(16) pa je $CB_{optimal}$ na drugem mestu. Pričakovano pa $CB_{default}$ nekoliko zaostaja pri vseh primerih, vendar vseeno izkazuje zelo dobre rezultate.

Tabela 5.2: Primerjava rezultatov FDRs [%] iz [263] s predlagano metodo ($CB_{optimal}$ in $CB_{default}$).

	IDV(1)	IDV(2)	IDV(16)
FDA	100.00	98.75	83.25
PCA	99.88	98.75	55.25
ICA	100.00	98.25	92.38
$CB_{optimal}$	100.00	99.38	84.52
$CB_{default}$	97.13	95.26	74.91



Slika 5.2: Prikaz vpliva parametrov γ_{max} in δ na vrednost ACC.



Slika 5.3: Prikaz zaznavanja napak. Primerjava med maksimalno gostoto oblakov brez napake (modra črta) in z napako (rdeča črta) (zgornja slika) ter prikaz dejanskega zaznavanja napak (spodnja slika).

5.4. Zaključek

V tem poglavju smo predstavili metodo za zaznavanje napak na osnovi samorazvijajočim se modelom. Metoda je namenjena zaznavanju napak na dinamičnih in nelinearnih procesih. Algoritem temelji na oblakih podatkov, ki jih razdelimo v dve skupini: prva skupina predstavlja oblake za normalno obratovanje procesa in druga skupina oblake za delovanje s prisotno napako. Z vsakim novim sprejetim podatkom izračunamo lokalne gostote za obe skupini in določimo, ali podatek predstavlja normalno delovanje ali napako.

Algoritem smo preskusili na znanem procesu Tennessee Eastman in preverili tri različne napake. Rezultate algoritma smo primerjali z že uveljavljenimi metodami na tem področju (FDA, PCA in ICA). Izkazalo se je, da optimalno nastavljeni parametri algoritma dosegajo enake oziroma boljše rezultate v primerjavi z drugimi metodami. Tudi privzeti parametri dajejo primerljive rezultate. Zaključimo lahko, da z uporabo predstavljene metode lahko na enostaven in računsko hiter način zaznamo napake v delovanju procesa.

6. Zaznavanje manevrov pri voznikih

6.1. Uvod in pregled literature

Avtonomni inteligentni agenti morajo v določenih okoliščinah razumeti, kako delujejo tudi ostali agenti v sistemu. Prepoznavanje (inteligentnega) obnašanja (ang. *intelligent behaviour*) je ključen del celovitega inteligentnega sistema. Dandanes v času velikih podatkov (ang. *big data*) in računalništva v oblaku (ang. *cloud computing*) ima vse več inteligentnih sistemov to lastnost, da prepoznajo obnašanje ostalih agentov v sistemu. S tem namenom je glavni fokus raziskav in aplikacij na tem področju usmerjen na več pristopov, kot so: prepoznavanje načrtov (ang. *recognition of plans*) [30, 63], modeliranje uporabnikov (ang. *user modelling*) [258], modeliranje agentov (ang. *agents modelling*) [233], modeliranje nasprotnikov (ang. *opponent modelling*) [115, 152], prepoznavanje namena (ang. *intention recognition*) [236], prepoznavanje obnašanja (ang. *behavioural recognition*) [54, 114] itn. Prav tako so bile na tem področju razvite različne aplikacije, ki so povezane z osebnimi (inteligentnimi) asistenti, elektronskim poslovanjem in modeliranjem voznikov [68, 140], kar bo predmet raziskovanja v tem poglavju.

Znotraj področja modeliranja voznikovega obnašanja[1] spada več podpodročij, kot so zaznavanje manevrov, zaznavanje psihofizičnega stanja voznika (npr. utrujenost, zaspanost, zehanje itn.), zaznavanje deviacij, gradnja kognitivnih modelov, zaznavanje načina vožnje (npr. športna, agresivna, normalna vožnja itd.) in podobno.

V nadaljevanju tega poglavja se bomo osredotočili na področje zaznavanja voznikovih manevrov. Dandanes so tako imenovani napredni sistemi za pomoč voznikov (ang. *advanced driver assistance system* – ADAS) [36] del skoraj vsakega osebnega vozila. Glavni cilj teh sistemov je povečati varnost oziroma olajšati delo voznika. Komponente, ki so del sistemov ADAS, ponujajo različne funkcionalnosti, kot so opozorilni sistem za spremembo pasu ali zaznavanje zaspanosti voznika [50]. Glavna cilja zaznavanja pri sistemih ADAS sta kaj voznik

[1] V doktorski disertaciji obravnavamo voznike osebnih avtomobilov.

počne in kaj namerava početi. Tovrstno zaznavanje je lahko izvedeno na več načinov, in sicer z neposrednim opazovanjem voznika (npr. spremljanje premikanja oči [164, 227, 270], zaznavanjem zehanja [3, 240] z uporabo kamere) ali s posrednim opazovanjem (npr. z uporabo inercialnih senzorjev v pametnem telefonu [58, 59]). Ko enkrat zaznamo voznikovo obnašanje, lahko v določenem trenutku inteligentni sistem prevzame vlogo kopilota [267].

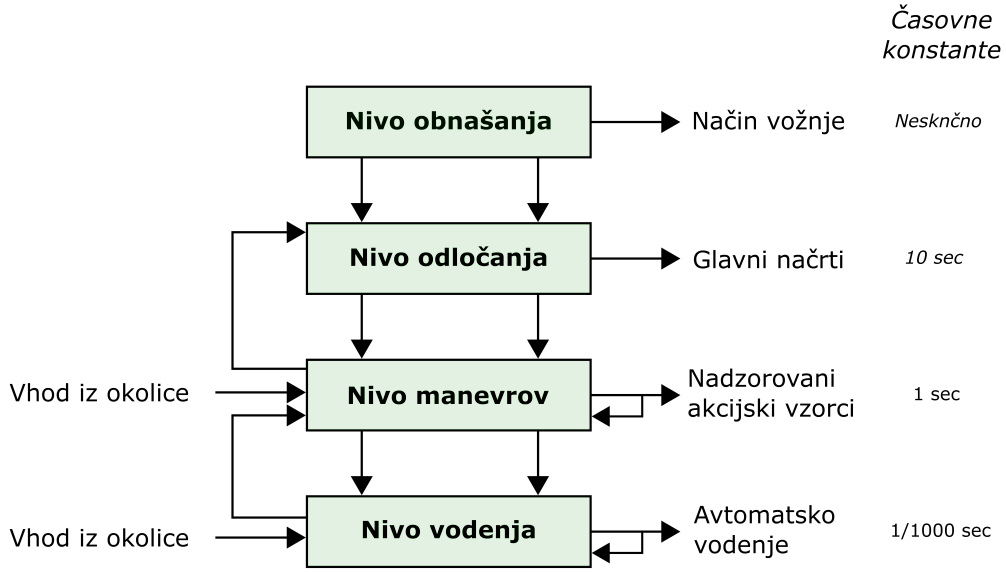
6.2. Opis problema

Modeliranje voznikovega obnašanja (ang. *driver behaviour modelling* – DBM) je raziskovalno področje, ki je že obravnavano v različnih literaturah [180, 185, 219, 225, 242]. Nenazadnje je bila v [241] predstavljena metoda strojnega učenja z namenom zaznavanja in klasifikacije različnih manevrov voznika z različnimi senzorji, ki so na voljo v avtomobilih. V [4] so avtorji naredili raziskavo trenutnega razvoja in zmogljivosti obstoječih aplikacij, ter poudarili raziskovalne izzive in nadaljnje usmeritve. Druga raziskava [84] se osredotoča na področje napovedovanja voznikovega obnašanja s posebnim poudarkom na taktičnih manevrih. Glavni cilj je z uporabo napovedanega obnašanja voznika napovedati trajektorijo vozila v realnem času in s tem preprečiti neželene nevarnosti ali neprijetne situacije. Številne (matematične) identifikacijske metode in metode modeliranja voznikovega obnašanja so predstavljene v [255]. Pridobljeni modeli so potem integrirani v napredni sistem za pomoč voznikom (ADAS) za evaluacijo in verifikacijo voznikovega obnašanja.

Poznamo veliko raznolikih komponent in faz, ki so del modela obnašanja voznika (DBM). V tem smislu lahko modele DBM razdelimo na predikcijske in reakcijske modele. Razlika je v tem, da predikcijski model zaznava obnašanje voznika v realnem času, medtem ko reakcijski model zazna obnašanje voznika šele potem, ko je manever končan. Iz tega tudi izhaja, da načrtovanje predikcijskih modelov predstavlja težje opravilo. V pričujoči doktorski disertaciji bomo predstavili model, ki se lahko uvrsti v obe skupini (kot predikcijski in kot reakcijski model). Druga klasifikacija modelov DBM je na podlagi hierarhije vodenja, ki se deli na operacijsko, taktično in strateško vodenje.

V [179] so predstavili hierarhično strukturo s tremi nivoji vodenja avtomobila, ki so razmeroma ohlapno povezani, to so strateški nivo (ali nivo odločanja), nivo

manevrov in nivo vodenja. Strateški nivo vsebuje planiranje in izbiro optimalne poti in je ta nivo zelo malo vključen v celoten proces vožnje avtomobila. Na nivoju manevrov je voznik v neposredni interakciji s trenutnimi razmerami na cesti (upoštevajoč prometne predpise in znake). Nivo vodenja pa se nanaša na bazične (atomske) funkcije avtomobila. V [192] so avtorji dodali še četrti nivo, ki se nanaša na obnašanje (slog vožnje) voznika. Ta nivo se nanaša na spretnosti in karakteristike voznika in ima najvišjo prioriteto, pri čemer ta nivo močno vpliva tudi na ostale nivoje. Slika 6.1 prikazuje hierarhično strukturo vseh štirih nivojev v procesu vožnje avtomobila. Na področju modeliranja voznikovega obnašanja



Slika 6.1: Prikaz hierarhičnih nivojev vožnje avtomobila [192].

(DBM) igrajo pomembno vlogo senzorji, ki jih izberemo kot vhode v naš sistem. Uporabimo lahko meritve, ki jih pridobimo iz omrežja CAN (ang. *controller area network*), lahko pa uporabimo tudi dodatne senzorje, kot so radarji, GPS (ang. *global positioning system*), pospeškometri, žiroskopi itn. V pričujoči doktorski disertaciji smo se osredotočili le na podatke, ki jih lahko pridobimo preko omrežja CAN.

Glede na kompleksnost različnih človeških aktivnosti (manevrov) in različnih nivojev abstrakcije, s katerimi jih lahko predstavimo, je treba vzpostaviti transparenten sistem za njihovo razpoznavanje in razvrščanje. Različni pristopi

razvrščanja teh aktivnosti so predstavljeni v [60, 181], vendar večina teh pristopov temelji na podobni ideji, kjer so aktivnosti razvrščene na podlagi trajanja, zahtevnosti in kompleksnosti akcije. V tem delu doktorske disertacije bomo predstavili nov način prepoznavanja voznikovih akcij (manevrov) na podlagi različnih hierarhičnih nivojev, kjer je vsak višji nivo sestavljen iz posameznih komponent nižjih.

Dobljeni modeli voznikovega obnašanja so lahko uporabni na področju usposabljanja voznikov in varčevanja z energijo ter zlasti za sisteme za pomoč voznikom (ADAS). Pravzaprav je modeliranje zelo pomembno za sam razvoj teh sistemov.

6.3. Zaznavanje manevrov pri voznikih

Kot smo že nakazali v prejšnjem podpoglavju, je naša glavna ideja zaznati različne manevre z uporabo predlagane večnivojske hierarhije. S tem namenom smo definirali tri nivoje: najnižji nivo je povezan z osnovnimi (atomskimi) akcijami (ang. *atomic actions*), kot so pritisk pedala za plin, menjava prestave, zavijanje z volanom itn.; z združitvijo teh akcij na drugem nivoju dobimo opravila (ang. *tasks*); na najvišjem nivoju pa zaznavamo manevre (npr. ustavljanje avtomobila, prehitevanje itn.), ki so sestavljeni iz več različnih opravil.

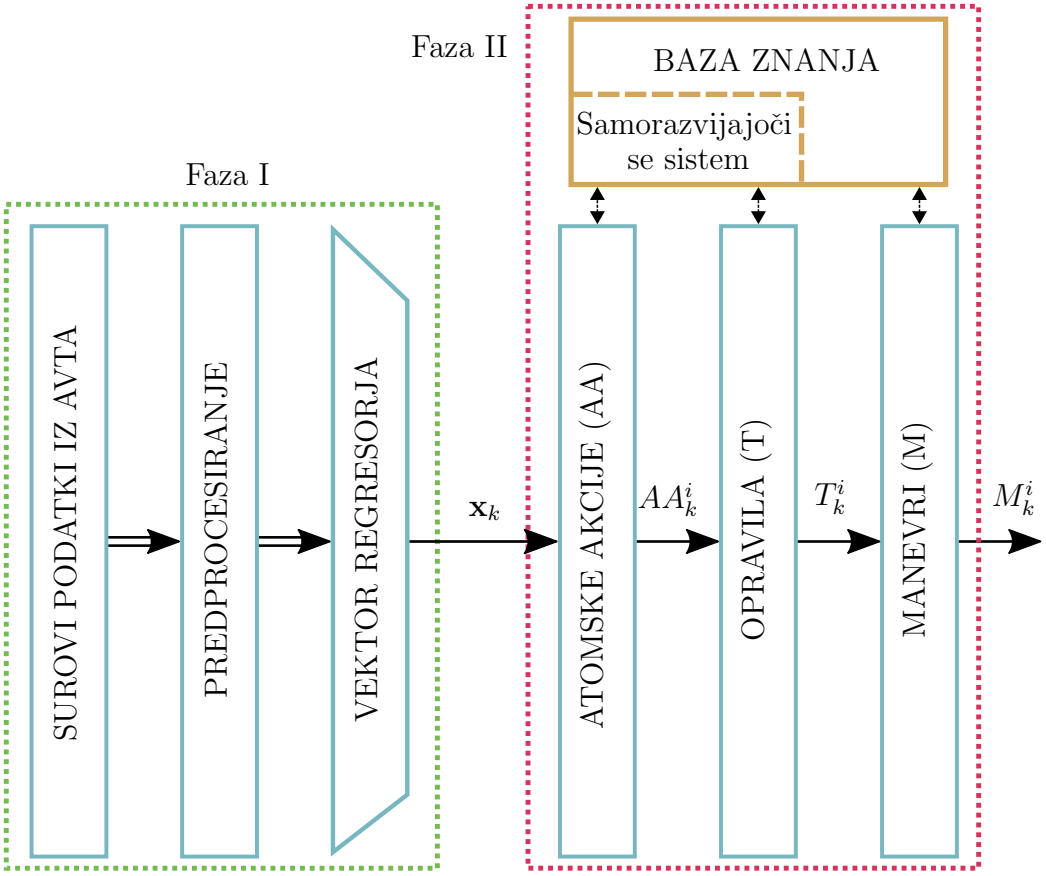
Na sliki 6.2 je predlagana rešitev za zaznavanje voznikovih manevrov. Proces zaznavanja je razdeljen v dve fazi. V prvi fazi zajamemo surove podatke iz avtomobila (npr. preko vodila CAN), ki jih predhodno procesiramo oziroma diskretiziramo ter sestavimo regresijski vektor, ki ga potem peljemo v predlagani algoritem samorazvijanja strukture za zaznavanje manevrov. V drugi fazi manevrov najprej zaznamo atomske akcije, potem zaznamo opravila in na koncu posamezne manevre.

Vsi prikazani koraki na sliki 6.2 bodo razloženi v nadaljevanju.

6.3.1 Zajemanje in obdelava surovih podatkov

Direktno zajemanje surovih podatkov iz avtomobila je možno preko vodila CAN. V našem primeru smo uporabili simulator vožnje, ki je predstavljen v [248]. Simulator omogoča zajemanje različnih podatkov iz avtomobila. Mi smo se omejili le na signale, ki so dostopni v skoraj vseh avtomobilih, npr. obračanje volana, plin,

zavora, sklopka, prestave, hitrost vozila in število vrtljajev na minuto. Simulator bo podrobno razložen v eksperimentalnem delu tega poglavja.



Slika 6.2: Prikaz sistema zajemanja in obdelava podatkov (Faza I) ter potek zaznavanja voznikovih manevrov (Faza II).

Pred-procesiranje

Postopek pred-procesiranja surovih podatkov je zelo pomemben del predlagane rešitve zaznavanja manevrov. S pred-procesiranjem odstranimo odvečne informacije in ohranimo le glavne trende pri signalih. Kot smo že rekli, v našem primeru zajamemo naslednje (surove) podatke: *hitrost* (v), *vrtljaji na minuto* (r), *pozicija volana* (w), *plin* (g), *zavora* (b), *sklopka* (p) in *prestave* (s). To so signali, ki so na voljo skoraj pri vseh standardnih avtomobilih.

Signale *hitrost* (v), *vrtljaji na minuto* (r) in *pozicija volana* (w) diskretiziramo

z naslednjo funkcijo $\hat{v}_k$:

$$\hat{v}_k = \begin{cases} 1, & \Delta v_k > 0 \\ 0, & \Delta v_k = 0 \\ -1, & \Delta v_k < 0 \end{cases} \tag{6.1}$$

kjer indeks k označuje časovni korak in $\Delta v_k = v_k - v_{k-1}$. Rezultat operacije v (6.1) je diskretni signal $\hat{v}_k$. Isti postopek je uporabljen za ostala dva signala r in w, tako da je diskretna vrednost teh signalov označena z $\hat{r}$ in $\hat{w}$.

Naslednja dva signala, *plin* (g) in *zavora* (b), sta pred-procesirana z uporabo naslednje funkcije:

$$\hat{g}_k = \begin{cases} 1, & \Delta g_k > 0 \quad or \quad g_k = 1 \\ 0, & \Delta g_k = 0 \\ -1, & \Delta g_k < 0 \end{cases} \tag{6.2}$$

kjer je $\Delta g_k = g_k - g_{k-1}$. Prva vrstica v enačbi (6.2) upošteva tudi maksimalno vrednost signala pri $g_k = 1$ in v tem primeru je tudi diskretna vrednost $\hat{g}_k = 1$. To pomeni, da nas ne zanima le sprememba signala Δg_k, ampak nas zanima kdaj je *plin* (g) ali *zavora* (b) pritisnjen/-a do konca. Diskretne spremenljivke so označene z $\hat{g}$ in $\hat{b}$.

Pri analizi signala *sklopka* (p) nas zanima, ali je sklopka pritisnjena ali ne. Z uporabo naslednjega postopka diskretizacije signala dobimo želeno informacijo:

$$\hat{p}_k = \begin{cases} 1, & p_k > 0 \\ 0, & p_k = 0 \end{cases} \tag{6.3}$$

Zadnji signal, *prestave* (s), je pravzaprav že diskretne narave in lahko zavzame naslednja stanja $s = \{1, 2, 3, 4, 5, N, R\}$, kjer $N = 0$ označuje nevtralno in $R = -1$ vzvratno prestavo. V tem primeru nas zanimajo trenutki, ko se prestava spremeni navzgor ali navzdol:

$$\hat{s}_k = \begin{cases} 1, & \Delta s_k > 0 \\ -1, & \Delta s_k < 0 \end{cases} \tag{6.4}$$

kjer je $\Delta s_k = s_k - s_{k-1}$.

Regresijski vektor

Regresijski vektor je sestavljen iz vseh diskretiziranih signalov iz prejšnjega razdelka (ima dimenzijo $n = 7$) in ima naslednjo obliko:

$$\mathbf{x}_k = \left[\hat{v}_k,\ \hat{r}_k,\ \hat{w}_k,\ \hat{g}_k,\ \hat{b}_k,\ \hat{p}_k,\ \hat{s}_k\right]^T \tag{6.5}$$

Regresijski vektor predstavlja osnovni gradnik algoritma za zaznavanje voznikovih manevrov in vsebuje $3^5 \cdot 2^2$ diskretnih stanj, ki tvorijo osnovne atomske akcije. Čeprav je, kot bomo videli v nadaljevanju, v praksi skoraj nemogoče doseči vsa ta diskretna stanja. Podrobnosti algoritma so razložene v naslednjih poglavjih.

6.3.2 Zaznavanje atomskih akcij

Kot smo že povedali v razdelku 6.3.1, je posamezna atomska akcija definirana kot eno od možnih $3^5 \cdot 2^2$ diskretnih stanj, ki jih lahko zavzame regresijski vektor. Pravzaprav je vsaka atomska akcija (označujemo z AA^i) predstavljena kot točka (ali področje točk) v m-dimenzijskem Evklidskem prostoru, kjer m označuje dimenzijo regresijskega vektorja (6.5).

Za zaznavanje novih atomskih akcij bomo uporabili postopek samorazvijanja (glej Algoritem 3) za dodajanje novih oblakov (mehkih pravil) na podlagi lokalne gostote podatkov:

$$\gamma_k^{i_A} = \frac{1}{1 + \|\mathbf{x}_k - \boldsymbol{\mu}_k^{i_A}\|^2 + \sigma_k^{i_A} - \|\boldsymbol{\mu}_k^{i_A}\|^2}, \qquad i_A = 1, \ldots, c_A \tag{6.6}$$

kjer i_A označuje indeks atomske akcije, c_A število trenutno znanih atomskih akcij, $\boldsymbol{\mu}_k^{i_A}$ (glej enačbo (2.14)) je srednja vrednost in $\sigma_k^{i_A}$ (glej enačbo (2.15)) je povprečje vsote dolžin vseh podatkov, ki pripadajo i_A-ti atomski akciji.

V algoritmu 3 (glej vrstice 1-21) so predstavljeni vsi koraki, ki se nanašajo na detekcijo novih atomskih akcij. Potrebno je povedati, da se prva atomska akcija inicializira s prvim prejetim podatkom (vrstice 7-9). Z vsakim naslednjim podatkom $\mathbf{x}_k$ se izračunajo vse lokalne gostote po enačbi (6.6) in če je maksimalna vrednost lokalnih gostot nižja od praga $\gamma_{max} = 0.85$, se doda nova atomska akcija (vrstica 11).

Algoritem 3 Psevdokoda algoritma za zaznavanje atomskih akcij in opravil.

Inicializacija parametrov: $\gamma_{max} = 0.85$.
repeat ▷ *k je indeks časovnega koraka*
  Vhod: Zajem surovih podatkov iz avta
  Pred-procesiranje: (6.1), (6.2), (6.3) in (6.4)
  Regresijski vektor: $\mathbf{x}_k$.
  Izračun: $\gamma_k^{i_A}, \quad i_A = 1, \ldots, c_A$.
  if k==1 **then**
    Inicializacija: $c_A \leftarrow 1$, $\mu_k^{c_A} \leftarrow \mathbf{x}_k$, $\sigma_k^{c_A} \leftarrow \|\mathbf{x}_k\|^2$, $N_k^{c_A} \leftarrow 1$.
    Določi: $AA_k^{c_A} \in \{\mu_k^{c_A}, \sigma_k^{c_A}, N_k^{c_A}\}$
  else
    if $\max_{i_A}(\gamma_k^{i_A}) < \gamma_{max}$ **then** ▷ *Postopek samorazvijanja za atomske akcije.*
      Zaznamo novo atomsko akcijo.
      Povečaj: c_A.
      Inicializacija: $\mu_k^{c_A} \leftarrow \mathbf{x}_k$, $\sigma_k^{c_A} \leftarrow \|\mathbf{x}_k\|^2$, $N_k^{c_A} \leftarrow 1$.
      Določi: $AA^{c_A} \in \{\mu_k^{c_A}, \sigma_k^{c_A}, N_k^{c_A}\}$.
    else
      Poveži podatek $\mathbf{x}_k$ z AA ($max_{i_A}\gamma_k^{i_A}$)
      Posodobi $\mu_k^{i_A}$, $\sigma_k^{i_A}$ in $N_k^{i_A}$ za izbran AA
    end if
  end if
  Izhod: Center trenutno aktivnega AA ($\mu_k^{i_A}$).
  Inicializacija parametrov: $\gamma_{max} = 0.25$, $n_{add} = 10$.
  Vhod: Center trenutno aktivnega AA ($\mu_k^{i_A}$).
  Izračun: $\gamma_k^{i_T}, \quad i_T = 1, \ldots, c_T$.
  if k==1 **then**
    Inicializacija: $c_T \leftarrow 1$, $\mu_k^{c_T} \leftarrow \mu_k^{i_A}$, $\sigma_k^{c_T} \leftarrow \|\mu_k^{i_A}\|^2$, $N_k^{c_T} \leftarrow 1$.
    Določi: $T_k^{c_T} \in (\mu_k^{c_T}, \sigma_k^{c_T}, N_k^{c_T})$
  else
    if $(\max_{i_T}(\gamma_k^{i_T}) < \gamma_{max}) \wedge (k > k_{add} + n_{add})$ **then** ▷ *Postopek samorazvijanja za opravila.*
      Zaznamo novo opravilo.
      Povečaj: c_T,
      Inicializacija: $\mu_k^{c_T} \leftarrow \mathbf{x}_k$, $\sigma_k^{c_T} \leftarrow \|\mathbf{x}_k\|^2$, $N_k^{c_T} \leftarrow 1$.
      Določi: $T^{c_T} \in (\mu_k^{c_T}, \sigma_k^{c_T}, N_k^{c_T})$
    else
      Poveži trenuten AA center ($\mu_k^{i_A}$) z T ($\max_{i_T}(\gamma_k^{i_T})$)
      Posodobi $\mu_k^{i_T}$, $\sigma_k^{i_T}$ in $N_k^{i_T}$ za trenuten $T_s^{i_T}$
    end if
  end if
  Izhod: Indeks trenutno aktivnega opravila (i_T).
until Konec toka podatkov.

6.3.3 Zaznavanje opravil

Opravila predstavljajo drugi korak faze II pri zaznavanju manevrov (glej sliko 6.2) in so pravzaprav predstavljene kot skupina med seboj podobnih atomskih akcij. Tako kot atomske akcije tudi opravila zaznavamo z uporabo samorazvi-

jajočim se modelom. Drugi del algoritma 3 (glej vrstice 22-39) prikazuje korake za zaznavanje opravil. Najprej se inicializirajo vhodni parametri ($\gamma_{max} = 0.25$, $n_{add} = 10$) algoritma. Vhod v algoritem je center trenutno aktivne atomske akcije $\mu_k^{i_A}$, potem sledi razvrščanje k obstoječemu opravilu ali pa dodamo novo opravilo. Postopek samorazvijanja za zaznavanje novih opravil temelji na pragu lokalne gostote $\gamma_{max} = 0.25$ ter na zadrževalnem parametru $n_{add} = 10$ (vrstica 29 v algoritmu 3). Izhod iz algoritma je indeks ($i_T,\ i_T = 1, \ldots, c_T$) trenutno aktivnega opravila. Tako kot je prikazano na sliki 6.2, temu koraku sledi končni korak za zaznavanje manevrov.

6.3.4 Zaznavanje manevrov

Zaznavanje manevrov predstavlja zadnji korak celotne procedure, opisane na sliki 6.2. Določen manever je definiran kot ponavljajoča se sekvenca različnih opravil. Na primer, i-ti manever (označujemo z M^i) je definiran kot vektor opravil $M^i = [T^1, T^3, T^2, T^4, T^3]^T$, kjer se opravila T^1, T^2, T^3 in T^4 izvedejo v določenem zaporedju. Psevdokoda, ki je predstavljena v algoritmu 4, opisuje korake za zaznavanje manevrov na podlagi opravil. Inicializacija algoritma se začne z edinim parametrom $\delta_T = 7$, ki je določen eksperimentalno. Parameter lahko razumemo kot širino okna nizkoprepustnega filtra, ki odseka vse hitre spremembe. To pomeni, da pride do spremembe opravila le v primeru, če ta sprememba traja več kot δ_T časovnih korakov. V algoritmu 4 označuje spremenljivka $last_{i_T}$ zadnji zaznani indeks opravila, κ je notranji števec, i_T je indeks trenutnega opravila in M_k vektor manevra, ki vsebuje določeno sekvenco zaznanih opravil.

V fazi učenja se zaznane atomske akcije, opravila in manevre shranjuje v bazo znanja (ang. *knowledge base*). V fazi vrednotenja trenutno sekvenco opravil primerjamo z znanimi manevri iz baze znanja. Primerjava temelji na Hammingovi razdalji $d_H(x, y)$. Hammingova razdalja je definirana tako, da primerja istoležne komponente med dvema vektorjema x in y. Rezultat predstavlja število komponent, ki ne sovpadajo med seboj. Trenutna sekvenca opravil, ki se sprotno shranjuje v vektor $M_k \in \mathbb{R}^n$, se primerja z znanimi manevri iz baze znanja z naslednjim izrazom:

$$\left(\min_{i_M = 1, \ldots, c_M} \frac{d_H(M_k, M^{i_M})}{n} \right) < m_f \tag{6.7}$$

Algoritem 4 Psevdokoda algoritma za zaznavanje manevrov M_s.

Inicializacija parametrov: $\delta_T = 7$.
Nastavi: $last_{i_T} = -1$, $\kappa = 0$.
Vhod: Indeks trenutno aktivnega opravila (i_T).
repeat ▷ *Začetek toka podatkov.*
    if $i_T \neq last_{i_T}$ **then**
        Povečaj: $\kappa \leftarrow \kappa + 1$
        if $\kappa > \delta_T$ **then**
            $M_k \leftarrow [M_{k-1} \mid i_T]$
            $\kappa \leftarrow 0$
            $last_{i_T} \leftarrow i_T$
        else
            $M_k \leftarrow [M_{k-1}]$
        end if
    else
        $\kappa \leftarrow 0$
        $M_k \leftarrow [M_{k-1}]$
    end if
    Izhod: Sekvenca trenutnega manevra M_k.
until Konec toka podatkov.

kjer je $m_f = 0.35$ in predstavlja prag zaznavanja manevrov, ki je nastavljen na osnovi eksperimentalnih poskusov in n predstavlja število elementov vektorja M_k.

6.4. Rezultati

V tem podpoglavju bodo predstavljeni rezultati zaznavanja manevrov na podlagi podatkov, pridobljenih s simulatorjem vožnje STISIM Drive® [237]. Simulator vsebuje pogonske komponente (volan, sklopka, zavora itn.), ki simulirajo obnašanje kot v realnem vozilu. Na primer, simulator vključuje in simulira silo pri obračanju volana vozila, tako da ima voznik občutek, kot v realnem vozilu. Okolje simulatorja je predstavljeno na sliki 6.3.

Metodologija, ki je bila uporabljena za pridobivanje podatkov, je sestavljena iz naslednjih korakov:

Slika 6.3: Slika simulatorja vožnje STISIM Drive®.

1. *Oblikovanje simulacijskega scenarija.* Vsak scenarij mora biti oblikovan tako, da se določen manever lahko izvede. Poleg tega vsebuje elemente, kot so npr. križišča, semaforji, druga vozila itn., ki nam omogočajo dodatno vizualno informacijo ob izvajanju in detekciji manevra. Na sliki 6.4 so predstavljene tri slike, ki prikazujejo potek izvajanja manevra *prehitevanje*.

2. *Pridobivanje podatkov.* V času izvajanja določenega scenarija se surovi podatki iz simulatorja shranjujejo v realnem času. Prav tako se shranjuje tudi video zapis izvajanja, da se lažje označi začetek in konec vsakega manevra. Na koncu poskusa se vsi podatki zapišejo v besedilno datoteko v urejenem zaporedju. To nam omogoča nadaljnjo obdelavo in analizo podatkov.

3. *Označevanje manevrov.* Pridobljene podatke iz prejšnjega koraka označimo na osnovi video zapisa. Trenutke, ko se izvaja določen manever, označimo zastavico (ang. *flag*) manevra z $f_M = 1$, medtem ko trenutke, ko ni manevra pa z $f_M = 0$.

V tej študiji smo obravnavali štiri različne manevre: *prehitevanje*, *ustavljanje*, *ustavljanje na semaforju* in *varnostna razdalja*.

Slika 6.4: Slikovni prikaz manevra *prehitevanje*.

6.4.1 Faza učenja

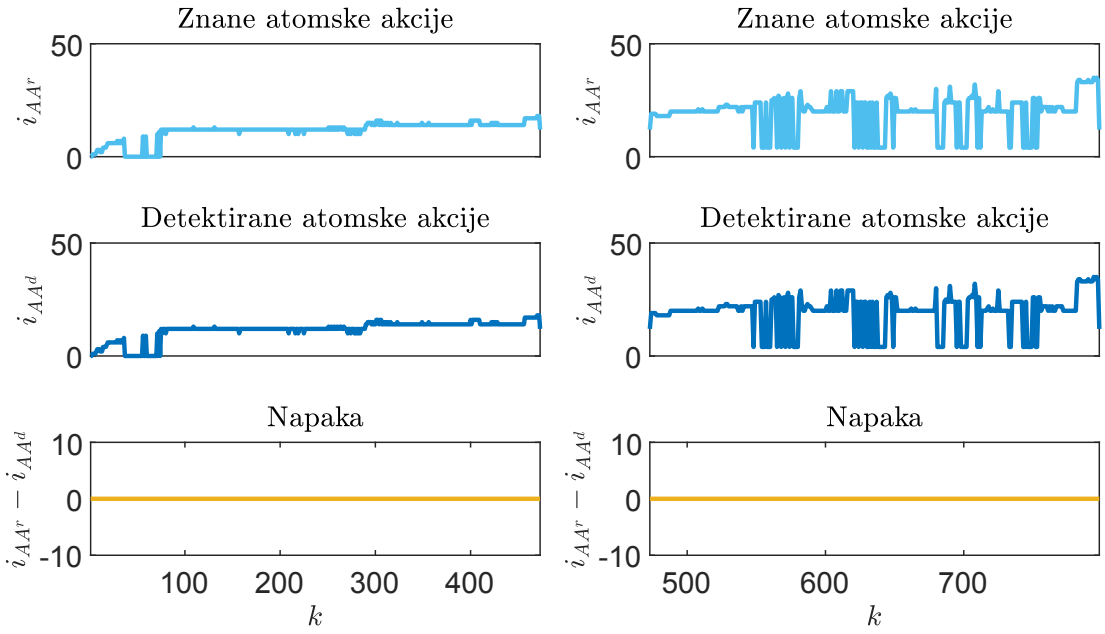
Glavni cilj v fazi učenja je pridobiti prototipe za vsak manever posebej. To je izvedeno preko zaznave atomskih akcij in opravil. V nadaljevanju bomo predstavili rezultate faze učenja.

Zaznavanje atomskih akcij

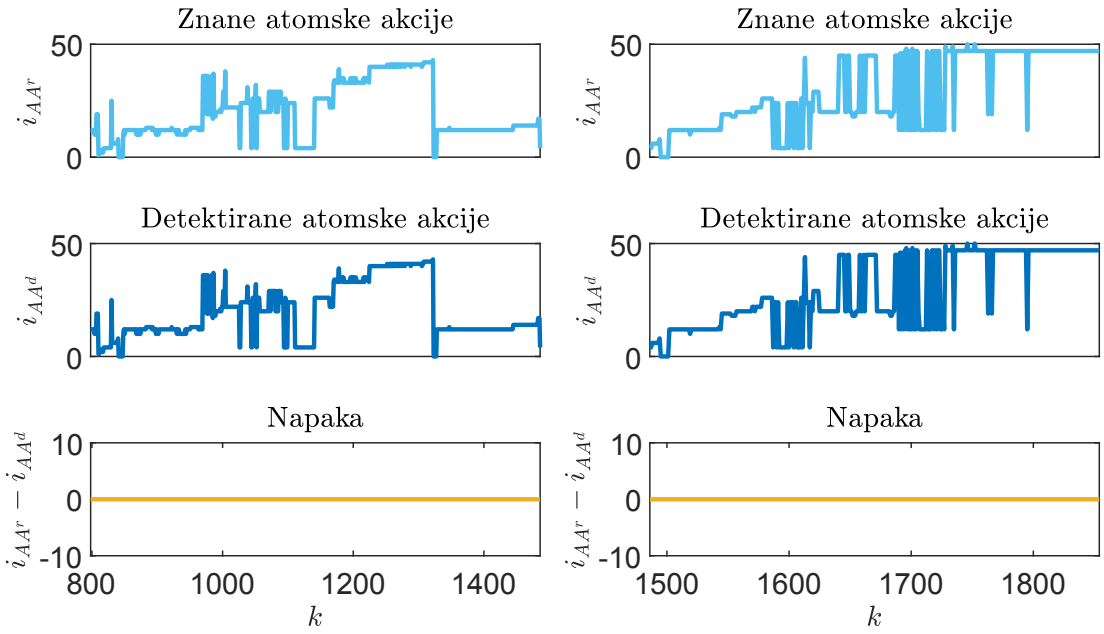
Kot je opisano v prvem delu algoritma 3 (vrstice 1 do 21), surove podatke iz simulatorja najprej pred-procesiramo z uporabo enačb (6.1), (6.2), (6.3) in (6.4). Potem sestavimo regresijski vektor (6.5), ki predstavlja vhod v samorazvijajoči se model. Z uporabo tega avtomatsko zaznamo centre atomskih akcij.

Na sliki 6.5 so prikazani rezultati za vsak manever posebej (Slike 6.5a, 6.5b, 6.5c in 6.5d). Atomske akcije so v osnovi določene s strani operaterja (znane atomske akcije) in predstavljajo zgolj ročni test delovanja predlaganega algoritma. Vsaka slika posameznega manevra vsebuje tri grafe; prvi označuje znane, drugi detektirane in tretji razlike med prvima dvema. Ordinatna os na slikah predstavlja indeks atomske akcije. Iz slike 6.5 je razvidno, da z uporabo samorazvijajočega se modela zaznamo vse atomske akcije, ki jih je predvidel operater.

V tem poskusu smo zaznali $c_A = 51$ različnih atomskih akcij, kar je bistveno manj, kot vseh možnih $3^5 \cdot 2^2$.



(a) Manever: *prehitevanje.* (b) Manever: *ustavljanje.*



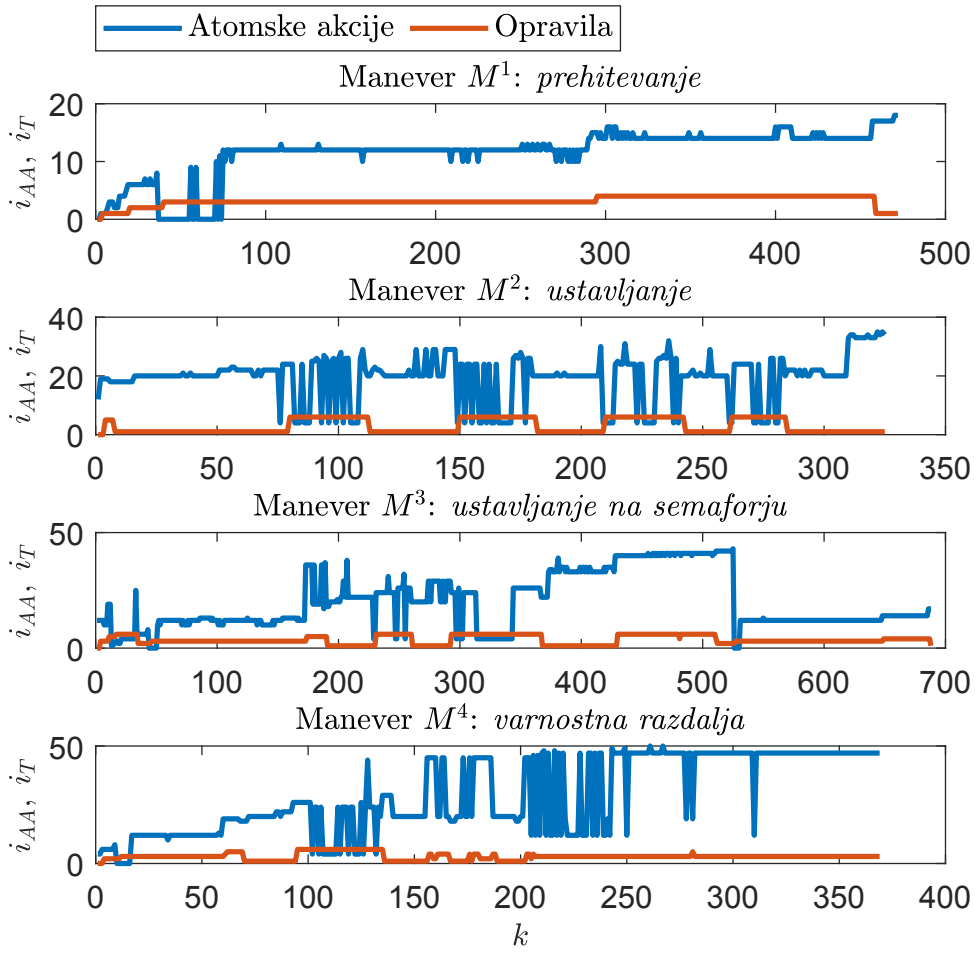
(c) Manever: *ustavljanje na semaforju.* (d) Manever: *varnostna razdalja.*

Slika 6.5: Faza učenja. Prikaz zaznanih atomskih akcij z algoritmom 3.

Zaznavanje opravil

Hkrati s postopkom zaznavanja atomskih akcij poteka tudi postopek zaznavanja opravil, ki predstavlja višji nivo abstrakcije. Kot je opisano v algoritmu 3, zaznamo opravila prav tako s samorazvijajočim se modelom, vendar pa z drugačnimi nastavitvami. Pravzaprav ta del algoritma združuje atomske akcije, ki ležijo blizu skupaj v 7-dimenzijskem Evklidskem prostoru.

Na sliki 6.6 so prikazani rezultati zaznavanja opravil (rdeča barva) za vsak manever posebej. Poleg tega so prikazane tudi atomske akcije (modra barva), ki so podrobno razložene na sliki 6.5. Na ordinatni osi so prikazani indeksi atomskih akcij (i_{AA}) in opravil (i_T). V tem primeru in z nastavitvami, kot so opisane v algoritmu 3, smo dobili $c_T = 6$ različnih opravil.



Slika 6.6: Faza učenja. Prikaz zaznanih opravil z algoritmom 3.

Zaznavanje manevrov

V zadnjem koraku z algoritmom 4 zaznamo zaporedja opravil za vsak manever posebej. Pravzaprav algoritem 4 za vhode sprejme indekse opravil in jih uredi v pripadajoče zaporedje v vektorski obliki. Tako za vsak manever dobimo ustrezno zaporedje:

prehitevanje:
$M^1 = [T^1\ T^2\ T^3\ T^4\ T^5]$,
ustavljanje:
$M^2 = [T^6\ T^1\ T^6\ T^1\ T^6\ T^1\ T^6\ T^1]$,
ustavljanje na semaforju:
$M^3 = [T^5\ T^6\ T^2\ T^3\ T^5\ T^1\ T^6\ T^1\ T^6\ T^1\ T^6\ T^2\ T^3\ T^4]$,
varnostna razdalja:
$M^4 = [T^2\ T^3\ T^5\ T^1\ T^6\ T^1\ T^4\ T^1\ T^2\ T^1\ T^3]$.

Če povzamemo fazo učenja, smo v bazo znanja pridobili in shranili 51 atomskih akcij, 6 opravil in 4 manevre. V nadaljevanju se shranjeni podatki uporabljajo ob postopku vrednotenja.

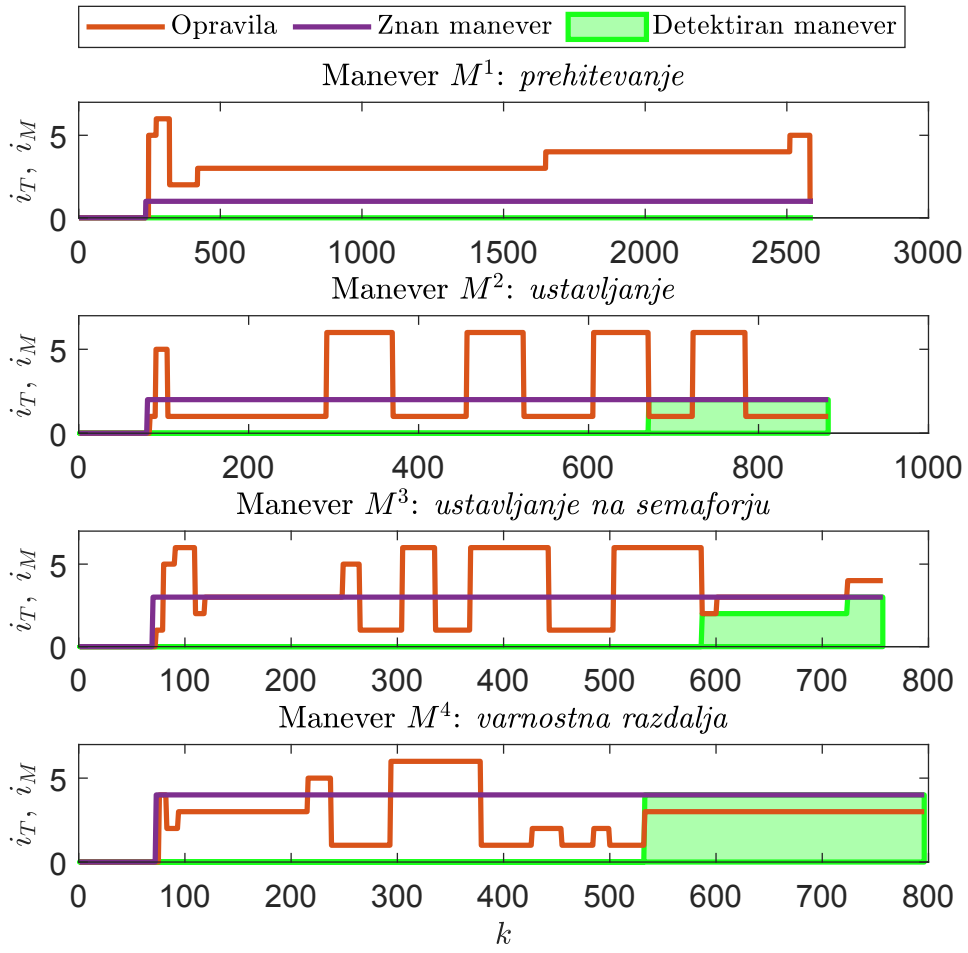
6.4.2 Faza vrednotenja

Med postopkom učenja smo pridobili znanje na treh nivojih (centri atomskih akcij, centri opravil in sekvence manevrov), ki ga uporabljamo v fazi ovrednotenja. V fazi vrednotenja je mehanizem samorazvijanja (dodajanje novih oblakov) ustavljen in algoritem 3 služi le kot klasifikator. To pomeni, da sprotno zajete podatke razvrstimo najprej v pripadajočo atomsko akcijo in potem k pripadajočim opravilom. Ko pridobimo zadostno velik niz opravil, ga z uporabo enačbe (6.7) primerjamo s pridobljeno bazo manevrov (M^1, M^2, M^3, M^4).

Na sliki 6.7 so prikazani rezultati zaznavanja posameznih manevrov. Rdeče črte pomenijo indekse detektiranih opravil, medtem ko je z vijolično barvo označeno področje, kjer nastopa znan manever. Na koncu je z zeleno barvo prikazano področje, ko zaznamo določen manever. V trenutku, ko zelena črta spremeni svojo vrednost, je izpolnjen pogoj (6.7).

Občutljivost zaznavanja manevrov je neposredno odvisna od nastavitvijo parametra $m_f = 0.35$, ki mu rečemo tudi parameter občutljivosti (ang. *sensitivity*

parameter). Če je vrednost parametra previsoka, zaznamo manever bistveno prej, vendar pa to poveča tudi napake pri zaznavanju ostalih manevrov. S trenutno nastavitvijo parametra za občutljivost do napake pride v primeru prvega in tretjega manevra (M^1 in M^3 na sliki 6.7). Prvega manevra pa s trenutno nastavitvijo nismo zaznali. Z višanjem vrednosti parametra m_f bi ga lahko zaznali ampak bi se pojavile tudi dve novi napaki pri drugem in četrtem manevru. V primeru tretjega manevra pa najprej zaznamo manever M^2, in šele proti koncu eksperimenta M^3. To je pravzaprav tudi pričakovano, saj je manever *ustavljanje* sestavni del manevra *ustavljanje na semaforju*.



Slika 6.7: Faza vrednotenja. Prikaz zaznavanja manevrov.

6.5. Zaključek

V tem poglavju smo predstavili rezultate raziskave novega koncepta zaznavanja voznikovih manevrov z uporabo samorazvijajočega se sistema na podlagi mehkih oblakov podatkov. Prednost predlaganega algoritma je, da lahko na podlagi osnovnih, že vgrajenih senzorjev v avtomobilu zaznamo različne manevre, ki jih voznik izvaja. To olajša in predvsem zmanjša stroške implementacije.

Predlagani koncept zaznavanja manevrov daje obetavne rezultate. Za bolj verodostojne rezultate bi ga bilo potrebno testirati na večji podatkovni bazi z različnimi vozniki in pod spremenljivimi pogoji. Zaznavanje voznikovega obnašanja predstavlja dodatni izziv za katerega v literaturi nismo našli ustreznih rešitev.

Zaznavanje voznikovega početja je del kompleksnejšega sistema, kjer le z ekspertnim (inteligentnim) kopilotom lahko zaznamo stanje voznika. S tem namenom lahko samorazvijajoči se mehki sistem nadgradimo z zaznavanjem dinamičnega modela voznika. Na ta način lahko z uporabo modela voznika lažje razumemo njegovo početje. Poleg tega lahko integracija z naprednimi senzorji (lidar, kamera, radar itn.) znatno izboljša učinkovitost pristopa. Nadaljnja raziskava na tem področju odpira različne možne nadgradnje predstavljenega pristopa.

7. Sklep

V doktorski disertaciji smo raziskali in predstavili možnosti uporabe samorazvijajočega se modela na osnovi oblakov podatkov (ang. *data clouds*) na več različnih področjih, kot so vodenje, identifikacija in spremljanje nelinearnih procesov. Predstavili smo lokalno gostoto (in delno lokalno gostoto) kot mero za delitev podatkovnega prostora. Raziskali smo mehanizme za dodajanje in odstranjevanje oblakov. Prav tako smo predlagali mehanizme za preprečevanje dodajanja oblakov na osnovi osamelcev.

Najprej smo predstavili primer uporabe samorazvijajočega se modela na področju vodenja. Predstavili smo robusten samorazvijajoči se adaptivni regulator RECCo. Postopek samorazvijanja skrbi za zaznavanje nelinearnih področij v procesu in adaptivni zakon prilagaja parametre posameznih regulatorjev za vsako področje posebej. Poleg tega smo podali nekaj zaščitnih mehanizmov adaptacije za povečanje robustnosti celotnega algoritma. Regulator smo najprej testirali na simulacijskih primerih ter dobljene rezultate primerjali z rezultati dobljenimi s klasičnimi regulatorji. Potem smo delovanje regulatorja preizkusili na realnih napravah. Pri vseh primerih je regulator izkazal pozitivne rezultate predvsem zaradi možnosti prilagajanja strukture in parametrov. Prednost predlagane metode ja ta, da se ves čas prilagaja novim razmeram, ki se dogajajo na vodenem procesu (motnje, spremembe zunanjih pogojev, delovne točke itn.).

Sledile so raziskave na področju sprotne identifikacije dinamičnih procesov. Predlagali smo kombinacijo samorazvijajočega se sistema na osnovi oblakov z rekurzivnimi najmanjšimi kvadrati. Obravnavali smo različne mehanizme dodajanja in odstranjevanja mehkih pravil. Metodo smo preizkusili za identifikacijo kazalnikov proizvodne učinkovitosti na kompleksnem procesu Tennessee Eastman. Izkazalo se je, da je metoda primerna za sprotno identifikacijo dinamičnih in spremenljivih procesov. Prednost je v enostavnosti nastavljanja začetnih parametrov ter možnosti sprotnega učenja z novimi podatki.

Na koncu disertacije smo obravnavali področje spremljanja procesov na dveh različnih primerih uporabe. Najprej smo predstavili samorazvijajoči se sistem,

ki temelji na osnovi delnih lokalnih gostot in je namenjen zaznavanju napak na industrijskih procesih. Izračun delnih lokalnih gostot nam omogoča lažjo in hitrejšo izbiro najbolj vplivnih komponent. Sistem se na osnovi testnih podatkov nauči in razdeli prostor problema na področje normalnega delovanja in področje, ki označuje napako. Nato, sistem lahko sprotno zaznava (ob vsakem sprejetem podatku) ali je prišlo do napake na sistemu oziroma ali sistem deluje v normalnem obratovanju. Metodo smo testirali na kompleksnem procesu Tennessee Eastman in rezultate smo primerjali z znanimi in uveljavljenimi metodami na področju zaznavanja napak. Predlagana metoda daje primerljive rezultate tudi s privzetimi začetnimi parametri, medtem ko optimalna nastavitev parametrov deluje bolje od ostalih metod.

Na področju spremljanja smo izvedli začetno študijo in pokazali, da so lahko samorazvijajoči se sistemi uporabni tudi za zaznavanje manevrov pri voznikih. Metoda rešuje kompleksnost problema tako, da ga razdeli na hierarhične nivoje, ki predstavljajo svojo raven abstrakcije. Na vsakem nivoju uporabimo logiko samorazvijanja za zaznavanje posameznih segmentov. Končni rezultat je sekvenca opravil, ki jih izvede voznik, in predstavljajo določen manever. Z metodo smo uspešno zaznali nekaj osnovnih manevrov, kot so prehitevanje, ustavljanje, ustavljanje na semaforju in varnostna razdalja.

Na podlagi vseh podanih primerov in primerjav lahko ugotovimo, da je samorazvijajoči se model na osnovi oblakov uporaben na področjih vodenja, identifikacije in spremljanja procesov. Model omogoča sprotno obdelavo podatkov in je primeren za nelinearne, dinamične in časovno spremenljive procese. Sprotno spremljanje procesa v veliki meri prispeva k bolj optimalnemu delovanju celotnega sistema.

8. Prispevek k znanosti

Glavne izvirne prispevke k znanosti lahko povzamemo v naslednjih točkah:

1. **Zasnova in razvoj robustnega samorazvijajočega se adaptivnega mehkega regulatorja;**
   Na podlagi mehkega modela AnYa smo razvili regulator, ki samodejno prilagaja strukturo različnim delovnim točkam procesa ter adaptira parametre lokalnih regulatorjev.

2. **Razvoj rekurzivne metode za identifikacijo nelinearnih dinamičnih sistemov, ki temelji na samorazvijajočem se mehkem modelu;**
   S kombinacijo samorazvijajočega se mehanizma in z rekurzivno metodo mehkih/uteženih najmanjših kvadratov smo razvili metodo za identifikacijo dinamičnih in nelinearnih sistemov.

3. **Razvoj samorazvijajočega se mehkega prediktivno funkcijskega regulatorja;**
   Z uporabo rekurzivne metode za identifikacijo procesov smo razvili prediktivno funkcijski regulator, ki je sposoben sprotno prilagajati svojo strukturo.

4. **Razvoj metode za zaznavanja napak, ki temelji na samorazvijajočem se modelu;**
   Razvili smo samorazvijajočo se metodo na osnovi delnih lokalnih gostot za zaznavanje napak na dinamičnih sistemih. Metoda upošteva najbolj vplivne komponente pri izračunu pripadnosti k določenemu oblaku podatkov.

5. **Razvoj metode za zaznavanje manevrov pri voznikih, ki temelji na hierarhični strukturi samorazvijajočega se modela;**
   Metoda razdeli kompleksnost problema na več hierarhičnih nivojev. Vsak nivo predstavlja določeno raven abstrakcije. Znotraj vsakega nivoja smo vnesli samorazvijajoči se model, ki zaznava segmente posameznega nivoja. Sekvenca segmentov na najvišjem nivoju predstavlja manever, ki ga opravi voznik.

Literatura

[1] T. Abdelazim and O. P. Malik. An Adaptive Power System Stabilizer Using On-line Self-leaming Fuzzy Systems. In *IEEE Power Engineering Society General Meeting*, pages 1715–1720, Toronto, Ont., Canada, 2003.

[2] J. Abonyi. *Fuzzy Model Identification for Control*. Springer Science+Business Media New York, 2003.

[3] S. Abtahi, B. Hariri, and S. Shirmohammadi. Driver drowsiness monitoring based on yawning detection. In *Instrumentation and Measurement Technology Conference (I2MTC)*, pages 7–10, 2011.

[4] N. Abuali and H. Abou-zeid. Driver Behavior Modeling : Developments and Future Directions. *International Journal of Vehicular Technology*, 2016:1–12, 2016.

[5] A. Akhenak, M. Chadli, J. Ragot, and D. Maquin. Fault detection and isolation using sliding mode observer for uncertain Takagi-Sugeno fuzzy model. In *2008 16th Mediterranean Conference on Control and Automation*, pages 286–291, Ajaccio, France, 2008.

[6] M. Albieri, A. Beghi, C. Bodo, and L. Cecchinato. A simulation environment for the design of advanced chiller control systems. In *Proceedings of the 3rd Annual IEEE Conference on Automation Science and Engineering*, pages 962–967, Scottsdale, AZ, USA, 2007.

[7] E. Alcorta-Garcia and S. Saucedo-Flores. Fault detection and isolation based on Takagi-Sugeno modelling. In *Proceedings of the 2003 IEEE International Symposium on Intelligent Control*, pages 673–678, 2003.

[8] N. Almalki Al-Jehani, H. N. Nounou, and M. N. Nounou. Fuzzy control of a CSTR process. In *8th International Symposium on Mechatronics and its Applications (ISMA 2012)*, pages 1–6, 2012.

[9] A. Altinten, S. Erdogan, F. Alioglu, H. Hapoglu, and M. Alpbaz. Application of adaptive PID control with genetic algorithm to a polymerization reactor. *Chemical Engineering Communications*, 191(9):1158–1172, 2004.

[10] G. Andonovski, P. Angelov, S. Blažič, and I. Škrjanc. A practical implementation of Robust Evolving Cloud-based Controller with normalized data space for heat-exchanger plant. *Applied Soft Computing*, 48:29–38, 2016.

[11] G. Andonovski, S. Blažič, P. Angelov, and I. Škrjanc. Analysis of Adaptation Law of the Robust Evolving Cloud-based Controller. In *2015 IEEE International Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, pages 1–7, 2015.

[12] G. Andonovski, G. Mušič, S. Blažič, and I. Škrjanc. Evolving model identification for process monitoring and prediction of non-linear systems. *Engineering Applications of Artificial Intelligence*, 68(October):214–221, 2017.

[13] P. Angelov. *Evolving Rule-Based Models: A Tool for Design of Flexible Adaptive Systems*. Springer-Verlag, London, 1st edition, 2002.

[14] P. Angelov. A fuzzy controller with evolving structure. *Information Sciences*, 161(1-2):21–35, 2004.

[15] P. Angelov. An approach for fuzzy rule-base adaptation using on-line clustering. *International Journal of Approximate Reasoning*, 35(3):275–289, 2004.

[16] P. Angelov. Evolving Takagi-Sugeno Fuzzy Systems from Streaming Data (eTS+). In P. Angelov, D. Filev, and N. Kasabov, editors, *Evolving Intelligent Systems: Methodology and Applications*, pages 21–50. John Wiley & Sons, New Jersey, 1st edition, 2010.

[17] P. Angelov. *Autonomous Learning Systems: From Data Streams to Knowledge in Real-time*. John Wiley & Sons, Ltd, 1st edition, 2012.

[18] P. Angelov. Anomaly Detection Based on eccentricity analysis. In *IEEE Symposium on Evolving and Autonomous Learning Systems (EALS)*, pages 1–8, Orlando, FL, USA, 2014.

[19] P. Angelov and D. Filev. Simpl_eTS : A simplified method for learning evolving Takagi-Sugeno fuzzy models. *IEEE International Conference on Fuzzy Systems*, pages 1068–1073, 2005.

[20] P. Angelov, D. Filev, and N. Kasabov. *Evolving Intelligent Systems: Methodology and Applications*. John Wiley & Sons, New Jersey, 2010.

[21] P. Angelov, D. Filev, and N. Kasabov. *Evolving Systems*. Springer Berlin Heidelberg, 2010.

[22] P. Angelov, I. Škrjanc, and S. Blažič. Robust Evolving Cloud-Based Controller for a Hydraulic Plant. In *IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, pages 1–8, 2013.

[23] P. Angelov and R. Yager. A simple fuzzy rule-based system through vector membership and kernel-based granulation. In *5th IEEE International Conference Intelligent Systems*, pages 349–354, 2010.

[24] P. Angelov and R. Yager. Simplified fuzzy rule-based systems using non-parametric antecedents and relative data density. In *Symposium Series on Computational Intelligence (IEEE SSCI 2011) - IEEE Workshop on Evolving and Adaptive Intelligent Systems (EAIS 2011)*, pages 62–69, 2011.

[25] P. Angelov and R. Yager. A new type of simplified fuzzy rule-based system. *International Journal of General Systems*, 41(2):163–185, 2012.

[26] P. P. Angelov and D. P. Filev. An approach to online identification of Takagi-Sugeno fuzzy models. *IEEE transactions on systems, man, and cybernetics.*, 34(1):484–498, 2004.

[27] L. T. Antelo, J. R. Banga, and A. A. Alonso. Hierarchical design of decentralized control structures for the Tennessee Eastman Process. *Computers and Chemical Engineering*, 32(9):1995–2015, 2008.

[28] K. J. Åström and T. Hägglund. *PID controllers: theory, design and tuning*. Instrument Society of America, 2nd edition, 1995.

[29] K. J. Åström and B. Wittænmark. *Adaptive Control*. Addison-Wesley Longman Publishing Co., Boston, MA, USA, 2 edition, 1994.

[30] D. Avrahami-Zilberbrand and G. A. Kaminka. Fast and complete symbolic plan recognition. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 653–658, Edinburgh, Scotland, 2005. Morgan Kaufmann Publishers Inc.

[31] M. F. Azeem, M. Hanmandlu, and N. Ahmad. Structure Identification of Generalized Adaptive Neuro-Fuzzy Inference Systems. *IEEE Transactions on Fuzzy Systems*, 11(5):666–681, 2003.

[32] R. Babuška. *Fuzzy Modeling for Control*. Kluwer Academic Publishers Norwell, MA, USA, 1998.

[33] U. S. Banu and G. Uma. Fuzzy Gain Scheduled Pole Placement Based State Feedback Control of CSTR. *International Conference on Information and Communication Technology in Electrical Sciences*, pages 63–68, 2007.

[34] A. Bathelt, N. L. Ricker, and M. Jelali. Revision of the Tennessee eastman process model. *IFAC-PapersOnLine*, 28(8):309–314, 2015.

[35] I. Baturone, F. Moreno-Velo, S. Sanchez-Solano, and A. Ollero. Automatic Design of Fuzzy Controllers for Car-Like Autonomous Robots. *IEEE Transactions on Fuzzy Systems*, 12(4):447–465, 2004.

[36] K. Bengler, K. Dietmayer, B. Farber, M. Maurer, C. Stiller, and H. Winner. Three decades of driver assistance systems: Review and future perspectives. *IEEE Intelligent Transportation Systems Magazine*, 6(4):6–22, 2014.

[37] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Springer-Verlag US, Boston, MA, USA, 1981.

[38] C. G. Bezerra, B. S. J. Costa, L. A. Guedes, and P. P. Angelov. An evolving approach to unsupervised and Real-Time fault detection in industrial processes. *Expert Systems with Applications*, 63:134–144, 2016.

[39] M. Biglarbegian, W. W. Melek, and J. M. Mendel. Design of novel interval type-2 fuzzy controllers for modular and reconfigurable robots: Theory and experiments. *IEEE Transactions on Industrial Electronics*, 58(4):1371–1384, 2011.

[40] Z. Bingiil, G. E. Cook, and A. M. Strauss. Application of fuzzy logic to spatial thermal control in fusion welding. *IEEE Transactions on Industry Applications*, 36(6):1523–1530, 2000.

[41] S. Blažič, P. Angelov, and I. Škrjanc. Comparison of Approaches for Identification of All-data Cloud-based Evolving Systems. In *2nd IFAC Conference on Embedded Systems, Computer Intelligence and Telematics CESCIT 2015*, pages 129–134, Maribor, Slovenia, 2015.

[42] S. Blažič, D. Dovžan, and I. Škrjanc. Cloud-based Identification of an Evolving system With Supervisory Mechanisms. In *IEEE International Symposium on Intelligent Control (ISIC)*, pages 1906–1911, Antibes, France, 2014.

[43] S. Blažič, D. Dovžan, and I. Škrjanc. Robust evolving fuzzy adaptive control with input-domain clustering. *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 19:5387–5392, 2014.

[44] S. Blažič and I. Škrjanc. Design and stability analysis of fuzzy model-based predictive control — a case study. *Journal of Intelligent and Robotic Systems*, 49(3):279–292, 2007.

[45] S. Blažič, I. Škrjanc, and D. Matko. Globally stable direct fuzzy model reference adaptive control. *Fuzzy Sets and Systems*, 139(1):3–33, 2003.

[46] S. Blažič, I. Škrjanc, and D. Matko. A new fuzzy adaptive law with leakage. In *2012 IEEE Conference on Evolving and Adaptive Intelligent Systems*, pages 47–50, 2012.

[47] S. Blažič, I. Škrjanc, and D. Matko. A robust fuzzy adaptive law for evolving control systems. *Evolving Systems*, 5(1):3–10, 2014.

[48] P. Bonissone, V. Badami, K. Chiang, P. Khedkar, K. Marcelle, and M. Schutten. Industrial applications of fuzzy logic at General Electric. *Proceedings of the IEEE*, 83(3):450–465, 1995.

[49] R. Boukezzoula, S. Galichet, and L. Foulloy. Observer-Based Fuzzy Adaptive Control for a Class of Nonlinear Systems: Real-Time Implementation

for a Robot Wrist. *IEEE Transactions on Control Systems Technology*, 12(3):340–351, 2004.

[50] C. Cacciabue. *Modelling Driver Behaviourin Automotive Environments*. Springer-Verlag London Limited, 2007.

[51] E. F. Camacho, B. Manuel, F. R. Rubio, and D. Martinez. *Control of Solar Energy Systems*. Springer-Verlag London Limited 2012, 2012.

[52] E. F. Camacho, F. R. Rubio, M. Berenguel, and L. Valenzuela. A survey on control schemes for distributed solar collector fields. Part I: Modeling and basic control approaches. *Solar Energy*, 81(10):1240–1251, 2007.

[53] D. U. Campos-Delgado and D. R. Espinoza-Trejo. An observer based diagnosis scheme for single and simultaneous open switch faults in induction motor drives. *IEEE Transactions on Industrial Electronics*, 58(2):671–679, 2011.

[54] J. Candamo, M. Shreve, D. B. Goldgof, D. B. Sapper, and R. Kasturi. Understanding transit scenes: A survey on human behavior-recognition algorithms. *IEEE Transactions on Intelligent Transportation Systems*, 11(1):206–224, 2010.

[55] A. B. Cara, L. J. Herrera, H. Pomares, and I. Rojas. New Online Self-Evolving Neuro Fuzzy controller based on the TaSe-NF model. *Information Sciences*, 220:226–243, 2013.

[56] A. B. Cara, H. Pomares, I. Rojas, Z. Lendek, and R. Babuška. Online self-evolving fuzzy controller with global learning capabilities. *Evolving Systems*, 1(4):225–239, 2010.

[57] R. Carmona Contreras. *Análisis, Modelado, y control de un campo de colectores solares distribuidos con sistema de seguimiento en un eje*. PhD thesis, Universidad de Sevilla, España, 1985.

[58] G. Castignani, T. Derrmann, R. Frank, and T. Engel. Smartphone-Based Adaptive Driving Maneuver Detection: A Large-Scale Evaluation Study. *IEEE Transactions on Intelligent Transportation Systems*, 18(9):2330–2339, 2017.

[59] J. Cervantes-Villanueva, D. Carrillo-Zapata, F. Terroso-Saenz, M. Valdes-Vela, and A. F. Skarmeta. Vehicle maneuver detection with accelerometer-based classification. *Sensors (Switzerland)*, 16(10):1–23, 2016.

[60] A. A. Chaaraoui, P. Climent-Perez, and F. Florez-Revuelta. A review on vision techniques applied to Human Behaviour Analysis for Ambient-Assisted Living. *Expert Systems with Applications*, 39(12):10873–10888, 2012.

[61] W.-D. Chang, R.-C. Hwang, and J.-G. Hsieh. A self-tuning PID control for a class of nonlinear systems based on the Lyapunov approach. *Journal of Process Control*, 12(2):233–242, 2002.

[62] S. Charfeddine and L. Sbita. Performances Compared to the Sliding Mode and Gain Scheduling Control Methods of a CSTR Chemical Reactor. In *12th International Multi-Conference on Systems, Signals & Devices Performances*, pages 1–7, 2015.

[63] E. Charniak and R. P. Goldman. A Bayesian model of plan recognition. *Artificial Intelligence*, 64(1):53–79, 1993.

[64] A. Chen, H. Zhou, Y. An, and W. Sun. PCA and PLS Monitoring Approaches for Fault Detection of Wastewater Treatment Process. In *2016 IEEE 25th International Symposium on Industrial Electronics (ISIE)*, pages 1022–1027, 2016.

[65] B. Chen, S. Tong, and X. Liu. Fuzzy approximate disturbance decoupling of MIMO nonlinear systems by backstepping approach. *Fuzzy Sets and Systems*, 158(10):1097–1125, 2007.

[66] B. S. Chen, Y. S. Yang, B. K. Lee, and T. H. Lee. Fuzzy adaptive predictive flow control of ATM network traffic. *IEEE Transactions on Fuzzy Systems*, 11(4):568–581, 2003.

[67] D. Chen, Z. Li, and Z. He. Research on fault detection of tennessee eastman process based on PCA. In *25th Chinese Control and Decision Conference, CCDC 2013*, pages 1078–1081, 2013.

[68] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, and Z. Yu. Sensor-based activity recognition. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, 42(6):790–808, 2012.

[69] C. Y. Cheng, C. C. Hsu, and M. C. Chen. Adaptive kernel principal component analysis (KPCA) for monitoring small disturbances of nonlinear processes. *Industrial and Engineering Chemistry Research*, 49(5):2254–2262, 2010.

[70] L. Chiang, E. Russell, and R. Braatz. *Fault detection and diagnosis in industrial systems*. Springer-Verlag London Ltd., London, 2001.

[71] L. H. Chiang, M. E. Kotanchek, and A. K. Kordon. Fault diagnosis based on Fisher discriminant analysis and support vector machines. *Computers and Chemical Engineering*, 28(8):1389–1401, 2004.

[72] S. L. Chiu. Fuzzy model identification based on cluster estimation. *Journal of Intelligent and Fuzzy Systems*, 2(3):267–278, 1994.

[73] G. H. Cohen and G. A. Coon. Theoretical consideration of retarded control. *Transactions of ASME*, 75(1):827–834, 1953.

[74] B. S. J. Costa, P. P. Angelov, and L. A. Guedes. Real-time fault detection using recursive density estimation. *Journal of Control, Automation and Electrical Systems*, 25(4):428–437, 2014.

[75] B. S. J. Costa, P. P. Angelov, and L. A. Guedes. Fully unsupervised fault detection and identification based on recursive density estimation and self-evolving cloud-based classifier. *Neurocomputing*, 150(Part A):289–303, 2015.

[76] B. S. J. Costa, C. G. Bezerra, and L. A. H. G. De Oliveira. A multistage fuzzy controller: Toolbox for industrial applications. In *IEEE International Conference on Industrial Technology (ICIT)*, pages 1142–1147, 2012.

[77] B. S. J. Costa, C. G. Bezerra, L. A. Guedes, P. P. Angelov, and N. Z. Norte. Unsupervised Classification of Data Streams based on Typicality and Eccentricity Data Analytics. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 58–63, 2016.

[78] B. S. J. Costa, I. Škrjanc, S. Blažič, and P. Angelov. A practical implementation of self-evolving cloud-based control of a pilot plant. In *IEEE International Conference on Cybernetics, CYBCONF*, pages 7–12, 2013.

[79] J. C. de Barros and A. L. Dexter. On-line identification of computationally undemanding evolving fuzzy models. *Fuzzy Sets and Systems*, 158(18):1997–2012, 2007.

[80] DeLorenzo. Didactic process control pilot plant. In *DeLorenzo Italy: Catalog*, 2009.

[81] J. E. Dennis, Jr and R. B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, NJ, 1983.

[82] L. Desborough and R. Miller. Increasing Customer Value of Industrial Control Performance Monitoring — Honeywell ' s Experience. Technical Report Figure 1, Honeywell Hi-Spec Solutions, 2002.

[83] X. U. Dezhi, L. I. U. Fei, Z. Hongcheng, and Z. Qiang. Data-driven PID controller design for continuous stirred tank reactor. In *34th Chinese Control Conference (CCC)*, pages 251–254, 2015.

[84] A. Doshi and M. M. Trivedi. Tactical driver behavior prediction and intent inference: A review. In *IEEE Conference on Intelligent Transportation Systems,*, pages 1892–1897, Washington, DC, USA, 2011.

[85] D. Dovžan, S. Blažic, and I. Škrjanc. Towards evolving fuzzy reference controller. In *IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, pages 1–8, Linz, Austria, 2014.

[86] D. Dovžan, V. Logar, and I. Škrjanc. Solving the sales prediction problem with fuzzy evolving methods. In *IEEE World Congress on Computational Intelligence (WCCI)*, pages 10–15, Brisbane, Australia, 2012.

[87] D. Dovžan, V. Logar, and I. Škrjanc. Implementation of an Evolving Fuzzy Model (eFuMo) in a Monitoring System for a Waste-Water Treatment Process. *IEEE Transactions on Fuzzy Systems*, 23(5):1761–1776, 2015.

[88] D. Dovžan and I. Škrjanc. Predictive functional control based on an adaptive fuzzy model of a hybrid semi-batch reactor. *Control Engineering Practice*, 18(8):979–989, 2010.

[89] D. Dovžan and I. Škrjanc. Recursive clustering based on a Gustafson-Kessel algorithm. *Evolving Systems*, 2(1):15–24, 2011.

[90] D. Dovžan and I. Škrjanc. Recursive fuzzy c-means clustering for recursive fuzzy identification of time-varying processes. *ISA Transactions*, 50(2):159–169, 2011.

[91] J. J. Downs and E. F. Vogel. A plant-wide industrial process control problem. *Computers and Chemical Engineering*, 17(3):245–255, 1993.

[92] M. J. Er and S. Mandal. A Survey on Adaptive Fuzzy Controllers: Nonlinearities and Classifications. *IEEE Transactions on Fuzzy Systems*, 24(5):1095–1107, 2016.

[93] G. Feng. A Survey on Analysis and Design of Model-Based Fuzzy Control Systems. *Fuzzy Systems, IEEE Transactions on*, 14(5):676–697, 2006.

[94] a. Flores, D. Saez, J. Araya, M. Berenguel, and A. Cipriano. Fuzzy predictive control of a solar power plant. *IEEE Transactions on Fuzzy Systems*, 13(1):58–68, 2005.

[95] P. M. Frank. Fault diagnosis in dynamic systems using analytical knowledge-based redundancy - A survey and some new results. *Automatica*, 26(3):459–474, 1990.

[96] J. Gama. *Knowledge Discovery from Data Streams*. Chapman & Hall/CRC, 1st edition, 2010.

[97] J. Gertler and J. Cao. PCA-Based Fault Diagnosis in the Presence of Control and Dynamics. *AIChE Journal, Process Systems Engineering*, 50(2):388–402, 2004.

[98] J. J. Gertler. *Fault Detection and Diagnosis in Engineering Systems*. CRC Press, New York, USA, 1998.

[99] M. Glavan, D. Gradišar, M. Atanasijević-Kunc, S. Strmčnik, and G. Mušič. Input variable selection for model-based production control and optimisation. *International Journal of Advanced Manufacturing Technology*, 68(9-12):2743–2759, 2013.

[100] M. Glavan, D. Gradišar, S. Strmčnik, and G. Mušič. Production modelling for holistic production control. *Simulation Modelling Practice and Theory*, 30:1–20, 2012.

[101] D. Gradišar, M. Glavan, B. Hauptman, and G. Mušič. Model tehnološke in stroškovne učinkovitosti proizvodnega procesa. Technical report, Kompetenčni center za sodobne tehnologije vodenja, Ljubljana, 2013.

[102] P. Gupta, V. Kumar, K. P. S. Rana, and P. Mishra. Velocity and Position forms of Self-tuning Fuzzy PI Controller Applied to Jacketed CSTR. In *Annual IEEE India Conference (INDICON)*, pages 1–6, 2015.

[103] D. E. Gustafson, W. C. Kessel, and S. Systems. Fuzzy clustering with a fuzzy covariance matrix. In *1978 IEEE Conference on Decision and Control including the 17th Symposium on Adaptive Processes*, pages 761–766, San Diego, CA, USA, 1979.

[104] H. A. Hagras. A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots. *IEEE Transactions on Fuzzy Systems*, 12(4):524–539, 2004.

[105] L. Hartert, M. S. Mouchaweh, and P. Billaudel. A semi-supervised dynamic version of Fuzzy K-Nearest Neighbours to monitor evolving systems. *Evolving Systems*, 1(1):3–15, 2010.

[106] B. Hartmann, O. Banfer, O. Nelles, A. Sodja, L. Teslič, and I. Škrjanc. Supervised Hierarchical Clustering in Fuzzy Model Identification. *IEEE Transactions on Fuzzy Systems*, 19(6):1163–1176, 2011.

[107] B. Hassibi and D. G. Stork. Second order derivaties for network prunning: Optimal Brain Surgeon. In *Advances in Neural Information Processing Systems 5*, pages 164–171, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.

[108] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, volume 2. Springer, 2nd edition, 2009.

[109] D. He, R. Li, and J. Zhu. Plastic Bearing Fault Diagnosis Based on a Two-Step Data Mining Approach. *IEEE Transactions on Industrial Electronics*, 60(8):3429–3440, 2013.

[110] Q. P. He, S. J. Qin, and J. Wang. A new fault diagnosis method using fault directions in Fisher discriminant analysis. *AIChE Journal*, 51(2):555–571, 2005.

[111] H. Henderson and S. Searle. On Deriving the Inverse of a Sum of Matrices. *Society for Industrial and Applied Mathematics*, 23(1):53–60, 2008.

[112] W. K. Ho, K. C. Tan, A. Tay, and R. Srinivasan. Using the OPC standard for real-time process monitoring and control. *IEEE Software*, 22(6):54–59, 2005.

[113] S. Huang, K. K. Tan, and T. H. Lee. Fault diagnosis and fault-tolerant control in linear drives using the Kalman filter. *IEEE Transactions on Industrial Electronics*, 59(11):4285–4292, 2012.

[114] J. A. Iglesias, P. Angelov, A. Ledezma, and A. Sanchis. Evolving classification of agents' behaviors: A general approach. *Evolving Systems*, 1(3):161–171, 2010.

[115] J. A. Iglesias, A. Ledezma, and A. Sanchis. Caos Coach 2006 Simulation Team: An opponent modelling approach. *Computing and Informatics*, 28(1):57–80, 2009.

[116] P. Ioannou and B. Fidan. *Adaptive Control Tutorial*, volume 1. Society for Industrial and Applied Mathematics, 2006.

[117] P. Ioannou and P. Kokotovic. Instability analysis and improvement of robustness of adaptive control. *Automatica*, 20(5):583–594, 1984.

[118] P. Ioannou and J. Sun. *Robust Adaptive Control*. PTR Prentice-Hall, 1996.

[119] P. Ioannou and K. Tsakalis. A robust direct adaptive controller. *IEEE Transactions on Automatic Control*, 31(11):1033–1043, 1986.

[120] R. Isermann. Model-based Fault Detection and Diagnosis - Status and Applications. *IFAC Automatic Control in Aerospace*, 29(c):71–85, 2004.

[121] R. Isermann. *Fault-diagnosis systems: An introduction from fault detection to fault tolerance*. Springer- Verlag Berlin Heidelberg, 2006.

[122] R. Isermann. *Fault-Diagnosis Applications*. Springer-Verlag Berlin Heidelberg 2011, 2011.

[123] R. Isermann and M. Münchhof. *Identification of Dynamic Systems*. Springer-Verlag Berlin Heidelberg GmbH, 2011.

[124] J. S. R. Jang. ANFIS: Adaptive-Network-Based Fuzzy Inference System. *IEEE Transactions on Systems, Man and Cybernetics*, 23(3):665–685, 1993.

[125] U. Javed, M. M. Riaz, A. Ghafoor, and T. A. Cheema. Local Features and Takagi-Sugeno Fuzzy Logic based Medical Image Segmentation. *Radio Engineering*, 22(4):1091–1097, 2013.

[126] G. M. J. Jerome, A. K. P, and K. Anadhan. Comparative Performance Analysis of Extended Kalman Filter and Neural Observer for State Estimation of Continuous Stirred Tank Reactor. In *2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, pages 1–7, Tiruchengode, India, 2013. IEEE.

[127] K. Jouili and H. Jerbi. A Computationally Lyapunov Nonlinear Gain Scheduling Control of Nonlinear Systems with Stability Guarantees. In *1th International Conference on Computer Modelling and Simulation*, pages 374–379, 2009.

[128] C.-f. Juang and Y.-w. Tsao. A Self-Evolving Interval Type-2 Fuzzy Neural Network With Online Structure. *IEEE Transactions on Fuzzy Systems*, 16(6):1411–1424, 2008.

[129] A. Kalhor, B. N. Araabi, and C. Lucas. An online predictor model as adaptive habitually linear and transiently nonlinear model. *Evolving Systems*, 1(1):29–41, 2010.

[130] A. Kandel, O. Manor, Y. Klein, and S. Fluss. ATM traffic management and congestion control using fuzzy logic. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 29(3):474–480, 1999.

[131] M. Kano, S. Tanaka, S. Hasebe, I. Hashimoto, and H. Ohno. Monitoring independent components for fault detection. *AIChE Journal*, 49(4):969–976, 2003.

[132] Y. Kansha, L. Jia, and M. S. Chiu. Self-tuning PID controllers based on the Lyapunov approach. *Chemical Engineering Science*, 63(10):2732–2740, 2008.

[133] G. Karer and I. Škrjanc. *Predictive approaches to control of complex systems*. Springer-Verlag Berlin Heidelberg, 2013.

[134] G. Karer, I. Škrjanc, and B. Zupančič. Self-adaptive predictive functional control of the temperature in an exothermic batch reactor. *Chemical Engineering and Processing: Process Intensification*, 47(12):2379–2385, 2008.

[135] N. Kasabov. Evolving fuzzy neural networks — algorithms, applications and biological motivation. *Methodologies for the Conception, Design and Application of Soft Computing*, pages 271–274, 1998.

[136] N. Kasabov. Evolving Fuzzy Neural Networks for Supervised/Unsupervised Online Knowledge-Based Learning. *IEEE Transactions on Systems, Man and Cybernetics*, 31(6):902–918, 2001.

[137] N. Kasabov. *Evolving connectionist systems: The knowledge engineering approach*. Springer, London, 2nd edition, 2007.

[138] N. K. Kasabov and Q. Song. DENFIS : Dynamic Evolving Neural-Fuzzy Inference System and Its Application for Time-Series Prediction. *IEEE Transactions on Fuzzy Systems*, 10(2):144–154, 2002.

[139] H. Kaufman, S. Kenneth, and I. Barkana. *Direct Adaptive Control Aigorithms Theory and Applications*. Springer-Verlag New York, Inc., 2nd edition, 1998.

[140] S.-R. Ke, H. Thuc, Y.-J. Lee, J.-N. Hwang, J.-H. Yoo, and K.-H. Choi. A Review on Video-Based Human Activity Recognition. *Computers*, 2(2):88–131, 2013.

[141] K. Kiguchi and Y. Hayashi. An EMG-based control for an upper-limb power-assist exoskeleton robot. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 42(4):1064–1071, 2012.

[142] G. Klančar. Fault Detection and Isolation by means of Principal Component Analysis. In *Cybernetics & Informatics Eurodays: Young Generation Viewpoint : PhD Workshop,*, pages 1–6, Marianska, Czech Republic, 2000. Institute of Information Theory and Automation.

[143] G. Klančar, a. Juričić, and R. Karba. Robust fault detection based on compensation of the modelling error. *International Journal of Systems Science*, 33(2):97–105, 2002.

[144] G. Klančar and I. Škrjanc. Metoda glavnih komponent pri odkrivanju in izolaciji napak: primer hidravličnega procesa in procesa fermentacije. *Elektrotehniški vestnik*, 69(5):311–316, 2002.

[145] J. Kmetova, A. Vasickaninova, and J. Dvoran. Neuro-fuzzy control of exothermic chemical reactor. *Proceedings of the 2013 International Conference on Process Control, PC 2013*, pages 168–172, 2013.

[146] H. S. Ko and T. Niimura. Power system stabilization using fuzzy-neural hybrid intelligent control. In *Proceedings of the 2002 Ieee International Symposium on Intelligent Control*, pages 879–884, 2002.

[147] J. Kocijan and D. Petelin. Closed-loop control with evolving Gaussian process models. In *Complex Systems: Relationships between Control, Communications and Computing*, volume 55, pages 505–521. Springer International Publishing, 2016.

[148] B. Kosko. Fuzzy Systems as Universal Approximators. *IEEE Transactions on Computers*, 43(11):1329–1333, 1994.

[149] G. Kreisselmeier and K. Narendra. Stable model reference adaptive control in the presence of bounded disturbances. *IEEE Transactions on Automatic Control*, 27(6):1169–1175, 1982.

[150] S. S. Kumar and C. Shreesha. Design of Robust PID Controller for a CSTR Plant with Interval Parametric Uncertainty using Kharitonov theorem. In *2016 International Conference on Computation of Power, Energy Information and Commuincation (ICCPEIC)*, pages 430–433, 2016.

[151] T. Larsson, K. Hestetun, E. Hovland, and S. Skogestad. Self-optimizing control of a large-scale plant: The Tennessee Eastman process. *Industrial and Engineering Chemistry Research*, 40(22):4889–4901, 2001.

[152] A. Ledezma, R. Aler, A. Sanchis, and D. Borrajo. OMBO: An opponent modeling approach. *AI Communications*, 22(1):21–35, 2009.

[153] J.-M. Lee and S. J. Qin. Fault Detection and Diagnosis Based on Modified Independent Component Analysis. *AIChE Journal*, 52(10):3501–3514, 2006.

[154] D. Leite, R. Ballini, P. Costa, and F. Gomide. Evolving fuzzy granular modeling from nonstationary fuzzy data streams. *Evolving Systems*, 3(2):65–79, 2012.

[155] D. Leite, R. M. Palhares, V. C. S. Campos, and F. Gomide. Evolving Granular Fuzzy Model-Based Control of Nonlinear Dynamic Systems. *IEEE Transactions on Fuzzy Systems*, 23(4):923–938, 2015.

[156] A. Lemos, W. Caminhas, and F. Gomide. Multivariable gaussian evolving fuzzy modeling system. *IEEE Transactions on Fuzzy Systems*, 19(1):91–104, 2011.

[157] A. Lemos, W. Caminhas, and F. Gomide. Adaptive fault detection and diagnosis using an evolving fuzzy classifier. *Information Sciences*, 220:64–85, 2013.

[158] G. Leng, X.-J. Zeng, and J. a. Keane. An improved approach of self-organising fuzzy neural network based on similarity measures. *Evolving Systems*, 3(1):19–30, 2012.

[159] M. Lepetič, I. Škrjanc, H. Chiacchiarini, and D. Matko. Predictive control based on fuzzy model: a case study. *10th IEEE International Conference on Fuzzy Systems*, 2:868–871, 2001.

[160] G. Li, S. J. Qin, and D. Zhou. Geometric properties of partial least squares for process monitoring. *Automatica*, 46(1):204–210, 2010.

[161] R. Li and X. Wang. Dimension reduction of process dynamic trends using independent component analysis. *Computers & Chemical Engineering*, 26(3):467–473, 2002.

[162] W. Li, H. Yue, S. Valle-Cervantes, and S. J. Qin. Recursive PCA for adaptive process monitoring. *Journal of Process Control*, 10:471–486, 2000.

[163] E. Lima, M. Hell, R. Ballini, and F. Gomide. Evolving Fuzzy Modeling Using Participatory Learning. In P. Angelov, D. Filev, and N. Kasabov, editors, *Evolving Intelligent Systems: Methodology and Applications*, chapter 4, pages 67–86. John Wiley & Sons, New Jersey, 1 edition, 2010.

[164] X. Liu, F. Xu, and K. Fujimura. Real-Time Eye Detection and Tracking for Driver Observation Under Various Light Conditions. In *Intelligent Vehicle Symposium*, pages 344–351, 2002.

[165] L. Ljung. *System Identification: Theory for the User*. Prentice Hall, New Jersey, 1987.

[166] E. Lughofer. *Evolving fuzzy systems Methodologies, advanced oncepts and applications*. Springer-Verlag Berlin Heidelberg, 1st edition, 2011.

[167] E. Lughofer. Flexible Evolving Fuzzy Inference Systems from Data Streams (FLEXFIS++). In M. Sayed-Mouchaweh and E. Lughofer, editors, *Learning in Non-Stationary Environments: Methods and Applications*, pages 205–246. Springer, New York, 2012.

[168] E. Lughofer. Hybrid active learning for reducing the annotation effort of operators in classification systems. *Pattern Recognition*, 45(2):884–896, 2012.

[169] E. Lughofer. Evolving Fuzzy Systems - Fundamentals, Reliability, Interpretability, Useability, Applications. In P. P. Angelov, editor, *Handbook on Computational Intelligence*, chapter 3, pages 67–136. Springer Verlag, Berlin Heidelberg, 2015.

[170] E. Lughofer and E. P. Klement. FLEXFIS: A Variant for Incremental Learning of Takagi-Sugeno Fuzzy Systems. In *Proceedings of The 2005 IEEE International Conference on Fuzzy Systems*, pages 915–920, 2005.

[171] E. D. Lughofer. FLEXFIS: A robust incremental learning approach for evolving Takagi-Sugeno fuzzy models. *IEEE Transactions on Fuzzy Systems*, 16(6):1393–1410, 2008.

[172] Y. Ma, F. Borrelli, B. Hencey, B. Coffey, S. Bengea, and P. Haves. Model Predictive Control for the Operation of Building Cooling Systems. *IEEE Transactions on Control Systems Technology*, 20(3):796–803, 2012.

[173] J. A. Machado. Optimal tuning of fractional controllers using genetic algorithms. *Nonlinear Dynamics*, 62(1-2):447–452, 2010.

[174] J. Macias-Hernandez and P. Angelov. Applications of Evolving Intelligent Systems to Oil and Gas Industry. In P. Angelov, D. Filev, and N. Kasabov, editors, *Evolving Intelligent Systems: Methodology and Applications*, chapter 17, pages 401–421. John Wiley & Sons, New Jersey, 1 edition, 2010.

[175] C. R. Madhuranthakam, A. Elkamel, and H. Budman. Optimal tuning of PID controllers for FOPTD, SOPTD and SOPTD with lead processes. *Chemical Engineering and Processing: Process Intensification*, 47(2):251–264, 2006.

[176] E. Mamdani. Application of fuzzy algorithms for control of simple dynamic plant. *Proceedings of the Institution of Electrical Engineers*, 121(12):1585, 1974.

[177] E. Mamdani and S. Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, 7(1):1–13, 1975.

[178] F. Matia, G. N. Marichal, and E. Jimenez. *Fuzzy modeling and control: Theory and Applications*. Atlantis press, 2014.

[179] J. A. Michon. A Critical View of Driver Behavior Models: What Do We Know, What Should We Do? In *Human behavior and traffic safety*, pages 485–520. Plenum press, New York, London, 1985.

[180] C. Miyajima, Y. Nishiwaki, K. Ozawa, T. Wakita, K. Itou, K. Takeda, and F. Itakura. Driver modeling based on driving behavior and its evaluation in driver identification. *Proceedings of the IEEE*, 95(2):427–437, 2007.

[181] T. B. Moeslund, A. Hilton, and V. Kruger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2-3 SPEC. ISS.):90–126, 2006.

[182] A. Moharam, M. A. El-Hosseini, and H. A. Ali. Design of optimal PID controller using hybrid differential evolution and particle swarm optimization with an aging leader and challengers. *Applied Soft Computing Journal*, 38:727–737, 2016.

[183] Y. Mousavi and A. Alfi. A memetic algorithm applied to trajectory control by tuning of Fractional Order Proportional-Integral-Derivative controllers. *Applied Soft Computing Journal*, 36:599–617, 2015.

[184] B. Mu, Y. Li, T. I. Salsbury, and J. M. House. Optimization and Sequencing of Chilled-water Plant Based on Extremum Seeking Control *. In *2016 American Control Conference (ACC)*, pages 2373–2378, Boston, MA, USA, 2016.

[185] A. Nakano, H. Okuda, T. Suzuki, S. Inagaki, and S. Hayakawa. Symbolic Modeling of Driving Behavior based on Hierarchical Mode Segmentation and Formal Grammar. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5516–5521, St. Louis, USA, 2009.

[186] K. S. Narendra and A. M. Annaswamy. A New Adaptive Law for Robust Adaptation without Persistent Excitation. *1986 American Control Conference*, c:134–145, 1987.

[187] O. Nelles. *Nonlinear System Identification*. Springer-Verlag Berlin Heidelberg, 1st edition, 2001.

[188] O. Nelles and R. Isermann. Basis Function Networks for Interpolation of Local Linear Modells. *Proceedings of 35th IEEE Conference on Decision and Control*, 9(December):470–475, 1996.

[189] M. Norgaard, O. Ravn, N. Poulsen, and L. Hansen. *Neural networks for modelling and control of dynamic systems: A Practitioner's Handbook*, volume 1. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2000.

[190] M. Noruzi Nashalji, M. Aliyari Shoorehdeli, and M. Teshnehlab. Fault Detection of the Tennessee Eastman Process Using Improved PCA and Neural Classifier. In X.-Z. Gao, A. Gaspar-Cunha, M. Köppen, G. Schaefer, and J. Wang, editors, *Advances in Intelligent and Soft Computing*, pages 41–50. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[191] D. Đorđević, D. Kukolj, and P. L. Callet. 3D Image Quality Assessment Using Takagi-Sugeno-Kang Fuzzy Model. In *IEEE 12th International Symposium on Intelligent Systems and Informatics*, pages 309–314, Subotica, 2014.

[192] M. Panou, E. Bekiaris, and V. Papakostopoulos. Modelling Driver Behaviour in European and International Projects. In P. C. Cacciabue, editor, *Modelling Driver Behaviour in Automotive Environments: Critical Issues in Driver Interactions with Intelligent Transport Systems*, pages 3–25. Springer London, London, 2007.

[193] D. Petelin and J. Kocijan. Control system with evolving Gaussian process models. In *IEEE SSCI 2011: Symposium Series on Computational Intelligence - EAIS 2011: IEEE Workshop on Evolving and Adaptive Intelligent Systems*, pages 178–184, 2011.

[194] B. B. Peterson and K. S. Narendra. Bounded Error Adaptive Control. *IEEE Transactions on Automatic Control*, 27(6):1161–1168, 1982.

[195] A. Piegat. *Fuzzy Modeling and Control*. Springer-Verlag Berlin Heidelberg GmbH, 2001.

[196] I. Podlubny. Fractional-order systems and PI/sup /spl lambda//D/sup /spl mu//-controllers. *IEEE Transactions on Automatic Control*, 44(1):208–214, 1999.

[197] M. Pratama, S. G. Anavatti, P. P. Angelov, and E. Lughofer. PANFIS: A novel incremental learning machine. *IEEE Transactions on Neural Networks and Learning Systems*, 25(1):55–68, 2014.

[198] M. Pratama, S. G. Anavatti, and E. Lughofer. Genefis: Toward an effective localist network. *IEEE Transactions on Fuzzy Systems*, 22(3):547–562, 2014.

[199] R.-E. Precup, P. Angelov, B. S. J. Costa, and M. Sayed-Mouchaweh. An overview on fault diagnosis and nature-inspired optimal control of industrial process applications. *Computers in Industry*, 74:75–94, 2015.

[200] R.-E. Precup, M.-B. Radac, E. M. Petriu, and R.-C. Roman. Evolving Fuzzy Models for the Position Control of Twin Rotor Aerodynamic Systems. In *2016 IEEE 14th International Conference on Industrial Informatics (INDIN)*, pages 237–242, 2016.

[201] R.-E. Precup, T.-A. Teban, T. E. A. de Oliveira, and E. M. Petriu. Evolving Fuzzy Models for Myoelectric-based Control of a Prosthetic Hand. In *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 72–77, 2016.

[202] S. J. Qin. Statistical process monitoring: basics and beyond. *Journal of Chemometrics*, 17(8-9):480–502, 2003.

[203] S. J. Qin and T. A. Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11:733–764, 2003.

[204] M. Radeerom and K. Tharathanmathikorn. Intelligent system based supervision for energy management of water chiller plant. In *2015 12th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, pages 1–6, 2015.

[205] A. Ragab, M. El-Koujok, M. Amazouz, and S. Yacout. Fault detection and diagnosis in the Tennessee Eastman Process using interpretable knowledge

discovery. In *Annual Reliability and Maintainability Symposium (RAMS)*, pages 1–7, Orlando, FL, USA, 2017. IEEE.

[206] L. E. Ramos-Velasco, O. A. Domínguez-Ramírez, and V. Parra-Vega. Wavenet fuzzy PID controller for nonlinear MIMO systems: Experimental validation on a high-end haptic robotic interface. *Applied Soft Computing Journal*, 40:199–205, 2016.

[207] T. J. Rato and M. S. Reis. Fault detection in the Tennessee Eastman benchmark process using dynamic principal components analysis based on decorrelated residuals (DPCA-DR). *Chemometrics and Intelligent Laboratory Systems*, 125:101–108, 2013.

[208] M. M. Riaz and A. Ghaforr. Spectral and textural weighting using Takagi-Sugeno fuzzy system for through wall image inhancement. *Progress In Electromagnetics Research B*, 48:115–130, 2013.

[209] J. Richalet. Industrial applications of model based predictive control. *Automatica*, 29(5):1251–1274, 1993.

[210] N. L. Ricker. Decentralized control of the Tennessee Eastman Challenge Process. *Journal of Process Control*, 6(4):205–221, 1996.

[211] I. Rivals and L. Personnaz. Neural-network construction and selection in nonlinear modeling. *IEEE Transactions on Neural Networks*, 14(4):804–819, 2003.

[212] C. Riverol and V. Napolitano. Application in a Heat Exchanger. *Changes*, 78(November):1115–1119, 2000.

[213] C. Rohrs, L. Valavani, M. Athans, and G. Stein. Robustness of continuous-time adaptive control algorithms in the presence of unmodeled dynamics. *IEEE Transactions on Automatic Control*, 30(9):881–889, 1985.

[214] H.-J. Rong, S. Han, and G.-S. Zhao. Adaptive fuzzy control of aircraft wing-rock motion. *Applied Soft Computing*, 14:181–193, 2014.

[215] H. J. Rong, N. Sundararajan, G. B. Huang, and P. Saratchandran. Sequential Adaptive Fuzzy Inference System (SAFIS) for nonlinear system identification and prediction. *Fuzzy Sets and Systems*, 157(9):1260–1275, 2006.

[216] H. J. Rong, N. Sundararajan, G. B. Huang, and G. S. Zhao. Extended sequential adaptive fuzzy inference system for classification problems. *Evolving Systems*, 2(2):71–82, 2011.

[217] J. D. J. Rubio. SOFMLS: Online self-organizing fuzzy modified least-squares network. *IEEE Transactions on Fuzzy Systems*, 17(6):1296–1309, 2009.

[218] P. Sarma. Multivariable gain-scheduled fuzzy logic control of an exothermic reactor. *Engineering Applications of Artificial Intelligence*, 14(4):457–471, 2001.

[219] A. Sathyanarayana, P. Boyraz, and J. H. Hansen. Driver behavior analysis and route recognition by Hidden Markov Models. In *2008 IEEE International Conference on Vehicular Electronics and Safety*, pages 276–281, Columbus, OH, USA, 2008.

[220] M. Sayed-Mouchaweh and E. Lughofer. *Learning in Non-Stationary Environments: Methods and Applications*. Springer Publishing Company, New York, 2012.

[221] D. E. Seborg, T. F. Edgar, and D. A. Mellichamp. *Process Dynamics and Control*. John Wiley & Sons, 2nd edition, 2004.

[222] E. D. Seborg and A. M. Henson. Introduction. In A. M. Henson and E. D. Seborg, editors, *Nonlinear Process Control*, pages 5–7. Prentice Hall, 1st edition, 1997.

[223] F. Serdio, E. Lughofer, K. Pichler, T. Buchegger, M. Pichler, and H. Efendic. Fault detection in multi-sensor networks based on multivariate time-series models and orthogonal transformations. *Information Fusion*, 20(1):272–291, 2014.

[224] Q. M. Shao and A. Cinar. System identification and distributed control for multi-rate sampled systems. *Journal of Process Control*, 34:1–12, 2015.

[225] B. Shi, L. Xu, J. Hu, Y. Tang, H. Jiang, W. Meng, and H. Liu. Evaluating Driving Styles by Normalizing Driving Behavior Based on Personalized

Driver Modeling. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(12):1502–1508, 2015.

[226] M. Shyamalagowri and R. Rajeswari. Neural Network Predictive Controller Based Nonlinearity Identification Case Study: Nonlinear Process Reactor - CSTR. *Advanced Materials Research*, 984-985:1326–1334, 2014.

[227] M. H. Sigari, M. Fathy, and M. Soryani. A driver face monitoring system for fatigue and distraction detection. *International Journal of Vehicular Technology*, 2013:1–11, 2013.

[228] A. Silva, W. Caminhas, A. Lemos, and F. Gomide. Real-time nonlinear modeling of a twin rotor MIMO system using evolving neuro-fuzzy network. In *IEEE Symposium on Computational Intelligence in Control and Automation (CICA)*, pages 1–8, 2014.

[229] I. Škrjanc, S. Blažič, and P. Angelov. Robust Evolving Cloud-based PID Control Adjusted by Gradient Learning Method. *2014 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, pages 1–8, 2014.

[230] I. Škrjanc, A. S. de Miguel, J. A. Iglesias, A. Ledezma, and D. Dovžan. Evolving Cauchy possibilistic clustering based on cosine similarity for monitoring cyber systems. In *IEEE International Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, pages 1–5, Ljubljana, 2017.

[231] I. Škrjanc, M. Lepetič, J. L. Figueroa, and S. Blažič. Fuzzy model-based predictive control for a CSTR with multiple steady state: A simulation study and a comparison with other nonlinear MBPC control algorithms. *Transactions on Systems WSEAS*, 3(2):789–794, 2004.

[232] I. Škrjanc and D. Matko. Predictive functional control based on fuzzy model for heat-exchanger pilot plant. *IEEE Transactions on Fuzzy Systems*, 8(6):705–712, 2000.

[233] T. Steffens. Feature-Based Declarative Opponent-Modelling. In D. Polani, B. Browning, A. Bonarini, and K. Yoshida, editors, *RoboCup 2003: Robot Soccer World Cup VII*, pages 125–136. Springer, Berlin, Heidelberg, 2004.

[234] M. Stepančič and J. Kocijan. On-line identification with regularised Evolving Gaussian process. In *Evolving and Adaptive Intelligent Systems (EAIS)*, pages 1–7, Ljubljana, Slovenia, 2017.

[235] N. Storey. *Safety-critical computer systems*. Addison-Wesley, 1996, 1996.

[236] G. Sukthankar, C. Geib, H. Hai Bui, D. Pynadath, and R. P. Goldman. An Introduction to Plan, Activity, and Intent Recognition. In G. Sukthankar, R. P. Goldman, C. Geib, D. V. Pynadath, and H. H. Bui, editors, *Plan, Activity, and Intent Recognition*, page 22. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2014.

[237] Systems Technology. Car driving simulator & simulation software, 2016.

[238] T. Takagi and M. Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-15(1):116–132, 1985.

[239] J. A. Tenreiro Machado, A. M. Galhano, A. M. Oliveira, and J. K. Tar. Optimal approximation of fractional derivatives through discrete-time fractions using genetic algorithms. *Communications in Nonlinear Science and Numerical Simulation*, 15(3):482–490, 2010.

[240] Tiesheng Wang, Pengfei Shi, T. Wang, and P. Shi. Yawning detection for determining driver drowsiness. In *Proceedings of International Workshop on VLSI Design and Video Technology*, pages 373–376, 2005.

[241] K. Torkkola, M. Gardner, C. Schreiner, K. Zhang, B. Leivian, and H. Zhang. Understanding Driving Activity Using Ensemble Methods. In D. Prokhorov, editor, *Computational Intelligence in Automotive Applications*, volume 132, pages 39–58. Springer Berlin Heidelberg, 2008.

[242] K. Torkkola, S. Venkatesan, and H. Liu. Sensor Sequence Modeling for Driving Sequence Modeling for Driving. In *FLAIRS Conference*, pages 721–727. AAAI Press, 2005.

[243] D.-M. Tsai, S.-C. Wu, and W.-Y. Chiu. Defect Detection in Solar Modules Using ICA Basis Images. *IEEE Transactions on Industrial Informatics*, 9(1):122–131, 2013.

[244] S. W. Tung, C. Quek, and C. Guan. ET2FIS: An evolving type-2 neural fuzzy inference system. *Information Sciences*, 220:124–148, 2013.

[245] V. Venkatasubramanian, R. Rengaswamy, and S. N. Kavuri. A review of process fault detection and diagnosis part II: Qualitative models and search strategies. *Computers and Chemical Engineering*, 27(3):313–326, 2003.

[246] V. Venkatasubramanian, R. Rengaswamy, S. N. Kavuri, and K. Yin. Areview of process fault detection and diagnosis Part III: Process history based methods. *Computers & Chemical Engineering*, 27(3):327–346, 2003.

[247] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri. A review of process fault detection and diagnosis Part I: Quantitative model-based methods. *Computers & Chemical Engineering*, 27(3):293–311, 2003.

[248] A. Vernaza, A. Ledezma, and A. Sanchis. Simul-A2: Agent-based Simulator for evaluate ADA Systems. In *17th International Conference on Information Fusion (FUSION)*, pages 1–7, Salamanca, Spain, 2014.

[249] W.A.Cebushar and V. Costanza. Nonlinear control of CSTR's. *Chemical Engineering Science*, 39(12):1715–1722, 1984.

[250] D. Wang, X.-J. Zeng, and J. A. Keane. A structure evolving learning method for fuzzy systems. *Evolving Systems*, 1(2):83–95, 2010.

[251] D. Wang, X. J. Zeng, and J. A. Keane. An evolving-construction scheme for fuzzy systems. *IEEE Transactions on Fuzzy Systems*, 18(4):755–770, 2010.

[252] D. Wang, X. J. Zeng, and J. A. Keane. A simplified structure evolving method for Mamdani fuzzy system identification and its application to high-dimensional problems. *Information Sciences*, 220:110–123, 2012.

[253] L.-X. Wang. Fuzzy Systems are Universal Approximators. In *IEEE International Conference on Fuzzy Systems*, pages 1163–1170, 1992.

[254] W. Wang and J. Vrbanek. An evolving fuzzy predictor for industrial applications. *IEEE Transactions on Fuzzy Systems*, 16(6):1439–1449, 2008.

[255] W. Wang, J. Xi, and H. Chen. Modeling and recognizing driver behavior based on driving data: A survey. *Mathematical Problems in Engineering*, 2014:1–20, 2014.

[256] X. Wang, U. Kruger, and B. Lennox. Recursive partial least squares algorithms for monitoring complex industrial processes. *Control Engineering Practice*, 11(6):613–632, 2003.

[257] Y.-x. Wang, F.-f. Qin, K. Ou, Y.-b. Kim, and A. E. Model. Temperature Control for a Polymer Electrolyte Membrane Fuel Cell by Using Fuzzy Rule. *IEEE Transactions on Energy Conversion*, 31(2):1–9, 2016.

[258] G. Webb, M. Pazzani, and D. Billsus. Machine learning for user modeling. *User Modeling and User-Adapted Interaction*, 11(1):19–29, 2001.

[259] D. J. Wilson and G. W. Irwin. PLS modelling and fault detection on the Tennessee Eastman benchmark. *International Journal of Systems Science*, 31(11):1449–1457, 2000.

[260] H. Xie, J. Wang, and X. Tang. Robust model predictive control of uncertain discrete-time T-S fuzzy systems. *2016 Chinese Control and Decision Conference (CCDC)*, pages 4066–4071, 2016.

[261] R. R. Yager and D. P. Filev. Learning of fuzzy rules by mountain clustering. In *Proceedings of SPIE Conference on Applications of Fuzzy Logic Technolgy*, volume 2061, pages 246–254, 1993.

[262] S. Yilmaz and Y. Oysal. Nonlinear System Modeling and Control with Dynamic Fuzzy Wavelet Neural Network. In *International Symposium on Innovations in Intelligent SysTems and Applications (INISTA)*, pages 1–7, 2015.

[263] S. Yin, S. X. Ding, A. Haghani, H. Hao, and P. Zhang. A comparison study of basic data-driven fault diagnosis and process monitoring methods on the benchmark Tennessee Eastman process. *Journal of Process Control*, 22(9):1567–1581, 2012.

[264] S. Yin, S. X. Ding, X. Xie, and H. Luo. A review on basic data-driven approaches for industrial process monitoring. *IEEE Transactions on Industrial Electronics*, 61(11):6414–6428, 2014.

[265] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.

[266] L. A. Zadeh. Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Transactions on Systems, Man and Cybernetics,*, 3(1):28–44, 1973.

[267] V. Zamora, O. Sipele, A. Ledezma, and A. Sanchis. Intelligent Agents for Supporting Driving Tasks: An Ontology-Based Alarms System. In *Proceedings of the 3rd International Conference on Vehicle Technology and Intelligent Transport Systems*, pages 165–172, Porto, Portugal, 2017. SciTePress.

[268] A. Zdešar, O. Cerman, D. Dovžan, P. Hušek, and I. Škrjanc. Fuzzy control of a helio-crane: comparison of two control approaches. *Journal of Intelligent & Robotic Systems*, 72(3-4):497–515, 2013.

[269] A. Zdešar, D. Dovžan, and I. Škrjanc. Self-tuning of 2 DOF control based on evolving fuzzy model. *Applied Soft Computing*, 19:403–418, 2014.

[270] G. Zhang, B. Cheng, R. Feng, and X. Zhang. A Real-Time Adaptive Learning Method for Driver Eye Detection. In *2008 Digital Image Computing: Techniques and Applications*, pages 300–304, 2008.

[271] J. Zhang and K.-Y. Zhang. Application of tabu search heuristic algorithms for the purpose of energy saving in optimal load distribution strategy for multiple chiller water units. In *IEEE International Conference on Progress in Informatics and Computing (PIC)*, volume 1, pages 303–307, 2010.

[272] J. Zhang and K.-Y. Zhang. Optimal load distribution strategy for multiple chiller water units based on adaptive genetic algorithms. In *Proceedings of 2nd WRI Global Congress on Intelligent Systems, GCIS 2010*, volume 2, pages 5–8, 2010.

[273] R. Zhang, Y. A. Phillis, and S. Member. Fuzzy Control of Queueing Systems with Heterogeneous Servers. *IEEE Transactions on Fuzzy Systems*, 7(1):17–26, 1999.

[274] Y. Zhang and S. J. Qin. Improved Nonlinear Fault Detection Technique and Statistical Analysis. *AIChE Journal, Process Systems Engineering*, 54(12):3207–3220, 2008.

[275] Y. Zhang, H. Zhou, S. J. Qin, and T. Chai. Decentralized fault diagnosis of large-scale processes using multiblock kernel partial least squares. *IEEE Transactions on Industrial Informatics*, 6(1):3–10, 2010.

[276] Y. Zheng, Y. Wang, D. S. H. Wong, and Y. Wang. A time series model coefficients monitoring approach for controlled processes. *Chemical Engineering Research and Design*, 100(May):228–236, 2015.

[277] J. Ziegler and N. Nichols. Optimum settings for automatic controllers. *Transactions of ASME*, pages 759–768, 1993.