

**Univerza v Ljubljani
Fakulteta za elektrotehniko**

Inteligentni sistemi za podporo odločanju

Igor Škrjanc

Ljubljana, 2016

Predgovor

Učbenik je namenjen slušateljem predmeta Inteligentni sistemi za podporo pri odločanju. Poučarek je na intuitivnem razumevanju problematike, ki je z izpeljavami podprt le tam, kjer se zdi to nujno potrebno. V njem so predstavljeni osnovni principi, metode in sistemi za raziskovanje podatkov. Govori o inteligenčnih sistemih, identifikaciji in uporabi nelinearnih sistemov, osnovah statističnih metod, osnovah linearnih optimizacijskih in nelinearnih optimizacijskih metod, metodah globalne optimizacije, metodah rojenja, mehkih in nevronskeih modelih. Predstavljeno je področje identifikacije linearnih in nelinearnih dinamičnih sistemov, ki omogoča izgradnjo modelov za pomoč pri nadaljnjem odločanju. Na modelih temelječe odločanje je predstavljeno v poglavjih o adaptivnem in prediktivnem vodenju.

Raziskovanje podatkov je nujna faza pri gradnji modelov in je eno najbolj rastočih znanstvenih področij v zadnjem času. Vzroki za to so v hitrem razvoju informacijskih in računalniških tehnologij v povezavi z razvojem različnih sistemov za zajem, shranjevanje in manipulacijo s podatki v realnem času. Spremljanje podatkov o delovanju različnih procesov je pomemben del informatizacije in avtomatizacije v realnem življenju. Problem v današnjem času pa je iz velike množice podatkov avtomsatko razbrati informacijo. Pot do tega cilja pa vodi preko zajemanja podatkov, njihove obdelave, do interpretacije. Z raziskovanjem podatkov lahko izboljšamo razumevanje procesov, ki te podatke generirajo, jih lažje spremljamo in se odločamo. Z analizo velikih podatkovnih zbirk lahko iščemo napake v sistemu in jih v naprej predvidimo.

Poleg klasičnih metod raziskovanja podatkov se neprestano pojavljajo nove metode. Zadnje take metode, ki so se zelo močno uveljavile, so metode za raziskovanje velikih baz podatkov (*ang. big data*), metode globokega učenja (*ang. deep learning*) in učenje iz toka podatkov (*ang. learning from data stream*), če omenimo samo nekatere.

Raziskovanje velikih podatkovnih zbirk srečamo na različnih področjih, na področju znanosti, računalništva in tehnike, znanosti o okolju, medicini, ekonomiji in drugod. Tehnologija velikih podatkovnih zbirk omogoča analizo človekovega vedenja, analizo delovanja računalniških sistemov, razpoznavanje človekovega genoma, analizo trgovanja na trgu vrednostnih papirjev, odkrivanje napak v sodobnih avtomatiziranih sistemih in še mnogo drugega. Je tipično interdisciplinarno področje, ki z metodološkega stališča povezuje metode umetne inteligence, strojnega učenja, statistike in podatkovnih baz. Cilj raziskovanja podatkovnih množic je pridobivanje informacije v človeku razumljivi obliki. Podatkovnih množice so navadno prevelike, da bi jih lahko človek s svojim pregledovanjem razumel.

Algoritmi za raziskovanje podatkov morajo informacijo pridobiti, jo preoblikovati in predstaviti v strukturo primerno za nadaljnjo obdelavo ali pa v človeku razumljivo obliko. Glavni postopki pri raziskovanju podatkov so statistična analiza podatkov, predobdelava podatkov za nadaljnjo obdelavo, raziskovanje in iskanje funkcionalnosti med podatki v obliki pravil, regresije in modelov, razvrščanje podatkov v razrede, iskanje anomalij v podatkih, stiskanje in vizualizacija podatkov.

Raziskovanje podatkov je pomemben korak do odkrivanja znanja v podatkih (*ang. knowledge discovery in databases* - KDD). Poleg klasičnih Bayesovih metod, metod regresijske analize

(*ang. regression*), statistike, optimizacije in klasične linearne in nelinearne identifikacije, so se na področju raziskovanja podatkov uveljavile umetne nevronske mreže (*ang. artificial neural networks*), mehki sistemi (*ang. fuzzy systems*), genetski algoritmi (*ang. genetic algorithms*), metode hierarhičnega in nehierarhičnega rojenja, metode podpornih vektorjev (*ang. support vector machines*) in algoritmi strojnega učenja (*ang. machine learning*).

Vsebina je delno povzeta po knjigi Nonlinear System Identifications avtorja Oliverje Nellesa in je razdeljena na pet delov. V delu I so v prvem poglavju predstavljeni problemi uporabe modelov pri raziskovanju podatkov, v drugem poglavju pa je uvod v statistične metode. V delu II so predstavljene optimizacijske metode. V tretjem poglavju je uvod v optimizacijo, v četrtem poglavju je pregled linearnih optimizacijskih metod, od metode najmanjših kvadratov, do metode uteženih najmanjših kvadratov. V petem poglavju so opisane najpomembnejše metode lokalne nelinearne optimizacije, od gradientnih metod, do metode nelinearnih najmanjših kvadratov. Globalna nelinearna optimizacija je predstavljena v šestem poglavju. Predstavljene so metode simuliranega ohlajanja, optimizacija z roji delcev, evolucijski algoritmi, metoda vejenja in omejevanja in optimizacija s tabu področji. Metode nenadzorovanega učenja so vsebina sedmega poglavja, kjer je predstavljena metoda glavnih komponent, rojenje s k -povprečji, metode mehkega rojenja s točkastimi središči, metode rojenja z linearnimi prototipi in samoorganizirajoče mreže. V osmem poglavju se lotevamo optimizacije kompleksnosti modelov, ki je zelo pomembna za dobro modeliranje v praksi. V delu III so opisane razne arhitekture aproksimacijskih modelov. V desetem poglavju so predstavljeni linearni in polinomski modeli, v enajstem umetne nevronske mreže, mehki modeli v dvanajstem poglavju in lokalni linearni nevro-mehki modeli v trinajstem poglavju. Del IV je namenjen dinamičnim modelom, kjer v štirinajstem poglavju govorimo o identifikaciji linearnih dinamičnih sistemov, v petnjistem pa o identifikaciji nelinearnih dinamičnih sistemov. Uvod v metode odločanja na osnovi modela je predstavljen v delu V. Najprej je podan pregled metod adaptivnega vodenja in nato prediktivno vodenje. Sledi dodatek s pregledom osnovnih pravil matrične algebре, ki je uporabljena v učbeniku in kratek pregled mehke logike.

Ljubljana, september 2016

Igor Škrjanc

Kazalo

Del I. Uvod

1. Uvod v inteligentne sisteme za podporo odločanju	3
1.1 Pomen nelinearne identifikacije sistemov	3
1.1.1 Linearni ali nelinearni problem	3
1.1.2 Predikcija	3
1.1.3 Simulacija	4
1.1.4 Optimizacija	5
1.1.5 Analiza	5
1.1.6 Vodenje	5
1.1.7 Odkrivanje napak	5
1.2 Koraki pri nelinearni identifikaciji	6
1.2.1 Izbera vhodov modela	7
1.2.2 Izbera oblike signalov vzbujanja	8
1.2.3 Izbera arhitekture modela	8
1.2.4 Izbera vrste dinamike modela	8
1.2.5 Izbera reda modela	8
1.2.6 Izbera strukture in kompleksnosti modela	9
1.2.7 Ocena parametrov modela	9
1.2.8 Ocena kvalitete modela	9
1.2.9 Vrednotenje vpliva parametrov metode	9
1.3 Teoretični, eksperimentalni in kombinirani modeli	10
1.4 Terminologija	10
2. Uvod v statistične metode	13
2.1 Deterministične in naključne spremenljivke	13
2.2 Porazdelitev gostote verjetnosti	14
2.3 Stohastični procesi in ergodičnost	16
2.4 Pričakovana vrednost	17
2.5 Varianca	20
2.6 Korelacija in kovarianca	21
2.7 Lastnosti estimatorjev	23
2.8 Koncepti pristranskosti in konvergencije	26

Del II. Optimizacijske metode

3. Uvod v optimizacijo	31
3.1 Pregled optimizacijskih metod	31
3.2 Kriterijske funkcije za nadzorovano učenje	32
3.2.1 Metoda vsote najmanjših kvadratov	32
3.2.2 Metoda največje podobnosti	35
3.2.3 Metoda največje a-posteriori ocene in Bayesova metoda	36
3.3 Kriterijske funkcije pri nenadzorovanem učenju	37
4. Linearna optimizacija	39
4.1 Metoda najmanjših kvadratov (LS)	40
4.1.1 Razlaga metode najmanjših kvadratov	40
4.1.2 Kovariančna matrika estimiranih parametrov	43
4.1.3 Interval zaupanja	45
4.1.4 Ortogonalni regresorji	46
4.1.5 Regularizacija	48
4.1.6 Upoštevanje lastnosti šuma	50
4.1.7 Metoda uteženih najmanjših kvadratov (WLS)	51
4.1.8 Filtrirna jedra	52
4.2 Metoda rekurzivnih najmanjših kvadratov (RLS)	52
4.2.1 Zmanjševanje računske kompleksnosti	54
4.2.2 Identifikacija časovno spremenljivih procesov	55
4.3 Izbira regresorjev	55
4.3.1 Metode za izbiro regresorjev	56
4.3.2 Ortogonalni najmanji kvadrati (OLS) pri izbiri z dodajanjem	58
4.3.3 Regularizacija proti izbiri regresorjev	59
4.4 Povzetek	59
5. Lokalna nelinearna optimizacija	61
5.1 Optimizacija z nizom podatkov in po posameznih podatkih	62
5.2 Inicializacija parametrov	65
5.3 Algoritmi direktnega iskanja optimuma	65
5.4 Gradientne metode optimizacije	65
5.4.1 Metoda končnih differenc	66
5.4.2 Metoda največjega gradiента	67
5.4.3 Newtonova metoda	68
5.4.4 Kvazi-Newtonova metoda	70
5.4.5 Metoda konjugiranih gradientov	71
5.5 Metoda nelinearnih najmanjših kvadratov	72
5.5.1 Gauss-Newtonova metoda	76
5.5.2 Levenberg-Marquardtova metoda	77
6. Globalna nelinearna optimizacija	79
6.1 Simulirano ohlajjanje (SA)	80
6.2 Optimizacija z roji delcev (PSO)	85
6.3 Evolucijski algoritmi (EA)	86
6.3.1 Evolucijska strategija (ES)	89
6.3.2 Genetski algoritmi (GA)	92
6.3.3 Genetsko programiranje (GP)	95

6.4	Metoda vejenja in omejevanja (B&B)	97
6.5	Optimizacija s tabu področji tabuja (TS)	98
6.6	Povzetek	99
7.	Metode nenadzorovanega učenja	101
7.1	Uvod v metode nenadzorovanga učenja	102
7.2	Metoda glavnih komponent	103
7.2.1	Invertiranje matrike s kolinearnimi podatki in regresija na osnovi PCA	106
7.2.2	Sprotni nadzor procesa na osnovi metode PCA	110
7.3	Tehnike rojenja	116
7.4	Mere podobnosti	117
7.5	Algoritem rojenja s k-povprečji	118
7.6	Metode mehkega rojenja s točkastimi središči	121
7.6.1	Mehko rojenje s c-središči (FCM)	121
7.6.2	Rojenje po metodi Gustafson-Kessel (GK)	126
7.6.3	Rojenje po metodi največje podobnosti (FMLE)	129
7.6.4	Rojenje s c-možnimi roji (PCM)	131
7.7	Rojenje z linearimi prototipi - posplošena metoda rojenja	135
7.7.1	Rojenje po metodi glavnih komponent	137
7.8	Kohonenova samoorganizirajoča mreža (SOM)	140
7.9	Povzetek	142
8.	Optimizacija kompleksnosti modela	143
8.1	Uvod	143
8.2	Kompromis med pristranskostjo in varianco	143
8.2.1	Napaka pristranskosti	145
8.2.2	Napaka variance	145
8.2.3	Kompromis med pristranskostjo in napako variance	147
8.3	Ocena kvalitete modela	148
8.3.1	Učenje, preverjanje in potrjevanje modela	148
8.3.2	Navzkrižno preverjanje	150
8.4	Povzetek	151

Del III. Arhitekture aproksimacijskih modelov

9.	Uvod v aproksimacijske modele	155
9.1	Modeli na osnovi baznih funkcij	155
9.1.1	Globalne in lokalne bazne funkcije	155
9.1.2	Linearni in nelinearni parametri	156
9.2	Razširjen zapis baznih funkcij	159
9.3	Kriterij kvalitete aproksimacijskega modela	159
10.	Linearni in polinomski modeli	161
10.1	Linearni model	161
10.2	Polinomski model	162
10.3	Povzetek	165

11. Umetne nevronske mreže	167
11.1 Konstrukcija nevronske mreže	168
11.1.1 Projekcijska konstrukcija	169
11.1.2 Radialna konstrukcija	169
11.2 Večnivojski perceptron (MLP)	171
11.2.1 Nevron večnivojskega perceptronja	171
11.2.2 Struktura mreže	173
11.2.3 Učenje z algoritmom vzvratnega širjenja napake	175
11.2.4 Učenje perceptronja	176
11.2.5 Simulacijski primer	177
11.2.6 Lastnosti večnivojskega perceptronja	178
11.2.7 Mreže z več skritimi nivoji	181
11.3 Povzetek	182
12. Mehki modeli	185
12.1 Mehka logika	185
12.1.1 Pripadnostne funkcije	186
12.1.2 Logični operatorji	187
12.1.3 Stopnja izpoljenosti pravila	188
12.1.4 Akumulacija in ostrenje	189
12.2 Tipi mehkih sistemov	189
12.2.1 Lingvistični mehki sistemi	189
12.2.2 Posebni mehki sistemi	190
12.2.3 Mehki sistem Takagi-Sugeno	192
12.3 Nevro-mehke mreže	193
12.3.1 Mehke bazne funkcije	193
12.3.2 Optimizacija mehkega modela	194
12.3.3 Interpretacija mehkih modelov	195
12.3.4 Simulacijski primer	197
12.4 Povzetek	198
13. Lokalno linearne nevro-mehki modeli	201
13.1 Osnovna ideja	201
13.1.1 Prikaz lokalnih linearnih nevro-mehkih modelov	202
13.1.2 Interpretacija afinega člena pri lokalno linearnih modelih	204
13.1.3 Interpretacija modela v obliki Takagi-Sugeno modela	206
13.2 Optimizacija parametrov lokalno linearnih modelov	207
13.2.1 Globalno ocenevanje parametrov modela	208
13.2.2 Lokalno ocenevanje parametrov modela	209
13.2.3 Primerjava med globalnim in lokalnim ocenevanjem parametrov modela ..	210
13.3 Strukturna optimizacija premise pravil	213
13.3.1 Algoritem lokalnega linearnega modela drevesa (LOLIMOT)	214
13.4 Povzetek	216

14. Identifikacija linearnih dinamičnih sistemov	219
14.1 Pregled metod identifikacije linearnih sistemov	219
14.2 Signali vzbujanja	220
14.3 Splošne strukture modelov	221
14.3.1 Terminologija in klasifikacija	224
14.3.2 Optimalni prediktor	230
14.3.3 Modeli predikcijskega pogreška PEM	232
14.4 Modeli časovnih vrst	233
14.4.1 Model AR	233
14.5 Modeli z izhodno povratno zanko	234
14.5.1 Model ARX	234
14.5.2 Model ARMAX	238
14.5.3 Model OE	240
14.5.4 Simulacijski primer	242
14.6 Modeli brez povratne zanke	245
14.6.1 Model FIR	245
14.7 Nekaj pomembnih vidikov identifikacije	247
14.7.1 Konsistentnost	247
14.7.2 Relacija med modelom šuma in filtriranjem	247
14.7.3 Delovna točka	248
14.8 Povzetek	249
15. Identifikacija nelinearnih dinamičnih sistemov	251
15.1 Razširitev linearne identifikacije na nelinearno identifikacijo	251
15.2 Model z zunanjim dinamikom	252
15.2.1 Metoda modeliranja z zunanjim dinamikom	252
15.2.2 Serijsko-paralelni in paralelni modeli	257
15.2.3 Nelinarni razredi dinamičnih vhodno-izhodnih modelov	259
15.2.4 Omejitve nelinearnih vhodno-izhodnih modelov	261
15.3 Vzbujanje sistemov	262
15.4 Ocenjevanje reda sistema	265
15.4.1 Struktturna identifikacija na osnovi metode Lipschitzevih koeficientov	265
15.4.2 Struktturna identifikacija z uporabo metode PCA	266
15.5 Povzetek	269

Del V. Odločanje na osnovi modela

16. Adaptivno vodenje	273
16.1 Prinzipi adaptivnega vodenja	274
16.2 Razdelitev adaptivnih sistemov glede na vodljivostno shemo	274
16.2.1 Odprtozančni adaptivni sistemi	275
16.2.2 Zaprtozančni adaptivni sistemi	275
16.3 Razdelitev adaptivnih sistemov glede na njihove funkcionalne podsklope	280
16.3.1 Direktno adaptivno vodenje	281
16.3.2 Indirektno adaptivno vodenje	281
16.4 Mehko adaptivno vodenje	281
16.4.1 Direktno mehko adaptivno vodenje	282

16.4.2 Indirektno mehko adaptivno vodenje	286
17. Prediktivno vodenje	289
17.1 Osnovni principi prediktivnega vodenja	290
17.2 Predikcijski modeli	291
17.2.1 Model impulznega odziva	292
17.2.2 Model v obliki odziva na stopnico	293
17.2.3 Model v obliki prenosne funkcije	293
17.2.4 Model z zapisom v prostoru stanj	294
17.2.5 Ostali modeli	295
17.3 Napoved izhoda na osnovi modela v obliki prenosne funkcije	295
17.3.1 Primer izračuna napovedi izhodnega signala	297
17.4 Modelno-referenčna trajektorija	298
17.5 Strukturiranje regulacijskega zakona	300
17.5.1 Regulirni horizont	301
17.5.2 Princip grupiranja	302
17.5.3 Razvoj po baznih funkcijah	302
17.6 Algoritmiziran izračun regulirnega signala	302
17.6.1 Kriterijska funkcija	302
17.7 Nastavitevna pravila za parametre prediktivnega vodenja	304
17.7.1 Nastavitevna pravila za nadkritično dušene procese:	304
17.7.2 Nastavitevna pravila za slabo dušene procese:	304
17.8 Prediktivno vodenje z dinamično matriko (DMC)	305
17.9 Posplošen prediktivni regulator (GPC)	307
17.9.1 Izpeljava regulacijskega zakona	309
17.9.2 Primer razvoja osnovnega prediktivnega regulatorja	310
17.10 Splošni časovo-zvezni predikcijski regulator (CGPC)	312
17.10.1 Osnovni CGPC algoritmom	312
17.10.2 Vpliv parametrov regulatorja CGPC	321
17.11 Prediktivno funkcijski regulator (PFC)	322
17.12 Regulator PFC z modelom prvega reda za sisteme z mrtvim časom	326
17.13 Regulator PFC v prostoru stanj	327
17.14 Regulator PFC v prostoru stanj za sisteme z mrtvim časom	328
17.15 Regulator PFC z neparametričnim modelom	329
17.16 Povzetek	331
A. Vektorji in matrike	333
A.1 Odvodi vektorjev in matrik	333
A.2 Gradient, Hessova in Jacobijeva matrika	334
B. Mehka logika	337
B.1 Mehke množice	337
B.1.1 Definicija mehke množice	337
B.1.2 Lastnosti mehkih množic	338
B.1.3 Mehki logični operatorji	342
B.2 Mehko sklepanje	347
B.2.1 Mehka izjava in predikat	347
B.2.2 Mehka pravila	348

B.2.3 Mehka implikacija	349
Reference	351

Del I

Uvod

1. Uvod v inteligentne sisteme za podporo odločanju

V uvodu bomo osvetlili vlogo inteligentnih sistemov pri odločanju. Pri odločanju v najširšem pomenu besede gre za delovanje, kjer na osnovi določenih izkušenj in znanj sprejemamo odločitve preko katerih vplivamo na delovanje obravnavanega sistema v splošnem. V veliki meri je odločanje povezano s poznavanjem modela procesa, zato bomo najprej obravnavali problem identifikacije nelinearnih sistemov. Osnovni problemi, ki jih je moče reševati na osnovi modelov so predstavljeni v podpoglavlju 1.1. Podpoglavlje 1.2 predstavlja in analizira različne procedure, ki jih moramo izvesti v primeru identifikacije nelinearnega sistema. Predstavljen je pregled različnih paradigm modeliranja v podpoglavlju 1.3. V podpoglavlju 1.4 so izpostavljeni določeni terminološkimi pojmi, ki jih pogosteje srečujemo v knjigi.

1.1 Pomen nelinearne identifikacije sistemov

Modele uporabljamo pri analizi, za napovedovanje in simulacijo obnašanja v prihodnosti. V inženirstvu jih uporabljamo za načrtovanje novih procesov in analizo obstoječih in za načrtovanje sistemov vodenja, optimizacijo, nadzor, odkrivanje in diagnosticiranje napak.

1.1.1 Linearni ali nelinearni problem

Predno se lotimo razvoja nelinearnega modela, moramo najprej ugotoviti, če je za to res potreba. Zato najprej zgradimo linearni model. Če ta ne zadostuje, potem je razlog lahko v nelinearnosti. Če je temu res tako, lahko preverimo z različnimi testi in primerjavo med procesom in linearnim modelom. To lahko storimo z uporabo odzivov na stopnico v različnih delovnih točkah (nelinearna odvisnost od delovne točke je zelo pogosta) ali pa z uporabo *nelinearnih testov*. Pregled nelinearnih testov je prikazan v [37], kjer je so testi v časovnem prostoru in korelacijski testi.

Pri prehodu na nelinearni model se pogosto zgodi, da je nelinearni model v prvi aproksimaciji slabši od linearnega. Zato je pametno, da uporabimo tako strukturo nelinearnega modela, ki vsebuje linearni model kot poseben primer, celoten model pa je vsota paralelnih linearnih modelov, ki delujejo v določenih delovnih točkah.

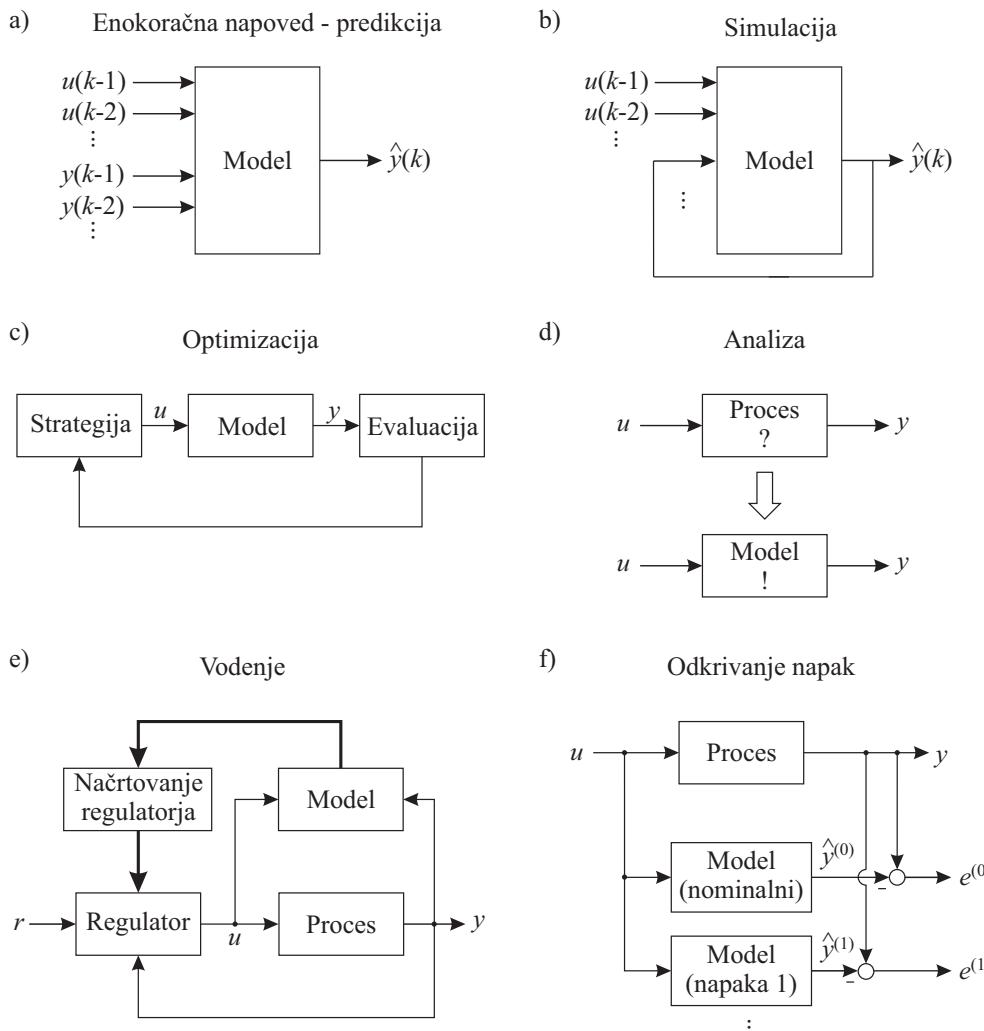
1.1.2 Predikcija

Slika 1.1a prikazuje enokoračno predikcijo (*ang. one-step ahead prediction*). Za enokoračno napoved izhoda procesa $\hat{y}(k)$ potrebujemo zakasnjene vhode $u(k-i)$ in zakasnjene izhode $y(k-i)$ procesa. Pri tem predpostavimo, da $u(k)$ ne vpliva direktno na $y(k)$. Za predikcijo izhoda za določeno število korakov v prihodnosti je potrebno meriti izhod procesa (ali stanja procesa).

Če želimo napovedovati določen izhod procesa za več korakov v prihodnosti to imenujemo večkoračna predikcija (*ang. multi-step ahead ali h-step ahead prediction*). Število korakov h v prihodnosti imenujemo horizont predikcije (*ang. prediction horizon*). Pri tem lahko uporabimo dva alternativna načina predikcije:

- model lahko zgradimo tako, da direktno napoveduje h korakov v prihodnost. V tak model moramo dodatno vključiti vhodne signale v prihodnosti $u(k), u(k+1), \dots, u(k+h-1)$. Pomanjkljivost je v tem, da dimenzija problema raste s horizontom predikcije h in da lahko napovemo samo vrednost pri horizontu h , čeprav pogosto potrebujemo tudi predikcijo za $1, 2, \dots, h$ korakov.
- Model na sliki 1.1a lahko uporabimo za enokoračno predikcijo. V naslednjem koraku pa isti model uporabimo za napoved na ta način, da nadomestimo $k \rightarrow k+1$ in pri tem uporabimo napoved v prejšnjem koraku $\hat{y}(k)$. To ponovimo h krat, da dosežemo h -ti horizont. Tak način akumulira napako in vsebuje povratno zanko, ki je lahko nestabilna.

Zadnji način je enak simulaciji, če predpostavimo neskončen korak predikcije ($h \rightarrow \infty$).



Slika 1.1. Uporaba modelov za a) predikcijo, b) simulacijo, c) optimizacijo, d) analizo, e) vodenje in f) detekcijo napak.

1.1.3 Simulacija

Slika 1.1b kaže primer simulacijske uporabe modela, kjer potrebujemo za izračun napovedi izhodnega signala v prihodnosti samo vhodni signal procesa. Takšna uporaba modela je zelo pogo-

sta, posebno v primeru optimizacije, vodenja in odkrivanja napak. Posebno pomembno področje uporabe je področje mehkih senzorjev (programskega senzora), kjer z modelom nadomestimo fizični senzor. To lahko storimo zaradi okoljske pogojenosti, če so fizični senzorji preveliki, ne zadostni robustni, ali pa predragi. Ali pa z mehkim senzorjem ocenjujemo spremenljivke, ki niso merljive v realnem času, jih pa lahko ocenimo s pomočjo modela in merljivih vhodov v tak model, ki ga zgradimo na osnovi laboratorijskih meritev.

Bistvena razlika med napovedjo (predikcijo) in simulacijo je v tem, da je simulacija povratno-zančna struktura, kar dodatno oteži modeliranje in identifikacijo in zahteva dodatno pazljivost v smislu stabilnosti modela.

1.1.4 Optimizacija

Slika 1.1c prikazuje uporabo predikcijskega ali simulacijskega modela v primeru optimizacije določene kriterijske funkcije, ki omogoča iskanje optimalne delovne točke ali optimalnega poteka vhoda procesa.

1.1.5 Analiza

Slika 1.1d prikazuje uporabo modela pri spoznavanju in razumevanju delovanja procesa, pri različnih parametrih in različnih vhodnih procesa.

1.1.6 Vodenje

Slika 1.1e nakazuje uporabo modela za namene načrtovanja vodenja. Moderni sistemi vodenja temeljijo na modelu procesa, ki omogoča implementacijo celovite informacije o procesu in celo njegove nedoločenosti (*ang. uncertainty*).

Čeprav je pričakovati, da je vodenje na osnovi modela tesno povezano s kvaliteto modela, velja da ne gre za monotono relacijo [43, 114]. Področje *identifikacije za namene vodenja* raziskuje kako najenostavnije identificirati model, ki bo zadostil uporabi za namene načrtovanja vodenja.

V primeru linearnih modelov navadno poizkušamo najti model, ki je najbolj natančen v srednjem frekvenčnem področju okoli mejne frekvence zaprtozančnega sistema, ker je to ključnega pomena za stabilnost zaprte zanke [52, 114]. Za sledilno delovanje sistema je potrebna natančnost modela v področju frekvenc referenčnega signala, ki je običajno v nizkem področju. V primeru regulacijskega načina delovanja pa rešujemo problem z uvedbo integracijskega zakona, ki odpravlja napake med modelom in procesom in odpravlja tudi motnje na vhodu in izhodu procesa.

1.1.7 Odkrivanje napak

Slika 1.1f prikazuje eno od možnih strategij uporabe modela pri odkrivanju napak. Prikazana strategija najlaže prikaže prednosti uporabe modelov, ki jih dobimo z identifikacijo. V tem primeru zgradimo model, ki opisuje delovanje procesa v normalnih okoliščinah (nominalni model), in modele napak. Ko se napaka pojavi, za vsako zgradimo svoj model (napaka 1, 2, ...). S primerjavo izhodov modela za nominalni primer ($\hat{y}^{(0)}$), izhodov za modele napak ($\hat{y}^{(i)}$ za napako i) in izhodom modela y , lahko odkrijemo napako i . Primerjamo statistične vrednosti vektorjev napak med izhodom in izhodom modela napake $e^{(i)}$ za $i > 0$ in nominalnega modela $e^{(0)}$. Primerjave so običajno izvedene na filtriranih signalih napake, da izločimo vpliv napak zaradi modeliranja in šuma in se izognemo napačnim alarmom.

Največkrat zahtevamo, da so modeli za odkrivanje napak zelo natančni in nižjem frekvenčnem območju, ki največkrat definira obnašanje procesa. Področje visokih frekvenc, oziroma prehodnih

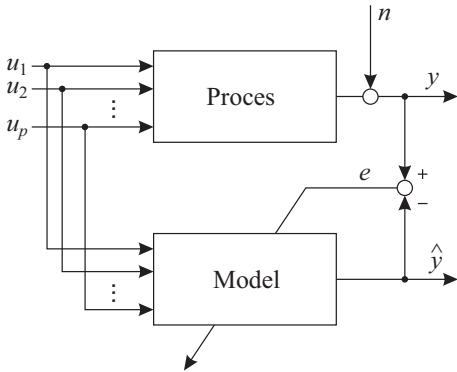
pojavov ni tako pomembno, saj trajajo kratek čas in znotraj teh pojavov zelo težko detektiramo napake. Seveda se s povečevanjem natančnosti v širšem frekvenčnem področju poveča tudi hitrost detekcije napak.

Značilnost delovanja sistemov za odkrivanje napak je ta, da je cena neodkrivite napake (nealarmiranje ob napaki) nekaj velikostnih redov večja kot cena napačnega alarma (alarm, ko ni napake). To pomeni, da moramo načrtati sistem, ki bo maksimalno občutljiv in pri katrem bomo tolerirali tudi pogoste napačne alarne. Hitro odkrivanje napak, majhno število napačnih alarmov in odkrivanje večkratnih napak je kvaliteta dobrih sistemov za odkrivanje napak.

1.2 Koraki pri nelinearni identifikaciji

Modeliranje in identifikacija nelienarnih procesov je zahteven postopek, ker so nelinearni procesi unikatni, t.j. medsebojno nimajo veliko enakih lastnosti. Ena najpomembnejših lastnosti metod identifikacije pa je ravno univerzalnost, kar pomeni, da lahko z določeno metodo opišemo širok razred fizikalno in strukturno zelo različnih procesov.

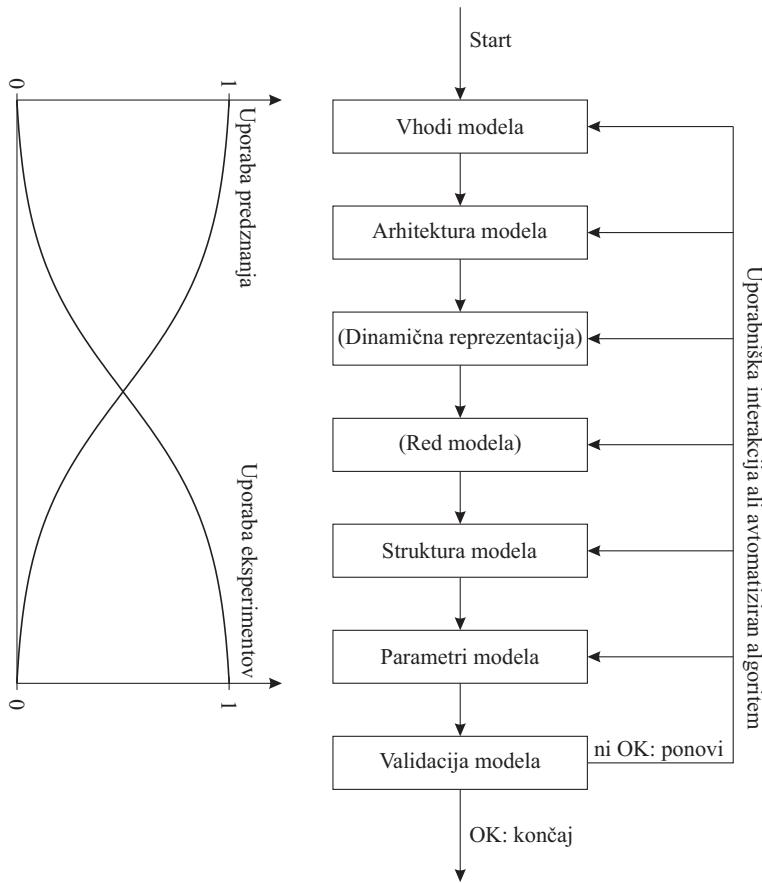
Slika 1.2 prikazuje korake pri identifikaciji sistemov. Model mora opisati delovanje procesa v celotnem delovnem področju z določeno natančnostjo. Kvaliteta modela je običajno definirana z napako med izhodom procesa in modela. Glede na to napako navadno prilagajamo parameter modela.



Slika 1.2. Identifikacija sistema: Adaptacija parametrov modela, da ta kar najbolje opisuje obnašanje procesa. Proses in model z enakimi vhodi $\underline{u} = [u_1 \ u_2 \ \dots \ u_p]^T$, in izhodoma y in \hat{y} primerjamo na osnovi napake e , ki je uporabljena za adaptacijo modela.

V tem poglavju bodo predstavljeni glavni koraki za uspešno identifikacijo modela procesa. Slika 1.3 prikazuje te korake (v oklepaju so koraki, ki so potrebni v primeru dinamičnih sistemov):

1. izbira vhodov modela
2. izbira oblike signalov vzbujanja
3. izbira arhitekture modela
4. izbira vrste dinamike modela
5. izbira reda modela
6. izbira strukture in kompleksnosti modela
7. ocena parametrov modela
8. ocenjevanje kvalitete modela
9. vrednotenje vpliva parametrov metode



Slika 1.3. Postopek identifikacije in z njim povezana povratnozančna struktura (oklepaji definirajo korake, ki so potrebni samo v primeru dinamičnih sistemov).

1.2.1 Izbira vhodov modela

Za izbiro vhodov navadno izbiramo med štirimi strategijami:

- *Uporaba vseh vhodov*: vodi do modelov višjih dimenzij z višjim številom parametrov. Lahko jo uporabimo skupaj z regularizacijo, ki z dodanjem kriterijev v kriterijsko funkcijo preprečuje preparametriranje in slabo pogojenost problema (singularnosti matrik).
- *Preizkus vseh vhodnih kombinacij*: Čeprav vodi ta strategija do najboljše rešitve, v praksi ni uporabna, ker število preizkusov narašča kombinatorno s številom potencialnih vhodov.
- *Nenadzorovana izbira vhodov*: Zelo uporabna izbira vhodov v tem primeru temelji na analizi glavnih komponent (*ang. Principal Component Analysis - PCA*), ki omogoča odpravo nepOMEMBNIH vhodov na relativno enostaven način. Glavna pomankljivost metode PCA je v tem, da je izbira izvedena na podatkih vhodnega prostora, brez izhodnih podatkov. To pa lahko pomeni, da v posameznih primerih ne upoštevamo vhodov, ki so lahko zelo pomembni za kvaliteto modela.
- *Nadzorovan izbor vhodov*: Vhodi so izbrani tako, da dobimo največno možno natančnost modela. Za linearne modele uporabljam korelacijsko analizo, v primeru nelinearnih modelov pa je to kompleksen optimizacijski problem, ki ga lahko rešimo z npr. evolucijskimi algoritmi (EA).

1.2.2 Izbira oblike signalov vzbujanja

Korak 2 zahteva predhodno poznavanje procesa in namena uporabe modela. V primeru modelov, ki jih dobimo na osnovi identifikacije (*ang. black-box modelling*) so meritve najpomembnejši in edini izvor informacije. Obnašanja procesa, ki ni zajeto v izmerjenih podatkih, ne moremo identificirati, ker ga nismo zaznali. To seveda omejuje kvaliteto modela.

1.2.3 Izbira arhitekture modela

Korak 3 je verjetno najbolj občutljiv in subjektiven del procedure. Arhitektura modela se izbira predvsem na osnovi naslednjih kriterijev:

- *Problemska domena*: Klasifikacija (razvrščanje), aproksimacija (regresija, modeliranje) sistema.
- *Uporaba*: Model lahko uprabimo za simulacijske namene, optimizacijo, vodenje, detekcijo napak, itd.
- *Problem dimenzije*: Število relevantnih vhodov (in izhodov) omejuje izbiro arhitekture.
- *Dostopnost in kvaliteta meritev*: Pri pomanjkljivih in zelo osumljenih podatki je težko zgraditi dober model.
- *Omejitve glede razvoja modela*: Časovne, materialne, itd.
- *Omejitve spominskih kapacitet*: Na določenih področjih so še vedno omejitve spominskih kapacitet (avtomobilska industrija), vendar te omejitve hitro izginjajo.
- *Nesprotno* (*ang. off-line*) in *sprotno* (*ang. on-line*) učenje: Vse arhitekture so primerne za nesprotno učenje, za sprotno pa to ni res. Področje sprotnega učenja je še vedno stvar raziskovanja.
- *Izkušnje uporabnika*: Izkušnje uporabnika v veliki meri vplivajo na izbiro metod in arhitektur modelov.
- *Dostopnost orodij*: Dostopnost orodij v veliki meri vpliva na izbor določenih arhitektur modelov.
- *Sprejemanje s strani uporabnikov*: Glavno pri sprejemanju novih metod v modeliranju in identifikaciji je razumevanje osnovnih metod in interpretabilnost modelov.

1.2.4 Izbira vrste dinamike modela

Korak 4 je najbolj odvisen od uporabe modela. Če ga uporabljam za enokoračno predikcijo, potem lahko uprabimo zapis v obliki NARX ali NARMAX strukture; glej poglavje 14. Na izbiro vpliva tudi apriori znanje o procesu. Osnovne značilnosti različnih dinamičnih modelov za predstavljenje in analizirane v poglavju 15.

1.2.5 Izbira reda modela

Korak 5 se običajno izvaja na ta način, da združimo znanje o procesu in poizkušanje. Izbira višjega reda rezultira v višji kompleksnosti modela. Pri nelinearni identifikaciji je uporabnik največkrat prisiljen sprejeti to, da zanemari določeno manj vplivno dinamiko, saj se določena napaka vnese že z aproksimacijo, npr. statične nelinearnosti. Po izkušnjah je običajno izbira modelov nižjega reda zadostna, saj je napaka modela največkrat najbolj odvisna od nelinearnosti procesa.

1.2.6 Izbira strukture in kompleksnosti modela

Korak za 6 je mnogo zahtevnejši kot optimizacija parametrov. Lahko ga izvedemo avtomatično, če uporabimo metode kot so ortogonalna metoda najmanjših kvadratov (*ang. Orthogonal Least Squares - OLS*). Za določevanje kompleksnosti modela lahko uporabimo metodo regularizacije.

1.2.7 Ocena parametrov modela

Korak 7 je najenostavnejši za avtomatizacijo. Navadno ga rešimo z uporabo linearne ali nelinearne optimizacijske tehnike. Linearne tehnike navadno delujejo popolnoma avtomatsko, nelinearne pa zahtevajo določeno interakcijo z uporabnikom. Glede na izbrano tehniko optimizacije moramo v tem primeru izbrati začetno oceno parametrov in določene nastavitevne parametre metode.

1.2.8 Ocena kvalitete modela

Korak 8 služi kot ocena, če smo vse predhodne korake izvedli ustrezno in dobili zadovoljiv model. Kriterijske funkcije, ki so primerne za to oceno, so močno odvisne od problema. V mnogih primerih model ni končni cilj, temveč ga uporabimo v vodenju ali pa pri detekciji napak. Za oceno modela je v takem primeru lahko uporabljenha ocena delovanja celotnega zaprtozančnega sistema, ali pa občutljivost pri detekciji motenj.

Prvi korak pri vrednotenju modela je preizkus modela na podatkih, ki smo jih uporabili za učenje modela. Ta način imenujemo tudi **verifikacija** ali **preverjanje**. Če verifikacija ne da ustreznih rezultatov, potem prekinemo z nadaljnimi koraki. Razlog je lahko v premajhnem številu učnih podatkov ali pa v neustreznji strukturi modela.

Če je verifikacija modela ustrezna, potem model preizkusimo še na množici podatkov, ki pri učenju niso bili uporabljeni. Posebno pomembno je vzbujanje v okolici delovnih točk, kjer bo proces deloval. Če model uporabljamo za načrtovanje zaprtozančnega vodenja, potem moramo model dobro vzbujati okoli področja pričakovane zaprtozančne pasovne širine. Pri modeliranju za namene odkrivanja napak, pa je najpomembnejša dobro modeliranje v širokem delovnem področju. Ta korak imenujemo tudi **validacija** ali **potrjevanje**.

1.2.9 Vrednotenje vpliva parametrov metode

Parametri metode so parametri, ki jih ne izračuna metoda avtomatično, ampak jih nastavi uporabnik. Tipični taki parametri so začetne vrednosti parametrov modela, ki jih identificiramo, konstante učenja, število rojev, dimenzijske regresorjev, število nevronov in število plasti v nevronskih mrežah, število pravil v mehkih sistemih, parametri, ki zaključijo algoritem. Te parametre običajno nastavimo ročno med izvajanjem simulacijskih preizkusov. Večino teh parametrov lahko določimo tudi avtomatsko, vendar se potem običajno srečamo s problemom večkriterijske optimizacije.

Dobre lastnosti metode glede parametrov so naslednji:

- *Razdružen vpliv parametrov metode.* Pri nastavljanju je problem, če vpliva posameznega parametra ne moremo opazovati posamezno, ampak se vpliv posameznega parametra spremeni pri spremembi drugega parametra.
- *Majhno število parametrov metode.*
- *Parametri metode morajo imeti jasen vpliv na delovanje algoritma.*
- *Enostavna izbira prednastavljenih vrednosti parametrov metode.*
- *Sprejemljiva in enostavna občutljivostna karakteristika parametrov metode.*

1.3 Teoretični, eksperimentalni in kombinirani modeli

Metode modeliranja lahko razdelimo na tri različne pristope; glej slika 1.4:

- *Teoretično modeliranje* temelji na poplnem razvoju modela na osnovi ravnotežnih enačb, t.j. fizikalnih enačb, kemijskih, bioloških, ekonometričnih, itd. Takšni modeli ne temeljijo na izmerjenih podatkih, ki jih izmerimo na procesu. Parametri modela imajo fizikalni pomen.
- *Eksperimentalno modeliranje* temelji zgolj na izmerjenih podatkih. Tako struktura kot tudi parametri modela so ocenjeni iz izmerjenih podatkov. Običajno ne poskušamo vnašati znanja o procesu, oziroma zelo malo. Parametri modela nimajo direktne fizikalne interpretacije.
- *Kombinirano modeliranje* predstavlja kompromis med teoretičnim in eksperimentalnim modeliranjem. Navadno je struktura modela tisto, kar določimo na osnovi znanja o procesu, parametre pa ocenimo na osnovi izmerjenih podatkov.

	Teoretični model	Semi-teoretični model	Identifikacijski model
Potrebitno znanje	fizikalni zakoni poznavanje procesa	kvalitativno znanje (pravila) znanje in podatki	eksperimentiranje podatki
Prednosti	dobra ekstrapolacija dobro razumevanje delovanja velika zanesljivost reskaliranje		kratki čas razvoja modela malо znanja o procesu lahko jo uporabimo kot dopolnilo tudi za nerazumljive procese
Slabosti	časovno potratno potrebitno procesno znanje natančnost odvisna od znanja samo za dobro poznane procese		nezanesljiva ekstrapolacija ne moremo ga reskalirati natančnost podatkovno odvisna pridobim malо znanja o procesu
Področje uporabe	optimizacija, konstruiranje za enostavne procese		samo za obstoječe procese za zahtevne procese

Slika 1.4. Teoretični, eksperimentalni in kombinirani modeli

Teoretični, eksperimentalni in kombinirani modeli. Pri kombiniranih modelih lahko srečamo zelo različne kombinacije med teoretičnim in eksperimentalnim načinom modeliranja.

1.4 Terminologija

Terminologija, ki jo uporabljamо je klasična za področje identifikacije sistemov in optimizacije:

- *Proces*: Uporabljamо kot sinonim za industrijski proces ali sistem, ki ga raziskujemo. Lahko gre za statični ali dinamični proces.
- *Linearni parametri*: Parametri, ki nastopajo linearno.
- *Nelinearni parametri*: Parametri, ki nastopajo nelinearno.
- *Učenje*: Optimizacija strukture ali parametrov modela, da dosežemo optimalno vrednost določene kriterijske funkcije. Če je poudarek na identifikaciji ali optimizaciji strukture, potem navadno govorimo o *učenju*, če pa je poudarek na iskanju parametrov pa govorimo o *oceni* ali *estimaciji*.

- *Generalizacija*: Evaluacija izhoda modela za vhodni vektor, ki ni vsebovan v učni množici. Generalizacija je lahko *interpolacija*, če je vhodni vektor znotraj področja, ki ga pokrivajo vhodni vektorji iz učne množice, ali pa *ekstrapolacija*, če vhodni vektor leži izven tega področja.

2. Uvod v statistične metode

V tem poglavju bomo podali kratek uvod v statistiko, deterministične in naključne spremenljivke, elementarne metode in nekaj statističnih lastnosti.

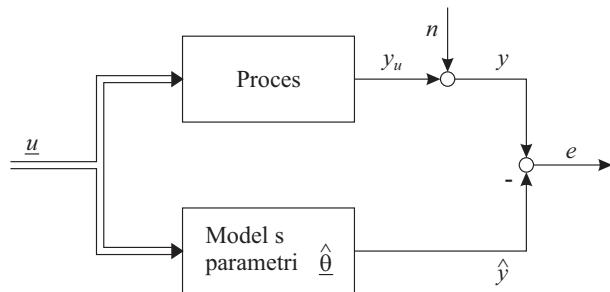
2.1 Deterministične in naključne spremenljivke

Če izvedemo določen preizkus na procesu in dobimo vedno enak izhod, potem rečemo, da je izhod determinističen in da ga na ta način lahko tudi zapišemo. Na ta način lahko pri izbranem vhodu napovemo izhod. Če točnega rezultata na moremo napovedati, potem govorimo o nedeterminističnem sistemu ali spremenljivki. Pogosto rečemo, da je razlog za neponovljivost eksperimenta v šumu procesa, ki ga seveda ne znamo opisati in zato ne moremo predvideti njegove vrednosti.

Naključne spremenljivke so zelo pomembne za opis narave procesov, čeprav je klasični fizikalni zapis determinističen. Veliko determinističnih pojavov ne znamo modelirati in ne razumemo fizikalnega ozadja in zato se zdi kot da so naključne narave. Določeni pojavi so posledica vpliva spremenljivk, ki jih ne merimo ali pa ne znamo meriti. Vse to je združeno v motnjo, ki ji rečemo šum.

Glede na naravo šuma je smiselno za njegov opis uporabiti naključne spremenljivke in stohastične procese, ki jih opisujemo s pričakovanimi vrednostmi, variancami itd. Za razlago koncepta determinističnih in naključnih spremenljivk smo izbrali primer na sliki 2.1. Nemerljiv izhod procesa y_u je moten z aditivnim šumom n , kar rezultira v merljivem izhodnem signalu y .

Potrebno je poudariti, da je tudi izhod modela \hat{y} naključne narave, saj je posledica vhoda in ocjenjenih parametrov modela in ti so naključnih vrednosti. Zaradi tega je potrebno uporabiti metode statistične analize pri obravnavi ocjenjenih parametrov. Pričakovane vrednosti ocjenjenih parametrov in kovariančne matrike dajo informacijo o kvaliteti ocene.



Slika 2.1. Deterministične in naključne spremenljivke pri modeliranju procesov.

2.2 Porazdelitev gostote verjetnosti

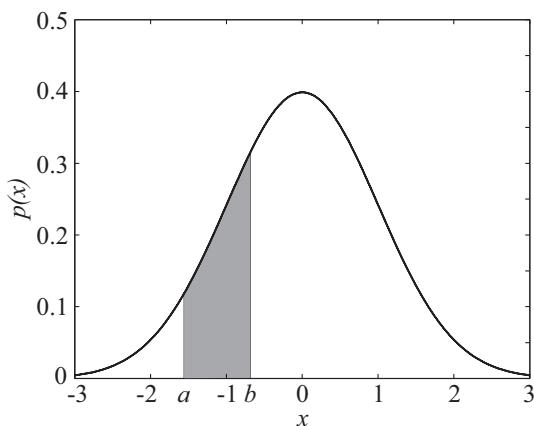
Naključne spremenljivke se podajajo z verjetnostjo. Funkcija porazdelitve gostote verjetnosti (*ang. probability density distribution*) je najbolj celovit način predstavitev informacije o naključnih spremenljivkah. Na sliki 2.2 je *Gaussova* porazdelitev gostote verjetnosti $p(x)$ za naključno spremenljivko x . Ta porazdelitev je *normalna* porazdelitev. Interpretacija je naslednja. Verjetnost, da x leži na intervalu med $[a, b]$ je enaka sivo obarvani ploskvi pod funkcijo porazdelitve gostote verjetnosti. Ker je x zvezna spremenljivka, je verjetnost za točno določeno vrednost x enaka 0, na intervalu med $[-\infty, \infty]$ pa je enaka 1.

Pdf določa verjetnost za realizacijo določene naključne spremenljivke, če izvedemo eksperiment. Slika 2.3 kaže realizacijo za Gaussovo porazdelitev gostote verjetnosti in 100 eksperimentov. Vsak eksperiment ima za posledico določeno realizacijo. Vrednosti okoli nič so najpogosteje glede na porazdelitev gostote verjetnosti. Eksperimenti niso časovno odvisni in jih lahko realiziramo sočasno.

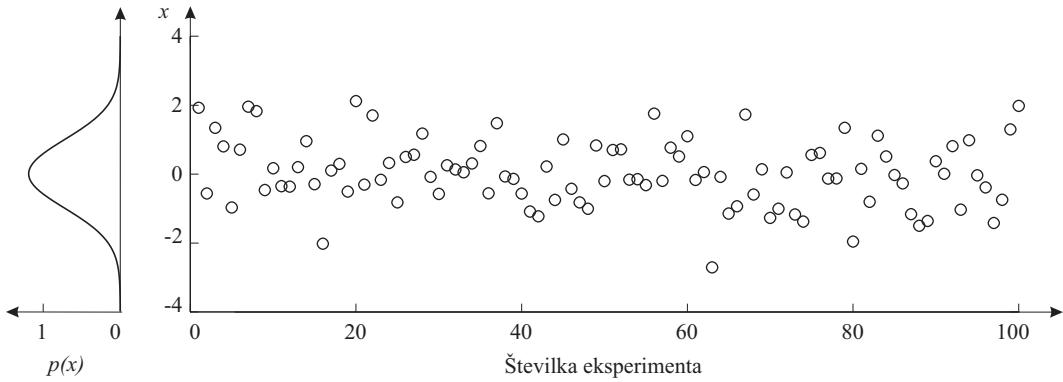
V primeru več naključnih spremenljivk dobimo vektor $\underline{x} = [x_1 \ x_2 \ \dots \ x_n]^T$ in v tem primeru je porazdelitev gostote verjetnosti $p(\underline{x})$ večdimensionalna. Taka *združena funkcija porazdelitve gostote verjetnosti* je prikazana na sliki 2.4, kjer sta dve dvodimenzionalni Gaussovi porazdelitvi gostote verjetnosti in njuna konturna diagrama.

Pdf na levi je osno ortogonalna, to pomeni, da jo lahko konstruiramo kot produkt dveh enodimenzionalnih funkcij porazdelitve gostote verjetnosti $p_{x_1}(x_1)$ in $p_{x_2}(x_2)$. To ne velja za porazdelitev gostote verjetnosti na desni.

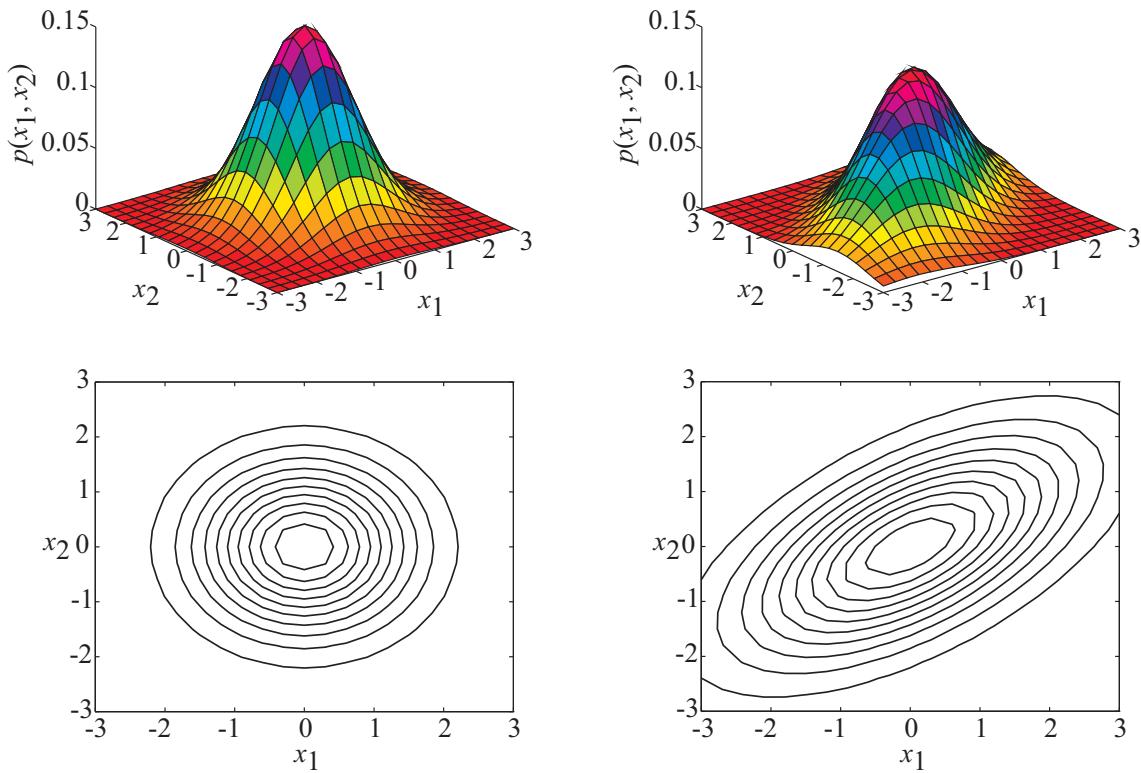
Zanimiva je bolj natančna primerjava porazdelitvenih funkcij na sliki 2.4. Poglejmo posledice različnih oblik porazdelitvenih funkcij za spremenljivki x_1 in x_2 . Najprej obravnavajmo levo porazdelitev gostote verjetnosti. Če je realizacija prve spremenljivke npr. $x_1 = c$ znana, kakšno informacijo s tem dobimo o naključni spremenljivki x_2 ? Z drugimi besedami nas zanima, kako se spremeni enodimenzionalna porazdelitev gostote verjetnosti, ki jo dobimo s prerezom skozi $p(\underline{x})$ pri $x_1 = c$? Ugotovimo lahko, da so vsi prerezi po obliku enaki in neodvisni od c . Torej lahko ugotovimo, da pri poznani realizaciji ene naključne spremenljivke, ne pridobimo informacije glede druge spremenljivke. Dve spremenljivki s to lastnostjo sta neodvisni.



Slika 2.2. Enodimenzionalna Gaussova porazdelitev gostote verjetnosti. Verjetnost, da x leži na intervalu med $[a, b]$ je enaka sivo obarvani ploskvi pod funkcijo porazdelitve gostote verjetnosti.



Slika 2.3. Naključna spremenljivka z Gaussovo distribucijo (pdf) ima različne realizacije za vsak eksperiment. Verjetnost za vsak eksperiment je določena s funkcijo porazdelitve gostote verjetnosti. Številka eksperimenta ni povezana s časom.



Slika 2.4. Dvodimenzionalna Gaussova porazdelitev gostote verjetnosti. Leva porazdelitev gostote verjetnosti opisuje dve neodvisni naključni spremenljivki. Informacija o realizaciji ene od njih ne prinaša nobene informacije o porazdelitvi druge naključne spremenljivke. Desna slika pa prikazuje dve odvisni spremenljivki. Informacija o eni spremenljivki prinaša informacijo tudi o drugi spremenljivki.

Dve naključni spremenljivki x_1 in x_2 sta *neodvisni*, če produkt njunih funkcij porazdelitve gostote verjetnosti vodi k skupni funkciji porazdelitvi gostote verjetnosti

$$p(x_1, x_2) = p_{x_1}(x_1) \cdot p_{x_2}(x_2). \quad (2.1)$$

To pomeni, da lahko neodvisne naključne spremenljivke lahko opišemo popolnoma samo z enodimenzionalnimi porazdelitvami gostote verjetnosti $p_{x_i}(x_i)$. Na sliki 2.4 desno vidimo dve odvisni spremenljivki. Če poznamo realizacijo za x_1 , dobimo tudi informacijo za x_2 . Za majhne vrednosti x_1 so majhne vrednosti x_2 bolj verjetne, kot velike.

2.3 Stohastični procesi in ergodičnost

Naključno spremenljivko opišemo s funkcijo gostote verjetnosti, ki vodi v realizacijo spremenljivke, ko izvedemo eksperiment. Dostikrat moramo modelirati naključno spremenljivko x , ki ni odvisna samo od eksperimenta temveč tudi od časa. Naključna spremenljivka $x(k)$, ki je odvisna od časa k se imenuje *stohastični proces*. Stohastični proces $x(k)$ je opisan s porazdelitvijo gostote verjetnosti $p(x, k)$, ki je tudi odvisen od časa k . Na ta način lahko stohastični proces vidimo tudi kot neskončno število naključnih spremenljivk, vsaka za en časovni trenutek k [93].

Slika 2.5 prikazuje več različnih realizacij stohastičnega procesa. Pomembno je poudariti, da je odvisna od določene realizacije in časa. Na sliki 2.5 je prikazan šum meritve. V vsaki meritvi izmerimo eno realizacijo (eno vrstico na sliki 2.5) stohastičnega procesa. Če se statistične lastnosti procesa ne spremenijo, dobimo v drugi meritvi različno drugo realizacijo stohastičnega procesa. To je stohastična lastnost naključne spremenljivke.

Zelo tipičen in pogost šum je Gaussov beli šum. To pomeni, da gre za amplitudno porazdelitev stohastičnega procesa v vsakem časovnem trenutku k_0 , ki jo opišemo s $p(x, k_0)$. Ta porazdelitev gostote verjetnosti določa s kakšno verjetnostjo se pojavljajo majhne in večje vrednosti $x(k_0)$. Nasprotno pa beli pomeni relacijo med dvema zaporednima časovnima vzorcema $p(x, k_1)$ in $p(x, k_2)$ stohastičnega procesa. Če se statistične lastnosti stohastičnega procesa s časom *ne* spreminja, se taki stohastični procesi imenujejo stacionarni.

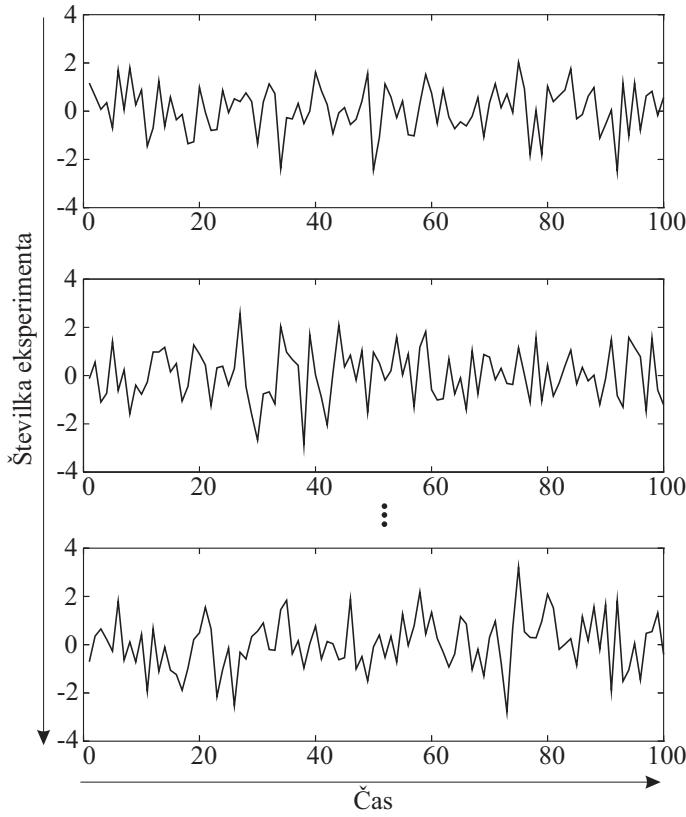
Stohastični proces je *stacionaren*, če se statistične lastnosti procesa ne spreminja s časom. To pomeni, da je

$$p(x, k_1) = p(x, k_2) \quad \text{za vse } k_1 \text{ in } k_2. \quad (2.2)$$

Šum običajno modeliramo kot stacionaren proces. Lahko pa seveda zgodi, da se šum s časom spreminja, npr. monotono narašča, kar poimenujemo lezenje signala (*ang. drift*). Običajno pa pri šumu predpostavimo, da so pričakovanja pri *vseh* različnih realizacijah, od zgoraj navzdol na sliki 2.5, enaka pričakovaju *ene same* časovne realizacije na sliki 2.5.

Če lahko za stohastični proces izračunamo pričakovano vrednost iz ene časovne realizacije, potem tak proces imenujemo *ergodičen*.

Stohastični proces je *ergodičen*, če lahko pričakovano vrednost preko različnih realizacij izračunamo kot časovno povprečje ene realizacije.



Slika 2.5. Stohastični proces je odvisen od eksperimentalne realizacije in časa, [39].

Ergodičnost implicira, da ena realizacija, če gre $k \rightarrow \infty$, vsebuje vso informacijo o statističnih lastnosti stohastičnega procesa. Za tak proces lahko izračunamo osnovne statistične lastnosti, srednjo vrednost in varianco.

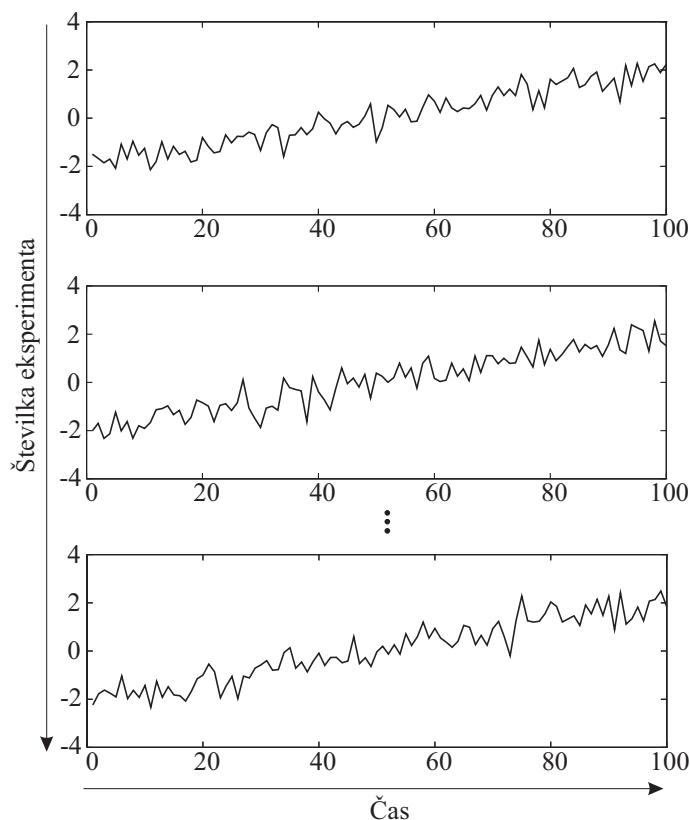
Ergodičen proces je stacionaren. Povprečnje po času izloči odvisnost od časa. Zaradi tega so statistične lastnosti ergodičnega procesa neodvisne od časa. Poglejmo si nekaj primerov nestacionarnih procesov, ki so hkrati tudi neergodični. Na sliki 2.6 je primer nestacionarnega in neergodičnega procesa, kjer se vrednost signala s časom spreminja. Na sliki 2.7 pa je primer procesa, kjer je $x(k)$ konstantna vrednost določena s porazdelitvijo $p(x, k)$. Ta stohastični proces je neergodičen, [93]. Z izbiro določene realizacije in izračunom povprečja dobimo vsakič drugo vrednost, ki pa ni enaka pričakovani vrednosti gled na izbrano porazdelitev $p(x, k)$.

2.4 Pričakovana vrednost

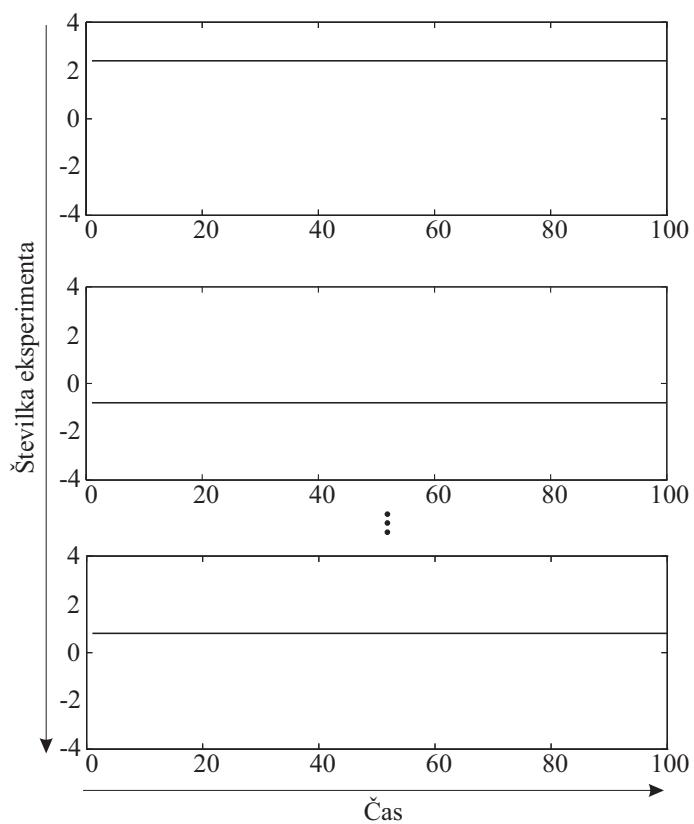
Dve najpomembnejši veličini, ki določata lastnosti naključnih spremenljivk in stohastičnih procesov sta njihova *pričakovana vrednost* ali *povprečje* in *varianca*.

Osnova za vse statistične izračune oziroma operacije je operator *pričakovane* vrednosti. Argument je običajno naključna spremenljivka ali stohastični proces, rezultat pa je pričakovana vrednost spremenljivke oziroma stohastičnega procesa. Če je argument operatorja determinističen, potem dobimo $E\{x\} = x$.

Če želimo izračunati pričakovano vrednost naključne spremenljivke, moramo vse realizacije utežiti z njihovo verjetnostjo in integrirati po celotni zalogi vrednosti. Definicija pričakovane vrednosti naključne spremenljivke x je tako



Slika 2.6. Nestacionarni in neergodični stohastični proces. Srednja vrednost stohastičnega procesa se spreminja s časom. Zaradi tega se statistične lastnosti s časom spreminjajo in je proces nestacionaren. Zaradi tega je povprečje glede na čas (od leve proti desni) popolnoma različno od povprečja po realizacijah (od zgoraj navzdol).



Slika 2.7. Stacionarni toda neergodični procesi. Ker statistične lastnosti procesa niso odvisne od časa, je proces stacionaren. Ni pa ta proces ergodičen, ker ena realizacija ne nosi informacije o statističnih lastnostih procesa.

$$\text{povprečje}\{x\} = \mu_x = E\{x\} = \int_{-\infty}^{\infty} x p(x) dx . \quad (2.3)$$

Posledično je pričakovana vrednost stohastičnega procesa enaka

$$\text{povprečje}\{x(k)\} = \mu_x(k) = E\{x(k)\} = \int_{-\infty}^{\infty} x(k) p(x, k) dx . \quad (2.4)$$

Izračun pričakovane vrednosti zahteva funkcijo porazdelitve gostote verjetnosti za naključno spremenljivko ali proces. Ta pa v praksi običajno ni znana. Zaradi tega predpostavimo, da je stohastični proces ergodičen in namesto povprečenja po vseh realizacijah, povprečimo vrednosti po eni realizaciji glede na čas. Če je $x(k)$ realizacija stohastičnega procesa za časovni trenutek $k = 1, \dots, N$, potem je pričakovana vrednost ocenjena kot

$$E\{x\} \approx \frac{1}{N} \sum_{k=1}^N x(k) = \bar{x} . \quad (2.5)$$

Včasih to oceno pričakovane vrednosti za x zapišemo kot \bar{x} , ki pomeni tudi, da je to eksperimentalno določena srednja vrednost. Pripomnimo, da v enačbi 2.5 $E\{x\}$ ni odvisen od časa k , ker po tej spremenljivki delamo povprečje. Zaradi tega je nujno, da mora biti stohastični proces stacionaren, če želimo uporabiti enačbo 2.5. Če je stohastični proces ergodičen, potem ocena v enačbi 2.5 konvergira proti pravi pričakovani vrednosti, če gre $N \rightarrow \infty$. V nadaljevanju povsod predpostavljam ergodičnost procesov, čeprav to ni posebej izpostavljeno. Enačba 2.5 predstavlja nepristransko oceno pričakovane vrednosti x , če je obravnavani proces ergodičen [93].

Definicijo operatorja pričakovane vrednosti lahko razširimo na vektorje in matrike kot sledi

$$E\{\underline{x}\} = E \left\{ \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \right\} = \begin{bmatrix} E\{x_1\} \\ E\{x_2\} \\ \vdots \\ E\{x_n\} \end{bmatrix} , \quad (2.6)$$

$$\begin{aligned} E\{\underline{X}\} &= E \left\{ \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix} \right\} \\ &= \begin{bmatrix} E\{x_{11}\} & E\{x_{12}\} & \cdots & E\{x_{1n}\} \\ E\{x_{21}\} & E\{x_{22}\} & \cdots & E\{x_{2n}\} \\ \vdots & \vdots & \ddots & \vdots \\ E\{x_{m1}\} & E\{x_{m2}\} & \cdots & E\{x_{mn}\} \end{bmatrix} . \end{aligned} \quad (2.7)$$

2.5 Varianca

Poenostavljeni bi lahko rekli, da z varianco ocenimo širino porazdelitev gostote verjetnosti. Nizka varianca pomeni, da so realizacije blizu pričakovane vrednosti zelo verjetne, pri visoki varianci pa je visoka verjetnost, da bodo realizacije lahko tudi zelo oddaljene od pričakovane vrednosti.

Varianco naključne spremenljivke izračunamo kot drugi moment, vsoto kvadratov odmikov od pričakovane vrednsoti, ki je utežena z ustrezeno verjetnostjo:

$$\text{var}\{x\} = \sigma_x^2 = E\{(x - \mu_x)^2\} = \int_{-\infty}^{\infty} (x - \mu_x)^2 p(x) dx. \quad (2.8)$$

Posledično je varianca stohastičnega procesa enaka

$$\begin{aligned} \text{var}\{x(k)\} &= \sigma_x^2(k) = E\{(x(k) - \mu_x(k))^2\} \\ &= \int_{-\infty}^{\infty} (x(k) - \mu_x(k))^2 p(x, k) dx. \end{aligned} \quad (2.9)$$

Pri izračunu variance lahko ločimo dva različna primera: (i) srednja vrednost stohastičnega procesa je znana, (ii) srednja vrednost stohastičnega procesa je ocenjena z enačbo 2.5.

Pri znani srednji vrednosti μ_x dobimo (primer (i)) nepristransko oceno variance:

$$E\{(x - \mu_x)^2\} \approx \frac{1}{N} \sum_{i=1}^N (x(i) - \mu_x)^2. \quad (2.10)$$

Za primer (ii) z eksperimentalno ocenjeno srednjo vrednostjo \bar{x} je nepristranska ocena variance enaka

$$E\{(x - \bar{x})^2\} \approx \frac{1}{N-1} \sum_{i=1}^N (x(i) - \bar{x})^2 = s^2. \quad (2.11)$$

Ker je varianca σ^2 kvadratična mera razdalje, je večkrat enostavnejše stvari predstaviti z *standardno deviacijo* σ , ki je kvadratni koren variance.

Če imamo Gaussovo porazdelitev z določeno srednjo vrednostjo μ in standardno deviacijo σ , potem predstavljajo intervali okoli srednje vrednosti $[\mu \pm \sigma]$, $[\mu \pm 2\sigma]$ in $[\mu \pm 3\sigma]$, ki jih imenujemo ena, dva in tri-sigma interval, 68.3%, 95.4% in 99.7% celotne površine pod funkcijo porazdelitve gostote verjetnosti. Z drugimi besedamo, samo 31.7%, 4.6% in 0.3% vseh realizacij naključne spremenljivke je izven teh intervalov. Zaradi tega je standardna deviacija dobra intuitivna ocena o širini funkcije porazdelitve gostote verjetnosti. Na primer, intervali zaupanja v poglavju 4.1.3 predstavljajo interval $[\mu - \sigma, \mu + \sigma]$, kjer je σ standardna deviacija šuma. Tudi razmerje šum-signal je podano z razmerjem med standardno deviacijo šuma in signalom.

2.6 Korelacija in kovarianca

Lastnost povprečja in variance sta definirani za eno samo naključno spremenljivko ali stohastični proces. V nasprotju s *korelacijo* in *kovarianco*, ki sta definirani za *dve* naključni spremenljivki ali stohastična procesa, na primer x in y ali $x(k)$ in $y(k)$. Interpretacija korelacije in kovariance je pravzaprav enaka. Obe merita linearno odvisnost oziroma podobnost med dvema naključnima spremenljivkama ali procesoma. Za dve naključni spremenljivki x in y je korelacija definirana z

$$\text{corr}\{x, y\} = \text{corr}_{xy} = E\{x \cdot y\}, \quad (2.12)$$

kovarianca pa z

$$\text{cov}\{x, y\} = \text{cov}_{xy} = E\{(x - \mu_x) \cdot (y - \mu_y)\}. \quad (2.13)$$

Če sta μ_x in μ_y enaki nič, potem sta korelacija in kovarianca enaki. Vrednost korelacije je velika, če sta x in y zelo korelirani in majhna, če nista. Če sta spremenljivki enaki, t.j. $x = y$, potem kovarianca postane enaka varianci spremenljivke x : $\text{cov}\{x, x\} = \text{var}\{x\}$. Korelacijsko $\text{corr}\{x, x\}$ imenujemo tudi *avto-korelacija*. Korelacijsko $\text{corr}\{x, y\}$ (za $x \neq y$) pa se imenuje *križna-korelacija*.

Dve naključni spremenljivki sta *nekorelirani*, če velja

$$E\{x \cdot y\} = E\{x\} \cdot E\{y\} = \mu_x \cdot \mu_y. \quad (2.14)$$

Za stohastični proces sta korelacija in kovarianca definirani kot

$$\text{corr}\{x, y, k_1, k_2\} = \text{corr}_{xy}(k_1, k_2) = E\{x(k_1) \cdot y(k_2)\}, \quad (2.15)$$

$$\begin{aligned} \text{cov}\{x, y, k_1, k_2\} &= \text{cov}_{xy}(k_1, k_2) \\ &= E\{(x(k_1) - \mu_x(k_1)) \cdot (y(k_2) - \mu_y(k_2))\}. \end{aligned} \quad (2.16)$$

Korelacija in kovarianca sta odvisni od časa k_1 in k_2 . Če je stohastični proces stacionaren, potem statistične lastnosti niso odvisne od časa. Posledično sta korelacija in kovarianca odvisni samo od časovnega premika $\kappa = k_2 - k_1$ med $x(k_1)$ in $y(k_2)$. Dobimo naslednjo formulo, ki ni odvisna od časa)

$$\text{corr}\{x, y, \kappa\} = \text{corr}_{xy}(\kappa) = E\{x(k) \cdot y(k + \kappa)\}, \quad (2.17)$$

$$\text{cov}\{x, y, \kappa\} = \text{cov}_{xy}(\kappa) = E\{(x(k) - \mu_x) \cdot (y(k + \kappa) - \mu_y)\}. \quad (2.18)$$

V praksi je običajno tako, da ne poznamo porazdelitve gostote verjetnosti za izračun pričakovanih vrednosti. Zaradi tega predpostavimo ergodičnost in izračunamo korelacijsko in kovarianco iz časovnega povprečja. Če sta $x(k)$ in $y(k)$ meritve realizacije stohastičnega procesa za časovne trenutke $k = 1, \dots, N$, potem lahko korelacijsko in kovarianco ocenimo kot

$$\text{corr}_{xy}(\kappa) \approx \frac{1}{N - |\kappa|} \sum_{k=1}^{N-|\kappa|} x(k) y(k + \kappa), \quad (2.19)$$

$$\text{cov}_{xy}(\kappa) \approx \frac{1}{N - |\kappa| - 1} \sum_{k=1}^{N-|\kappa|} (x(k) - \bar{x}) (y(k + \kappa) - \bar{y}). \quad (2.20)$$

Obe gornji oceni sta nepristranski [53], ker s faktorjem pred vsoto, ki je enak številu členov v vsoti, delimo vsoto. Ker imamo na voljo samo N vzorcev, je $|\kappa| = N - 1$ največji možni časovni premik.

Čeprav so ocene nepristranske je ocena variance napake pri velikih premikih zelo velika ($\kappa \rightarrow N$), ker je vsota posledica majhnega števila členov. Zato je v tem primeru ocena zelo neuporabna. Zato se običajno rešuje problem z naslednjim oceno:

$$\text{corr}_{xy}(\kappa) \approx \frac{1}{N} \sum_{k=1}^{N-|\kappa|} x(k) y(k + \kappa), \quad (2.21)$$

$$\text{cov}_{xy}(\kappa) \approx \frac{1}{N-1} \sum_{k=1}^{N-|\kappa|} (x(k) - \bar{x}) (y(k + \kappa) - \bar{y}). \quad (2.22)$$

Avto-korelacija ($x = y$) $\text{corr}_{xx}(\kappa)$ nam izračuna korelacijo med $x(k)$ in časovno premaknjениm signalom $x(k - \kappa)$. Avto-korelacija je vedno simetrična.

Poseben primer je avto-korelacija *belega šuma*, kjer je korelacija enaka nič za vse $\kappa \neq 0$ in $\text{corr}_{xx}(0) = 1$. Posledično lahko rečemo, da je za določeno realizacijo belega šuma značilno, da meritve pri določenem časovnem trenutku nima nobene korelacije s katerimkoli drugim vzorcem v drugem časovnem trenutku. To pomeni, da z meritvijo v določenem trenutku ne dobimo nobene informacije o meritvi v naslednjem trenutku. Cilj modeliranja procesa je napaka med modelom in procesom, ki je enaka belemu šumu, ker to pomeni, da je vsa koristna informacija zapisana v modelu, ostanek pa je popolnoma naključen. Beli šum je samo teoretični signal, ker je njegova moč neskončna. Barvni šum dobimo s filtriranjem belega šuma.

Pogosto srečamo tudi *kovariančno matriko*, ki je definirana v nadaljevanju. Če je $\underline{x} = [x_1 \ x_2 \ \dots \ x_n]$ vektor naključnih spremenljivk je kovariančna matrika definirana z

$$\text{cov}\{\underline{x}\} = \text{cov}\{\underline{x}, \underline{x}\} = \begin{bmatrix} \text{cov}\{x_1, x_1\} & \text{cov}\{x_1, x_2\} & \cdots & \text{cov}\{x_1, x_n\} \\ \text{cov}\{x_2, x_1\} & \text{cov}\{x_2, x_2\} & \cdots & \text{cov}\{x_2, x_n\} \\ \vdots & \vdots & \ddots & \vdots \\ \text{cov}\{x_n, x_1\} & \text{cov}\{x_n, x_2\} & \cdots & \text{cov}\{x_n, x_n\} \end{bmatrix}. \quad (2.23)$$

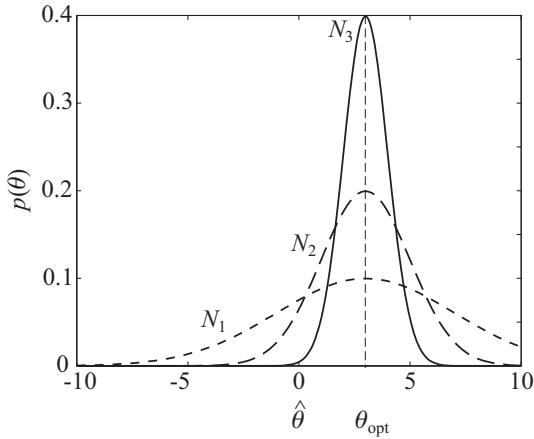
Diagonala vsebuje variance. Zato to matriko včasih imenujemo variančno-kovariančna matrika. Primer uporabe kovariančne matrike je pri oceni kvalitete estimacije parametrov $\hat{\underline{\theta}} = [\hat{\theta}_1 \ \hat{\theta}_2 \ \dots \ \hat{\theta}_n]^T$. Matrika $\text{cov}\{\hat{\underline{\theta}}\}$ vsebuje variance in kovariance ocjenjenega vektorja parametrov; glej (4.26). Drugi primer je kovariančna matrika šuma $\text{cov}\{\underline{n}\}$, kjer je $\underline{n} = [n(1) \ n(2) \ \dots \ n(N)]^T$ vektor realizacije šuma za merjene vzorce $i = 1, \dots, N$; glej (4.49).

Kovariančni vektor dobimo z naslednjo definicijo kovariance med vektorjem naključnih spremenljivk $\underline{x} = [x_1 \ x_2 \ \dots \ x_n]$ in skalarno naključno spremenljivko y :

$$\text{cov}\{\underline{x}, y\} = \begin{bmatrix} \text{cov}\{x_1, y\} \\ \text{cov}\{x_2, y\} \\ \vdots \\ \text{cov}\{x_n, y\} \end{bmatrix}. \quad (2.24)$$

2.7 Lastnosti estimatorjev

Če želimo oceniti kvaliteto estimacije, moramo ugotoviti lastnost estimatorja. Pri estimaciji vzamemo običajno veliko število meritov N , ki se na nek način preslikajo v manjše število parametrov modela $\underline{\theta} = [\theta_1 \ \theta_2 \ \dots \ \theta_n]^T$. Lahko jih statistično predstavimo s funkcijo porazdelitve gostote verjetnosti $p(\hat{\underline{\theta}}, N)$, ki je tudi funkcija števila meritov N . Slike 2.8 in 2.9 prikazujeta tri različne primere. Najpomembnejši lastnosti funkcije porazdelitve gostote verjetnosti sta pričakovana vrednost (povprečje) $E\{\hat{\underline{\theta}}\}$ in kovariančna matrika (varianca v enodimenzionalnem primeru) $\text{cov}\{\hat{\underline{\theta}}\}$.



Slika 2.8. Porazdelitev gostote verjetnosti nepristranskega estimatorja za različno število podatkov $N_1 < N_2 < N_3$. Pričakovana vrednost ocjenjenega parametra $\hat{\theta}$, ki je enaka pravi vrednosti θ_{opt} . Ta lastnost je neodvisna od števila podatkov, ki jih uporabimo pri ocenjevanju.

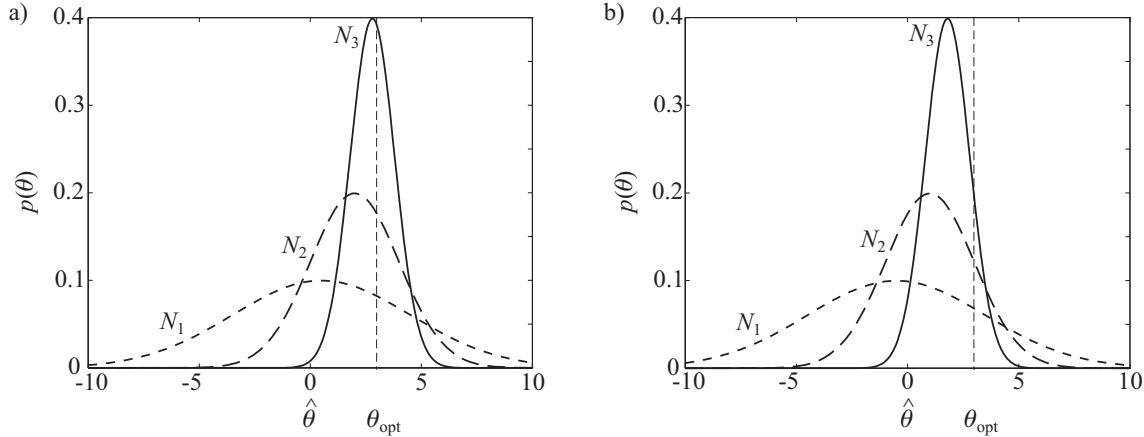
Če gre $N \rightarrow \infty$, je funkcije porazdelitve gostote verjetnosti pogosto Gaussove oblike, ki je enoumno definirana s povprečjem in kovariančno matriko.

Za analizo in interpretacijo funkcije porazdelitve gostote verjetnosti $p(\hat{\theta}, N)$ in njenih lastnosti si najprej oglejmo sliko 2.8, ki prikazuje tri funkcije porazdelitve gostote verjetnosti za enodimenzionalni problem $n = 1$ (en parameter). Pravi, toda neznan parameter, ki ga estimiramo ima vrednost θ_{opt} . Vzemimo N_1 vhodno-izhodnih meritev za oceno parametrov. Izhod je moten s stacionarnim šumom. Estimator da oceno $\hat{\theta}^{(1)}$. Nato za natančno isto vhodno množico, izmerimo novih N_1 meritev. Čeprav šum ne spremeni svojih statističnih lastnosti, dobimo drugo realizacijo motnje in posledično druge izhodne vrednosti. Zato dobimo tudi druge ocene parametra $\hat{\theta}^{(2)}$. Ponovimo postopek in dobimo različne ocene parametra, zaradi različnih motenj v obliki realizacije šuma. Porazdelitev gostote verjetnosti $p(\hat{\theta}, N_1)$ na sliki 2.8 prikazuje porazdelitev ocjenjenega parametra.

Dobra estimacija rezultira v ostri obliki funkcije porazdelitve gostote verjetnosti, ki daje vrednostim okoli pričakovane vrednosti bistveno večjo verjetnost kot ostalim. Optimalna funkcija porazdelitve gostote verjetnosti bi bila seveda Diracov impulz pri vrednosti θ_{opt} . Pomembna lastnost pa je tudi izboljševanje estimacije ob povečanju števila meritev.

Tri funkcije porazdelitve gostote verjetnosti na sliki 2.8 ustrezajo različnemu številu meritev N_1 , N_2 in N_3 , kjer velja $N_1 < N_2 < N_3$. Slika 2.8 prikazuje porazdelitev gostote verjetnosti $p(\hat{\theta}, N)$, ki postaja vedno bolj ostra, to pomeni, da njena varianca pada s povečevanjem števila učnih vzorcev N . Pri $N \rightarrow \infty$ gre varianca proti nič. Za večino estimatorjev varianca pada s faktorjem $1/N$. Ta relacija velja za skoraj vse estimatorje.

Primerjava funkcij porazdelitev gostote verjetnosti na slikah 2.8 in 2.9 kaže, da je zmanjšanje variance pri povečanem številu podatkov. Obnašanje povprečne vrednosti je različno. Na sliki 2.8 je povprečje estimiranih parametrov enako pravi vrednosti parametra θ_{opt} ne glede na število podatkov. Tak estimator se imenuje nepristranski.



Slika 2.9. a) Porazdelitev gostote verjetnosti *pristranskega* a *konsistentnega* estimatorja za različno število podatkov $N_1 < N_2 < N_3$. Pričakovana vrednost parametra $\hat{\theta}$ ni enaka pravi vrednosti θ_{opt} . S povečevanjem števila meritev se pričakovana vrednost ocjenjenega parametra približuje pravi vrednosti, če gre število proti neskončnosti. b) Porazdelitev gostote verjetnosti *nekonsistentnega* (in zato tudi *pristranskega*) estimatorja za različno število podatkov $N_1 < N_2 < N_3$. Pričakovana vrednost ocjenjenega parametra $\hat{\theta}$ ni enaka pravi vrednosti θ_{opt} in se ji ne približuje, če gre število meritev proti neskončnosti. Obstaja sistematična napaka med pričakovano vrednostjo ocene in pravo vrednostjo. To je seveda zelo nezaželeno.

Estimator je *nepristranski*, če za vsako število podatkov N velja

$$\mathbb{E}\{\hat{\theta}\} = \underline{\theta}_{\text{opt}}. \quad (2.25)$$

Drugače pa je estimator *pristranski*. Sistematična napaka med pričakovano vrednostjo ocjenjenih parametrov $\mathbb{E}\{\hat{\theta}\}$ in pravo vrednostjo $\underline{\theta}_{\text{opt}}$ imenujemo *pristranskost* (*ang. bias*):

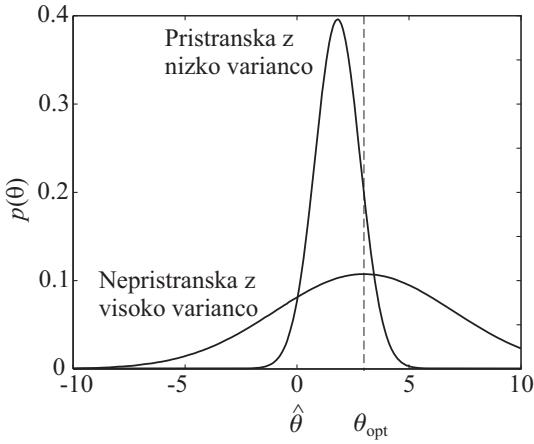
$$\underline{b} = \mathbb{E}\{\hat{\theta}\} - \underline{\theta}_{\text{opt}}. \quad (2.26)$$

V nasprotju z estimatorjem na sliki 2.8, ja na sliki 2.9a prikazan pristranski estimator. Povprečna ocena kaže sistematično napako glede na pravo vrednost parametra. S povečevanjem števila podatkov se deviacija zmanjšuje: $\underline{b}(N_1) > \underline{b}(N_2) > \underline{b}(N_3)$. Če gre N proti neskončnosti, gre ta napaka proti nič. Tak estimator ali ocena se imenuje *konsistenten ali asimptotično nepristranski*. Vsi nepristranski estimatorji so seveda tudi konsistentni.

Ocena je *konsistentna*, če gre s povečevanjem števila podatkov $N \rightarrow \infty$

$$\hat{\theta} \rightarrow \underline{\theta}_{\text{opt}}. \quad (2.27)$$

Čeprav je nepristranski estimator zelo zaželen, je dostikrat zadovoljiv tudi pristranski, ampak konsistenten estimator, saj dostikrat lahko zagotovimo zelo veliko število podatkov. Lahko se pokaže, da so vsi estimatorji največje podobnosti konsistentni. Slika 2.9b kaže sliko nekonsistentnega estimatorja. To pomeni, da ocena parametra ne konvergira k pravi vrednosti, če gre $N \rightarrow \infty$. Nekonsistentna estimacija ni sprejemljiva.



Slika 2.10. Pristranski estimator z nizko varianco proti nepristranskemu z visoko varianco.

Dober estimator je nepristranski ali pa vsaj konsistenten. Ta lastnost pa je precej neuporabna, če je varianca parametrov zelo velika. Zaradi tega mora biti poleg vsega tudi varianca v določenih mejah.

Slika 2.10 prikazuje zakaj je lahko pristranski estimator z nizko varianco boljši kot nepristranski estimator z veliko varianco. Kompromis med pristranskostjo in varianco je znan problem iz statistike, ki ga pogosto srečamo pri estimaciji parametrov. Metode, ki upoštevajo ta kompromis se imenujejo *regularizacijske tehnike*.

2.8 Koncepti pristranskosti in konvergencije

Meje integralov v teoretičnih formulah so neskončne. Pri praktičnih izračunih te meje nadomeščamo s končnimi vrednostmi. Veličina, ki jo na ta način dobimo je ocena dejanske veličine in jo označimo s strešico ($\hat{\cdot}$). V prisotnosti stohastičnih motenj se pojavita dve vprašanji o pravilnosti ocene. Ugotoviti je potrebno ali je matematično upanje ocene enako pravi vrednosti. Če to velja, pravimo, da je ocena *nepristranska*. Poleg tega pa rečemo, da je ocena *konsistentna*, če se z večanjem intervala opazovanja izboljuje in limitira k pravi vrednosti, ko gre interval opazovanja proti neskončnosti.

Pomembno vprašanje pa je tudi, če gre tudi varianca napake ocene proti nič, ko gre interval opazovanja proti neskončnosti. Če to velja, potem rečemo, da je ocena konsistentna v srednjekvadratični vrednosti.

Primer 2.8.1. Ocena srednje vrednosti stohastične spremenljivke

Srednjo vrednost stohastične spremenljivke x , to je

$$\bar{x} = E\{x(k)\} = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N x(k) , \quad (2.28)$$

ocenjujemo po formuli

$$\hat{x} = \frac{1}{N} \sum_{k=1}^N x(k) . \quad (2.29)$$

Ugotoviti je potrebno ali je ocena nepristranska, konsistentna in konsistentna v srednjekvadratični vrednosti. Poglejmo najprej matematično upanje ocene srednje vrednosti

$$E\{\hat{x}\} = E\left\{\frac{1}{N} \sum_{k=1}^N x(k)\right\} = \frac{1}{N} \sum_{k=1}^N E\{x(k)\} = \frac{1}{N} N \bar{x} = \bar{x}. \quad (2.30)$$

Matematično upanje ocene je torej enako pravi vrednosti že za končne intervale opazovanja N . Ocena je torej nepristranska in je seveda tudi konsistentna. Poglejmo, kako je z **varianco pogreška ocene srednje vrednosti**

$$E\{(\hat{x} - \bar{x})^2\} = E\left\{\left[\left(\frac{1}{N} \sum_{k=1}^N x(k)\right) - \bar{x}\right]^2\right\} = E\left\{\left[\frac{1}{N} \sum_{k=1}^N (x(k) - \bar{x})\right]^2\right\} \quad (2.31)$$

Če so posamezne vrednosti naključne spremenljivke x medsebojno statistično neodvisne (beli šum), velja $E\{[\sum_{k=1}^N (x(k) - \bar{x})]^2\} = \sum_{k=1}^N E\{(x(k) - \bar{x})^2\} = N\sigma_x^2$ (primer Matlab), ker velja

$$\begin{aligned} E\left\{\left[\sum_{k=1}^N (x(k) - \bar{x})\right]^2\right\} &= E\left\{\sum_{k=1}^N (x(k) - \bar{x})^2 + 2 \sum_{i=1}^N \sum_{j=1, j \neq i}^N (x(i) - \bar{x}) \cdot (x(j) - \bar{x})\right\} \\ &= E\left\{\sum_{k=1}^N (x(k) - \bar{x})^2\right\} + 2 \cdot E\left\{\sum_{i=1}^N \sum_{j=1, j \neq i}^N (x(i) - \bar{x}) \cdot (x(j) - \bar{x})\right\} \\ &= N\sigma_x^2 + 0 \end{aligned} \quad (2.32)$$

in je $E\left\{\sum_{i=1}^N \sum_{j=1, j \neq i}^N (x(i) - \bar{x}) \cdot (x(j) - \bar{x})\right\} = 0$.

Z uporabo zgornjega dobimo

$$\begin{aligned} E\{(\hat{x} - \bar{x})^2\} &= E\left\{\frac{1}{N^2} \sum_{k=1}^N (x(k) - \bar{x})^2\right\} \\ &= \frac{1}{N^2} \sum_{k=1}^N E\{(x(k) - \bar{x})^2\} \\ &= \frac{1}{N} \sigma_x^2, \end{aligned} \quad (2.33)$$

Varianca pogreška ocene se z naraščajočim intervalom opazovanja manjša in gre v limiti proti nič. Ocena je zato konsistentna v srednjekvadratični vrednosti

□

Primer 2.8.2. Ocena variance stohastične spremenljivke

Varianco stohastične spremenljivke x ocenujemo po formuli

$$\hat{\sigma}_x^2 = \frac{1}{N} \sum_{k=1}^N [x(k) - \hat{x}]^2. \quad (2.34)$$

Poskusimo ugotoviti ali je ocena nepristranska in konsistentna. Poglejmo najprej matematično upanje ocene variance.

$$E\{\hat{\sigma}_x^2\} = E\left\{ \frac{1}{N} \sum_{k=1}^N (x(k) - \hat{x})^2 \right\}. \quad (2.35)$$

S preoblikovanjem dobimo

$$\begin{aligned} \sum_{k=1}^N (x(k) - \hat{x})^2 &= \sum_{k=1}^N [(x(k) - \bar{x}) - (\hat{x} - \bar{x})]^2 \\ &= \sum_{k=1}^N [x(k) - \bar{x}]^2 + \sum_{k=1}^N (\hat{x} - \bar{x})^2 - 2(\hat{x} - \bar{x}) \sum_{k=1}^N [x(k) - \bar{x}] \\ &= \sum_{k=1}^N [x(k) - \bar{x}]^2 + N(\hat{x} - \bar{x})^2 - 2(\hat{x} - \bar{x})N(\hat{x} - \bar{x}) \\ &= \sum_{k=1}^N [x(k) - \bar{x}]^2 - N(\hat{x} - \bar{x})^2. \end{aligned} \quad (2.36)$$

Z upoštevanjem en. (2.33) dobimo

$$E\{\hat{\sigma}_x^2\} = \frac{1}{N} \left[\sum_{k=1}^N E\{[x(k) - \bar{x}]^2\} - NE\{(\hat{x} - \bar{x})^2\} \right] = \sigma_x^2 - \frac{1}{N} \sigma_x^2. \quad (2.37)$$

Ocena variance je za končne vrednosti N pristranska. Pristranskost je tem manjša, čim večji je N . Za $N \rightarrow \infty$ pristranskost izgine. Ocena je torej konsistentna.

□

Del II

Optimizacijske metode

3. Uvod v optimizacijo

V tem poglavju bo podan uvod v optimizacijo z vidika modeliranja in identifikacije. Z uporabo predlaganih metod lahko določimo parametre ali strukturo modela iz izmerjenih podatkov na procesu.

Lahko ločimo med tremi različnimi pristopi k optimizaciji. Razlikujejo se glede na informacijo, ki jo potrebujemo, da identificiramo model:

- nadzorovano učenje (*ang. supervised learning*),
- spodbujevano učenje (*ang. reinforcement learning*),
- nenadzorovano učenje (*ang. unsupervised learning*).

Nadzorovano učenje išče relacije v množici vhodno-izhodnih parov podatkov. Za vsak vhodni vektor poznamo vrednost izhoda. Cilj nadzorovanega učenja je v minimizaciji izbrane mere napake med izhodom procesa in izhodom modela, da dobimo najboljši možni model.

V primeru spodbujevanega učenja ne poznamo izhoda procesa za vsak vhodni vektor. Tipičen primer za to vrsto učenja so igre, kjer ne moremo določiti primernosti vsake poteze, lahko pa ugotovimo končni rezultat (zmaga, poraz, neodločeno), ki govori o ustreznosti uporabljene strategije. Vsako nadzorovano učenje lahko pretvorimo v primer spodbujevano učenje, če ne upoštevamo določene informacije.

Primer. Učenje sistema invertiranega nihala, da izregulira palico v navpični legi. če v tem primeru upoštevamo kot odmika od navpičnice in njegov odvod, potem lahko stvari učimo nadzorovano. Če pa za učenje uporabimo samo informacijo o tem, če je bilo učenje v končni fazi uspešno ali ne, če sistem izregulira palico v zgornji legi, potem pa govorimo o spodbujevanem učenju.

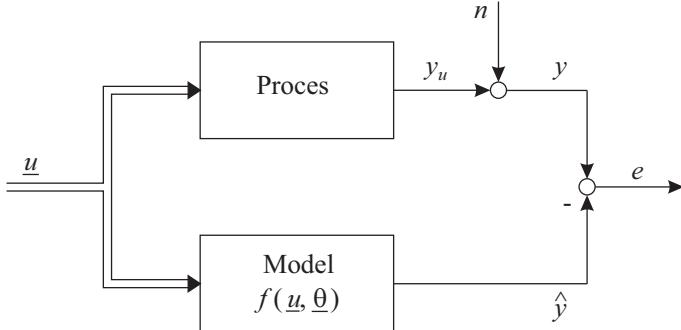
Metoda je zanimiva pri učenju raznih strategij takrat, ko nimamo nobene informacije o želenem izhodu modela in zato tudi ne moremo izmeriti trenutnega odstopanja od želene vrednosti.

V primeru nenadzorovanega učenja delamo samo na vhodnih podatkih. Glavni namen nenadzorovanih metod je v iskanju podobnosti znotraj množice podatkov in njegovo grupiranje v skupine ali roje. Ta metoda učenja je uporabna predvsem pri predprocesiranju podatkov. Natančneje je ta vrsta učenja obravnavana v poglavju 7.

Model, ki ga lahko opišemo s funkcijo $f(\cdot)$ preslika vhodni vektor \underline{u} v skalarni izhod modela y . Model je parametriziran z n parametri, ki jih lahko zapišemo v vektor parametrov $\underline{\theta}$ tako, da je $\hat{y} = f(\underline{u}, \underline{\theta})$. Cilj optimizacije je poiskati take parametre $\underline{\theta}$, da bo aproksimacija izhoda modela \hat{y} najbolje opisala izmerjeni izhod procesa y , ki je moten s šumom \mathcal{N} . Optimizacija je iskanje točke rešitve v n -dimenzionalnem prostoru parametrov modela $\underline{\theta}$.

3.1 Pregled optimizacijskih metod

Nadzorovane metode učenja lahko razdelimo na tri skupine: linearne, nelinearne lokalne in nelinearne globalne optimizacijske metode, kar prikazuje slika 3.2



Slika 3.1. Proces in model.

3.2 Kriterijske funkcije za nadzorovano učenje

3.2.1 Metoda vsote najmanjših kvadratov

V primeru nadzorovanega učenja uporabljamo določene kriterijske funkcije, ki bodo obravnavane v nadaljevanju. Optimiramo napako $e(i)$, ki je definirana kot razlika med izmerjenimi izhodi modela $y(i)$ in izhodom modela $\hat{y}(i)$ pri celotni množici učnih podatkov $i = 1, \dots, N$, kjer je N število vseh meritev (glej sliko 3.3). Ker je navadno izmerjeni izhod modela $y(i)$ moten s šumom \mathcal{N} , prvega izhoda procesa sploh ne poznamo $y_u(i)$.

Prva kriterijska funkcija je vsota kvadratov napake (*ang. Least Squares - LS*)

$$I(\underline{\theta}) = \sum_{i=1}^N e^2(i) \quad \text{kjer je } e(i) = y(i) - \hat{y}(i) \quad (3.1)$$

Metoda je veliko uporabnosta dosegla zaradi tega, ker minimum dosežemo, ko je gradient enak nič. To vodi k linearemu sistemu enačb, če je napaka linearna v neznanih optimiranih parametrih, ki je enostavno rešljiv. Druga dobra lastnost je kvadratično uteževanje napake, kar vodi v rešitev z velikim številom majhnih odstopanj od izmerjenih vrednosti, ne pa majhnim številom velikih odstopanj. To dejstvo pa govori tudi o tem, da je metod občutljiva na izjemne podatke (*ang. outlier*).

Kriterijsko funkcijo vsote kvadratov napake lahko razširimo v obliko, kjer je vsak prispevek utežen z utrežjo q_i in dobimo metodo vsote uteženih kvadratov napake

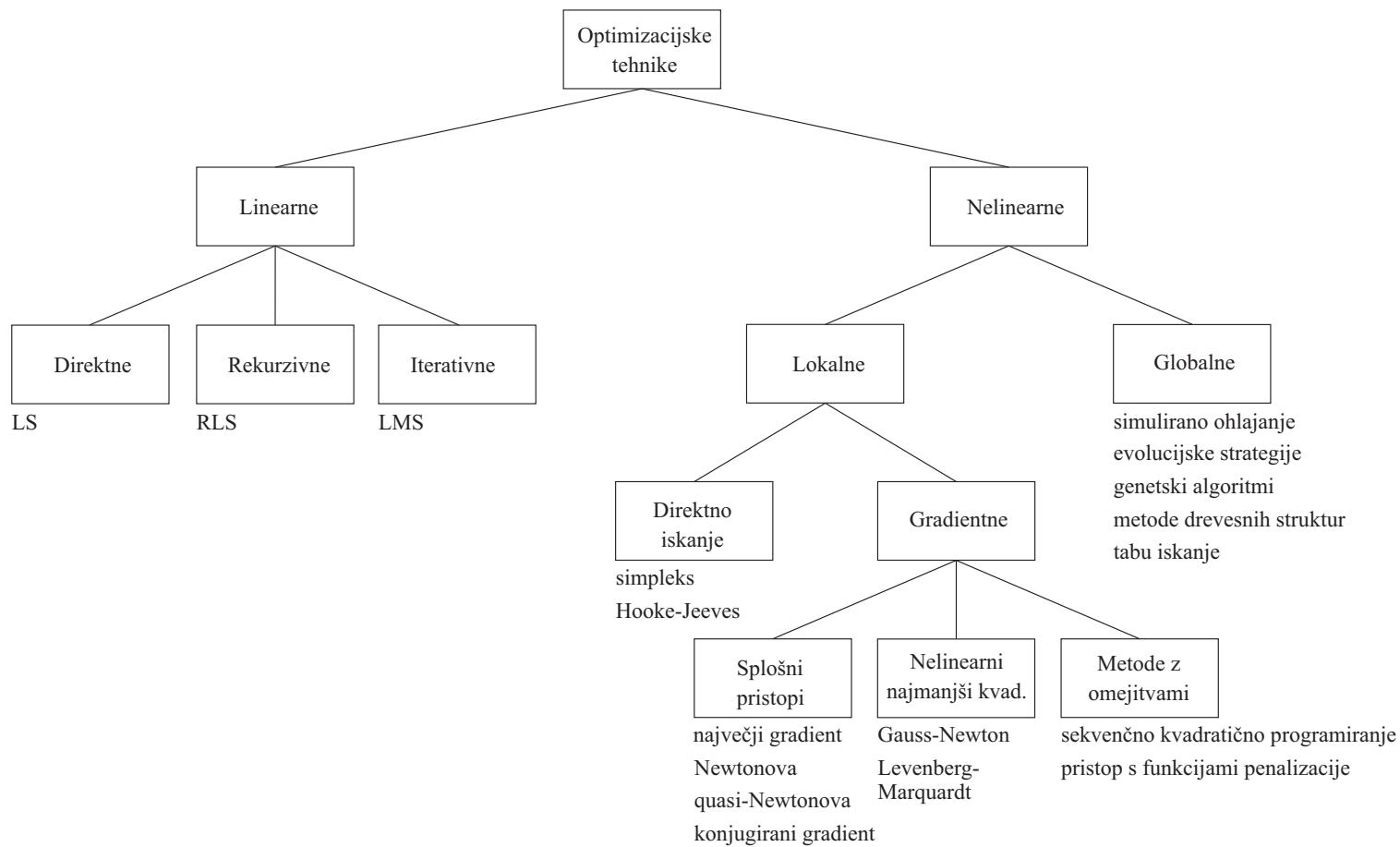
$$I(\underline{\theta}) = \sum_{i=1}^N q_i e^2(i) . \quad (3.2)$$

Metodo imeujemo tudi metoda uteženih najmanjših kvadratov (*ang. weighted least squares, WLS*).

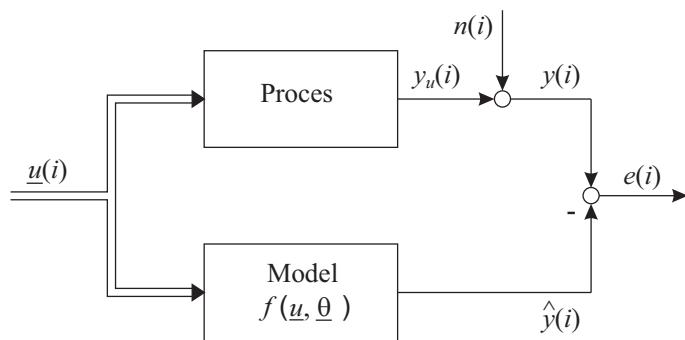
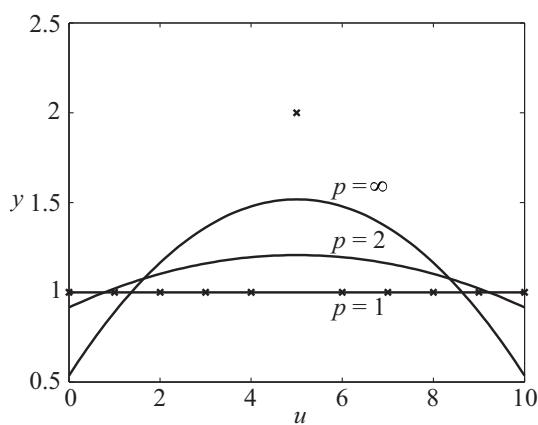
Generalizacija te metode pa je kriterijska funkcija oblike

$$I(\underline{\theta}) = \left(\sum_{i=1}^N q_i \|e(i)\|^p \right)^{1/p} . \quad (3.3)$$

Poleg izbire $p = 2$ se navadno uporablja tudi kriterijske funkcije, kjer je $p = 1$ (v tem primeru je to absolutna vrednost pogreška) in $p = \infty$ (kar pomeni največjo napako). Slika 3.4 prikazuje prilagajanje polinoma drugega reda na 11 vhodno-izhodnih parov podatkov, kjer minimiziramo tri različne kriterijske funkcije iz enačbe 3.3 in izberemo $p = 1, 2$ in ∞ .



Slika 3.2. Pregled linearnih in nelinearnih optimizacijskih metod.

Slika 3.3. Proces in model za vzorec i .Slika 3.4. Prilagajanje polinoma drugega reda na 11 vhodno-izhodnih parov za kriterijsko funkcijo iz enačbe 3.3 pri $p = 1, 2$, in ∞ . Višji red p vodi k višji občutljivosti na izjeme (5,2).

3.2.2 Metoda največje podobnosti

Nobena od prejšnjih metod ni vključevala predpostavk o motilnem šumu \mathcal{N} . Lahko pa pokažemo, da metoda najmanjših kvadratov implicitno predpostavlja nekoreliran šum \mathcal{N} (t.j., $E\{n(i)n(j)\} = 0$ za $i \neq j$, če je $E\{n(i)\} = 0$) z Gaussovo distribucijo, srednjo vrednostjo enako nič in varianco, ki je enaka σ^2 . Na ta način ima pogrešek $e(i)$ enako distribucijo kot šum \mathcal{N} , gostota verjetnosti naključne spremenljivke (*ang. probability density function - pdf*) pogreška modela $e(i)$ je enaka

$$p(e(i)) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2\sigma^2}e^2(i)\right). \quad (3.4)$$

Ker predpostavimo, da so napake med seboj neodvisne, je funkcija največje podobnosti enaka produktu vseh funkcij gostot za vsak vzorec

$$L(e(1), e(2), \dots, e(N)) = p(e(1)) \cdot p(e(2)) \cdot \dots \cdot p(e(N)), \quad (3.5)$$

ali enako

$$L(e(1), e(2), \dots, e(N)) = (2\pi\sigma^2)^{-\frac{N}{2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^N e(i)^2\right). \quad (3.6)$$

Optimalni parametri so tisti, ki maksimizirajo funkcijo podobnosti v enačbi 3.5. Z uporabo logaritma, prevedemo problem spet v iskanje minimuma in dobimo

$$I(\underline{\theta}) = -\ln(p(e(1))) - \ln(p(e(2))) - \dots - \ln(p(e(N))), \quad (3.7)$$

ali drugače

$$I(\underline{\theta}) = -\frac{N}{2} \ln(2\pi) - \frac{N}{2} \ln(\sigma^2) - \frac{1}{2\sigma^2} \sum_{i=1}^N e(i)^2. \quad (3.8)$$

Enostavno lahko vidimo, da za šum z Gaussovo distribucijo in srednjo vrednost, ki je enaka nič, enačba 3.7 rezultira v kriterijski funkciji uteženih najmanjših kvadratov.

Največkrat porazdelitvena gostota (pdf) šuma $p(n)$ in posledično tudi porazdelitev pogreška $p(e)$ nista znani in zato metode maksimalne podobnosti ne moremo direktno uporabiti. Običajno pa je predpostavka o normalni distribuciji upravičena, ker je šum največkrat sestavljen iz števila različnih izvorov. Po centralnem limitnem teoremu vsota porazdelitvenih funkcij poljubno distribuiranih naključnih spremenljivk limitira k Gaussovi distribuciji, če število naključnih spremenljivk narašča. Na ta način je predpostavka Gaussove distribucije upravičena in je najpomembnejši kriterij kar vsota kvadratov napake. V tem primeru sta metodi največje podobnosti in najmanjših kvadratov enaki. Za drugačne šumne porazdelitve dobimo z uporabo principa največje podobnosti druge kriterijske funkcije. Na primer, za nekoreliran šum $n(i)$, ki ima dvojno eksponencialno porazdelitev [29]

$$p(e(i)) = \frac{1}{2\sigma} \exp\left(-\frac{|e(i)|}{\sigma}\right), \quad (3.9)$$

dobimo kriterijsko funkcijo

$$I(\underline{\theta}) = |e(1)| + |e(2)| + \dots + |e(N)|. \quad (3.10)$$

Lahko tudi pokažemo, da je v primeru enakomerno porazdeljene gostote šuma

$$p(e(i)) = \begin{cases} 1/2c & \text{for } |e(i)| \leq c \\ 0 & \text{for } |e(i)| > c \end{cases} \quad (3.11)$$

kriterijska funkcija, ki temelji na principu največje podobnosti enaka

$$I(\underline{\theta}) = \max(e(1), e(2), \dots, e(N)). \quad (3.12)$$

3.2.3 Metoda največje a-posterioriorne ocene in Bayesova metoda

Ideja teh dveh metod je modeliranje izmerjenega izhoda procesa $\underline{y} = [y_1 \ y_2 \ \dots \ y_N]^T$ in parametrov modela $\underline{\theta} = [\theta_1 \ \theta_2 \ \dots \ \theta_n]^T$ s skupno funkcijo gostote verjetnosti [31]. To pomeni, da so tudi parametri modela obravnavani kot naključne spremenljivke. Maksimalna a-posteriori ocena je dana s parametri, ki maksimizirajo pogojno funkcijo gostote verjetnosti

$$p(\underline{\theta}|\underline{y}) \longrightarrow \max_{\underline{\theta}} . \quad (3.13)$$

Ta funkcija gostote verjetnosti je znana kot *a-posteriori* funkcija porazdelitve gostote verjetnosti, ker opisuje porazdelitev parametrov po tem, ko smo uporabili podatke \underline{y} . Zaradi tege to imenujemo maksimalna a-posteriori (MAP) ocena. Lahko jo izračunamo z uporabo Bayesovega theorema, če poznamo *a-priorno* funkcijo porazdelitve gostote verjetnosti $p(\underline{\theta})$. A-priorno funkcijo porazdelitve gostote verjetnosti moramo oceniti na osnovi predhodnega znanja o parametrih. Prav tako pa moramo poznati pogojno porazdelitev gostote verjetnosti $p(\underline{y}|\underline{\theta})$. Ta izraža kako so merjeni izhodi odvisni od parametrov. Ta pogojna porazdelitev gostote verjetnosti ima "ostro" obliko, če je prisotnega malo šuma, ker takrat lahko z majhno negotovostjo na osnovi parametrov $\underline{\theta}$ ocenimo izhod \underline{y} .

Funkcija porazdelitvene gostote verjetnosti a-posteriorne ocene $p(\underline{\theta}|\underline{y})$ lahko izračunamo na osnovi Bayesovega theorema

$$p(\underline{\theta}|\underline{y}) p(\underline{y}) = p(\underline{y}, \underline{\theta}) = p(\underline{y}|\underline{\theta}) p(\underline{\theta}) \quad (3.14)$$

ki je enak

$$p(\underline{\theta}|\underline{y}) = \frac{p(\underline{y}|\underline{\theta})}{p(\underline{y}) p(\underline{\theta})} = \frac{p(\underline{y}|\underline{\theta}) p(\underline{\theta})}{\int p(\underline{y}, \underline{\theta}) d\underline{\theta}} \quad (3.15)$$

S to enačbo je a-priori verjetnost $p(\underline{\theta})$ za parametre, to je, ocena predno uporabimo meritve \underline{y} , preslikana v a-posteriori verjetnost $p(\underline{\theta}|\underline{y})$ z uporabo informacije, ki jo vsebujejo meritve. S povečevanjem števila meritov, ki jih upoštevamo pri izračunu, bolj "ostra" postaja a-posteriorna ocena funkcije gostote verjetnosti, to pomeni, da so parametri $\underline{\theta}$ vedno bolj natančno definirani.

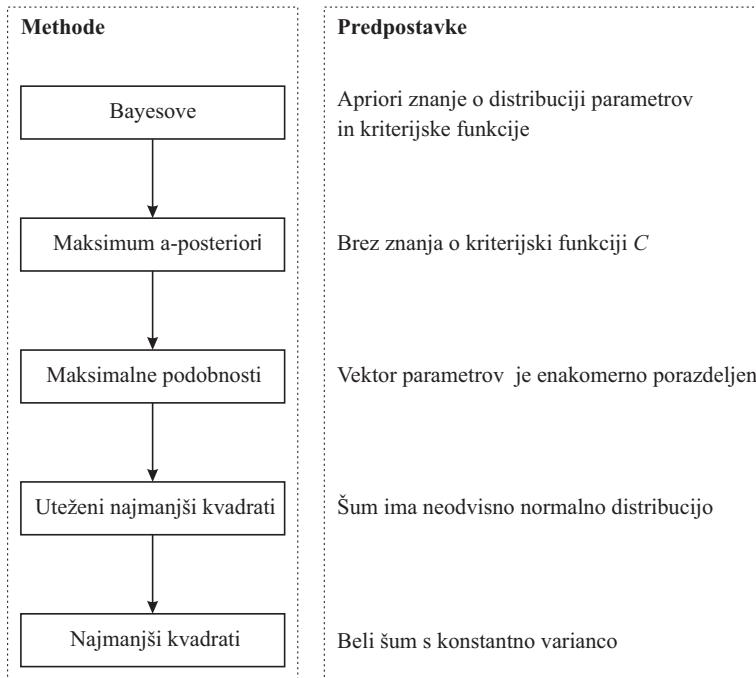
Bayesova metoda razširja MAP pristop z uvedbo funkcije cene (*ang. cost function*) $C(\underline{\theta})$. Opozorimo na to, da za razliko od kriterijske funkcije $I(\underline{\theta})$, funkcija cene $C(\underline{\theta})$ deluje *neposredno* na *parametrih* in ne na izhodu procesa in modela. Na ta način Bayesova metoda ne maksimizira a-posteriorne funkcije gostote verjetnosti, ampak jo ocenjuje skozi funkcijo cene

$$\int C(\underline{\theta}) p(\underline{\theta}|\underline{y}) d\underline{\theta} \longrightarrow \min_{\underline{\theta}} \quad (3.16)$$

Če nimamo nobene informacije o funkciji cene $C(\underline{\theta})$, potem jo izberemo konstantno. V tem primeru se Bayesova metoda poenostavi v MAP metodo.

V določenih primerih je a-priorna funkcija gostote verjetnosti $p(\underline{\theta})$ znana iz predhodnega modeliranja ali predhodnega znanja o procesu. Določeni parametri so lahko znani z manjšo nedoločenostjo od drugih. To lahko izrazimo vsaj kvalitativno z različnimi oblikami a-priorne gostote verjetnosti. MAP ali Bayesove ocene potem ohranijo to predhodno znanje tako, da vpliv novih meritov močnejši na parametra, ki so ocenjeni z večjo nedoločenostjo, in manjši, če so parametri ocenjeni z manjšo nedoločenoastjo. Ker je običajno na voljo zelo malo a-priornega znanja o procesu, se MAP in Bayesova metoda poenostavita v metodo največje podobnosti, ker upoštevamo lahko samo konstantne a-priorne funkcije gostote verjetnosti $p(\underline{\theta})$, to pomeni, da vse parametre upoštevamo z enako negotovostjo. Več lahko najdete v [95].

Odnos med metodo maksimalne a-posteriorne ocene, maksimalno podobnostjo, uteženih najmanjših kvadratov in najmanjših kvadratov je prikazan na sliki 3.5, [54].



Slika 3.5. Bayesova metoda je najsplošnejši pristop k optimizaciji, vendar zahteva poznavanje funkcij gostote verjetnosti. Metoda maksimalne a-posteriorne ocene, maksimalne podobnosti, uteženih najmanjših kvadratov in najmanjših kvadratov izvirajo iz Bayesove metode ob upoštevanju določenih pogojev, [31].

3.3 Kriterijske funkcije pri nenadzorovanem učenju

Do sedaj obravnavane kriterijske funkcije so v primeru nenadzorovanega učenja neuporabne, ker nimamo meritve izhoda $y(i)$, oziramo nimamo vhodno-izhodnaga para meritev. Ker imamo samo vhodne vektorje $\underline{u}(i)$ se kriterijske funkcije ravnajo po relacijah z okolico, oziroma med vhodnimi podatki. V vhodnem večdimensionalnem prostoru (hiperprostoru) iščemo razdalje med hipertočkami, lahko iščemo hiperpremice, hiperkulige, hiperelipsoide, itd. Torej je običajno kriterijska funkcija razdalja izmerjenega vhodnega vektorja do določenih hiper objektov v prostoru, v najenostavnejšem primeru npr. razdalja točke do centra roja ali do premice, ki določa roj.

4. Linearna optimizacija

Kadar je napaka med izhodom procesa in modela linearna v parametrih in zahtevamo optimalnost v smislu metode najmanjših kvadratov, potem se problem prevede v problem linearne optimizacije. Linearna optimizacija ima naslednje pomembne lastnosti:

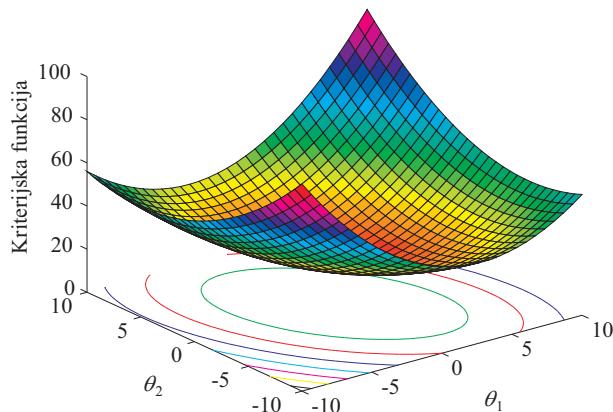
- eksistenca optimalne rešitve, ki je globalni optimum;
- površina kriterijske funkcije je hiperbola (slika 4.1) v obliki $\frac{1}{2}\underline{\theta}^T \underline{H} \underline{\theta} + \underline{h}^T \underline{\theta} + h_0$ z n -dimenzionalnim vektorjem parametrov $\underline{\theta}$, $n \times n$ -dimenzionalno matriko \underline{H} , n -dimenzionalnim vektorjem \underline{h} , in skalarjem h_0 ;
- vsak korak lahko izračunamo analitično;
- obstaja veliko numerično stabilnih in hitrih algoritmov;
- možne so rekurzivne oblike;
- lahko jo implementiramo v sprotni obliki.

Predpostavimo naslednjo strukturo izhoda modela \hat{y} , t.j. njegovo linearno odvisnost od n parametrov θ_i :

$$\hat{y} = \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n = \sum_{i=1}^n \theta_i x_i \quad \text{pri } x_i = g_i(\underline{u}). \quad (4.1)$$

Spremenljivke x_i imenujemo *regresorji* ali *neodvisne spremenljivke*, parametre θ_i imenujemo *regresijski koeficienti*, y imenujemo *odvisna spremenljivka*, in celoten problem imenujemo *linearna regresija*. Število parametrov je enako številu regresorjev.

V tem problemu morajo samo parametri θ_i nastopati linearno; regresorji x_i pa lahko nastopajo nelinearno in so funkcije vhodov \underline{u} . Na ta način lahko veliko število nelinearnih optimizacijskih problemov prevedemo v linearne. Z uporabo znanja o procesu ali intelligentnega predprocesiranja.



Slika 4.1. Površina kriterijske funkcije v primeru linearne optimizacijskega problema z dvema parametrom.

4.1 Metoda najmanjših kvadratov (LS)

Predpostavimo, da imamo $i = 1, \dots, N$ vhodno-izhodnih parov $\{\underline{u}(i), y(i)\}$, ki smo jih dobili iz meritev; glej sliko 3.3. Izhod procesa y je moten z belim šumom \mathcal{N}_y . Število regresorjev je enako n in jih lahko določimo na osnovi izmerjenih podatkov. Cilj je izračun izhoda modela \hat{y} , ki najbolje aproksimira izmerjeni izhod procesa y v smislu vsote najmanjših kvadratov. Pri tem seveda optimiramo linearno kombinacijo regresorjev in parametov, ki rezultira v izbiri optimalnih parametrov $\theta_1, \dots, \theta_n$.

Najprej definirajmo nekaj vektorjev in matrik, ki jih potrebujemo za zapis problema z n parametri in N učnimi pari:

$$\underline{e} = \begin{bmatrix} e(1) \\ e(2) \\ \vdots \\ e(N) \end{bmatrix}, \quad \underline{y} = \begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(N) \end{bmatrix}, \quad \hat{\underline{y}} = \begin{bmatrix} \hat{y}(1) \\ \hat{y}(2) \\ \vdots \\ \hat{y}(N) \end{bmatrix}, \quad \underline{n} = \begin{bmatrix} n(1) \\ n(2) \\ \vdots \\ n(N) \end{bmatrix}, \quad (4.2)$$

$$\underline{X} = \begin{bmatrix} x_1(1) & x_2(1) & \cdots & x_n(1) \\ x_1(2) & x_2(2) & \cdots & x_n(2) \\ \vdots & \vdots & & \vdots \\ x_1(N) & x_2(N) & \cdots & x_n(N) \end{bmatrix} \quad \text{in} \quad \underline{\theta} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}. \quad (4.3)$$

Stolpci v *regresijski matriki* \underline{X} so posamezni *regresijski vektorji*

$$\underline{x}_i = \begin{bmatrix} x_i(1) \\ x_i(2) \\ \vdots \\ x_i(N) \end{bmatrix} \quad \text{za} \quad i = 1, \dots, n. \quad (4.4)$$

Regresijsko matriko lahko zapišemo

$$\underline{X} = [\underline{x}_1 \ \underline{x}_2 \ \cdots \ \underline{x}_n]. \quad (4.5)$$

4.1.1 Razlaga metode najmanjših kvadratov

Vektorsko lahko izhod modela zapišemos $\hat{y} = \underline{X}\underline{\theta}$. Če to vnesemo v metodo najmanjših kvadratov, dobimo

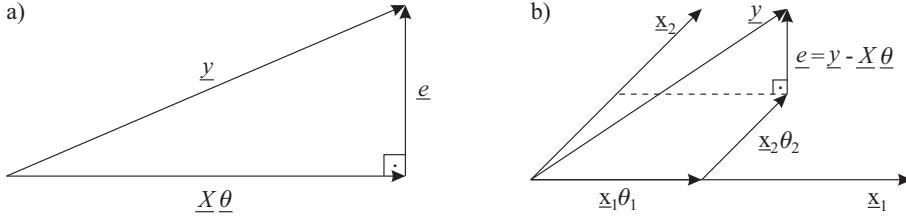
$$I(\underline{\theta}) = \frac{1}{2} \underline{e}^T \underline{e} \longrightarrow \min_{\underline{\theta}} \quad \text{pri} \quad \underline{e} = \underline{y} - \hat{\underline{y}} = \underline{y} - \underline{X}\underline{\theta}. \quad (4.6)$$

Kriterijska funkcija je tako enaka

$$I(\underline{\theta}) = \frac{1}{2} (\underline{y}^T \underline{y} - \underline{y}^T \underline{X} \underline{\theta} - \underline{\theta}^T \underline{X}^T \underline{y} + \underline{\theta}^T \underline{X}^T \underline{X} \underline{\theta}). \quad (4.7)$$

Kriterijska funkcija v enačbi 4.7 je parabolična funkcija, če jo predstavimo v parameterskem prostoru vektorja $\underline{\theta}$:

$$I(\underline{\theta}) = \frac{1}{2} \underline{\theta}^T \underline{H} \underline{\theta} + \underline{h}^T \underline{\theta} + h_0 \quad (4.8)$$



Slika 4.2. Optimum kriterijske funkcije pomeni, da je vektor $\underline{X}\underline{\theta}$ najbližje vektorju \underline{y} . Ortogonalni vektor \underline{e} je najmanjši in ortogonalen na vse regresorje matrike \underline{X} : a) projekcija za n -regresorski problem, b) problem z dvema regresorjema.

s kvadratnim členom

$$\underline{H} = \underline{X}^T \underline{X}. \quad (4.9)$$

Ker velja $\underline{X}^T \underline{y} = \underline{y}^T \underline{X}$ je linearni člen enak

$$\underline{h} = -\underline{X}^T \underline{y}. \quad (4.10)$$

Konstantna vrednost pa je enaka

$$h_0 = \frac{1}{2} \underline{y}^T \underline{y}. \quad (4.11)$$

Kvadratni člen imenujemo \underline{H} Hessova matrika, t.j. drugi odvod kriterijske funkcije.

V optimumu je odvod $I(\underline{\theta})$ po vektorju parametrov $\underline{\theta}$ enak nič. Iz tega sledi, da je pri optimumu pogrešek \underline{e} ortogonalen na vse regresorje \underline{x}_i (stolpce matrike \underline{X}):

$$\frac{\partial I(\underline{\theta})}{\partial \underline{\theta}} = \underline{H} \underline{\theta} + \underline{h}, \quad (4.12)$$

$$\frac{\partial I(\underline{\theta})}{\partial \underline{\theta}} = -\underline{X}^T (\underline{y} - \underline{X} \underline{\theta}) = -\underline{X}^T \underline{e} = \underline{0}. \quad (4.13)$$

Slika 4.2 prikazuje ortogonalnost med pogreškom in regresorji.

Dobimo rešitev po metodi najmanjših kvadratov

$$\hat{\underline{\theta}} = (\underline{X}^T \underline{X})^{-1} \underline{X}^T \underline{y}. \quad (4.14)$$

Pogrešek \underline{e} med izmerjenim izhodom procesa \underline{y} in izhodom modela $\hat{\underline{y}} = \underline{X} \hat{\underline{\theta}}$, ki smo ga dobili z metodo najmanjših kvadratov, se imenuje *residual*. V praktičnih primerih poskušamo skozi residual ugotoviti kvaliteto modela. Residual, ki ima lastnosti belega šuma, kaže na dobro modeliranje, ker je koristna informacija opisana z modelom, ostanek pa je medsebojno nekoreliran, kar je lastnost belega šuma.

Metoda najmanjših kvadratov z vidika korelacije. Metodo najmanjših kvadratov iz enačbe 4.14 lahko predstavimo tudi z vidika korelacije. Če izračunamo korelacijsko matriko $\underline{X}^T \underline{X}$

$$\underline{X}^T \underline{X} = N \begin{bmatrix} \frac{1}{N} \sum_{i=1}^N x_1^2(i) & \frac{1}{N} \sum_{i=1}^N x_1(i)x_2(i) & \cdots & \frac{1}{N} \sum_{i=1}^N x_1(i)x_n(i) \\ \frac{1}{N} \sum_{i=1}^N x_2(i)x_1(i) & \frac{1}{N} \sum_{i=1}^N x_2^2(i) & \cdots & \frac{1}{N} \sum_{i=1}^N x_2(i)x_n(i) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{N} \sum_{i=1}^N x_n(i)x_1(i) & \frac{1}{N} \sum_{i=1}^N x_n(i)x_2(i) & \cdots & \frac{1}{N} \sum_{i=1}^N x_n^2(i) \end{bmatrix} \quad (4.15)$$

in izračunamo $\underline{X}^T \underline{y}$

$$\underline{X}^T \underline{y} = N \begin{bmatrix} \frac{1}{N} \sum_{i=1}^N x_1(i)y(i) \\ \frac{1}{N} \sum_{i=1}^N x_2(i)y(i) \\ \vdots \\ \frac{1}{N} \sum_{i=1}^N x_n(i)y(i) \end{bmatrix}, \quad (4.16)$$

potem lahko oceno parametrov modela zapišemo

$$\hat{\underline{\theta}} = \text{corr}\{\underline{X}, \underline{X}\}^{-1} \cdot \text{corr}\{\underline{X}, \underline{y}\}, \quad (4.17)$$

kjer je $\text{corr}\{\underline{X}, \underline{X}\}$ avtokorelacijska funkcija, ki je definirana kot

$$\text{corr}\{\underline{X}, \underline{X}\} = \begin{bmatrix} \text{corr}\{\underline{x}_1, \underline{x}_1\} & \text{corr}\{\underline{x}_1, \underline{x}_2\} & \cdots & \text{corr}\{\underline{x}_1, \underline{x}_n\} \\ \text{corr}\{\underline{x}_2, \underline{x}_1\} & \text{corr}\{\underline{x}_2, \underline{x}_2\} & \cdots & \text{corr}\{\underline{x}_2, \underline{x}_n\} \\ \vdots & \vdots & \ddots & \vdots \\ \text{corr}\{\underline{x}_n, \underline{x}_1\} & \text{corr}\{\underline{x}_n, \underline{x}_2\} & \cdots & \text{corr}\{\underline{x}_n, \underline{x}_n\} \end{bmatrix} \quad (4.18)$$

in $\text{corr}\{\underline{X}, \underline{y}\}$ predstavlja križnokorelacijsko funkcijo

$$\text{corr}\{\underline{X}, \underline{y}\} = \begin{bmatrix} \text{corr}\{\underline{x}_1, \underline{y}\} \\ \text{corr}\{\underline{x}_2, \underline{y}\} \\ \vdots \\ \text{corr}\{\underline{x}_n, \underline{y}\} \end{bmatrix}. \quad (4.19)$$

Pogojenost regresijske matrike. V rešitvi problema najmanjših kvadratov (glej enačbo 4.14) imenujemo izraz $(\underline{X}^T \underline{X})^{-1} \underline{X}^T$ tudi *pseudo inverz* matrike \underline{X} , in ga včasih zapišemo kot \underline{X}^+ . Če ima \underline{X} poln rank n , kar zahteva vsaj toliko meritev kolikor je neznanih parametrov ($N \geq n$). Torej mora imeti matrika vsaj toliko vrstic kot ima stolpcov. Ta pogoj pa ni zadosten, ker morajo biti stolpci matrike linearno neodvisni.

Za izračun inverza matrike $\underline{X}^T \underline{X}$, ki dejansko predstavlja Hessovo matriko kriterijske funkcije, je odločilna njena stopnja pogojenosti

$$\underline{H} = \frac{\partial^2 I(\underline{\theta})}{\partial \underline{\theta}^2} = \underline{X}^T \underline{X}. \quad (4.20)$$

Samo v primeru dobre pogojenosti Hessianove matirke dobimo dobro oceno parametrov modela. Pogojenost matrike je definirana kot kvocient med največjo in najmanjšo lastno vrednostjo matrike \underline{H} in jo označimo z χ . Ker je matrika $\underline{H} = \underline{X}^T \underline{X}$ simetrična in pozitivno semi-definitna matrika so vse lastne vrednosti realne in nenegativne. Če je \underline{H} nesingularna, potem so vse lastne vrednosti striktno pozitivne (v primeru polnega ranga matrike)

$$\chi = \frac{\lambda_{\max}}{\lambda_{\min}}. \quad (4.21)$$

4.1.2 Kovariančna matrika estimiranih parametrov

Parametre modela lahko ocenimo z različno natančnostjo. Natančnost parametrov izrazimo s kovariančno matriko ocenjenih parametrov

$$\text{cov}\{\hat{\theta}\} = E \left\{ (\hat{\theta} - E\{\hat{\theta}\}) (\hat{\theta} - E\{\hat{\theta}\})^T \right\}, \quad (4.22)$$

kjer je

$$\hat{\theta} = (\underline{X}^T \underline{X})^{-1} \underline{X}^T \underline{y} \quad \text{in} \quad E\{\hat{\theta}\} = (\underline{X}^T \underline{X})^{-1} \underline{X}^T E\{\underline{y}\}. \quad (4.23)$$

V primeru, ko je $E\{\underline{X}\} = \underline{X}$, je \underline{X} deterministična. Zato dobimo

$$\hat{\theta} - E\{\hat{\theta}\} = (\underline{X}^T \underline{X})^{-1} \underline{X}^T (\underline{y} - E\{\underline{y}\}). \quad (4.24)$$

Ker velja, da je $\underline{y} - E\{\underline{y}\} = \underline{y} - E\{\underline{y}_u + \underline{n}_y\} = \underline{y} - \underline{y}_u = \underline{n}_y$, so ocene parametrov odvisne od lastnosti šuma. Zato iz enačbe 4.22 dobimo

$$\begin{aligned} \text{cov}\{\hat{\theta}\} &= E \left\{ ((\underline{X}^T \underline{X})^{-1} \underline{X}^T \underline{n}_y) ((\underline{X}^T \underline{X})^{-1} \underline{X}^T \underline{n}_y)^T \right\} \\ &= (\underline{X}^T \underline{X})^{-1} \underline{X}^T E\{\underline{n}_y \underline{n}_y^T\} \underline{X} (\underline{X}^T \underline{X})^{-1}, \end{aligned} \quad (4.25)$$

ker je Hessova matrika simetrična: $(\underline{X}^T \underline{X})^T = \underline{X}^T \underline{X}$. Če je \underline{n}_y beli šum z varianco σ^2 , potem $E\{\underline{n}_y \underline{n}_y^T\} = \sigma^2 I$ ($E\{\underline{n}_y(i) \underline{n}_y(i)\} = \sigma^2$ in $E\{\underline{n}_y(i) \underline{n}_y(j)\} = 0$ za $i \neq j$). Dobimo končno kovariančno matriko ocenjenih parametrov

$$\text{cov}\{\hat{\theta}\} = \sigma^2 (\underline{X}^T \underline{X})^{-1} = \sigma^2 \underline{H}^{-1}. \quad (4.26)$$

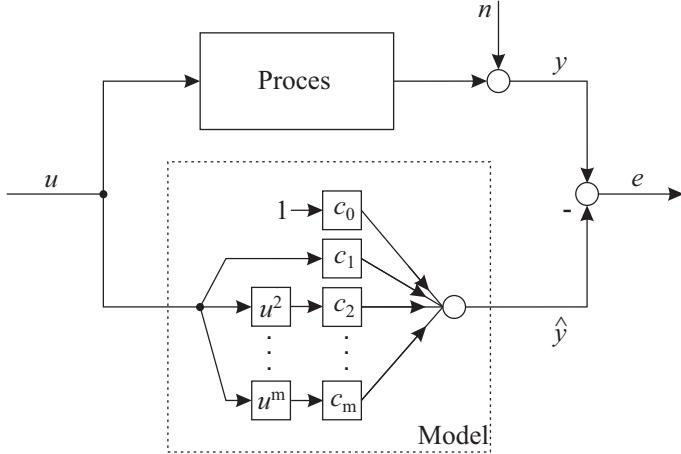
Diagonalni elementi so variance ocenjenih parametrov. Dobre oceno dobimo takrat, ko je signal regresorja \underline{x}_i velik in ima šum majhno varianco σ , to pomeni, da je razmerje med koristnim signalom in šumom je veliko.

Kovariančna matrika parametrov je proporcionalno odvisna od $1/N$. To pomeni, da lahko z zadostno količino vzorcev kompenziramo vpliv šuma. Lahko tudi ugotovimo, da je $-\underline{X}$ odvod vektorja \underline{e} glede na parametre $\underline{\theta}$, kar predstavlja občutljivost na te parametre. Zaključimo lahko, da pri manjši varianci ocene določenega parametra postane pogrešek bolj občutljiv na ta parameter. Če je napaka neodvisna od določenega parametra, potem tega parametra ne moremo oceniti, iz tega sledi, da gre njegova varianca proti neskončnosti. Enačba 4.26 nam daje relativno informacijo o natančnosti ocene parametrov.

Za določitev absolutnih vrednosti varianc ocenjenih parametrov $\text{cov}\{\hat{\theta}\}$ pa moramo najprej oceniti varianco šuma σ^2 , ki je običajno neznana. Ocenimo jo iz vektorja residualov [107]:

$$\hat{\sigma}^2 = \frac{\underline{e}^T \underline{e}}{N - n}. \quad (4.27)$$

Imenovalec predstavlja prostostno stopnjo residualov, to je število vzorcev, ki mu odštejemo število parametrov modela. Ocena variance residuala je dobljena na osnovi predpostavljenega dodanega belega šuma in optimalni strukturi modela, kar je precej nerealistično.



Slika 4.3. Polinomski model $\hat{y} = c_0 + c_1 u + c_2 u^2 + \dots + c_m u^m$.

Primer 4.1.1. Metoda najmanjših kvadratov v primeru polinomskega zapisa

Struktura našega modela je polinom m reda, s katerim bomo aproksimirali podatke. Izhod modela \hat{y} je (glej sliko 4.3)

$$\hat{y} = c_0 + c_1 u + c_2 u^2 + \dots + c_m u^m = \sum_{i=0}^m c_i u^i. \quad (4.28)$$

Dobimo pogrešek $e(k) = y(k) - \hat{y}(k)$, ki je enak (glej sliko 4.3)

$$e(k) = y(k) - c_0 - c_1 u - c_2 u^2 - \dots - c_m u^m. \quad (4.29)$$

Regresijska matrika \underline{X} pri N meritvah in vektor parametrov modela $\underline{\theta}$ sta enaka

$$\underline{X} = \begin{bmatrix} 1 & u(1) & u^2(1) & \dots & u^m(1) \\ 1 & u(2) & u^2(2) & \dots & u^m(2) \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & u(N) & u^2(N) & \dots & u^m(N) \end{bmatrix} \quad \underline{\theta} = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_m \end{bmatrix}. \quad (4.30)$$

Primer 4.1.2. Metoda najmanjših kvadratov za identifikacijo modela IIR strukture

Poglejmo si primer identifikacije modela v obliki časovno diskretnega linearne dinamičnega procesa s strukturo IIR (infinite impulse response) filtra. Model za enokoračno predikcijo izhoda procesa lahko v primeru dinamike m -tega reda zapišemo

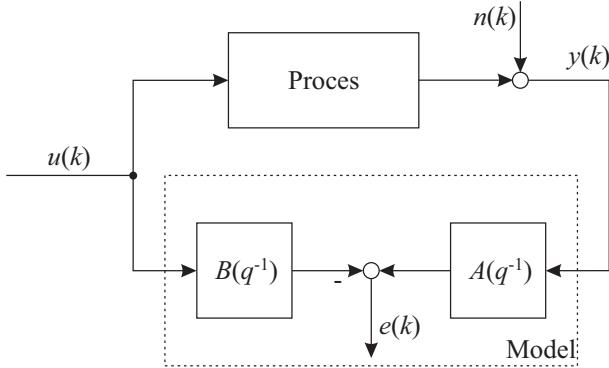
$$\hat{y}(k) = b_1 u(k-1) + \dots + b_m u(k-m) - a_1 y(k-1) - \dots - a_m y(k-m) \quad (4.31)$$

V tem primeru so določeni regresorji odvisni od izhoda procesa ($y(k-i)$, $i = 1, \dots, m$). To pomeni kršitev predpostavke (4.1), da so regresorji odvisni samo od vhodov.

Napako lahko zapišemo $e(k) = y(k) - \hat{y}(k)$ in dobimo

$$e(k) = y(k) + \dots + a_m y(k-m) - b_1 u(k-1) - \dots - b_m u(k-m). \quad (4.32)$$

Regresijska matrika \underline{X} pri N vzorcih meritev in vektor parametrov $\underline{\theta}$ so sedaj



Slika 4.4. IIR filter za enokoračno predikcijo in $B(q^{-1}) = b_1q^{-1} + b_2q^{-2} + \dots + b_mq^{-m}$ and $A(q^{-1}) = 1 + a_1q^{-1} + a_2q^{-2} + \dots + a_mq^{-m}$.

$$\underline{X} = \begin{bmatrix} u(m) & \cdots & u(1) & -y(m) & \cdots & -y(1) \\ u(m+1) & \cdots & u(2) & -y(m+1) & \cdots & -y(2) \\ \vdots & & \vdots & \vdots & & \vdots \\ u(N-1) & \cdots & u(N-m) & -y(N-1) & \cdots & -y(N-m) \end{bmatrix}, \quad (4.33)$$

$$\underline{\theta} = [b_1 \ \cdots \ b_m \ \ a_1 \ \cdots \ a_m]^T. \quad (4.34)$$

Regresijska matrika ima dimenzijs $(N - m) \times n$.

4.1.3 Interval zaupanja

Interval zaupanja omogoča, da ocenimo zanesljivost linearnega modela procesa. S stališča vedenja je pristop pomemben, ker lahko načrtamo delovanje regulatorja glede na zanesljivost modela. V območju natančnega modela je njegova akcija lahko bolj agresivna, v območju manjše zanesljivosti pa manj agresivna. V primeru prediktivnega vodenja lahko izbiramo horizont predikcije tako, da je tolerančni pas predikcije še vedno sprejemljiv.

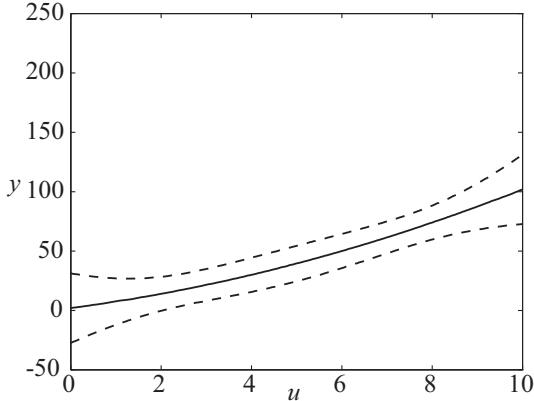
Koncept tolerančnega pasu temelji na učnih podatkih. Za model lahko pričakujemo, da bo dober na področju, kjer imamo veliko število podatkov in slabši na področju, kjer je število podatkov majhno. Parametre modela lahko ocenimo do določene natančnosti, ki jo zapišemo s kovariančno matriko parametrov $\text{cov}\{\hat{\underline{\theta}}\}$, ki hkrati določa natančnost izhoda modela pri določenem vhodu:

$$\begin{aligned} \text{cov}\{\hat{\underline{y}}\} &= E\left\{(\hat{\underline{y}} - E\{\hat{\underline{y}}\})(\hat{\underline{y}} - E\{\hat{\underline{y}}\})^T\right\} \\ &= E\left\{\left(X(\hat{\underline{\theta}} - E\{\hat{\underline{\theta}}\})\right)\left(X(\hat{\underline{\theta}} - E\{\hat{\underline{\theta}}\})\right)^T\right\} \\ &= \underline{X} E\left\{(\hat{\underline{\theta}} - E\{\hat{\underline{\theta}}\})(\hat{\underline{\theta}} - E\{\hat{\underline{\theta}}\})^T\right\} \underline{X}^T. \end{aligned} \quad (4.35)$$

Tako dobimo kovariančno matriko izhoda modela $\hat{\underline{y}}$

$$\text{cov}\{\hat{\underline{y}}\} = \underline{X} \text{cov}\{\hat{\underline{\theta}}\} \underline{X}^T \quad (4.36)$$

Tolerančni pas $\hat{\underline{y}}$ je definiran kot pozitivni in negativni odmik standardne deviacije (lahko tudi večkratnik standardne deviacije) od ocjenjenega izhoda modela:



Slika 4.5. Aproksimacija funkcije $y = 0.5u^2 + 5u + 2$ s polinomom drugega reda. Učni podatki so enakomerno razporejeni na intervalu $[0, 10]$. Prekinjena črta označuje tolerančni pas. Širina tolerančnega pasu se povečuje na robovih učnega intervala. To je posledica manjkajočih podakov levo od točke 0 in desno od točke 10 .

$$\hat{y} \pm \sqrt{\text{diag}(\text{cov}\{\hat{y}\})} \quad (4.37)$$

Če želimo določiti, t.i. interval zaupanja (*ang. confidence interval*) izhodnega signala modela, potem moramo zagotoviti, da bo izhod z določeno verjetnostjo ležal znotraj predpisanega intervala pri znani porazdelitveni funkciji šuma. Če predpostavimo Gaussovo distribucijo, potem interval σ določa 68.2% verjetnost, interval 2σ pa 95.4% verjetnost, da izhod leži znotraj intervala, [29].

Primer 4.1.3. Tolerančni pas in manjkajoči podatki

Učinek manjkajočih podatkov na tolerančni pas bomo predstavili na primeru funkcije

$$y = 0.5u^2 + 5u + 2 , \quad (4.38)$$

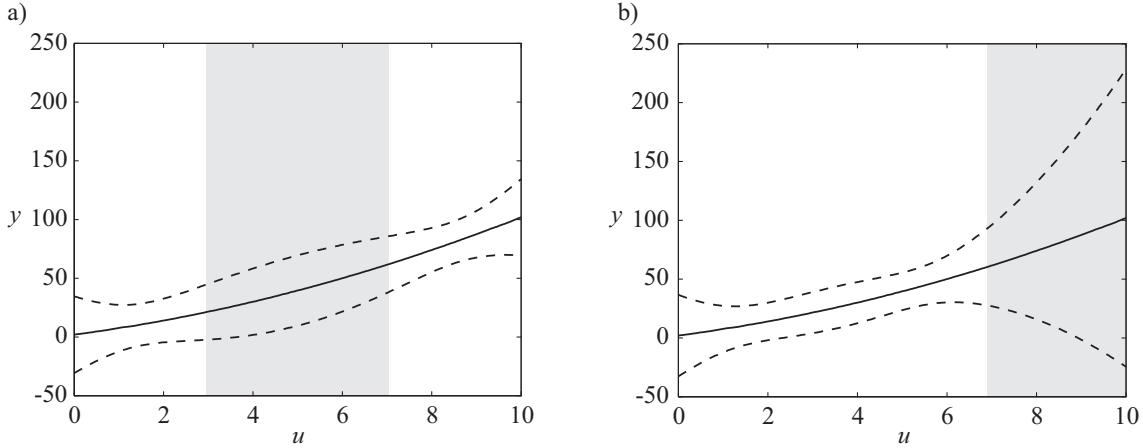
ki jo bomo aproksimirali s polinomom drugega reda na osnovi učne množice, ki vsebuje 1000 podatkov, ki so enakomerno porazdelejni v intervalu $[0, 10]$. Slike 4.5 in 4.6 prikazujeta tri različne primere: primer, kjer so vsi podatki uporabljeni za učenje na sliki 4.5), primer z manjkajočimi podatki v sredinskem področju (slika 4.6a), in primer, kjer podatki manjkajo na mejah intervala (slika 4.6b). Manjkajoči podatki v sredinskem področju niso tako kritični kot v primeru, ko podatki manjkajo na mejah intervala, ker lahko globalni aproksimatorji (primer polinomov) zapolnijo del področja z interpolacijo.

V primeru tolerančnega pasu je napaka posledica napake pri oceni parametrov modela, ki odstopajo od teoretičnih parametrov. Ne moremo pa upoštevati napake, ki nastane zaradi strukturne napake med modelom in procesom.

4.1.4 Ortogonalni regresorji

Zelo pomemben posebni primer metode najmanjših kvadratov dobimo, če so regresorji med seboj ortogonalni, i.e., $\underline{x}_i^T \underline{x}_j = 0$ za $i \neq j$. V tem primeru je Hessova matrika \underline{H} enaka

$$\underline{H} = \underline{X}^T \underline{X} = \text{diag}(\underline{x}_1^T \underline{x}_1, \underline{x}_2^T \underline{x}_2, \dots, \underline{x}_n^T \underline{x}_n) , \quad (4.39)$$



Slika 4.6. a) Manjkajoči podatki na intervalu $(3, 7)$. b) Manjkajoči podatki na intervalu $(7, 10]$. Napaka ocjenjenega izhoda modela se povečuje na področju, kjer ni učnih podatkov; glej sliko 4.5. Ugotovimo, da je tolerančni pas v primeru (b) veliko večji kot v primeru (a), čeprav več podatkov manjka v primeru (a). Razlog za to je v tem, da gre v primeru (b) pri $u > 7$ za ekstrapolacijski problem.

potem postane inverz enak

$$\underline{H}^{-1} = (\underline{X}^T \underline{X})^{-1} = \text{diag} \left(\frac{1}{\underline{x}_1^T \underline{x}_1}, \frac{1}{\underline{x}_2^T \underline{x}_2}, \dots, \frac{1}{\underline{x}_n^T \underline{x}_n} \right). \quad (4.40)$$

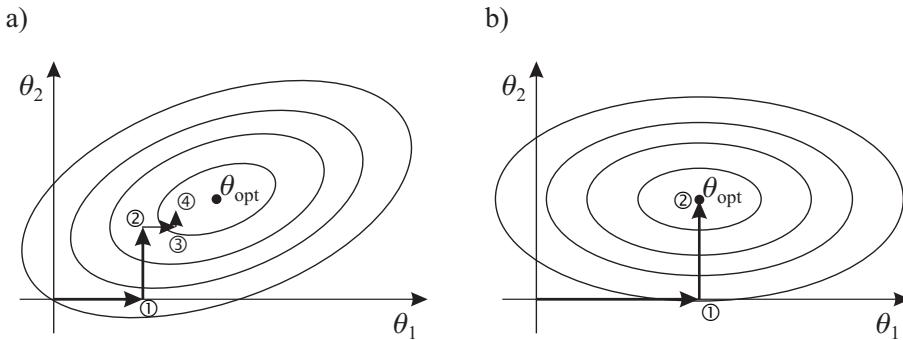
Optimalni parametri modela so potem enaki

$$\hat{\underline{\theta}} = (\underline{X}^T \underline{X})^{-1} \underline{X}^T \underline{y} = \text{diag} \left(\frac{1}{\underline{x}_1^T \underline{x}_1}, \frac{1}{\underline{x}_2^T \underline{x}_2}, \dots, \frac{1}{\underline{x}_n^T \underline{x}_n} \right) \underline{X}^T \underline{y}. \quad (4.41)$$

Vsak parameter θ_i iz vektorja $\underline{\theta}$ lahko določimo ločeno

$$\hat{\theta}_i = \frac{\underline{x}_i^T \underline{y}}{\underline{x}_i^T \underline{x}_i}. \quad (4.42)$$

Optimalni parameter $\hat{\theta}_i$ je odvisen samo od regresorja \underline{x}_i in merjenega izhoda \underline{y} . Med ostalimi regresorji v tem primeru ne sme biti korelacije. Za primer neortogonalnih regresorjev lahko rečemo, da z inverzom Hessianove matrike \underline{H}^{-1} dekoreliramo korelirane regresorje.



Slika 4.7. Konturni diagrami za kriterijske funkcije a) neortogonalnih in b) ortogonalnih regresorjev. Če so regresorji ortogonalni lahko izračunamo minimum funkcije tako, da izračnamo optimizacije po posameznih parametrih. To občutno zmanjša kompleksnost problema. V primeru neortogonalnih regresorjev moramo ponoviti več korakov optimizacije po posameznih parametrih. Bolj, ko so regresorji ortogonalni, boljša je pogojenost Hessianove matrike in hitrejša je konvergenca optimizacije.

4.1.5 Regularizacija

Problem slabe pogojenosti Hessove matrike se pojavlja zelo pogosto, posebno takrat, ko je število regresorjev veliko. Varianca najslabše ocjenjenega parametra modela v tem primeru zelo naraste. To pomeni, da povečevanje števila regresorjev, oziroma povečevanje kompleksnosti modela, vodi v slabše ocene parametrov modela. Zato je izbira števila regresorjev in s tem povečana fleksibilnost modela vedno povezana s slabostmi povečane variance ocjenjenih parametrov. Slaba pogojenost Hessove matrike je posledica slabega vzbujanja procesa. Z vzbujanjem moramo doseči, da se v izhodu odražajo vsi regresorji, edino na ta način lahko odkrijemo njihov vpliv.

Metode, kjer v kriterijski funkciji upoštevamo tudi vrednosti parametrov modela se imenujejo *regularizacijske metode*. Na ta način seveda posredno vplivamo tudi na varianco parametrov

$$I(\underline{\theta}, \alpha) = \frac{1}{2} (\underline{e}^T \underline{e} + \alpha |\underline{\theta}|^2) \longrightarrow \min_{\underline{\theta}} . \quad (4.43)$$

Ozadje dodajanja člena $\alpha |\underline{\theta}|^2$ je enostavno. Parametri, ki nimajo večjega vpliva na rešitev enačbe, težijo k majhnim vrednostim, ker s tem zmanjšamo vrednost kriterijske funkcije. Na ta način ostanejo samo parametri, ki so nujno potrebni, ker je njihov vpliv na izhodni signal pomembnejši kot pa prispevek v drugem členu kriterijske funkcije. Pomembnost prispevka tega člena vrednotimo z izbiro konstante α . Pozitivna konstanta α vodi vedno do pristranske ocene parametrov, ki pa ima manjšo varianco kot v primeru klasične metode najmanjših kvadratov, kjer je $\alpha = 0$ (glej enačbo 4.14).

Ocena parametrov v primeru regularizirane metode najmanjših kvadratov je enaka

$$\hat{\underline{\theta}} = (\underline{X}^T \underline{X} + \alpha \underline{I})^{-1} \underline{X}^T \underline{y} . \quad (4.44)$$

Ker je α dodan diagonalnim členom v Hessovi matriki $\underline{X}^T \underline{X}$, se njene lastne vrednosti λ_i , $i = 1, \dots, n$, spremenijo. Vpliv α na večje lastne vrednosti je zanemarljiv ($\lambda_i \gg \alpha$), manjše lastne vrednosti ($\lambda_i \ll \alpha$) pa virtualno postavimo na α . Iz tega sledi, da je pogojenost matrike neposredno odvisna od parametra α

$$\chi_{\text{reg}} \approx \frac{\lambda_{\max}}{\alpha} . \quad (4.45)$$

Primer 4.1.4. Regularizacija: pogojenost matrike

Pogojenost Hessove matrike

$$\underline{H} = \begin{bmatrix} 100 & 0 \\ 0 & 0.01 \end{bmatrix}$$

je enaka $\chi = 10000$. Pri $\alpha = 1$ pa je pogojenost modificirane Hessove matrike

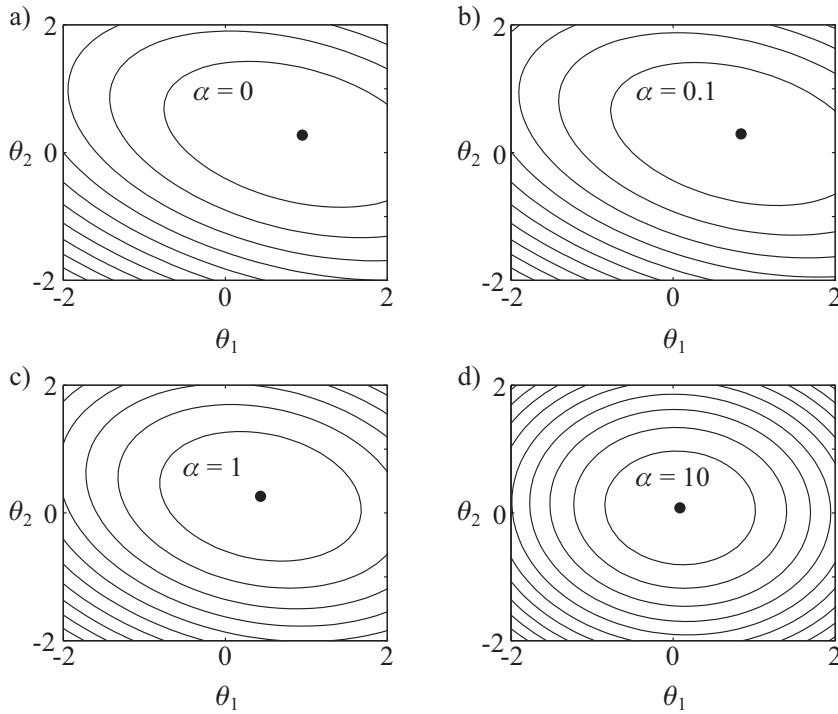
$$\underline{H}_{\text{reg}} = \begin{bmatrix} 100 & 0 \\ 0 & 0.01 \end{bmatrix} + 1 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 101 & 0 \\ 0 & 1.01 \end{bmatrix}$$

samo še $\chi_{\text{reg}} \approx 100$.

Primer 4.1.5. Regularizacija: konturni diagram

Ta primer ilustrira vpliv regularizacije na obliko kriterijske funkcije. V primeru je izbrana naslednja kriterijska funkcija

$$I(\underline{\theta}) = (\theta_1 - 1)^2 + 2(\theta_2 - 0.5)^2 + \theta_1 \theta_2 . \quad (4.46)$$



Slika 4.8. Konturne krivulje kriterijske funkcije (enačba 4.47) pri različnem regularizacijskem parametru α . Večja vrednost regularizacijskega parametra vodi k bolj okroglim konturam grafa, kjer je minimum blizu izhodišča koordinatnega sistema.

Konturni diagram je prikazan na sliki 4.8a. Minimum je pri vrednosti $\underline{\theta} \approx [0.86 \ 0.29]^T$. Regularizacija s parametrom α da naslednjo kriterijsko funkcijo

$$I(\underline{\theta}, \alpha) = (\theta_1 - 1)^2 + 2(\theta_2 - 0.5)^2 + \theta_1\theta_2 + \alpha(\theta_1^2 + \theta_2^2) . \quad (4.47)$$

Slike 4.8b–d prikazujejo učinek regularizacijskega parametra α na obliko konturnega diagrama. Če α narašča, se minimum kriterijske funkcije približuje točki $[0 \ 0]$ in konturni diagram prehaja od elipse proti krogu. To pomeni, da Hessova matrika postaja bolje pogojena.

Primer 4.1.6. Regresija z regularizacijo za polinomski model

Za ilustracijo regresije z regularizacijo bomo vzeli primer aproksimacije polinomske funkcije, kjer imamo majhno množico ošumljenih podatkov. Polinomska funkcija, ki jo uporabimo za generiranje podatkov je enaka

$$y = 1 + \sin 3u + \sin 4u . \quad (4.48)$$

Za zgornjo funkcijo imamo 41 enakomerno porazdeljenih meritev na intervalu $[-1, 1]$, ki so ošumljene s šumom, ki ima Gaussovo porazdelitev s srednjo vrednostjo enako 0 in varianco 0.1. Ob predpostavki, da ne poznamo strukture modela je najbolj naravno, da začnemo s strukturo polinoma prvega reda in postopno povečujemo red polinoma. Na koncu ugotovimo, da polinom 4. reda najbolje aproksimira podatke. Ocenjeni parametri so podani v tabeli 4.1. Ker je množica meritev majhna in zelo ošumljena, se ocenjeni parametri zelo razlikujejo od pravih parametrov. Ugotovimo lahko, da z metodo LS dobimo zelo velike variance ocenjenih parametrov.

Z uporabo regularizacijske regresije lahko izboljšamo rešitev problema ocene parametrov model, ki da rešitve, ki pa so močno odvisne od izbire regularizacijskega parametra α . Navadno se s povečevanjem vrednosti parametra α kriterijska funkcija najprej zmanjšuje, potem pa začne naraščati. V našem primeru je smo zapisali rešitev pri $\alpha \approx 0.1$, in predstavlja dober kompromis

Tabela 4.1. Primerjava med standardnimi najmanjšimi kvadrati in regularizacijsko metodo.

	Prava vrednost	LS	σ_{LS}^2	Regularizacija pri $\alpha = 0.1$	σ_α^2
c_0	1.0000	1.0164	0.0183	1.0139	0.0183
c_1	0.0000	-19.2580	28.3170	-0.0251	0.0755
c_2	0.0000	21.7994	31.5783	0.2098	0.0155
c_3	1.0000	-10.5965	16.3279	0.6824	0.664
c_4	1.0000	3.6492	3.5519	1.1300	0.0562

med napako ocene in varianco parametrov. Nadaljnje povečevanje regularizacijskega parametra poslabša aproksimacijo funkcije. Parametri, ki jih dobimo v primeru regularizacijske regresije so veliko bližji pravim vrednostim, kot v primeru navadnih najmanjših kvadratov. Na ta način lahko optimiramo tudi strukturo modela, saj lahko odstranimo regresorje, ki imajo parametre modela blizu vrednosti 0 ($\hat{c}_1 \approx \hat{c}_2 \approx 0$). Na ta način lahko zgornji postopek uporabimo kot alternativno metodo za optimiranje strukture modela, oziroma izbiro regresorjev. V nadaljevanju potem izberemo samo najpomembnejše regresorje $[1 \sin 3u \sin 4u]$.

Regularizacijska regresija je poseben primer t.i. *Tikhonkove regularizacije* [113], kjer namesto enotske matrike \underline{I} uporabimo posebno matriko.

4.1.6 Upoštevanje lastnosti šuma

Optimalna ocena parametrov modela je odvisna od lastnosti šuma. V primeru belega šuma dobimo z metodo najmanjših kvadratov najboljšo možno nepristransko oceno parametrov $\hat{\underline{\theta}}$.

V primeru, če šum ni bel pa je potrebno pri ocenjevanju upoštevati karakteristiko šuma

$$\hat{\underline{\theta}} = (\underline{X}^T \underline{\Omega}^{-1} \underline{X})^{-1} \underline{X}^T \underline{\Omega}^{-1} \underline{y}, \quad (4.49)$$

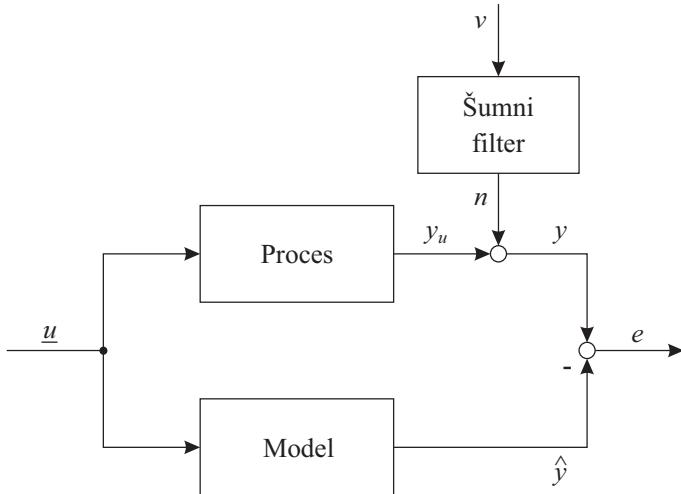
kjer imamo ocenjeno kovariančno matriko šumov $\underline{\Omega} = \text{cov}\{\underline{n}\}$. Za beli šum je varianca enaka σ^2 in je potem kovariančna matrika enaka $\sigma^2 \underline{I}$ in je zato enačba 4.49 enaka navadnim najmanjšim kvadratom. Če pa imamo barvni šum, potem v enačbi 4.49 uporabimo informacijo o šumu za oceno parametrov.

Kovariančna matrika šuma je diagonalne strukture

$$\underline{\Omega} = \begin{bmatrix} \Omega_{11} & 0 & \cdots & 0 \\ 0 & \Omega_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Omega_{NN} \end{bmatrix}. \quad (4.50)$$

Običajno je težko oceniti kovariančno matriko šuma. Lahko jo ocenimo preko residuov metode najmanjših kvadratov, vendar to delamo običajno samo pri zelo ošumljenih podatkih.

Izkaže se, da vsaka tudi zelo približna ocena zelo izboljša oceno parametrov modela. Zato je pomembno, da v primeru, ko pričakujemo šum, ki zelo odstopa od belega šuma, to upoštevamo pri oceni.



Slika 4.9. Modeliranje šuma n s filtriranjem belega šuma v .

4.1.7 Metoda uteženih najmanjših kvadratov (WLS)

Metodo najmanjših kvadratov lahko razširimo v metodo uteženih najmanjših kvadratov (*ang. weighted least squares*). Kriterijska funkcija se v tem primeru lahko zapiše kot

$$I(\underline{\theta}) = \frac{1}{2} \underline{e}^T \underline{Q} \underline{e}, \quad (4.51)$$

kjer je \underline{Q} matrika uteži. Najpogosteje ima matrika uteži diagonalno obliko

$$\underline{Q} = \begin{bmatrix} Q_{11} & 0 & \cdots & 0 \\ 0 & Q_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & Q_{NN} \end{bmatrix}, \quad (4.52)$$

v tem primeru je vsak kvadrat pogreška $e(i)$ utežen z ustreznim elementom matrike uteži Q_{ii} iz matrike \underline{Q} . Splošna rešitev za metodo uteženih najmanjših kvadratov je tako enaka

$$\hat{\underline{\theta}} = (\underline{X}^T \underline{Q} \underline{X})^{-1} \underline{X}^T \underline{Q} \underline{y}. \quad (4.53)$$

Za $\underline{Q} = q\underline{I}$, se enačba 4.53 povrne v navadno metodo najmanjših kvadratov iz enačbe 4.14. Metodo uteženih najmanjših kvadratov uporabljamo povsod tam, kjer imajo vzorci in s tem povezani pogreški različni pomen ali pomembnost za oceno parametrov. Zelo pomembna je pri ocenjevanju parametrov mehkih modelov v Takagi-Sugeno oblik. Lahko jo uporabimo tudi takrat, ko je v različnih področjih zahtevana različna toleranca, oziroma natančnost modela. V tem primeru izberemo velike uteži v področju velike natančnosti in manjše v področju večje tolerance.

Če zapišemo enačbo za izračun uteženih najmanjših kvadratov v navadni obliki, potem dobimo $\hat{\underline{\theta}} = (\tilde{\underline{X}}^T \tilde{\underline{X}})^{-1} \tilde{\underline{X}}^T \tilde{\underline{y}}$, kjer je $\tilde{\underline{X}} = \sqrt{\underline{Q}} \underline{X}$ in zato je $\tilde{\underline{X}}^T = \underline{X}^T \sqrt{\underline{Q}^T}$ in $\tilde{\underline{y}} = \sqrt{\underline{Q}} \underline{y}$. Kovariančno matriko ocenjenih parametrov pa zapišemo

$$\text{cov}\{\hat{\underline{\theta}}\} = \sigma_n^2 (\underline{X}^T \underline{Q} \underline{X})^{-1} \underline{X}^T \underline{Q} \underline{Q} \underline{X} (\underline{X}^T \underline{Q} \underline{X})^{-1}. \quad (4.54)$$

4.1.8 Filtrirna jedra

Do sedaj smo model zapisovali v obliki linearne kombinacije parametrov in regresorjev

$$\hat{\underline{y}} = \underline{X} \hat{\underline{\theta}} . \quad (4.55)$$

Lahko pa vektor parametrov zapišemo z $\hat{\underline{\theta}} = (\underline{X}^T \underline{X})^{-1} \underline{X}^T \underline{y}$. Na ta način dobimo direktno relacijo med merjenim izhodom procesa in aproksimacijo izhoda modela:

$$\hat{\underline{y}} = \underline{X} (\underline{X}^T \underline{X})^{-1} \underline{X}^T \underline{y} = \underline{S} \underline{y} . \quad (4.56)$$

Relacija med $\hat{\underline{y}}$ in \underline{y} je linear, ker je zapis linear. Matrika \underline{S} , ki je dimenzije $N \times N$ določa prispevek vsakega vzorca izhodnega signala k izhodu modela

$$\hat{y}(j) = s_{j1}y(1) + s_{j2}y(2) + \dots + s_{jN}y(N) = \underline{s}_j^T \underline{y} , \quad (4.57)$$

kjer je s_{ji} element matrike \underline{S} , in je \underline{s}_j^T vrstica matrike \underline{S} , ki predstavlja j -to meritvev. Na ta način lahko izhod modela v določenem trenutku vidimo kot filtrirano verzijo meritvev izhodnega signala. Vektor \underline{s}_j^T (vrstica \underline{S}) imenujemo filtrirno jedro. Zapis lahko tudi posplošimo na poljubno jedro K

$$\hat{y}(j) = \sum_{i=1}^n \theta_i K(u_i) . \quad (4.58)$$

4.2 Metoda rekurzivnih najmanjših kvadratov (RLS)

Do sedaj smo predpostavljali, da so vsi podatki $\{\underline{u}(i), y(i)\}$, ki tvorijo matriko \underline{X} in vektor \underline{y} za izvedbo metod na voljo že na začetku in jih obdelamo vse hkrati. V primeru, ko zaradi različnih razlogov zahtevamo, da metoda deluje v realnem času, takrat moramo metodo izvesti v *rekurzivni obliki* (*ang. Recursive Least Square - RLS*), ki izračuna spremembo parametrov modela glede na prejšnjo vrednost $\hat{\underline{\theta}}(k-1)$ vsakič, ko se pojavijo nove meritve. Metoda zahteva ekvidistančne časovne premike za adaptacijo parametrov.

Ideja je v izračunu novih ocenjenih parametrov $\hat{\underline{\theta}}(k)$ v časovnem trenutku k z dodajanjem korekcijskega vektorja prejšnjemu vektorju parametrov $\hat{\underline{\theta}}(k-1)$. Korekcijski vektor je odvisen od meritvev oziroma regresorja

$$\underline{x}(k) = [x_1(k) \ x_2(k) \ \cdots \ x_n(k)]^T$$

in izmerjenega izhoda procesa $y(k)$. V tem zapisu je regresor $\underline{x}(k)$ n dimenzionalni vektor, ki vsebuje meritve vseh vhodnih signalov, ki nastopajo v regresiji v k -ti meritvi. To pomeni k -to vrstico v matriki \underline{X}).

Rekurzivno metodo najmanjših kvadratov lahko zapišemo ([53])

$$\begin{aligned} \hat{\underline{\theta}}(k) &= \hat{\underline{\theta}}(k-1) + \underline{P}(k) \underline{x}(k) e(k) \\ \text{pri } \underline{P}^{-1}(k) &= \underline{P}^{-1}(k-1) + \underline{x}(k) \underline{x}^T(k) , \end{aligned} \quad (4.59)$$

kjer je $\underline{P}^{-1}(k) = \underline{X}^T(k)\underline{X}(k)$ aproksimacija Hessove matrike \hat{H} zato je $\underline{P}(k)$ proporcionalna kovariančni matriki ocenjenih parametrov; glej enačbo 4.26. Zaradi tega včasih površno imenujejo $\underline{P}(k)$ tudi kovariančna matrika. Enokoračna napaka predikcije $e(k) = y(k) - \underline{x}^T(k)\hat{\theta}(k-1)$ je napaka med merjenim izhodom procesa $y(k)$ in enokoračno predikcijo izhoda $\hat{y}(k|k-1) = \underline{x}^T(k)\hat{\theta}(k-1)$, kjer je vektor parametrov ocenjen iz vseh starih meritev.

RLS je v svojem konceptu tako imenovanih *inovacijskih* algoritmov, ki so znani na področju statistike in identifikacij. Za vsak nov podatek se inovacija izračuna na naslednji način:

$$\text{inovacija} = \text{nov podatek} - \text{predikcija novega podatka glede na obstoječi model}$$

Inovacija je nična, če lahko nov podatek brez napake predvidimo na osnovi starega modela. Inovacije so vedno nekorelirane z ostalimi podatki.

Korekcija je proporcionalna napaki predikcije, začetne vrednosti parametrov $\hat{\theta}(0)$ pa morajo zagotavljati stabilno delovanje sistema. Začetna vrednost $\underline{P}(k)$ je običajno izbrana kot $\underline{P}(0) = \alpha \underline{I}$ z zadost veliko vrednostjo α (100 ali 1000), ker to vodi k veliki začetni korekciji in hitri konvergenci parametrov. Seveda pa $\underline{P}(0)$ hkrati pomeni začetno varianco parametru $\hat{\theta}(0)$, ker je matrika \underline{P} proporcionalna kovariančni matriki parametrov.

RLS algoritem v enačbi 4.59 lahko intuitivno razložimo tudi kot uporabo gradientne optimizacijske tehnike. Člen $-\underline{x}(k)e(k)$ je gradient za nove meritve in je enak gradientu v izrazu 4.13 za en podatkovni vzorec. To pomeni, da je RLS korekcijski vektor negativna vrednost inverzne Hessove matrike $\underline{P}(k) = \hat{H}^{-1}(k)$ pomnoženo z gradientom.

RLS algoritem je rekurzivna verzija Newtonove optimizacijske metode in v primeru linearnega optimizacijskega problema ustreza metodi največjega spusta. Največji spust sledi nasprotni smeri gradienta in ga lahko nadomestimo z inverzom Hessove matrike. Dobimo metodo Widrowih najmanjših kvadratov (LMS)

$$\hat{\theta}(k) = \hat{\theta}(k-1) + \eta \underline{x}(k) e(k). \quad (4.60)$$

Izpeljava rekurzivne metode najmanjših kvadratov. Iz metode LS lahko zapišemo izračun parametrov v naslednjem trenutku k kot $\theta(k)$, ki je enak

$$\hat{\theta}(k) = \underline{P}(k) \underline{X}^T(k) \underline{y}(k), \quad (4.61)$$

kjer je matrika $\underline{P}(k)$ inverzna kovariančna matrika v trenutku k in jo zapišemo kot

$$\underline{P}(k) = (\underline{X}^T(k)\underline{X}(k))^{-1}.$$

Če ločimo podatkovno matriko $\underline{X}(k)$ na matriko regresorjev do trenutka $k-1$, ki jo zapišemo z $\underline{X}(k-1)$ in trenutni regresor $\underline{x}(k)$, dobimo naslednji zapis

$$\underline{P}(k) = \left(\begin{bmatrix} \underline{X}(k-1) \\ \underline{x}^T(k) \end{bmatrix}^T \begin{bmatrix} \underline{X}(k-1) \\ \underline{x}^T(k) \end{bmatrix} \right)^{-1}. \quad (4.62)$$

Z razvojem enačbe 4.62 dobimo

$$\underline{P}(k) = (\underline{X}^T(k-1)\underline{X}(k-1) + \underline{x}(k)\underline{x}^T(k))^{-1},$$

ali naprej

$$\underline{P}^{-1}(k) = \underline{P}^{-1}(k-1) + \underline{x}(k)\underline{x}^T(k). \quad (4.63a)$$

Sedaj lahko izraz iz enačbe 4.61 razširimo po zadnjih dveh členih in dobimo

$$\hat{\underline{\theta}}(k) = \underline{P}(k) (\underline{X}^T(k-1)\underline{y}(k-1) + \underline{x}\underline{y}(k)) .$$

Ker velja, da je $\underline{X}^T(k-1)\underline{y}(k-1) = \underline{P}^{-1}(k-1)\hat{\underline{\theta}}(k-1)$ iz tega sledi

$$\hat{\underline{\theta}}(k) = \underline{P}(k) (\underline{P}^{-1}(k-1)\hat{\underline{\theta}}(k-1) + \underline{x}\underline{y}(k)) . \quad (4.65)$$

Če enačbo 4.63a vnesemo v enačbo 4.65, dobimo naslednji zapis

$$\hat{\underline{\theta}}(k) = \underline{P}(k) ((\underline{P}^{-1}(k) - \underline{x}(k)\underline{x}^T(k))\hat{\underline{\theta}}(k-1) + \underline{x}\underline{y}(k)) .$$

Naprej dobimo

$$\hat{\underline{\theta}}(k) = \underline{P}(k)\underline{P}^{-1}(k)\hat{\underline{\theta}}(k-1) - \underline{P}(k)\underline{x}(k)\underline{x}^T(k)\hat{\underline{\theta}}(k-1) + \underline{P}(k)\underline{x}(k)\underline{y}(k) ,$$

kar nam da končno obliko

$$\hat{\underline{\theta}}(k) = \hat{\underline{\theta}}(k-1) + \underline{P}(k)\underline{x}(k) (y(k) - \underline{x}^T(k)\hat{\underline{\theta}}(k-1)) .$$

V enačbi 4.66, ki definira spremembo prejšnje vrednosti vektorja parametrov $\hat{\underline{\theta}}(k-1)$ nastopa matrika $\underline{P}(k)$. V enačbi 4.66 smo definirali izračun njene inverzne vrednosti. S pomočjo Shermann-Morrisonovega izreka o invertiranju, ki se glasi

$$(\underline{A} + \underline{b}\underline{c}^T)^{-1} = \underline{A}^{-1} - \frac{\underline{A}^{-1}\underline{b}\underline{c}^T\underline{A}^{-1}}{1 + \underline{c}^T\underline{A}^{-1}\underline{b}} ,$$

dobimo izračun matrike $\underline{P}(k)$ v rekurzivni obliki kot sledi

$$\underline{P}(k) = \underline{P}(k-1) - \frac{\underline{P}(k-1)\underline{x}(k)\underline{x}^T(k)\underline{P}(k-1)}{1 + \underline{x}^T(k)\underline{P}(k-1)\underline{x}(k)} .$$

4.2.1 Zmanjševanje računske kompleksnosti

Z uvedbo leme o invertiranju matrike dobimo poenostavljen algoritem RLS

$$\hat{\underline{\theta}}(k) = \hat{\underline{\theta}}(k-1) + \underline{\gamma}(k)e(k) , \quad e(k) = y(k) - \underline{x}^T(k)\hat{\underline{\theta}}(k-1) \quad (4.66a)$$

$$\underline{\gamma}(k) = \frac{1}{\underline{x}^T(k)\underline{P}(k-1)\underline{x}(k) + 1} \underline{P}(k-1)\underline{x}(k) \quad (4.66b)$$

$$\underline{P}(k) = (\underline{I} - \underline{\gamma}(k)\underline{x}^T(k))\underline{P}(k-1) . \quad (4.66c)$$

Rekurzivna metoda uteženih najmanjših kvadratov (RWLS), kjer je uteževanje podakov $\underline{x}(k)$ definirano z $q(k)$ postane enaka

$$\hat{\underline{\theta}}(k) = \hat{\underline{\theta}}(k-1) + \underline{\gamma}(k)e(k) , \quad e(k) = y(k) - \underline{x}^T(k)\hat{\underline{\theta}}(k-1) \quad (4.67a)$$

$$\underline{\gamma}(k) = \frac{1}{\underline{x}^T(k)\underline{P}(k-1)\underline{x}(k) + 1/q(k)} \underline{P}(k-1)\underline{x}(k) \quad (4.67b)$$

$$\underline{P}(k) = (\underline{I} - \underline{\gamma}(k)\underline{x}^T(k))\underline{P}(k-1) . \quad (4.67c)$$

Ta algoritem je po svoji kompleksnosti enak $\mathcal{O}(n^2)$, in ne več $\mathcal{O}(n^3)$ kot je to v primeru osnovnega RLS algoritma. Zaradi tega je pogosto uporabljen v praktičnih sprotnih aplikacijah.

4.2.2 Identifikacija časovno spremenljivih procesov

RLS v splošnem konvergira k optimalnim parametrom. V primeru velikega števila vzorcev ($k \rightarrow \infty$) se stopnja konvergence zmanjša, ker gre \underline{P} proti nič. To ni problematično, če gre za proces, ki je časovno nespremenljiv. Če pa gre za časovno spremenljive procese, potem to predstavlja problem. V tem primeru je največkrat potrebno ocenjevati časovno spremenljive parametre, kar lahko dosežemo, če poskrbimo za to, da \underline{P} ne postane premajhna. To lahko zagotovimo na več načinov. Eden od teh načinov je uvedba faktorja pozabljanja $\lambda \leq 1$. Podatek, ki smo ga zajeli j vzorcev nazaj je utežen z vrednostjo λ^j . Na ta način uvedemo eksponencialno pozabljanje v algoritmu rekurzivnih najmanjših kvadratov

$$\hat{\underline{\theta}}(k) = \hat{\underline{\theta}}(k-1) + \underline{\gamma}(k)e(k), \quad e(k) = y(k) - \underline{x}^T(k)\hat{\underline{\theta}}(k-1) \quad (4.68a)$$

$$\underline{\gamma}(k) = \frac{1}{\underline{x}^T(k)\underline{P}(k-1)\underline{x}(k) + \lambda} \underline{P}(k-1)\underline{x}(k) \quad (4.68b)$$

$$\underline{P}(k) = \frac{1}{\lambda} (\underline{I} - \underline{\gamma}(k)\underline{x}^T(k)) \underline{P}(k-1). \quad (4.68c)$$

V primeru uteženih najmanjših kvadratov s pozabljanjem, kjer je podatkovni vektor $\underline{x}(k)$ utežen z utežjo $q(k)$ dobimo

$$\hat{\underline{\theta}}(k) = \hat{\underline{\theta}}(k-1) + \underline{\gamma}(k)e(k), \quad e(k) = y(k) - \underline{x}^T(k)\hat{\underline{\theta}}(k-1) \quad (4.69a)$$

$$\underline{\gamma}(k) = \frac{1}{\underline{x}^T(k)\underline{P}(k-1)\underline{x}(k) + \lambda/q(k)} \underline{P}(k-1)\underline{x}(k) \quad (4.69b)$$

$$\underline{P}(k) = \frac{1}{\lambda} (\underline{I} - \underline{\gamma}(k)\underline{x}^T(k)) \underline{P}(k-1). \quad (4.69c)$$

Faktor pozabljanja λ ima običajno vrednost na intervalu med 0.9 in 1. Medtem, ko pri izbiri $\lambda = 1$ dobimo RLS brez pozabljanja in so vsi podatki enako uteženi ne glede na to kako daleč v preteklosti so. V primeru $\lambda = 0.9$ je nov podatek približno 3, 8, 24 in 68 krat bolj utežen kot podatki, ki so 10, 20, 30 in 40 vzorcev v preteklosti. Nastavljanje λ je kompromis med visoko robustnostjo na motnje (velika λ) in hitrega sledenja spremembam ocenjevanih parametrov procesa (majhna λ). Sistem lahko vrednost faktorja pozabljanja prilagaja glede na vzbujanje procesa. V primeru visokega vzbujanja je nova informacija velika in lahko faktor λ zmanjšamo, drugače pa ga povečamo. V primeru RLS je smiselno opazovati sled matrike \underline{P} in ustrezno reagirati, če se njena vrednost približuje nič. Matriko \underline{P} lahko resetiramo na začetno vrednost, ali pa s spremjanjem faktorja pozabljanja preprečimo, da sled matrike \underline{P} pade pod določeno vrednost.

4.3 Izbira regresorjev

V nadaljevanju si bomo ogledali kako iz množice merjenih n spremenljivk, ki so znane apriori, izbrati n_s najvplivnejših, ki bodo izbrane za regresorje pri iskanju parametrov modela. Iskanje najprimernejših regresorjev je zahteven problem strukturne identifikacije ali iskanje podmnožice regresorjev. Problem je obširno predstavljen v monografiji [81] in članku [15].

Merjeni izhod procesa \underline{y} lahko predstavimo kot predikcijo modela $\hat{\underline{y}} = \underline{X} \hat{\underline{\theta}}$ in napake \underline{e} (glej poglavje 4.1)

$$\underline{y} = \underline{X} \hat{\underline{\theta}} + \underline{e}, \quad (4.70)$$

ali v razširjeni obliki

$$\begin{bmatrix} y(1) \\ y(2) \\ \vdots \\ y(N) \end{bmatrix} = \begin{bmatrix} x_1(1) & x_2(1) & \cdots & x_n(1) \\ x_1(2) & x_2(2) & \cdots & x_n(2) \\ \vdots & \vdots & & \vdots \\ x_1(N) & x_2(N) & \cdots & x_n(N) \end{bmatrix} \begin{bmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \\ \vdots \\ \hat{\theta}_n \end{bmatrix} + \begin{bmatrix} e(1) \\ e(2) \\ \vdots \\ e(N) \end{bmatrix}. \quad (4.71)$$

Poglejmo si enačbo 4.71 pobližje. Naša naloga je, da zapišemo izmerjeni izhod procesa \underline{y} z linearno kombinacijo regresorjev \underline{x}_i ($\underline{X} = [\underline{x}_1 \underline{x}_2 \cdots \underline{x}_n]$). Izhod \underline{y} je lahko predstavljen kot točka v N -dimenzionalnem prostoru, in regresorji \underline{x}_i so vektorji v tem prostoru, ki jih moramo ustrezno sestaviti, da se približamo \underline{y} kolikor je mogoče; glej sliko 4.2. V splošnem bi zahtevali $n = N$ (to je en regresor za vsako meritev) linearne neodvisnih regresorjev, da bi dosegli natančno \underline{y} . Tak interpolacijski primer, kjer je število parametrov enako številu meritev ni zaželen iz več razlogov; glej poglavje 8. Običajno je cilj modeliranja v iskanju skupine regresorjev, ki aproksimira \underline{y} do določene natančnosti s čim manjšim številom regresorjev. To pomeni, da moramo iz množice vseh spremenljivk (možnih regresorjev) izbrati najvplivnejše. Enak problem nastopi tudi pri iskanju reda procesa, oziroma števila zakasnitev v primeru linearnih dinamičnih procesov in reda polinoma ali baznih funkcij pri modeliranju nelinearnih procesov. Zaradi tega so metode izbire regresorjev zelo pomembne. Omejili se bomo na metode, kjer so modeli linearni v parametrih.

4.3.1 Metode za izbiro regresorjev

V nadaljevanju si bomo ogledali metode za izbiro n_s najvplivnejših regresorjev iz množice vseh n možnih regresorjev. Najenostavnejša rešitev bi bila v preizkusu vseh možnih kombinacij regresorjev, ki pa jih je $2^n - 1$. To število je odločno preveliko za praktično izvedbo. Zato se uporablja predvsem naslednje tri pristope:

- izbira z dodajanjem,
- vzvratna izbira,
- postopna izbira.

Najpogostejsi pristop k izberi regresorjev je t.i. pristop z dodajanjem (*ang. forward selection*). Najprej za vsakega posameznega od n možnih regresorjev izračunamo optimalne parametre, potem pa izberemo tistega, ki da najboljšo možno aproksimacijo izhoda \underline{y} . Ta regresor in pripadajoči parameter označimo z \underline{x}_A in $\hat{\theta}_A$. Nato pa del izhoda \underline{y} , ki ga ne uspemo aproksimirati zapišemo kot $\underline{y}_A = \underline{y} - \underline{x}_A \hat{\theta}_A$. V nadaljevanje pa enako ponovimo za vseh ostalih neizbranih $n - 1$ regresorjev na ostanku \underline{y}_A . Vse skupaj ponovimo z uporabo optimizacije. Drugi izbrani regresor zapišemo kot \underline{x}_B in pripadajoči parameter kot $\hat{\theta}_B$. Nato izračunamo ostanek izhoda $\underline{y}_B = \underline{y}_A - \underline{x}_B \hat{\theta}_B$, ki ga moramo aproksimirati z ostalimi regresorji. To proceduro ponavljamo dokler ne dobimo n_s regresorjev. Procedura je relativno hitra, ker $n - i + 1$ krat z optimizacijo določimo po en parameter. Glavna pomanjkljivost pa je v tem, da ne upoštevamo interakcije med regresorji. Parametri izbranih regresorjev so izbrani v vsaki sekvenci posamezno, medtem, ko bi za pravilno rešitev morali optimirati vse parametre sočasno; glej sliko 4.7 in 4.10. Samo, če so vsi regresorji ortogonalni, potem ni nobene interakcije med njimi in bi stvar rezultirala v dobrem rezultatu. Ker pa ortogonalnost ni pričakovana (celo ne aproksimativno gledano), običajno

tak postopek vodi v slabe rezultate. Rešitev tega problema je v ocenjevanju vseh parametrov sočasno. To pa vodi k ocenjevanju n parametrov v prvi fazi, $n - 1$ parametrov v drugi fazi, in tako naprej. Računska zahtevnost je v primeru velikega števila regresorjev precejšna.

Najpreprostejša rešitev bi torej bila, da regresorje transformiramo tako, da bodo ortogonalni in na ta način medsebojno neodvisni. Ta način se imenuje pristop z dodajanjem in *ortogonalizacijo*. Ker pa je običajno število regresorjev n veliko in je število izbranih regresorjev n_s majhno, bi bilo neučinkovito ortogonalizirati celotno matriko \underline{X} . Učinkovita procedura je takšna, da poiščemo najprej najvplivnejši regresor in potem ostalih $n - 1$ neizbranih regresorjev ortogonaliziramo glede na izbranega. V drugem koraku je izbran med ostalimi $n - 1$ regresorji tisti, ki najbolje opisuje izhodni signal in potem ostalih $n - 2$ neizbranih regresorjev spet ortogonaliziramo glede na izbranega, in tako naprej. Ker so regresorji ortogonalni je izbira parametrov, ki jim pripadajo enostavna, glej poglavje 4.1.

Primer 4.3.1. Ilustracija metode z dodajanjem regresorjev

V tem enostavnem primeru bomo ilustrirali izbiro strukture. Dani so $N = 3$ vzorci podatkov in trostolpna matrika regresorjev \underline{X}

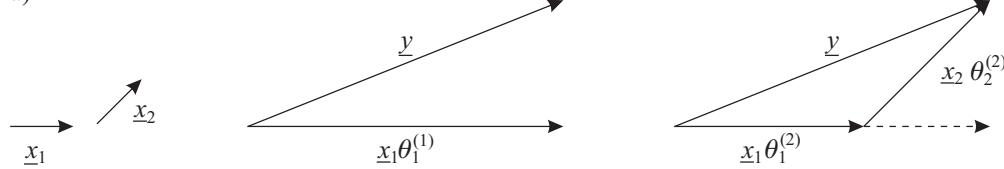
$$\begin{bmatrix} y(1) \\ y(2) \\ y(3) \end{bmatrix} = \begin{bmatrix} x_1(1) & x_2(1) & x_3(1) \\ x_1(2) & x_2(2) & x_3(2) \\ x_1(3) & x_2(3) & x_3(3) \end{bmatrix} \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} + \begin{bmatrix} e(1) \\ e(2) \\ e(3) \end{bmatrix}. \quad (4.72)$$

Ker je izhodni vektor \underline{y} popolnoma dosegljiv s tremi neodvisnimi regresorji v matriki \underline{X} je napaka e enaka 0 . Za \underline{y} in \underline{X} potem dobimo naslednje vrednosti

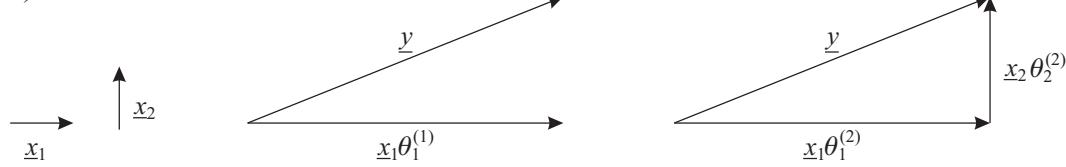
$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}. \quad (4.73)$$

Slika 4.11 prikazuje vektor izhodnih vrednosti \underline{y} . V nadaljevanju moramo izbrati enega od regresorjev iz matrike \underline{X} in optimirati ustrezeni parameter θ_i v smislu najboljše možne aproksimacije izhodnega vektorja \underline{y} . Kot kaže slika 4.11b je 3. regresor s parametrom $\theta_3 = 3$ najpomembnejši za aproksimacijo izhodnega vektorja \underline{y} . Drugi najpomembnejši regresor je dan v stolcu

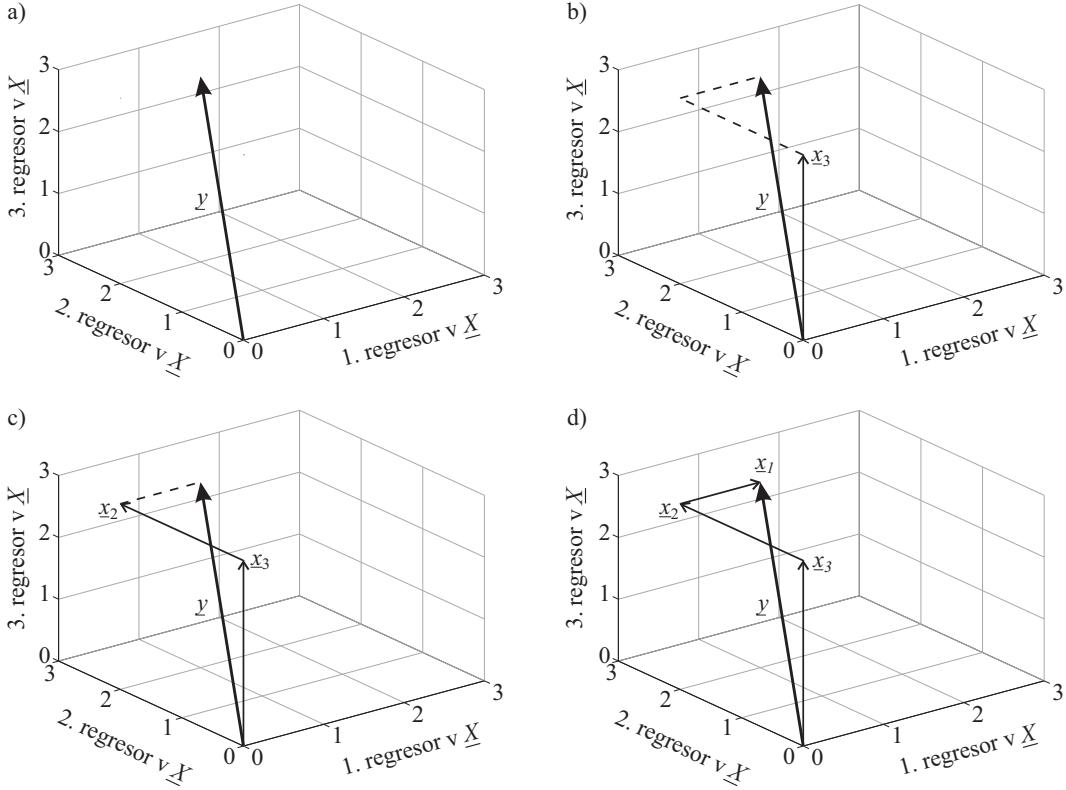
a)



b)



Slika 4.10. Neortogonalni in ortogonalni regresorji x_1 in x_2 : a) če sta regresorja neortogonalna, so optimalni parametri odvisni od korelacije med regresorjem. To pomeni, da moramo ob izbiri novega regresorja, na novo optimirati vse parametre. b) če so regresorji ortogonalni, potem so optimalni parametri neodvisni drug od drugega in lahko pasamezne parametre izračunamo neodvisno.



Slika 4.11. lustracija izbire regresorjev za $N = 3$ vzorce in tri ortogonalne regresorje.

2 in s parametrom $\theta_2 = 2$; glej sliko 4.11c. Nazadnje je regresor 1 s parametrom $\theta_1 = 1$; glej sliko 4.11d. Analiza je enostavna, ker so vektorji ortogonalni eden na drugega in je možno vpliv posameznega regresorja določiti ločno od drugih in seveda z njim povezan parameter. Praktično pa regresorji nikoli niso ortogonalni in je potrebno izvesti ortogonalizacijo stolpcov v matriki \underline{X} .

Alternativa metodi z dodajanjem regresorjev je vzvratna metoda (*ang. backward elimination*), ki temelji na odvzemanju regresorjev. V tem primeru algoritem začnemo z vsemi n regresorji in v vsakem koraku odvzamemo najmanj pomemben regresor. To je uporabno takrat, ko pričakujemo, da velja $n_s \approx n$. Seveda ni smiselno v primeru, kjer je $n_s \ll n$.

Tretja možnost pa je postopna izbira regresorjev (*ang. stepwise selection*). V vsaki iteraciji, predno izberemo novi regresor, vse že izbrane regresorje preverimo z uporabo določeni statističnih testov in odstranimo nepomembne regresorje. Tak način je precej bolj napreden, kot prejšnja dva. V literaturi [29] jo opisujejo kot zelo uspešno.

4.3.2 Ortogonalni najmanjši kvadrati (OLS) pri izbiri z dodajanjem

Naslednja razlaga za metodo ortogonalnih najmanjših kvadratov (OLS) pri izbiri regresorjev z dodajanjem je privzeta po [15, 16]. Predpostavimo naslednje

$$\underline{y} = \underline{X} \underline{\theta} + \underline{\epsilon}. \quad (4.74)$$

Metoda OLS vpelje transformacijo regresorjev \underline{x}_i v set ortogonalnih vektorjev in to omogoča izračun prispevka posameznega ortogonalnega vektorja k izhodnemu vektorju. Regresijsko matriko \underline{X} lahko zapišemo kot

$$\underline{X} = \underline{V} \underline{R}$$

kjer je \underline{R} matrika dimenzijs $n \times n$ in je trikotne oblike z naslednjo strukturo

$$\underline{R} = \begin{bmatrix} 1 & r_{12} & r_{13} & \cdots & r_{1n} \\ 0 & 1 & r_{23} & \cdots & r_{2n} \\ 0 & 0 & \ddots & & \vdots \\ \vdots & & & & r_{n-1n} \\ 0 & \cdots & 0 & 1 & \end{bmatrix} \quad (4.75)$$

in je \underline{V} matrika dimenzijs $N \times n$ z ortogonalnimi stolpcji \underline{v}_i ($\underline{V} = [\underline{v}_1 \ \underline{v}_2 \ \cdots \ \underline{v}_n]$) tako da je $\underline{V}^T \underline{V} = \underline{I}$ in je \underline{S} diagonalna z $s_i = \underline{v}_i^T \underline{v}_i$.

Prostor, ki ga definirajo ortogonalni bazni vektorji \underline{v}_i je enak kot prostor, ki ga definirajo regresorji \underline{x}_i . Zato enačbo 4.74 lahko zapišemo kot

$$\underline{y} = \underline{V} \underline{\vartheta} + \underline{\varepsilon}, \quad (4.76)$$

kjer je $\underline{\vartheta}$ vektor transformiranih parametrov in je v naslednji relaciji z vektorjem originalnih parametrov $\underline{\theta}$:

$$\underline{R} \underline{\theta} = \underline{\vartheta}. \quad (4.77)$$

Rešitev iz enačbe 4.76 zapišemo kot

$$\underline{\vartheta} = (\underline{V}^T \underline{V})^{-1} \underline{V}^T \underline{y} = \underline{S}^{-1} \underline{V}^T \underline{y}. \quad (4.78)$$

4.3.3 Regularizacija proti izbiri regresorjev

V primeru 4.1.6 smo pokazali, da je regularizacija lahko uporabna pri izbiri najpomembnejših regresorjev. Najprej izvedemo regularizacijo na celotnem naboru regresorjev pri različnih parametrih α . V nadaljevanju izločimo vse regresorje, ki imajo parametre manjše od izbranega praga. Regresorji morajo biti v tem primeru normirani. Po eliminaciji nevplivnih regresorjev izvedemo klasično LS metodo z regresorji, ki so ostali.

V mnogih primerih je uporaba regularizacije za izbiro regresorjev boljša kot OLS metoda. Razlog za to je v upoštevanju vseh regresorjev simultano. V nasprotju z OLS, ki izbira regresorje iterativno in ne odstrani regresorjev, ki postanejo relativno nepomembni. Slabost OLS je v iterativnem dodajanju regresorjev. Na drugi strani pa regresijska metoda lahko postane neizvedljiva zaradi velikih dimenzij, ker je matrična inverzija je $\mathcal{O}(n^3)$ operacija.

4.4 Povzetek

Linearne optimizacijske metode so dobro raziskane in ustaljene metode. Kadar imamo problem modeliranja je najprej potrebno poiskati linearni model in izkoristiti naslednje prednosti:

- globalni optimum lahko najdemo analitično v enem koraku,
- varianca ocjenjenih parametrov in intervali zaupanja so enostavno izračunljivi,
- robustna in hitra rekurzivna formulacija je eden od načinov izvedbe algoritma,
- regularizacija in izbira regresorjev sta metodi, ki omogočata učinkovito izbiro strukture modela.

5. Lokalna nelinearna optimizacija

Če je gradient kriterijske funkcije $I(\underline{\theta})$ nelinearen v parametrih $\underline{\theta}$ moramo pri iskanju optimalnih parametrov $\underline{\theta}_{opt}$ uporabiti nelinearno optimizacijsko tehniko. Parametre v takem primeru imenujemo *nelinearni parametri*. Nelinearne parametre srečamo pogosto. Nelinearni parametri so uteži v skritih plasteh nevronskih mrež, položaji in širine pripadnostnih funkcij v mehkih sistemih. Celo v kontekstu linearnih sistemov lahko naletimo na probleme nelinearnih parametrov, ki so povezani z minimizacijo izhodnega pogreška modela; glej poglavje 14.5.3. Pomembno je razumeti principe posameznih osnovnih konceptov nelinearne lokalne optimizacije, da se lahko odločimo katero bomo v določenem primeru uporabili. Večina metod je namreč dostopna v običajnih optimizacijskih knjižnicah [12].

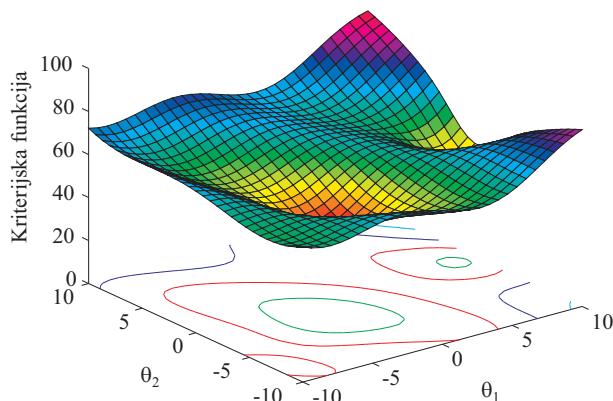
Nelinarni optimizacijski problemi imajo navadno naslednje lastnosti:

- obstaja več lokalnih optimumov (glej sliko 5.1),
- hiper površino v okolini lokalnega optima lahko aproksimiramo (uporaba Taylorjeve vrste drugega reda) s hiperparabolo v obliki $\underline{\theta}^T \underline{A} \underline{\theta} + \underline{b}^T \underline{\theta} + c$,
- analitične rešitve ne obstajajo,
- rešitev je mogoče poiskati samo iterativno,
- sprotna optimizacija je praktično neizvedljiva.

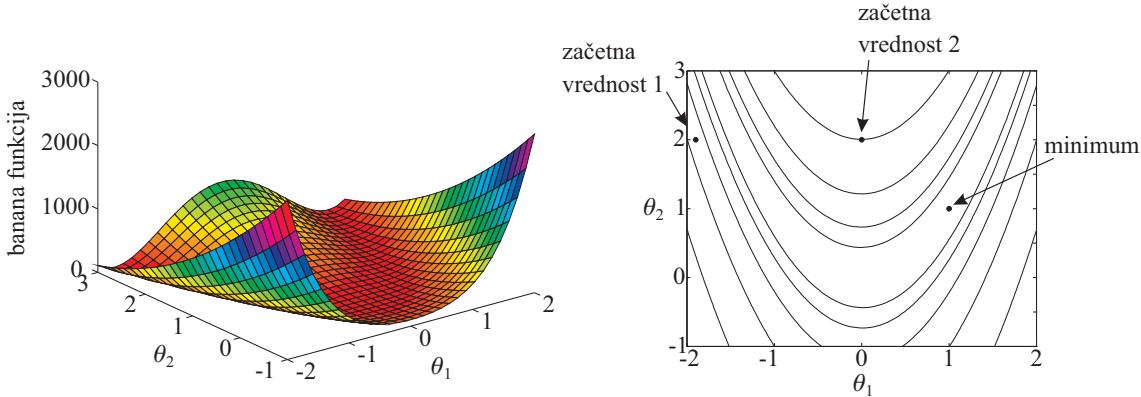
V tem poglavju si bom na kratko pogledali metode lokalne nelinearne optimizacije. Pri tem mislimo na optimizacijo, kjer izberemo začetno točko iskanja in preiskujemo samo okolico te točke. Običajno na ta način najdemo najbližji lokalni optum. Preiskovanje širšega prostora problema pa lahko razširimo tako, da zaženemo algoritom iz različnih začetnih točk (*ang. multi-start technique*) in na koncu izberemo najboljšo rešitev.

Primer 5.0.1. Rosenbrockova funkcija

Na primeru Rosenbrockove funkcije bomo pokazali najosnovnejše metode lokalne nelinearne



Slika 5.1. Kriterijska funkcija za nelinearni optimizacijski problem z večratnim lokalnim minimumom.



Slika 5.2. a) Rosenbrockova funkcija. b) Konturni diagram Rosenbrockove funkcije z minimumom pri $[1.0 \ 1.0]^T$ in dvema začetnima točkama $[-1.9 \ 2.0]^T$ in $[0.0 \ 2.0]^T$, ki jih uporabimo pri optimizaciji.

optimizacije. Funkcija $I(\underline{\theta})$ je prikazna na sliki 5.2 in je definirana z dvema parametromi θ_1 in θ_2

$$I(\underline{\theta}) = 100(\theta_2 - \theta_1^2)^2 + (1 - \theta_1)^2. \quad (5.1)$$

Funkcija običajno služi kot testni primer, saj pri določeni izberi konstant učenja postane nestabilna in je hkrati zelo slabo konvergentna. Funkcija ima minimum $I(\underline{\theta}_{opt}) = 0$ pri vrednosti optimizacijskih parametrov $\underline{\theta}_{opt} = [1.0 \ 1.0]^T$. Na sliki 5.2b je prikazan njen konturni diagram in dve začetni vrednosti parametrov pri $[-1.9 \ 2.0]^T$ in $[0.0 \ 2.0]^T$, ki jih uporabimo za preizkus optimizacijskih metod.

Cilj optimizacije je poiskati minimum funkcije z natančnostjo parametrov vsaj 0.001. Število iteracij in evaluacij funkcije v primeru obeh začetnih pogojev je prikazano za vse obravnavane optimizacijske metode. Za prvo začetno vrednost smo prikazali tudi konvergenčni diagram in pot od začetnega pogoja do končne konvergenčne točke. Primer osvetljuje lastnosti, slabosti in prednosti posameznih optimizacijskih metod. Kakšnih posebnih *splošno veljavnih* ugotovitev pa na osnovi ene sam posebne funkcije ne moremo dati.

Pri vseh gradientnih metodah potrebujemo gradient kriterijske funkcije, ki jo optimiramo. V našem primeru je to

$$\underline{g}(\underline{\theta}) = \begin{bmatrix} \frac{\partial I(\underline{\theta})}{\partial \theta_1} \\ \frac{\partial I(\underline{\theta})}{\partial \theta_2} \end{bmatrix} = \begin{bmatrix} 400\theta_1(\theta_1^2 - \theta_2) + 2(\theta_1 - 1) \\ 200(\theta_2 - \theta_1^2) \end{bmatrix} \quad (5.2)$$

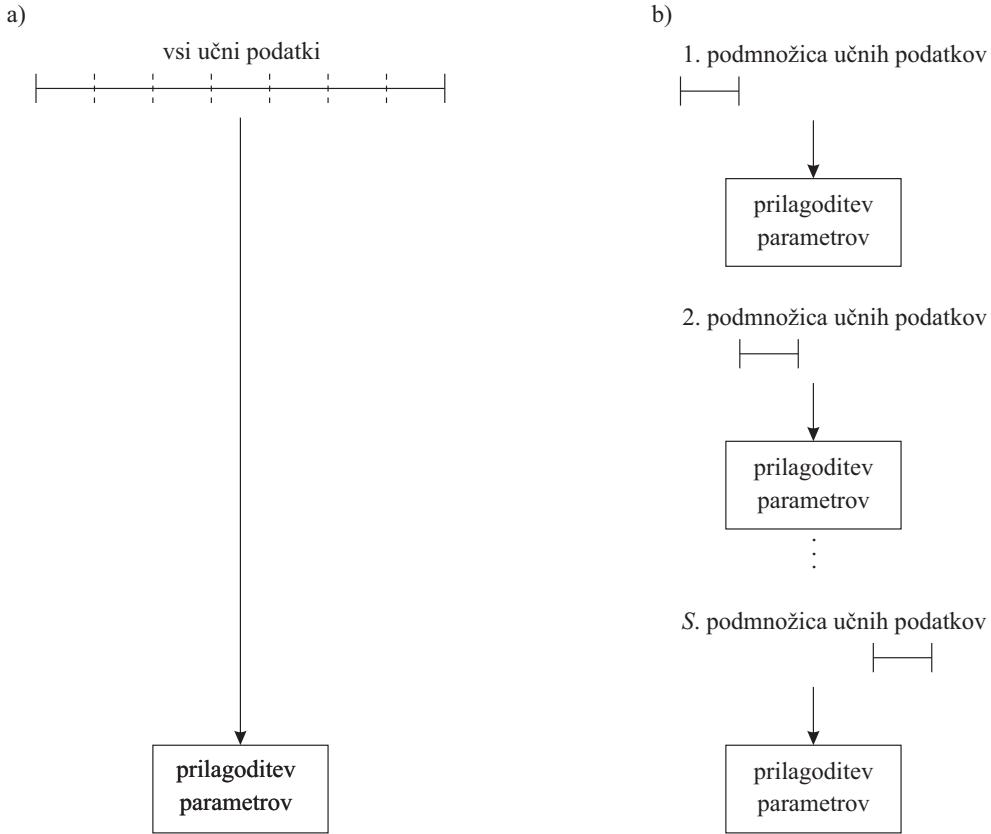
in Hessovo matriko (glej Dodatek A), ki jo potrebujemo v primeru Newtonove metode

$$\underline{H}(\underline{\theta}) = \begin{bmatrix} \frac{\partial^2 I(\underline{\theta})}{\partial \theta_1^2} & \frac{\partial^2 I(\underline{\theta})}{\partial \theta_1 \partial \theta_2} \\ \frac{\partial^2 I(\underline{\theta})}{\partial \theta_2 \partial \theta_1} & \frac{\partial^2 I(\underline{\theta})}{\partial \theta_2^2} \end{bmatrix} = \begin{bmatrix} 1200\theta_1^2 - 400\theta_2 + 2 & -400\theta_1 \\ -400\theta_1 & 200 \end{bmatrix}. \quad (5.3)$$

5.1 Optimizacija z nizom podatkov in po posameznih podatkih

V vsaki iteraciji k izračunamo novo vrednost parametrov glede na prejšnje vrednosti parametrov in prejšnjo vrednost kriterijske funkcije

$$\underline{\theta}_k = f \left(\underline{\theta}_j, I(\underline{\theta}_j), \frac{\partial}{\partial \underline{\theta}_j} I(\underline{\theta}_j), \dots \right) \quad \text{pri } j = k-1, k-2, \dots, 0. \quad (5.4)$$



Slika 5.3. a) Pri optimizaciji z nizom podatkov se uporabijo vsi podatki za eno adaptacijo parametrov. b) Za velike baze podatkov je smiselno parametre optimirati S krat na učnih množicah, ki so $1/S$ del celotne učne množice. Če je S enak številu podatkov N , potem taki optimizaciji rečemo optimizacija po posameznih podatkih.

Običajno pri optimizaciji, ki jo opisuje enačba 5.4, izvedemo samo prejšnji korak pri $j = k - 1$, ki upošteva, da izračun $\underline{\theta}_{k-1}$ že vključuje informacijo o preteklih vrednostij ($j < k - 1$)

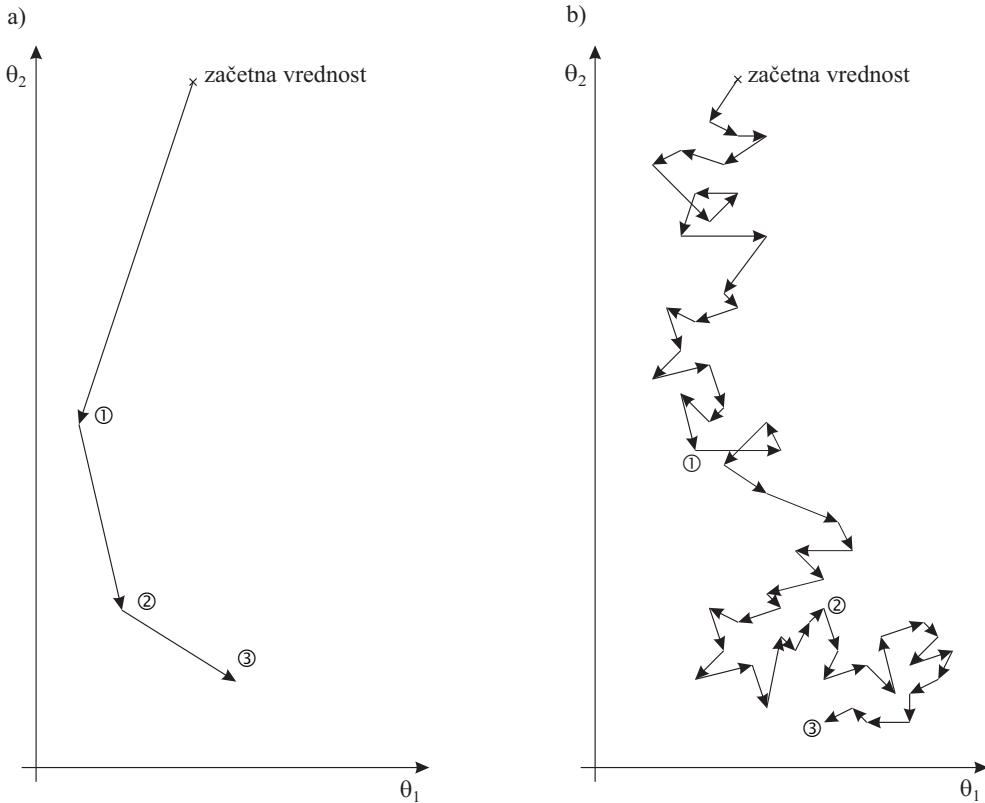
$$\underline{\theta}_k = f \left(\underline{\theta}_{k-1}, I(\underline{\theta}_{k-1}), \frac{\partial}{\partial \underline{\theta}_{k-1}} I(\underline{\theta}_{k-1}), \dots \right). \quad (5.5)$$

Direktna posledica enačbe 5.5 je v tem, da moramo za nov izračun parametrov izračunati kriterijsko funkcijo in po možnosti tudi njene odvode. Ta pristop imenujemo pristop z nizom podatkov (ang. *batch adaptation*), ker moramo pri vsaki adaptaciji parametrov uporabiti vse podatke iz učne množice. To pomeni veliko računske obremenitev. Rešitev za tak primer je razdelitev učne množice na S podmnožic in uspesivna uporaba enačbe 5.5 na vseh učnih podmnožicah; glej sliko 5.3. Prednost je v tem, da izračunamo S krat več parametrov (za enako učno množico). Pomanjkljivost pa je, da imamo S različnih optimizacij in vsako s $1/S$ celotne učne množice. Kvaliteta ocene tako pada s povečevanjem števila S .

Skrajna realizacija je primer, ko izberemo $S = N$ podmnožic (pri N podatkih v učni množici), to pomeni, da je v vsaki podmnožici po en podatek. Potem se iteracija v enačbi 5.5 poenostavi (predpostavimo kvadratično kriterijsko funkcijo)

$$\underline{\theta}_k = f \left(\underline{\theta}_{k-1}, e^2(\underline{\theta}_{k-1}), \frac{\partial}{\partial \underline{\theta}_{k-1}} e^2(\underline{\theta}_{k-1}), \dots \right). \quad (5.6)$$

To metodo imenujemo *optimizacija po posameznih podatkih* ali *trenutno učenje*.

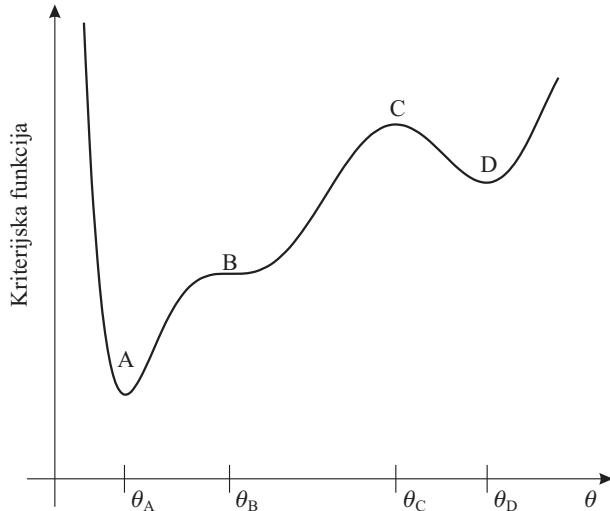


Slika 5.4. Shematični prikaz treh iteracij v primeru nelinearne optimizacije a) na nizu podatkov in b) na posameznih podatkih. Na nizu podatkov naredimo eno večjo adaptacijo parametrov na vsako iteracijo, v primeru adaptacije na posameznih vzorceh pa parametre adaptiramo pri vsakem novem podatku. Skupna povprečna smer je podobna optimizaciji na nizu podatkov. Oba postopka lahko konvergirata v različne rešitve pri zelo različnih časih.

Obstaja tudi razlika med **iterativnim algoritmom** in **rekurzivnim optimizacijskim algoritmom**. Rekurzivni algoritmi se lahko uporabijo v sočasnem načinu delovanja (*ang. online*) in predstavljajo natančno (eksaktно) rešitev problema. So samo prilagojeni sekvenčnemu prihajanju vzorcev podatkov, t.i. toku podatkov (*ang. data stream*). Rekurzivni algoritem z večkratnimi teki skozi isto podatkovno bazo ne pridobi nobene informacije. V primeru iterativnega algoritma pa je večkratni tek skozi podatke zahtevan in samo na ta način lahko zagotovimo konvergenco. Zato je tak način tudi ustrezen počasnejši. Čeprav je tudi iterativni algoritem možno zaganjati v načinu posameznih podatkov in sočasnem principu, običajno daje slabše rezultate, počasi konvergira in je nerobusten na dimenzije podatkov.

Optimizacija po posameznih podatkih izvira iz metode vzvratnega širjenja napake (*ang. backpropagation*) za učenje večplastnih nevronskih mrež. Optimizacija na nizu podatkov je klasičen pristop in je osnova za vse bolj pomembne algoritme. Največji problem pri učenju na posameznih podatkih je v tem, da je adaptacija izvedena na osnovi enega samega vzorca, kar je vprašljiva rešitev. Adaptacija parametrov na osnovi enega podatka se se s prehodom skozi celotno učno množico povpreči.

Adaptacijo na nizu podatkov in adaptacijo na posameznih podatkih lahko intuitivno vidimo na naslednji način: adaptacija na nizu podatkov zbere veliko informacije (N vzorcev) za adaptacijo parametre v naslednjem koraku. Pri adaptaciji na posameznih podatkih pa izvedemo adaptacijo po vsakem pridobljenem vzorcu. Metoda nikoli ne konvergira v točko prostora parametrov. Na sliki 5.4 sta predstavljena oba pristopa.



Slika 5.5. Kriterijska funkcija za nelinearni optimizacijski problem z enim parametrom.

5.2 Inicializacija parametrov

Ker so nelinearne optimizacijske metode iterativne, potrebujemo začetne vrednosti parametrov $\underline{\theta}_0$ v prvi iteraciji $k = 1$. Dobra izbira začetnih parametrov $\underline{\theta}_0$ vodi k hitri konvergenci k optimalnim parametrom $\underline{\theta}_{opt}$. Če so parametri fizikalno razložljivi ali drugače interpretabilni, potem lahko običajno smiselno izberemo začetne vrednosti vektorja parametrov $\underline{\theta}_0$. Na vsak način pa lahko s poznavanjem fizikalnega problema izberemo vsaj določen interval, kjer se optimalna rešitev $\underline{\theta}_{opt}$ nahaja. Nasproten problem srečamo pri identifikaciji sistemov z nevronskimi mrežami, kjer parametri, ki jih optimiramo nimajo nikakršne fizikalne interpretacije.

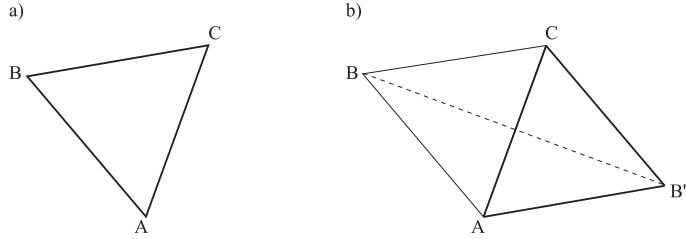
Slika 5.5 prikazuje primer kriterijske funkcije enega parametra θ . V tem primeru dobimo rešitev, ki konvergira k globalnemu minimumu pri θ_A samo v primeru začetnega parametra, ki je manjši od θ_C . V primeru inicializacije pri vrednosti $\theta \geq \theta_C$ pa optimizacija konvergira k lokalnemu minimumu pri θ_D . Prevojne točke, kot je to v primeru točka B, običajno ne povzročajo težav pri konvergenci. Algoritem praktično nikoli ne konvergira k rešitvi v prevojni točki θ_B .

5.3 Algoritmi direktnega iskanja optimuma

Algoritmi direktnega iskanja optimuma temeljijo samo na izračunavanju kriterijske funkcije v $n + 1$ ekvidistančnih točkah parameterskega prostora, kjer je n število parametrov. Točko maksimalne vrednosti kriterijske funkcije preslikamo preko lika, ki ga tvori $n + 1$ točk in določimo nov simpleks. To ponavljamo dokler rezultat ne konvergira v določeno okolico. Metoda je direktna, ker ne potrebujemo odvodov. Prednosti direktnih metod so v enostavnosti in razumljivosti. Metodi, ki sodita v ta razred sta metodi simpleksov in Hooke-Jeevesova metoda, ki sta bolj natančno razloženi v [98].

5.4 Gradientne metode optimizacije

Gradientne metode optimizacije so najbolj običajne in najpomembnejše metode nelinearne lokalne optimizacije. V nadaljevanju predpostavimo, da je gradient $\underline{g} = \partial I(\underline{\theta}) / \partial \underline{\theta}$ (odvod kriterijske funkcije $I(\underline{\theta})$ glede na parametre $\underline{\theta}$) analitično izračunljiv ali pa ga lahko aproksimiramo s



Slika 5.6. a) Simpleks (ekvidistančni trikotnik) v tridimenzionalnem parameterskem prostoru. b) Ogljišče z največjo kriterijsko funkcijo (v tem primeru B) se prezrcali (v tem primeru v B') za kreiranje novega simpleksa.

končnimi diferencami; glej poglavje 5.4.1. Princip vseh gradientnih metod je v spremembni vektorja parametrov $\underline{\theta}_{k-1}$ proporcionalno glede na korak adaptacije η_{k-1} v smeri \underline{p}_{k-1} , ki je smer gradienta \underline{g}_{k-1} , ki jo zavrtimo in skaliramo z matriko \underline{R}_{k-1}

$$\underline{\theta}_k = \underline{\theta}_{k-1} - \eta_{k-1} \underline{p}_{k-1} \quad \text{pri } \underline{p}_{k-1} = \underline{R}_{k-1} \underline{g}_{k-1}. \quad (5.7)$$

Cilj optimizacije je v zmanjševanju kriterijske funkcije v vsaki iteraciji, to pomeni, da je $I(\underline{\theta}_k) < I(\underline{\theta}_{k-1})$. Najenostavnejša izbira za rotacijsko matriko je $\underline{R}_{k-1} = \underline{I}$, ki privede do metode največjega spusta.

Obstoječi algoritmi se razlikujejo po matriki \underline{R} in koraku optimizacije η . Za izračun vsake iteracije iz enačbe 5.7 moramo izbrati korak η . Nekateri algoritmi delujejo s konstantnim korakom, vendar je metoda bistveno učinkovitejša, če izberemo optimalni korak η_{k-1} , ki optimira kriterijsko funkcijo v smeri \underline{p}_{k-1} .

5.4.1 Metoda končnih differenc

Za vse nadaljnje metode, ki jih bomo obravnavali, potrebujemo gradient, t.j., prvi odvod kriterijske funkcije za izračun smeri iskanja \underline{p} . Za mnoge primere lahko analitično izračunamo gradient, tako kot tudi v primeru Rosenbrockove funkcije.

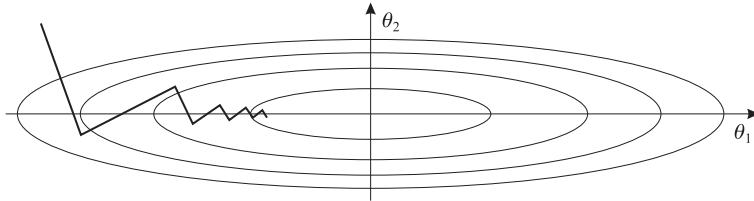
Za izračun gradienta kriterijske funkcije v primeru vsote kvadratov napake je potrebno izračunati tudi prvi odvod izhoda modela \hat{y} glede na parametre $\underline{\theta}$

$$\begin{aligned} \underline{g} &= \frac{\partial I(\underline{\theta})}{\partial \underline{\theta}} = \frac{\partial \frac{1}{2} \sum_{i=1}^N e^2(i)}{\partial \underline{\theta}} = \frac{1}{2} \sum_{i=1}^N \frac{\partial e^2(i)}{\partial \underline{\theta}} = \sum_{i=1}^N e(i) \frac{\partial e(i)}{\partial \underline{\theta}} \\ &= - \sum_{i=1}^N e(i) \frac{\partial \hat{y}(i)}{\partial \underline{\theta}}, \end{aligned} \quad (5.8)$$

kjer so napake $e(i) = y(i) - \hat{y}(i)$ razlike med izmerjenim izhodom $y(i)$ in izhodom modela $\hat{y}(i)$, ki je funkcija parametrov $\underline{\theta}$, N pa je število vzorcev iz učne množice.

V določenih primerih pa je nemogoče analitično izraziti gradient, ali pa je izračunavanje prezahtevno. Takrat si pomagamo z *metodo končnih differenc*, ki jo lahko uporabimo za numerični izračun gradienta.

Komponento gradienta i pri vrednosti parametra $\underline{\theta}$ lahko aproksimiramo z diferenčnim kvocienjem



Slika 5.7. “Cik-cak obnašanje”: Značilno obnašanje metode maksimalnega gradienta za kriterijsko funkcijo, kjer je Hessova matrika zelo različnimi lastnimi vrednostmi.

$$g_i(\underline{\theta}) \approx \frac{I(\underline{\theta} + \Delta\theta_i) - I(\underline{\theta})}{\Delta\theta_i}, \quad (5.9)$$

kjer je $\Delta\theta_i$ majhna spremembra v smeri i te komponente. To pomeni, da moramo za aproksimacijo celotnega gradienata \underline{g} izračunati po enačbi 5.9 pri vrednosti $\underline{\theta}$ za $i = 1, 2, \dots, n$, za vse smeri prostora problema. Posledično je računska zahtevnost določena z n izračuni kriterijske funkcije pri vrednostih $I(\underline{\theta} + \Delta\theta_i)$, kjer je n število parametrov. Na ta način postane izračun gradienata dominanten računski korak pri optimizaciji.

Seveda lahko aproksimiramo tudi Hessovo z uporabo končnih differenc. Stolpec pri i ti komponenti matrike \underline{H} , ki ga označimo kot \underline{h}_i lahko izračunamo

$$\underline{h}_i(\underline{\theta}) \approx \frac{\underline{g}(\underline{\theta} + \Delta\theta_i) - \underline{g}(\underline{\theta})}{\Delta\theta_i}. \quad (5.10)$$

Praktično je stvar izvedljiva samo, če je gradienat podan analitično, drugače je računska zahtevnost prevelika.

Praktični problem je tudi izbira koraka $\Delta\theta_i$. Na eni strani mora biti poljubno majhen, na drugi strani pa zadosti velik, da ne dosežemo prevelikega kvantiziranja prostora parametrov [105].

5.4.2 Metoda največjega gradienata

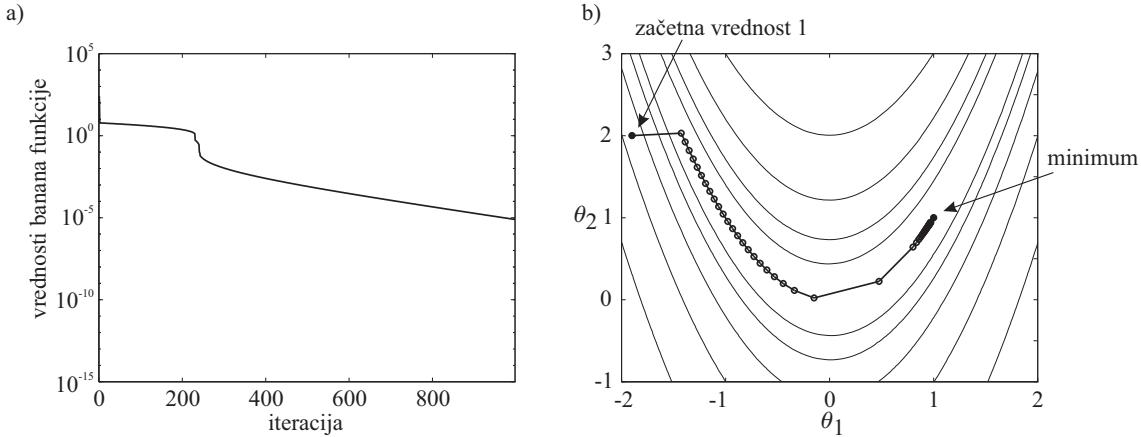
Metoda največjega spusta je najpreprostejša izvedenka gradientnih metod iz enačbe 5.7, kjer je rotacijska matrika \underline{R} enaka enotski \underline{I} :

$$\underline{\theta}_k = \underline{\theta}_{k-1} - \eta_{k-1} \underline{g}_{k-1}. \quad (5.11)$$

To pomeni, da je smer iskanja optimuma nasprotna od smeri gradienata. To zagotavlja zmanjševanje kriterijske funkcije v vsaki iteraciji. Strategija daje dobre rezultate v začetnih iteracijah, ko pa se približujemo minimumu, nastane težava, če imamo primer, ki ima slabo pogojeno Hessovo matriko (vrednost χ), to je zelo različne vrednosti lastnih vrednosti. Slika 5.7 kaže tipično obnašanje te metode v takem primeru, tako imenovano “cik-cak” obnašanje. To obnašanje je pogojeno z ortogonalnostjo zaporednih smeri pri tej metodi, ker v vsaki iteraciji najdemo gradienat v nasprotni smeri. Metoda omogoča dobro konvergenco samo za parametre z dominantnimi lastnimi vrednostmi Hessove matrike. Celo v primeru linearnih problemov je potrebno za konvergenco neskončno število iteracij.

Primer 5.4.1. Metoda največjega gradienata za minimizacijo funkcije

V tem primeru smo Rosenbrockovo funkcijo minimizirali s tehniko največjega gradienata. Slika 5.8 kaže hitrost konvergence. Krogci na sliki zaradi slabe konvergencije kažejo vsako deseto iteracijo. Običajno je tudi res, da metoda največjega gradienata potrebuje bistveno več iteracij kot kakšna od metod, ki izračunavajo tudi drugi odvod.



Slika 5.8. Metoda največjega gradijenta za minimizacijo Rosenbrockove funkcije. Hitrost konvergencije a) glede na iteracije, b) v parameterskem prostoru (krogci prikazujejo vsako deseto iteracijo).

5.4.3 Newtonova metoda

Pri Newtonovi metodi je rotacijska matrika \underline{R} v enačbi 5.7 izbrana kot vrednost inverza Hessove matrike \underline{H}_{k-1}^{-1} pri vektorju parametrov $\underline{\theta}_{k-1}$

$$\underline{\theta}_k = \underline{\theta}_{k-1} - \eta_{k-1} \underline{H}_{k-1}^{-1} \underline{g}_{k-1}. \quad (5.12)$$

Hessovo matriko izračunamo analitično ali pa ocenimo s končnimi diferencami. V klasični Newtonovi metodi je korak η enak 1. V primeru linearnega optimizacijskega problema nas ta izbira privede do optimuma v eni iteraciji. To sledi direktno iz razvoja kriterijske funkcije v Taylorjevo vrsto. Za nelinearne probleme optimuma ne moremo doseči z enim samim korakom.

Problem pri Newtonovi metodi je v tem, da (enačba 5.12) zmanjšuje kriterijsko funkcijo (gre navzdol po strmini) samo za pozitivne vrednosti Hessove matrike \underline{H}_{k-1} . To je vedno res v okolici optimuma, vendar pa ni nujno v okolici začetne točke $\underline{\theta}_0$ in s tem pri prvi iteraciji. Zaradi tega večkrat uporabimo modificirano metodo, kjer je Hessova matrika zamenjana z matriko $\bar{\underline{H}}_{k-1}$, ki je pozitivna vendar blizu \underline{H}_{k-1} .

Pomanjkljivost Newtonove metode je v zahtevi po izračunu drugega odvoda kriterijske funkcije. To zmanjšuje uporabnost te metode na probleme majhnih dimenzij.

Izpeljava Newtonove metode za enodimensionalni problem. Newtonovo metodo bomo najprej osvetlili na primeru funkcije ene spremenljivke $y = f(x)$, kjer jo pogosto uporabimo za izračun ničel funkcije.

Funkcijo $f(x)$ lahko razvijemo pa Taylorjevi vrsti na nasleden način

$$f(x_k) = f(x_{k-1}) + f'(x_{k-1})(x_k - x_{k-1}) + \frac{1}{2!} f''(x_{k-1})(x_k - x_{k-1})^2 + \delta, \quad (5.13)$$

kjer je v δ združen ostanek od člena druge vrste do neskončnosti. Če predpostavimo, da izračunavamo novo vrednost funkcije $f(x_k)$ v bližnji okolici že izračunane vrednosti $f(x_{k-1})$, potem lahko ostanek δ in drugi odvod zanemarimo in funkcijo zapišemo aproksimativno kot

$$f(x_k) = f(x_{k-1}) + f'(x_{k-1})(x_k - x_{k-1}). \quad (5.14)$$

Če iščemo ničlo funkcije $f(x_k)$, potem lahko enačbo zapišemo kot

$$0 = f(x_{k-1}) + f'(x_{k-1})(x_k - x_{k-1}). \quad (5.15)$$

Sedaj lahko izrazimo izračun nove vrednosti argumenta funkcije kot

$$x_k = x_{k-1} - \frac{f(x_{k-1})}{f'(x_{k-1})}. \quad (5.16)$$

Iteriranje enačbe 5.19 pripelje v lokalno ničlo funkcije $f(x)$.

Če želimo izračunati lokalni ekstrem funkcije $f(x)$, potem moramo iskati lokalno ničlo odvoda funkcije $f'(x)$. To pomeni, da razvijemo odvod funkcije po Taylorjevi vrsti in v aproksimativni obliki dobimo

$$f'(x_k) = f'(x_{k-1}) + f''(x_{k-1})(x_k - x_{k-1}). \quad (5.17)$$

Ker iščemo ničlo funkcije $f'(x_k)$, lahko enačbo zapišemo kot

$$0 = f'(x_{k-1}) + f''(x_{k-1})(x_k - x_{k-1}). \quad (5.18)$$

Izračun vrednost argumenta x v naslednji iteraciji lahko izrazimo kot

$$x_k = x_{k-1} - \frac{f'(x_{k-1})}{f''(x_{k-1})}. \quad (5.19)$$

Iteriranje enačbe 5.19 pripelje v lokalni ekstrem funkcije $f(x)$. Ta zapis je poenostavitev zapisa v enačbi 5.12 za enodimensionalni primer s parametrom $\eta = 1$.

Izpeljava Newtonove metode za večdimenzionalni problem. Ničlo nelinearne kriterijske funkcije $I(\underline{\theta})$ lahko poiščemo z razvojem okoli vektorja parametrov $\underline{\theta}_{k-1}$ po Taylorjevi vrsti

$$I(\underline{\theta}_k) = I(\underline{\theta}_{k-1}) + \underline{g}_{k-1}^T \Delta \underline{\theta}_{k-1} + \frac{1}{2!} \Delta \underline{\theta}_{k-1}^T \underline{H}_{k-1} \Delta \underline{\theta}_{k-1} + \delta, \quad (5.20)$$

kjer je δ ostanek, ki predstavlja vsoto od drugega člena do neskončnosti. Če pri razvoju upoštevamo samo člene do prve vrste (do linearnega člena), dobimo naslednji zapis

$$I(\underline{\theta}_k) \approx I(\underline{\theta}_{k-1}) + \underline{g}_{k-1}^T \Delta \underline{\theta}_{k-1}, \quad (5.21)$$

kjer je $\Delta \underline{\theta}_{k-1} = \underline{\theta}_k - \underline{\theta}_{k-1}$ in $\underline{g}_{k-1} = \underline{g}(\underline{\theta}_{k-1})$. Za iskanje ničle funkcije moramo rešiti enačbo

$$0 = I(\underline{\theta}_{k-1}) + \underline{g}_{k-1}^T \Delta \underline{\theta}_{k-1}, \quad (5.22)$$

kar privede do rekurzivne rešitve, pri iskanju ničle

$$\underline{\theta}_k = \underline{\theta}_{k-1} - \left(\underline{g}_{k-1}^T \underline{g}_{k-1} \right)^{-1} \underline{g}_{k-1}^T I(\underline{\theta}_{k-1}). \quad (5.23)$$

Ker pa iščemo optimum kriterijske funkcije, moramo iskati ničlo njenega gradijenta, oziroma vrednost argumenta, kjer je gradienat po vseh komponentah enak nič. Zaradi tega po enakem principu razvijemo gradienat kriterijske funkcije \underline{g}_{k-1} . Tako dobimo

$$\underline{g}_k \approx \underline{g}_{k-1} + \underline{H}_{k-1} \Delta \underline{\theta}_{k-1}, \quad (5.24)$$

kjer je $\underline{H}_{k-1} = \underline{H}(\underline{\theta}_{k-1})$. Za iskanje ničle gradijenta moramo rešiti enačbo

$$\underline{g}_k = \underline{g}_{k-1} + \underline{H}_{k-1} \Delta \underline{\theta}_{k-1}, \quad (5.25)$$

kar nas privede do končne enačbe za Newtonovo optimizacijsko metodo

$$\underline{\theta}_k = \underline{\theta}_{k-1} - \underline{H}_{k-1}^{-1} \underline{g}_{k-1}. \quad (5.26)$$

Lastnosti Newtonove metode. Newtonova metoda ima naslednje lastnosti:

- zahteva izračun drugega odvoda,
- kubična računska zahtevnost ($\mathcal{O}(n^3)$), ker je zahtevana matrična inverzija,
- kvadratična zahtevnost po spominskem prostoru, ker shranjujemo Hessovo matriko,
- najhitrejsa konvergenca med nelinearnimi optimizacijami,
- omogoča eno samo iteracijo za linearne optimizacijske probleme,
- najprimernejša za majhne probleme (do deset parametrov).

Primer 5.4.2. Newtonova minimizacija funkcije

V tem primeru iščemo minimum Rosenbrockove funkcije z uporabo Newtonove metode. Drugi odvod je izračunan analitično. Število iteracij za rešitev problema je enako 17 in 11 za začetek iz prvega začetnega pogoja in iz drugega začetnega pogoja. Slika 5.9 kaže konvergenco iz prvega začetnega pogoja. Boljše rezultate od te metode na tej funkciji daje samo Gauss-Newton metoda. Je pa edina metoda, ki zahteva izračun Hessove matrike. V okolini minimuma ima zelo hitro konvergenco, ker je drugi red razvoja po Taylerjovi vrsti tam zelo natančen.

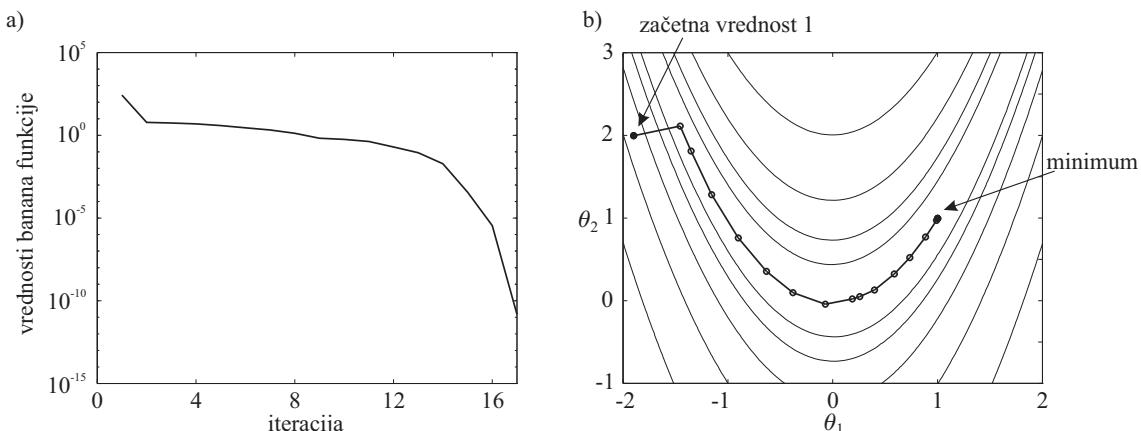
5.4.4 Kvazi-Newtonova metoda

Največja pomanjkljivost Newtonove metode je zahteva po odvodu drugega reda. Če ga ni mogoče izračunati analitično, je potrebno Hessovo matriko izračunati z metodo končnih differenc, kar pa zahteva $\mathcal{O}(n^2)$ operacij. V tem primeru problemov srednjih dimenzij je metoda nepraktična. Tudi v primeru, ko je drugi odvod izračunljiv analitično, se pojavi problem matrične inverzije matrike $n \times n$, ki omeji metodo na manjše dimenzije. Ideja Kvazi-Newtonove metode je v izračunu aproksimativne Hessove matrike ali njenega inverza, ki je dana z enačbo 5.12

$$\underline{\theta}_k = \underline{\theta}_{k-1} - \eta_{k-1} \hat{\underline{H}}_{k-1}^{-1} \underline{g}_{k-1}, \quad (5.27)$$

$$\text{pri } \hat{\underline{H}}_k = \hat{\underline{H}}_{k-1} + \underline{Q}_{k-1} \quad \text{ali} \quad \hat{\underline{H}}_k^{-1} = \hat{\underline{H}}_{k-1}^{-1} + \tilde{\underline{Q}}_{k-1}. \quad (5.28)$$

Metoda z aproksimacijo inverza Hessove je daleč najbolj uporabna metoda, saj pri tem ni potrebno računati matričnega inverza. Inverz izračunamo v vsakem koraku direktno. Začnemo z začetno vrednostjo $\underline{H}_0 = \underline{I}$.



Slika 5.9. Newtonova minimizacija Rosenbrockove funkcije a) glede na iteracije, b) v parameterskem prostoru (krogci označujejo evaluacijo funkcije).

Najpogostejsi in v večini primerov najboljši način izračunavanja inverza matrike je Broyden-Fletcher-Goldfarb-Shannova (BFGS) formula [105], ki jo zapišemo z

$$\hat{H}_k^{-1} = \left(I - \frac{\Delta \underline{\theta}_{k-1} \Delta \underline{g}_{k-1}^T}{\Delta \underline{\theta}_{k-1}^T \Delta \underline{g}_{k-1}} \right) \hat{H}_{k-1}^{-1} \left(I - \frac{\Delta \underline{\theta}_{k-1} \Delta \underline{g}_{k-1}^T}{\Delta \underline{\theta}_{k-1}^T \Delta \underline{g}_{k-1}} \right)^T + \frac{\Delta \underline{\theta}_{k-1} \Delta \underline{\theta}_{k-1}^T}{\Delta \underline{\theta}_{k-1}^T \Delta \underline{g}_{k-1}}, \quad (5.29)$$

kjer je $\Delta \underline{\theta}_{k-1} = \underline{\theta}_k - \underline{\theta}_{k-1}$ in $\Delta \underline{g}_{k-1} = \underline{g}_k - \underline{g}_{k-1}$.

Najpomembnejše lastnosti Kvazi-Newtonovih metod so:

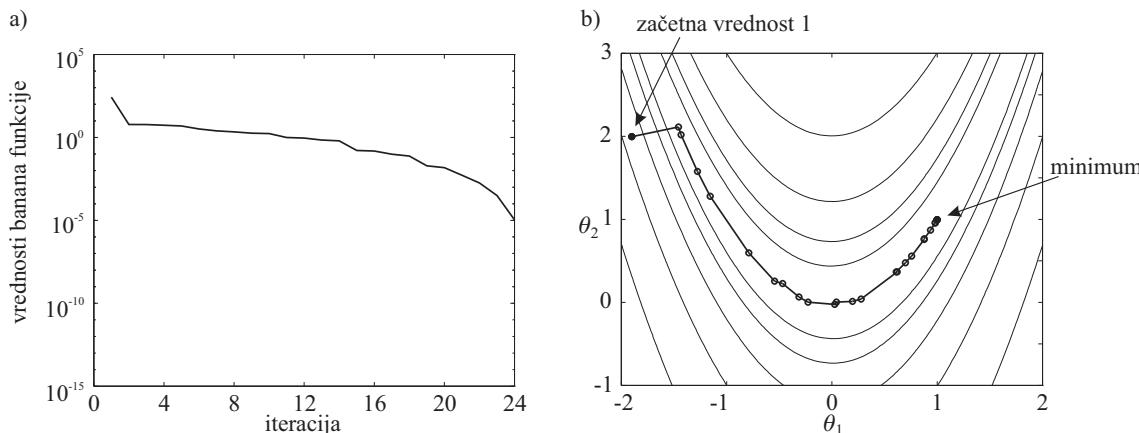
- ne potrebujemo drugih odovodov,
- kvadratična računska zahtevnost zaradi multiplikacije matrik,
- kvadratične zahteve po spominskem prostoru zaradi shranjevanja Hessove matrike,
- hitra konvergenca,
- zahteva največ n iteracij v primeru linearnega optimizacijskega problema,
- najbolj primerena za probleme srednje velikosti (okoli 100 parametrov).

Primer 5.4.3. Kvazi-Newtonova metoda za minimizacijo funkcije

V tem primeru iščemo minimum Rosenbrockove funkcije z uporabo BFGS Kvazi-Newtonove metode. Metoda uporablja le informacijo o gradientu in izračunava inverzno Hessianovo matriko glede na enačbo 5.29. Število izračunov funkcije in število iteracij je v primeru prve začetne vrednosti enako 79 in 24, v primeru druge začetne vrednosti pa 55 in 17. Na sliki 5.10 vidimo konvergenco iz prve začetne vrednosti. Metoda konvergira počasneje kot klasična Newtonova metoda, saj ne uporabimo direktno Hessove matrike. Seveda pa je Kvazi-Newtonova metoda neprimerljivo hitrejša od metod končnih differenc.

5.4.5 Metoda konjugiranih gradientov

Vse Kvazi-Newtonove metode imajo kvadratično naraščajoče zahteve po spominskem prostoru glede na število optimiranih parametrov n . Za probleme z velikim številom parameterov je izračunavanje Hessove matrike včasih nesmiselno. Metoda konjugiranih gradientov ne izračunava aproksimacije Hessove matrike in je zato v spominskih zahtevah linearne glede na število parametrov n . Metodo konjugiranih gradientov dobimo, če matriko \underline{H}_{k-1}^{-1} v BFGS enačbi 5.29 v vsaki iteraciji resetiramo in postavimo na I . Na ta način je možno korigirati smer iskanja



Slika 5.10. Kvazi-Newtonova metoda v primeru iskanja minimuma Rosenbrockove funkcije. Hitrost konvergencije a) glede na iteracije, b) v parameterskem prostoru.

\underline{p} direktno. Zaradi konjugiranih parametrov izgleda metoda kot zelo približna aproksimacija Kvazi-Newtonove metode, splošno gledano pa je smer iskanja veliko manj ustrezena in zahteva metoda veliko več iteracij. Seveda pa je računski čas manjši, ker je vsaka iteracija računsko bistveno manj zahtevna. Metodo zapišemo z naslednjimi enačbami

$$\underline{\theta}_k = \underline{\theta}_{k-1} - \eta_{k-1} \underline{p}_{k-1} \quad (5.30a)$$

$$\text{pri } \underline{p}_{k-1} = \underline{g}_{k-1} - \beta_{k-1} \underline{p}_{k-2}, \quad (5.30b)$$

kjer glede na skalar β_{k-1} razlikujemo več različnih metod konjugiranih gradientov. Najpogostejsa je metoda Fletcher-Reeves, kjer je

$$\beta_{k-1} = \frac{\underline{g}_{k-1}^T \underline{g}_{k-1}}{\underline{g}_{k-2}^T \underline{g}_{k-2}}. \quad (5.31)$$

Skalar β predstavlja akumulacijo znanja iz predhodnjih iteracij. Na ta način lahko metodo konjugiranih gradientov vidimo kot kompromis med gradientno in Kvazi-Newtonovo metodo.

V primeru linearnih problemov so vse izbire za β_{k-1} enakovredne in metoda konvergira v n iteracijah.

V enačbi 5.30b določimo konjugirane smeri iskanja. Dve smeri sta konjugirani, če med \underline{p}_i in \underline{p}_j velja, da je $\underline{p}_i^T \underline{H} \underline{p}_j = 0$. To pomeni, da sta \underline{p}_i in \underline{p}_j neodvisni smeri iskanja. Geometrijska interpretacija pa je v tem, da se v vsaki iteraciji k iskanje izvaja v smeri, ki je ortogonalna na zadnje spremembe gradienta $\Delta \underline{g}_i$, $i = 0, \dots, k-1$. Na ta način se lahko prepreči efekt "cik-cakanja", ki ga dobimo pri metodi največjega gradienta. V [10] je izpostavljeno, da se z iteracijami konjugiranost smeri poslabšuje in je potrebno algoritem ponovno zagnati po vsakih n iteracijah in nastaviti smer iskanja \underline{p} v smeri negativnega gradienta.

Glavne lastnosti metode konjugiranih gradientov so naslednje:

- ni zahteve po izračunu drugega odvoda,
- linearna računska zahtevnost,
- linearna zahtevnost po spominskem prostoru,
- hitra konvergenca,
- potrebuje največ n iteracij v primeru linearne optimizacijskega problema,
- najbolj primerna za velike probleme (več kot 1000 parameterov).

Primer 5.4.4. Metoda konjugiranih gradientov za minimizacijo funkcije

V primeru je prikazana minimizacija Rosenbrockove funkcije z uporabo metode konjugiranih gradientov po Fletcher in Reevesu. Metoda uporablja poleg gradienta tudi informacijo o predhodnih gradientih. Ne uporablja pa izračuna Hessove matrike tako kot je to v primeru Kvazi-Newtonove metode. Računska zahtevnost je zato manjša, potrebno pa je več iteracij. V tem primeru so bili potrebeni 203 izračuni funkcije in 36 iteracij za prvi začetni pogoj in 261 izračunov funkcije in 44 iteracij za drugi začetni pogoj. Slika 5.11 prikazuje konvergenco iz prvega začetnega pogoja.

5.5 Metoda nelinearnih najmanjših kvadratov

V prejšnjem poglavju smo si ogledali optimizacijske metode, ki temeljijo na gradientu. Pri teh metodah nimamo nobenih predpostavk glede kriterijske funkcije, razen njene gladkosti. Ena od možnih kriterijskih funkcij je lahko tudi

$$I(\underline{\theta}) = \sum_{i=1}^N q_i e^2(i, \underline{\theta}), \quad (5.32)$$

kjer je $e(i)$ napaka med izhodnimi podatki in izhodom modela. To je zelo pogost primer. Če je problem linearen v parametrih, potem dobimo metodo najmanjših kvadratov. Če pa so parametri nelinearni, potem v primeru kriterijske funkcije

$$I(\underline{\theta}) = \sum_{i=1}^N f^2(i, \underline{\theta}) \quad (5.33)$$

dobimo problem nelinearnih najmanjših kvadratov. Enačba 5.32 je poseben primer enačbe 5.33. V vektorski obliki lahko enačbo 5.33 zapišemo

$$I(\underline{\theta}) = \underline{f}^T \underline{f} \quad \text{pri } \underline{f} = [f(1, \underline{\theta}) \ f(2, \underline{\theta}) \ \cdots \ f(N, \underline{\theta})]^T. \quad (5.34)$$

Če izračunamo gradijenta glede na parameter θ_j , dobimo

$$g_j = \frac{\partial I(\underline{\theta})}{\partial \theta_j} = 2 \sum_{i=1}^N f(i) \frac{\partial f(i)}{\partial \theta_j}. \quad (5.35)$$

Odvajanje vektorske funkcije \underline{f} po vseh parametrih $\underline{\theta}$ rezultira v Jacobijevi matriki

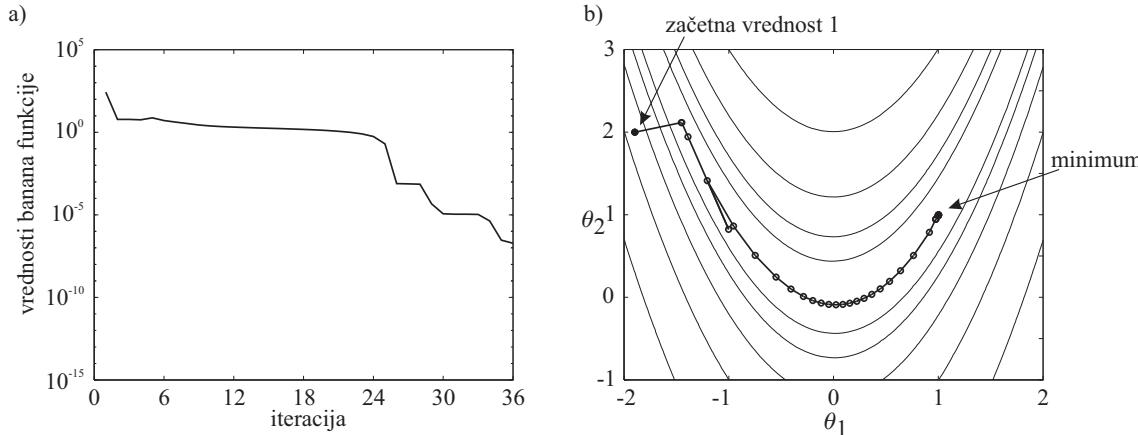
$$\underline{J} = \begin{bmatrix} \frac{\partial f(1)}{\partial \theta_1} & \cdots & \frac{\partial f(1)}{\partial \theta_n} \\ \vdots & & \vdots \\ \frac{\partial f(N)}{\partial \theta_1} & \cdots & \frac{\partial f(N)}{\partial \theta_n} \end{bmatrix}. \quad (5.36)$$

Tako lahko gradient zapišemo kot

$$\underline{g} = 2 \underline{J}^T \underline{f}. \quad (5.37)$$

Elemente Hessove matrike kriterijske funkcije dobimo z izračunom odvodov gradijenta v enačbi 5.46 glede na parametere θ_p

$$H_{pj} = \frac{\partial^2 I(\underline{\theta})}{\partial \theta_p \partial \theta_j} = 2 \sum_{i=1}^N \left(\frac{\partial f(i)}{\partial \theta_p} \frac{\partial f(i)}{\partial \theta_j} + f(i) \frac{\partial^2 f(i)}{\partial \theta_p \partial \theta_j} \right). \quad (5.38)$$



Slika 5.11. Metoda konjugiranih gradijentov za minimizacijo Rosenbrockove funkcije a) glede na iteracije, b) v parameterskem prostoru.

Prvi člen v enačbi 5.38 je kvadrat Jacobijeve matrike za funkcijo \underline{f} , drugi člen pa je funkcija $\underline{f}(i)$ multiplicirana s Hessovo matriko funkcije $\underline{f}(i)$. Če izrazimo člene Hessove matrike za funkcijo $\underline{f}(i)$ kot $T_{pj}(i) = \partial^2 f(i) / \partial \theta_p \partial \theta_j$, potem dobimo Hessovo matriko kriterijske funkcije v enačbi 5.38 kot

$$\underline{H} = 2 \underline{J}^T \underline{J} + 2 \sum_{i=1}^N f(i) \underline{T}(i). \quad (5.39)$$

Če drugi člen v enačbi 5.39 zapišemo kot \underline{S} , potem lahko Hessovo matriko za kriterijsko funkcijo zapišemo kot

$$\underline{H} = 2 \underline{J}^T \underline{J} + 2 \underline{S}. \quad (5.40)$$

Na ta način lahko metoda nelinearnih najmanjših kvadratov doseže večjo učinkovitost kot splošne optimizacijske metode, ker izkorišča informacijo o Hessovi matriki. Obstajata dve vrsti metod: ena zanemari \underline{S} , druga pa ta člen upošteva.

Aproksimacija Hessove matrike $\underline{H} \approx 2 \underline{J}^T \underline{J}$ je upravičena samo takrat, ko je $\underline{S} \approx 0$. To velja samo za majhne vrednosti $f(i)$. Največja prednost teh metod je v tem, da je za izračun Hessove matrike potreben samo prvi odvodi (Jacobijeva matrika). Najpomembnejši sta Gauss-Newtonova in Levenberg-Marquardtova metoda.

Druga skupina metod pa aproksimira ali izračunava \underline{S} člen, kar pa prinaša veliko računska zahtevnost.

Izpeljava metode nelinearnih najmanjših kvadratov. Z razvojem gradiента po Taylorju dobimo naslednji zapis

$$\underline{g}_k \approx \underline{g}_{k-1} + \underline{H}_{k-1} \Delta \underline{\theta}_{k-1}. \quad (5.41)$$

Če v razvoju upoštevamo izpeljavo za gradient iz enačbe 5.47 in Hessovo matriko iz enačbe 5.39 in pri tem upoštevamo aproksimacijo z razvojem z vrsto prvega reda, dobimo naslednji zapis

$$2 \underline{J}_k^T \underline{f}_k = 2 \underline{J}_{k-1}^T \underline{f}_{k-1} + 2 \underline{J}_{k-1}^T \underline{J}_{k-1} \Delta \underline{\theta}_{k-1}. \quad (5.42)$$

Ob tem, da iterativno iščemo ničlo tega gradiента, lahko levo stran enačbe postavimo na nič in po preureeditvi enačbe dobimo naslednji zapis

$$\Delta \underline{\theta}_{k-1} = - (\underline{J}_{k-1}^T \underline{J}_{k-1})^{-1} \underline{J}_{k-1}^T \underline{f}_{k-1} \quad (5.43)$$

ali

$$\underline{\theta}_k = \underline{\theta}_{k-1} - (\underline{J}_{k-1}^T \underline{J}_{k-1})^{-1} \underline{J}_{k-1}^T \underline{f}_{k-1}. \quad (5.44)$$

Predpostavimo, da je \underline{f} napaka med merjenim izhodom \underline{y} in izhodom modela $\hat{\underline{y}}(\underline{\theta})$, ki je nelinearen v parametrih $\underline{\theta}$

$$\underline{f} = \underline{y} - \hat{\underline{y}}(\underline{\theta}). \quad (5.45)$$

Če izračunamo gradient glede na parameter θ_j , dobimo

$$g_j = \frac{\partial I(\underline{\theta})}{\partial \theta_j} = 2 \sum_{i=1}^N f(i) \frac{\partial(y(i) - \hat{y}(i, \underline{\theta}))}{\partial \theta_j} = -2 \sum_{i=1}^N f(i) \frac{\partial \hat{y}(i, \underline{\theta})}{\partial \theta_j}. \quad (5.46)$$

Tako lahko gradient zapišemo kot

$$\underline{g} = -2 \underline{J}^{*\top} \underline{f}, \quad (5.47)$$

kjer je Jacobijeva matrika definirana kot odvod izhoda modela $\hat{\underline{y}}(\underline{\theta})$ po vseh parametrih $\underline{\theta}$

$$\underline{J}^* = \begin{bmatrix} \partial \hat{y}(1, \underline{\theta}) / \partial \theta_1 & \cdots & \partial \hat{y}(1, \underline{\theta}) / \partial \theta_n \\ \vdots & & \vdots \\ \partial \hat{y}(N, \underline{\theta}) / \partial \theta_1 & \cdots & \partial \hat{y}(N, \underline{\theta}) / \partial \theta_n \end{bmatrix}. \quad (5.48)$$

Ob upoštevanju novega izračuna Jacobijeve matrike dobimo naslednji izračun za adaptacijo parametrov

$$\underline{\theta}_k = \underline{\theta}_{k-1} + (\underline{J}_{k-1}^{*T} \underline{J}_{k-1}^*)^{-1} \underline{J}_{k-1}^{*T} \underline{f}_{k-1}. \quad (5.49)$$

Izpeljava metode nelinearnih najmanjših kvadratov na primeru. Metodo nelienarnih najmanjših kvadratov bomo izpeljali še malo drugače.

Predpostavimo funkcijo $y(u)$ in model te funkcije $\hat{y}(u, \underline{\theta})$, ki je nelinearen v parametrih in kjer je $\underline{\theta}$ vektor nelinearnih parametrov in u vrednost vhodne spremenljivke. Vektor parametrov je dimenzije $m \times 1$ in ga zapišemo kot $\underline{\theta}_k = [\theta_k^1, \dots, \theta_k^m]^T$.

Vrednost funkcije v časovnem trenutku i , kjer je vhodna spremenljivka enaka u_i , lahko aproksimiramo s prvim členom Taylorjeve vrste

$$\hat{y}(u_i, \underline{\theta}_k) = \hat{y}(u_i, \underline{\theta}_{k-1}) + \sum_{j=1}^m \frac{\partial \hat{y}(u_i, \underline{\theta}_{k-1})}{\partial \theta_j} (\theta_k^j - \theta_{k-1}^j), \quad (5.50)$$

kjer sta θ_{k-1}^j in θ_k^j vrednosti j -tega parametra v vektorju $\underline{\theta}_{k-1}$ in $\underline{\theta}_k$. Vrednost parcialnega odvoda po j -tem parametru in pri vhodni spremenljivki u_i lahko enostavnejše zapišemo s koeficientom Jacobijeve matrike na naslednji način

$$J_{ij} = \frac{\partial \hat{y}(u_i, \underline{\theta}_{k-1})}{\partial \theta_j}, \quad (5.51)$$

razliko v parametrih pa definiramo kot $\Delta \theta_{k-1}^j = \theta_k^j - \theta_{k-1}^j$. Ob upoštevnju tega lahko aproksimacijo funkcije zapišemo v bolj zgoščeni obliki

$$\hat{y}(u_i, \underline{\theta}_k) = \hat{y}(u_i, \underline{\theta}_{k-1}) + \underline{J}_i^*(u_i, \underline{\theta}_{k-1}) \underline{\Delta \theta}_{k-1}, \quad (5.52)$$

kjer je \underline{J}_i^* vrstični vektor Jacobijevih koeficientov $\underline{J}_i^* = [J_{i1}, \dots, J_{im}]$ pri vrednosti vektorja parametrov $\underline{\theta}_{k-1}$. Pogrešek med dejanskim izhodom $y(u_i)$ in aproksimiranim izhodom $\hat{y}(u_i, \underline{\theta}_k)$ lahko zapišemo

$$e_i = y(u_i) - \hat{y}(u_i, \underline{\theta}_k), \quad i = 1, \dots, n, \quad (5.53)$$

kjer je n število n vhodno-izhodnih parov podatkov. Gornji pogrešek lahko razširimo na

$$e_i = y(u_i) - \hat{y}(u_i, \underline{\theta}_{k-1}) - \underline{J}_i^*(u_i, \underline{\theta}_{k-1}) \underline{\Delta \theta}_{k-1}, \quad i = 1, \dots, n. \quad (5.54)$$

Če zapis razširimo v matrično obliko, ki upošteva celoten niz vhodno-izhodnih podatkov potem dobimo naslednji zapis

$$\underline{e}(\underline{\theta}_{k-1}) = \underline{y} - \hat{\underline{y}}(\underline{\theta}_{k-1}) - \underline{J}^*(\underline{\theta}_{k-1}) \underline{\Delta \theta}_{k-1}, \quad (5.55)$$

kjer je $\underline{J}^*(\underline{\theta}_{k-1})$ Jacobijeva matrika pri vrednosti vektorja parametrov $\underline{\theta}_{k-1}$, ki je enaka $\underline{J}^*(\underline{\theta}_{k-1}) = [\underline{J}_1^{*T}(\underline{\theta}_{k-1}), \dots, \underline{J}_n^{*T}(\underline{\theta}_{k-1})]^T$.

S spremenjanjem vektorja parametrov $\underline{\theta}$ iščemo minimum funkcije

$$I(\underline{\theta}_{k-1}) = \underline{e}^T(\underline{\theta}_{k-1}) \underline{e}(\underline{\theta}_{k-1}), \quad (5.56)$$

kar nam da rešitev

$$\underline{\Delta\theta}_{k-1} = (\underline{J}^{*T}(\underline{\theta}_{k-1}) \underline{J}^*(\underline{\theta}_{k-1}))^{-1} \underline{J}^{*T}(\underline{\theta}_{k-1}) (\underline{y} - \hat{\underline{y}}(\underline{\theta}_{k-1})) \quad (5.57)$$

ali

$$\underline{\theta}_k = \underline{\theta}_{k-1} + (\underline{J}^{*T}(\underline{\theta}_{k-1}) \underline{J}^*(\underline{\theta}_{k-1}))^{-1} \underline{J}^{*T}(\underline{\theta}_{k-1}) (\underline{y} - \hat{\underline{y}}(\underline{\theta}_{k-1})). \quad (5.58)$$

Primer 5.5.1. Za primer bomo poskušali poiskati model eksponencialne funkcije

$$y = A \exp(-(u - \mu)^2 / (2\sigma^2)). \quad (5.59)$$

Imamo množico n vhodno-izhodnih podatkov, parov u_i in y_i . Parcialni odvodi so v tem primeru enaki

$$\frac{\partial y}{\partial A} = \exp(-(u - \mu)^2 / (2\sigma^2)), \quad (5.60)$$

$$\frac{\partial y}{\partial \mu} = \frac{A(u - \mu)}{\sigma^2} \exp(-(u - \mu)^2 / (2\sigma^2)), \quad (5.61)$$

$$\frac{\partial y}{\partial \sigma} = \frac{A(u - \mu)^2}{\sigma^3} \exp(-(u - \mu)^2 / (2\sigma^2)). \quad (5.62)$$

Nelinearne parametre v primeru obravnavane funkcije razvrstimo v vektor $\underline{\theta} = [A \mu \sigma]^T$. Poljubno izberemo začetno vrednost vektorja parametrov $\underline{\theta}_0$ in izračunamo $\hat{\underline{y}}(\underline{\theta}_0)$ in izračunamo Jacobijevu matriko $\underline{J}(\underline{\theta}_0)$ dimenzije $n \times m$, kjer imamo vseh m parcialnih odvodov izračunanih pri n vhodnih vrednostih u_i in vrednosti vektorja parametrov $\underline{\theta}_0$. V našem primeru je $m = 3$. Z uporabo enačbe 5.57 izračunamo spremembo vektorja parametrov in z njo ustrezno spremenimo vektor parametrov. Postopek ponovimo, dokler parametri ne konvergirajo.

Primer je prikazan v datoteki *NLS_Gauss.m*.

5.5.1 Gauss-Newtonova metoda

Gauss-Newtonova metoda je verzija splošne Newtonove metode, ki je predstavljena v enačbi 5.12. Ker gradient predstavimo kot $\underline{g} = 2\underline{J}^T \underline{f}$ in Hessovo matriko kot $\underline{H} \approx 2\underline{J}^T \underline{J}$, potem dobimo Gauss-Newtonov algoritem

$$\underline{\theta}_k = \underline{\theta}_{k-1} - \eta_{k-1} (\underline{J}_{k-1}^T \underline{J}_{k-1})^{-1} \underline{J}_{k-1}^T \underline{f}_{k-1}. \quad (5.63)$$

Primer 5.5.2. Gauss-Newtonova metoda za minimizacijo funkcije

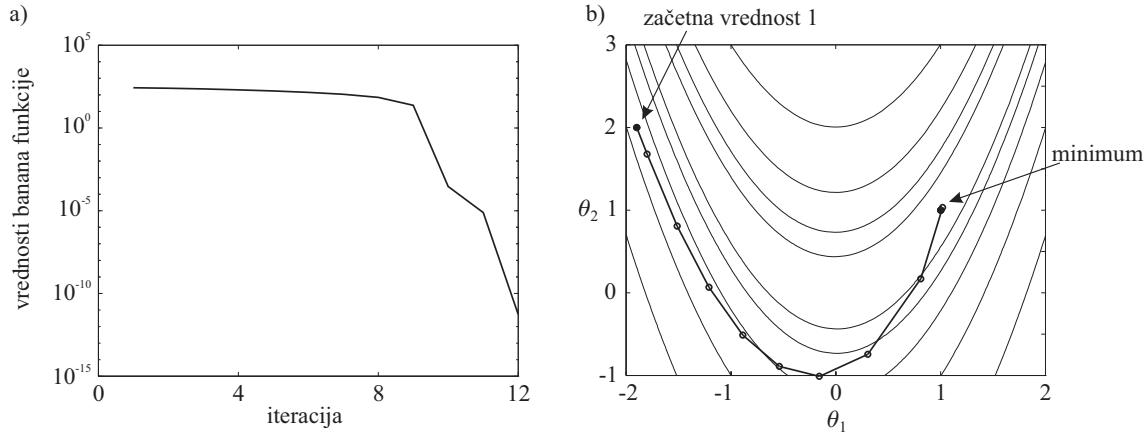
V tem primeru minimum Rosenbrockove funkcije iščemo z uporabo Gauss-Newtonove metode. Funkcijo moramo zapisati v obliki vsote kvadratov, tako kot v enačbi 5.33. Rosenbrockova funkcija $I(\underline{\theta}) = 100(\theta_2 - \theta_1^2)^2 + (1 - \theta_1)^2$ je sestavljena iz dveh kvadratičnih členov

$$\underline{f}(\underline{\theta}) = \begin{bmatrix} 10(\theta_2 - \theta_1^2) \\ 1 - \theta_1 \end{bmatrix}. \quad (5.64)$$

Tako dobimo Jacobijevu matriko

$$\underline{J}(\underline{\theta}) = \begin{bmatrix} -20\theta_1 & 10 \\ -1 & 0 \end{bmatrix}. \quad (5.65)$$

V Gauss-Newtonovi metodi je Hessova matrika kriterijske funkcije aproksimirana z $2\underline{J}^T \underline{J}$. Za konvergenco metode potrebujemo 40 izračunov funkcije in 12 iteracij, če začnemo iz prvega začetnega pogoja in 21 izračunov in 7 iteracij, če začnemo iz drugega začetnega pogoja. Slika 5.12 prikazuje konvergenco iz prvega začetnega pogoja. Ta metoda je v primeru te funkcije najhitrejša



Slika 5.12. Gauss-Newtonova metoda za minimizacijo funkcije. Hitrost konvergencije a) glede na iteracije, b) v parameterskem prostoru.

5.5.2 Levenberg-Marquardtova metoda

Levenberg-Marquardtova metoda je razširitev Gauss-Newtonove metode. Ideja je v regularizacijski modifikaciji enačbe 5.62

$$\underline{\theta}_k = \underline{\theta}_{k-1} - \eta_{k-1} (\underline{J}_{k-1}^T \underline{J}_{k-1} + \alpha_{k-1} \underline{I})^{-1} \underline{J}_{k-1}^T \underline{f}_{k-1}, \quad (5.66)$$

kjer je člen $\alpha_{k-1} \underline{I}$ regularizacijski parameter, ki rešuje problem slabe pogojenosti matrike $\underline{J}_{k-1}^T \underline{J}_{k-1}$.

6. Globalna nelinearna optimizacija

Razlikovanje med lokalno in globalno optimizacijo je smiselno samo v kontekstu nelinearnih optimizacijskih problemov, saj linearni problemi nimajo lokalnih ekstremov. Nelinearne lokalne optimizacijske metode začno iz začetnih, ki jih poda uporabnik in iščejo v smeri, ki jo dobimo s pomočjo odvodov, prvega in včasih tudi drugega. Na ta način najdemo optimum, ki je blizu začetnih pogojev in seveda ni nujno globalni. Metode, ki jih obravnavamo v tem poglavju iščejo globalni optimum. V vseh navedenih primerih je težko pokazati konvergenco v končnem času iskanja. Včasih se zadovoljimo z zadost dobrim lokalnim optimumom. Težko pa je ugotoviti kako dober je lokalni optimum, ker običajno nimamo informacije o vrednosti globalnega optimuma.

Najenostavnejša strategija za iskanje dobrega lokalnega optimuma je pristop z večkratnim zaganjanjem lokalne nelinearne optimizacije iz različnih začetnih vrednosti parametrov. v vsakem teku konvergira optimizacija v najbližji lokalni optimum in najboljšega izberemo kot končni rezultat. Metoda je po svoji naravi paralelna, ker lahko vsak tek zaganjam na svojem računalniku. Problem pa ostaja v izbiri začetnih vrednosti.

Druga strategija za razširitev lokalnih metod na iskanje globalnega optimuma je dodajanje naključnih vrednosti na nove izračunane vrednosti parametrov. Dodane naključne vrednosti upočasnijo hitrost konvergencije, vendar pa hkrati omogočajo izhod iz lokalnega optimuma. Standardna deviacija dodane naključne vrednosti se z številom iteracij zmanjšuje, da pospešimo konvergenco. Glede na svojo naravo pa takšne metode vseeno iščejo le v okolini začetnih parametrov.

Globalne nelinearne tehnike uporabimo v primeru, če je izpolnjen vsaj en spodnji kriterij:

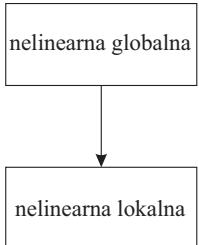
- iščemo globalni optimum,
- funkcija cene in njeni odvodi so zelo nelinearni,
- določeni parametri niso realna števila, temveč cela ali binarna, ali pa celo mešan problem.

Največji problem vseh globalnih optimizacijskih metod je računska kompleksnost. Delitev prostora parametrov na posamezne intervale, ki določajo računsko mrežo problema narašča eksponencialno s številom parametrov n . Problem je lasten vsem algoritmom globalne optimizacije in se ne nanaša samo na določene algoritme. Problem se v večini algoritmov rešuje tako, da bolj natančno pregledamo področja, ki bolj verjetno vsebujejo optimum. Parametri v drugih regijah pa se pregledujejo z manjšo natančnostjo. Tak princip je lahko uspešen samo takrat, ko je sistem predvidljiv, kar pa je v realnih problemih največkrat tudi res.

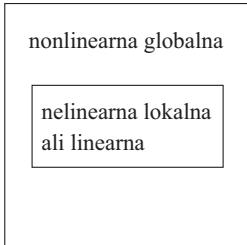
Večina globalnih nelinearnih tehnik vsebuje stohastične elemente. Stohastični elementi omogočajo pobeg iz lokalnih optimumov in širše iskanje. Na ta način ni nujno, da je funkcija cenalke monotono padajoča glede na iteracije. Zaradi tega je seveda najbolj smiselno, da ob zaključku optimizacije vzamemo najboljšo rešitev, ne pa najboljše rešitve ob zaključni iteraciji.

V grobem bi lahko rekli, da *globalne metode omogočajo najti dobro področje, z lokalnimi metodami pa lahko najdemo dobre točke parameterskega prostora*.

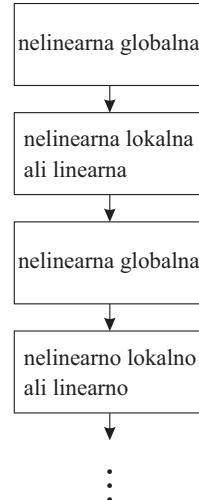
a) dvodelna optimizacija



b) vgnezdena optimizacija



c) postopna optimizacija



Slika 6.1. Kombinacija različnih optimizacijskih tehnik: a) globalna optimizacija vseh parametrov, ki ji sledi lokalna optimizacija vseh parametrov, b) globalna optimizacija nekaterih parametrov ali strukture z vgnezdeno (v vsaki iteraciji), ki ji sledi lokalna optimizacija vseh parametrov nelinearno lokalno ali linearne optimizacije ostalih parametrov, c) ponavljajoča globalna optimizacija z nekaterimi parametri, ki ji sledi nelinearna lokalna ali linearne optimizacije ostalih parametrov.

Zaradi dobrih in slabih lastnosti nelinearne globalne, nelinearne lokalne in linearne optimizacije je v praksi smiselno kombinirati različne tehnike. Slika 6.1 prikazuje kombinacije različnih optimizacijskih tehnik.

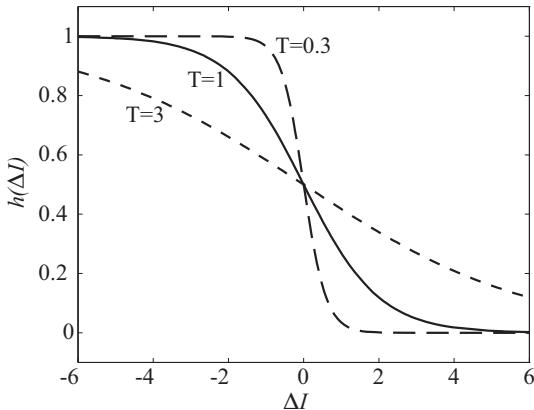
6.1 Simulirano ohlajanje (SA)

Simulirano ohlajanje je (*ang. Simulated Annealing - SA*) je Monte-Carlo (stohastična) metoda za globalno optimizacijo. Ime ima po fizikalni analogiji potencialnega polja in hitrega ali vročega delca znotraj tega polja. V splošnem se delec giblje v smeri manjše potencialne energije, ker pa ima neničelno temperaturo, oziroma kinetično energijo, se lahko giblje tudi tudi v nasprotni smeri. To povzroča naključne skoke na višjo potencialno energijo. Tako lahko delec pobegne iz lokalnega minimuma in lahko pridemo do globalne rešitve. Delec izgublja energijo in možnost premikanja nasproti potencialnega polja se tako zmanjšuje. Energija delca se mora zmanjševati počasi, da lahko dosežemo globalni optimum, drugače se lahko delec izgubi v lokalnem minimumu. V kontekstu optimizacije predstavlja delec točko v prostoru parametrov in potencialno polje predstavlja kriterijsko funkcijo. V nadaljevanju bomo predstavili originalno verzijo SA, ki jo je predlagal Kirkpatrick [64] leta 1983.

Algoritem inicializiramo z začetno temperaturo, začetno izbiro vrednosti vektorja parametrov, konstantno učenja, številom iteracij in številom poizkusov znotraj ene iteracije. Izbira začetne temperature T_0 ni enostavna, ker so vsi nelinearni problemi različni in zato pridemo do izbire s poskušanjem. Izbira T_0 pomeni določitev razmerja med lokalnim in globalnim iskanjem. Če izberemo previsoko vrednost, bo konvergenca zelo počasna, če pa je izbira prenizka, potem se algoritem ustavi prehitro v okolici začetne točke. V naslednjem določimo nov vektor parametrov s pomočjo generatorja funkcijске gostote verjetnosti $g(\cdot)$. Ta funkcija gostote verjetnosti je izbrana tako, da: (i) je verjetnost majhnih sprememb parametrov večja kot verjetnost za velike spremembe, (ii) so velike spremembe parametrov bolj verjetne pri visoki temperaturi

Algoritem 1 Pseudo koda za SA

-
- 1: Inicializacija začetne temperature delca T_0 , števila iteracij in števila poizkusov znotraj ene iteracije.
 - 2: Inicializacija začetne vrednosti vektorja parametrov $\underline{\theta}_0$ v prostoru parametrov.
 - 3: Izračun kriterijske funkcije za začetni parameter $I(\underline{\theta}_0)$.
 - 4: **repeat**
 - 5: nova iteracija $k = k + 1$
 - 6: Izračun nove točke v prostoru parametrov $\underline{\theta}_{\text{new}}$, ki jo izračunamo kot za
$$\underline{\theta}_{\text{new}} = \underline{\theta}_{k-1} + \Delta\underline{\theta}_k$$
in jo generiramo s funkcijo gostote verjetnosti $g(\Delta\underline{\theta}_k, T_k)$.
 - 7: Izračun kriterijske funkcije za nov vektor parametrov (delec) $I(\underline{\theta}_{\text{new}})$.
 - 8: Sprejmemo nov vektor parametrov z verjetnostjo $h(\Delta I_k, T_k)$, kjer je
$$\Delta I_k = I(\underline{\theta}_{\text{new}}) - I(\underline{\theta}_{k-1})$$
t.j., nastavimo $\underline{\theta}_k = \underline{\theta}_{\text{new}}$ ali pa obdržimo star vektor parametrov, t.j., $\underline{\theta}_k = \underline{\theta}_{k-1}$.
 - 9: Znižamo temperaturo delca glede na predvideno ohlajanje T_k .
 - 10: **until** izpolnjen pogoj za zaključek algoritma.
-



Slika 6.2. Verjetnost za sprejem rešitve v primeru simulacijskega ohlajanja za visoko, srednjo in nizko temperaturo T . Delci, ki zmanjšujejo kriterijsko funkcijo ($\Delta I < 0$) so vedno sprejeti z verjetnostjo, ki je višja od 0.5, delci, ki povečujejo kriterijsko funkcijo ($\Delta I > 0$) pa so sprejeti z manjšo verjetnostjo. Za zelo veliko temperaturo ($T \rightarrow \infty$) so vsi delci sprejeti z enako verjetnostjo, medtem, ko so pri nizkih temperaturah ($T \rightarrow 0$) sprejeti delci samo takrat, ko se kriterijska funkcija zmanjšuje.

delcev, kot pri nizki temperaturi. Na ta način algoritem začne preiskovati zelo širok prostor parametrov in s časom preide v preiskovanje ozkega področja. Kriterijska funkcija novega vektorja parametrov se lahko izboljša (manjša vrednost) ali pa poslabša (višja vrednost), kot v primeru prejšnjega vektorja parametrov. Tipičen pristop po metodi spusta bi sprejel vse boljše parametre in eliminiral vse slabše vrednosti vektorjev parametrov. Da bi se izognili problemu konvergencije v lokalni minimum, pa pri SA novega vektorja ne sprejmemo deterministično temveč z določeno verjetnostjo, ki je večja od izbrane. Funkcija gostote verjetnosti za sprejem rešitve $h(\cdot)$ ima naslednje lastnosti: (i) sprejem je bolj verjeten, če nov vektor parametrov izboljša kriterijsko funkcijo, (ii) verjetnost sprejema vektorja parametrov s slabšo kriterijsko funkcijo je večja pri višjih temperaturah kot pri nižjih. Funkcija gostote verjetnosti za sprejem novega vektorja parametrov je običajno naslednja [49]

$$\begin{aligned}
h(\Delta I_k, T_k) &= \frac{\exp(-I_k/T_k)}{\exp(-I_k/T_k) + \exp(-I_{k-1}/T_k)} \\
&= \frac{1}{1 + \exp(\Delta I_k/T_k)}, \tag{6.1}
\end{aligned}$$

kjer ΔI_k predstavlja razliko med kriterijsko funkcijo pri novih in starih parametrih. Slika 6.2 kaže funkcijo gostote verjetnosti za sprejem novega vektorja parametrov pri različnih temperaturah. V 8. koraku znižamo temperaturo delca po predvideni funkciji ohlajanja (*ang. annealing schedule*). Funkcija ohlajanja je temeljnega pomena za konvergenco algoritma proti globalnemu optimumu. Če izberemo Gaussovo porazdelitev za funkcijo generiranja novih vrednosti parametrov $g(\Delta \underline{\theta}, T)$ in funkcijo ohlajanja

$$T_k = \frac{T_0}{\ln k}, \tag{6.2}$$

potem je možno statistično dokazati, da algoritem konvergira v globalni optimum (*ang. annealing proof*). Ta funkcija ohlajanja se imenuje Boltzmanova (BA) in rezultira v zelo počasnem ohlajanju.

Gaussovo porazdelitve lahko zapišemo za splošen primer na naslednji način

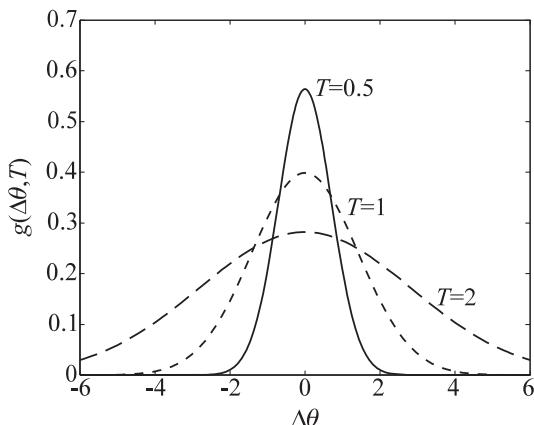
$$g(\Delta \underline{\theta}, T) = \frac{1}{(2\pi)^{\frac{d}{2}} \det(\underline{\Sigma})^{\frac{1}{2}}} \exp\left(-\frac{1}{2} \Delta \underline{\theta}^T \underline{\Sigma}^{-1} \Delta \underline{\theta}\right), \tag{6.3}$$

kjer je $\underline{\Sigma}$ kovariančna matrika, ki vsebuje informacijo o standardnih deviacijah posameznih parametrov in kovariankah med posameznimi parametri in je v primeru neodvisnih parametrov enaka $\underline{\Sigma} = \text{diag}(\sigma_1^2, \sigma_2^2, \dots, \sigma_d^2)$ in v primeru enakih varianc dobimo $\underline{\Sigma} = \sigma^2 \underline{I}$. Parameter d definira dimenzijo parameterskega prostora.

Za najenostavnejše spremjanje parametrov, kjer je $\underline{\Sigma} = \sigma^2 \underline{I}$ in je varianca odvisna od temperature T dobimo naslednjo funkcijo

$$g(\Delta \underline{\theta}, T) = \frac{1}{(2\pi)^{\frac{d}{2}} T} \exp\left(-\frac{\Delta \underline{\theta}^T \Delta \underline{\theta}}{2T^2}\right). \tag{6.4}$$

Gaussova funkcija pri različnih temperaturah T je prikazana na sliki 6.3.



Slika 6.3. Gaussova funkcija verjetnostne porazdelitve za različne temperature ohlajanja T . Nižja temperatura implicira inkrementne parametrov, ki so blizu nič. To pomeni, da so novi parametri blizu starih in da preiskujemo lokalni prostor.

Sprememba posameznega parametra je v primeru Gaussove porazdelitve $g(\Delta\theta_k^i, T_k)$ enaka kar

$$\Delta\theta_k^i = \mathcal{N}(0, T_k^2), \quad (6.5)$$

kjer je $\mathcal{N}(0, T_k^2)$ generator naključne spremenljivke z normalno porazdelitvijo.

Pogosto se za hitrejše ohlajanje (kaljenje) uporablja eksponencialna funkcija (*ang. quenching*). To lahko v določenih primerih deluje dobro, v določenih pa ne, ker so pogoji za globalno konvergenco kršeni. V smislu doseganja hitrejšega ohlajanja in s tem konvergencije, brez kršenja pogojev globalne konvergencije, so razvili metodo hitrega ohlajanja (FA) [49]. V tem primeru je Gaussova porazdelitvena funkcija zamenjana s Cauchyevim porazdelitvijo, ki je prikazana na sliki 6.4a. Ker ima funkcija "širši repe" to vodi k širšemu globalnemu iskanju. Cauchyeva porazdelitev v osredičeni obliki je podana z naslednjim enačbo

$$g(\Delta\theta, T) = \frac{\Gamma(\frac{d+1}{2})}{\Gamma(\frac{1}{2})} \frac{1}{\pi^{d/2} \det(\underline{\Sigma})^{1/2}} \frac{1}{(1 + \Delta\theta^T \underline{\Sigma}^{-1} \Delta\theta)^{\frac{d+1}{2}}}, \quad (6.6)$$

kjer z Γ označujemo gama funkcijo, ki ima vrednost $\Gamma(n) = (n - 1)!$ ali

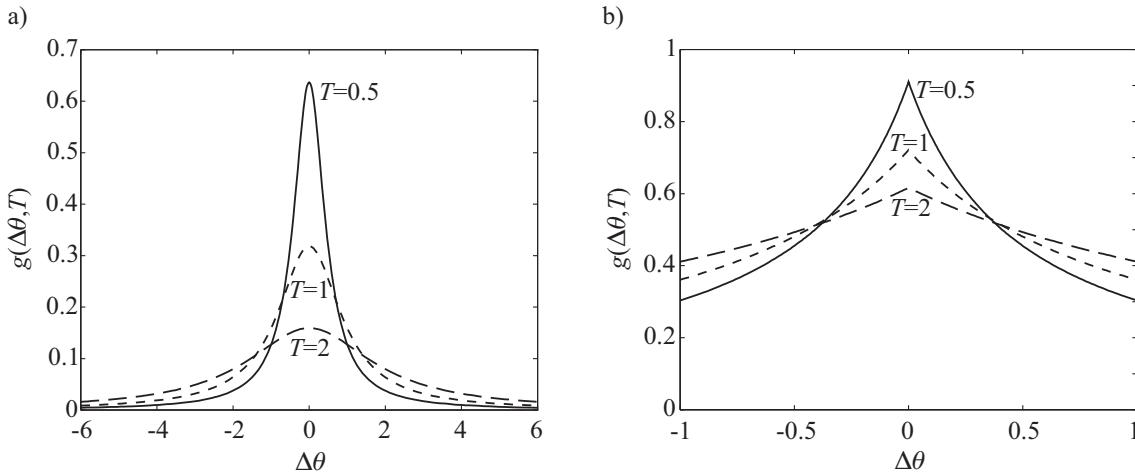
$$\Gamma(n + \frac{1}{2}) = \frac{(2n)!}{4^n n!} \pi^{\frac{1}{2}}.$$

V primeru dimenzije $d = 1$ dobimo poenostavljen zapis

$$g(\Delta\theta, T) = \frac{1}{\pi} \left(\frac{T}{T^2 + \Delta\theta^T \Delta\theta} \right). \quad (6.7)$$

V primeru optimizacije FA ima funkcija ohlajanja naslednjo obliko

$$T_k = \frac{T_0}{k}. \quad (6.8)$$



Slika 6.4. a) Cauchyeva porazdelitev za različne temperature ohlajanja T . Cauchyeva porazdelitev ima "širši rep" kot Gaussov kar prikazuje slika 6.3 zato dovoljuje boljše globalno iskanje optima. b) Funkcija gostote verjetnosti za VFSR in ASA je definirana na intervalu $[-1, 1]$ in jo lahko linearno transformiramo na poljuben interval $[A, B]$. Ta ima še "širši rep" kot Cauchyeva funkcija gostote verjetnosti (pdf) na sliki 6.4a, in zato dovoljuje lažje testiranje lokalnih minimumov pri iskanju globalnega minimuma.

V poznih 1980 je Ingber razvil še metodo, ki se imenuje zelo hitro simulirano ponovno ohljanje (*ang. Very Fast Simulated Reannealing - VFSR*) [48], ki je bila potem razširjena še v adaptivno simulirano ohljanje (*ang. adaptive simulated annealing - ASA*) - ASA) [49, 50]. V nasprotju z BA in FA, ki teoretično preiskujeta neskončen prostor parametrov z Gaussovo in Cauchyevim porazdelitvijo, sta ta dva algoritma primerena za *omejene in normirane* parameterje. Slika 6.4b kaže pdf funkcijo za izbiro parametrov v primeru ASA za interval med $[-1, 1]$. Pri tej modifikaciji so parametri linearno transformirani na interval med $[A, B]$. Ta pristop zadosti zahtevam večine realnih problemov. Modifikacija ASA je uporabljena v primeru velikega števila realnih problemov na različnih področjih, od fizike, medicine do financ. Kot lahko vidimo na sliki 6.4b izbrana pdf daje prednost globalnemu preiskovanju prostora še bolj kot to doseženo v primeru Cauchyeve porazdelitve. Zaradi tega je število spretetih točk manjše. Sprememba parametrov se izvede na osnovi funkcije porazdelitve gostote, ki je v primeru dimenzije $d = 1$ enaka

$$g(\Delta\theta, T) = \frac{1}{2(|\Delta\theta| + T) \ln(1 + \frac{1}{T})}, \quad (6.9)$$

če pa se srečamo s problemom optimizacije, kjer optimiramo d parametrov, potem dobimo funkcije porazdelitve gostote

$$g(\Delta\theta, T) = \prod_{i=1}^d \frac{1}{2(|\Delta\theta_i| + T_i) \ln\left(1 + \frac{1}{T_i}\right)}. \quad (6.10)$$

Statistična konvergenca k globalnemu minimumu je dokazana za primer funkcije eksponencialnega hitrega ohljanja, ki je definirana z

$$T_{ki} = T_0 \exp\left(-c_i k^{1/d}\right), \quad (6.11)$$

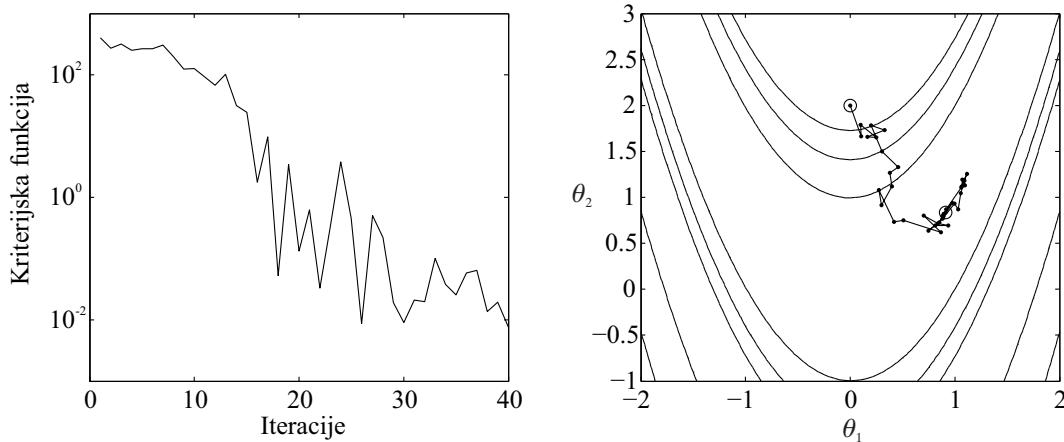
kjer je k število iteracij, d število parametrov in c_i konstanta, ki jo izberemo za vsak parameter, kar pomeni, da ima vsak parameter svojo funkcijo ohljanja. Člen $1/d$ v enačbi 6.11 je direktna posledica problema dimenzij. Naraščajoče število parametrov mora eksponencialno upočasnititi ohljanje, če želimo izpolniti pogoje za globalno konvergenco. Zaradi tega postane metoda ASA v primeru velikega števila parametrov zelo počasna. Sprememba funkcije ohljanja v $T_{ki} = T_0 \exp(-c_i k^{Q/d})$ pri $Q > 1$ se imenuje *kaljenje* (*ang. quenching*). Za primer lahko izberemo $Q = d$ in na ta način dosežemo hitrost algoritma, ki je neodvisna od števila parametrov. Ta izbira sicer krši pravilo za dokaz globalne konvergencije, vseeno pa dobro deluje v mnogih primerih [49].

Kadarkoli izvajamo optimizacijo v multidimenzionalnem prostoru parametrov, vedno naležimo na problem različne občutljivosti kriterijske funkcije na spremembe teh parametrov. Zato se zdi smiselno, da v primeru parametrov, ki ne vplivajo pomembno na kriterijsko funkcijo, korak spremembe teh parametrov povečamo. Če želimo spremnijati korak spremembe parametrov, moramo poznati njihovo občutljivost, to je gradient kriterijske funkcije, ki ga moramo na določen način oceniti. V tem primeru vsakih nekaj iteracij inicializiramo temperaturo ohljanja za vsak parameter posebej. Tako se metoda prilagaja hiperpovršini kriterijske funkcije. Druga značilnost ASA je samooptimizacija, ki pomeni, da lahko metoda sama optimira svoje nastavitevne parametre. Čeprav izgleda to zelo privlačno, zahtev zelo velik računski napor. V literaturi [51] je narejena primerjava med ASA in genetskim algoritmom (GA), ki izkazuje znatno boljše rezultate na številnih primerih.

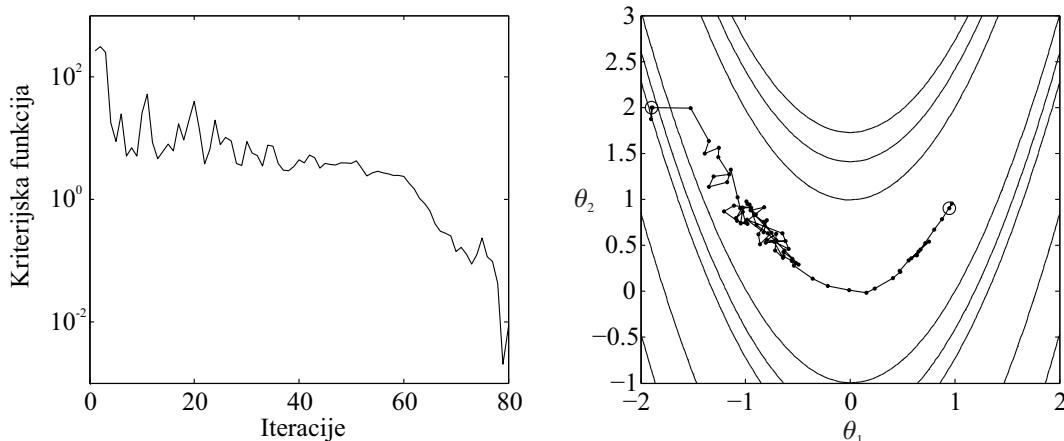
Primer 6.1.1. SA minimizacija funkcije

V tem primeru je Rosenbrockova funkcija minimizirana z uporabo SA optimizacije. V prvem

primeru, ko smo začeli iz prvega začetnega pogoja smo izvedli 40, v drugem pa 80 iteracij. Slika 6.5 a) kaže konvergenco logaritmične napake, b) pa potek najboljše ocene v parameterskem prostoru. Enako velja za sliko 6.6. Posebno pozornost je v tem primeru potrebno posvetiti številu poizkusov znotraj ene iteracije in nastavitev začetne temperature. Prav tako je zelo pomembna nastavitev naključne spremembe parametrov.



Slika 6.5. a) Primer iskanja minimuma Rosenbrockove funkcije pri začetnih pogojih $[0.0 \ 2.0]^T$ glede na iteracije,
b) v parameterskem prostoru.



Slika 6.6. a) Primer iskanja minimuma Rosenbrockove funkcije pri začetnih pogojih $[-1.9 \ 2.0]^T$ glede na iteracije,
b) v parameterskem prostoru.

6.2 Optimizacija z roji delcev (PSO)

Optimizacija z roji delcev (*ang. Particle Swarm Optimisation - PSO*) spada v družino stohastičnih optimizacijskih tehnik. Temelji na iskanju optimuma s pomočjo skupine n delcev, ki so kandidati za rešitev optimizacijskega problema. Delci so naključno porazdeljeni po prostoru parametrov, ki ima v tem primeru dimenzijo d . Dimenzija d pomeni seveda tudi število parametrov obravnavanega problema.

Delec lahko na ta način zapišemo kot točko v parameterskem prostoru $\underline{\theta}_i = [\theta_{i1}, \dots, \theta_{id}]$ in predstavlja kandidata za rešitev optimizacijskega problema. Delce inicializiramo naključno in potem spremojamo njihov položaj iterativno. Vsak delec predstavlja eno od možnih rešitev optimizacijskega problema.

Vsek delec ima svoj položaj $\underline{\theta}_i$ in hitrost $\Delta\underline{\theta}_i$. Za vsak delec določimo njegovo vrednost kriterijske funkcije, ki jo na pogosto imenujejo tudi funkcija kvalitete (*ang. fitness*). Relacija med kriterijsko funkcijo I in funkcijo kvalitete je $F = -I$. Če v primeru kriterijske funkcije običajno iščemo minimum, potem v primeru funkcije kvalitete iščemo maksimum.

Sprememba položaja vektorja parametrov (delca) je odvisna od vrednosti vektorja parametrov pri najvišji funkciji kvalitete za ta določen delec, ki jo imenujemo \underline{p}^{best} in vrednostjo vektorja parametrov z najvišjo funkcijo kvalitete v celotni populaciji (roju) delcev \underline{g}^{best} . Odvisnost spremembe položaja delca od \underline{p}^{best} imenujemo kognitivna komponenta, odvisnost spremembe od \underline{g}^{best} pa socialna komponenta

$$\Delta\underline{\theta}_i(k+1) = \omega\Delta\underline{\theta}_i(k) + c_1\underline{r}_p \circ (\underline{p}^{best} - \underline{\theta}_i(k)) + c_2\underline{r}_g \circ (\underline{g}^{best} - \underline{\theta}_i(k)), \quad i = 1, \dots, n. \quad (6.12)$$

kjer je ω konstanta vztrajnosti ali konstanta samozaupanja, c_1 je kognitivna konstanta in c_2 konstanta socialnega obnašanja ali zaupanja v roj in n je število delcev, operator \circ pa predstavlja množenje vektorjev po komponentah, $\underline{a} \circ \underline{b} = [a_1 \cdot b_1 \cdots a_n \cdot b_n]^T$. Vektorja \underline{r}_p in \underline{r}_g sta vektorja uniformno porazdeljenih naključnih števil na intervalu med $[-1, 1]$ z dimenzijo d . Naključnost, ki jo vnašata ta dva vektorja, prinaša raznovrstnost celotnemu roju delcev. Delci se gibljejo glede na svojo vztrajnost, glede na izkušnje najboljše rešitve celotnega roja in glede na osebno najboljšo rešitev. V smislu zagotavljanja konvergenco rešitve je hitrost premikanja delcev omejena $\Delta\theta_r^i \in [\Delta\theta_r^{\min}, \Delta\theta_r^{\max}]$, $r = 1, \dots, d$, $i = 1, \dots, n$, hkrati pa je omejen tudi položaj delcev $\theta_r^i \in [\theta_r^{\min}, \theta_r^{\max}]$, $r = 1, \dots, d$, $i = 1, \dots, n$.

Algoritem se zaključi, ko dosežemo maksimalno število iteracij, če so vsi delci blizu optimalne rešitve ali pa če ne dosežemo izboljšanja rešitve. Algoritem 6.2 prikazuje pseudokodo za optimizacijo z rojem delcev.

Primer 6.2.1. PSO minimizacija funkcije

V tem primeru je Rosenbrockova funkcija minimizirana z uporabo PSO optimizacije. V prvem primeru opazujemo potek kriterijske funkcije za prvi delec, ki ima začetno vrednost $[-2 \ 2]$ in v drugem primeru delec, ki začne pri vrednosti $[0 \ 2]$. PSO algoritem ima 25 delcev, ki so enakomerno porazdeljeni po dvodimenzionalni mreži s korakom 1. Izberemo začetne vrednosti parametrov na intervalu $\theta_1 \in [-2 \ 2]$ in $\theta_2 \in [-2 \ 2]$. V obeh primerih smo algoritem zaključili pri 50 iteracijah. Slika 6.7 a) kaže konvergenco logaritmične napake, b) pa potez delca z začetno vrednostjo $[-2 \ 2]$. V primeru slike 6.8 pa je prikazano enako za delec z začetno vrednostjo $[0 \ 2]$.

6.3 Evolucijski algoritmi (EA)

Evolucijski algoritmi (*ang. Evolutionary Algorithms - EA*) spadajo v družino stohastičnih optimizacijskih tehnik, ki temeljijo na posnemanju narave, tako kot simulirano ohlajanje (SA) in optimizacija z roji delcev (PSO). Evolucijski algoritmi posnemajo naravno evolucijo, ki je dosegla velik uspeh z razvojem novih vrst in njihovo adaptacijo na okolje v zelo spremenljivih okoljskih pogojih. Vsi evolucijski algoritmi temeljijo na *populaciji posameznikov*, kjer je vsak posameznik ena od rešitev optimizacijskega problema, t.j. ena točka v prostoru parametrov. Za razliko od SA, kjer navadno delamo z enim delcem, ki predstavlja en parameter, se pri metodi

Algoritem 2 Pseudo koda za optimizacijo z roji delcev (PSO)

- 1: Inicializacija števila delcev n , konstante inercije ω , kognitivne konstante c_1 in konstante socialnega obnašanja c_2 .
 - 2: Inicializacija začetnih vrednosti položaja $\underline{\theta}_i$, $i = 1, \dots, n$ in hitrosti delcev $\Delta\underline{\theta}_i$, $i = 1, \dots, n$
 - 3: Inicializacija $I(\underline{g}^{best}) >> 0$, $I(\underline{p}_i^{best}) >> 0$
 - 4: **repeat**
 - 5: nova iteracija $k = k + 1$
 - 6: Izračun kriterijske $I(\underline{\theta}_i)$ za vse delce, $i = 1, \dots, n$.
 - 7: Izračun globalno najboljšega delca $\underline{g}_{best} = argmin(I(\underline{g}^{best}), I(\underline{\theta}_1), \dots, I(\underline{\theta}_n))$
 - 8: Izračun osebne najboljše vrednosti delca $\underline{p}_i^{best} = argmin(I(\underline{p}_i^{best}), I(\underline{\theta}_i))$, $i = 1, \dots, n$.
 - 9: Izračun vektorjev naključnih vrednosti \underline{r}_p in \underline{r}_g z dimenzijo d .
 - 10: Izračun premika delcev (hitrosti)
$$\Delta\underline{\theta}_i = \omega\Delta\underline{\theta}_i + c_1\underline{r}_p \circ (\underline{p}_i^{best} - \underline{\theta}_i) + c_2\underline{r}_g \circ (\underline{g}^{best} - \underline{\theta}_i), i = 1, \dots, n$$
 - 11: Izračun novih vrednosti delcev
$$\underline{\theta}_i = \underline{\theta}_i + \Delta\underline{\theta}_i, i = 1, \dots, n$$
 - 12: Izračun omejitev položaja
$$\theta_r^i \in [\theta_r^{\min}, \theta_r^{\max}], r = 1, \dots, d, i = 1, \dots, n$$

in hitrosti delcev po njihovih komponentah

$$\Delta\theta_r^i \in [\Delta\theta_r^{\min}, \Delta\theta_r^{\max}], r = 1, \dots, d, i = 1, \dots, n$$
 - 13: **until** izpolnjen pogoj za zaključek algoritma.
-

GA in PSO pojavlja cela populacija oziroma roj posameznikov ali delcev. Roj delcev pomeni paralelno delovanje algoritma, ki ga zaradi tega lahko zelo pohitrimo.

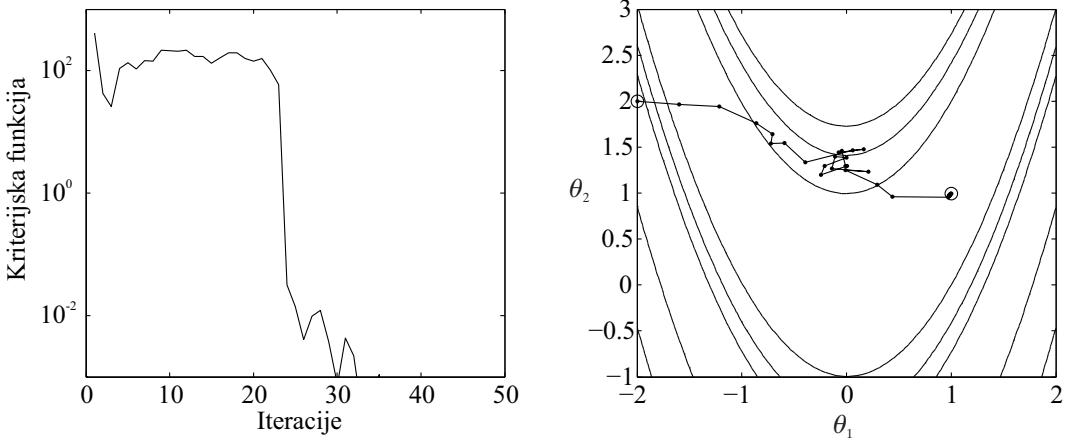
Tako kot v naravi se tudi umetna populacija iterativno razvija iz *generacije* v generacijo. To pomeni, da se posamezniki iz generacijo v generacijo spreminjajo, ker so potomci dveh posameznikov (*rekombinacija*). Spreminjajo se tudi zaradi *mutacij* in drugih genetskih operatorjev. S temi genetskimi operatorji se generirajo novi posamezniki, ki dejansko predstavljajo novo točko v parametrskem prostoru. Vsakemu novemu posamezniku izračunamo njegovo kriterijsko funkcijo. S *selekcijo* pa izberemo boljše posameznike, tiste z manjšo vrednostjo kriterijske funkcije.

Kriterijska funkcija se tudi tukaj običajno imenuje funkcija kvalitete in predstavlja inverz klasične funkcije cene, t.j., $1/I(\underline{\theta})$ ali $-I(\underline{\theta})$. Selekcija skrbi za to, da se populacija razvija v smeri boljših posameznikov, kar pomeni reševanje optimizacijskega problema.

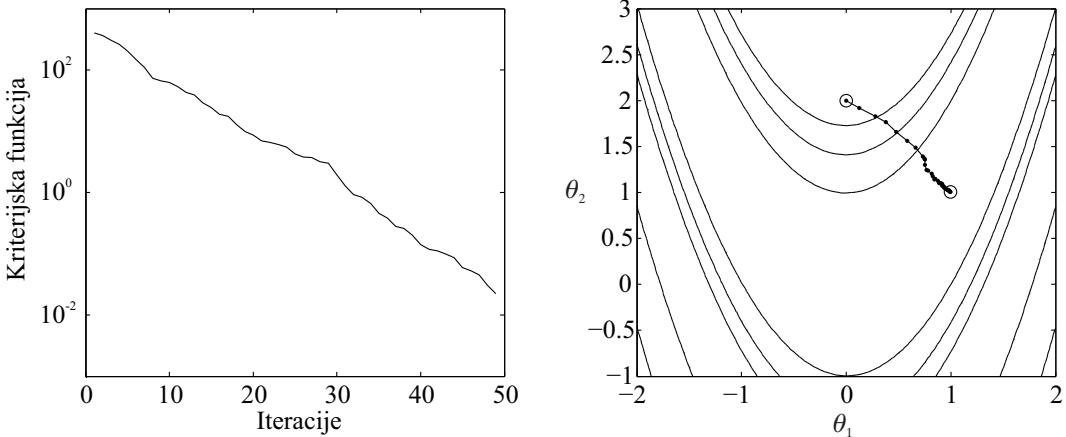
V tabeli 6.1 je primerjava terminologije evolucijskih algoritmov s klasičnimi optimizacijskimi termini. Delovanje evolucijskih algoritmov je predstavljeno s psevdokodo v algoritmu 6.3.

Dobra lastnost evolucijskih algoritmov je ta, da so po svoji naravi paralelni. Paralelni so zaradi množice osebkov, ki tvori populacijo in se generira v vsaki iteraciji. Populacija paralelnih delcev omogoča paralelizacijo algoritma na paralelnih procesorjih računalnika. Evolucijo v naravi lahko razlagamo tudi kot velikansko paralelno optimizacijsko eksperiment.

Evolucijski algoritmi se razlikujejo v tipih izbire in različnih genetskih operatorjih. Izbira strateških parametrov, kot so frekvence mutacije in rekombinacije je stvar načrtovalca, ali pa se optimira z EA samim, kar seveda spet loči algoritme med seboj. Velike razlike so tudi v kodiranju parametrov ali struktur, ki jih optimiramo.



Slika 6.7. a) Primer iskanja minimuma Rosenbrockove funkcije s PSO pri začetnih pogojih $[-2 \ 2.0]^T$ glede na iteracije, b) v parameterskem prostoru.



Slika 6.8. a) Primer iskanja minimuma Rosenbrockove funkcije s PSO pri začetnih pogojih $[0 \ 2.0]^T$ glede na iteracije, b) v parameterskem prostoru.

Algoritem 3 Pseudo koda evolucijskega algoritma - EAs

1: Naključna zbera začetne populacije S posameznikov (točke v parameterskem prostoru)

$$\underline{\theta}_0 = [\underline{\theta}_{1,0} \ \underline{\theta}_{2,0} \ \cdots \ \underline{\theta}_{S,0}]$$

2: Izračun kriterijske funkcije za začetno populacijo, oziroma funkcije kvalitete $F(\underline{\theta}_0)$

3: **repeat**

4: Nova iteracija $k = k + 1$

5: Izbira najprimernejših posameznikov $\underline{\theta}_{\text{new}} = \text{Select}(\underline{\theta}_{k-1})$

6: Genetske operacije (mutacija, rekombinacija, itd.) $\underline{\theta}_k = \text{GenOps}(\underline{\theta}_{\text{new}})$

7: **until** izpolnjen pogoj za zaključek algoritma

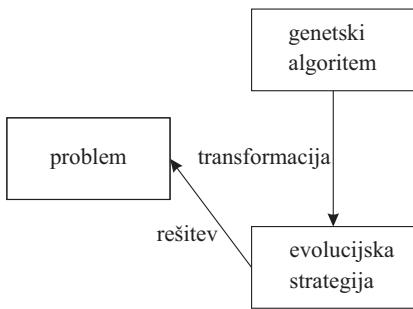
Evolucijske algoritme lahko razdelimo na *evolucijske strategije* (ES), *genetske algoritme* (GA), *genetsko programiranje* (GP) in *evolucijsko programiranje* (EP).

Medtem, ko so evolucijske strategije in genetski algoritmi največkrat uporabljeni za optimizacijo parametrov, se genetsko programiranje uporablja na višjih nivojih optimiranja drevesnih struktur. Na sliki 6.9 je primerjava med evolucijskimi strategijami in genetskimi algoritmi. Ge-

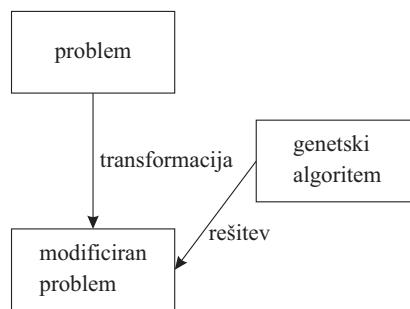
Tabela 6.1. Primerjava terminologije na področju evolucijskih algoritmov in klasične optimizacije.

Evolucijski algoritem	Klasična optimizacija
Posameznik (individuum)	Vektor parametrov
Populacija	Množica vektorjev parametrov
Funkcija kvalitete (<i>ang. fitness function</i>)	Inverzna vrednost funkcije cene
Kvaliteta (<i>ang. fitness</i>)	Inverz funkcije cene
Generacija	Iteracija
Vpliv evolucijskih operatorjev	Sprememba vektorja parametrov

a) evolucijska strategija (ES)



b) genetski algoritem (GA)

**Slika 6.9.** Primerjava med evolucijskimi strategijami in genetskimi algoritmi: a) genetski algoritem je modificiran v evolucijsko strategijo, ki lahko reši problem direktno, b) genetski algoritem lahko reši le modificiran problem, [80].

netski algoritmi delujejo na binarnem nivoju in so zelo podobni principom iz narave, kjer se informacija kodira s štirimi bazami v DNA. Če želimo rešiti realni optimizacijski problem, kjer se pojavljajo parametri realnih vrednosti, potem moramo te parametre zakodirati v binarni zapis. Genetski algoritem potem deluje na tem zapisu. V nasprotju s tem pa evolucijske strategije ne posnemajo narave tako natančno, temveč delujejo na parametrih, ki jih zapisemo z realnimi števili, kar omogoča, da rešimo optimizacijski problem bolj neposredno. Obstaja velika množica različnih modifikacij izvornih algoritmov.

V naslednjih poglavjih si bomo ogledali evolucijske strategije, genetske algoritme in genetsko programiranje. Več pa podaja literatura [7, 23, 35, 36, 42, 44, 80, 97, 106].

6.3.1 Evolucijska strategija (ES)

Glavna lastnost evolucijske strategije (ES) je obravnavava optimizacijskih parametrov v njihovi prvotni obliki. Zvezni parametri so obravnavani kot realna števila, celoštevilski parametri pa kot cela števila.

Pri ES je *mutacija* eden od dominantnih genetskih operatorjev. Parametri mutirajo glede na normalno Gaussovo distribucijo. V najenostavnnejšem primeru se vsi parametri spreminja po isti Gaussovi distribuciji (glej sliko 6.10a)

$$\theta_i^{(\text{new})} = \theta_i + \Delta\theta, \quad (6.13)$$

kjer je $\Delta\theta$ porazdeljena glede na naslednjo funkcijo gostote verjetnosti:

$$p(\Delta\theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2}\frac{\Delta\theta^2}{\sigma^2}\right). \quad (6.14)$$

Glede na enačbo 6.14 lahko ugotovimo, da imajo manjše mutacije večjo verjetnost kot večje. Standardna deviacija σ definira korak spremembe parametra $\Delta\theta$.

Če funkcija porazdelitve verjetnosti v enačbi 6.14 ni zadosti fleksibilna, lahko razširimo funkcijo na ta način, da dovolimo individualne velikosti korakov (standardne deviacije) za vsak parameter posebej (glej sliko 6.10 b)

$$\theta_i^{(\text{new})} = \theta_i + \Delta\theta_i, \quad (6.15)$$

kjer je $\Delta\theta_i$ porazdeljena glede na naslednjo funkcijo gostote verjetnosti

$$p(\Delta\theta_i) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left(-\frac{1}{2}\frac{\Delta\theta^2}{\sigma_i^2}\right). \quad (6.16)$$

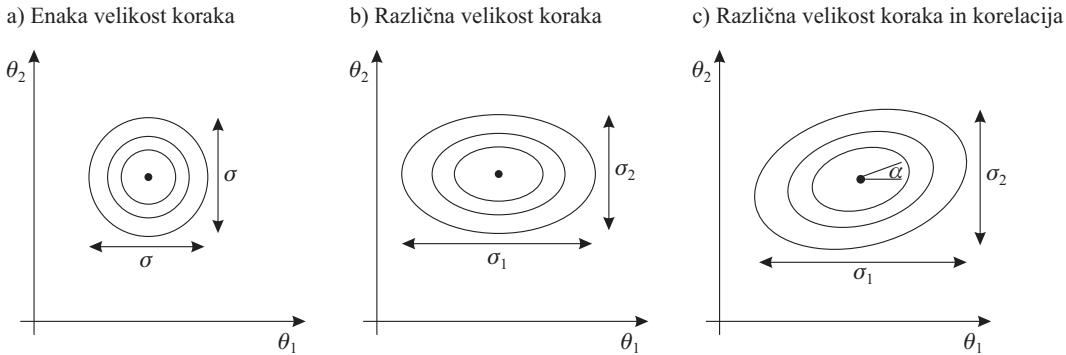
Ta interpretacija je najpogostejsa. Lahko pa v mutacije vnesemo tudi korelacijo med parametri. To dovoljuje smeri iskanja, ki niso paralelne osem parameterskega prostora. Tako dobimo skupno funkcijo gostote verjetnosti za vse različne korake sprememb parametrov $\underline{\Delta\theta} = [\Delta\theta_1 \Delta\theta_2 \dots \Delta\theta_n]^T$, ki je enaka (glej sliko 6.10 c)

$$p(\underline{\Delta\theta}) = \frac{1}{(2\pi)^{\frac{d}{2}} T} \exp\left(-\frac{1}{2}\underline{\Delta\theta}^T \underline{\Sigma}^{-1} \underline{\Delta\theta}\right), \quad (6.17)$$

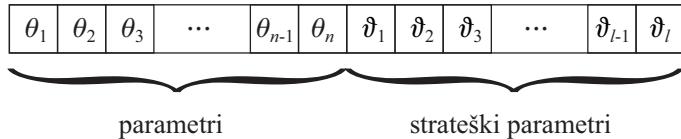
kjer je $\underline{\Sigma}$ kovariančna matrika, ki vsebuje informacijo o standardnih deviacijah posameznih parametrov in kovariankah med posameznimi parametri. Prejšnji dve porazdelitvi sta posebna primera porazdelitve v enačbi 6.17, kjer je $\underline{\Sigma} = \sigma^2 \underline{I}$ ali pa $\underline{\Sigma} = \text{diag}(\sigma_1^2, \sigma_2^2, \dots, \sigma_n^2)$.

Če parametri niso realni, ampak so celoštivilski ali binarni, potem moramo mutacijo prilagoditi tipu parametra. Celostivilske parameterje največkrat uporabimo pri optimizaciji strukture modela nevronske mreže, za optimizacijo števila nevronov, števila pravil v mehkem sistemu ali pa število koeficientov v primeru polinomskega modela. Za take primer korak $\underline{\Delta\theta}$ zaokrožimo, da dobimo celoštivilsko vrednost $\theta_i^{(\text{new})}$. Za binarne parametre pa uporabimo mutacije, ki ne temeljijo na verjetnosti. Te bomo srečali pri genetskih algoritmih.

Iz gornjega je jasno razvidno, da je uporabniku zelo težko določiti prave vrednosti standardnih deviacij, ki določajo korake sprememb parametrov. Korake sprememb parametrov je dodamo osebek in so na ta način ravno tako predmet optimizacije. Tipični osebek v primeru

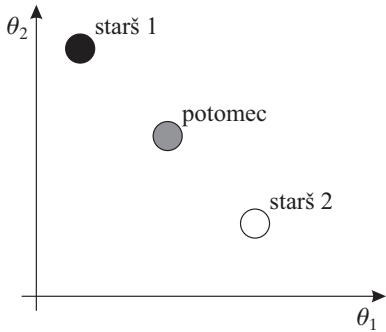


Slika 6.10. Mutacija v algoritmu evolucijske strategije temelji na normalni funkciji gostote verjetnosti: a) vsi parametri mutirajo neodvisno s skupno standardno deviacijo, b) vsi parametri mutirajo neodvisno vendar ima vsak svojo individualno standardno deviacijo, c) vsi parametri mutirajo z individualno standardno deviacijo in so korelirani (vsi elementi kovariančne matrike v Gaussovi distribuciji so različni od nič).

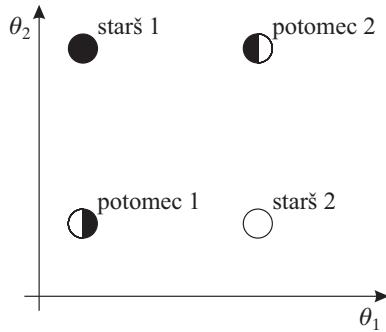


Slika 6.11. Osebek v primeru ES vsebuje poleg parametrov, ki jih optimiramo tudi strateške parametre (meta parametere), ki določajo interno obnašanje ES. Ti so ravno tako podvrženi optimizaciji.

a) rekombinacija s povprečenjem



b) diskretna rekombinacija



Slika 6.12. Rekombinacija v evolucijski strategiji se izvede kot a) povprečje ali b) diskretna kombinacija.

metode ES je prikazan na sliki 6.11, kjer z ϑ_i označimo *strateške parametere*, ki na primer lahko predstavljajo standardne deviacije. Na ta način dobimo n parameterov θ_i in dodatnih n individualnih strateških parametrov ϑ_i . Optimizacija strateških parametrov se imenuje *samoadaptacija* ali *drugi nivo učenja*. Mutacij strateških parametrov ne moremo realizirati po normalni distribuciji, ker morajo biti pozitivni.

Ideja je v tem, da so za dobro optimizacijo potrebni dobro prilagojeni koraki spremembe parametrov, ki so seveda odvisni od narave kriterijske funkcije, ki jo optimiramo (hiper površina, ki jo določa kriterijska funkcija).

Poleg mutacije igra pomembno vlogo tudi *rekombinacija*. Slika 6.12 kaže dva principa rekombinacije. Prva je *metoda povprečenja*, ki izračuna določeno povprečje dveh staršev in njegovo vrednost prenese na potomca. Druga metoda pa je *diskretna rekombinacija*, kjer potomec preuzeče določene parameterje po enem staršu, druge pa po drugem. Oba pristopa lahko razširimo na več staršev.

Selekcija je v primeru ES popolnoma deterministična. To pomeni, da so posamezniki razvrščeni po njihovi kriterijski (fitness) funkciji in izberemo najboljše. Poznamo dve različni varianti deterministične selekcije: $(\mu + \lambda)$ -strategija in (μ, λ) -strategija. To je običajno označevanje, kjer je μ število staršev v vsaki generaciji in λ število potomcev, ki so posledica mutacij in rekombinacij. V vsaki generaciji izberemo izmed vseh staršev in potomcev samo μ najboljših posameznikov. Večje vrednosti μ vodijo do bolj globalnega iskanja. Predlagano razmerje μ/λ je $1/7$ [72], kar predstavlja dober kompromis med lokalnim in globalnim iskanjem.

Pri strategiji $(\mu + \lambda)$ starši tekmujejo s svojimi potomci pri selekciji. Na ta način se informacija o najboljšem posamezniku ne more izgubiti, t.j., kriterijska funkcija monotono narašča (kvaliteta monotono narašča) iz generacije v generacijo. Ta lastnost je zaželjena v primeru hitrega iskanja lokalnega optimuma. Se pa celotna populacija zbore okoli dobrega posameznika, če je na začetku izbran. Strategijo izberemo skupaj s principom samoadaptacije [42].

Za bolj globalno iskanje pa je bolj primerna strategija (μ, λ) . V tem primeru je selekcija izvedena samo na potomcih. V tem primeru žrtvujemo hitrost konvergencije na račun širšega

Tabela 6.2. Primerjava med evolucijsko strategijo in genetskim algoritmom.

	Evolucijska strategija	Genetski algoritem
Kodiranje	Realne vrednosti (problemsko orientirano)	Binarno, permutacijsko, z realnimi števili
Mutacije	Pomembne	Zanemarljive
Rekombinacija	Zanemarljiva	Pomembna
Selekcija	Deterministična	Naključna
Strateški parametri	Adaptivni	Konstantni

preiskovanja prostora. V tem primeru korak optimizacije konvergira k zelo majhnim vrednostim, če se bližamo lokalnemu optimumu, kar upočasni optimizacijo.

Optimalno število staršev in potomcev je zelo odvisno od primera. Pri bolj kompleksnih problemih je potrebno izbrati večje populacije in več generacij. Tipične izbire so okoli 100; ali (15, 100)-strategija. So pa algoritmi običajno neobčutljivi na te izbire in je približno nastavljanje teh vrednosti zadostno.

Tabla 6.2 povzema glavne značilnosti evolucijskih strategij in genetskih algoritmov, ki so razloženi v nadaljevanju.

Algoritem 4 Pseudo koda za evolucijsko strategijo $\mu + \lambda$ - ES

- 1: Inicializacija parametrov λ, μ in σ in maksimalnega števila generacij k_{max} .
 - 2: Naključna zbera začetne populacije μ posameznikov (točke v parameterskem prostoru) za določitev začetne populacije $gen = 0$

$$\theta_i = \mathcal{N}(0, \sigma^2), i = 1, \dots, \mu$$
 - 3: Izračun kriterijske funkcije $I(\theta_i)$, $i = 1, \dots, \mu$
 - 4: **repeat**
 - 5: Nova generacija $k = k + 1$
 - 6: Rekombinacija λ potomcev $\theta_j^{new} = (\theta_a + \theta_b)/2$, $j = 1, \dots, \lambda$
 - 7: Mutacija potomcev $\theta_j^{new} = \theta_j^{new} + \mathcal{N}(0, \sigma^2)$, $j = 1, \dots, \lambda$
 - 8: Izračun kriterijske funkcije skupino potomcev $I(\theta_j^{new})$, $j = 1, \dots, \lambda$
 - 9: Selekcija najboljših μ posameznikov iz množice $\mu + \lambda$ posameznikov glede na kriterijsko funkcijo
 - 10: **until** izpolnjen pogoj za zaključek algoritma
 - 11: Izbor najboljšega posameznika kot rezultat algoritma
-

6.3.2 Genetski algoritmi (GA)

Genetski algoritmi so bili razviti neodvisno in skoraj sočasno z evolucijskimi strategijami [44]. Parametre v primeru metode GA imenujemo $geni$, ti pa sestavljajo *kromosom*, ki je vektor parametrov. Za razliko od evolucijskih strategij se pri GA vse različne tipe parametrov (realne, celoštevilske, binarne) zapiše v binarni obliki. Kromosom je definiran kot binarni zapis, ki je prikazan na sliki 6.13. Dolžina bitnega zapisa je odvisna od uporabnika in z njeno dolžino se povečuje tudi natančnost zapisa, ki se podvoji z vsakim dodanim bitom. S številom bitov pa se povečuje kompleksnost optimizacije in zato težimo k krajšim zapisom, če je to mogoče.

Kodiranje. Kodiranje z binarnim zapisom je povzeto po analogiji z naravo, kjer je genski zapis shranjen s 4 simboli "A", "G", "C", "T", ki so bazni zapisi za DNA. Binarni zapis z "0" in "1" je poenostavitev tega. Binarno kodiranje rezultira v velikih zapisih (kromosomih), ja pa zelo enostavno. Če poznamo spodnjo in zgornjo mejo parametra, potem lahko interval razdelimo znotraj celotnega binarnega zapisa.

Primer 6.3.1. Primer binarnega kodiranja Predpostavimo, da imamo parameter $\theta = 0.95$, ki ima spodnjo mejo $\underline{\theta} = -1$ in zgornjo mejo $\bar{\theta} = 2$. Njegovo vrednost kodiramo z 12 bitnim binarnim zapisom $b = 12$. Dolžina binarnega zapisa b določa 2^b različnih nivojev vrednosti in $2^b - 1$ intervalov med spodnjo in zgornjo vrednostjo (kvantov). Najmanjši interval med dvema nivojema je interval ločljivosti in je enak

$$\Delta\theta = (\bar{\theta} - \underline{\theta}) / (2^b - 1).$$

Razliko med realnim parametrom θ in njegovo spodnjo mejo $\underline{\theta}$ lahko na ta način zapišemo s najbližjim naravnim številom intervalov ločljivosti $N \in \mathcal{N}$ in ostankom δ

$$\theta - \underline{\theta} = N \cdot \Delta\theta + \delta, \quad (6.18)$$

kjer je

$$N = \text{round} \left(\frac{\theta - \underline{\theta}}{\Delta\theta} \right). \quad (6.19)$$

V našem primeru je interval ločljivosti enaka $\Delta\theta = 7.3260 \cdot 10^{-4}$. Iz tega sledi, da je $N = 2662$ in $\text{delta} = 2 \cdot 10^{-4}$. Vrednost δ zanemarimo in je napaka pri kvantizaciji. Vrednost števila intervalov ločljivosti pa sedaj lahko zapišemo v binarni obliki

$$2662 = 1 \cdot 2^{11} + 1 \cdot 2^9 + 1 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^2 + 1 \cdot 2^1$$

Dobimo binarni zapis realnega parametra $\theta_{bin} = 101001100110$.

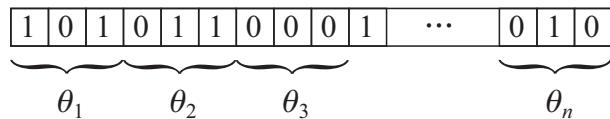
Za pretvorbo binarnega zapisa parametra z znano spodnjo in zgornjo vrednostjo v realno vrednost uporabimo naslednjo enačbo

$$\theta = (\bar{\theta} - \underline{\theta}) \frac{\sum_{i=0}^{b-1} a_i 2^i}{2^b - 1} + \underline{\theta}, \quad (6.20)$$

kjer so a_i , $i = 0, \dots, b-1$ vrednosti digitov v binarnem zapisu po naraščajočih potencah. Vrednost binarnega zapisa je v našem primeru enaka

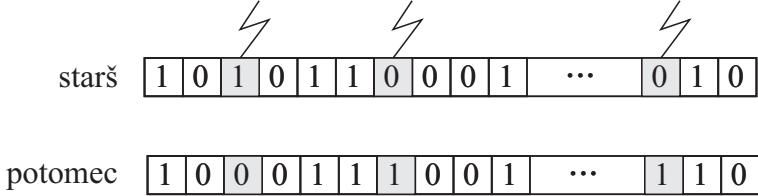
$$\theta = \frac{(2 - (-1))}{2^{12} - 1} (2^{11} + 2^9 + 2^6 + 2^5 + 2^2 + 2^1) + (-1) = 0.9502. \quad (6.21)$$

V primeru binarnega kodiranja zapišemo spodnjo in zgornjo mejo parametra z vrednostima "00...0" in "11...1". V primeru dobro znanih parametrov je to prednost. V primeru neznanih vrednosti meja parametrov pa to povzroča resne probleme, saj moramo meje določiti, kar pa ni vedno enostavno. Posebno ne takrat, ko parametri nimajo jasne interpretacije. V tem primeru je edina rešitve v zadosti velikem intervalu, ki pa na drugi strani zahteva večji bitni zapis ob enaki resoluciji parametrov ali pa se zniža resolucija, če ohranimo enak bitni zapis. Na eni strani se s povečanjem bitnega zapisa podaljša čas računanja, manjša resolucija pa vodi do slabših rezultatov optimizacije. Več o kodiranju lahko najdete v literaturi [23, 35].



Slika 6.13. Kromosom vsebuje parametere, ki jih optimiramo v binarnem zapisu.

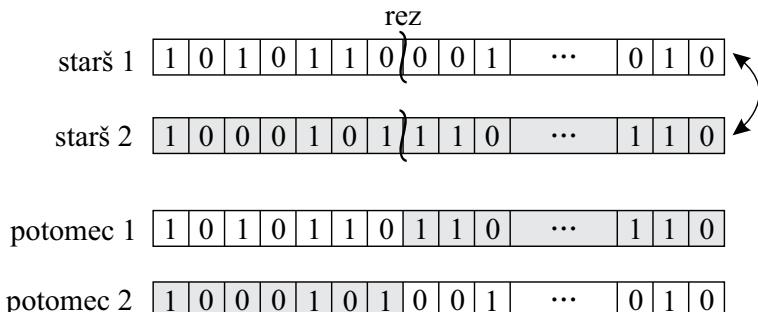
Mutacija. Mutacija pri metodi GA ne igra tako pomembne vloge kot pri ES. Slika 6.14 prikazuje operacijo mutacije pri GA. Mutacija enostavno invertira bit v zapisu. Stopnja mutacije, ki je običajno okoli 0.001 do 0.01, določa verjetnost stopnje mutacije za vsak bit zapisa posameznega kromosoma (bitni zapis posameznega vektorja parametrov).



Slika 6.14. Mutacija v genetskih algoritmih invertira bit v binarnem zapisu kromosoma.

Mutacija ima določene pomanjkljivosti, saj je stopnja verjetnosti za vse bite enaka, njihova pomembnost pa se močno spreminja glede na mesto bita v kodnem zapisu. Celoštevilske vrednosti med 0 in 15 lahko kodiramo z zapisom "0000" (=0) do "1111" (=15). Mutacija na evem bitu povzroči spremembo za +8 ali -8, mutacija na desnem bitu pa samo +1 ali -1. Obstajajo metode, ki stopnjo mutacije prilagodijo pomembnosti bita v zapisu.

Rekombinacija. Rekombinacija je glavni operator pri metodi GA. Binarni zapis (kromosom) dveh (ali več) staršev se razdeli na dva (ali več) dela in deli teh kromosomov se križno sestavijo; glej sliko 6.15. Zato se ta tip rekombinacije imenuje križanje (*ang. crossover*). Naslednji zapis vsebuje del zapisa enega in drugega starša, kot je to prikazano na sliki 6.12. Stopnja verjetnosti za križanje je tipično med 0.6 in 0.8. Mesto, kjer se razdeli zapis staršev je določeno naključno. S to omejitvijo, da so delitve možne samo na mestih kromosoma (vektorja parametrov), kjer se sestavljajo parametri in ne znotraj zapisa enega parametra.

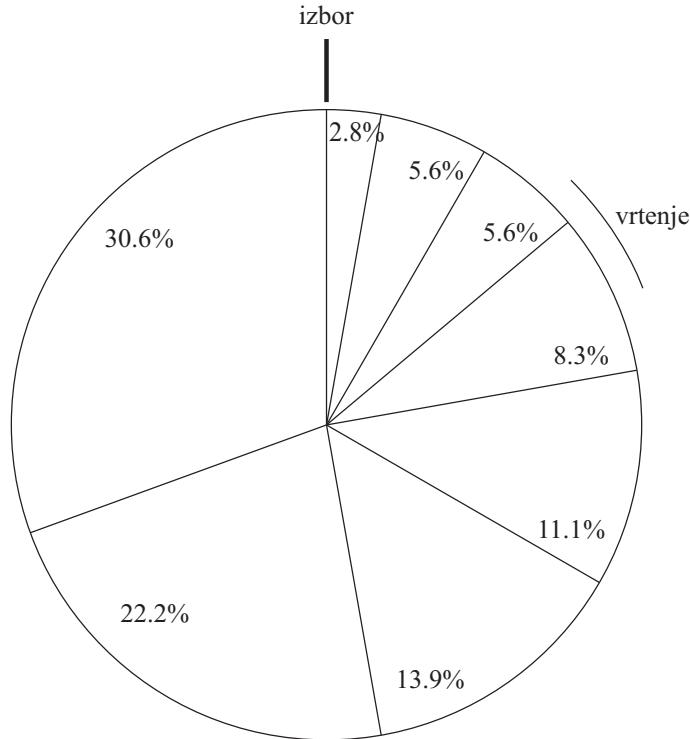


Slika 6.15. Rekombinacija v genetskih algoritmih se imenuje križanje, ker se kodni zapisi staršev razdelijo v eni ali več točkah in se medsebojno rekombinirajo.

Selekcija. Selekcija se v primeru metode GA izvaja naključno, v nasprotju z deterministično selekcijo pri ES. To pomeni, da ni nujno, da posameznik preživi, čeprav ima zelo dobro kriterijsko funkcijo. To je v skladu s posnemanjem narave. Najbolj običajen je postopek izbire z *ruletnim kolesom*. Vsak posameznik ima možnost preživetja, ki je sorazmerna z njegovo funkcijo kvalitete F_i , $i = 1, \dots, n$. Slika 6.16 kaže princip izbire z ruleto. Verjetnost izbire za vsakega posameznika $P_i^{(\text{select})}$ izračunamo z normirano (pozitivno) fitness vrednostjo

$$P_i^{(\text{select})} = \frac{F_i}{\sum_{j=1}^n F_j}, \quad (6.22)$$

kjer je n velikost populacije.



Slika 6.16. Pri izbiri z ruletnim kolesom ima vsak posameznik verjetnost preživetja, ki je sorazmerna njegovi funkciji kvalitete. Na sliki je primer 8 posameznikov.

Funkcija kvalitete mora biti pozitivna, da lahko izračunamo verjetnost preživetja po enačbi 6.22. To lahko storimo tako, da invertiramo kriterijsko funkcijo $1/I(\underline{\theta})$ ali pa invertiramo predznak kriterijske funkcije tako, da dobimo $C - I(\underline{\theta})$, kjer je C večja kot največja vrednost kriterijske funkcije. Problem rulete je v tem, da so na začetku verjetnosti za preživetje zelo različne, s časom pa konvergirajo k enakim vrednostim in na ta način je pritisk izbire vse manjši, kot to prikazuje slika 6.17. Za rešitev tega problema je predlagano nekaj rešitev. Ena od zelo učinkovitih je ta, da preslikamo najboljšo vrednost v določeno maksimalno vrednost, najslabšo pa v 0. Na ta način je pritisk selekcije bolj enakovraven porazdeljen po vseh generacijah [36].

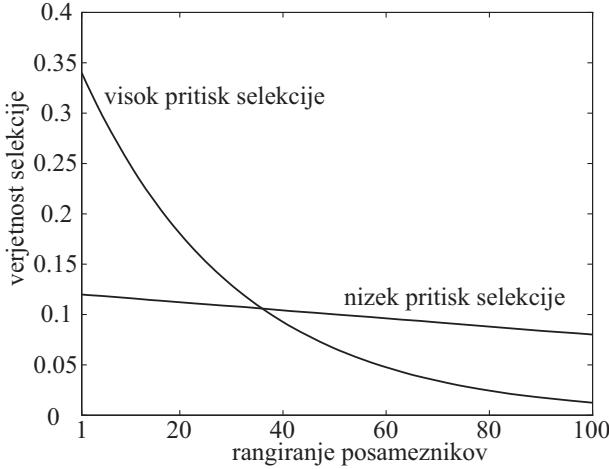
Slika 6.17 prikazuje, da se za zelo visok pritisk izbire različnost populacije zmanjšuje, ker le najboljši posamezniki preživijo. To vodi v prehitro konvergenco (*ang. premature convergence*), to je, GA izgubi sposobnost globalnega preiskovanja.

Alternativna metoda selekcije *turnirska izbira*. Iz množice vseh možnih posameznikov izberemo dva in glede na njuni kriterijski funkciji, boljšega izberemo kot enega od staršev. Enako storimo tudi za drugega starša [36].

Izbira števila staršev in potomcev je pri GA zelo problematična in odvisna od primera. Tipične vrednosti so okoli 100. Tabla 6.2 daje pregled lastnosti GA in ES. Pametno kodiranje parametrov pri GA je najpomembnejši korak od dobre rešitve kompleksnih problemov.

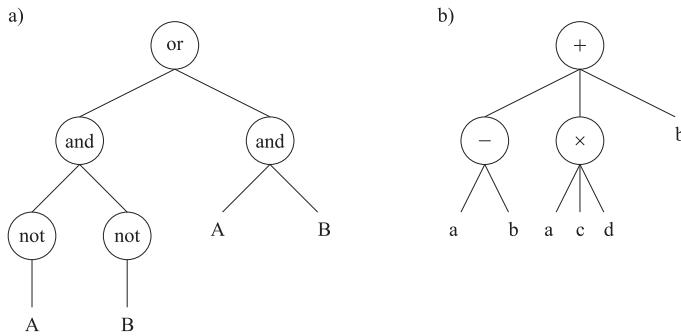
6.3.3 Genetsko programiranje (GP)

Genetsko programiranje je predlagal Koza [68, 69, 70] in je zelo podobno genetskim algoritmom, vendar pa deluje na drevesnih strukturah in ne na binarnih zapisih. Drevesa so spremenljive



Slika 6.17. Deset posameznikov izberemo iz množice 100. Pritisk selekcije je velik, če so boljši posamezniki favorizirani pred slabimi. Najnižji pritisk izbere bi bil, če bi imeli vsi posamezniki enako možnost preživetja. Na ta način GA degenerira v naključno iskanje.

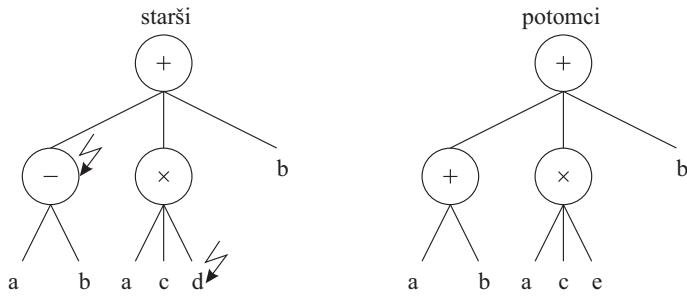
strukture, ki dovoljujejo da lahko zapišemo določene relacije zelo učinkovito. Slika 6.18 prikazuje dva primera kako lahko logične izraze in matematične enačbe zapišemo z drevesi. Listi drevesa običajno predstavljajo spremenljivke in konstante, ostale povezave pa so operatorji. Uporabnik določi nabor spremenljivk, konstant in operatorjev, ki bodo uporabljeni v drevesu.



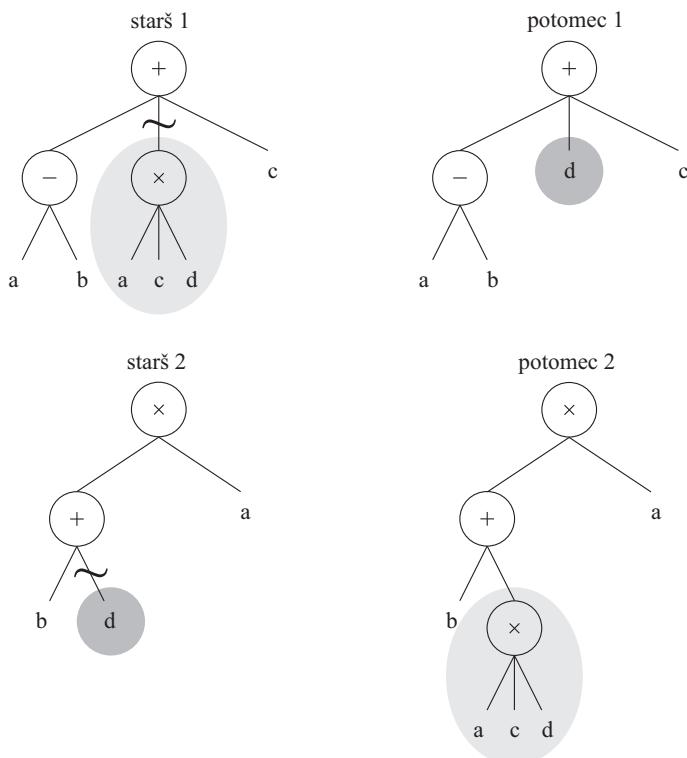
Slika 6.18. Osebek je v primeru genetskega programiranja predstavljen z drevesno strukturo. Drevesna struktura omogoča izvedbo a) logičnih operacij (tukaj: XOR), b) matematičnih enačb (tukaj imamo operacijo $a + acd$ realizirano na redundantni način), in mnogo drugih struktur.

Mutacijo in križanje kot ga poznamo pri metodi GA, je potrebno za uporabo v metodi GP spremeniti. Slika 6.19 prikazuje kako lahko izvedemo mutacijo. Deli drevesa (spremenljivke, konstante in operatorji) so lahko naključno spremenjeni, izbrisani ali na novo generirani. Slika 6.20 prikazuje primer križanja v primeru GP. Deli drevesa pri obeh starših so odrezani in zamenjani tako, da tvorijo novega potomca.

Ena od pomembnih stvari je tudi kako ravnati z realnimi parametri. Če imamo, na primer, enačbo $y = 2u_1 + 3.5u_2^3 + 1.75$, ki jo je potrebno poiskati z metodo GP, potem je potrebno poiskati pravilni rezultat in tudi numerične vrednosti 2, 3.5 in 1.75. En od možnih načinov je zapisu parametrov v bitni obliki, tako kot pri GA.



Slika 6.19. Mutacija v genetskem programiraju zamenjuje dele drevesne strukture z novimi spremenljivkami in operatorji.

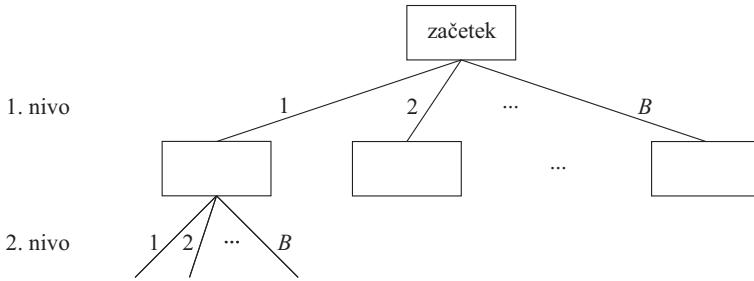


Slika 6.20. Rekombinacija v genetskem programiranju zamenjuje dele drevesa med starši.

6.4 Metoda vejenja in omejevanja (B&B)

Metoda vejenja in omejevanja (*ang. Branch and bound - B&B*) je drevesna metoda iskanja optimuma kriterijske funkcije in je zelo popularna pri reševanju kombinacijskih problemov. Z določenimi omejitvami je uporabna tudi za optimizacijo parametrov. V tem primeru morajo biti parametri diskretni ali pa jih diskretiziramo v določeno dolžino binarnega zapisa. Ideja B&B metode je v gradnji drevesa, ki vsebuje vse možne kombinacije diskretnih parametrov (glej sliko 6.21), in v preiskovanju drevesa. Metoda glede na izračun kriterijske funkcije v vsakem vozlišču drevesa določa nadaljnjo pot in eliminira dele drevesa, ki ne vodijo v optimalno smer.

V primeru metode B&B je potebno imeti informacijo o spodnji meji kriterijske funkcije v obravnavani veji, od obravnavanega vozlišča naprej. To informacijo izkoristimo za eliminacijo dela drevesa (*ang. pruning*) za katerega vemo, da rezultira v slabši kriterijski funkciji kot je trenutna optimalna. Enostavno lahko tudi vnesemo druge omejitve, ki jih lahko na vsakem vozlišču preverimo in eliminiramo del drevesa, če so omejitve kršene.



Slika 6.21. Ponazoritev B&B metode v primeru treh parametrov, ki imajo vsak po B diskretnih vrednosti.

V primeru praktičnih aplikacij se lahko odločimo za številne različne pristope, kjer gremo z iskanjem zelo globoko po drevesu, ali pa sledimo najboljšemu vozlišču. Z dodatnimi metodami krčenja drevesa pa se seveda izgubi lastnost globalnega preiskovanja drevesa.

Problem razširitve B&B na optimizacijo parametrov realnih vrednosti je v tem, da ne poznamo vpliva posameznih parametrov na kriterijsko funkcijo. Na primer, poglejmo primer optimizacije treh parametrov, ki so diskretizirani vsak na B vrednosti. Drevesna struktura ima v tem primeru 3 nivoje in B^3 končnih vozlišč (listov). Ta struktura predstavlja celotni model. Če želimo uporabiti metodo B&B, nastane problem, ker kriterijske funkcije ne moremo izračunati po prvi ali drugi veji. Vrednosti ne moremo izračunati, ker se zadnji parameter pojavi šele po drugi veji. Ta parameter pa ima lahko zelo pomemben vpliv na kriterijsko funkcijo. Na ta način ni mogoče *krčiti drevesa* in postane metoda neučinkovita. Več o metodi lahko najdete v literaturi [17, 73, 83].

Primer 6.4.1. Primer optimizacije izdelave izdelka z B&B optimizacijo

Izdelek se obdelva v štirih fazah (1, 2, 3 in 4). Na voljo so štiri naprave (A,B,C in D), ki pa vsaka za izvedbo določene faze porabi različno časa. Vsaka naprava se lahko uporabi samo za eno fazo v celotnem postopku izdelave. Časi posameznih naprav za določeno fazo so prikazni v tabeli 6.3. Za začetek B&B optimizacije je potrebno določiti spodnjo mejo časa izdelave (LB), ki je enaka $LB_0 = 2 + 3 + 1 + 4 = 10$, ki pa ni nujno izvedljiva. Iz začetne točke ($LB_0 = 10$) razvezjimo drevo v 4 veje, za vsako napravo po eno. Ko predvidimo za prvo fazo napravo A, najprej eliminiramo prvi stolpec in prvo vrstico v tabeli. Pogledamo kakšna je minimalna vsota v ostalih treh stolpcih in vrsticah s tem, da ni nujno izvedljiva in dobimo $LB_1 = 9+3+1+4 = 17$. Enako ponovimo za drugi stolpec in napravo B. Dobimo $LB_2 = 2 + 3 + 1 + 4 = 10$, za napravo C dobimo $LB_3 = 7 + 4 + 5 + 4 = 20$, za napravo D pa $LB_4 = 8 + 3 + 1 + 6 = 18$. Najnižja spodnja meja je $LB_2 = 10$ in zato v tem vozlišču nadaljujemo vejenje. Pomeni, da imamo v tem primeru za fazo 1 določeno napravo B, za drugo fazo pa iz tega vozlišča napravimo 3 veje, prvo za napravo A, drugo za napravo C in tretjo za napravo D. Prva veja ima $LB_5 = 2+6+1+4 = 13$, druga veja $LB_6 = 2 + 3 + 5 + 4 = 14$ in tretja veja $LB_7 = 2 + 7 + 1 + 7 = 17$. Vidimo, da je najboljša rešitev $LB_5 = 13$, ki gre preko naprave B na napravo A. Iz vozlišča LB_5 vejimo naprej na dve veji. Pva veja ima za fazo 3 izbrano napravo C in za fazo 4 napravo D, kar nam da $LB_8 = 2 + 6 + 1 + 4 = 13$, druga veja pa obratno izbiro, ki rezultira v $LB_9 = 2 + 6 + 8 + 9 = 25$. Vidimo torej, da je optimalna rešitev izbora naprav po fazah naslednja: $B \rightarrow A \rightarrow C \rightarrow D$.

6.5 Optimizacija s tabu področji tabuja (TS)

Optimizacija s tabu področji (ang. *Tabu Search* - TS) je zelo obetajoča nova metoda kot alternativa uveljavljenim metodam simuliranega ohlajanja in evolucijskim algoritmom. Metoda je zelo

Tabela 6.3. Časi obdelav za različne faze obdelave in različne naprave

	Naprava A	Naprava B	Naprava C	Naprava D
Faza 1	9	2	7	8
Faza 2	6	4	3	7
Faza 3	5	8	1	8
Faza 4	7	6	9	4

uprabna pri reševanju kombinacijskih problemov in operacijskih problemih (problem trgovskega potnika, iskanje najboljših poti, optimalni urniki, itd.). Glavna ideja pri TS je v kombinaciji uspešnih lokalnih optimizacij v globalno optimizacijsko strategijo. Ena od enostavnejših verzij je večkratno zaporedno zaganjanje lokalnih optimizacijskih metod iz različnih naključno izbranih začetnih vrednosti, kjer si TS algoritom zapomni že preiskane točke in se jim v nadaljevanju izogiba (že preiskane točke so tabu). V smislu globalnega preiskovanja prostora je nujno dopustiti, da algoritom naredi tudi korak v smeri k slabši rešitvi, saj se le na ta način lahko izogne lokalnim optimumom. Pri SA in EA je odločitev za sprejem novega vektorja parametrov naključna, v primeru TS pa se korak nazaj dovoli samo v primeru, ko se ne pričakuje boljšega lokalnega rezultata v naslednji lokaciji in če v tej lokaciji še nismo bili.

6.6 Povzetek

Nelinearne globalne metode optimizacije so primerne takrat, ko problema ne moremo rešiti z nelinearno lokalno metodo. Tovrstni problemi izkazujejo vsaj eno od naslednjih lastnosti: so večmodalni, nevezni odvodi kriterijske funkcije, nevezni parametri, problemi z optimizacijo strukture. Na račun globalne optimizacije se zmanjša hitrost konvergence metode. Pri nobeni od metod ne moremo zagotoviti končnega časa pri iskanju globalnega optimuma. Vendar je v mnogih praktičnih primerih določena približna optimalna rešitev zadovoljiva. Zelo priporočljivo je globalne metode kombinirati z nelinearnimi lokalnimi ali linearimi optimizacijami. Zelo težko je tudi predvideti katera globalna optimizacija bo najboljša za določen razred problemov. Za zahtevne aplikacije zahtevajo globalne metode veliko več izkušenj in poizkušanja, kot ostale klasične optimizacijske metode. Poznavanje problema pa je ena od najpomembnejših stvari pri uporabi teh metod, saj ne izboljša samo rezultatov in zmanjša časa reševanja problema, ampak je pogosto tudi pogoj za izvedljivosti rešitve [23, 80].

7. Metode nенадзорованega učenja

Ločimo tri najpomembnejše vrste učenja, ki so naslednje: nadzorovano učenje (*ang. supervised learning*), nенадзорovano učenje (*ang. unsupervised learning*) in spodbujevano učenje (tudi učenje s povratnim ojačenjem) (*ang. reinforcement learning*).

Nadzorovano učenje. Pri nadzorovanem učenju iščemo preslikavo f v danem naboru funkcij na osnovi množic podatkov $\underline{u} \in U$, $y \in Y$. Funkcija f preslika vhodne podatke v izhodne, $f : \underline{u} \rightarrow y$. Dejanske preslikave ne poznamo, zato iščemo aproksimacijo funkcije, ki optimalno preslika vhodne podatke v izhodne v smislu določene kriterijske funkcije. V kriterijski funkciji je vključeno odstopanje med preslikanimi podatki in ustreznimi izhodnimi vzorci.

Splošno uporabljeni kriterijski funkcija v primeru nadzorovanega učenja temelji na kvadratih pogreškov (*ang. mean square error*) med izhodi funkcije preslikave $f(\underline{u}(k))$ in pripadajočimi izhodnimi vzorci $y(k)$, na celotni množici vzorcev.

Metode nadzorovanega učenja srečamo na področju razpoznavanja vzorcev, govorimo o razvrščanju in identifikaciji, kjer govorimo o regresiji, oziroma aproksimaciji funkcijskih preslikav. Uporabljajo se tudi na področju razpoznavanja govora.

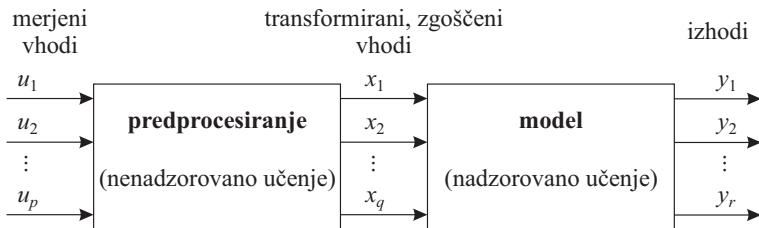
Nenadzorovano učenje. Nenadzorovano učenje se od nadzorovanega učenja razlikuje po tem, da nimamo definiranega apriornega izhoda, temveč so vsi podatki obravnavani kot vhodni. Tehnike nенадзорованega učenja se uporabljajo predvsem pri razvrščanju podatkov v roje. Rezultat nенадзорованega učenja je model v obliki porazdelitve podatkov po prostoru problema. Metoda je uporabna predvsem pri predprocesiraju podatkov, preden uporabimo eno od metod nadzorovanega učenja. Nenadzorovano učenje je učinkovito tudi v primeru stiskanja podatkov (*ang. data compression*). Vsi algoritmi stiskanja podatkov implicitno ali eksplizitno temeljijo na verjetnostni porazdelitvi po prostoru, ki ga določajo vhodne spremenljivke.

Področje uporabe nенадзорovanih metod učenja je na splošno pri uporabi metod identifikacije, rojenju, razvrščanju podatkov, stiskanju podatkov in filtriranju.

Spodbujevano učenje. Spodbujevano učenje temelji na informaciji o končnem izidu opazovanega poizkusa. Značilen primer te vrste učenja so igre (šah, karte), kjer ni možno ocenjevati vsake poteze posebej. Glede na končni rezultat igre (zmaga, poraz, neodločeno) sklepamo o kvaliteti izbrane strategije. Katerikoli primer nadzorovanega učenja lahko umetno transformiramo v problem spodbujevanega učenja z eliminacijo določene informacije. Primer balansiranja palice v navpični legi lahko obravnavamo kot nadzorovan primer učenja, če upoštevamo informacijo o napaki med željeno navpično pozicijo in dejansko pozicijo palice. Na drugi strani pa problem postane problem spodbujevanega učenja, če upoštevamo pri učenju samo informacijo o tem ali je bil poizkus uspešen ali ne. Seveda je vedno prednost, če uporabimo vso informacijo, ki je na voljo. Metode spodbujevanega učenja so zanimive na področjih učenja strategij na dolgi rok, kjer nimamo eksplizitno določenega želenega izhoda in posledično spremenljivke napake v vsakem koraku.

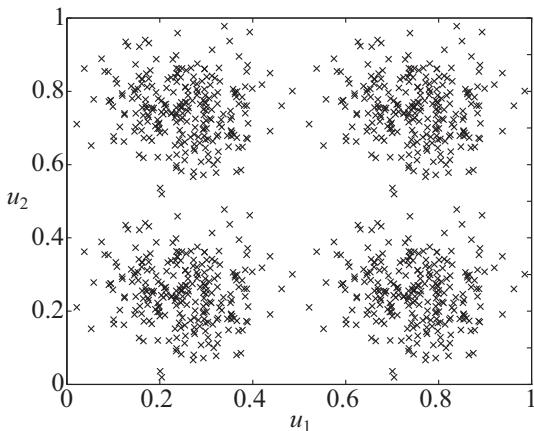
7.1 Uvod v metode nенадзорованга učenja

V nadaljevanju si bomo bolj natančno ogledali nekaj metod nенадзорованega učenja. Glavna lastnost nенадзорованega učenja je ta, da so vse spremenljivke enako obravnavane. To pomeni, da jih ne moremo ločiti na vhodne $\{u(i)\}$, $i = 1, \dots, N$ in izhodne spremenljivke y . Če to lahko storimo, potem lahko takrat uporabimo katero od metod nadzorovanega učenja. Seveda imamo največkrat izhode dosegljive in lahko to informacijo dobro vključimo tudi v nенадзоровано učenje. Nенадзорованo učenje je lahko zelo učinkovito pri predprocesiranju podatkov; glej sliko 7.1. Predprocesiranje transformira podatke iz ene oblike v obliko, ki je bolj primerna za nadaljnjo obdelavo.



Slika 7.1. Predprocesiranje podatkov se običajno izvaja z nенадзорovanimi tehnikami učenja. Originalnih spremenljivk u_1, u_2, \dots, u_p ne uporabimo direktno pri preslikavi na izhod modela. Običajno jih najprej predobdelamo, da dobimo njihove transformirane x_1, x_2, \dots, x_q . Število transformirank q je navadno manjše od števila originalnih spremenljivk p , ker procesiranje običajno pomeni tudi zgoščevanje vhodnega prostora in zmanjšanje dimenzije modela.

Naslednji primer ilustrira uporabo nенадзорованega učenja za enostaven primer klasifikacije. Na sliki 7.2 vidimo razdelitev podatkov v dvodimenzionalnem prostoru spremenljivk u_1-u_2 . Predpostavimo, da spremenljivki u_1 in u_2 predstavljata dve lastnosti na osnovi katerih lahko ločimo različne razrede, ki pa so predstavljeni s celoštevilskimi vrednostmi. Vrednost razreda predstavlja izhod modela y . Predpostavimo, da imamo problem, ki je podoben XOR operaciji. Tako pripadata skupini podatkov zgoraj levo in spodaj desno (roja) razredu 1, ostala dva pa razredu 0. Pri nadzorovanem učenju bi vsak učni vzorec vseboval spremenljivki u_1 in u_2 in pripadajočo vrednost izhoda y , to je 1 ali 0 (ang. *labeled data*). Na ta način lahko enostavno rešimo ta problem razvrščanja vzorcev.



Slika 7.2. Štirje roji v dvodimenzionalnem prostoru. Brez informacije o ustreznom izhodu lahko s pomočjo nенадзорованega učenja ločimo štiri skupine podatkov, oziroma roje.

Pri nenadzorovanem učenju pa izhoda ne poznamo (*ang. unlabeled data*), kar pomeni, da je učna množica sestavljena samo iz vhodnih podatkov brez informacije o pripadajočih razredih. Največ kar lahko storimo z nenadzorovanim učenjem, je to, da poiščemo grupe oziroma roje v vhodnih podatkih. Na ta način lahko najdemo, na primer centre posameznih rojev z vrednostmi $(0.25, 0.25)$, $(0.75, 0.25)$, $(0.25, 0.75)$ in $(0.75, 0.75)$ v obliki krogov z radijem 0.25. V drugem koraku potem te roje z metodo nadzorovanega učenja preslikamo v izhodne vrednosti pripadajočih razredov. Na ta način smo z uporabo nenadzorovanega učenja transformirali problem preslikave celotne množice vhodnih podatkov na preslikavo 4 centrov rojev v 2 razreda. S tem smo zelo zmanjšali kompleksnost druge transformacije.

Zgornji primer kaže na pogosto uporabo nenadzorovanega učenja. Enostopenjsko nadzorovano učenje lahko zamenjamo z dvostopenjskim učenjem. Najprej je to nenadzorovano učenje, ki stisne podatke, potem pa sledi še enostavno nadzorovano učenje. Ponavadi je dvostopenjsko učenje bistveno manj računsko zahtevno kot enostopenjsko. Prva stopnja nenadzorovanega učenja je v tem primeru stiskanje podatkov (*ang. information compression*) ali zmanjšanje dimenzije podatkov (*ang. dimensionality reduction*).

Nenadzorovano učenje na vhodno-izhodnih podatkih. Največja omejitev uporabe nenadzorovanega učenja je ta, da navadno ne uporabimo izhodna procesa. Obstaja strategija kako vključiti tudi izhodni signal procesa, ne da bi pri tem eksplicitno optimirali kriterijsko funkcijo, ki temelji na izhodnem pogrešku. Metoda se imenuje rojenje vhodno-izhodnega prostora (*ang. product-space clustering*). Namesto uporabe metode rojenja na vhodnem prostoru spremenljivk, (*ang. input space*) ki jo definirajo $\underline{u} = [u_1 \ u_2 \ \dots \ u_m]^T$, izvedemo rojenje na celotnem vhodno-izhodnem prostoru (*ang. product space*), ki ga tvorijo spremenljivke

$$\underline{u}_e = [u_1 \ u_2 \ \dots \ u_m \ y]^T. \quad (7.1)$$

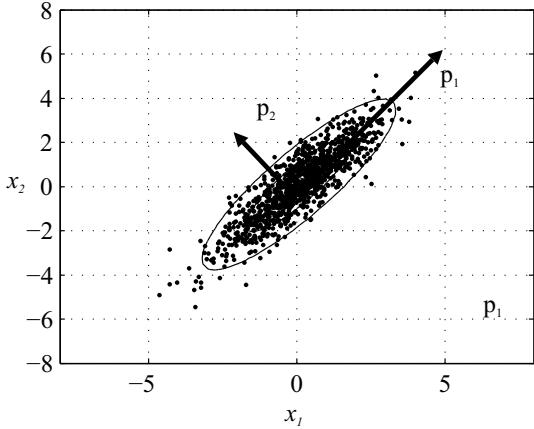
Na ta način se v rojih odražajo relacije med vhodno-izhodnimi spremenljivkami, ki na ta način definirajo model znotraj tega roja. Na osnovi metode rojenja Gustafson-Kessel lahko znotraj vhodno-izhodnega prostora poiščemo roje in njihove kovariančne matrike, ki preko svojih lastnih vektorjev definirajo model procesa, oziroma relacijo med vhodi in izhodom. Model je običajno zapisan v obliki hiperravnine $y = w_0 + w_1 u_1 + w_2 u_2 + \dots + w_p u_m$. V nadaljevanju si bomo ogledali tudi to, kako iz kovariančne matrike F_j ocenimo parametre hiperravnine w_i . Tako ocenjeni parametri so enaki kot, če bi jih ocenili s pomočjo metode popolnih najmanjših kvadratov (*Total Least Squares - TLS*). Vse metode rojenja, ki jih bomo spoznali, lahko uporabimo na vhodno-izhodnem prostoru preprosto tako, da vektor \underline{u} zamenjamo z vektorjem \underline{u}_e .

V nadaljevanju si bomo najprej ogledali metodo glavnih komponent (*ang. Principal Component Analysis - PCA*), ki je ena od osnovnih metod nenadzorovanega učenja. Uporabljam jo za transformacijo koordinatnih osi, redukcijo dimenzij in odpravo kolinearnosti v podatkih. Po tem pa bomo v poglavju 7.3 osvetlili nekaj metod rojenja, od klasične metode rojenja s k-povprečji do metode Gustafson-Kessel in metod, ki temeljijo na nevronskih mrežah.

7.2 Metoda glavnih komponent

Metoda glavnih komponent temelji na obravnavi kovariančne matrike osrediščenih in normiranih podatkov. Gre za transformacijo (rotacijo) osi koordinatnega sistema v nov ortogonalni sistem, ki rezultira v največji kovarianci podatkov glede na novo izbrane osi.

Za primer si poglejmo preprost dvodimensionalni primer, ki ga prikazuje slika 7.3. Predstavljene so meritve dveh naključnih spremenljivk x_1 in x_2 v n časovnih trenutkih. Vsak par (x_1, x_2) predstavlja točko v dvodimensionalnem prostoru meritov. Meritve zapišemo v matriko



Slika 7.3. Analiza podatkov po metodi glavnih komponent (PCA) za dvodimenzionalni problem.

podatkov $\underline{X} \in \mathbb{R}^{n \times 2}$, kjer vsaka vrstica $\underline{x}_k = [x_{1k}, x_{2k}]$ predstavlja zajem obeh spremenljivk v določenem časovnem trenutku. Spremenljivki sta na primer posledici dveh procesov, ki sta močno korelirana, kar pomeni, da med njima obstaja določena linearna odvisnost in so odstopenja od te linearne povezave posledica šuma. To pomeni, da lahko podatke predstavimo s samo eno novo spremenljivko \underline{p}_1 , ki je linearna kombinacija baznih vektorjev $\underline{x}_1 = [x_1 \ 0]^T$ in $\underline{x}_2 = [0 \ x_2]^T$. Smer vektorja \underline{p}_1 je smer največje varianco podatkov in jo lahko zapišemo tudi kot $\underline{p}_1 = \underline{x}_1 + k_c \underline{x}_2$. Smer vektorja \underline{p}_2 je smer, kjer je varianca najmanjša. To smer lahko zapišemo z vektorjem $\underline{p}_2 = \underline{x}_2 - k_c \underline{x}_1$. Na ta način smo dobili zapis, ki v smeri komponente \underline{p}_1 opisuje večji del varianco podatkov in latentno smer \underline{p}_2 , ki predstavlja šum. Tako se dvodimenzionalni problem, opisan z baznima vektorjema \underline{x}_1 in \underline{x}_2 , pretvori v enodimenzionalni problem baznega vektorja \underline{p}_1 z eliminacijo vektorja \underline{p}_2 . Seveda gre pri tem za predpostavko linearne odvisnosti med spremenljivkama x_1 in x_2 .

V primeru m -dimenzionalnih podatkov lahko problem glavnih komponent (PCA) predstavimo na naslednji način. Cilj je v premiku in rotaciji osi tako, da dobimo maksimalno varianco podatkov okoli nove osi, ki jo definiramo z vektorjem $\underline{p}_i = [p_{i1}, \dots, p_{im}]^T$, $i = 1, \dots, m$, kjer je vsaka nova koordinatna os linearna kombinacija starih osi. Ker velja, da mora biti tudi nova baza ortonormirana, lahko zapišemo $\underline{p}_i^T \underline{p}_i = 1$, $i = 1, \dots, m$. Podatke lahko zapišemo z matriko $\underline{X} \in \mathbb{R}^{n \times m}$, kjer je n število meritev in m število merjenih spremenljivk. Projekcijo podatkovne matrike \underline{X} na nov nabor baznih vektorjev \underline{p} lahko zapišemo kot $\underline{X}\underline{p}$. Iščemo bazni vektor okoli katerega je varianca podatkov največja. Optimizacijo maksimalne variance z omejitvijo dolžine baznih vektorjev lahko najlaže rešimo z uporabo Lagrangeevih koeficientov, kjer optimiramo funkcional:

$$I(\underline{p}) = (\underline{X}\underline{p})^T (\underline{X}\underline{p}) + \lambda (1 - \underline{p}^T \underline{p}) . \quad (7.2)$$

Če izračunamo parcialni odvod $\frac{\partial I}{\partial \underline{p}}$, dobimo naslednjo enačbo

$$\frac{\partial I}{\partial \underline{p}} = (\underline{X}^T \underline{X}) \underline{p} - \lambda \underline{p} = 0 . \quad (7.3)$$

Vidimo, da so osi, ki optimirajo varianco podatkov, kar smeri lastnih vektorjev \underline{p}_i , $i = 1, \dots, m$, pripadajoče lastne vrednosti λ_i pa predstavljajo variance v teh smereh ($(\underline{X}^T \underline{X} - \lambda \underline{I}) \underline{p} = 0$), kjer je \underline{I} enotska matrika. Lastne vektorje in lastne vrednosti kovariančne matrike $\underline{X}^T \underline{X}$ lahko izračunamo s pomočjo dekompozicije singularnih vrednosti matrike \underline{X} , (ang. *Singular Value Decomposition* - SVD), kot to prikazuje naslednja enačba

$$\underline{\Sigma} = \frac{1}{n-1} \underline{X}^T \underline{X} = \underline{P} \underline{\Lambda} \underline{P}^T = \sum_{j=1}^m \lambda_j \underline{p}_j \underline{p}_j^T, \quad (7.4)$$

kjer je $\underline{P} \in \mathbb{R}^{m \times m}$ matrika m ortonormalnih baznih vektorjev \underline{p}_j , $j = 1, \dots, m$ in $\underline{\Lambda}$ diagonalna matrika lastnih vrednosti λ_j , $j = 1, \dots, m$. Vsaki lastni vrednosti λ_j pripada določen ortonormalni bazni lastni vektor \underline{p}_j . Matrika lastnih vektorjev \underline{P} je urejena tako, da so stoljni vektorji razvrščeni od leve proti desni po pripadajočih lastnih vrednostih ($\lambda_1 > \lambda_2 > \dots > \lambda_m$).

Navadno imamo v matriki podatkov tudi določene kolinearne spremenljivke, kar pomeni, da za opis celotne informacije, ki jo nosi matrika podatkov, ne potrebujemo vseh lastnih vektorjev kovariančne matrike $\underline{\Sigma}$. Predpostavimo lahko, da določen del informacije razložimo s s lastnimi vektorji, ki jih imenujemo *glavne komponente*. Kriterij za izbiro števila lastnih vektorjev je naslednji

$$\frac{\sum_{j=1}^s \lambda_j}{\sum_{j=1}^m \lambda_j} \geq \epsilon, \quad (7.5)$$

kjer je ϵ izbrani prag, ki ima običajno vrednost $\epsilon = 0.95$.

Kovariančno matriko lahko sedaj aproksimiramo z matriko, ki jo dobimo tako, da lastne vektorje (z majhnimi lastnimi vrednostmi) \underline{p}_i , $i = s+1, \dots, m$ enostavno zanemarimo. Aproksimirana kovariančna matrika je sedaj enaka

$$\underline{\Sigma}_s = \sum_{j=1}^s \lambda_j \underline{p}_j \underline{p}_j^T, \quad (7.6)$$

kjer vektorji \underline{p}_j , $j = 1, \dots, s$ definirajo glavne smeri širjenja podatkov in so uteženi s pripadajočimi koreni lastnih vrednosti.

Analiza glavnih komponent (PCA) je metoda, ki omogoča obdelavo in prikaz množice podatkov, shranjenih v matriki X . Vrstice predstavljajo vzorčene vrednosti procesnih spremenljivk v določenih časovnih trenutkih, stolci pa vzorčene časovne poteke posameznih procesnih spremenljivk, ki jih pred tem normiramo (srednja vrednost nič in varianca ena). Matrika podatkov je prikazana v enačbi

$$X = \begin{bmatrix} x_1(1) & x_2(1) & \dots & x_m(1) \\ x_1(2) & x_2(2) & \dots & x_m(2) \\ \vdots & \vdots & \vdots & \vdots \\ x_1(n) & x_2(n) & \dots & x_m(n) \end{bmatrix}, \quad (7.7)$$

kjer je n število časovnih vzorcev oziroma število meritev procesnih spremenljivk, m pa je število procesnih spremenljivk, ki jih merimo.

Ortogonalne spremenljivke, imenovane zadetki, so notranji produkt merjenih spremenljivk in matrike preslikave \underline{P} , kar opišemo z enačbo

$$\underline{T} = \underline{X} \underline{P}, \quad (7.8)$$

kjer \underline{T} pomeni matriko zadetkov.

Matrika glavnih vektorjev \underline{P} je statistični model procesa, zadetki pa so psevdospremenljivke, ki nimajo realnega ozadja, saj so linearne kombinacije stolpcev matrike \underline{X} . Če sestavimo delno matriko glavnih lastnih vektorjev \underline{P}_s tako, da vzamemo le s prvih stolpcev oziroma lastnih

vektorjev matrike \underline{P} , ki pomenijo večino variance oziroma informacije vhodnih podatkov matrike \underline{X} , dobimo aproksimirano matriko podatkov

$$\underline{T}_s = \underline{X}\underline{P}_s , \quad (7.9)$$

$$\underline{X}_s = \underline{T}_s \underline{P}_s^T . \quad (7.10)$$

Matriko podatkov lahko zapišemo z reduciranim prostorom in ostankom

$$\underline{X} = \underline{T}_s \underline{P}_s^T + \underline{E}_x , \quad (7.11)$$

kjer indeks s stoji zaradi izbranih prvih s lastnih vektorjev z največjimi lastnimi vrednostmi, ki jih zaradi tega imenujemo glavne komponente prostora zadetkov, \underline{E}_x pa označuje matriko ostankov oziroma residualov, ki jih imenujemo tudi prostor šuma. To pomeni, da so z izbiro glavnih smeri v prostoru podatkov odkrite vse najpomembnejše linearne relacije med podatki vhodne matrike \underline{X} . Ortogonalizacija je ena od najpomembnejši lastnosti metode glavnih komponent, saj omogoča eliminacijo kolinearnih podatkov in podatkov, ki imajo manjši vpliv.

7.2.1 Invertiranje matrike s kolinearnimi podatki in regresija na osnovi PCA

V nadaljevanju si bomo ogledali primer regresije, ko je informacijska matrika singularna zaradi kolinearnosti podatkov. Regresija na osnovi glavnih komponent je nadaljevanje metode PCA s pomočjo katere lahko modeliramo izhodne podatke \underline{y} na osnovi merjenih vhodnih podatkov procesa \underline{X}

$$\underline{y} = \underline{X}\underline{\theta} + \underline{e}_y , \quad (7.12)$$

$$\underline{y} = \underline{T}\underline{P}^T\underline{\theta} + \underline{e}_y , \quad (7.13)$$

$$\underline{y} = \underline{T}\underline{\theta}_T + \underline{e}_y , \quad (7.14)$$

kjer je \underline{e}_y pogrešek med modelom in izmerjenimi podatki in $\underline{\theta}_T = \underline{P}^T\underline{\theta}$. Parametre modela dobimo z uporabo metode najmanjših kvadratov, kjer je kriterijska funkcija enaka $I(\underline{\theta}) = \frac{1}{2}\underline{e}_y^T \underline{e}_y$

$$\begin{aligned} \frac{\partial I(\underline{\theta})}{\partial \underline{\theta}} &= (\underline{y} - \underline{X}\underline{\theta})^T (\underline{y} - \underline{X}\underline{\theta}) = 0 , \\ \underline{\theta} &= (\underline{X}^T \underline{X})^{-1} \underline{X}^T \underline{y} , \end{aligned} \quad (7.15)$$

ali v primeru, ko delamo z zadetki \underline{T}

$$\underline{\theta} = \underline{P} (\underline{T}^T \underline{T})^{-1} \underline{T}^T \underline{y} . \quad (7.16)$$

V enačbi 7.15 vektor $\underline{\theta}$ predstavlja regresijski model na osnovi glavnih komponent, ki preslikava matriko vhodnih podatkov \underline{X} v vektor izhodnih podatkov \underline{y} . Do parametrov modela pridemo preko invertiranja informacijska matrike $\underline{X}^T \underline{X}$. Inverz te matrike je singularen v primeru, ko delamo s podatki, ki so kolinearni oziroma linearno odvisni. V tem primeru uporabimo za izračun inverza metodo singularnih vrednosti matrike, kjer informacijsko matriko aproksimiramo z glavnimi komponentami, s tistimi komponentami, ki imajo od nič različne lastne vrednosti

$$(\underline{X}^T \underline{X})^{-1} \approx (\underline{X}_s^T \underline{X}_s)^{-1} , \quad (7.17)$$

$$(\underline{X}_s^T \underline{X}_s)^{-1} = (\underline{P}_s \underline{T}_s \underline{T}_s^T \underline{P}_s^T)^{-1} . \quad (7.18)$$

Če upoštevamo relacijo med matriko lastnih vrednosti \underline{A}_s in zadetkov \underline{T}_s , dobimo

$$(\underline{X}_s^T \underline{X}_s)^{-1} = (\underline{P}_s \underline{\Lambda}_s \underline{P}_s^T)^{-1}. \quad (7.19)$$

Inverz informacijske matrike je tako enak

$$(\underline{X}_s^T \underline{X}_s)^{-1} = \underline{P}_s \underline{\Lambda}_s^\dagger \underline{P}_s^T, \quad (7.20)$$

kjer je $\underline{\Lambda}_s^\dagger$ diagonalna matrika inverznih lastnih vrednosti za s glavnih vektorjev

$$\underline{\Lambda}_s^\dagger = \begin{bmatrix} \frac{1}{\lambda_1} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \frac{1}{\lambda_s} \end{bmatrix}. \quad (7.21)$$

Vektor parametrov modela lahko sedaj izračunamo kot

$$\underline{\theta} = \underline{P}_s \underline{\Lambda}_s^\dagger \underline{P}_s^T \underline{X}_s^T \underline{y}. \quad (7.22)$$

Primer 7.2.1. Na primeru bo prikazana uporaba metode regresije na osnovi glavnih komponent (ang. *principal component regression - PCR*). Predpostavimo sistem, ki ga lahko modeliramo z linearno funkcijo, ki je dana z naslednjo enačbo

$$y = -0.58x_1 + 1.333x_3 + \mathcal{N}(0, 0.02),$$

kjer z $\mathcal{N}(\mu_x, \sigma_x^2)$ zapišemo stohastični signal s srednjo vrednostjo μ_x in varianco σ_x^2 . Predpostavimo matriko vhodnih podatkov \underline{X} , ki je sestavljena na naslednji način iz spremenljivk $x_1 = [1 \ -1 \ 0 \ 1]^T$, $x_2 = [0 \ 1 \ -1 \ 1]^T$ in linearno odvisne tretje spremenljivke $x_3 = -2x_1 + 1.5x_2 + \mathcal{N}(0, 0.01)$. V našem primeru so vrednosti matrike \underline{X} naslednje:

$$\underline{X} = (x_1, x_2, x_3) = \begin{bmatrix} 1.0 & 0.0 & -1.9985 \\ -1.0 & 1.0 & 3.4944 \\ 0.0 & -1.0 & -1.5034 \\ 1.0 & 1.0 & -0.4958 \end{bmatrix}. \quad (7.23)$$

Gornja preslikava vhodno matriko \underline{X} preslika v izhodne vrednosti, ki jih merimo (pri predpostavljeni napaki pri meritvi)

$$\underline{y} = [-3.2475 \ 5.2389 \ -2.0067 \ -1.2417]^T.$$

Matrika \underline{X} je slabo pogojena, zato metoda navadne linearne regresije ne da primernih rezultatov. Dobimo model, ki ga lahko zapišemo z enačbo

$$\underline{y} = 0.8166x_1 - 1.050x_2 + 2.0335x_3.$$

Problem nastane zaradi kolinearnosti stolpcov matrike \underline{X} , kar se odraža v singularnosti informacijske matrike.

Če izračunamo singularne vrednosti matrike \underline{X} , dobimo tri lastne vrednosti $\lambda_1 = 21.7109$, $\lambda_2 = 3.0000$ in $\lambda_3 = 0$. To pomeni, da sta za primerno aproksimacijo sistema potrebna dva lastna vektorja, oziroma dva vektorja zadetkov s katerima aproksimiramo podatkovno matriko \underline{X} . Razlog je v tem, da je tretja lastna vrednost informacijske matrike enaka nič, kar pomeni, da je varianca podatkov v smeri tretjega lastnega vektorja enaka nič. Glavna lastna vektorja sta $\underline{p}_1 = [-0.2971 \ 0.2234 \ 0.9283]^T$ in $\underline{p}_2 = [-0.6009 \ -0.7993 \ -0.0000]^T$. Glavna lastna vektorja tvorita matriko $\underline{P}_2 = [\underline{p}_1 \ \underline{p}_2]$.

Dobili smo model, ki podatke aproksimira bistveno bolje, kot model, ki ga dobimo z navadno regresijsko metodo. Model v primeru regresije na osnovi glavnih komponent je naslednji

$$y = -0.6287x_1 + 0.0362x_2 + 1.3095x_3 .$$

Ugotovimo lahko, da metoda glavnih komponent zagotavlja dobro oceno parametrov modela tudi v primeru, ko so podatki, ki jih opazujemo medsebojno korelirani.

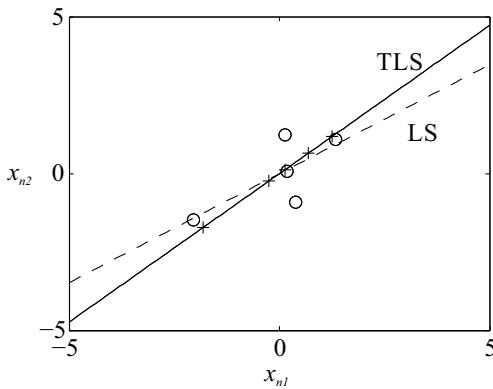
Primer 7.2.2. Na enostavnem primeru podatkov v dvodimensionalnem prostoru bo prikazana uporaba regresije na osnovi glavnih komponent. Imamo množico podatkov, ki jo zapišemo z matriko $\underline{X} = [\underline{x}_1 \underline{x}_2]$. Opazovani spremenljivki sta medsebojno odvisni. Relacija med njima je $x_2 = x_1 + 2 + \mathcal{N}(0, 1)$. Število meritev prve in druge spremenljivke je enako $n = 5$. Če izračunamo srednje vrednosti posameznih spremenljivk, dobimo naslednje vrednosti $\mu_{x_1} = 1.2432$ in $\mu_{x_2} = 3.3393$. Osrediščene podatke ($\underline{x}_{nj}(k) = \underline{x}_j(k) - \mu_{x_j}$, $k = 1, \dots, n$, $j = 1, 2$) zložimo v matriko $\underline{X}_n = [\underline{x}_{n1} \underline{x}_{n2}]$. Matrika osrediščenih podatkov je enaka

$$\underline{X}_n = \begin{bmatrix} -2.0429 & -1.4704 \\ 0.1368 & 1.2316 \\ 0.3881 & -0.9105 \\ 0.1807 & 0.0648 \\ 1.3373 & 1.0845 \end{bmatrix} . \quad (7.24)$$

Kovariančna matrika osrediščenih podatkov je enaka

$$\underline{\Sigma}_n = \frac{1}{n-1} \underline{X}_n^T \underline{X}_n = \begin{bmatrix} 1.5410 & 1.0703 \\ 1.0703 & 1.4221 \end{bmatrix} . \quad (7.25)$$

Če razstavimo kovariančno matriko po metodi singularnih vrednosti, dobimo matriko lastnih



Slika 7.4. Primerjava regresije po metodi najmanjših kvadratov (LS) in totalnih najmanjših kvadratov z uporabo metode glavnih komponent (TLS).

vektorjev

$$\underline{P} = \begin{bmatrix} -0.7265 & -0.6872 \\ -0.6872 & 0.7265 \end{bmatrix} \quad (7.26)$$

in matriko lastnih vrednosti

$$\underline{A} = \begin{bmatrix} 10.2140 & 0 \\ 0 & 1.6383 \end{bmatrix} . \quad (7.27)$$

Ugotovimo, da s prvim lastnim vektorjem izpolnimo okoli 0.85 celotne informacije in lahko informacijo v smeri drugega lastnega vektorja interpretiramo kot šum v podatkih, ki je v tem primeru relativno velik. Zadetke v smeri glavnih lastnih vektorjev (tistih z največjimi lastnimi vrednostmi) dobimo tako, da matriko normiranih podatkov preslikamo preko matrike glavnih lastnih vektorjev

$$\underline{T}_s = \underline{X}_n \underline{P}_s = [2.4946 \ -0.9458 \ 0.3438 \ -0.1758 \ -1.7168]^T, \quad (7.28)$$

kjer je \underline{P}_s matrika glavnih lastnih vektorjev. V našem primeru je matrika glavnih lastnih vektorjev enaka enemu samemu vektorju \underline{P}_1 . Matrika zadetkov pa je v našem primeru vektor zadetkov \underline{T}_1 . Če sedaj vektor zadetkov \underline{T}_1 pomnožimo z matriko glavnih lastnih vektorjev, dobimo matriko linearno odvisnih podatkov

$$\hat{\underline{X}}_n = \underline{T}_1 \underline{P}_1^T. \quad (7.29)$$

Transformacija preko matrike glavnih vektorjev pomeni iskanje linearne relacije v matriki podatkov s tem, da eliminiramo smeri manj pomembne informacije. Linearno relacijo iščemo v smislu minimizacije matrične norme

$$\max_{\theta} \|\underline{X}_n - \hat{\underline{X}}_n\|_2.$$

Kar pomeni minimizacijo kvadratnih razdalj podatkov od linearnega modela podatkov (hiperravnine), ki je v našem primeru premica.

Lastni vektor, ki ustreza najmanjši lastni vrednosti (v našem primeru je to vektor \underline{p}_2) je normalen na hiperravnino (splošno) glavnine podatkov in nam služi za definicijo linearnega modela

$$(\underline{x} - \underline{v})^T \cdot \underline{p}_2 = 0,$$

kjer je $\underline{v} = [\mu_{x_1} \ \mu_{x_2}]$.

Dobimo model, ki je v našem primeru definirana kot

$$-0.6872x_1 + 0.7265x_2 + 0.6872\mu_{x_1} - 0.7265\mu_{x_2} = 0$$

ali izraženo v eksplisitni obliki

$$x_2 = 0.9459x_1 + 2.1633.$$

Tako izračunana linearna funkcija predstavlja model v prostoru dejanskih spremenljivk.

Model, ki opisuje normirane podatke pa je definiran v normiranem prostoru podatkov in je enak

$$-0.6872x_1 + 0.7265x_2 = 0.$$

Premica, ki predstavlja ta model je na sliki 7.4 in je polno izvlečena črta, označena s TLS. To je primer ocene parametrov modela po metodi popolnih najmanjših kvadratov (*ang. total least square*). Na sliki so normirani podatki narisani kot krogci, s križci pa so narisane projekcije teh podatkov na glavni vektor $\underline{p}_1 = [-0.7265 \ -0.6872]^T$. Matrika $\hat{\underline{X}}_n$ je matrika aproksimiranih podatkov.

$$\hat{\underline{X}}_n = [\hat{x}_{n1}, \hat{x}_{n2}] = \begin{bmatrix} -1.8122 & -1.7143 \\ 0.6870 & 0.6499 \\ -0.2497 & -0.2363 \\ 0.1277 & 0.1208 \\ 1.2471 & 1.1798 \end{bmatrix}. \quad (7.30)$$

Na osnovi aproksimirane matrike normiranih podatkov izvedemo metodo najmanjših kvadratov tako, da določimo regresor na naslednji način

$$\hat{\underline{X}}(k) = [\hat{x}_{n1}(k) \ 1], \quad k = 1, \dots, n,$$

dobimo po metodi najmanjših kvadratov vektor parametrov modela

$$\underline{\theta}_2 = (\hat{\underline{X}}^T \hat{\underline{X}})^{-1} \hat{\underline{X}}^T \hat{\underline{x}}_{n2} = [0.9460 \ 0.0000]^T,$$

ki je popolnoma enak tistemu, ki ga definira lastni vektor z najmanjšo (zanemarljivo) lastno vrednostjo.

Če regresijsko premico izračunamo po metodi najmanjših kvadratov direktno iz normiranih podatkov, kjer tvorimo naslednji regresor

$$\underline{X}(k) = [x_{n1}(k) \ 1], \quad k = 1, \dots, n,$$

potem po metodi najmanjših kvadratov dobimo naslednji vektor parametrov modela

$$\underline{\theta}_3 = (\underline{X}^T \underline{X})^{-1} \underline{X}^T \underline{x}_{n2} = [0.6945 \ 0.0000]^T.$$

Regresijska premica je v tem primeru na sliki 7.4 prikazana s črtkano črto in oznako LS.

7.2.2 Sprotni nadzor procesa na osnovi metode PCA

Z metodo PCA lahko skrčimo večdimensionalni prostor izmerjenih podatkov na prostor ortogonalnih spremenljivk, ki je nižje dimenzije. Prikaz spremenljivk v nižji dimenziji omogoča učinkovitejši nadzor procesa [25, 91]. Hkrati se izognemo problemom zaradi redundantnosti podatkov (kolinearnost meritev) in izločimo šum. Pogosto iz matrike podatkov generiramo skupno statistično oceno ali mero, ki jo nato sproti spremljamo. Skupna statistična ocena, ki definira odstopanje zadetkov opazovanega sistema od referenčnih zadetkov, se imenuje Mahalanobisova razdalja. Ena od statističnih mer je tudi Hotellingova T^2 mera, ki se od Mahalanobisove razlikuje le v faktorju zaupanja, ki je konstanten pri določenem eksperimentu, zato se bomo v nadaljevanju omejili le na Mahalanobisovo mero.

Mahalanobisovo mero zapišemo jo z enačbo:

$$m_k = (\underline{x}_k - \underline{v})^T \underline{P}_s \underline{\Lambda}_s^\dagger \underline{P}_s^T (\underline{x}_k - \underline{v}), \quad (7.31)$$

kjer je \underline{x}_k^T k -ta vrstica podatkov v matriki podatkov \underline{X} , \underline{v} je vektor srednje vrednosti opazovanih podatkov, s je število glavnih komponent v matriki podatkov in sta \underline{P}_s in $\underline{\Lambda}_s^\dagger$ matriki glavnih lastnih vektorjev in inverznih lastnih vrednosti. Matriki glavnih lastnih vektorjev in lastnih vrednosti tvorita aproksimirano kovariančno matriko C_s , ki je definirana z enačbo

$$C_s = \underline{P}_s \underline{\Lambda}_s \underline{P}_s^T. \quad (7.32)$$

Kadar so spremenljivke v prostoru podatkov osrediščene in skalirane s svojimi variancami, takrat se Mahalanobisova mera v prostoru zadetkov poenostavi v

$$m_k = \underline{x}_k^T (\underline{P}_s \underline{\Lambda} \underline{P}_s^T)^{-1} \underline{x}_k = \underline{t}_k^T \underline{\Lambda}^{-1} \underline{t}_k, \quad (7.33)$$

kjer \underline{t}_k^T pomeni k -to vrstico zadetkov oziroma vektor zadetkov k -te vrstice podatkov ($\underline{t}_k = \underline{x}_k \underline{P}_s$).

V prostoru zadetkov Mahalanobisovo mero definiramo na naslednji način

$$m_k = \sum_{i=1}^s \frac{t_{ki}^T t_{ki}}{\lambda_i}, \quad (7.34)$$

kjer je vektor zadetkov v trenutku k enak $\underline{t}_k^T = [t_{k1}, \dots, t_{ks}]$ in λ_i i -ta lastna vrednost matrike lastnih vrednosti \underline{A}_s .

Druga pomembna statistična mera na osnovi katere lahko spremljamo delovanje procesa je *mera q* , ki ocenjuje trenutne kvadratne vrednosti pogreška med vektorjem izmerjenih podatkov in prototipnim modelom danim z matriko lastnih vektorjev \underline{P}_s

$$\underline{q}_k = (\underline{x}_k - \underline{t}_k \underline{P}_s^T)^T (\underline{x}_k - \underline{t}_k \underline{P}_s^T). \quad (7.35)$$

Če za določen niz vse kvadratne vrednosti pogreškov seštejemo, dobimo mero Q . Mera Q je definirana z naslednjo enačbo

$$\underline{Q} = \sum_{k=1}^n (\underline{x}_k - \underline{t}_k \underline{P}_s^T)^T (\underline{x}_k - \underline{t}_k \underline{P}_s^T). \quad (7.36)$$

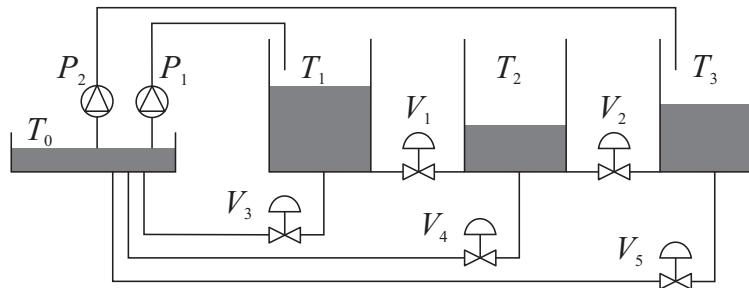
Razvrščanje procesov, odkrivanje napak in spremljanje procesov. Razvrščanje procesov temelji na primerjavi podatkov opazovanega procesa z referenčnimi podatki, ki jih dobimo na osnovi podatkov izmerjenih na procesu med različnimi načini delovanja. Razvrščanje in detekcija temeljita na reduciranem prostoru zadetkov in se izvajata paketno in nesprotno.

Eden od načinov razvrščanja procesov *temelji na vsoti kvadratov residuov* ali na **meri Q** . Razvrščanje procesov v posamezne razrede je mogoče tudi na osnovi **diagramov zadetkov**. V tem primeru opazujemo odvisnost enega vektorja zadetkov glede na drugega, npr. diagram (t_1, t_2) za določen časovni trenutek. V tem primeru procesi s podobnim obnašanjem definirajo določen *roj*. Glede na roje diagrama zadetkov lahko razvrstimo procese v različne razrede.

V primeru spremljanja procesa pa v vsakem časovnem trenutku izračunavamo kriterij na osnovi katerega ugotavljamo, če se novi podatki nahajajo v določenem razredu, ki je definiran z referenčnimi podatki. Spremljanje je prvi korak pri iskanju napak v delovanju procesa.

Ena od metod spremljanja delovanja sistema temelji na **Mahalanobisovi razdalji**, ki definira odstopanje zadetkov opazovanega sistema od referenčnih zadetkov. S pomočjo te mere ugotovimo ali lahko zadetke novih izmerjenih podatkov razvrstimo v določen razred.

Metoda PCA pri odkrivanju napak v primeru treh zbiralnikov. V tem primeru si bomo ogledali primer odkrivanja napak z uporabo metode PCA pri procesu treh zbiralnikov. Sistem treh zbiralnikov, ki ga prikazuje slika 7.5, je sestavljen iz zbiralnikov T_1 , T_2 , T_3 in zbirnega rezervoarja T_0 . Nivo v zbiralniku T_1 uravnavamo z delovanjem črpalk P_1 , medtem ko se nivo



Slika 7.5. Shema procesa treh zbiralnikov.

v zbiralniku T_3 uravnava z odprtjem servoventila V_5 (črpalka P_2 dela s konstantnimi obrati). Vsebina iz zbiralnika T_1 in T_3 se meša v zbiralniku T_2 in nato vrača v zbiralnik T_0 . Izhodi procesa so nivoji v vseh treh zbiralnikih, vhoda pa sta delovanje črpalke P_1 in odprtje ventila V_5 . Zaradi izolacije napak, oziroma ugotavljanja vzroka za nepravilno delovanje procesa, smo poleg normalnega načina delovanja obravnavali še delovanje ob prisotnosti določenih napak. Za vsak obravnavani način delovanja smo iz posnetih meritev določili model PCA, ki se je temu delovanju dobro prilegal. Za vsa druga delovanja pa se statistične lastnosti procesa spremenijo, glavnina podatkov torej ni več v smeri glavnih osi, kar se odraža z odklonom obeh mer, opisanih v podoglavlju 7.2.2. Simboličen prikaz zaznavanja in izoliranja napak podaja slika 7.6.

Nadzor in spremljanje procesa pomeni razvrščanje novih podatkov glede na nominalne podatke. Izbrani nominalni podatki, ki opisujejo normalno delovanje, služijo za izračun nominalnega modela. Detekcija pomeni iskanje najbolj podobnega modela, ki opisuje eno od možnih napak pri delovanju sistema.

Pri meri M (Mahalanobisova razdalja m_k) sestavimo transformacijsko matriko tako, da pretvorimo prostor meritev v prostor dimenzijske ene. Mera pove kako zadetki novih podatkov \underline{x}_k pripadajo množici nominalnih podatkov, ki je lahko množica podatkov normalnega delovanja ali pa so to množice, ki opisujejo delovanje z napako. V primeru spremljanja procesa definiramo Mahalanobisovo razdaljo kot

$$m_k^f = \left(\underline{x}_k - \underline{v}_f \right)^T \underline{P}_f \underline{\Lambda}_f^\dagger \underline{P}_f^T \left(\underline{x}_k - \underline{v}_f \right), \quad (7.37)$$

kjer so \underline{x}_k novo izmerjeni podatki, \underline{v}_f je srednja vrednost nominalnih podatkov množice f in sta \underline{P}_f in $\underline{\Lambda}_f$ matriki glavnih s lastnih vektorjev in lastnih vrednosti.

Pri meri Q po enačbi 7.36 pretvorimo prvotni prostor podatkov v prostor zadetkov z uporabo matrike lastnih vektorjev \underline{P}_f , kjer z indeksom f zapišemo množico podatkov pri določeni napaki za katero smo posneli odzive in izračunali kovariančno matriko. Matrika s katero transformiramo opazovane podatke vsebuje le glavne lastne vektorje matrike \underline{P}_f

$$\underline{q}_k = \left(\underline{x}_k - \underline{t}_{fk} \underline{P}_f^T \right)^T \left(\underline{x}_k - \underline{t}_{fk} \underline{P}_f^T \right). \quad (7.38)$$

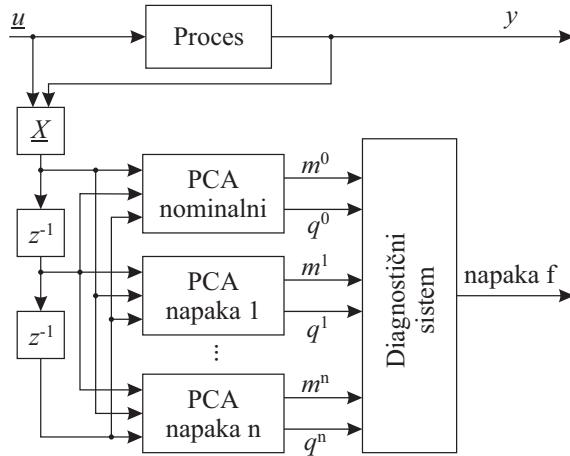
Na sliki 7.6 je predstavljena shema za zaznavanje in izolacijo napak. Trenutne podatke primerjamo z referenčnimi modeli posameznih napak in normalnega delovanja. Referenčni modeli so zapisani v obliki kovariančnih matrik in srednjih vrednosti posameznih merjenih spremenljivk.

Na nelinearnem simulacijskem modelu treh zbiralnikov smo posneli vhode in odzive v širši okolici delovne točke tako, da smo zajeli tudi nelinearnosti. Čas vzorčenja smo določili tako, da ta ni presegal 0.1 časa vzpona (5s) pri stopničastem vzbujanju. S postopkom, opisanim v podoglavlju 15.4.2, smo določili potrebno število (l) podvojenih, a zakasnjenih stolpcev v matriki podatkov \underline{X} . Celoten postopek načrtovanja je viden iz tabele 1.

m	l	s	r
5	0	4	1
10	1	4	4
15	2	4	0

Tabela 7.1. Postopek načrtovanja dinamične PCA, kjer r pomeni število novo odkritih kolinearnih stolpcev.

Ugotovimo, da je za načrtovanje dinamične PCA povsem dovolj prvi red ($l = 1$), saj za $l > 1$ ne odkrijemo nobene nove relacije več. Za vsak način delovanja (glej sliko 7.6) določimo



Slika 7.6. Shema za zaznavanje in izoliranje napak.

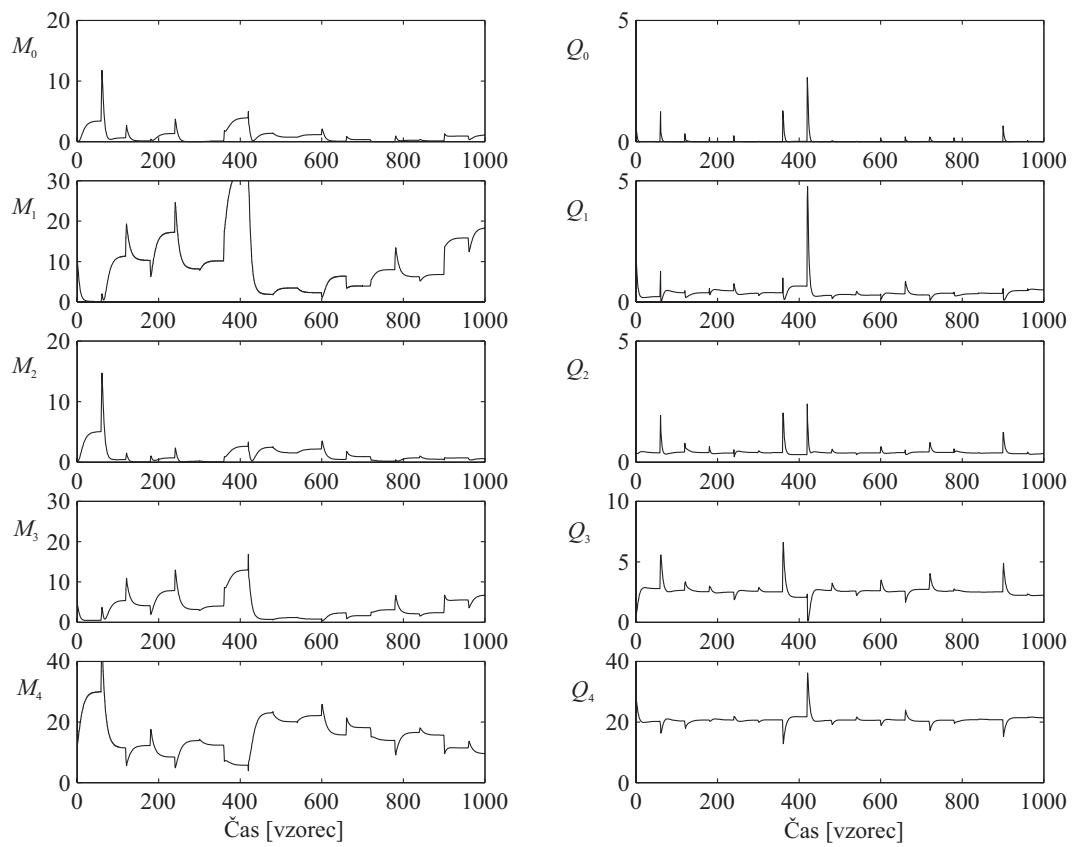
svoj model PCA, ki je zapisan v obliki kovariančne matrike in srednjih vrednosti posameznih merjenih spremenljivk. Iz tabele 7.1 razberemo, da prostor zadetkov povsem opišemo s štirimi glavnimi komponentami, ki pokrijejo več kot 95 odstotkov variance vhodnih podatkov. Za izračun mere M smo določili transformacijsko matriko lastnih vektorjev, kjer smo izbrali $s = 3$. Na sliki 7.7 so prikazani časovni poteki obeh mer (M_f in Q_f), izračunanih za vse znane prototipne načine delovanja, ki so opisani z različnimi modeli PCA. Za vse znane načine delovanja, oziroma napak, smo izmerili podatke in izračunali PCA modele. Znani načini delovanja so bili normalno delovanje ($f = 0$), puščanje zbiralnika T_1 ($f = 1$), pristranskost senzorja nivoja v zbiralniku T_1 ($f = 2$), obloge v črpalki P_2 ($f = 3$) ter pristranskost senzorja nivoja v zbiralniku T_3 ($f = 4$). Na sliki 7.7 vidimo uspešnost postopka, saj je večji odklon od ničelne vrednosti povsod, razen pri Q_0 in M_0 , iz česar sklepamo, da proces deluje pravilno. Koničasti skoki so posledica nelinearnosti procesa, ki ga aproksimiramo z linearnim PCA modelom. Da bi preverili pravilnost delovanja tudi pri drugih načinih delovanja, smo posneli odziv procesa na enako vzbujanje kot na sliki 7.7, le da smo med delovanjem procesa simulirali različne napake (glej tabelo 7.2). Slika 7.8 prikazuje omenjeni primer. Na sliki 7.8 opazimo, da zavzamejo mere najmanjšo vrednost pri

Nastop napake [vzorec]	Vrsta napake
100-200	pristranskost senzorja v T_3 (3 cm)
300-400	puščanje zbiralnika T_1
500-600	obloge v črpalki P_2
700-800	pristranskost senzorja v T_1 (5 cm)

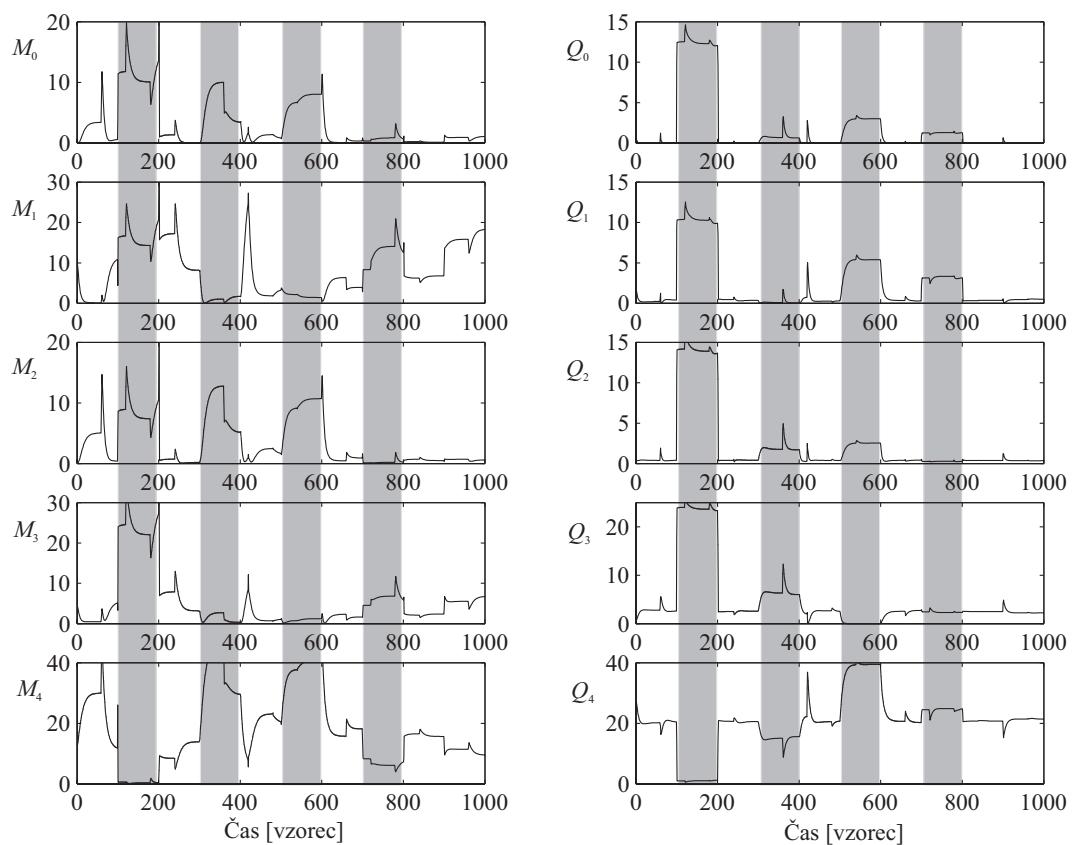
Tabela 7.2. Časovna razporeditev napak na sliki 7.8.

tistih indeksih, ki ustrezajo trenutni napaki v delovanju. Pri nastopu napake pristranskosti senzorja nivoja v zbiralniku T_3 (100 – 200 vzorec) zavzameta meri z indeksom $f = 4$ najmanjšo vrednost, druge pa se močno odklonijo. Napako izoliramo (poimenujemo) s tem, da poiščemo indeks mere, ki ima najmanjšo vrednost. To pomeni, da je trenutno delovanje najbolj podobno modelu s tem indeksom. Vse generirane napake lahko odkrijemo le z opazovanjem mere Q . Mera M (Mahalanobisova razdalja) torej služi kot dodatna.

Sistem za odkrivanje napak je neuporaben, če ni sposoben detektirati napačnega delovanja procesa pri nastopu neznane napake, torej tiste, katere pri načrtovanju še nismo poznali. Pri-

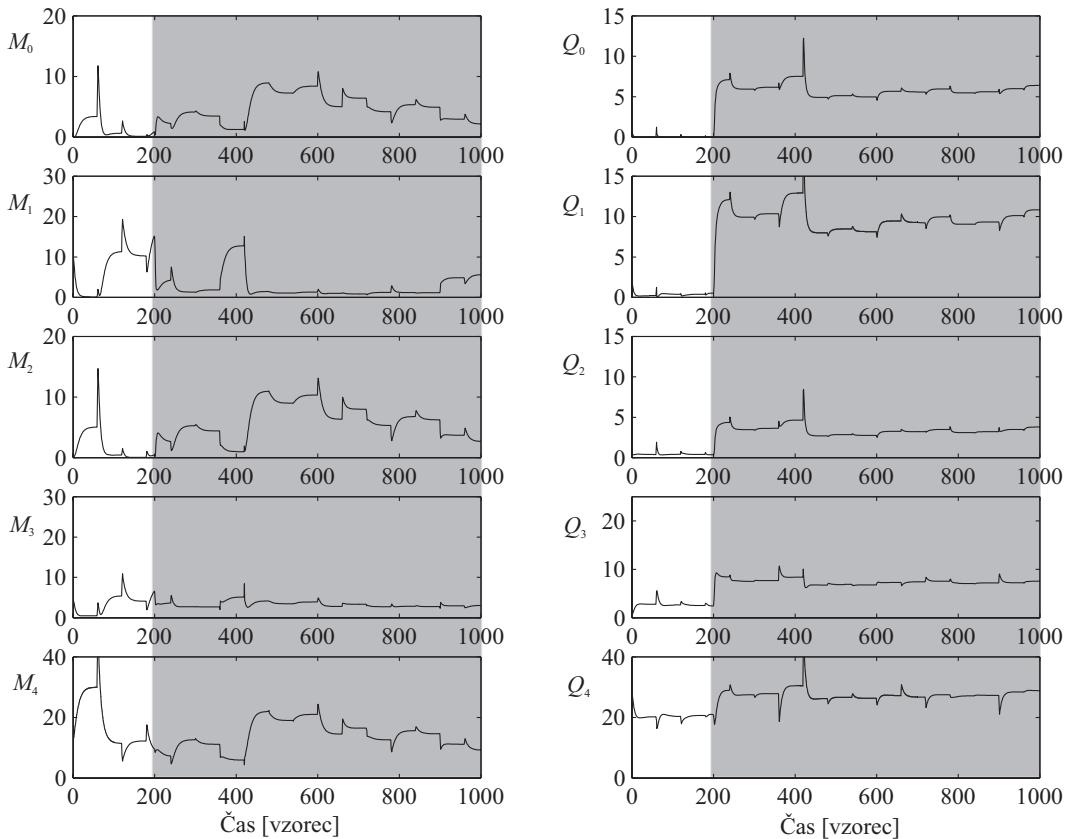


Slika 7.7. Mere pri normalnem delovanju procesa za primer vrednotenja.



Slika 7.8. Mere pri napakah v delovanju (področje delovanja napak je označeno zasenčeno).

mer nastopa neznane napake ob dvestotem vzorcu prikazuje slika 7.9 (delna zamašitev ventila V_1). Vse mere se ob nastopu te neznane napake močno odklonijo ozziroma ostanejo odklonjene. Sklepamo lahko, da se je zgodila neznana napaka in jo kasneje, ko nam jo uspe izolirati in identificirati, vključimo v naš sistem za odkrivanje napak, kar pomeni, da model te motnje dodamo med prototipne modele.



Slika 7.9. Nastop neznane napake v delovanju procesa (področje delovanja napake je označeno zasenčeno).

7.3 Tehnike rojenja

S tehnikami rojenja iščemo skupine podobnih podatkov. Roj je definiran kot skupina podatkov, ki so si med seboj bolj podobni, kot so si podobni s podatki iz drugih rojev [9]. Slika 7.10 prikazuje štiri primere. Uporabnik mora z mero podobnosti definirati kakšne roje bo poskušal najti (*ang. similarity measure*). Najbolj običajne oblike rojev so okrogle ali hiperkrogle, če govorimo o več dimenzionalnih problemih. V tem primeru je mera podobnosti razdalja točk do središča (centra) roja. Zato mu rečemo tudi prototip roja (*ang. prototype*). Za druge mere podobnosti pa lahko za prototipe dobimo tudi premice, kot je to prikazano na sliki 7.10b, ali pa kroge ozziroma elipse, kot je to na sliki 7.10c in d.

Pričakovana oblika rojev vpliva na izbiro algoritma rojenja. Prav tako mora uporabnik sam določiti število rojev. Bolj napredne metode same določijo število rojev glede na mero granulacije (*ang. granulation*), ki jo ravno tako poda uporabnik. V nadaljevanju bodo predstavljene samo nehierarhične metode. Hierarhične metode temeljijo na gnezdeni strukturi rojev.

Klasične metode rojenja, tako kot metoda K-povprečij, priredijo določen vzorec (poglavlje 7.5) samo enemu roju. Govorimo o ostrih particijah ali rojih. V novejših metodah pa se pojavi mehke particije ali roji, kjer vzorci pripadajo določenemu roju z določeno mero pripadnosti $\mu(\underline{x})$ (*ang. degree of membership*). Pripadnosti so normirane in zato je vsota pripadnosti za vsak vzorec enaka 1.

Algoritmi rojenja začno z naključno s pripadnostjo vzorcev določenim rojem ali z naključno izbranimi središči rojev. V splošnem konvergenca k globalnemu optimumu ni zagotovljena, občutljivost na izbiro začetnih pogojev pa je odvisna od izbrane metode rojenja.

Druga zelo pomembna stvar, ki se nanaša na izbiro mere podobnosti, je normiranje podatkov. Večina algoritmov je zelo občutljiva na velikostni razred podatkov. Na primer metoda rojenja, ki išče okrogle roje je zelo občutljiva na skaliranje, ker to spremeni kroge v elipse. Nenormirani podatki, npr. dve spremenljivki, prva v območju $0 < u_1 < 1$ in druga v območju $0 < u_2 < 1000$, zelo vplivata na kvaliteto rojenja, ker je izračun razdalj pri spremenljivki u_1 skoraj irelevanten glede na spremenljivko u_2 . Zaradi tega moramo podatke normirati preden uporabimo tehnike rojenja. Izjema so metode, ki adaptivno prilagajajo mere podobnosti ali normirajo podatke, kot je to metoda rojenja Gustafson-Kessel. Vendar je tudi v teh primerih priporočljivo normiranje, saj vodi do numerično bolje pogojenih rešitev.

Pomembne lastnosti metod rojenja, po katerih se med seboj razlikujejo, so naslednje:

- tip spremenljivk za katere je metoda primerna (zvezne, celoštevilske, binarne),
- možne mere podobnosti,
- hierarhičnostje algoritma,
- določeno ali samoadaptivno število rojev,
- dobimo ostre ali mehke roje.

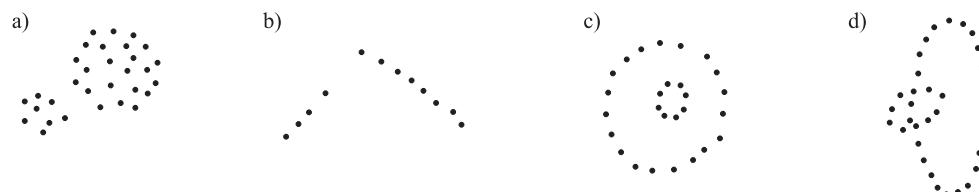
Več lahko najdete v literaturi [1, 4, 45, 102].

7.4 Mere podobnosti

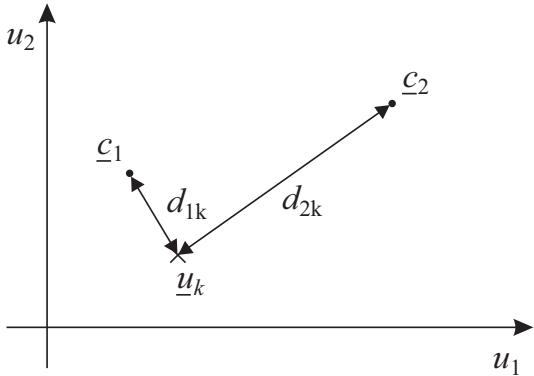
Mero podobnosti definiramo kot razdaljo med podatki. Razdalje so lahko zelo različne in glede na to ločimo tudi različne metode rojenja. Z izračunom razdalje lahko ugotovimo, kako daleč sta si dva vzorca glede na določeno mero, oziroma kako sta si podobna. Formalno mora funkcija razdalje med dvema točkama multidimenzionalnega prostora zadostiti naslednjim pogojem:

- $d(\underline{x}, \underline{y})$ je skalarna funkcija dveh spremenljivk,
- je pozitivna funkcija $d(\underline{x}, \underline{y}) \geq 0, \forall \underline{x}, \underline{y}$,
- $d(\underline{x}, \underline{x}) = 0, \forall \underline{x}$,
- in velja simetrija $d(\underline{x}, \underline{y}) = d(\underline{y}, \underline{x})$.

V nadaljevanju se bomo omejili na zvezne spremenljivke. Mera podobnosti med dvema podatkoma ali med dvema točkama prostora je mera ali norma vektorja, ki te dve točki povezuje. Normo največkrat definiramo kot razliko med njunima krajevnima vektorjem.



Slika 7.10. Primeri različnih oblik rojev: a) krog, b) premica, c) krožnica, d) krivulja, elipse, [4].



Slika 7.11. Prikaz razdalj pri mehkem rojenju s c-povprečji. Razdalje so lahko dobljene z različnimi normami.

Vsaka norma ima določene posebne lastnosti. Osnovna norma je posplošena norma Minkowskega, ki jo zapišemo kot $\|\underline{x}\|_p$, kjer je $p \geq 1$. Norma Minkowskega je definirana z enačbo 7.39

$$\|\underline{x}\|_p = \left(\sum_{i=1}^m |\underline{x}_i|^p \right)^{\frac{1}{p}}, \quad p \geq 1. \quad (7.39)$$

Če je $p = 2$, dobimo najbolj znano Evklidovo normo (L_2), če je $p = 1$ dobimo Manhattansko ali Hammingovo normo (L_1), če pa je $p \rightarrow \infty$ potem je to Čebiševljeva norma (L_∞).

Iz lastnosti posameznih norm sledijo oblike rojev, ki jih s pomočjo norme lahko odkrijemo. Evklidova norma je primerna za radialno obliko rojev. Zelo pogosta generalizacija norm je tudi generalizirana kvadratna norma, ki je definirana kot

$$d(\underline{x}, \underline{y}) = (\underline{x} - \underline{y})^T \underline{A}^{-1} (\underline{x} - \underline{y}), \quad (7.40)$$

kjer je \underline{A} pozitivno definitna matrika, oziroma norma v notranjem produktu razdalje. Z izbiro matrike lahko definiramo elipsoidno obliko rojev in vplivamo na njene dimenzije in rotacijo. Glede na izbiro matrike \underline{A} lahko ločimo nekaj osnovnih algoritmov rojenja, ki si jih bomo ogledali v nadaljevanju. Najbolj pogosta izbira matrike \underline{A} je kovariančna matrika.

Pregled norm, ki jih uporabljamo v rojenju, je v tabeli 7.3.

7.5 Algoritem rojenja s k-povprečji

Metoda rojenja s k-povprečji (*ang. k-means*) je najbolj običajna in enostavna metoda, ki pa je osnova vsem ostalim metodam. Ime k-povprečij, oziroma c-povprečji se pojavi prvič v [1], ker je to okrajšava za konstantno število rojev, ki ga vnaprej določeno.

Algoritem optimira naslednjo kriterijsko funkcijo:

$$I(\underline{v}) = \sum_{i=1}^c \sum_{k \in \mathcal{S}_i} \|\underline{x}(k) - \underline{v}_i\|^2 \longrightarrow \min_{\underline{v}_i}, \quad (7.41)$$

kjer indeks k definira elemente roja \mathcal{S}_i , c je število rojev in \underline{v}_i središče ali prototip roja. Množica \mathcal{S}_i vsebuje vse indekse tistih vzorcev (izmed vseh N), ki pripadajo roju i , to pomeni, da so najbližje središču roja \underline{v}_i . Središča rojev so optimizacijski parametri, ki jih algoritem spreminja tako, da minimizira kriterijsko funkcijo v enačbi 7.41. Kriterijska funkcija vsebuje vsoto vseh kvadratičnih odstopanj med središči rojev in vzorci, ki temu roju pripadajo. Zato jo lahko zapišemo tudi kot

Tabela 7.3. Izbrane norme.

Norme	
Norma Minkowskega	$d(\underline{x}, \underline{y}) = (\sum_{i=1}^m x(k) - y(k) ^p)^{\frac{1}{p}}, p \geq 1$
Evklidova norma	$d(\underline{x}, \underline{y}) = (\sum_{k=1}^m (x(k) - y(k))^2)^{\frac{1}{2}}$
Generalizirana kvadratna norma	$d(\underline{x}, \underline{y}) = (\underline{x} - \underline{y})^T \underline{A}^{-1} (\underline{x} - \underline{y})$
Hammingova	$d(\underline{x}, \underline{y}) = \sum_{k=1}^m x(k) - y(k) $
Norma Čebiševa	$d(\underline{x}, \underline{y}) = \max_{k=1, \dots, m} x(k) - y(k) $
Canberraska norma	$d(\underline{x}, \underline{y}) = \sum_{k=1}^m \frac{ x(k) - y(k) }{x(k) + y(k)}, x(k), y(k) > 0$
Kotna norma	$d(\underline{x}, \underline{y}) = \frac{\sum_{k=1}^m x(k)y(k)}{\sum_{k=1}^m x^2(k) \sum_{k=1}^m y^2(k)}$

$$I(\underline{v}) = \sum_{i=1}^c \sum_{k=1}^N \mu_{ik} \|\underline{x}(k) - \underline{v}_i\|^2, \quad (7.42)$$

kjer je $\mu_{ik} = 1$, če je vzorec $\underline{x}(k)$ element roja i , in $\mu_{ik} = 0$, če ni element.

Algoritem 5 Psevdokoda rojenja s k-povprečji.

1: Inicializacija števila rojev c in središč rojev \underline{v}_i , $i = 1, \dots, c$. To lahko naredimo tudi z naključno izbiro c vzorcev.

2: **repeat**

3: Uvrstitev vzorcev v roje z najbližjim središčem.

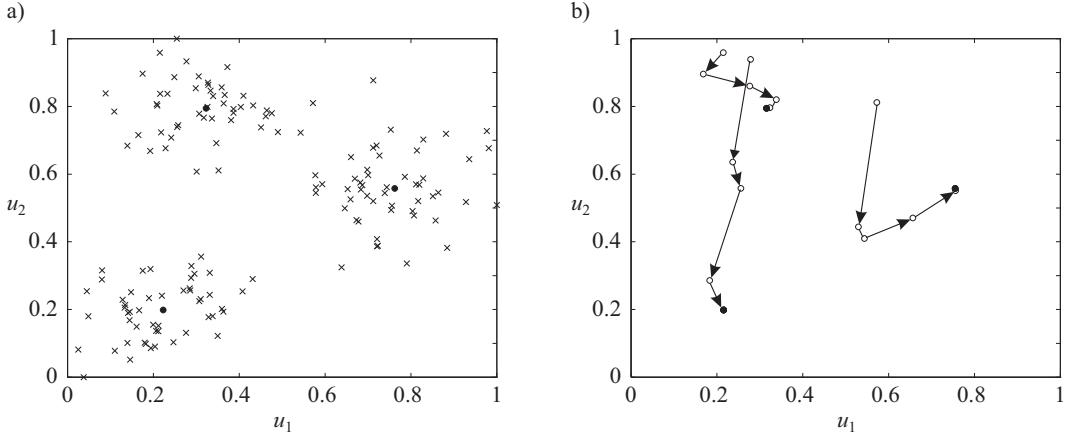
4: Izračun središč rojev:

$$\underline{v}_i = \frac{\sum_{k \in \mathcal{S}_i} \underline{x}(k)}{N_i}, \quad (7.43)$$

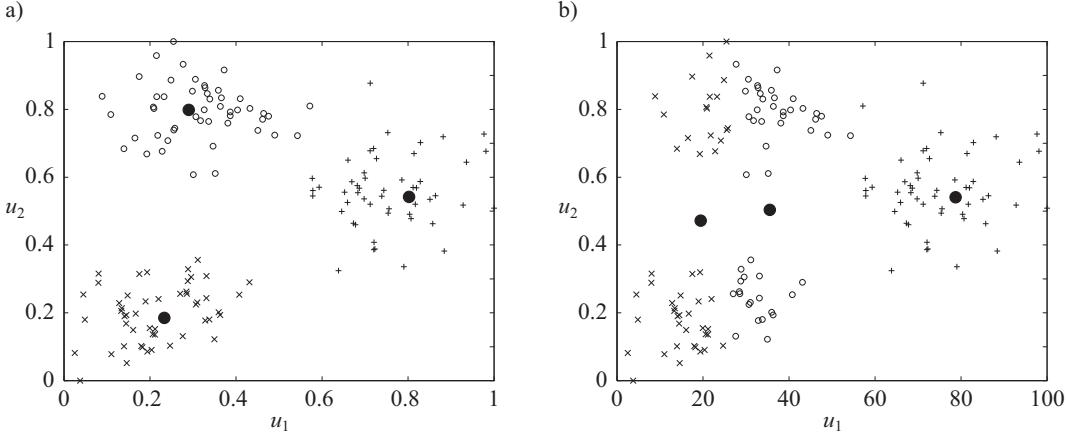
kjer je i indeks vzorcev, ki pripadajo i -temu, so elementi \mathcal{S}_i , in je N_i število elementov v \mathcal{S}_i ($\sum_{i=1}^c N_i = N$)

5: **until** $\|\underline{v}_{new} - \underline{v}\|_\infty < min_{impro}$

Slika 7.12 prikazuje konvergenco metode rojenja s k -povprečji za $c = 3$ roje. Slika 7.12a prikazuje dvodimensionalno množico podatkov, slika 7.12b pa prikazuje 3 središča rojev po petih iteracijah, ki so potrebne za dosego konvergencije. Začetna središča rojev so bila izbrana naključno iz množice podatkov. Ne glede na slabo inicializacijo, je konvergencia algoritma zelo hitra.



Slika 7.12. a) Rojenje z metodo rojenja s k-povprečji b) Prikaz položajev središč rojev skozi 5 iteracij. Črni krogci predstavljajo končne položaje središč rojev.



Slika 7.13. a) Primerjava med normiranimi in b) nenormiranimi podatki v primeru metode s k-povprečji. Na sliki b) so vzorci prijejeni rojem izključno glede na odstopanja po spremenljivki u_1 , ker vrednosti po tej spremenljivki domirajo v meri razdalje.

Slika 7.13 prikazuje pomen normiranja podatkov. Slika 7.13a prikazuje pripadnost normiranih vzorcev posameznim rojem. Slika 7.13b pa prikazuje roje za nenormirane podatke, kjer u_1 leži med 0 in 100 in u_2 med 0 in 1. Pri rojenju v primeru nenormiranih podatkov prevladuje vpliv spremenljivke u_1 in zato so rezultati odvisni predvsem od te spremenljivke.

Alternativa normiranju podatkov je uporaba druge mer za podobnost. Kvadratično Evklidovo normo, ki je uporabljena v enačbi 7.42

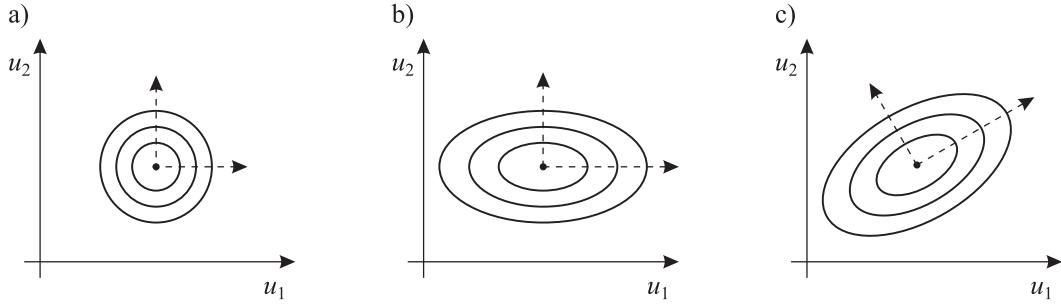
$$d_{ik}^2 = \|\underline{x}(k) - \underline{v}_i\|^2 = (\underline{x}(k) - \underline{v}_i)^T (\underline{x}(k) - \underline{v}_i), \quad (7.44)$$

lahko razširimo v splošno Mahalanobisovo normo

$$d_{ik,\Sigma}^2 = \|\underline{x}(k) - \underline{v}_i\|_{\Sigma}^2 = (\underline{x}(k) - \underline{v}_i)^T \Sigma^{-1} (\underline{x}(k) - \underline{v}_i). \quad (7.45)$$

Matrična norma Σ skalira in rotira koordinatne osi. V posebnem primeru, ko je $\Sigma = I$ je Mahalanobisova norma enaka Evklidovi normi. Za

$$\Sigma^{-1} = \begin{bmatrix} 1/\sigma_1^2 & 0 & \cdots & 0 \\ 0 & 1/\sigma_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1/\sigma_p^2 \end{bmatrix}, \quad (7.46)$$



Slika 7.14. Črte enake razdalje pri različnih normah: a) Evklidova ($\Sigma = I$), b) diagonalna ($\Sigma = \text{diagonalna}$) in c) Mahalanobisova norma ($\Sigma = \text{splošna notranja matrična norma}$).

kjer je p dimenzija prostora podatkov je Mahalanobisova norma enaka Evklidovi normi z normiranimi spremenljivkami $x_l^{(\text{norm})} = x_l / \sigma_l$. Slika 7.14 prikazuje mere razdalj. Kriterijska funkcija v enačbi 7.42 je enaka v obeh primerih. Norma je enaka za celoten prostor podatkov v primeru metode k-povprečij. V primeru metode Gustafson-Kessel pa imamo za vsak roj svojo kovariančno matriko in individualno mero.

7.6 Metode mehkega rojenja s točkastimi središči

V tem poglavju bo predstavljeno mehko rojenja kot eden od pri osnovnih principov pri raziskovanju podatkov v smislu razvrščanja in stiskanja (zgoščanja) podatkov. Predstavljeni bodo primeri mehkega rojenja z različnimi metodami, kjer je središče roja točka v prostoru podatkov.

7.6.1 Mehko rojenje s c-središči (FCM)

Najprej bo predstavljena osnovna metoda mehkega rojenja, mehko rojenje s c-središči (*ang. Fuzzy C-Means clustering*).

Predpostavimo, da imamo množico podatkov, ki smo jih dobili z meritvami m različnih spremenljivk v n različnih časovnih trenutkih. Meritve v enem časovnem trenutku sestavljajo vzorec, ki ga zapišemo kot m -dimenzionalni vektor $\underline{x}^T(k) = [x_1(k), \dots, x_m(k)]$, $\underline{x}(k) \in \mathbb{R}^m$, in kjer je $x_m(k)$ meritev m -tega senzorja v časovnem trenutku k . Celotno množico podatkov, kjer imamo n vzorcev meritev, lahko zapišemo kot $\underline{X} = \{\underline{x}^T(k) \mid k = 1, 2, \dots, n\}$, $\underline{X} \in \mathbb{R}^{n \times m}$. Vektor $\underline{x}(k)$ bomo na mestih, kjer bo to zaradi notacije enostavnejše, pisali tudi kot \underline{x}_k .

Glavni cilj rojenja je v razdelitvi množice podatkov na podmnožice podatkov, ki vsebujejo elemente, ki so si med seboj bolj podobni kot so si podobni s tistimi iz drugih rojev. Podatke, ki jih bomo razdelili v c rojev zapišemo v matriko \underline{X} , kot sledi

$$\underline{X} = \begin{bmatrix} x_1(1) & x_2(1) & \dots & x_m(1) \\ x_1(2) & x_2(2) & \dots & x_m(2) \\ \vdots & \vdots & \vdots & \vdots \\ x_1(n) & x_2(n) & \dots & x_m(n) \end{bmatrix}. \quad (7.47)$$

Mehke podmnožice (roje) celotne množice \underline{X} zapišemo kot $\{A_i \mid 1 \leq i \leq c\}$. Mehke podmnožice so definirane s pripadnostjo določenega vzorca meritev določeni mehki podmnožici s pripadnostjo $\mu_i(k)$, oziroma μ_{ik} , ki je na intervalu $[0, 1]$. Če zapišemo pripadnosti za vse roje in za vse vzorce meritev, potem dobimo matriko mehkih pripadnosti $U = [\mu_{ik}] \in \mathbb{R}^{c \times n}$. V matriki pripadnosti so v i -ti vrstici pripadnosti vzorcev meritev ($k = 1, \dots, n$) i -ti mehki podmnožici A_i . Matrika pripadnosti ima naslednje lastnosti:

– mere pripadnosti so realna števila na intervalu

$$\mu_{ik} \in [0, 1], \quad 1 \leq i \leq c, \quad 1 \leq k \leq n;$$

– vsota vseh pripadnosti posameznega vzorca meritev posameznim mehkim podmnožicam je enaka ena

$$\sum_{i=1}^c \mu_{ik} = 1, \quad 1 \leq k \leq n;$$

– nobena mehka množica ni prazna in tudi nobena ne vsebuje vseh podatkov

$$0 < \sum_{k=1}^n \mu_{ik} < n, \quad 1 \leq i \leq c.$$

Če matrika U izpolnjuje vse zgornje lastnosti, potem pripada mehki množici, ki je definirana kot

$$M = \{U \in \mathbb{R}^{c \times n} \mid \mu_{ik} \in [0, 1], \forall i, k; \sum_{i=1}^c \mu_{ik} = 1, \forall k; 0 < \sum_{k=1}^n \mu_{ik} < n, \forall i\}. \quad (7.48)$$

Algoritem mehkega rojenja s c-povprečji temelji na razdalji med izbranim vzorcem podatkov in središčem mehke podmnožice $d(\underline{x}(k), \underline{v}_i)$, kjer je $\underline{x}(k)$ izbrani vzorec podatkov $\underline{x}(k) \in \mathbb{R}^m$ in \underline{v}_i središče i -te mehke množice. Cilj algoritma rojenja je v minimizaciji kriterijske funkcije (*ang. objective function*), ki je definirana kot

$$I(\underline{X}, \underline{U}, \underline{V}) = \sum_{i=1}^c \sum_{k=1}^n (\mu_{ik})^\eta d^2(\underline{x}(k), \underline{v}_i), \quad (7.49)$$

pri naslednjih pogojih (omejitvah)

$$\sum_{i=1}^c \mu_{ik} = 1 \quad \forall k, \quad (7.50)$$

kjer je \underline{V} matrika sestavljena iz vektorjev, ki predstavljajo središča rojev \underline{v}_i , $\underline{V} = [\underline{v}_1, \dots, \underline{v}_c]^T$, η je parameter, ki definira mehkost prehodov iz ene množice v drugo (od popolnoma ločenih množic $\eta \rightarrow 1$ do popolnoma mehkih podmnožic $\eta \rightarrow \infty$). V naši obravnavi bo privzeta vrednost $\eta = 2$. Za mero razdalje pa izberemo Evklidovo normo.

Iskanje mehkih rojev v množici podatkov \underline{X} rešimo z uporabo nelinearne optimizacijske metode z omejitvami, metode Lagrangeovih koeficientov, ki znotraj domene \underline{X} minimizira kriterijsko funkcijo iz enačbe 7.49 ob upoštevanju pogojev, ki so dani v enačbi 7.50.

Kriterijsko funkcijo ob upoštevanju omejitev lahko sedaj zapišemo v obliki Lagrangeovega funkcionala, ki ga rešujemo z uporabo Picardove iteracijske metode

$$I(\underline{X}, \underline{U}, \underline{V}, \lambda) = \sum_{i=1}^c \sum_{k=1}^n (\mu_{ik})^\eta (\underline{x}(k) - \underline{v}_i)^T (\underline{x}(k) - \underline{v}_i) + \sum_{k=1}^n \lambda_k \sum_{i=1}^c (\mu_{ik} - 1), \quad (7.51)$$

kjer so λ_k , $k = 1, \dots, n$ Lagrangeovi koeficienti. Minimum kriterijske funkcije $I(\underline{X}, \underline{U}, \underline{V}, \lambda)$ dobimo po iterativnem postopku optimizacije. Mere pripadnosti izračunamo kot normirane razdalje od vzorca do središč. Izračun je naslednji

$$\mu_{ik} = \left(d_{ik}^2 \sum_{j=1}^c \left(\frac{1}{d_{jk}^2} \right)^{\frac{1}{\eta-1}} \right)^{-1}, \quad (7.52)$$

ali

$$\mu_{ik} = \frac{\frac{1}{d_{ik}^w}}{\sum_{j=1}^c \frac{1}{d_{jk}^w}}, \quad (7.53)$$

kjer je $w = \frac{2}{\eta-1}$, d_{ik} je Evklidova razdalja med vzorcem meritev $\underline{x}(k)$ in centrom i -tega roja \underline{v}_i in je definirana kot sledi

$$d_{ik}^2 = (\underline{x}(k) - \underline{v}_i)^T (\underline{x}(k) - \underline{v}_i), \quad 1 \leq i \leq c, \quad 1 \leq k \leq n. \quad (7.54)$$

Središče roja \underline{v}_i dobimo kot uteženo povprečje, kjer so uteži pripadnosti vzorca meritev i -temu roju

$$\underline{v}_i = \frac{\sum_{k=1}^n \mu_{ik}^\eta \underline{x}(k)}{\sum_{k=1}^n \mu_{ik}^\eta}, \quad i = 1, \dots, c. \quad (7.55)$$

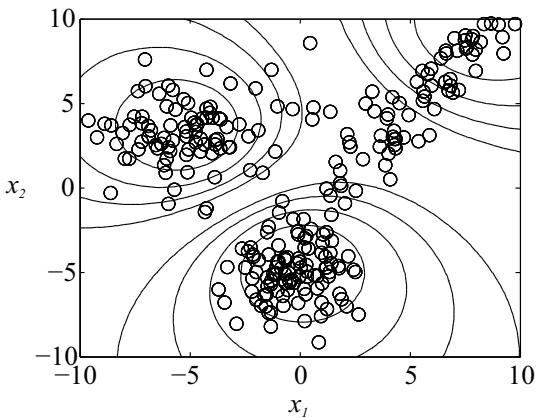
V tabeli 7.6.1 je po korakih predstavljen iterativni algoritem mehkega rojenja s c -povprečji. Opozoriti moramo na notacijo, na primer $\mu_{ik}^{(r)}$, ki pomeni izračun pripadnosti μ_{ik} v r -ti iteraciji.

Algoritem 6 Psevdokoda mehkega rojenja c-središč.

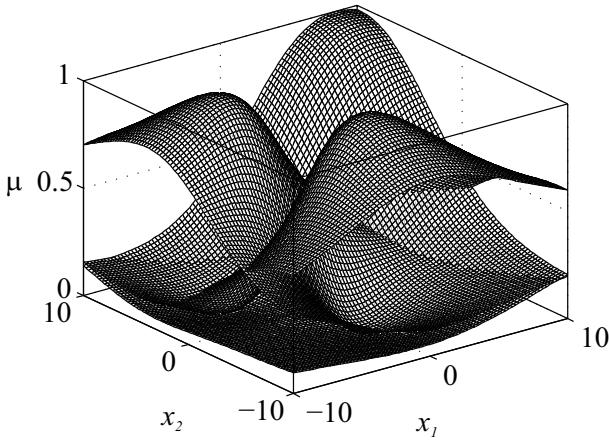
- 1: Inicializacija števila rojev c
 - 2: Izbira uteži η
 - 3: Izbira tolerančnega praga za zaključek iteracije min_{impro} in izbira maksimalnega števila iteracij r_{max}
 - 4: Inicializacija matrike pripadnosti z naključnimi vrednostmi $U^0 \in M$
 - 5: **repeat**
 - 6: Izračun središč rojev: $\underline{v}_i^{(r)} = \frac{\sum_{k=1}^n (\mu_{ik}^{(r)})^\eta \underline{x}(k)}{\sum_{k=1}^n (\mu_{ik}^{(r)})^\eta}, \quad i = 1, \dots, c$
 - 7: Izračun razdalj: $d_{ik}^2 = (\underline{x}(k) - \underline{v}_i^{(r)})^T (\underline{x}(k) - \underline{v}_i^{(r)}), \quad i = 1, \dots, c, \quad k = 1, \dots, n$
 - 8: Izračun matrike pripadnosti: $\mu_{ik}^{(r)} = \frac{\frac{1}{d_{ik}^w}}{\sum_{j=1}^c \frac{1}{d_{jk}^w}}, \quad w = \frac{2}{\eta-1}$
 - 9: **until** $\|\underline{U}^{(r)} - \underline{U}^{(r-1)}\|_\infty < min_{impro}$ ali $r \geq r_{max}$
-

- **Opažanje 1.** Algoritem lahko pride v točko singularnost, če je $d_{ik} = 0$ v primeru k -te meritve in i -tega roja. V takem primeru zaradi singularnosti ne moremo izračunati pripadnosti μ_{ik} . Problem rešimo tako, da algoritem dopolnimo s pogojem, ki v takem primeru vsem točkam, kjer je $d_{ik} = 0$, določi pripadnost, ki je enaka ena (ob izpolnjevanju pogoja $\sum_{i=1}^c \mu_{ik} = 1$).
- **Opažanje 2.** Kriteriji, po katerih zaključimo algoritem, so lahko različni: ($\|\underline{U}^{(r)} - \underline{U}^{(r-1)}\|_\infty < min_{impro}$, $\|\underline{V}^{(r)} - \underline{V}^{(r-1)}\|_\infty < min_{impro}$ ali $\max_{i,k} (|\mu_{ik}^{(r)} - \mu_{ik}^{(r-1)}|) < min_{impro}$).
- **Opažanje 3.** Faktor mehkosti η igra pomembno vlogo pri določevanju rojev. Z izbiro limitne vrednosti $\eta \rightarrow 1$ teoretično dobimo klasično rojenje z ostrimi množicami. Vendar moramo v praksi zaradi numeričnih problemov izbrati vrednost, ki je večja od ena. Kolikšna je ta vrednost, pa je odvisno od numerične zahtevnosti problema. V primeru, ko gre $\eta \rightarrow \infty$ pa dobimo popolno mehko prekrivanje prostora ($\mu_{ik} = \frac{1}{c}, \forall i, k$) in so središča enaka srednji vrednosti neutežene množice podatkov.
- **Opažanje 4.** Izbira števila rojev c je najpomembnejši parameter. Ustrezno število rojev c lahko določimo z večkratnim zaganjanjem algoritma.

Primer algoritma FCM. Izbera matrike $\underline{\Sigma}$, ki je enaka enotski matriki $\underline{I} \in \mathbb{R}^{m \times m}$, omogoča dobro razvrščanje podatkov, ki imajo radialno obliko, zato množice podatkov, ki imajo drugačno obliko, niso optimalno razvrščene. To lahko vidimo na sliki 7.15, kjer je množica podatkov, ki tvori eliptično obliko, le delno pripada roju. Boljšo pripadnost roju imajo podatki, ki so radialno razvrščeni okoli središča. Konturni diagram, ki povezujejo točke prostora s pripadnostmi od 0.5 do 0.9 s korakom 0.1, je prikazan na sliki 7.15. Na sliki 7.16 pa vidimo pripadnostne funkcije v 3D prostoru.



Slika 7.15. Primer rojenja z Evklidovo normo.



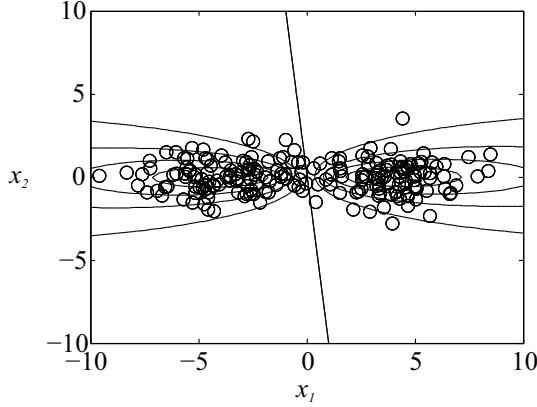
Slika 7.16. Prikaz pripadnostnih funkcij v 3D prostoru pri rojenju z Evklidovo normo.

Primer matrike \underline{A} z diagonalno inverznih varianc. Če izberemo matriko \underline{A} , ki ima na diagonali elemente, ki so inverzne vrednosti variance podatkov v posameznih smereh,

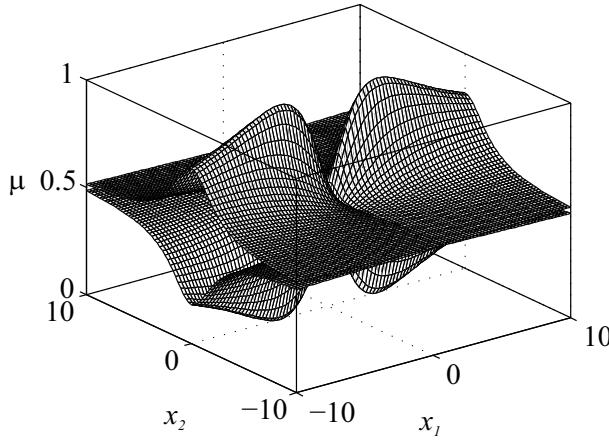
$$\underline{\Sigma}^{-1} = \begin{bmatrix} \frac{1}{\sigma_1^2} & 0 & \cdots & 0 \\ 0 & \frac{1}{\sigma_2^2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\sigma_m^2} \end{bmatrix}, \quad (7.56)$$

dobimo mero, ki omogoča dobro razvrščanje podatkov, ki imajo spremenljivke z različnimi variancami, kjer je varianca enaka $\sigma_i^2 = \frac{1}{n-1} \sum_{k=1}^n (x_i(k) - \mu_{x_i})^2$ in kjer je srednja vrednost

$\mu_{x_i} = \frac{1}{n} \sum_{k=1}^n x_i(k)$. Lastnosti te norme lahko vidimo na sliki 7.17, kjer vidimo, da množici podatkov, ki imata eliptično obliko in enake variance v obeh smereh, dobro pripadata vsaka svojemu roju. Na sliki 7.18 pa vidimo pripadnostne funkcije v 3D prostoru. Metoda pa pokaže



Slika 7.17. Primer rojenja z normo inverznih varianc.



Slika 7.18. Prikaz pripadnostnih funkcij v 3D prostoru pri rojenju z normo inverznih varianc.

slabosti, ko imamo med podatki roje, ki so različnih oblik, kar vidimo na sliki 7.19.

Primer Mahalanobisove norme. Mahalonobisova norma razdalje med dvema vektorjem je definirana kot

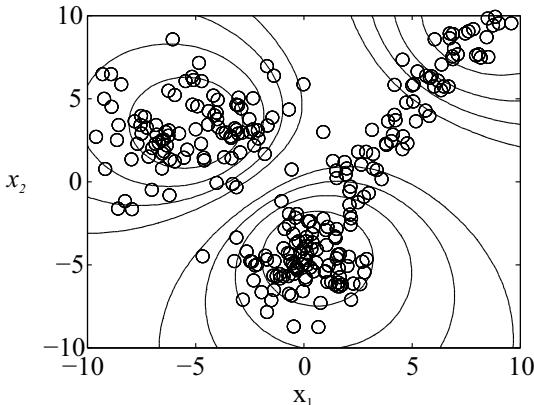
$$d(\underline{x}, \underline{y}) = (\underline{x} - \underline{y})^T \underline{\Sigma}^{-1} (\underline{x} - \underline{y}), \quad (7.57)$$

kjer je $\underline{\Sigma}$ kovariančna matrika

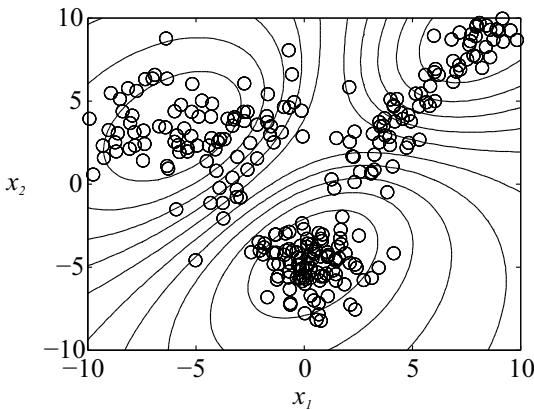
$$\underline{\Sigma} = \frac{1}{n-1} \sum_{k=1}^n (\underline{x}(k) - \underline{\mu}_x)(\underline{x}(k) - \underline{\mu}_x)^T, \quad (7.58)$$

in je $\underline{\mu}_x = [\mu_{x_1}, \dots, \mu_{x_m}]$ vektor srednjih vrednosti posameznih spremenljivk. Torej je v tem primeru norma notranjega produkta enaka $A = \underline{\Sigma}$.

Rezultate rojenja z Mahalanobisovo normo lahko vidimo na sliki 7.20, kjer vidimo, da množica podatkov, ki ima obliko rotiranih elips, dobro pripada roju. Na sliki 7.21 pa vidimo pripadnostne



Slika 7.19. Primer rojenja z normo inverznih varianc.



Slika 7.20. Primer rojenja z Mahalanobisovo normo.

funkcije v 3D prostoru. Vendar je norma $\underline{\Sigma}$ konstantna ne glede na obliko roja in zato v primeru različnih oblik rojev ne da optimalnega segmentiranja prostora podatkov.

7.6.2 Rojenje po metodi Gustafson-Kessel (GK)

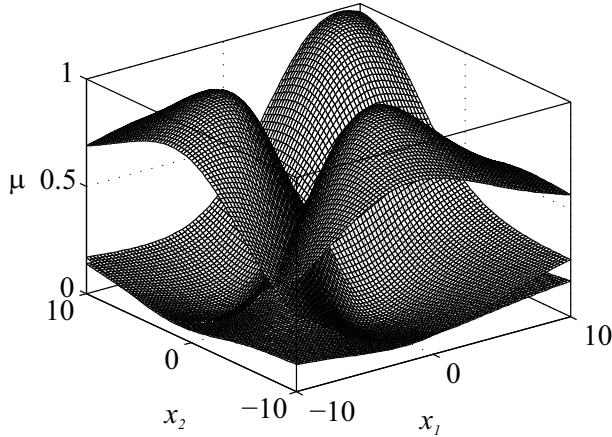
Slabost algoritma z matriko notranjega produkta $\underline{\Sigma}$ je v njeni konstantni vrednosti, kar one-mogoča detekcijo različnih oblik rojev v isti množici podatkov. Ena od poglavitnih prednosti pa je hitra konvergenca in robustnost algoritma.

Slabost konstantne matrike notranjega produkta odpravlja metoda rojenja Gustafson-Kessel, ki uvede spremenljivo matriko notranjega produkta $\underline{\Sigma}_i$, $i = 1, \dots, c$. Pri metodi GK ima vsak roj svojo matriko notranjega produkta

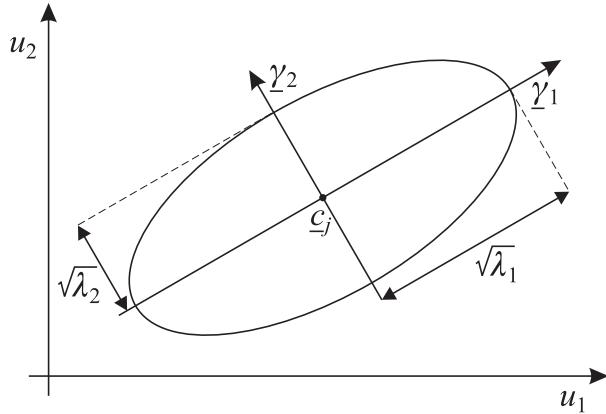
$$d_{ik}^2 = (\underline{x}(k) - \underline{v}_i^{(r)})^T \underline{F}_i (\underline{x}(k) - \underline{v}_i^{(r)}), \quad 1 \leq i \leq c, \quad 1 \leq k \leq n. \quad (7.59)$$

Matriko notranjega produkta za določen roj lahko izračunamo iz njegove mehke kovariančne matrike $\underline{\Sigma}_i$, $i = 1, \dots, c$, ki vsebuje informacijo o porazdeljenosti podatkov okoli središča roja. Normirana mehka kovariančna matrika je nato uporabljena kot matrika notranjega produkta pri izračunu razdalji d_{ik} . Mehka kovariančna matrika roja i , ki je definirana kot

$$\underline{\Sigma}_i = \frac{\sum_{k=1}^n \mu_{ik}^\eta (\underline{x}(k) - \underline{v}_i)^T (\underline{x}(k) - \underline{v}_i)^T}{\sum_{k=1}^n \mu_{ik}^\eta}, \quad i = 1, \dots, c, \quad (7.60)$$



Slika 7.21. Prikaz pripadnostnih funkcij v 3D prostoru pri rojenju z Mahalanobisovo normo.



Slika 7.22. Interpretacija lastnih vrednosti λ_l in lastnih vektorjev $\underline{\gamma}_l$, $l = 1, \dots, p$ mehke kovariančne matrike $\underline{\Sigma}_j$. Elipsa je konturni diagram funkcije $(\underline{u}(i) - \underline{c}_j)\underline{\Sigma}^{-1}(\underline{u}(i) - \underline{c}_j)^T = 1$. Ploščina (hipervolumen v primeru višjih dimenzij) je določena z lastnimi vrednostmi, orientacija pa se spreminja z lastnimi vektorji matrike $\underline{\Sigma}_j$, ki jih izračuna algoritem rojenja Gustafson-Kessel.

je dimenzijske $\underline{\Sigma}_i \in \mathbb{R}^{m \times m}$ in je simetrična, pozitivno definitna matrika.

Slika 7.22 prikazuje relacijo med merami razdalj, lastnimi vektorji in lastnimi vrednostmi mehke kovariančne matrike $\underline{\Sigma}_j$. Razdalje med podatki in lastnimi vektorji so namreč v primeru Mahalonobisova razdalje normirane s pripadajočimi lastnimi vrednostmi. Vse točke konturnega diagrama na sliki 7.22 imajo enake pripadnosti roju. Množica točk pa zaradi razmerja med lastnimi vrednostmi predstavlja elipso. Večje razmerje med $\lambda_{\max}/\lambda_{\min}$ določa bolj raztegnjeno elipso, razmerje blizu 1 pa rezultira v krožni obliko. Algoritmom rojenja Gustafson-Kessel lahko uporabimo za odkrivanje linearnih hiperravnin v množici podatkov, ki imajo eno zelo majhno lastno vrednost z lastnim vektorjem, ki je ortogonalen na hiperravnino, kjer leži večji del podatkov.

Pri izpeljavi algoritma Gustafson-Kessel s pomočjo Lagrangejevih koeficientov dobimo naslednjo relacijo za matriko notranjega produkta:

$$\underline{F}_i = (\rho_i |\underline{\Sigma}_i|)^{\frac{1}{m}} \underline{\Sigma}_i^{-1}, \quad (7.61)$$

kjer je ρ_i faktor, s katerim omejimo matriko \underline{F}_i in bo definiran v nadaljevanju, in $|\underline{\Sigma}_i|$ determinanta mehke kovariančne matrike.

V tabeli 7 je po korakih predstavljen algoritem mehkega rojenja po metodi Gustafson-Kessel.

Algoritem 7 Psevdokoda mehkega rojenja po metodi Gustafson-Kessel.

- 1: Inicializacija števila rojev c
- 2: Izberi uteži η
- 3: Izberi tolerančnega praga za zaključek iteracije \min_{impro} in izberi maksimalnega števila iteracij r_{max}
- 4: Inicializacija matrike pripadnosti z naključnimi vrednostmi $U^0 \in M$
- 5: Izberi začetne vrednosti faktorja $\rho_i^0 = 1$, $i = 1, \dots, c$
- 6: **repeat**
- 7: Izračun središč rojev: $\underline{v}_i^{(r)} = \frac{\sum_{k=1}^n (\mu_{ik}^{(r)})^\eta \underline{x}(k)}{\sum_{k=1}^n (\mu_{ik}^{(r)})^\eta}$, $i = 1, \dots, c$
- 8: Izračun mehke kovariančne matrike: $\underline{\Sigma}_i = \frac{\sum_{k=1}^n (\mu_{ik}^{(r-1)})^\eta (\underline{x}(k) - \underline{v}_i^{(r)}) (\underline{x}(k) - \underline{v}_i^{(r)})^T}{\sum_{k=1}^n (\mu_{ik}^{(r-1)})^\eta}$, $i = 1, \dots, c$
- 9: Izračun matrike notranjega produkta: $\underline{F}_i = \left(\rho_i^{(r-1)} |\underline{\Sigma}_i| \right)^{\frac{1}{m}} \underline{\Sigma}_i^{-1}$, $i = 1, \dots, c$
 $\rho_i^{(r)} = |F_i|$, $i = 1, \dots, c$
- 10: Izračun razdalj: $d_{ik}^2 = (\underline{x}(k) - \underline{v}_i^{(r)})^T \underline{F}_i (\underline{x}(k) - \underline{v}_i^{(r)})$, $i = 1, \dots, c$, $k = 1, \dots, n$
- 11: Izračun matrike pripadnosti: $\mu_{ik}^{(r)} = \frac{\frac{1}{d_{ik}^w}}{\sum_{j=1}^c \frac{1}{d_{jk}^w}}$, $w = \frac{2}{\eta-1}$
- 12: **until** $\|\underline{U}^{(r)} - \underline{U}^{(r-1)}\|_\infty < \min_{impro}$ ali $r \geq r_{max}$

- **Opažanje 1.** Algoritem Gustafson-Kessel je v določenih primerih numerično nestabilen, kar je največkrat posledica singularnosti mehke kovariančne matrike, ki jo je potrebno invertirati. Ta problem rešujejo modifikacije v izračunu mehke kovariančne matrike in invertiranje s pomočjo razcepa matrike v njene singularne vrednosti (SVD).
- **Opažanje 2.** Mehka kovariančna matrika daje informacijo o obliku in orientaciji roja. To informacijo dobimo, če razcepimo mehko kovariančno matriko na njene lastne vektorje in lastne vrednosti. Na osnovi ortonormalnega vektorja z najmanjšo lastno vrednostjo lahko dobimo model.

Primer mehkega rojenja po metodi Gustafson-Kessel. Primer podaja rojenje v množici podatkov z različnimi oblikami rojev in z podobnim številom podatkov znotraj enega roja. Mero števila podatkov znotraj enega roja seveda obravnavamo mehko, zato je volumen roja definiran kot

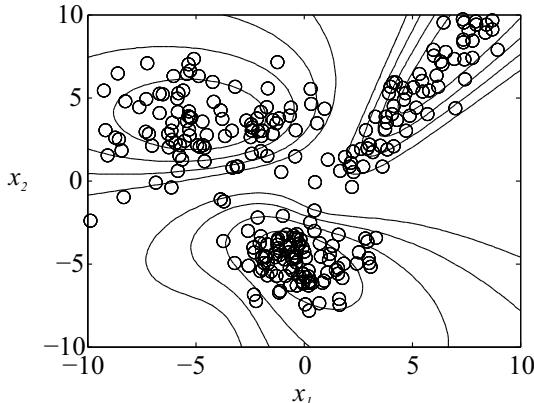
$$L_i = \sum_{k=1}^n \mu_{ik} . \quad (7.62)$$

Kadar pa govorimo o gostoti roja (ϱ_i) potem moramo volumen roja deliti s številom podatkov, kot to definira naslednja enačba

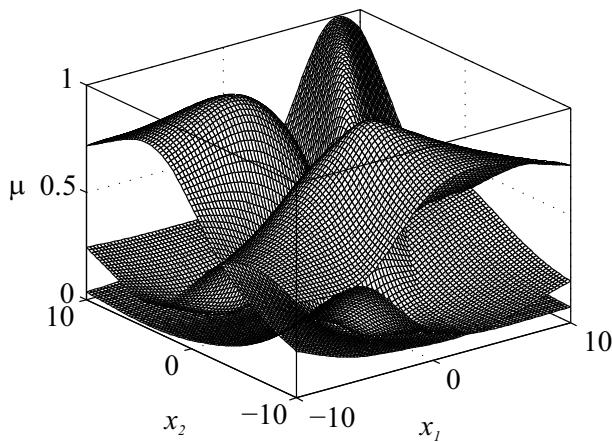
$$\varrho_i = \frac{1}{N} \sum_{k=1}^n \mu_{ik} . \quad (7.63)$$

Na sliki 7.23 vidimo, da algoritem dobro detektira različne oblike rojev in jih razvrsti. Pripadnosti posameznim rojem so prikazane v 3D prostoru na sliki 7.24.

Primer neenakih volumnov rojev. Primer podaja rojenje v množici podatkov z različnimi oblikami rojev, vendar pa so volumni rojev zelo različni. Na sliki 7.25 vidimo, da klasični algoritem Gustafson-Kessel v tem primeru ne daje najboljše razdelitve, saj teži k razdelitvi z enakimi



Slika 7.23. Primer mehkega rojenja po metodi Gustafson-Kessel.



Slika 7.24. Prikaz pripadnostnih funkcij v 3D prostoru pri rojenju po metodi Gustafson-Kessel.

volumni rojev. Za detekcijo rojev z različnimi volumeni moramo modificirati metodo. Modifikacija je v izračunu faktorja ρ_i (glej 9. korak v algoritmu 7), ki ga definiramo kot

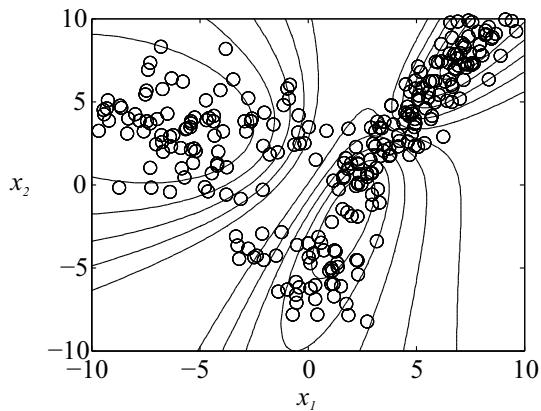
$$\rho_i = \frac{1}{\sum_{k=1}^n \mu_{ik}} = \frac{1}{L_i}. \quad (7.64)$$

Na sliki 7.26 vidimo, da modificirani algoritem Gustafson-Kessel v tem primeru dobro loči roje, ki so neenakih volumnov, saj smo matriko notranjih produktov normirali z inverzno vrednostjo volumna roja.

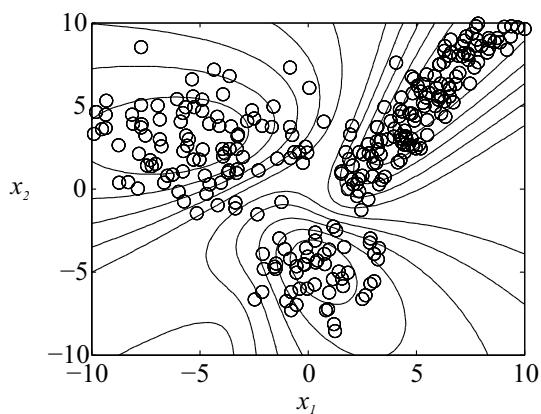
Primer posameznih izjemnih podatkov. Metoda rojenja Gustafson-Kessel dobro deluje v primeru rojev različnih elipsoidnih oblik. Problemi nastanejo, če se v množici točk pojavijo izjemni podatki, ki zelo odstopajo od ostalih podatkov (*ang. outlier*). V takem primeru relativna pripadnost rojem (vsota pripadnosti podatka k vsem rojem je enaka ena) povzroča težave pri segmentaciji prostora. Oblike rojev se deformirajo v smeri oddaljenega podatka, kar je prikazano na sliki 7.27.

7.6.3 Rojenje po metodi največje podobnosti (FMLE)

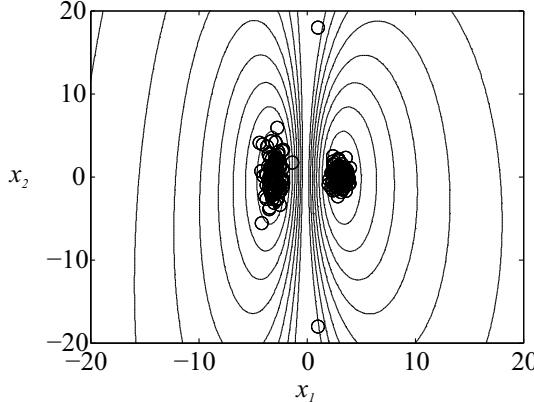
Rojenje po metodi največje podobnosti (*ang. Fuzzy Maximum Likelihood clustering*) temelji na izračunu razdalj s pomočjo funkcije največje podobnosti. Izračun razdalj po tem principu je predstavljen v naslednji enačbi



Slika 7.25. Primer mehkega rojenja po metodi Gustafson-Kessel v primeru različnih volumnov rojev.



Slika 7.26. Primer mehkega rojenja po modificirani metodi Gustafson-Kessel v primeru različnih volumnov rojev.



Slika 7.27. Primer mehkega rojenja po metodi Gustafson-Kessel v primeru posameznih izjemnih podatkov.

$$d_{ik}^2 = \rho_i |\underline{\Sigma}_i|^{\frac{1}{2}} e^{\frac{(\underline{x}(k) - \underline{v}_i)^T \underline{\Sigma}_i^{-1} (\underline{x}(k) - \underline{v}_i)}{2}}, \quad i = 1, \dots, c, \quad k = 1, \dots, n, \quad (7.65)$$

kjer je ρ_i , $i = 1, \dots, c$ inverzna vrednost volumna roja definirana v enačbi 7.64 in je $\underline{\Sigma}_i$, $i = 1, \dots, c$ mehka kovariančna matrika roja in $|\underline{\Sigma}_i|$ njena determinanta. Metoda največje podobnosti temelji na eksponencialni funkciji pri izračunu mere razdalje med podatki, kar pomeni eksponencialno uteževanje Evklidske razdalje med podatki. To pa prinaša segmentiranje prostora, ki je bistveno bolj ostro, kot je to pri Gustafson-Kessel metodi, kjer imamo navadno normo notranjega produkta. Po tej metodi lahko detektiramo roje najrazličnejših oblik (eksponencialna funkcija in mehka kovariančna matrika v funkciji razdalje) in gostot, oziroma volumnov (normiranje z volumnom roja). Metoda je manj numerično robustna od metode Gustafson-Kessla in zato v določenih primerih potrebuje inicializacijo z uporabo enostavnejših metod rojenja (FCM, GK). Pri originalni izpeljavi [9] je zahtevana vrednost parametra $\eta = 1$. Z izbiro η lahko delno vplivamo na ostrost prehodov med posameznimi roji. V tabeli 8 je po korakih predstavljen algoritem mehkega rojenja po metodi največje podobnosti.

Primer rojenja po metodi največje podobnosti. Metoda rojenja po Gustafson-Kesslu rezultira v mehkih prehodih med roji. To lahko dobro vidimo na 3D slikah pripadnosti posameznim rojem. Metoda največje podobnosti prehode, zaradi uteževanja z eksponencialno funkcijo, zelo izostri. Rezultati rojenja s to metodo so prikazani na sliki 7.28 in sliki 7.29 za množico točk, ki so različnih oblik in volumnov. V tem primeru je izbran faktor $\eta = 2$ in število rojev $c = 3$.

7.6.4 Rojenje s c-možnimi roji (PCM)

Vse zgoraj opisane metode imajo težave pri rojenju, kjer so prisotni tudi izjemni podatki (*ang. outlier*) in veliko šum. Takšni podatki onemogočajo izvedbo glavnega namena rojenja, to je iskanje podobnosti v podatkih in razdeljevanje podatkov v različne roje. Problem v takem primeru lahko uspešno rešujemo z uporabo rojenja s c-možnimi roji (*ang. Possibilistic C-Means clustering*).

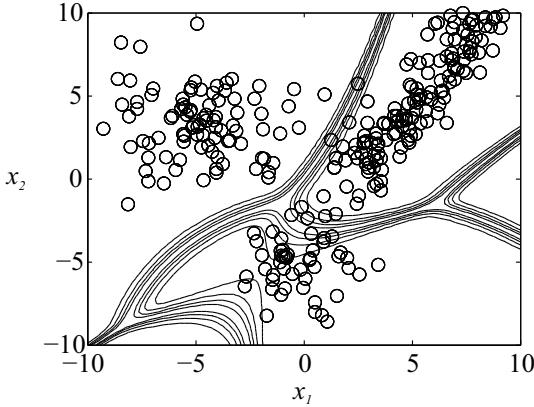
Rojenje z možnimi roji temelji na minimizaciji kriterijske funkcije

$$I_{PCM} = \sum_{i=1}^c \sum_{k=1}^n \mu_{jk}^\eta d_{ik}^2 + \sum_{i=1}^c \nu_i^2 \sum_{k=1}^n (1 - \mu_{ik})^\eta, \quad (7.66)$$

kjer je c število rojev, n število podatkovnih vektorjev, d_{ik}^2 je razdalja med k -tim podatkom (vzorcem) in i -tim središčem roja, μ_{ik} pa je v tem primeru tipičnost vzorca, glede na določen roj, ν_j pa je mehka varianca i -tega roja.

Algoritem 8 Psevdokoda mehkega rojenja po metodi največje podobnosti.

- 1: Inicializacija števila rojev c ,
- 2: Izbira uteži η
- 3: Izbira tolerančnega praga za zaključek iteracije \min_{impro} in izbira maksimalnega števila r_{max}
- 4: Inicializacija matrike pripadnosti z naključnimi vrednostmi $U^0 \in M$
- 5: Izbira začetne vrednosti faktorja $\rho_i^0 = 1$, $i = 1, \dots, c$
- 6: **repeat**
- 7: Izračun središč rojev: $\underline{v}_i^{(r)} = \frac{\sum_{k=1}^n (\mu_{ik}^{(r)})^\eta \underline{x}(k)}{\sum_{k=1}^n (\mu_{ik}^{(r)})^\eta}$, $i = 1, \dots, c$
- 8: Izračun mehke kovariančne matrike: $\underline{\Sigma}_i = \frac{\sum_{k=1}^n (\mu_{ik}^{(r-1)})^\eta (\underline{x}(k) - \underline{v}_i^{(r)}) (\underline{x}(k) - \underline{v}_i^{(r)})^T}{\sum_{k=1}^n (\mu_{ik}^{(r-1)})^\eta}$, $i = 1, \dots, c$
- 9: Izračun eksponenta podobnostne funkcije: $\gamma_{ik} = (\underline{x}(k) - \underline{v}_i^{(r)})^T \underline{\Sigma}_i^{-1} (\underline{x}(k) - \underline{v}_i^{(r)})$
- 10: Izračun razdalj: $d_{ik}^2 = \rho_i |\underline{\Sigma}_i|^{\frac{1}{2}} e^{-\gamma_{ik}}$, $i = 1, \dots, c$, $k = 1, \dots, n$
- 11: Izračun matrike pripadnosti: $\mu_{ik}^{(r)} = \left(d_{ik}^2 \sum_{j=1}^c \left(\frac{1}{d_{jk}^2} \right)^{\frac{1}{\eta-1}} \right)^{-1}$
- 12: **until** $\|\underline{U}^{(r)} - \underline{U}^{(r-1)}\|_\infty < \min_{impro}$ ali $r \geq r_{max}$



Slika 7.28. Primer mehkega rojenja po metodi največje podobnosti (FMLE) v primeru rojev različnih oblik in volumnov.

Z uporabo Picardove iteracije in ob predpostavki Evklidove razdalje za mero podobnosti

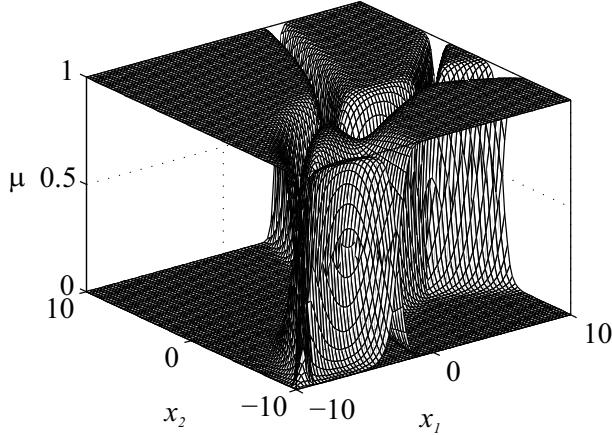
$$d_{ik}^2 = (\underline{x}(k) - \underline{v}_i)^T (\underline{x}(k) - \underline{v}_i), \quad (7.67)$$

kjer je $\underline{x}(k)$ podatkovni vektor dimenzije $m \times 1$ in \underline{v}_i , $i = 1, \dots, c$ je vektor, ki definira središče i -tega roja. Središče ali prototip roja je definiran na naslednji način

$$\underline{v}_j = \frac{\sum_{k=1}^n \mu_{ik}^\eta \underline{x}(k)}{\sum_{k=1}^n \mu_{ik}^\eta}, \quad (7.68)$$

kjer je μ_{ik} tipičnost, ki jo izračunamo kot

$$\mu_{ik} = \left(1 + \left(\frac{d_{ik}^2}{\nu_i^2} \right)^{\frac{1}{\eta-1}} \right)^{-1}, \quad (7.69)$$



Slika 7.29. Primer mehkega rojenja po metodi MLE v primeru rojev različnih oblik in volumnov v 3D.

kjer je ν_{ik}^2 mehka varianca roja in je definirana z

$$\nu_{ik}^2 = \frac{\sum_{k=1}^n \mu_{ik}^\eta d_{ik}^2}{\sum_{k=1}^n \mu_{ik}^\eta}, \quad (7.70)$$

η pa je mera mehkosti. Vse tipične vrednosti vzorcev glede na posamezne roje so shranjene v matriki tipičnosti

$$U = [\mu_{ik}], \quad i = 1, \dots, m, \quad k = 1, \dots, n.$$

Če za izračun razdalje d_{ik} vzamemo Mahalanobisovo razdaljo

$$d_{ik}^2 = (\underline{x}(k) - \underline{v}_i)^T \underline{F}_i (\underline{x}(k) - \underline{v}_i), \quad (7.71)$$

kjer je

$$\underline{F}_i = (\rho_i | \underline{\Sigma}_i |)^{\frac{1}{m}} \underline{\Sigma}_i^{-1}, \quad (7.72)$$

in je $\rho_i = |\underline{F}_i|$ in je $\underline{\Sigma}_i$ mehka kovariančna matrika dimenzije $m \times m$, definirana kot sledi

$$\underline{\Sigma}_i = \frac{\sum_{k=1}^n \mu_{ik}^\eta (\underline{x}(k) - \underline{v}_i)(\underline{x}(k) - \underline{v}_i)^T}{\sum_{k=1}^n \mu_{ik}^\eta}, \quad (7.73)$$

potem dobimo, tako imenovano rojenje po metodi Gustafson-Kessel s c-možnimi roji (GKPCM). Tipičnost se v primeru Mahalanobisove razdalje izračuna kot

$$\mu_{ik} = \left(1 + \left(\frac{(\underline{x}(k) - \underline{v}_i)^T \underline{F}_i (\underline{x}(k) - \underline{v}_i)}{\delta} \right)^{\frac{1}{\eta-1}} \right)^{-1}, \quad (7.74)$$

kjer je δ nastavitevni parameter, ki ga nastavimo glede na pričakovano obliko rojev. Če pričakujemo roje, ki so vzdolžni, oziroma z velikim razmerjem med največjo in najmanjšo lastno vrednostjo mehke kovariančne matirke, potem moramo nastaviti majhen faktor δ .

Predlagana metod c-možnih rojev (PCM) in metoda Gustafson-Kessel s c-možnimi roji rešuje probleme rojenja pošumljenih podatkov in probleme, kjer se pojavljajo izjemni podatki. Hkrati lahko z GKPCM odkrivamo roje različnih oblik. Slaba lastnost obeh metod je velika občutljivost na inicializacijo začetnih središč rojev, oziroma začetnih tipičnosti. Problem rešimo z uporabo metode FCM za inicializacijo rojev. Ker so tipičnosti med seboj neodvisne, metoda pogosto rezultira v sovpadajočih središčih rojev. To lastnost lahko izkoristimo za iskanje optimalnega števila rojev, saj odvečna središča sovpadejo s pravimi ali pa imajo degenerirane roje.

Psevdokoda za metodo rojenja GKPCM je prikazana v algoritmu 9.

Algoritem 9 Psevdokoda za metodo rojenja GKPCM.

1: Definiranje maksimalnega možnega števila rojev c_{max} , mehkega parametra η , kriterija za zaključek algoritma ϵ , parametra oblike rojev δ , parametra sprejemljive velikosti sledi kovariančne matrike roja t_{max} , parametra sprejemljivega razmerja med največjo in najmanjšo lastno vrednostjo kovariančne matrike roja γ_{max} in minimalno sprejemljivo število elementov roja n_{min} .

2: Inicializacija tipičnosti za c_{max} rojev z uporabo FCM.

3: **repeat**

4: Izračun središč rojev

$$\underline{v}_i = \frac{\sum_{k=1}^n \mu_{ik}^\eta \underline{x}(k)}{\sum_{k=1}^n \mu_{ik}^\eta}, \quad i = 1, \dots, c$$

5: Izračun kovariančne matrike in notranje norme distance norm

$$\underline{\Sigma}_i = \frac{\sum_{k=1}^n \mu_{ik}^\eta (\underline{x}_k - \underline{v}_i) (\underline{x}_k - \underline{v}_i)^T}{\sum_{k=1}^n \mu_{ik}^\eta}, \quad i = 1, \dots, c$$

$$\begin{aligned} \underline{F}_i &= (\rho_i | \underline{\Sigma}_i |)^{\frac{1}{m}} \underline{\Sigma}_i^{-1}, \quad i = 1, \dots, c \\ \rho_i &= |\underline{F}_i|, \quad i = 1, \dots, c \end{aligned}$$

6: Izračun razdalj

$$d_{ik}^2 = (\underline{x}(k) - \underline{v}_i)^T \underline{F}_i (\underline{x}(k) - \underline{v}_i), \quad i = 1, \dots, c_{max}, \quad k = 1, \dots, n.$$

7: Izračun tipičnosti

$$\begin{aligned} \mu_{ik} &= \left(1 + \left(\frac{d_{ik}^2}{\delta} \right)^{\frac{1}{\eta-1}} \right)^{-1} \\ i &= 1, \dots, c_{max}, \quad k = 1, \dots, n. \end{aligned}$$

8: **until** $\|\Delta U\|_\infty < \epsilon$

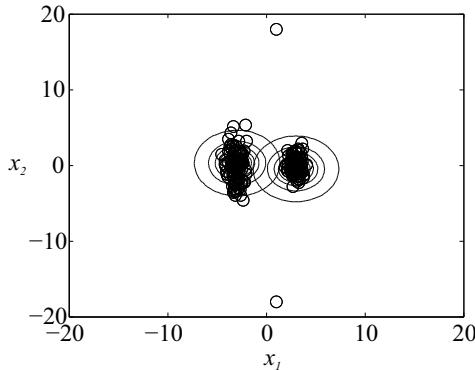
9: Eliminacija rojev, ki ne ustrezajo sprejemljivim kriterijem

$$\begin{aligned} \text{trace}(\underline{F}_i) &> t_{max} \text{ or } \frac{\lambda_{i_{max}}}{\lambda_{i_{min}}} > \gamma_{max} \text{ or } n_j < n_{min}, \\ i &= 1, \dots, c. \end{aligned}$$

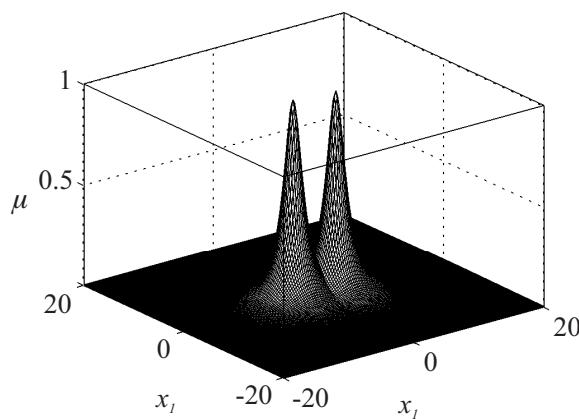
Primer uporabe metode PCM. Metodo rojenja z možnimi roji (PCM) smo uporabili v primeru, ko se med množico pojavijo tudi izjemni podatki (*ang. outlier*). Oblike rojev so zaradi uporabljeni Evklidove razdalje radialne. Problem izjemnih podatkov se z uvedbo tipičnosti, oziroma absolutne pripadnosti, dobro reši kar prikazuje slika 7.30. Na sliki 7.31 sta prikazani pripadnostni funkciji v 3D prostoru.

Reševanje problema neznanega števila rojev z uporabo GKPCM. V nadaljevanju bo prikazan način reševanja problema števila rojev. Predpostavimo najprej, da vedno lahko določimo maksimalno število rojev c_{max} .

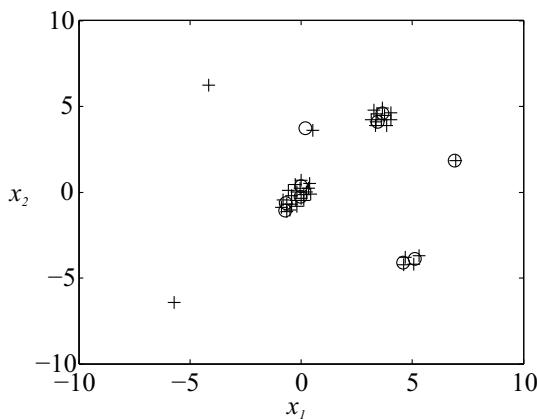
Pri uporabi metode GKPCM so izračuni tipičnosti absolutni. Tipičnosti posameznih rojev so na ta način med seboj neodvisne. V primeru, ko je $c_{max} - c$ središča rojev redundantnih, ti redundantni roji konvergirajo v enaka središča ali pa imajo degenerirane roje. Degenerirane roje lahko detektiramo glede na njihove volumne in oblike. Nesprejemljive roje odstranimo. Primer uporabe GKPCM je na sliki 7.32, kjer je prikazana inicializacija rojev z metodo FCM. Na sliki 7.33 so prikazani podatki in središča rojev dobljeni s GKPCM, na sliki 7.34 pa končni rezultat rojenja. Roji so obkroženi s konturo, ki definira področje z verjetnostjo 0.98. Pred tem so bili odstranjeni degenerirani roji in tisti, ki imajo premajhno število elementov.



Slika 7.30. Mehko rojenje po metodi PCM v primeru posameznih oddaljenih podatkov.



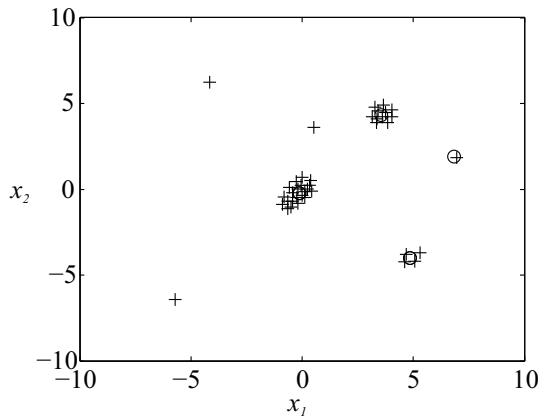
Slika 7.31. Pripadnostni funkciji v 3D prostoru za mehko rojenje po metodi PCM v primeru posameznih oddaljenih podatkov.



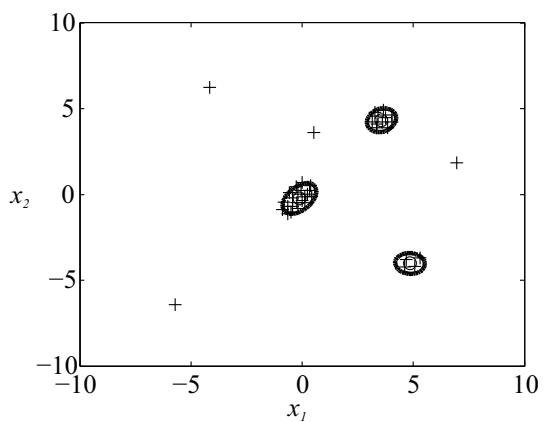
Slika 7.32. Inicializacija središč z uporabo FCM.

7.7 Rojenje z linearimi prototipi - posplošena metoda rojenja

V prejšnjem poglavju so bili prototipi rojev določeni s točkami prostora. V nadaljevanju pa si bomo ogledali metode mehkega rojenja z linearimi prototipi. To pomeni, da bodo središča rojev definirana s hiperravnino v prostoru problema. Dimenzijo prototipa definiramo s spremenljivko



Slika 7.33. Centri rojev dobljeni z metodo GKFCM.



Slika 7.34. Centri rojev s pripadajočim področjem 0.02 zaupanja dobljeni s GKPCM.

s. Velja, da je $0 \leq s \leq m - 1$, kjer je m dimenzija prostora podatkov. Dimenzija prototipa je odvisna tudi o izbire regresorja.

7.7.1 Rojenje po metodi glavnih komponent

Osnovna ideja rojenja po metodi glavnih komponent (*ang. fuzzy c-varieties*) je v definiciji razdalje med opazovanim podatkom, ki ga predstavlja določena točka v prostoru problema in prototipom roja, ki ga predstavimo v splošnem z s -dimenzionalnim linearnim mnogoterjem, ki opiše večino informacije, ki se skriva znotraj mehke kovariančne matrike roja. V primeru točkovnih središč rojev je dimenzija mnogoterja enaka $s = 0$, v primeru dimenzijs $s = 1$ so taki prototipi premice, ravnine dobimo v primeru $s = 2$ in hiperravnine v primeru $s > 2$.

Za razlago osnovne ideje si oglejmo primer v dvodimenzionalnem prostoru podatkov. Predpostavimo enodimenzionalni prototip roja ($s = 1$). Enodimenzionalni prototip je premica, ki jo lahko zapišemo s točko V_i na premici in smerjo enotinega vektorja \underline{p}_i . Krajevni vektor do točke V_i zapišemo z vektorjem \underline{v}_i . Točka podatka za katerega bomo računali razdaljo do prototipa je X_k in njen krajevni vektor \underline{x}_k .

Pravokotna razdalja med prototipom in točko X_k je enaka d_{ik} . V pravokotnem trikotniku $\triangle V_i X_k T$ velja enakost

$$d_{ik}^2 = \|\underline{x}_k - \underline{v}_i\|^2 - \|\underline{e}_i\|^2, \quad (7.75)$$

kjer je $\|\underline{e}_i\|$ dolžina vektorja med točko projekcije T in točko V_i in $\|\underline{x}_k - \underline{v}_i\|$ dolžina vektorja med opazovano točko X_k in točko V_i . Zapisa predstavljata kvadratno vektorsko normo $\|\underline{x}\| = \sqrt{\sum_{i=1}^m |x_i|^2} = \sqrt{\underline{x}\underline{x}^T}$.

Skalarni produkt vektorjev $\underline{x}_k - \underline{v}_i$ in \underline{p}_i lahko zapišemo kot

$$(\underline{x}_k - \underline{v}_i)^T \cdot \underline{p}_i = \|\underline{p}_i\| \cdot \|\underline{x}_k - \underline{v}_i\| \cos \varphi = \|\underline{e}_i\|. \quad (7.76)$$

Kvadrat pravokotne razdalje med opazovano točko in prototipom, v zapisu z vektorskimi normami, je enak

$$d_{ik}^2 = \|\underline{x}_k - \underline{v}_i\|^2 - \langle \underline{x}_k - \underline{v}_i, \underline{p}_i \rangle^2, \quad (7.77)$$

kjer je z $\langle \cdot \rangle$ zapisan skalarni produkt vektorjev. V navadnem vektorskem zapisom, pa sledi

$$d_{ik}^2 = (\underline{x}_k - \underline{v}_i)^T (\underline{x}_k - \underline{v}_i) - \langle \underline{x}_k - \underline{v}_i, \underline{p}_i \rangle^2. \quad (7.78)$$

Posplošimo sedaj izračun razdalje do poljubnega prototipa v m razsežnem prostoru problema. Prototip reda s določa s ortonormalnih vektorjev in točka, ki je določena s krajevnim vektorjem \underline{v}_i . Obravnavana točka prostora je določena s krajevnim vektorjem \underline{x}_k . Dolžino vektorjev, ki predstavljajo pravokotno projekcijo vektorja $\underline{x}_k - \underline{v}_i$ na posamezne bazne vektorje lahko zapišemo kot skalarni produkt $\|e_j\| = \langle \underline{x}_k - \underline{v}_i, \underline{p}_{ij} \rangle$. Dolžina vektorja med točko V_i in X_k je diagonalna pravokotnega mnogoterja, ki ima stranice dolžin $\|e_j\|, j = 1, \dots, m$. Zato lahko zapišemo relacijo

$$\langle \underline{x}_k - \underline{v}_i, \underline{x}_k - \underline{v}_i \rangle = \sum_{j=1}^m \langle \underline{x}_k - \underline{v}_i, \underline{p}_{ij} \rangle^2. \quad (7.79)$$

Če sedaj predpostavimo prototip reda s , potem lahko desni del enačbe 7.79 razdelimo na kvadrat razdalje d_{ik}^2 in del, ki leži na prototipu $\sum_{j=1}^s \langle \underline{x}_k - \underline{v}_i, \underline{p}_{ij} \rangle^2$. Razdaljo točke do prototipa sedaj zapišemo kot

$$d_{ik}^2 = (\underline{x}_k - \underline{v}_i)^T (\underline{x}_k - \underline{v}_i) - \sum_{j=1}^s \langle \underline{x}_k - \underline{v}_i, \underline{p}_{ij} \rangle^2. \quad (7.80)$$

Razdaljo za tridimenzionalni prostor podatkov, kjer prototip določata vektorja \underline{p}_1 in \underline{p}_2 in točka na prototipu z vektorjem \underline{v}_i , lahko izrazimo z enačbo

$$d_{ik}^2 = (\underline{x}_k - \underline{v}_i) (\underline{x}_k - \underline{v}_i)^T - \sum_{j=1}^2 \langle \underline{x}_k - \underline{v}_i, \underline{p}_{ij} \rangle^2. \quad (7.81)$$

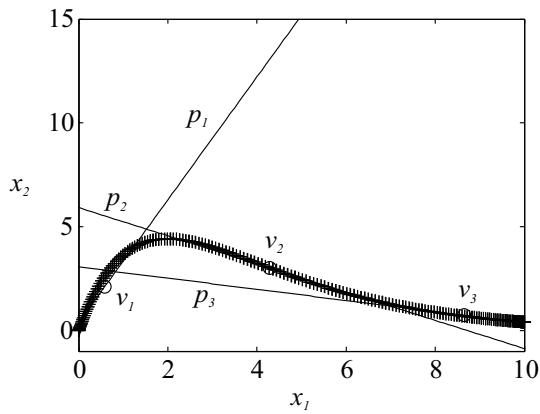
V tabeli 10 je po korakih predstavljen algoritem mehkega rojenja po metodi glavnih komponent.

Algoritem 10 Psevdokoda mehkega rojenja po metodi glavnih komponent.

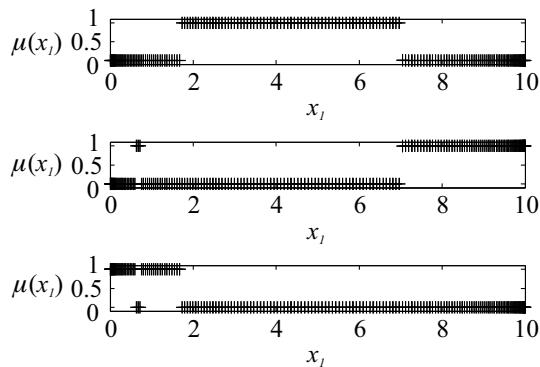
- 1: Inicializacija števila rojev c ,
 - 2: Izbira uteži η
 - 3: Izbira tolerančnega praga za zaključek iteracije min_{impro} in izbira maksimalnega števila r_{max}
 - 4: Inicializacija matrike pripadnosti z naključnimi vrednostmi $U^0 \in M$
 - 5: Izbira začetne vrednosti faktorja $\rho_i^0 = 1, i = 1, \dots, c$
 - 6: **repeat**
 - 7: Izračun središč rojev: $\underline{\underline{v}}_i^{(r)} = \frac{\sum_{k=1}^n (\mu_{ik}^{(r)})^\eta \underline{x}(k)}{\sum_{k=1}^n (\mu_{ik}^{(r)})^\eta}, i = 1, \dots, c$
 - 8: Izračun mehke kovariančne matrike: $\underline{\underline{\Lambda}}_i = \frac{\sum_{k=1}^n (\mu_{ik}^{(r-1)})^\eta (\underline{x}(k) - \underline{v}_i)^T (\underline{x}(k) - \underline{v}_i)}{\sum_{k=1}^n (\mu_{ik}^{(r-1)})^\eta}, i = 1, \dots, c$
 - 9: Izračun razcepa singularnih vrednosti matrike: $\underline{\underline{\Lambda}}_i^{(r)} = \underline{\underline{P}}_i \underline{\Lambda}_i \underline{\underline{P}}_i^T, \underline{p}_{ij}, j = 1, \dots, s, i = 1, \dots, c$
 - 10: Izračun razdalj: $d_{ik}^2 = \|\underline{x}_k - \underline{v}_i\|^2 - \sum_{j=1}^s \langle \underline{x}_k - \underline{v}_i^{(r)}, \underline{p}_{ij} \rangle^2, i = 1, \dots, c, k = 1, \dots, n$
 - 11: Izračun matrike pripadnosti: $\mu_{ik}^{(r)} = \left(d_{ik}^2 \sum_{j=1}^c \left(\frac{1}{d_{jk}^2} \right)^{\frac{1}{\eta-1}} \right)^{-1}$
 - 12: **until** $\|\underline{\underline{U}}^{(r)} - \underline{\underline{U}}^{(r-1)}\|_\infty < min_{impro}$ ali $r >= r_{max}$
-

- **Opažanje 1.** Metoda je občutljiva na inicializacijo matrike pripadnosti U^0 . Zato je navadno potrebno za inicializacijo zagnati enega od robustnih algoritmov rojenja (FCM).
- **Opažanje 2.** Algoritem ne omejuje velikosti linearnega prototipa in zato teži k združevanju kolinearnih prototipov, čeprav so roji med seboj dobro ločeni.
- **Opažanje 3.** Algoritem konvergira k napačnim particijam, če so roji različnih volumnov.

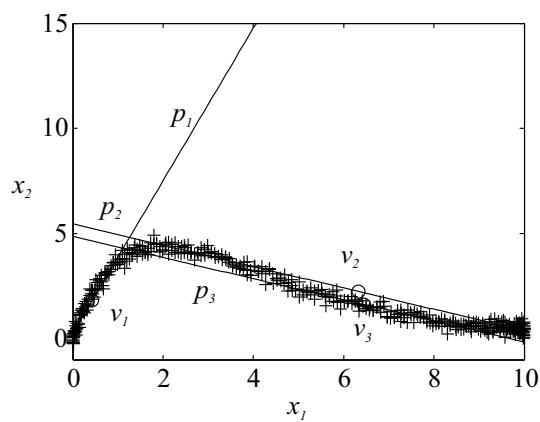
Primer rojenja po metodi glavnih komponent Za ta primer smo v dvorazsežnem prostoru z linearimi prototipi modelirali nelinearno funkcijo. Rezultati so prikazani na sliki 7.35, kjer vidimo tri linearne prototipe in točke skozi katere gredo. Linearni prototip je v tem primeru definiran s točko $\underline{v}_i, i = 1, \dots, 3$ in vektorjem $\underline{p}_i, i = 1, \dots, 3$. V tem primeru je izbran faktor $\eta = 2$ in $c = 3$. Na sliki 7.36 pa vidimo vrednosti pripadnosti posameznim rojem, ki so ne glede na izbiro parametra η , zelo ostre. V primeru, ko se pri meritvah pojavi šum, metoda pri izbranem faktorju nelinearnosti $\eta = 2$, ni robustna in ne da vedno zadovoljivih rezultatov, kar prikazuje slika 7.37 (čeprav jo inicializiramo z Evklidovim algoritmom). V primeru visokega šuma lahko z ustreznou izbiro parametra mehkosti $\eta = 1.05$ pridemo do ustreznih rešitev. Primer prikazuje slika 7.38, kjer je število segmentov $c = 5$. Metoda rojenja po metodi glavnih komponent je odlično orodje za identifikacijo odsekoma linearnih (*ang. piece-wise Linear systems - PWL*) statičnih in tudi dinamičnih sistemov.



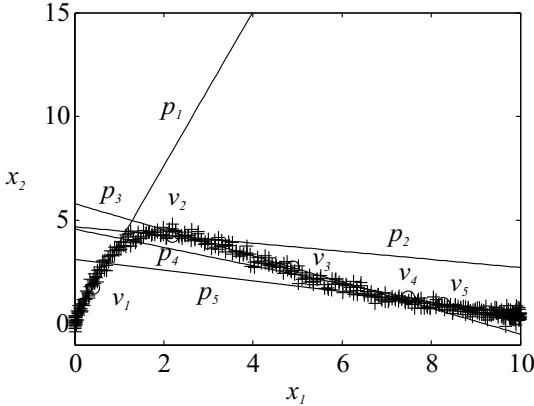
Slika 7.35. Primer mehkega rojenja z linearimi prototipi.



Slika 7.36. Vrednosti pripadnosti posameznim rojem za primer mehkega rojenja z linearimi prototipi.



Slika 7.37. Mehko rojenje z linearimi prototipi v primeru šuma ($\eta = 2$).

Slika 7.38. Mehko rojenje z linearnimi prototipi v primeru šuma ($\eta = 1.05$).

7.8 Kohonenova samoorganizirajoča mreža (SOM)

Kohonenova samoorganizirajoča mreža (SOM) [65] je eden bolj popularnih pristopov k rojenju z uporabo nevronskega mrež. Je razširitev metode vektorske kvantizacije (*ang. vector quantization - VQ*), ki jo je prav tako razvil Kohonen. Vektorska kvantizacija je poenostavljena verzija metode k-povprečij v smislu adaptacije. Ta ne popravlja parametrov po prehodu celotne množice podatkov (*ang. batch adaptation*), ampak sprotro z vsakim novim vzorcem (*ang. on-line adaptation from data stream*). Vektorska kvantizacija dela s c kvazi-nevroni, ki ustrezajo klasičnim rojem. Vsak nevron ima p (dimenzija problemske domene enaka p) parametrov ali uteži. Parametre za vsak nevron bomo zapisali kot \underline{v} . Učenje nevronov se izvede na osnovi naslednje enačbe

Algoritem 11 Psevdokoda rojenja vektorsko kvantizacijo.

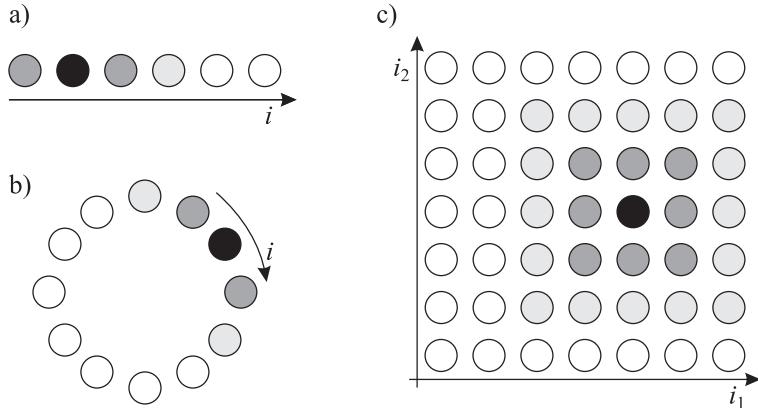
- 1: Inicializacija vektorjev parametrov nevronov \underline{v}_i , $i = 1, \dots, c$. To lahko naredimo z naključno izbiro c začetnih točk iz množice podatkov.
- 2: **repeat**
- 3: Naključno izberemo vzorec \underline{x} iz množice podatkov, ali pa gremo sistematično skozi celotno množico (ciklično).
- 4: Izračun razdalj (tipično Evklidova) med izbranim vzorcem in vsemi nevroni. Nevron, ki je najbližje izbranemu vzorcu je zmagovalni nevron (*winner neuron*).
- 5: Adaptacija vektorja zmagovitega nevrona tako, da se pomakne proti vzorcu \underline{x} :

$$\underline{v}_{\text{win}}^{(\text{new})} = \underline{v}_{\text{win}}^{(\text{old})} + \eta (\underline{x} - \underline{v}_{\text{win}}^{(\text{old})}). \quad (7.82)$$

- 6: **until** $\|\underline{v}_{\text{win}}^{(\text{new})} - \underline{v}_{\text{win}}^{(\text{old})}\|_\infty < \text{min}_{\text{impro}}$
-

$$\underline{v}_j^{(\text{new})} = \underline{v}_j^{(\text{old})} + \eta (\underline{x} - \underline{v}_j^{(\text{old})}), \quad (7.83)$$

kjer je \underline{x} vzorec podatkov, $\underline{v}_j^{(\text{old})}$ je stara vrednost parametrov in $\underline{v}_j^{(\text{new})}$ nova vrednost vektorja parametrov za nevron. Konstanta učenja (*ang. learning rate*) η mora biti pazljivo izbrana. Vrednost $\eta = 1$ bi rezultirala v trenutnem podatku in algoritem ne bi konvergiral. Za vse algoritme, ki imajo sprotro adaptacijo velja, da lahko konvergirajo samo, če konstanta učenja η konvergira k 0. Zato je za hitro konvergenco priporočljivo, da začnemo z relativno veliko konstanto, npr. 0.5,



Slika 7.39. Različne topologije za SOM: a) linearna, b) krožna, c) dvodimensionalna mreža. Črni nevron je zmagovalec za določen vhodni podatek, sivi pa so njegovi sosedje.

ki jo potem z vsako iteracijo zmanjšujemo. Učenje z vektorsko kvantizacijo je pravzaprav enako kot sprotno učenje pri metodi k-povprečij [1, 104], kjer je korak normiran s številom vzorcev, ki pripadajo nevronu.

Metode rojenja, ki temeljijo na nevronskih mrežah imenujemo učenje s tekmovanjem (*ang. competitive learning*), ker vsi nevroni tekmujejo za izbrani podatek. Strategija pri vektorski kvantizaciji pa se imenuje strategija zmagovalca (*ang. the winner takes it all*), ker adaptiramo samo zmagovalni nevron.

Kohonenova samoorganizirajoča mreža je (*ang. self-organizing maps*) razširitev algoritma vektorske kvantizacije, ki smo ga predstavili [33, 65, 102, 104]. V primeru mreže SOM nevroni niso samo abstraktne strukture, ampak so organizirani v eno, dvo ali večdimenzionalne topologije; glej sliko 7.39. V največ aplikacijah se uporablja dvodimensionalna topologija s heksogonalno ali pravokotno strukturo (kot na sliki 7.39c). Vektorji uteži nevronov (središča rojev) so p dimenzionalni kot pri vektorski kvantizaciji, in neodvisni od topologije.

Ideja samoorganizirajoče mreže (SOM) je v tem, da morajo imeti tudi sosedni nevroni v topologiji mreže podobne uteži (centre rojev). To pomeni, da je razlika med središči rojev v p -dimenzionalnem prostoru sedaj predstavljena v nižji (tipično drugi) dimenziji. Ta projekcija, iz multidimenzionalnega prostora v dvodimensionalni prostor, je lahko narejena samo s stiskanjem podatkov. Dvodimensijska mreža je odlično orodje za vizualizacijo multidimensijske množice podatkov z različnimi distribucijami.

Samoorganizirajoča mreža definira sosednjo funkcijo (*ang. neighborhood function*). Preko te funkcije se ne spremenijo se samo uteži nevrona, ki zmaga, ampak tudi njegovi sosedji. Sosednja funkcija $h(\underline{i})$ ima vrednost 1 za zmagovalni nevron, ta vrednost pa pada z oddaljenostjo od zmagovalca. Sosednja funkcija je definirana glede na topologijo mreže SOM. Na primer, mreži SOM na sliki 7.39a in b imata enodimensionalno sosednjo funkcijo, to je

$$h(i) = \exp\left(-\frac{1}{2} \frac{(i^{(\text{win})} - i)^2}{\sigma^2}\right), \quad (7.84)$$

kjer je $i^{(\text{win})}$ indeks zmagovitega nevrona in i indeks kateregakoli nevrona iz topologije. SOM na sliki 7.39c ima dvodimensionalno sosednjo funkcijo, t.j.,

$$h(i_1, i_2) = \exp\left(-\frac{1}{2} \frac{(i_1^{(\text{win})} - i_1)^2 + (i_2^{(\text{win})} - i_2)^2}{\sigma^2}\right), \quad (7.85)$$

kjer $i_1^{(\text{win})}$ in $i_2^{(\text{win})}$ definirata indeksa zmagovitega nevrona, i_1 in i_2 pa indeksa kateregakoli nevrona iz topologije. Črni, sivi in rahlo sivi nevroni na sliki 7.39 prikazujejo aktivacijsko vrednost sosednje funkcije. Učenje nevronov se sedaj razširi v

$$\underline{v}_j^{(\text{new})} = \underline{v}_j^{(\text{old})} + \eta h(\underline{i}) (\underline{x} - \underline{v}_j^{(\text{old})}) , \quad (7.86)$$

kjer je $h(\underline{i})$ sosednja funkcija glede na topologijo mreže SOM. Opozoriti velja, da se učenje po enačbi 7.86 izvede za vse aktivne nevrone $j = 1, \dots, c$ in ne samo za zmagovitega. Z učenjem po enačbi 7.86 se celotna skupina sosednjih nevronov pomakne proti vhodnemu podatku. Učenje je večje, če je nevron bližje zmagovitemu (večja vrednost $h(\underline{i})$). Mreža SOM običajno začne z široko sosednjo funkcijo in jo zožuje z iteracijami. Na ta način na začetku dobi informacijo o grobi distribuciji podatkov (globalno učenje) in jo potem prilagodi (adaptira), ko postaja sosednja funkcija vse bolj ostra (lokalno učenje). To je podobno kot v primeru koraka η . Če zoževanje poteka počasi, se zmanjša nevarnost konvergencije v lokalni minimum.

7.9 Povzetek

Nenadzorovane tehnike učenja omogočajo iskanje informacije o porazdeljenosti podatkov v določeni množici, oziroma omogočajo stiskanje informacije. Običajno jih uporabljamo za predprocesiranje podatkov pred uporabo nadzorovanih tehnik učenja, ki omogočajo iskanje preslikav med vhodi in izhodi. S procesiranjem vhodno-izhodnega prostora pa lahko za namene nadzorovanega učenja uporabimo tudi nenadzorovano učenje.

Nenadzorovane tehnike učenja uvrstimo v dve kategoriji:

- Metoda glavnih komponent (PCA), ki jo lahko uporabimo za transformacijo osi, zmanjšanje dimenzije in odpravljanje redundantnih podatkov, ki ne prinašajo nove informacije (kolinearnih podatkov). V primeru zelo velikih dimenzij podatkov je metoda PCA najboljša metoda za njihovo zmanjšanje. Odpravljanje kolinearnosti v podatkih rešuje tudi probleme singularnosti pri njihovi nadaljnji obdelavi.
- Metode rojenja iščejo skupine podatkov s podobnimi lastnostmi. Z izbiro mere podobnosti posredno določimo oblike rojev, ki jih algoritem zazna. Najbolj popularna tehnika je rojenje s k-povprečji, c-povprečji in rojenje po metodi Gustafson-Kessel. Tudi nevronske mreže se uporabljajo za namene rojenja. Najbolj razširjena je samoorganizirajoča mreža (SOM). Močno orodje rojenja je rojenje na vhodno-izhodnem prostoru, ki omogoča direktno iskanje funkcij-ske povezave med vhodnimi vektorji in izhodi ob hkratnem razdeljevanju prostora glede na distribucijo podatkov. Ta način je izredno pomemben in še vedno predmet raziskav.

8. Optimizacija kompleksnosti modela

V tem poglavju se bomo posvetili vprašanju kompleksnosti modela. Osvetlili bomo problem podparametrizacije in nadparametrizacije modelov, ki so povezani s premalo ali prekompleksnimi modeli. Na del teh vprašanj lahko odgovorimo zelo splošno, ne glede na tipe modelov, ki jih obravnavamo. Zaradi tega so obravnavana vprašanja pomembna, na glede na to ali identificiramo linearne ali nelinearne modele, s klasičnimi ali modernimi tehnikami.

Po kratkem uvodu v smislu kompleksnosti modelov se bomo lotili dileme pristranskosti in variance (*ang. bias, variance*), ki bo natančno razložena. Razloži nam vpliv kompleksnosti na kvaliteto modela. V nadaljevanju bo podan vpogled v pomen različnih množic podatkov za identifikacijo in vrednotenje modela (*ang. validation*). Na koncu je predstavljeno tudi nekaj pristopov, ki omogočajo zmanjšanje kompleksnosti modela glede na strukturne modifikacije.

8.1 Uvod

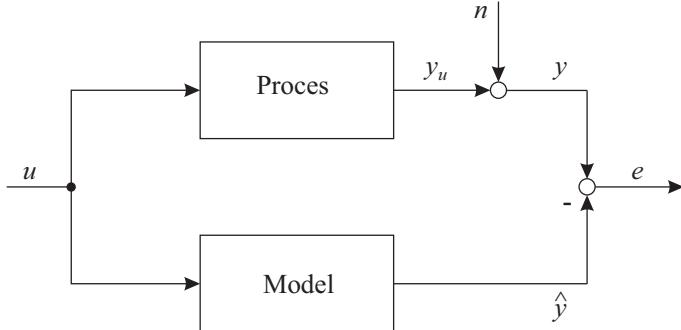
Izraz kompleksnost modela je relativno težko natančno definirati. Kompleksnost bi lahko opredelili glede na zapis modela, njegovo strukturo, proceduro, ki nas pripelje do končnega rezultata ali pa računalniški čas, ki je potreben za njegovo identifikacijo. V našem primeru se bomo osredotočili na kompleksnost v smislu števila parametrov modela. Gre za parametre, ki jih moramo oceniti, če želimo identificirati model. Na ta način rečemo, da postane model bolj kompleksen, če dodamo določene parametre in manj, če število parametrov zmanjšamo. Kompleksnost modela (*ang. complexity*) na drugi strani izraža tudi njegovo prilagodljivost (*ang. flexibility*), ki se uporablja kot sinonim.

Osnovna ideja predstavljena v tem poglavju je v tem, da model ne sme biti preenostaven, če želimo natančno opisati obnašanje procesa. Na drugi strani pa tudi ne sme biti preveč kompleksen, ker bomo potem morali oceniti večje število parametrov. Za oceno večjega števila parametrov pa potrebujemo tudi večje število podatkov. Obstaja določen kompromis, ki mu rečemo optimalna kompleksnost modela (*ang. optimal model complexity*).

8.2 Kompromis med pristranskostjo in varianco

V tem delu si bomo ogledali vpliv števila parametrov na kvaliteto modela. Pokazali bomo, da napako modela lahko razstavimo na dva dela: napako pristranskosti (*ang. bias error*) in variance (*ang. variance error*). Ta dekompozicija nam pomaga razumeti vpliv števila parametrov na kvaliteto modela. Najprej si bomo ogledali matematično razlago, v nadaljevanju pa bomo na primeru bolj natančno raložili problem.

Predpostavimo shemo na sliki 8.1, ki predstavlja proces z izhodom y_u , ki je moten s šumom n . Merjeni izhod procesa je označen z y . Model z izhodom \hat{y} predstavlja proces. Model smo dobili na osnovi optimizacije (minimizacije) kriterijske funkcije, ki je funkcija napake e , optimirali pa smo parametre modela.



Slika 8.1. Proses in model. Napako med izhodom procesa in modela e lahko razstavimo na del zaradi šuma in del zaradi pristranskosti in variance.

Če delamo z naključnimi spremenljivkami, potem lahko za kriterijsko funkcijo predlagamo pričakovano vrednost (*ang. expectation* - E) kvadrata napake med izhodom procesa in modela. To je analogno vsoti kvadratov napake, ki jo uporabljamo pri determinističnih signalih. Kriterijsko funkcijo najprej razdelimo na dva dela

$$E\{e^2\} = E\{(y - \hat{y})^2\} = E\{(y_u + n - \hat{y})^2\} = E\{(y_u - \hat{y})^2\} + E\{n^2\}, \quad (8.1)$$

kjer križni člen $E\{(y_u - \hat{y})n\}$ izgine, ker šum ni koreliran s koristnim izhodom procesa in niti ne z izhodom modela. Prvi člen na desni strani enačbe 8.1 predstavlja napako med pravim izhodom (nemerljivim) procesa in izhodom modela, drugi člen pa predstavlja varianco šuma. Kriterijska funkcija je minimalna, če model opisuje proces idealno, kadar je $\hat{y} = y_u$. V tem primeru prvi člen izgine in je minimalna kriterijska funkcija enaka varianci šuma. Ker na varianco šuma z modelom ne moremo vplivati, obravnavamo v nadaljevanju samo prvi del.

Napako modela $y_u - \hat{y}$ lahko naprej razčlenimo na naslednji način

$$\begin{aligned} E\{(y_u - \hat{y})^2\} &= E\{(y_u - E\{\hat{y}\} - (\hat{y} - E\{\hat{y}\}))^2\} \\ &= E\{(y_u - E\{\hat{y}\})^2\} - 2E\{y_u - E\{\hat{y}\}\}E\{\hat{y} - E\{\hat{y}\}\} + E\{(\hat{y} - E\{\hat{y}\})^2\}, \end{aligned}$$

ker je pričakovana vrednost v drugem členu enačbe 8.2 enaka $E\{\hat{y} - E\{\hat{y}\}\} = 0$, vsi navzkrižni členi izginejo in dobimo

$$E\{(y_u - \hat{y})^2\} = (y_u - E\{\hat{y}\})^2 + E\{(\hat{y} - E\{\hat{y}\})^2\}. \quad (8.2)$$

Na koncu lahko zapišemo

$$(\text{napaka modela})^2 = (\text{napaka pristranskosti})^2 + \text{varianca izhoda}. \quad (8.3)$$

Število parametrov modela odločilno vpliva na zgornje vrednosti. Kot bomo videli kasneje, sta pristranskost in varianca izhoda v nasprotju. Za optimalno izbiro kompleksnosti moramo izbrati kompromis med pristranskostjo in varianco napake. Več lahko najdete v literaturi [10, 34, 118].

Za boljše razumevanje v nadaljevanju, moramo razumeti razliko med dvema množicama podatkov: *učno množico* in *validacijsko množico*. Učna množica je uporabljena za učenje modela, to je za oceno (*ang. estimation*) ali optimizacijo parametrov. Validacijska množica pa je uporabljena za oceno kvalitete modela. Običajno za to uporabimo dve različni množici podatkov. Pomembno je, da model daje dobre rezultate na novi množici podatkov, na validacijski množici.

8.2.1 Napaka pristranskosti

Napaka pristranskosti je del napake modela, ki je posledica omejene opisne možnosti modela (fleksibilnosti). V realnosti so procesi lahko zelo kompleksi in modeli ne morejo opisati procesa idealno. Tudi, če bi parametre modela nastavili na optimalne vrednosti, bi vseeno imeli napako zaradi nemodelirane dinamike. Ta napaka se imenuje napaka pristranskosti in je prisotna zaradi strukturnih omejitev modela.

Kot primer vzemimo linearni proces petega reda, ki ga modeliramo z linearnim procesom tretjega reda. Čeprav parametre modela nastavimo na njihove optimalne vrednosti (glede na kriterijsko funkcijo), model vseeno ne bo popolnoma natančno opisal procesa. Napaka se v tem primeru imenuje napaka pristranskosti. Če bi proces v tem primeru modelirali z modelom petega reda, bi bila pristranskost enaka nič.

Če obravnavamo nelinearni proces, potem bomo v vsakem primeru dobili napako pristranskosti, ker noben linearni model ne more opisati poljubnega nelinearnega procesa s popolno natančnostjo.

Napako pristranskosti lahko zmanjšamo, če za model izberemo ustrezeno strukturo. V primeru nelinearnih procesov napake pristranskosti ne moremo izničiti, ker so vse strukture s katerimi modeliramo nelinearne sisteme samo univerzalni aproksimatorji, ker prave strukture navadno ne poznamo. V takih primerih lahko vedno pričakujemo napako pristranskosti. Največ kar lahko pričakujemo je to, da pristranskost s povečevanjem kompleksnosti postaja manjša. Če zadovolstvo temu kriteriju, potem govorimo, da je aproksimator za določen razred procesov univerzalni aproksimator (*ang. universal approximator*).

Pristranskost je sistematična napaka modela, ki je posledica neustrezne strukture modela. Napaka model je vedno večja od pristranskosti

$$\text{pristranskost} = y_u - E\{\hat{y}\}, \quad (8.4)$$

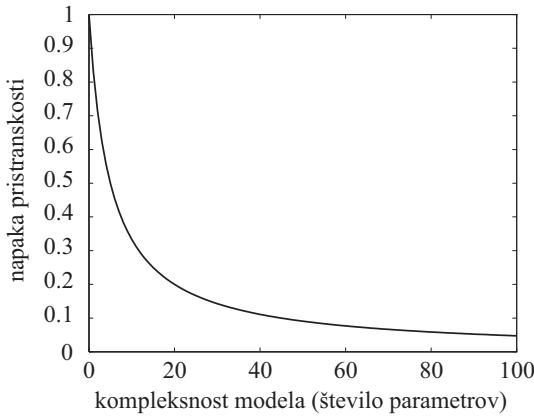
kjer je y_u nemerljiv izhod procesa, \hat{y} pa izhod modela. Izhod modela lahko obravnavamo kot naključno spremenljivko zaradi narave spremenljivk s pomočjo katerih ocenimo parametre modela.

Pristranskost je navadno velika, če imamo nefleksibilno strukturo modela in majhna, če je model bolj prilagodljiv oziroma kompleksen. To pomeni, da je pristranskost kvalitativno odvisna od števila parametrov modela, kot to prikazuje slika 8.2. Jasno je tudi, da se v primeru enostavnih modelov pristranskost hitro zmanjšuje. V primeru bolj kompleksnih pa z dodajanjem parametrov pridemo v nasičenje, kjer se pristranskost ne spreminja več. Število dodanih parametrov moramo obravnavati relativno glede na obstoječ model.

Pomemben cilj modeliranja je majhna pristranskost modela, kar zahteva primerno kompleksnost modela. Seveda se s povečevanjem kompleksnosti modela vpliv nadaljnjega povečevanja števila parametrov zmanjšuje. Če bi v napaki modela nastopala samo pristranskost, bi lahko s povečevanjem kompleksnosti prišli do poljubno majhne pristranskosti. Temu seveda ni tako, ker v napaki modela nastopa tudi napaka variance.

8.2.2 Napaka variance

Napako variance dobimo zaradi napake pri oceni parametrov modela. To je deviacija parametrov modela od pravih vrednosti. Ker so v praksi parametri modela ocenjeni na osnovi končne množice osumljenih podatkov, se ti parametri običajno razlikujejo od pravih vrednosti. Napaka variance je del napake modela, ki jo dobimo zaradi nenatančne ocene parametrov modela. Za linearne



Slika 8.2. Značilna odvisnost med pritranskostjo in številom parametrov modela.

modele je ta napaka izračunana eksplisitno z enačbo 4.26 (glej poglavje 4.1.2). Napaka variance določa velikost intervala zaupanja.

Za neskončno množico podatkov bi bila napaka variance enaka nič, če bi uporabili pravo metod identifikacije, to je ocenjeni parametri bi konvergirali k pravim. V drugi skrajnosti pa bi bila napaka variance v primeru, če bi imeli samo toliko podatkov kolikor je število parametrov modela, maksimalna. Glede na to je mogoče doseči idealno prileganje modela učnim podatkom, kar pa seveda pomeni, da tak model opisuje tudi šum v podatkih. Tak model se bo seveda veliko slabše obnašal na validacijskih podatkih, ki vsebujejo drugačno realizacijo šuma. Za manjše učne množice se lahko zgodi, da število prostostnih stopenj modela (število parametrov) celo preseže število podatkov. V tem primeru napaka variance postane zelo velika. Iz tega lahko zaključimo, da mora biti število podatkov na osnovi katerih ocenjujemo parametre modela ustrezno večje od števila parametrov modela.

Napako variance lahko izrazimo z

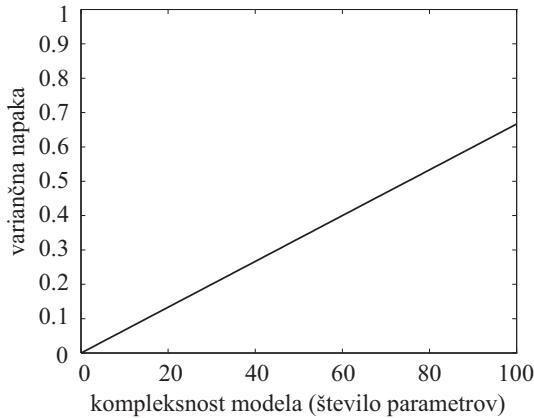
$$\text{napaka variance} = E\{(\hat{y} - E\{\hat{y}\})^2\}. \quad (8.5)$$

Zgornji zapis lahko razložimo na naslednji način. Predpostavimo, da imamo na vhodu procesa večkrat enak signal. Glede na naravo šuma n bomo dobili različne izhodne odzive. Če na osnovi teh različnih množic podatkov ocenimo parametre modela, potem enačba 8.5 določa srednje kvadratično odstopanje teh izhodov modelov od srednje vrednosti izhoda modela. V primeru procesa brez šuma bi bili vsi modeli enaki in posledično napaka variance enaka nič. Lahko pokažemo [78], da v primeru velikega števila učnih podatkov napaka variance narašča približno linearno s številom parametrov modela

$$\begin{aligned} \text{napaka variance} &\sim \sigma^2 \frac{n}{N} \\ &\sim \text{varianca šuma} \cdot \frac{\text{stevilo parametrov modela}}{\text{stevilo učnih podatkov}}. \end{aligned} \quad (8.6)$$

Ta trditev aproksimativno drži ne glede na vrsto modela. Seveda pa velja samo za naskončno veliko število podatkov v učni množici. Ne glede na to, pa je enačba 8.6 dober napotek za praktično načrtovanje. Več pozornosti pa je potrebno posvetiti obravnavi, ko je število učnih podatkov majhno, ali pa ko je model zelo kompleksen.

Slika 8.3 kaže odvisnost napake variance od števila parametrov modela. Hkrati lahko ugotovimo, da je strmina premice odvisna od variance šuma σ^2 in števila učnih podatkov N . Večja varianca šuma vodi do večje napake variance. Večje število učnih podatkov pa k manjši napaki



Slika 8.3. Tipična odvisnost napake variance in števila parametrov modela.

variance. Na ta način lahko z zadostnim številom učnih podatkov kompenziramo napako zaradi šuma. Ocena variance napake (enačba 4.27 v poglavju 4.1.2) je za nelinearne procese v splošnem težko izračunljiva, ker je napaka v večini posledica pristranskosti.

Za zelo fleksibilne modele lahko pristranskost zanemarimo in je napaka modela v celoti odvisna od napake variance. Kvadrat napake modela je približno enak napaki variance; glej enačbo 8.2. Kvadrat napake modela je proporcionalen z $1/N$ in napaka modela je proporcionalna $1/\sqrt{N}$. To velja splošno za vse tipe modelov. Za dovolj fleksibilne modele velja

$$\text{napaka modela} \approx \sqrt{\text{napaka variance}} \sim \sigma \frac{\sqrt{n}}{\sqrt{N}}. \quad (8.7)$$

Za primere iz prakse pa navadno velja, da je napaka modela bistveno odvisna tudi od pristranskosti. Poudariti je potrebno pomen števila podatkov učne množice. Iz enačbe 8.7 lahko vidimo, da je omejitev kvalitete modela dana s številom učnih podatkov N in kvaliteto meritev (σ^2).

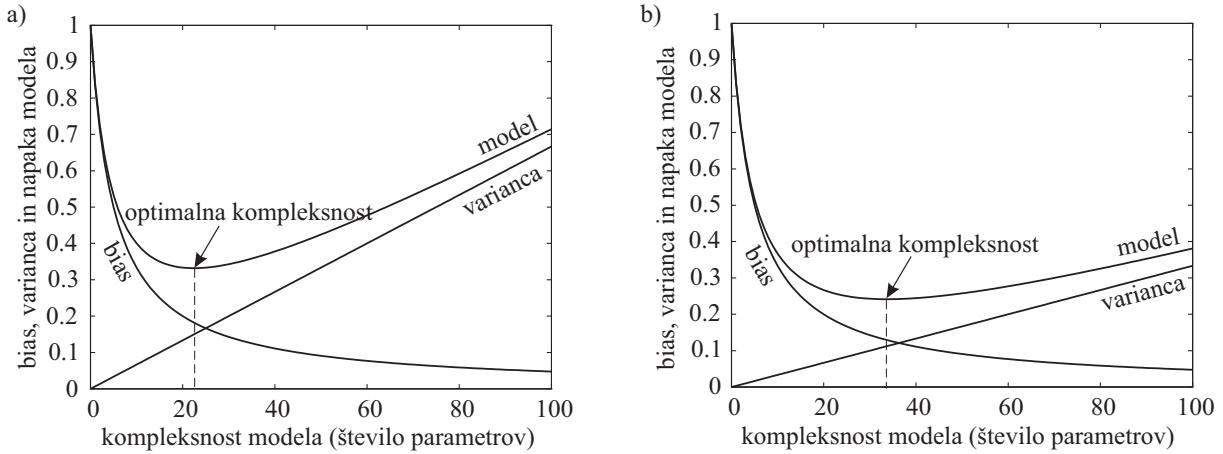
8.2.3 Kompromis med pristranskostjo in napako variance

Slika 8.4 a povzema vpliv pristranskosti in napake variance na napako modela. Vpliva si v smislu napake modela nasprotujeta. Problem imenujemo *dilema pristranskosti in variance*.

Slika 8.4 b osvetluje problem kompromisa med pristranskostjo in varianco za učno množico, ki je dvakrat večja kot na sliki 8.4 a. To vodi k napaki variance, ki je dvakrat manjša. Posledično je lahko večja tudi optimalna kompleksnost modela kot je to na sliki 8.4 a. Intuitivno lahko rečemo, da večje število učnih podatkov omogoča tudi oceno večjega števila parametrov modela. Omenimo lahko, da je isti efekt dosežen tudi z manjšo varianco šuma.

Pomembno je razumeti, da optimalna kompleksnost modela na sliki 8.4 ni nujno tudi na splošno najboljša rešitev. Je samo najboljši kompromis za izbrani razred modelov. Na splošno lahko seveda obstaja struktura, ki je bolj primerna za opis konkretnega primera procesa. Na ta način se pristranskost lahko bistveno zmanjša že pri enaki kompleksnosti. Problem pri izbiri kompromisne rešitve je v tem, da v praksi pristranskosti in variance ne poznamo.

Na koncu moramo komentirati še uporabo učne in testne množice podatkov. Če bi za oceno kvalitete modela uporabili učno množico podatkov, bi bila napaka variance enaka nič. Napaka je v tem primeru odvisna samo od pristranskosti. Napaka variance je prisotna samo, če je uporabljen druga množica podatkov z drugačno realizacijo šuma. Razlog za to je v tem, da je varianca napake posledica napak parametrov modela zaradi konkretnene realizacije šuma v učnih podatkih. Če



Slika 8.4. Kompromis med pristranskostjo in varianco. Napaka modela je sestavljena iz napake pristranskosti in variance : a) relativno visoka napaka variance, b) nižja napaka variance zaradi manjše variance šuma ali večjega števila učnih podatkov kot v a).

uporabimo iste podatke za učenje in za preizkus kvalitete modela, potem napak v parametrih ne moremo odkriti, ker je natančno ta realizacija šuma skrita v parametrih modela.

To pomeni, da napaka na učni množici (ki je približno enaka pristranskosti) pada s kompleksnostjo modela, napaka na testni množici (ki je enaka pristranskosti in varianci) pa raste. Če se ne zavedamo tega efekta, potem pridemo do modela, ki se dobro obnaša na učnih podatkih in slabo na testnih. Efekt se imenuje problem preparametrizacije (*ang. overfitting*) (nizka pristranskost in visoka varianca) in podparametrizacije (*ang. underfitting*), ki je rezultat preenostavne zgradbe modela (visoka pristranskost in nizka varianca).

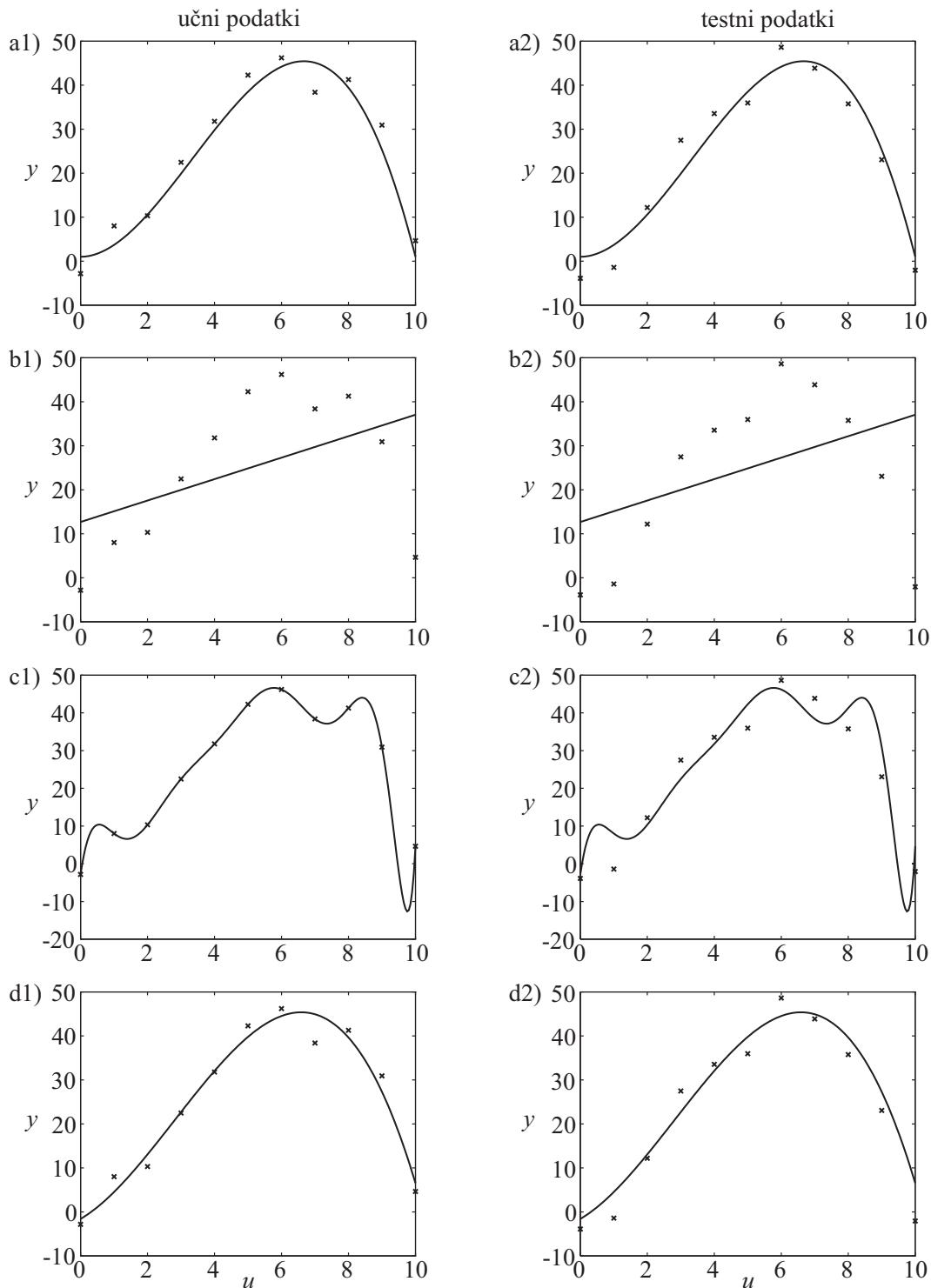
Slika 8.5 prikazuje problem podparametrizacije in preparametrizacije. Na levi strani je prikazan model na učnih podatkih, na desni pa na testnih podatkih.

8.3 Ocena kvalitete modela

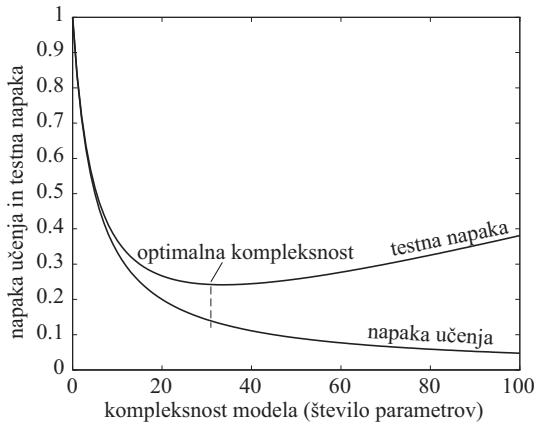
Model, ki ga ocenjujemo na učnih podatki, ne vsebuje variančne napake. Zaradi tega je potrebno evaluacijo izvesti na novi množici testnih podatkov. Cilj je dobiti model, ki se bo dobro obnašal na novih, neznanih podatkih, ki niso bili uporabljeni za učenje; glej poglavje 8.3.1. Navzkrižna validacija je bolj kompleksna metoda, ki pa v osnovi vsebuje ta princip; glej poglavje 8.3.2.

8.3.1 Učenje, preverjanje in potrjevanje modela

Najenostavnejša metoda za ocenjevanje kvalitete modela (vrednotenje) je ocenjevanje na novih podatkih, ki niso bili uporabljeni za učenje. Zaradi tega je potebno podatke razdeliti na učno množico in testno množico. V primeru velikega števila podatkov to ni problem. Paziti je potrebno, da so podatki informacijsko reprezentativni, da zajemajo vse delovne rezime procesa enakovredno. To seveda postane problem takrat, ko je število podatkov majhno. Če določena področja delovanja vsebujejo manjše število podatkov, potem ne moremo pričakovati dobrega modela za to področje. Če v testni množici manjkajo določeni podatki, prav tako ne moremo pričakovati zanesljive ocene o kvaliteti modela. V primeru majhnega števila podatkov si pomagamo z metodo navzkrižnega potrjevanja modela.

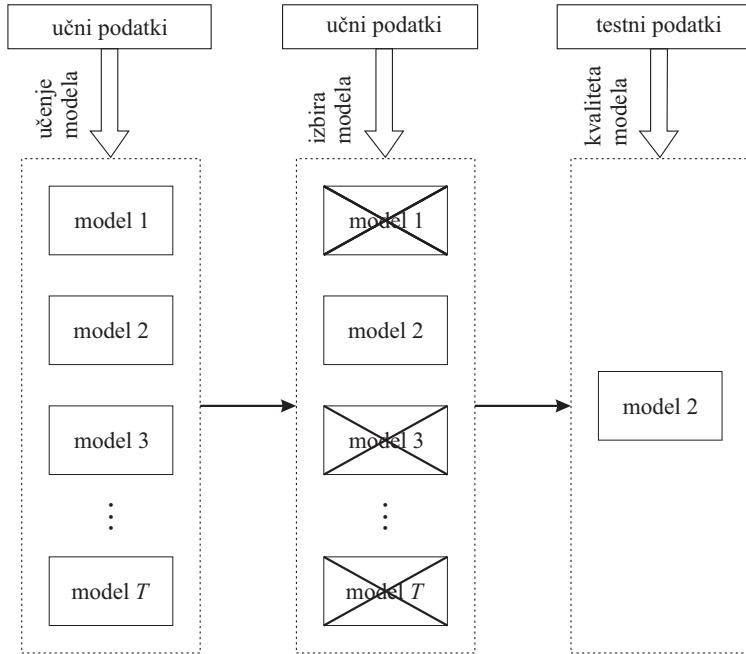


Slika 8.5. Prikaz podparametrizacije in preparametrizacije: a) originalna funkcija, b) aproksimacija s polinomom 1. reda (podparametrizacija: velika pristranskočnost, majhna varianca), c) aproksimacija s polinomom 10. reda (preparametrizacija: nična pristranskočnost, velika varianca), d) aproksimacija s polinomom 4. reda (dobra pristranskočnost in varianca, kompromis).



Slika 8.6. Učna in testna napaka. Učna napaka ne vsebuje napake variance, testna napaka pa vsebuje celotno napako modela.

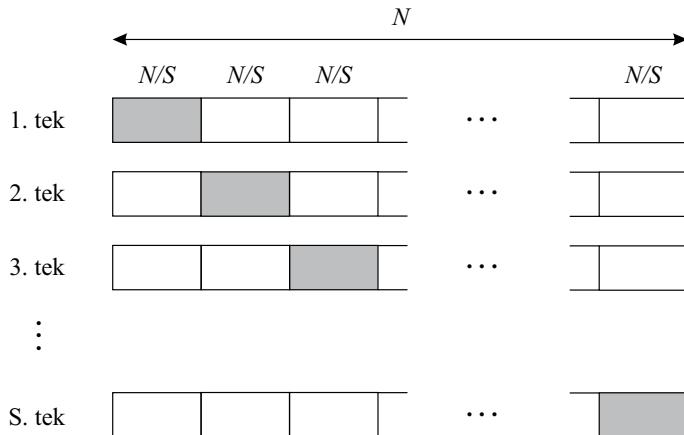
V primeru velikega števila podatkov podatke najprej razdelimo na manjše množice. Za določitev optimalne kompleksnosti modela je zelo primerna naslednja enostavna in učinkovita strategija, ki je prikazana na sliki 8.7. Učimo T modelov različnih kompleksnosti na učni množici in ocenimo njihovo kvaliteto na testni množici podatkov. Izberemo model, ki ima na učnih podatkih najmanjšo napako in ga potrdimo na testnih podatkih.



Slika 8.7. Učenje, preverjanje in potrjevanje. Različne model učimo na učnih podatkih. Preverjanje (verifikacijo) izvedemo na verifikacijski množici podatkov (lahko je enaka kot za učenje), kjer izberemo najboljšega, ki ga potem potrdimo (validiramo) na testnih podatkih.

8.3.2 Navzkrižno preverjanje

Običajno je pri identifikaciji problem s premajhnim številom podatkov. V tem primeru množico, ki vsebuje N podatkov razdelimo na S podmnožic. Nato $S - 1$ podmnožic uporabimo za učenje,



Slika 8.8. Pri navzkrižnem preverjanju je N podatkov razdeljenih na S podmnožic [10]. V vsakem teku $S - 1$ podmnožic uporabimo kot učno množico in ostanek za preverjanje. Po vseh S tekih povprečimo napako modela.

eno pa za potrjevanje modela. To proceduro ponovimo za vseh S različnih mogočih kombinacij; glej sliko 8.8. Na ta način lahko izkoristimo veliko več originalnih podatkov za učne namene. Ker pa je učenje in testiranje ponovljeno S krat z različnimi deli originalne množice, moramo na koncu povprečiti vse testne napake modela, da dobimo zanesljivo oceno kvalitete modela na novih podatkih. Ta metoda se imenuje metoda navzkrižnega preverjanja (*ang. cross validation*). Priponiti velja, da je navzkrižno preverjanje uporabljen samo za oceno kvalitete modela na novih podatkih. Za končno uporabo moramo model učiti na celotni množici učnih podatkov, da uporabimo vso možno informacijo.

Tipična vrednost za število podmnožic je $S = 10$. Ta delitev določa, da uporabimo 9/10 podatkov za učenje, namesto 1/2, kot je to običajno pri enostavnem pristopu. Seveda pa metoda zahteva 10 krat več računskega časa. Izvedba navzkrižnega preverjanja pri $S = 2$ je še vedno boljša kot enostavni pristop, ker je pri navzkrižnem postopku celotna učna množica uporabljena pri preverjanju. Vseh S tekov je upoštevanih.

Skrajna situacija nastopi, ko je $S = N$, kar je lahko uporabno v primerih z izredno majhnim številom podatkov. Metoda se imenuje izpusti enega (*ang. leave-one-out*) . V tem primeru izpuščimo en vzorec in na vseh ostalih učimo model. To storimo za vse vzorce, N -krat, kar zahteva N -kratni računski čas.

Obstajajo še druge močne in računalniško zahtevne metode.

8.4 Povzetek

Značilnosti procesa, količina in kvaliteta podatkov, ki so na voljo, predznanja in tip modela implizirajo izbiro optimalne kompleksnosti modela. Vsak dodatni parameter prispeva k fleksibilnosti (prostostna stopnja) modela. Hkrati pa je pri večjem številu parametrov njihovo ocenjevanje težje. Napako modela lahko razdelimo na dva dela: pristranskost in varianco. Pristranskost je posledica nazadostne fleksibilnosti modela. Variančna napaka pa je posledica nenatančne ocene parametrov modela. Pristranskost pada s številom parametrov, varianca pa s številom parametrov narašča. S kompromisom med pristranskostjo in varianco določimo optimalno kompleksnost modela.

Za določanje kompromisa kompleksnosti modela je potrebno ugotoviti napako modela na testnih podatkih. Tega ni moč storiti na učnih podatkih, ker v napaki modela ni vsebovana va-

riančna napaka. Optimalno strukturo lahko določimo tudi z eksplisitno strukturno optimizacijo, ki pa je računalniško zahtevna in dolgotrajna.

Del III

Arhitekture aproksimacijskih modelov

9. Uvod v aproksimacijske modele

V tem delu se bomo srečali z najpogostejšimi aproksimacijskimi modeli, ki jih srečamo na najrazličnejših področjih. Del je razdeljen na štiri poglavja. Najprej po uvodu sledi poglavje 10 o klasičnih aproksimacijskih modelih, od linearnih do polinomskih. V poglavju 11 je predstavljen in analiziran nevronska model. V poglavju 12 so predstavljene tri različne oblike mehkih in nevro-mehkih (*ang. neuro-fuzzy*) modelov.

Za uvod pa si bomo ogledali nakaj osnovnih stvari v zvezi z nelinearnimi aproksimatorji v obliki statičnih modelov. Predstavljen je zapis nelinearnih aproksimatorjev v obliki baznih funkcij in njegova razširjena oblika. Za konec pa so podani kriteriji za ocenjevanje različnih arhitektur modelov.

9.1 Modeli na osnovi baznih funkcij

Večino uporabnih načinov aproksimacije funkcije $f(\cdot)$ lahko zapišemo na osnovi baznih funkcij

$$\hat{y} = \sum_{i=1}^M \theta_i^{(l)} \Phi_i (\underline{u}, \underline{\theta}_i^{(nl)}) . \quad (9.1)$$

Izhod y modeliramo (aproksimiramo) z uteženo vsoto M baznih funkcij $\Phi_i(\cdot)$. Bazne funkcije so utežene z *linearimi* parameteri $\theta_i^{(l)}$ in so odvisne od vhodnega vektorja \underline{u} in množice *nelinearnih* parameterov $\underline{\theta}_i^{(nl)}$. Če želimo modelirati nelinearno funkcijo, potem morajo biti bazne funkcije nelinearne. Zato so parametri teh funkcij $\underline{\theta}_i^{(nl)}$ nujno nelinearni.

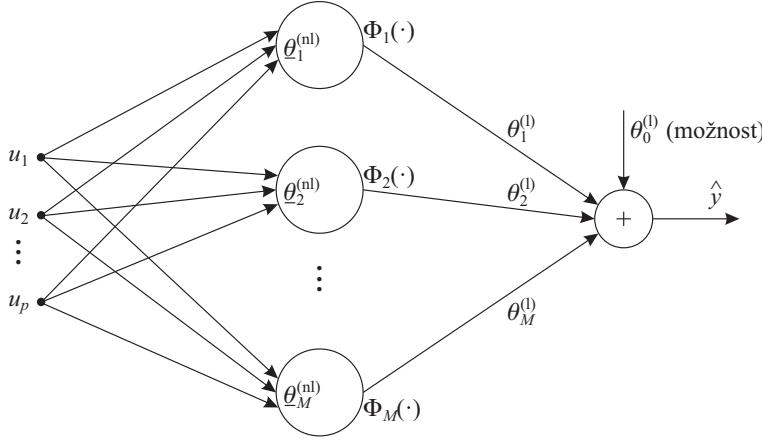
Pogosto imajo modeli tudi enosmerni člen (*ang. off-set parameter, bias*), ki določa delovno točko. V takih primerih je v množico baznih funkcij vključena še konstantna bazna funkcija $\Phi_0(\cdot)$, ki ima na celotnem intervalu vrednost enako 1. Ustrezeni linearni parameter $\theta_0^{(l)}$ vnaša enosmerni člen

$$\hat{y} = \sum_{i=0}^M \theta_i^{(l)} \Phi_i (\underline{u}, \underline{\theta}_i^{(nl)}) \quad \text{in } \Phi_0(\cdot) = 1 . \quad (9.2)$$

Zapis z baznimi funkcijami v enačbi 9.1 in 9.2 lahko prikažemo kot mrežo na sliki 9.1. Na splošno so lahko bazne funkcije $\Phi_i(\cdot)$ različnih tipov za vsako vozlišče. Če imamo enake bazne funkcije z različnimi nelinearnimi parametri, potem taki mreži rečemo *umetna nevronska mreža*. Vozlišča te mreže na sliki 9.1 imenujemo *nevroni*. Umetne nevronske mreže so obravnavane v poglavju 11.

9.1.1 Globalne in lokalne bazne funkcije

V splošnem so lahko bazne funkcije $\Phi_i(\cdot)$ kakršnekoli oblike. Še posebno v mehkih sistemih in nevronskeh mrežah so sestavljene iz elementarnih funkcij; glej poglavje 11.1. Navadne enodimensionalne bazne funkcije so prikazane na sliki 9.2 in 9.3. Lahko jih ločimo na:



Slika 9.1. Mreža baznih funkcij. Vsako vozlišče predstavlja bazno funkcijo, ki je odvisna od nonlinearnih parametrov $\underline{\theta}_i^{(nl)}$.

- *globalne* bazne funkcije, ki so različne od nič v celotnem v celotnem intervalu vhodne spremenljivke; slika 9.2,
- *lokalne* bazne funkcije, ki so različne od nič samo lokalno; slika 9.3.

9.1.2 Linearni in nelinearni parametri

Neodvisno od arhitekture modela lahko na osnovi baznih funkcij zaključimo nekaj osnovnih lastnosti za širok razred modelov:

- Model je linearen v svojih parametrih $\underline{\theta}_i^{(l)}$, če so bazne funkcije popolnoma definirane, to je, nelinearni parametri so popolnoma definirani. Linearne parametre lahko definiramo s poljubno linearno optimizacijsko tehniko, npr. najmanjšimi kvadrati; glej poglavje 4. Regresijska matrika \underline{X} in vektor parametrov $\underline{\theta}^{(l)}$ sta enaka

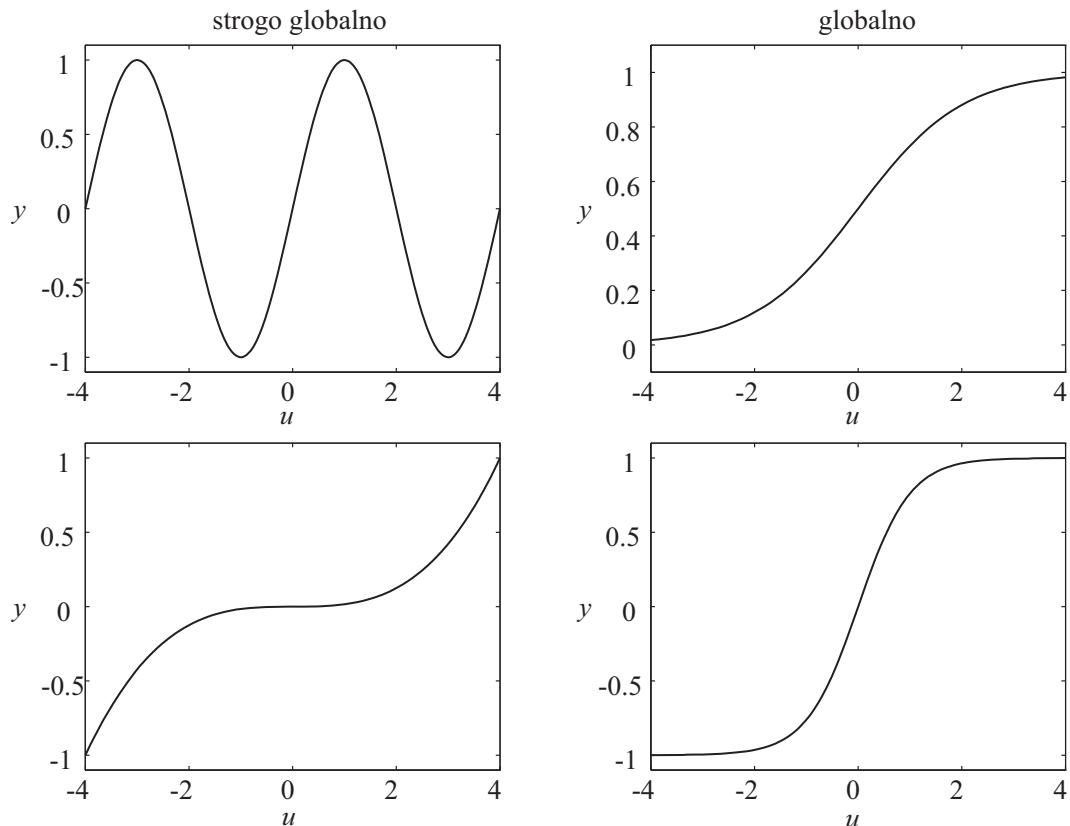
$$\underline{X} = \begin{bmatrix} 1 & \Phi_1(\underline{u}(1)) & \Phi_2(\underline{u}(1)) & \cdots & \Phi_M(\underline{u}(1)) \\ 1 & \Phi_1(\underline{u}(2)) & \Phi_2(\underline{u}(2)) & \cdots & \Phi_M(\underline{u}(2)) \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \Phi_1(\underline{u}(N)) & \Phi_2(\underline{u}(N)) & \cdots & \Phi_M(\underline{u}(N)) \end{bmatrix}, \quad \underline{\theta}^{(l)} = \begin{bmatrix} \theta_0^{(l)} \\ \theta_1^{(l)} \\ \theta_2^{(l)} \\ \vdots \\ \theta_M^{(l)} \end{bmatrix}, \quad (9.3)$$

kjer je N število podatkov, ki jih rabimo za učenje, odvisnost bazne funkcije od nelinearnih parametrov pa tukaj zaradi enostavnosti ni upoštevana. Prvi stolpec v \underline{X} in prvi člen v $\underline{\theta}^{(l)}$ predstavlja enosmerni člen.

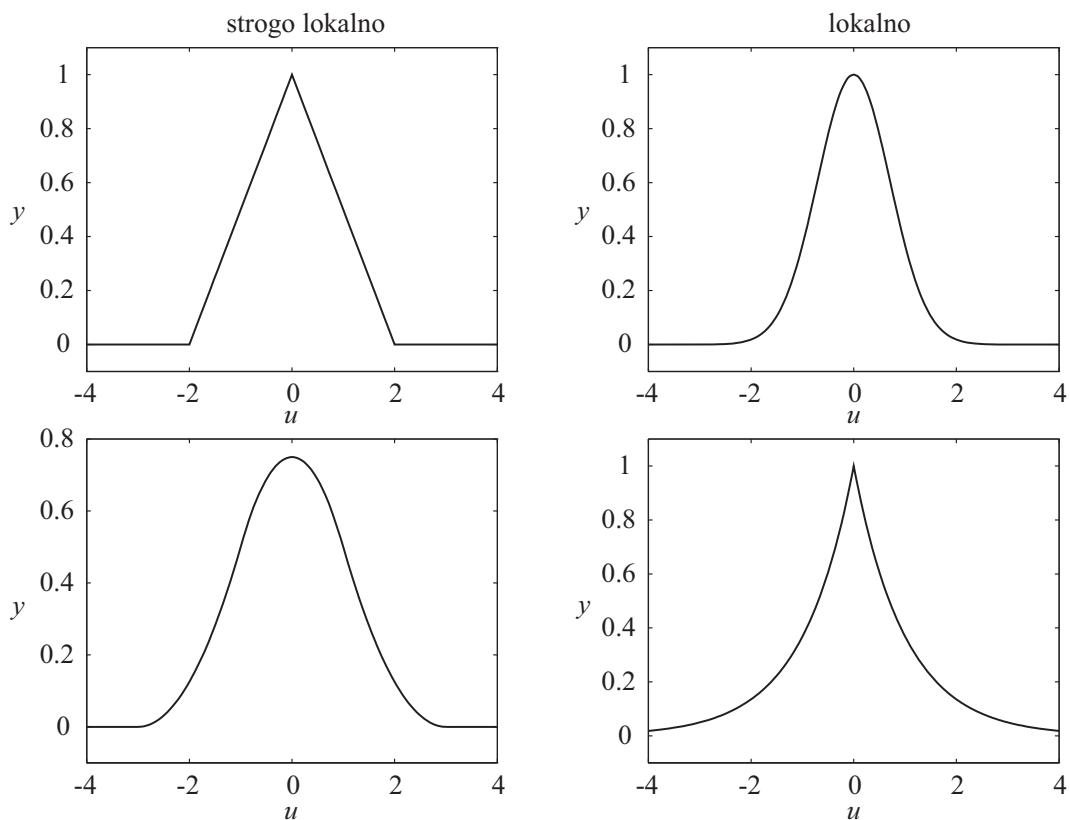
- Optimizacija nelinearnih parametrov $\underline{\theta}_i^{(nl)}$ je veliko bolj zahtevna, saj zahteva nonlinearno lokalno ali globalno optimizacijo (poglavje 5 in 6). Nelinearne parametre običajno optimiramo z eno od naslednjih tehnik učenja:

- nelinearno optimizacijo,
- nenadzorovanim učenjem, npr. rojenjem,
- uporabo apriornega znanja.

Nelinearni parametri vplivajo na bazne funkcije. V splošnem definirajo njihov položaj v prostoru problema, ki ga definira vhodni vektor \underline{u} , obliko funkcije in njeno širino. Na ta način nelinearni parametri omogočajo nastavljanje položaja in oblike bazne funkcije, da prilagodimo



Slika 9.2. Splošne globalne bazne funkcije. Bazna funkcija je *stogo globalna* (levo), če je globalen tudi odvod funkcije, na primer, sinusna funkcija (zgoraj) ali polinom (spodaj). Drugače je *nestrogo globalna*, npr. sigmoidna funkcija (zgoraj) ali tanh (spodaj).



Slika 9.3. Splošne lokalne bazne funkcije. Bazna funkcija je strogo lokalna (levo), če je enaka nič zunaj aktivacijskega območja, t.j., trikotna funkcija (zgoraj) ali B-zlepek 3. reda (spodaj). Drugače je lokalna, npr. Gaussova (zgoraj) ali dvojna eksponencialna funkcija (spodaj).

model in povečamo njegovo fleksibilnost. Če želimo uporabiti nelinearno optimizacijo, potrebujemo gradient izhoda modela glede na parametre modela. To lahko vidimo iz kriterijske funkcije:

$$I = \frac{1}{2} \sum_{i=1}^N e(i)^2 = \frac{1}{2} \sum_{i=1}^N (y(i) - \hat{y}(i))^2, \quad (9.4)$$

kjer je N število meritev, e napaka modela in y in \hat{y} sta izhod procesa in modela. Za aplikacijo gradientne metode optimizacije je potrebno izračunati gradient kriterijske funkcije glede na parameter θ

$$\frac{\partial I}{\partial \theta} = \sum_{i=1}^N e(i) \frac{\partial e(i)}{\partial \theta} = - \sum_{i=1}^N e(i) \frac{\partial \hat{y}(i)}{\partial \theta}. \quad (9.5)$$

Gradient izhoda modela glede na parametre $\partial \hat{y}(i)/\partial \theta$ je nujen za izvedbo metode.

Prednosti in slabosti različnih arhitektur modelov so navadno zelo povezane z ustreznimi metodami za optimizacijo nelinearnih parametrov. Včasih je zelo zanimivo vprašanje ali je bolje (i) uporabiti model s fiksнимi nelinearnimi parametri in optimirati samo linearne parametre ali (ii) ali optimirati model z nelinearnimi parametri.

9.2 Razširjen zapis baznih funkcij

Zapis baznih funkcij iz enačbe 9.1 lahko razširimo v bolj fleksibilno strukturo z zamenjavo linearnih parametrov s funkcijo (linearno parametrizirano) $L_i(\cdot)$:

$$\hat{y} = \sum_{i=1}^M L_i \left(\underline{u}, \underline{\theta}_i^{(l)} \right) \Phi_i \left(\underline{u}, \underline{\theta}_i^{(nl)} \right). \quad (9.6)$$

Dokler je $L_i(\cdot)$ linearno parametrizirana lahko parametre $\underline{\theta}_i^{(l)}$ ocenimo z linearno optimizacijo, če seveda poznamo bazne funkcije $\Phi_i(\cdot)$. V najenostavnejšem primeru je $L_i = \theta_i^{(l)}$ in dobimo osnovno bazno funkcijo. Druga pogosta alternativa pa je izbira $L_i(\cdot)$ kot linearne kombinacije komponent vektorja \underline{u} . Dobimo torej:

$$\hat{y} = \sum_{i=1}^M (w_{i0} + w_{i1}u_1 + w_{i2}u_2 + \dots + w_{ip}u_p) \Phi_i \left(\underline{u}, \underline{\theta}_i^{(nl)} \right), \quad (9.7)$$

kjer je

$$\underline{\theta}_i^{(l)} = [w_{i0} \ w_{i1} \ w_{i2} \ \dots \ w_{ip}]^T. \quad (9.8)$$

Bazne funkcije niso več utežene s konstantnimi utežmi (parametri) temveč z linearnim podpodmodelom. Na splošno so lahko podmodeli $L_i(\cdot)$ izbrani poljubno kompleksno.

9.3 Kriterij kvalitete aproksimacijskega modela

Aproksimacijski modeli morajo zadovoljiti določene kriterije kvalitete:

- *Interpolacija*: lastnosti modela znotraj območja učne množice.
- *Ekstrapolacija*: lastnosti modela izven območja učne množice.
- *Lokalnost*: so bazne funkcije lokalne, globalne, striktno globalne, ali striktno lokalne.

- *Natančnost*: kako natančen je model glede na število parametrov.
- *Gladkost*: izhoda modela.
- *Občutljivost na šum*: kako šum vpliva na obnašanje modela.
- *Optimizacija parametrov*: kako lahko optimiramo linearne in nelinearne parametre modela.
- *Struktturna optimizacija*: kako lahko optimiramo strukturo in kompleksnost modela.
- *Sprotna adaptacija*: kako lahko sproti adaptiramo parametre modela in ali so zanesljivi.
- *Hitrost učenja*: kako hitro lahko naučimo model, njegovo strukturo in parametre.
- *Hitrost izračuna modela*: kako hitro lahko izračunamo izhod modela pri določenem vhodu.
- *Problem dimenzij*: kako se model obnaša pri povečevanju števila vhodnih spremenljivk.
- *Interpretacija*: ali lahko interpretiramo parametre in strukturo modela v smislu delovnja procesa, ki ga modeliramo.
- *Vključevanje apriornega znanja*: ali lahko v model vključimo apriorno znanje.
- *Uporaba*: kako razširjena je arhitektura modela.

10. Linearni in polinomski modeli

V tem poglavju bomo analizirali klasične, tradicionalne arhitekture modelov za opis nelinearnih statičnih procesov. Najenostavnejši je linearni model, ki ga lahko uporabimo, če je nelinearnost šibka. Ogledali si ga bomo v prvem podpoglavlju, v naslednjem pa si bomo ogledali polinomski model.

10.1 Linearni model

Linearni model lahko aproksimira nelinearno obnašanje procesa samo takrat, ko je ta nelinearnost šibkega značaja. Prednost linearnega modela je v tem, da je enostaven in ima majhno število parametrov. Posebno v primeru, ko imamo zelo majhno število ošumljenih učnih podatkov, je opis z linearnim modelom lahko dobra izbira, saj bi imeli v primeru kompleksnega nelinearenega modela veliko večjo variančno napako. V takih primerih je aproksimacija z linearnim modelom lahko upravičena.

Linearni model zapišemo kot

$$\hat{y} = w_0 + w_1 u_1 + w_2 u_2 + \dots + w_p u_p , \quad (10.1)$$

ali

$$\hat{y} = \sum_{i=0}^p w_i u_i \quad \text{za } u_0 = 1 . \quad (10.2)$$

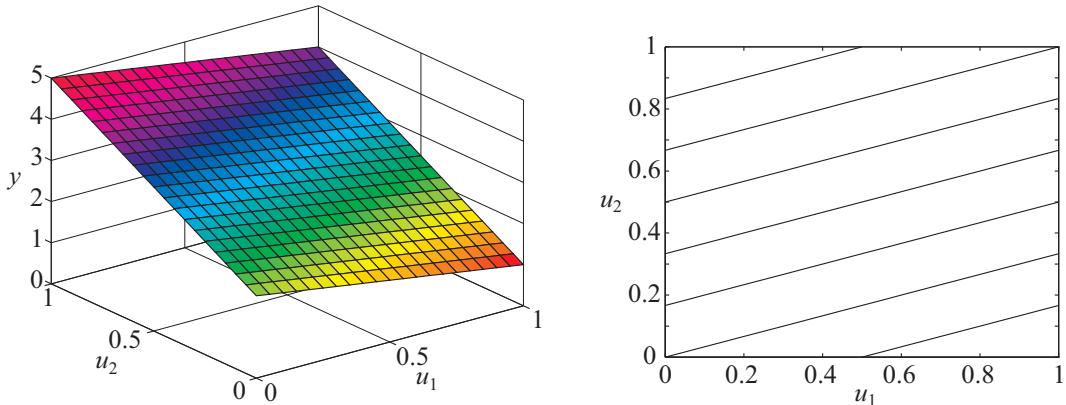
Slika 10.1 kaže linearni model za dve vhodni spremenljivki in eno izhodno, ki je predstavljen z ravnino v tridimenzionalnem prostoru. V primeru višjih dimenzij dobimo hiperravnino in enosmerno vrednost w_0 , ki je definirana z $\underline{u} = \underline{0}$. Parametri $w_i, i > 0$ definirajo strmino hiperravnine v smereh spremenljivk u_i . V smislu bazne formulacije so vhodi u_i bazne funkcije, koeficient w_i pa so linearni parametri. Ta zapis ne vsebuje nelinearnih parametrov.

Parametre linearnega modela ocenimo z metodo najmanjših kvadratov (poglavlje 4.1) z naslednjo regresijsko matriko \underline{X} in vektorjem parametrov $\underline{\theta}$

$$\underline{X} = \begin{bmatrix} 1 & u_1(1) & u_2(1) & \cdots & u_p(1) \\ 1 & u_1(2) & u_2(2) & \cdots & u_p(2) \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & u_1(N) & u_2(N) & \cdots & u_p(N) \end{bmatrix} , \quad \underline{\theta} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_p \end{bmatrix} . \quad (10.3)$$

Glavne lastnosti linearnega modela so naslednje:

- *Interpolacija* je linearна.
- *Ekstrapolacija* je linearна in zaradi tega gradient zunaj učnega področja ostane enak.
- *Lokalnost* ne obstaja. Linearni model je popolnoma globalen.



Slika 10.1. Linearni model za dva vhoda in izhod. Model procesa s funkcijo $y = 2 - u_1 + 3u_2$.

- *Natančnost* je odvisna od nelinearnosti procesa. Pada z nelinearnostjo.
- *Gladkost* je popolna, saj je gradient hiperravnine konstanten na celotnem področju.
- *Občutljivost na šum* je zelo nizka, saj za učenje lahko uporabimo vse podatke.
- *Optimizacija parametrov* je enostavna, saj lahko uporabimo metodo najmanjših kvadratov.
- *Strukturna optimizacija* je učinkovita z uporabo tehnike izbire strukture po ortogonalnih najmanjših kvadratih.
- *Sprotna adaptacija* je izvedljiva z metodo rekurzivnih najmanjših kvadratov.
- *Hitrost učenja* je zelo velika. Čas izračuna narašča s tretjo potenco glede na število parametrov.
- *Hitrost izračuna modela* je velika, ker potrebujemo samo p množic in sumacij.
- *Problem dimenzij* je majhen, ker število parametrov narašča linearno z vhodno dimenzijo.
- *Interpretacija* je možna skozi vpliv spremenljivk na izhod.
- *Vključevanje mejitev* izhoda in parametrov je možno, če za optimizacijo parametrov uporabimo optimizacijsko tehniko.
- *Vključevanje apriornega znanja* je možno in zahteva posebno regresijo.
- *Uporaba* je zelo razširjena. Linearni modeli so uveljavljeni standard.

10.2 Polinomski model

Polinomska formulacija je najenostavnejša razširitev linearnih modelov. S povečevanjem stopnje polinoma dobimo bolj fleksibilne modele. Polinom stopnje k in s številom spremenljivk p zapišemo kot

$$\begin{aligned}\hat{y} &= w_0 + \sum_{i=1}^p w_i u_i + \sum_{i_1=1}^p \sum_{i_2=i_1}^p w_{i_1 i_2} u_{i_1} u_{i_2} + \dots \\ &+ \sum_{i_1=1}^p \dots \sum_{i_k=i_{k-1}}^p w_{i_1 \dots i_k} u_{i_1} \dots u_{i_k}.\end{aligned}\quad (10.4)$$

Enosmerni člen in prva vsota definirata linearni model, druga vsota definira kvadratični model s členi u_1^2 , $u_1 u_2$ in tako dalje, in zadnja vsota definira člen stopnje k kot npr. členi u_1^k , $u_1^{k-1} u_2$, itd.

V vsaki vsoti imamo število parametrov, ki je odvisno od dimenzije problema p in reda členov v tej vsoti, ki ga označimo z r , kjer gre $r = 0, \dots, k$. Pri tem je k najvišji red v polinomskem

zapisu modela. Število kombinacij izbora r elementov iz množice p elementov, če dovolimo, da se v izbranih kombinacijah elementi iz p ponavljajo je

$$m_r = \binom{p+r-1}{r} . \quad (10.5)$$

Število vseh možnih parametrov v polinomskem zapisu potem lahko zapišemo kot vsoto členov 0. reda, 1. reda, do k . reda,

$$M = \sum_{r=0}^k m_r = \binom{p-1}{0} + \binom{p}{1} + \cdots + \binom{p+k-1}{k} . \quad (10.6)$$

Ker velja, da je

$$\binom{a}{b} + \binom{a}{b+1} = \binom{a+1}{b+1} , \quad (10.7)$$

lahko z uporabo Pascalovega trikotnika

$$\begin{array}{ccccccccccccc} & & & & & & \binom{0}{0} & & & & & & \\ & & & & & & & \binom{1}{0} & \quad \binom{1}{1} & & & & \\ & & & & & & & & \binom{2}{0} & \quad \binom{2}{1} & \quad \binom{2}{2} & & \\ & & & & & & & & & \binom{3}{0} & \quad \binom{3}{1} & \quad \binom{3}{2} & \quad \binom{3}{3} & \\ & & & & & & & & & & \binom{4}{0} & \quad \binom{4}{1} & \quad \binom{4}{2} & \quad \binom{4}{3} & \quad \binom{4}{4} \\ & & & & & & & & & & & \binom{5}{0} & \quad \binom{5}{1} & \quad \binom{5}{2} & \quad \binom{5}{3} & \quad \binom{5}{4} & \quad \binom{5}{5} \\ & & & & & & & & & & & & \binom{6}{0} & \quad \binom{6}{1} & \quad \binom{6}{2} & \quad \binom{6}{3} & \quad \binom{6}{4} & \quad \binom{6}{5} & \quad \binom{6}{6} \end{array}$$

hitro pokažemo, da je število parametrov v polinomskem zapisu stopnje k in pri številu spremenljivk p enako

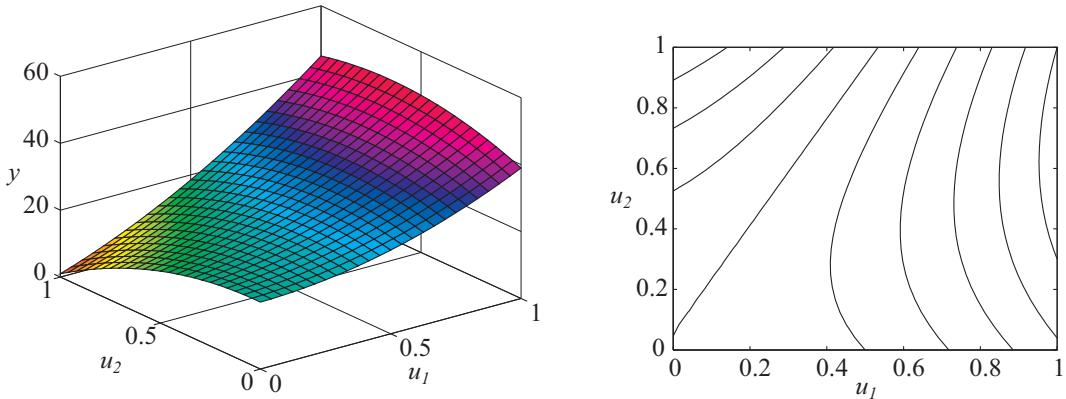
$$M = \binom{p+k}{k} = \frac{(p+k)!}{p! k!} . \quad (10.8)$$

Na ta način imamo $M - 1$ baznih funkcij in M parametrov. To pomeni, da ima polinom tretjega reda ($k = 3$) pri treh spremenljivkah ($p = 3$) v popolni konfiguraciji $M = 20$ parametrov.

Polinom laho sedaj zapišemo kot [66]

$$\hat{y} = \sum_{i=0}^{M-1} \theta_i x_i \quad \text{za } x_0 = 1 , \quad (10.9)$$

kjer $\theta_i x_i$, $i = 0, \dots, M-1$, ustrezajo i -temu členu v enačbi 10.4. V bazni formulaciji iz enačbe 9.1 člen $\Phi_i(\cdot)$ ustreza členu x_i . Zapis je linearen v parametrih θ_i in ne vsebuje nelinearnih parametrov. Polinom drugega reda v primeru treh vhodov je enak



Slika 10.2. Polinom drugega reda z dvema vhodoma in konture. Model ima funkcijo $y = 20 + u_1 + u_2 + 15u_1^2 + 20u_1u_2 - 20u_2^2$.

$$\begin{aligned}\hat{y} &= \theta_0 + \theta_1 u_1 + \theta_2 u_2 + \theta_3 u_3 \\ &\quad + \theta_4 u_1^2 + \theta_5 u_1 u_2 + \theta_6 u_1 u_3 + \theta_7 u_2^2 + \theta_8 u_2 u_3 + \theta_9 u_3^2.\end{aligned}\quad (10.10)$$

Slika 10.2 prikazuje karakteristike tega polinoma.

Parametre modela lahko ocenimo z metodo najmanjših kvadratov. Regresijska matrika \underline{X} in vektor parametrov $\underline{\theta}$ za polinomski model stopnje l in s p vhodi sta enaka

$$\underline{X} = \begin{bmatrix} 1 & u_1(1) & \cdots & u_p(1) & u_1^2(1) & \cdots & u_p^l(1) \\ 1 & u_1(2) & \cdots & u_p(2) & u_1^2(2) & \cdots & u_p^l(2) \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 1 & u_1(N) & \cdots & u_p(N) & u_1^2(N) & \cdots & u_p^l(N) \end{bmatrix} \text{ in } \underline{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_{M-1} \end{bmatrix}, \quad (10.11)$$

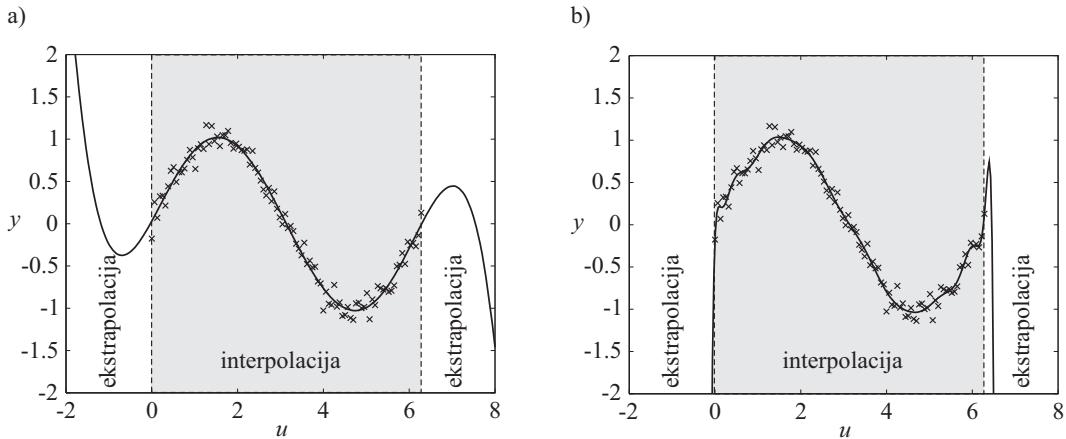
kjer je $M-1$ število baznih funkcij glede na enačbo 10.8. Ugotovimo lahko, da število parametrov in s tem kompleksnost modela močno naraste s številom vhodov p in s stopnjo polinoma l . Zato je že za manjše probleme zelo težko dobro oceniti parametre za celotno strukturo modela. Zaradi tega je smiselno v primeru polinomskih modelov določati tudi njihovo strukturo. To vodi do zmanjšane strukture polinomskega modela, v kateri pa potem lahko z metodo ortogonalnih najmanjših kvadratov določimo parametre modela.

Naslednja pomanjkljivost modela je tendenca k oscilatornemu interpolacijskemu obnašanju, posebno pri modelih višjih dimenzij, ki ga prikazuje slika 10.3b pri polinomu stopnje 20. Pri ekstrapolaciji je problem, da polinomi težijo proti $+\infty$ in $-\infty$ z velikimi vrednostmi odvodov. Na sliki 10.3a in sliki 10.3b vidimo problem ekstrapolacije, ki je večji v primeru modela višjega reda. Ekstrapolacija je determinirana v večini s členom najvišje stopnje. Ekstrapolacija je lahko tudi nemonotona in je neodvisna od odvoda na meji učnega področja. Čeprav so interpolacijske in ekstrapolacijske lastnosti polinomov slabe, jih včasih s pridom uporabimo, predvsem takrat, ko z njimi lahko zapišemo pravo strukturo procesa.

Slabosti polinomov vodijo v razvoj *zlepkov* (ang. *splines*), ki so lokalno definirani polinomi, in *ortonormalnih polinomov*.

Lastnosti polinomov lahko združimo v naslednje:

- *Interpolacija* je nemonotona in oscilatorna za polinome višje stopnje.
- *Ekstrapolacija* ima visoko tendenco k $+\infty$ ali $-\infty$ glede na člen z najvišjo potenco.
- *Lokalnost* ne obstaja. Polinomski model je popolnoma globalen.



Slika 10.3. a) Polinomski model stopnje 5 pri aproksimaciji sinusa. b) Polinom stopnje 20 pri aproksimaciji sinusa. Na sliki je 100 osumljenih podatkov označenih s “ \times ”. Prekinjena črta označuje interval $[0, 2\pi]$ v katerem so bili učni podatki.

- *Natančnost* je omejena, ker so višji redi polinomov praktično neprimerni.
- *Gladkost* je neizrazita. Gradient pogosto menja predznak.
- *Občutljivost na šum* je nizka, saj za učenje lahko uporabimo vse podatke.
- *Optimizacija parametrov* je enostavna, saj lahko uporabimo metodo najmanjših kvadratov. Vendar pa število parametrov hitro raste s povečano stopnjo polinoma.
- *Struktturna optimizacija* je učinkovita z uporabo tehnike izbire strukture po ortogonalnih najmanjših kvadratih. Vendar je lahko zelo počasna zaradi velikega števila regresorjev.
- *Sprotna adaptacija* je izvedljiva z metodo rekurzivnih najmanjših kvadratov, vendar je zelo nezanesljiva, saj imajo spremembe parametrov v eni delovni točki vpliv na vsa področja.
- *Hitrost učenja* je velika. Vendar pa upočasnjena, če upoštevamo še strukturno optimizacijo.
- *Hitrost izračuna modela* je srednja.
- *Problem dimenzij* je visok, saj število parametrov hitro narašča z dimenzijo problema.
- *Interpretacija* je komaj možna, razen v primerih, ko struktura dejansko opisuje fizikalne enačbe procesa in imajo parametri modela fizikalni pomen.
- *Vključevanje omejitev* izhoda in parametrov je možno, če za optimizacijo parametrov uporabimo optimizacijsko tehniko.
- *Vključevanje apriornega znanja* je skoraj nemogoče, razen v primeru posnemanja fizikalnega procesa.
- *Uporaba* je zelo razširjena.

10.3 Povzetek

Lastnosti linearnih in polinomskega modelov so prikazane v tabeli 10.1. Linearni model izkazuje prednosti v skoraj vseh kriterijih. Seveda pa lahko dosežemo določene prednosti z uporabo nelinearnih modelov. Prednosti linearnih modelov zelo dobro izkoriščajo pristopi z mehkimi modeli, kjer gre za sestavljanje več linearnih modelov med seboj.

Polinomi so klasični modeli za aproksimacijo. Imajo pa zelo pomembne pomanjkljivosti. Uporaba je smiselna takrat, ko je tudi struktura fizičnega modela približno polinomska. Glavna pomanjkljivost polinomske aproksimacije je v oscilatornem interpolacijskem in ekstrapolacijskem obnašanju.

Tabela 10.1. Primerjava lastnosti linearnih in polinomskeih modelov.

Lastnosti	Linearni model	Polinomski model
Interpolacija	+	-
Ekstrapolacija	+	--
Lokalnost	--	--
Natančnost	--	0
Gladkost	++	-
Občutljivost na šum	++	+
Optimizacija parametrov	++	++
Strukturna optimizacija	++	0
Sprotna adaptacija	+	-
Hitrost učenja	++	+
Hitrost izračuna modela	++	0
Problem dimenzij	++	-
Interpretacija	+	0
Vključevanje omejitev	0	-
Vključevanja apriornega znanja	0	-
Uporaba	++	+

++ / -- = zelo dobro / zelo slabo.

11. Umetne nevronske mreže

V tem poglavju bodpo predstavljene umetne nevronske mreže (ali nevronske mreže) in njihove aproksimacijske lastnosti. Najpomembnejše lastnosti nevronskih mrež so naslednje:

- veliko število enostavnih enot,
- močna paralelna arhitektura,
- močno povezane enote,
- robustnost glede na napake enot,
- učenje na osnovi podatkov.

Tabela 11.1 primerja terminologijo nevronskih mrež s terminologijo klasične identifikacije sistemov [104].

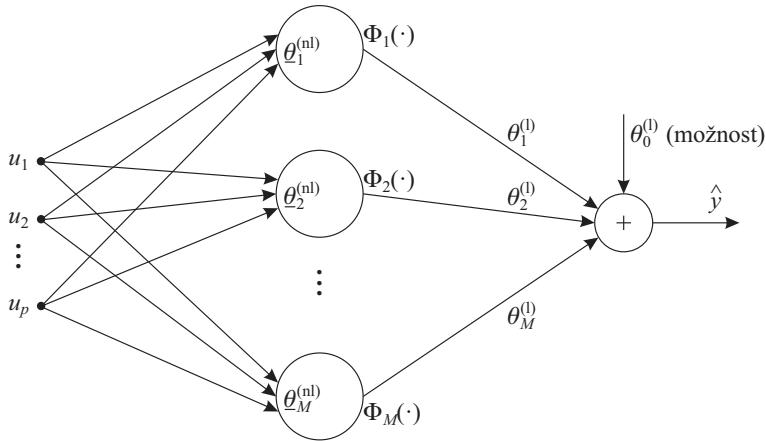
S terminologijo nevronskih mrež lahko mrežo na sliki 11.1 opišemo na naslednji način. Vozlišče na izhodu imenujemo *izhodni nevron*, vsi izhodni nevroni skupaj se imenujejo *izhodni nivo*. V našem primeru imamo samo en izhodni nevron, torej izhodni nivo vsebuje samo en nevron. Vsako od M vozlišč v sredini, ki realizirajo bazne funkcije, so *nevroni na skritem nivoju* in vsi ti nevroni skupaj tvorijo *skriti nivo*. Na koncu povejmo še, da se vhodi včasih imenujejo tudi *vhodni nevroni* in vsi skupaj tvorijo *vhodni nivo ali plast*. Taki mreži bom rekli mreža z enim skritim nivojem. Obstajajo tudi mreže, ki jih ne moremo predstaviti s shemo na sliki 11.1.

V primeru nevronske mreže se linearni parametri, ki so povezani z izhodnim nevronom imenujejo *izhodne uteži*

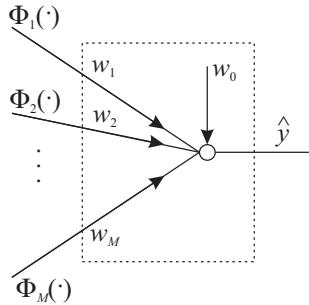
$$\theta_i^{(l)} = w_i . \quad (11.1)$$

Tabela 11.1. Treminologija nevronskih mrež in klasične identifikacije sistemov.

Nevronske mreže	Identifikacija sistemov in statistika
Preslikava ali aproksimacija	Regresija
Nevronska mreža	Model
Nevron	Bazna funkcija
Utež	Parameter
Prag (bias)	Odmik (offset)
Vhodna plast	Vektor vhodov
Napaka	Residual
Učenje ali trening	Estimacija, optimizacija ali adaptacija
Generalizacija	Interpolacija ali ekstrapolacija
Preparametrizacija ali pretreniranost	Visoka variančna napaka
Podparametrizacija ali podtreniranje	Visoka napaka pristranskosti
Sprotno učenje	Sprotna adaptacija



Slika 11.1. Nevronska mreža je mreža baznih funkcij, kjer so vse bazne funkcije enakega tipa.



Slika 11.2. Izhodni nevron v večini primerov mrež izvede linearno kombinacijo skritih nevronov.

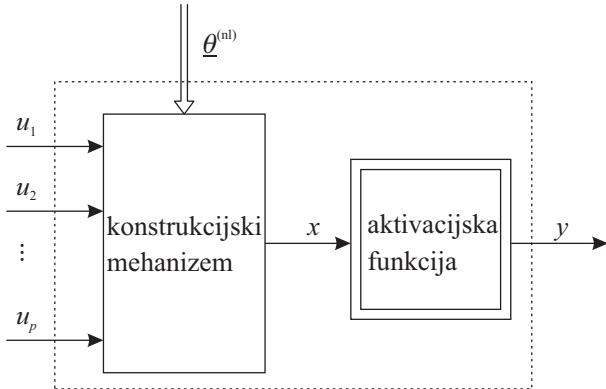
Izhodni nevron je običajno samo linearna kombinacija nevronov skritega nivoja z dodano utežjo w_0 , ki jo imenujemo prag; glej sliko 11.2. Vsi izhodi nevronov na skritem nivoju so uteženi z ustreznimi utežmi.

Najbolj razširjena arhitektura nevronskega mrež je večnivojski perceptron, ki je predstavljen v nadaljevanju v poglavju 11.2. Najprej bodo predstavljeni različni načini konstrukcije nevronskega mrež.

11.1 Konstrukcija nevronske mreže

Navadno se srečujemo s problemi, ki imajo več vhodnih spremenljivk. To zahteva, da so bazne funkcije $\Phi_i(\cdot)$ večdimenzijske in je njihova dimenzija enaka dimenziji vhodnega vektorja $p = \dim\{\underline{u}\}$. Za vse nevronske mreže in tudi za mnoge druge arhitekture modelov pa večdimenzijske bazne funkcije sestavimo iz enostavnih enodimenzijskih funkcij, ki jih na področju nevronskega mrež imenujemo *aktivacijske funkcije*. Slika 11.3 prikazuje konstrukcijo mreže, kjer vidimo, da zaradi več vhodov in enodimenzijsne aktivacijske funkcije potrebujemo konstrukcijski mehanizem, da vhode sestavimo. Pripomnimo naj, da je aktivacijska funkcija, ki preslikava skalar x v izhod y , zapisana z $g(\cdot)$. Nasprotno pa bazna funkcija $\Phi(\cdot)$ označuje večdimenzijsno preslikavo med vhodi nevrona in izhoda.

Obravnavali bomo dva najpomembnejša konstrukcijska mehanizma: princip projekcijske konstrukcije in radialni princip.



Slika 11.3. Z operacijo konstrukcijskega mehanizma preslikamo vhodni vektor \underline{u} v skalarni vhod x z uporabo nelinearnih aktivacijskih funkcij in njihovih parametrov. Nelinearna aktivacijska funkcija $g(x)$ preslikava skalar x v izhod y . Sivi blok realizira eno bazno funkcijo $\Phi(\cdot)$.

11.1.1 Projekcijska konstrukcija

Princip projekcijske konstrukcije je dobil ime po skalarnem produktu med vhodnim vektorjem in vektorjem uteži. Skalarni produkt pa geometrijsko interpretiramo kot projekcijo vektorja vhoda na vektor uteži. Ta princip konstrukcije je uporabljen pri mreži večnivojskega preceptronja (MLP); glej poglavje 11.2. Aktivacijska funkcija deluje na skalarju x , ki ga dobimo s projekcijo vhodnega vektorja v nelinearni prostor preko nelinearnih koeficientov; glej sliko 11.3. To lahko izvedemo s skalarnim produktom ali projekcijo vektorja

$$x = \underline{\theta}^{(nl)} \tilde{\underline{u}} = \theta_0^{(nl)} u_0 + \theta_1^{(nl)} u_1 + \dots + \theta_p^{(nl)} u_p , \quad (11.2)$$

kjer je $\tilde{\underline{u}}$ razširjeni vhodni vektor

$$\tilde{\underline{u}} = \begin{bmatrix} 1 \\ \underline{u} \end{bmatrix} = \begin{bmatrix} 1 \\ u_1 \\ \vdots \\ u_p \end{bmatrix} . \quad (11.3)$$

Vektor $\tilde{\underline{u}}$ vsebuje vhod $u_0 = 1$ zato, da dodamo enosmerno komponento v enačbi 11.2.

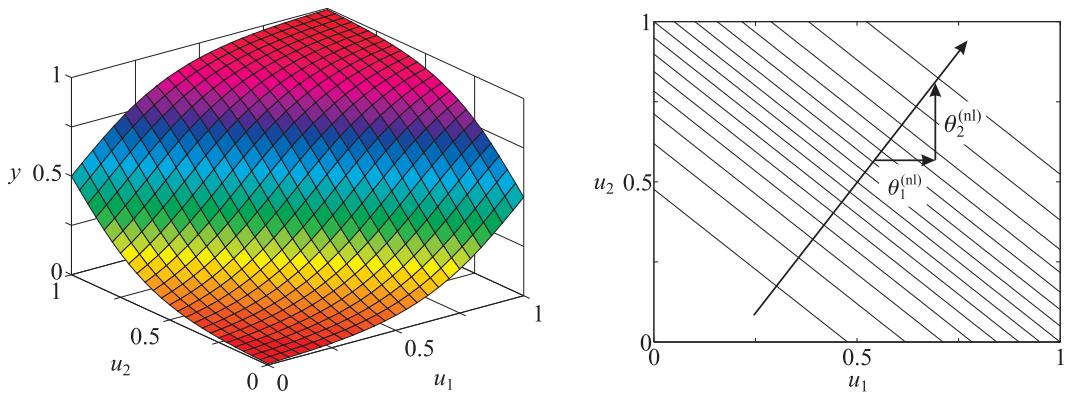
Tako konstruirana večdimenzijska aktivacijska funkcija lahko spreminja svojo vrednost le v smeri vektorjev nelinearnih parametrov posameznih enodimenzijskih aktivacijskih funkcij. V ortogonalnih smereh ostaja funkcija konstantna; glej sliko 11.4. Ta princip konstrukcije omogoča prilagoditev na probleme visokih dimenzij (vhodnih spremenljivk).

11.1.2 Radialna konstrukcija

Radialna konstrukcija mreže je uporabljena pri Gaussovih nevronskeih mrežah. Skalar x izračunamo kot razliko med vhodnim vektorjem in središčem aktivacijske funkcije:

$$\begin{aligned} x &= \|\underline{u} - \underline{c}\| = \sqrt{(\underline{u} - \underline{c})^T (\underline{u} - \underline{c})} \\ &= \sqrt{(u_1 - c_1)^2 + (u_2 - c_2)^2 + \dots + (u_p - c_p)^2} , \end{aligned} \quad (11.4)$$

kjer so središča aktivacijskih funkcij $\underline{c} = [c_1 \ c_2 \ \dots \ c_p]^T$ nelinearni parametri v nevronske mreži. Slika 11.5 prikazuje eno- in dvodimensionalni primer Gaussove aktivacijske funkcije.



Slika 11.4. V primeru konstrukcije s projekcijo se vrednost aktivacijskih funkcij spreminja samo v smeri vektorja parametrov (levo). Konturni diagram aktivacijske funkcije prikazuje (desno) kaže smer nelinearnosti. To smer in velikost definirajo nelinearni parametri v povezavi z vhodnim vektorjem.

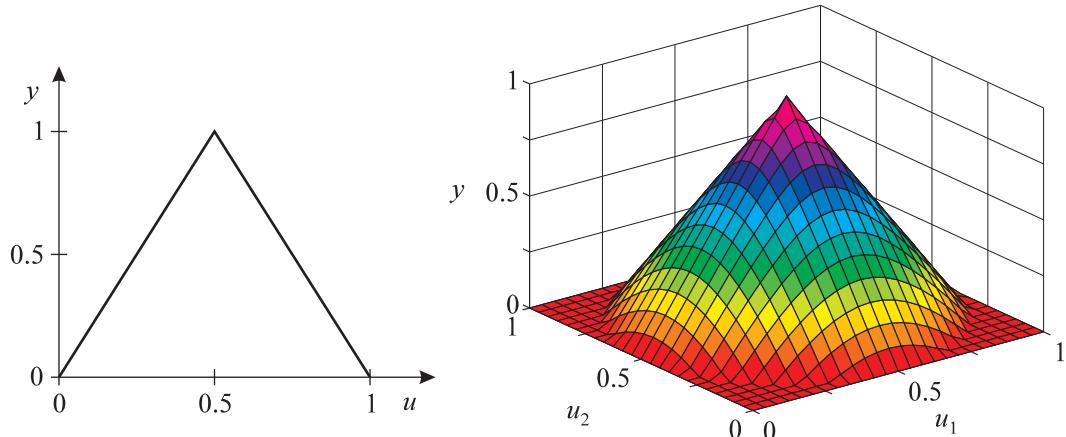
Če namesto Evklidove norme v enačbi 11.4 uporabimo Mahalanobisovo normo, se razdalja transformira glede na matrično normo $\underline{\Sigma}^{-1}$ (glej poglavje 7.5)

$$x = \|\underline{u} - \underline{c}\|_{\underline{\Sigma}^{-1}} = \sqrt{(\underline{u} - \underline{c})^T \underline{\Sigma}^{-1} (\underline{u} - \underline{c})}, \quad (11.5)$$

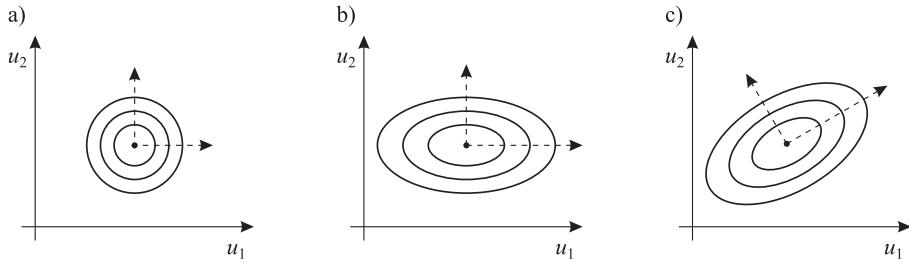
kjer so potem parametri maticne norme $\underline{\Sigma}^{-1}$ dodatni nelinearni parametri aktivacijske funkcije. Matrična norma $\underline{\Sigma}^{-1}$ skalira in rotira osi; njen inverz $\underline{\Sigma}$ predstavlja kovariančno matriko, ki je karakterizirana z elipso na sliki 11.6c. Za poseben primer, kjer je matrika ($\underline{\Sigma}^{-1} = \mathcal{I}$), Mahalanobisova norma postane enaka Evklidovi normi. Za

$$\underline{\Sigma}^{-1} = \begin{bmatrix} 1/\sigma_1^2 & 0 & \cdots & 0 \\ 0 & 1/\sigma_2^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1/\sigma_p^2 \end{bmatrix}, \quad (11.6)$$

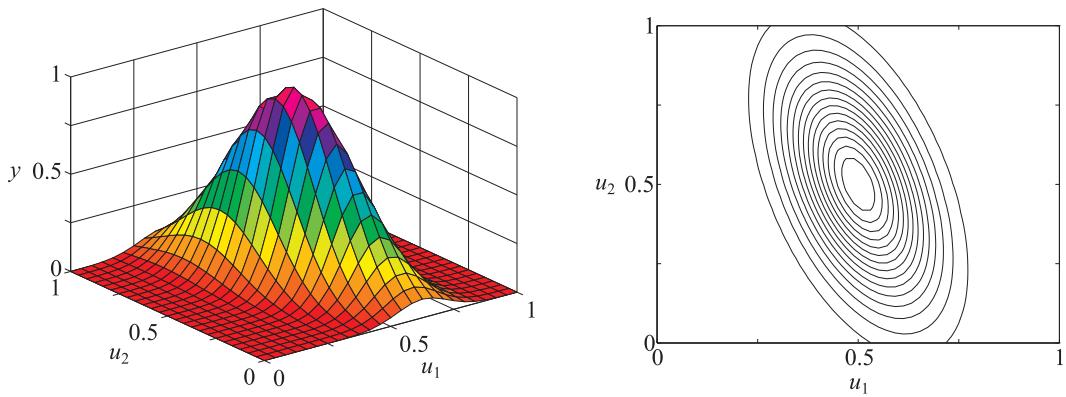
kjer je p dimenzija vhodnega prostora, je Mahalanobisova norma enaka Evklidovi normi s skaliranimi vhodnimi podatki $u_l^{(\text{norm})} = u_l / \sigma_l$. V najsplošnejšem primeru pa kovariančne matrične norme skalira in rotira osi. Slika 11.6 prikazuje različne norme razdalj. Razdalje za Evklidovo



Slika 11.5. Eno in dvodimensionalna Gaussova ali radialna trikotna aktivacijska funkcija.



Slika 11.6. Linije enakih razdalj za različne norme razdalj: a) Evklidovo ($\Sigma^{-1} = I$), b) diagonalno (Σ^{-1} = diagonalna), and c) Mahalanobisovo normo (Σ^{-1} = splošna, pozitivna simetrična matrika).



Slika 11.7. Gaussova radialna aktivacijska funkcija.

normo s transformiranimi vhodi so s konturnim diagramom prikazane na sliki 11.7. Funkcijo imenujemo radialna, čeprav ni radialna (ni niti simetrična) glede na vhodna originalna vektorja u_1 in u_2 , je pa radialna glede na njuna transforma.

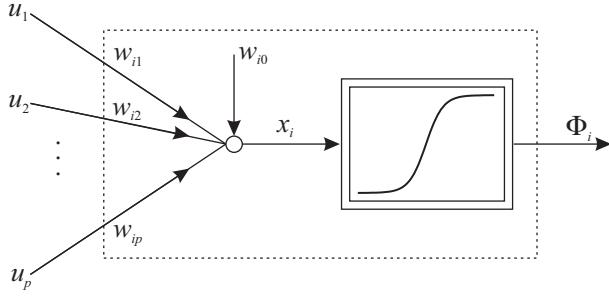
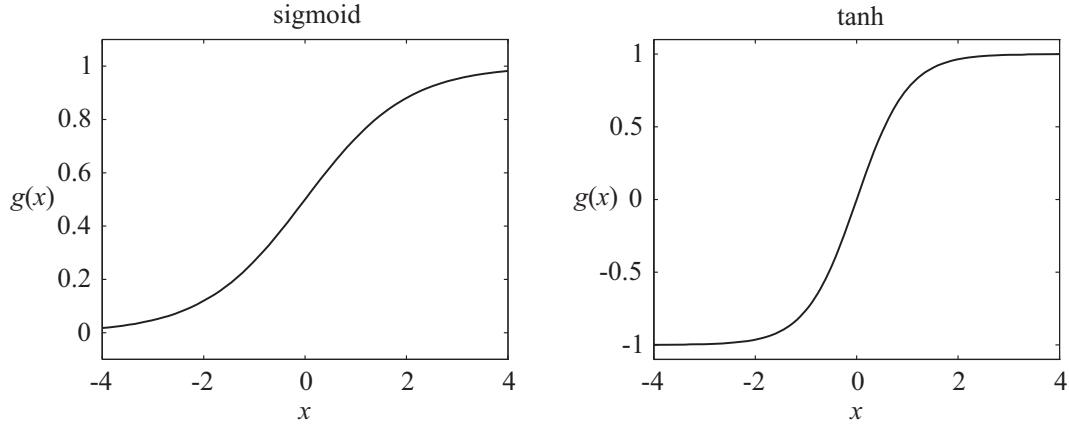
11.2 Večnivojski perceptron (MLP)

Večnivojski perceptron (*ang. multilayer perceptron*) je najbolj znana in uporabna arhitektura nevronske mreže. Dostikrat je tudi sinonim za nevronsko mrežo. Razlog za to je v zelo popularni knjigi avtorjev Rumelharta, Hintona in Williamsa [103]. V njej so pokazali, da je mogoče preseči funkcionalne omejitve enostavnega perceptrona z dodajanjem plasti v mrežo, [82].

V poglavju je najprej predstavljen večnivojski perceptron in njegovo delovanje. V drugem delu je predstavljena njegova struktura. V poglavju 11.2.3 je predstavljen učni algoritem mreže, metoda vzvratnega širjanja napake (*ang. backpropagation algorithm*). Druge metode učenja večnivojskega perceptrona so predstavljene v poglavju 11.2.4. V nadaljevanju govorimo o prednostih in slabostih večnivojskega perceptrona. V poglavju 11.2.7 pa je pregled nekaterih modifikacij standardne arhitekture mreže.

11.2.1 Nevron večnivojskega perceptrona

Slika 11.8 prikazuje skriti nevron v večnivojskem preceptronu. Funkcionalnost tega nevrona lahko razdelimo na dva dela. Najprej je uporabljena linearna konstrukcija za projekcijo vhodnega vektorja $\underline{u} = [u_1 \ u_2 \ \dots \ u_p]^T$ na vektor uteži (skalarni produkt, kot projekcija). V drugi fazi pa je preko nelinearne aktivacijske funkcije preslikamo vhod v rezultirajoči izhod, $g(x)$. Običajno je aktivacijska funkcija take oblike, da ima omejitev. Taki sta na primer sigmoidna funkcija

Slika 11.8. Perceptron: i -ti skriti nevron MLP.

Slika 11.9. Tipični aktivacijski funkciji perceptronov.

$$g(x) = \text{sigmoid}(x) = \frac{1}{1 + \exp(-x)}, \quad (11.7)$$

in hiperbolični tangens

$$g(x) = \tanh(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} = \frac{1 - \exp(-2x)}{1 + \exp(-2x)}. \quad (11.8)$$

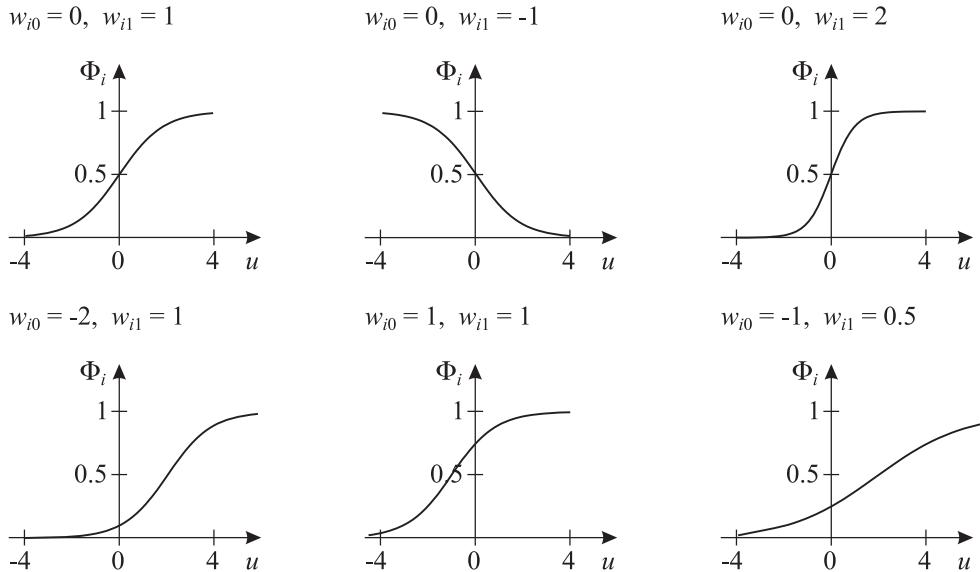
Obe funkciji sta prikazani na sliki 11.9. Med seboj ju lahko transformiramo na naslednji način:

$$\begin{aligned} \tanh(x) &= \frac{1 - \exp(-2x)}{1 + \exp(-2x)} = \frac{2}{1 + \exp(-2x)} + \frac{-1 - \exp(-2x)}{1 + \exp(-2x)} \\ &= 2 \text{sigmoid}(2x) - 1. \end{aligned} \quad (11.9)$$

Ti dve funkciji imata enako zanimivo lastnost, da v odvodu funkcije nastopa funkcija sama

$$\begin{aligned} \frac{d\Phi_i}{dx} &= \frac{d \text{sigmoid}(x)}{dx} = \frac{\exp(-x)}{(1 + \exp(-x))^2} = \frac{1 + \exp(-x) - 1}{(1 + \exp(-x))^2} \\ &= \frac{1}{1 + \exp(-x)} - \frac{1}{(1 + \exp(-x))^2} = \Phi_i - \Phi_i^2 = \Phi_i(1 - \Phi_i), \end{aligned} \quad (11.10)$$

$$\begin{aligned} \frac{d\Phi_i}{dx} &= \frac{d \tanh(x)}{dx} = \frac{1}{\cosh^2(x)} = \frac{\cosh^2(x) - \sinh^2(x)}{\cosh^2(x)} \\ &= 1 - \tanh^2(x) = 1 - \Phi_i^2. \end{aligned} \quad (11.11)$$



Slika 11.10. Učinek parametrov i -tega nevrona perceptronovega skritega nivoja z enim samim vhodom u . Medtem, ko utež w_{i0} definira položaj aktivacijske funkcije, utež w_{i1} definira strmino aktivacijske funkcije. Aktivacijske funkcije je v tem primeru enaka $\Phi_i(x) = 1/(1 + \exp(-w_{i1}u + w_{i0}))$.

To je zelo pomembna lastnost, saj te odvode nujno potrebujemo pri učenju mreže s katerokoli gradientno metodo optimizacije; glej poglavje 11.2.3.

Izhod perceptrona na sliki 11.8 je odvisen od nelinearnih parametrov v skritem nivoju. Te parametre imenujemo *uteži skritega nivoja*

$$\underline{\theta}_i^{(nl)} = [w_{i0} \ w_{i1} \ w_{i2} \ \dots \ w_{ip}]^T. \quad (11.12)$$

Utež w_{i0} ustvarja enosmerno komponento, ki ji rečemo tudi prag (*ang. bias, threshold*). Sliki 11.10 in 11.11 prikazuje kako te uteži definirajo obliko bazne funkcije.

11.2.2 Struktura mreže

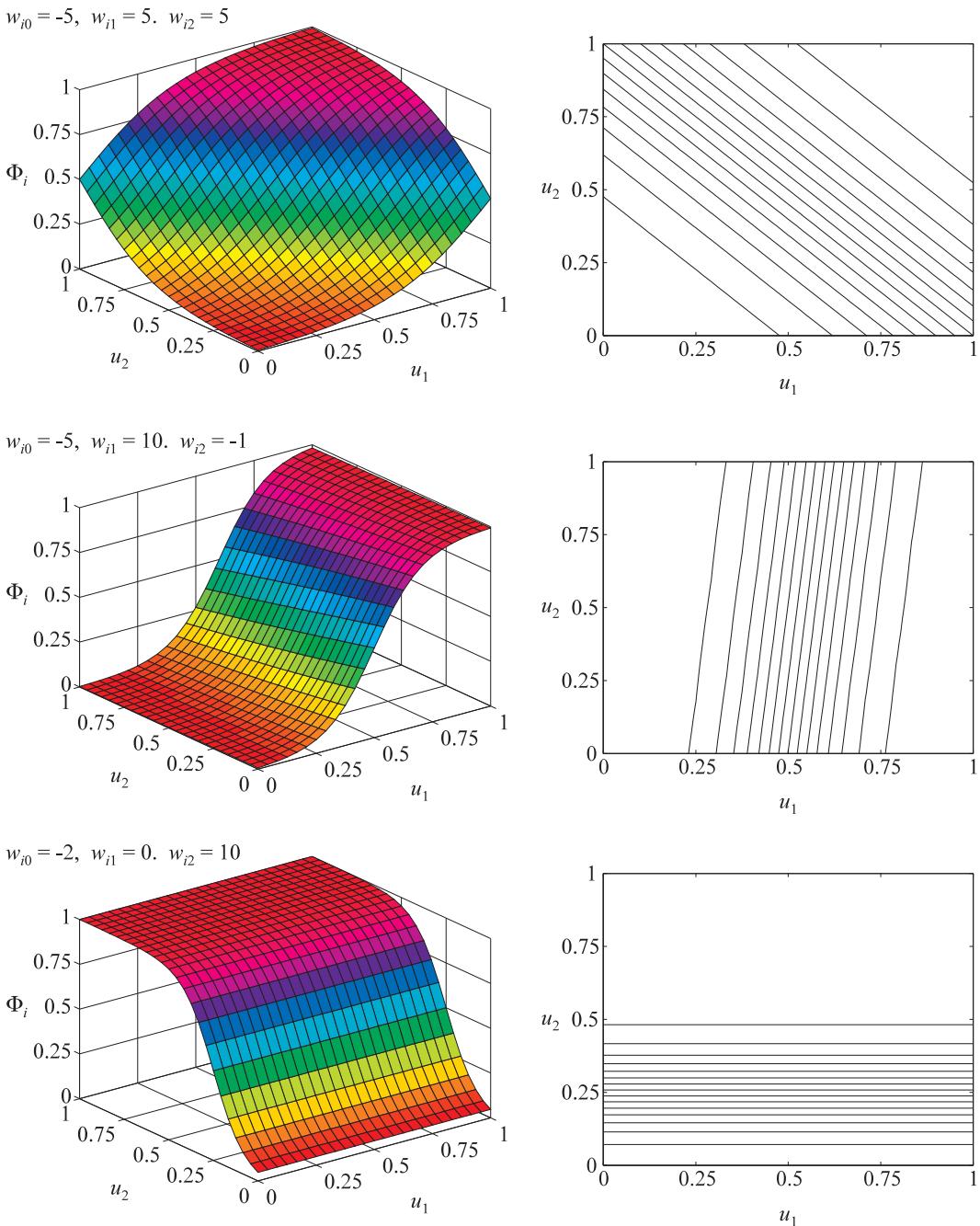
Če več nevronov paralelno združimo preko izhodnega nevrona, dobimo večnivojski perceptron z enim skritim nivojem; glej sliko 11.12. V zapisu z baznimi funkcijami to lahko zapišemo kot

$$\hat{y} = \sum_{i=0}^M w_i \Phi_i \left(\sum_{j=0}^p w_{ij} u_j \right) \text{ za } \Phi_0(\cdot) = 1 \text{ in } u_0 = 1, \quad (11.13)$$

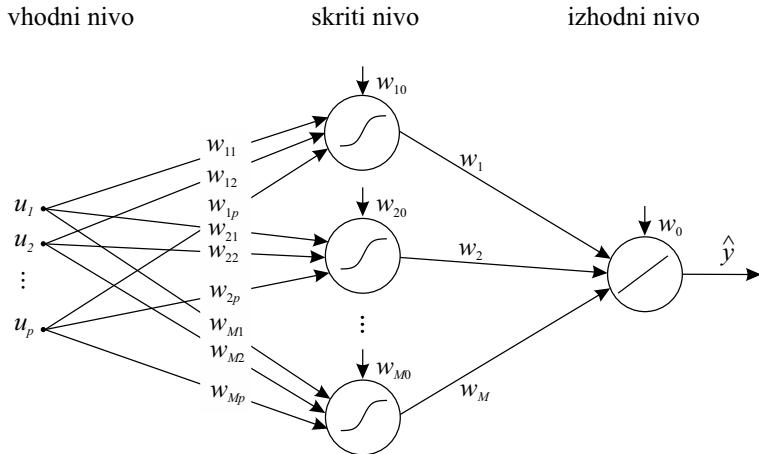
kjer so w_i uteži izhodnega nivoja in w_{ij} uteži skritega nivoja.

Potrebno je poudariti, da je aktivacijska funkcija Φ_i odvisna od argumenta, ki je linearna kombinacija p vhodov, enega dodatnega vhoda zaradi delovne točke in nelinearnih uteži w_{ij} , $j = 0, \dots, p$. To pomeni, da je izhod aktivacijske funkcije enak

$$\Phi_i \left(\sum_{j=0}^p w_{ij} u_j \right).$$



Slika 11.11. Učinek parametrov i -tega nevrona perceptronovega skritega nivoja z dvema vhodoma u_1 in u_2 . Medtem, ko utež w_{i0} definira razdaljo aktivacijske funkcije do koordinatnega izhodišča, utež w_{i1} in w_{i2} definirata strmino v smeri spremenljivke u_1 in u_2 . Vektor uteži $[w_{i1} \; w_{i2}]^T$ pa kaže v smeri nelinearnosti bazne funkcije. Pravokotno na $[w_{i1} \; w_{i2}]^T$ je aktivacijska funkcija konstantna.



Slika 11.12. Mreža večnivojskega perceptronja.

Skupno število parametrov MLP mreže je enako

$$M(p + 1) + M + 1 , \quad (11.14)$$

kjer je M število nevronov v skritem nivoju in p število vhodov. Ker je število vhodov odvisno od problema, ki ga rešujemo, je kompleksnost mreže odvisna od števila nevronov v skritem nivoju.

Večnivojski perceptron je *univerzalni aproksimator* [46]. To pomeni, da lahko perceptron aproksimira *poljubno* gladko funkcijo do poljubne natančnosti, če povečujemo število nevronov na skritem nivoju. Ta lastnost je zelo pomembna, saj upravičuje uporabo te mreže pri poljubnih problemih aproksimacije. Po drugi strani pa so tudi vsi ostali aproksimatorji, ki jih obravnavamo (polinomi, mehki modeli) univerzalni aproksimatorji.

Večnivojski perceptron konstuiramo kot kombinacijo enodimensionalnih funkcij (aktivacijskih funkcij) s projekcijsko metodo. Intuitivno je težko razložiti, da lahko poljubno gladko funkcijo aproksimiramo s poljubno natančnostjo z množico enodimensionalnih funkcij.

Večnivojski perceptron, ki smo ga opisali do sedaj je samo najbolj uporaben tip umeten nevronske mreže. Obstaja veliko različnih variant. V določenih modifikacijah mreže izhodni nevron ni samo linearne kombinacija, ampak je lahko celoten perceptron z dodano poljubno aktivacijsko funkcijo na izhodu. Na ta način izhodne uteži postanejo nelinearne in je zato učenje zahtevnejše. Pomembna modifikacija je v dodanem skritem nivoju, kar zelo poveča aproksimacijsko moč mreže; glej sliko 11.12. V principu ima lahko mreža poljubno število skritih nivojev. V uporabi so mreže, ki imajo do dva skrita nivoja.

Ce povzamemo, ima perceptron dva različna tipa parametrov:

- *Uteži izhodnega nivoja* so linearni parametri. Definirajo amplitude baznih funkcij in delovno točko.
- *Uteži skritega nivoja* so nelinearni parametri in definirajo smeri, strmine in položaje baznih funkcij.

Večnivojski perceptron učimo z optimizacijo teh uteži. Veliko učnih strategij je razvitih in obravnavanih; glej poglavje 11.2.4. Najprej je predstavljen algoritem vzvratnega učenja.

11.2.3 Učenje z algoritmom vzvratnega širjenja napake

Algoritem vzvratnega širjenja napake je dejansko le način izračuna gradientov izhoda perceptrona glede na uteži. Odvodi izhoda MLP mreže glede na i -to utež v izhodnem nivoju so enaki

$(i = 0, \dots, M)$

$$\frac{\partial \hat{y}}{\partial w_i} = \Phi_i \quad \text{in} \quad \Phi_0 = 1 . \quad (11.15)$$

Odvodi izhoda MLP mreže glede na uteži *skritega nivoja* pa so ($i = 1, \dots, M, j = 0, \dots, p$)

$$\frac{\partial \hat{y}}{\partial w_{ij}} = w_i \frac{dg(x)}{dx} u_j \quad \text{in} \quad u_0 = 1 \quad (11.16)$$

za uteži na povezavi med j -tim vhodom in i -tim skritim nevronom. V enačbi 11.16 realiziramo enačbo 11.10 ali enačbo 11.11, ki definirata odvode aktivacijske funkcije $g(x)$.

Na primer, če je $g(x) = \tanh(x)$, potem enačba 11.16 postane enaka

$$\frac{\partial \hat{y}}{\partial w_{ij}} = w_i (1 - \Phi_i^2) u_j \quad \text{in} \quad u_0 = 1 . \quad (11.17)$$

Ker lahko gornji zapis za odvod izhoda glede na uteži vzvratno razširjamo skozi mrežo, se algoritmom imenuje algoritmom vzvratnega učenja. To postane še bolj očitno, če imamo več skritih nivojev v mreži; glej poglavje 11.2.7.

11.2.4 Učenje perceptronova

Navadno učimo perceptron z uporabo *nelinearne optimizacije*, kjer vse uteži optimiramo simultano z lokalno ali globalno nelinearno optimizacijo. Metoda zahteva dobro inicializacijo uteži. Zato bomo probleme, povezane z inicializacijo, obravnavali najprej.

Incializacija uteži. Uteži nevronske mreže se zelo težko interpretirajo tudi ob dobrem poznavanju fizikalnega procesa. Zaradi tega jih praktično ne moremo smiselnno inicializirati. Hkrati pa je inicializacija tudi zelo odvisna od tipa problema, ki ga rešujemo.

Najbolj enostavna je naključna inicializacija, ki pa lahko vodi do tega, da je veliko nevronov v nasičenju zaradi izbire prevelikih uteži. Nevroni v nasičenju dajo enako izhodno veličino, ne glede na velikost vhoda. Na ta način ne vnašajo variabilnosti v model in so nesmiselni. Nevroni v nasičenju ne izboljšajo aproksimacijskih sposobnosti nevronske mreže, hkrati pa na ta način postane mreža slabo pogojena. Mreža v nasičenju je težko premakniti iz tega stanja, ker je njen izhodni pogrešek neobčutljiv na spremembe uteži. Taki inicializacije se je treba izogniti.

Nasičenju se lahko izognemo tako, da vse nevrone $i = 1, \dots, M$ inicializiramo v linearjem področju aktivacijske funkcije. To pomeni, da moramo inicializirati uteži tako, da je $x_i = 0$. Če gre $x_i \rightarrow -\infty$ ali $x_i \rightarrow \infty$, potem to vodi v nasičenje aktivacijske funkcije. Uteži moramo zaradi tega nastaviti na zelo majhne vrednosti (npr. na intervalu $[-5, 5]$) [32, 74]. Seveda pa morajo biti vsi vhodi skalirani, sicer lahko eden od vhodov, ki je višjih vrednosti, dominira nad ostalimi. V takem primeru bi bila mreža odvisna predvsem od vhoda velikih vrednosti, npr. u_j . Z normiranjem se izognemo temu pojavi. To lahko rešimo tudi z izbiro uteži, ki morajo odražati amplitudna razmerja med vhodi. To pomeni, da morajo biti vsi produkti $w_{ij}u_j$ v enakem velikostnem razredu.

Nelinearna optimizacija večnivojskega perceptronova. Najpogostejši način optimizacije MLP mreže je uporaba nelinearne lokalne optimizacije za določevanje uteži. V začetku so MLP učili z gradientno metodo, kot na primer

$$\underline{\theta}_k = \underline{\theta}_{k-1} - \eta \frac{\partial I_k}{\partial \underline{\theta}_{k-1}} , \quad (11.18)$$

kjer je

$$\underline{\theta} = [w_0 \ w_1 \ \cdots \ w_M \ w_{10} \ w_{11} \ \cdots \ w_{1p} \ w_{20} \ w_{21} \ \cdots \ w_{Mp}]^T. \quad (11.19)$$

Vektor uteži vsebuje vse uteži mreže, η je konstanten korak učenja, ki ga imenujemo konstanta učenja in I je kriterijska funkcija, ki jo optimiramo. Za kriterijsko funkcijo I navadno izberemo vsoto kvadratov napake:

$$I = \sum_{i=1}^N e_i^2 = \sum_{i=1}^N (y_i - \hat{y}_i)^2, \quad (11.20)$$

kar ustreza adaptaciji na celotni množici podatkov (*ang. batch adaptation*), ali pa

$$I_k = e_k^2 = (y_k - \hat{y}_k)^2, \quad (11.21)$$

kar ustreza sprotni adaptaciji. Če uporabimo kakšno od gradientnih metod za učenje mreže, moramo z metodo vzvratnega učenja določiti odvode kriterijske funkcije oziroma izhoda modela glede na uteži (*ang. backpropagation*).

Adaptacija uteži v enačbi 11.18 je zelo enostavna. Problem pa je konvergenca adaptacije, ki je zelo počasna. Konstantni faktor učenja η je težko dobro izbrati. Če je prevelik to lahko vodi v oscilacije ali divergenco, premajhen pa v prepočasno konvergenco. Pogosto pa učenje rezultira v efektu "žaganja"; glej poglavje 5.4.2. Obstaja veliko izboljšav in modifikacij učenja po enačbi 11.18), npr. metoda z adaptivnim prilagajanjem faktorja učenja ali individualni faktor učenja za posamezne uteži.

Najnovejše metode na področju učenja perceptrona mreže so Levenberg-Marquardtovi nelinearni najmanjši kvadrati (poglavlje 5.5.2) [38] ali Kvazi-Newtonova metoda optimizacije (poglavlje 5.4.4) za majhne in srednje velike mreže in reducirana Levenberg-Marquardtova, Kvazi-Newtonova metoda ali metoda konjugiranih gradientov (poglavlje 5.4.5) za velike mreže (po številu uteži).

Globalne metode optimizacije se redkeje uporabljajo za optimizacijo večnivojskega perceptrona, ker je konvergenca običajno počasna in lokalni optimumi niso tako resen problem. Največkrat globalni optimum niti ni končni cilj, ker se zadovoljimo že z lokalnim ciljem, ki zadovolji določen tolerančni prag.

11.2.5 Simulacijski primer

V nadaljevanju si bomo ogledali enostavno delovanje večnivojskega perceptrona, ki ima sigmoidne aktivacijske funkcije v skritem nivoju in linearne aktivacijske funkcije v izhodnem nivoju. Uporabili smo enostavno učenje (*ang. backpropagation*). Vsi parametri so bili inicializirani načelno na intervalu $[-1, 1]$, kar preprečuje nasičenje aktivacijskih funkcij, ker vhod leži med 0 in 1.

Na sliki 11.13a in b je aproksimacija funkcije z večnivojskim perceptronom, ki ima en nevron na skritem nivoju. Tudi na tako preprostem primeru pa lahko vidimo, da imamo dva enakovredna optimuma. Medtem, ko sta na sliki 11.13a parametra izhodnega nevrone negativna in enaka $w_{01} = -4.4$ in $w_{11} = -6.4$, sta parametra nevrone na skritem nivoju pozitivna in enaka $w_0 = 1.2$ in $w_1 = 635$. Rešitev na sliki 11.13b pa je ravno obratna ($w_{01} = 4.2, w_{11} = 6.4, w_0 = 552, w_1 = -551$). Hkrati ugotovimo zelo slabo občutljivost na določene uteži. Z drugimi besedami to pomeni, da ima kriterijska funkcija zelo ravno dno in zelo velike spremembe parametrov zelo malo vplivajo na spremembo kriterijske funkcije. Velike spremembe se zgodijo, če uporabimo različne metode optimizacije. Medtem, ko metode drugega reda konvergirajo dobro, metode prvega reda konvergirajo v določeno veliko območje okoli

minimuma. Zaradi tega lahko parametri uteži zelo varirajo glede na začetne pogoje in optimizacijsko tehniko, pri tem, da imajo primerljivo natančnost. Ta enostaven primer nam kaže, kako ne morem dati kakšnega posebnega pomena posameznim vrednostim parametrov.

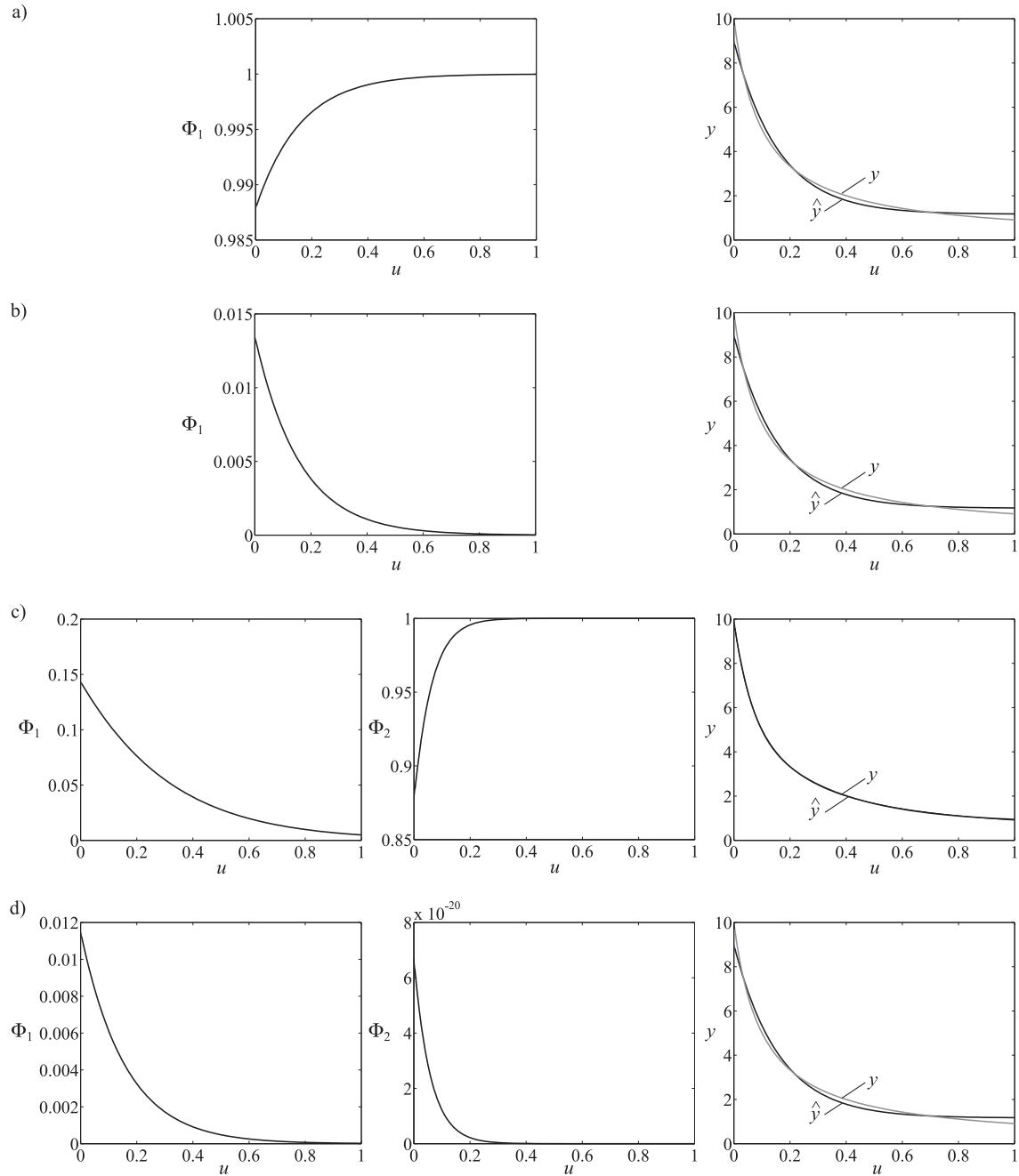
Sliki 11.13c–d prikazujeta aproksimacijo z MLP mrežo z dvema skritima nevronoma. Na sliki 11.13c je odlična aproksimacijska natančnost dosežena, kjer je prva aktivacijska funkcija (Φ_1) optimirana za opis desnega dela funkcije, druga aktivacijska funkcija (Φ_2), ki je multiplicirana z negativno izhodno utežjo, pa opisuje levi del nelinearne funkcije. Glede na različne začetne izbire pa lahko mreža konvergira tudi v rezultat na sliki 11.13d. Druga aktivacijska funkcija je skonvergirala proti nič in je globoko v nasičenju. Seveda rezultat na sliki 11.13d ne predstavlja lokalnega optimuma. Če bi zaganjali optimizacijo dalj časa, bi mogoče lahko prišli do rezultata na sliki 11.13c. V tem primeru je kriterijska funkcija tako neobčutljiva na spremembe v nasičenem nevronu, da izgleda kot, da je optimizacija doseglila lokalni optimum.

Aproksimacijski problem na sliki 11.14 bi potreboval več nevronov za dosego primerne natančnosti kot primer na sliki 11.13. Presenetljivo pa slika 11.14a govori nasprotno. Ta primer jasno kaže na prednosti globalne aproksimacije za nastavitev uteži na skritem nivoju. Vsi poiskusi pri učenju niso tako uspešni. Skoraj vedno se dogaja, da določeni poizkusni rezultirajo v neuspešnih rezultatih, kar kažejo slike 11.14b–d. Slika 11.14c prikazuje lokalni optimum, sliki 11.14b in d pa sta podobni sliki 11.13d. Primeri kažejo na slabosti in prednosti MLP. Na eni strani so mreže zelo fleksibilne, kar omogoča generiranje širokega nabora različnih oblik baznih funkcij, ki so primerne za določene probleme. Na druge strani pa seveda obstaja problem konvergencije v lokalni optimum in problem nasičenja nevronov, kar zahteva veliko poizkusov. To je seveda problematično, ko postanejo problemi bolj kompleksni. Takrat je tudi razumevanje delovanja mreže vprašljivo.

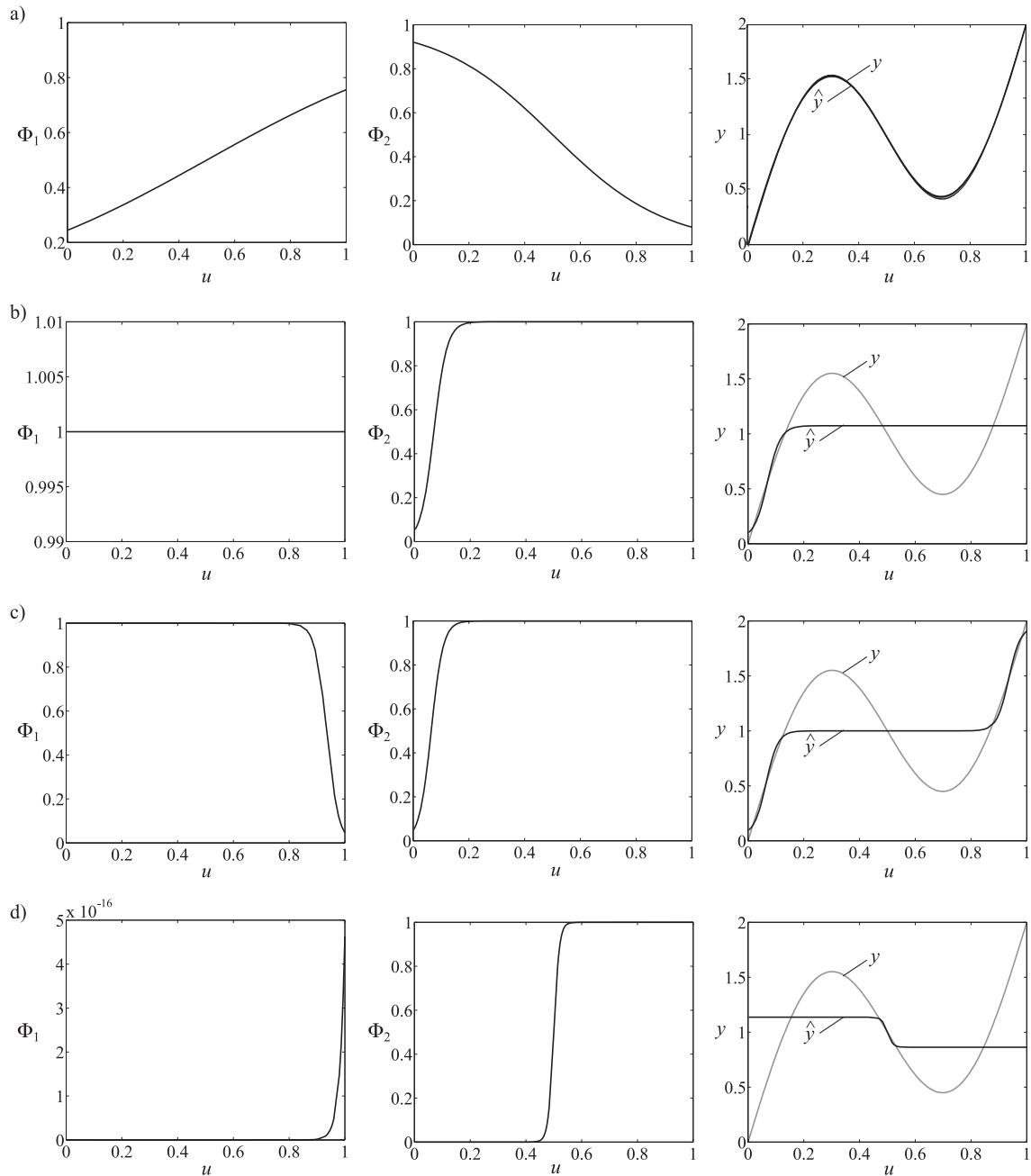
11.2.6 Lastnosti večnivojskega perceptronja

Najpomembnejše lastnosti večnivojskega perceptronja so naslednje:

- *Interpolacija* je glede na obliko sigmoidne aktivacijske funkcije monotona.
- *Ekstrapolacija* je konstantna v širokem področju glede na lastnost nasičenja sigmoidne funkcije. Lahko pa je ekstrapolacija tudi linear, v primeru, če je strmina sigmoidne funkcije majhna.
- *Lokalnost* ne obstaja, ker sprememba v eni sami uteži vpliva na izhod modela za širok interval vhodnega prostora. Ker pa aktivacijske funkcije niso strogo globalne, ima MLP aproksimacijsko fleksibilnost samo v delu, kjer aktivacijska funkcija ni v nasičenju. Zato ima aproksimacijski mehanizem določeno lokalnost.
- *Natančnost* je zelo visoka. Na račun optimizacije uteži v skritem nivoju je MLP mreža zelo učinkovit aproksimator in običajno potrebuje manj parametrov kot druge strukture za dosego enake natančnosti. To lahko interpretiramo tudi kot lastnost učinkovitega stiskanja informacije.
- *Gladkost* je visoka.
- *Občutljivost na šum* je nizka, saj za učenje lahko uporabimo vse podatke.
- *Optimizacijo parametrov* moramo izvesti z nelinearno optimizacijo in je zato počasna.
- *Struktturna optimizacija* zahteva računsko potratne metode dodajanja in odvzemanja nevronov (ang. *growing and pruning*).
- *Sprotna adaptacija* je zelo nezanesljiva, saj ima majhna sprememba parametra v eni delovni točki vpliv na vsa področja.
- *Hitrost učenja* je zelo majhna, ker moramo uporabiti nelinearne optimizacijske tehnike in jih ponoviti večkrat zaradi lokalnih minimumov.



Slika 11.13. Aproksimacija funkcije $y = 1/(u + 0.1)$ z MLP z enim skritim nevronom (zgornje slike) in dvema skritima nevronoma (spodnje slike). Dva mogoča rezultata učenja glede na inicializacijo parametrov, za primer z enim in dvema nevronoma sta prikazana na slikah a, b in c, d .



Slika 11.14. Aproksimacija funkcije $y = \sin(2\pi u) + 2u$ z MLP mrežo z dvema nevronoma v skriti plasti. Štirje različni rezultati treninga v odvisnosti od začetnih pogojev so na slikah a–d .

- Hitrost izračuna modela je velika zaradi relativno majhnega števila nevronov.
- Problem dimenzij je zelo majhen zaradi konstrukcije s projekcijo. MLP je najbolj primerna umetna nevronska mreža za probleme visokih dimenzij.
- Interpretacija je komaj možna.
- Vključevanje omejitev je zahtevno, ker je interpretacija slaba.
- Vključevanje apriornega znanja je skoraj nemogoče.
- Uporaba je zelo razširjena. MLP je standardna nevronska mreža.

11.2.7 Mreže z več skritimi nivoji

Večplastni perceptron na sliki 11.12 lahko razšrimo z dodatnim skritim nivojem. To seveda poveča kompleksnost strukture. Kaj je bolj učinkovito, mreža z manjšim številom skritih nevronov in večjim številom nivojev ali mreža, ki ima enako število parametrov, a samo en skriti nivo, je odvisno od primera. Seveda pa je učenje v primeru večjega števila skritih nivojev bistveno bolj zahtevno, ker so parametri bolj nelinearni.

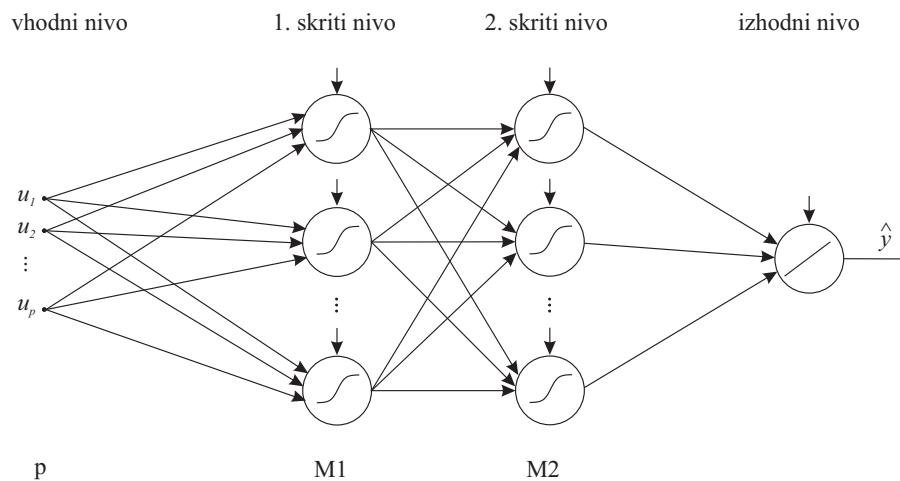
Največkrat se uporablja MLP z dvema skritima nivojema. Slika 11.15 predstavlja primer MLP z dvema skritima nivojema. Predstavitev baznih funkcij je bolj kompleksna, saj so nevroni na drugem nivoju odvisni od izhodov nevronov na prvem nivoju. Če sta M_1 in M_2 števili nevronov na prvem in drugem skritem nivoju, w_i uteži na izhodnem nivoju, $w_{jl}^{(1)}$ in $w_{ij}^{(2)}$ pa uteži prvega in drugega skritega nivoja, potem izhod mreže z baznimi funkcijami zapišemo kot

$$\hat{y} = \sum_{i=0}^{M_2} w_i \Phi_i \left(\sum_{j=0}^{M_1} w_{ij}^{(2)} \xi_j \right) \quad \text{in} \quad \Phi_0(\cdot) = 1 , \quad (11.22)$$

in z izhodi nevronov na prvem nivoju

$$\xi_j = \Psi_j \left(\sum_{l=0}^p w_{jl}^{(1)} u_l \right) \quad \text{pri} \quad \Psi_0(\cdot) = 1 \quad \text{in} \quad u_0 = 1 . \quad (11.23)$$

Navadno so aktivacijske funkcije obeh skritih nivojev, Φ_i za prvi nivo in Ψ_j za drugi nivo, saturacijskega tipa.



Slika 11.15. Večnivojski perceptron z dvema skritima nivojema (uteži niso zapisane zaradi bolj transparentne predstavitev).

Odvod izhoda mreže glede na izhodne uteži je enak ($i = 0, \dots, M$) in je neodvisen od strukture

$$\frac{\partial \hat{y}}{\partial w_i} = \Phi_i \quad \text{in} \quad \Phi_0 = 1 . \quad (11.24)$$

Odvod glede na uteži na drugem skritem nivoju pri izbiri $\tanh(\cdot)$ aktivacijske funkcije je enak ($i = 1, \dots, M_2, j = 0, \dots, M_1$):

$$\frac{\partial \hat{y}}{\partial w_{ij}^{(2)}} = w_i(1 - \Phi_i^2)\Psi_j \quad \text{pri} \quad \Psi_0 = 1 . \quad (11.25)$$

Odvod glede na uteži prvega skritega nivoja pri aktivacijski funkciji $\tanh(\cdot)$ ($j = 1, \dots, M_1, l = 0, \dots, p$) pa je enak

$$\frac{\partial \hat{y}}{\partial w_{jl}^{(1)}} = \sum_{i=0}^{M_2} w_i(1 - \Phi_i^2)w_{ij}^{(2)}(1 - \Psi_j^2)u_l \quad \text{in} \quad u_0 = 1 . \quad (11.26)$$

Pripomniti velja, da je vrednost aktivacijske funkcije Φ_i odvisna od vhodov in od uteži na skritem nivoju. Tega nismo zapisali zaradi enostavnosti zapisa.

Število uteži za MLP mrežo z dvema skritima nivojem je enako

$$M_1(p+1) + M_2(M_1+1) + M_2 + 1 . \quad (11.27)$$

Zaradi člena s produktom $M_1 M_2$ število uteži raste kvadratično s številom nevronov na skritem nivoju.

11.3 Povzetek

Tabela 11.2 prikazuje pregled prednosti in slabosti večnivojskih perceptronov, Gaussovih nevronskih mrež (RBF) in normiranih RBF mreže. Zadnjih dveh nismo natančno obravnavali in želimo s tabelo prikazati samo njune lastnosti. Najpomembnejša v tem poglavju je predstavitev konstrukcijskega mehanizma z metodo projekcije, ki omogoča rešitev problema velikih dimenzij. Seveda pa metodologija na drugi strani zahteva tehnike nelinearne optimizacije za izračun optimalnih uteži.

Tabela 11.2. Primerjava med MLP, RBF in normirano RBF mrežo.

Lastnosti	MLP	RBF	normalizirana RBF
Interpolacija	+	-	+
Ekstrapolacija	0	-	+
Lokalnost	-	++	+
Natančnost	++	0	0
Gladkost	++	0	+
Občutljivost na šum	++	+	+
Optimizacija parametrov	--	++*/--**	++*/--**
Strukturna optimizacija	-	+	-
Sprotna adaptacija	--	+	+
Hitrost učenja	--	+*/--**	+*/--**
Hitrost izračuna modela	+	0	0
Problem dimenzij	++	-	-
Interpretacija	--	0	0
Vključevanje omejitev	--	0	0
Uporaba apriorno znanja	--	0	0
Uporaba	++	0	-

* = linearna optimizacija, ** = nelinearna optimizacija,
 ++ / -- = zelo pozitivno / zelo negativno.

12. Mehki modeli

V tem poglavju je najprej uvod v mehko logiko in mehke sisteme, potem pa se osredotoči na učenje mehkih modelov na osnovi podatkov. Poglavlje je organizirano tako, da je v podpoglavlju 12.1 podan kratek uvod v mehko logiko. Različni tipi mehkih modelov so prikazani v podpoglavlju 12.2. Prikazana je tudi povezava med nevro-mehkimi sistemi in nevronskimi mrežami v poglavju 12.3. Na koncu je povzetek celotnega poglavja v 12.4.

12.1 Mehka logika

Mehka logika je bila razvita v letu 1965 [120] kot razsiritev klasične (Boolove) logike. Medtem, ko klasična logika pripisuje spremenljivki ali izjavi vrednost 1 za "pravilno" ali vrednost "0" za "napačno," mehka logika dovoljuje prireditve vrednosti na intervalu med [0, 1]. Razlog za to lahko najdemo v opazovanju načina človekovega razmišljanja, ki se odloča na zelo približnih ocenah raznih dejstev, ki si jih predstavi v obliki pravil. Za obravnavo takšnega koncepta je bil vpeljan mehanizem zapisa znanja na osnovi pravil v obliki *aproksimativnega sklepanja* (*ang. approximate reasoning*), ki temelji na mehki logiki.

V tem poglavju uvedemo nekaj osnovnih pojmov mehke logike in aproksimativnega sklepanja, ki so nujni za razumevanje mehkih modelov. Pri tem velja omeniti, da bomo uporabljali pri zapisu lingvističnih izjav uporabili angleške termine, ker gre za splošno uveljavljene operatorje.

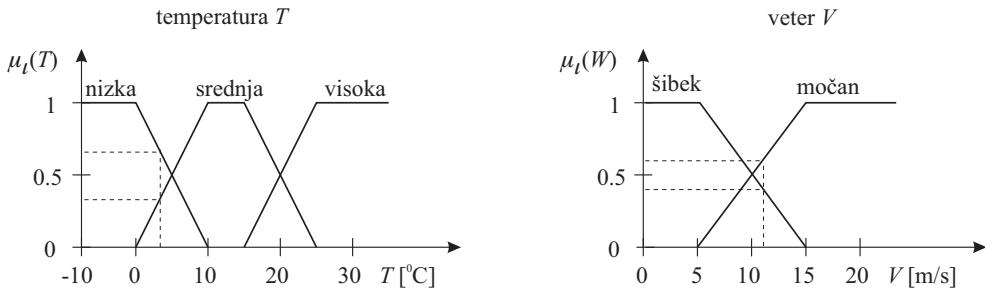
Za ilustracijo delovanja mehke logike si poglejmo enostaven primer. Mehka logika zapisuje relacije, znanje in odločitve v obliki pravil. V spodnjem primeru je zapisan vpliv vetra na občutek mraza, to pomeni, človekov občutek topote ni odvisen samo od temperature, temveč tudi od hitrosti vetra:

IF *temperatura = nizka* AND *veter = močan* THEN *občutek = zelo hladno* ,

kjer so elementi zapisa pravila naslednji:

IF - THEN	struktura pravil,
<i>temperatura, veter, občutek</i>	lingvistična spremenljivka,
<i>nizka, močan, zelo hladno</i>	lingvistični izrazi ali oznaake,
AND	operator, tukaj: konjunkcija,
<i>temperatura = nizka, veter = močan</i>	lingvistična izjava,
<i>temperatura = nizka AND veter = močan</i>	premisa, vzročni del pravila,
<i>občutek = zelo hladno</i>	posledica, konsekvenca, posledični del pravila.

Lingvistični spremenljivki *temperatura* in *veter* sta pogosto enostavno predstavljeni kot vhodni spremenljivki, linguistična spremenljivka *občutek* pa kot izhod. V celotni množici pravil (*ang. complete set of rules*) zapišemo vse mogoče kombinacije za *temperaturo* in *veter*:

Slika 12.1. Pripadnostne funkcije za linguistično spremenljivko *temperatura T* (levo) in *veter V* (desno).

- $R_1 : \text{IF } temp = \text{nizka} \text{ AND } veter = \text{močan} \text{ THEN } občutek = \text{zelo hladno},$
 $R_2 : \text{IF } temp = \text{srednja} \text{ AND } veter = \text{močan} \text{ THEN } občutek = \text{hladno},$
 $R_3 : \text{IF } temp = \text{visoka} \text{ AND } veter = \text{močan} \text{ THEN } občutek = \text{zmerno},$
 $R_4 : \text{IF } temp = \text{nizka} \text{ AND } veter = \text{šibek} \text{ THEN } občutek = \text{hladno},$
 $R_5 : \text{IF } temp = \text{srednja} \text{ AND } veter = \text{šibek} \text{ THEN } občutek = \text{zmerno},$
 $R_6 : \text{IF } temp = \text{visoka} \text{ AND } veter = \text{šibek} \text{ THEN } občutek = \text{toplo}.$

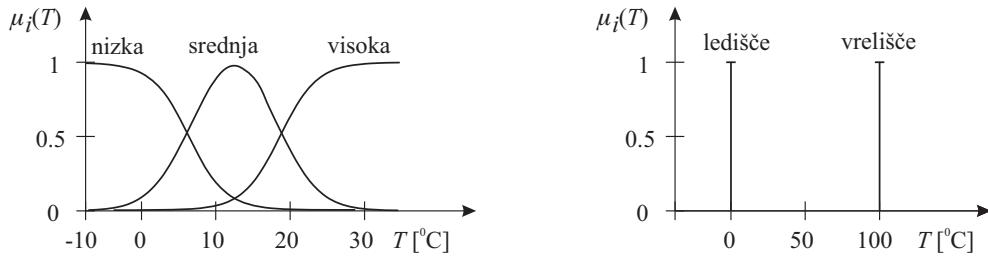
Skupno število pravil je odvisno od resolucije mehkih linguističnih spremenljivk. Od tega je bistveno odvisna tudi natančnost mehkega sistema. To razdelitev imenujemo tudi *granulacija*. Dokazali so, da lahko z mehkim sistemom aproksimiramo katerokoli vhodno-izhodno relacijo do poljubne natančnosti s povečevanjem granulacije. To pomeni, da so mehki sistemi univerzalni aproksimatorji [67, 117].

Poglejmo si posamezne dele mehkih pravil.

12.1.1 Pripadnostne funkcije

V mehkih pravilih so linguistične spremenljivke izražene z mehkimi množicami. V zgornjem primeru sta linguistični vhodni spremenljivki *temperatura* in *veter* definirani z linguističnimi izrazi **nizka**, **srednja**, **visoka** in **šibek**, **močan**. Ti linguistični izrazi so definirani z njimi povezanimi *pripadnostnimi funkcijami* (ang. *membership functions*). Slika 12.1 prikazuje možno obliko pripadnostnih funkcij. Preko pripadnostnih funkcij definiramo pripadnost (ang. *degree of membership*) določene *temperaturi* ali *vetru* ustrezni meksi množici, to pomeni, *temperatura T=3 °C* ima pripadnost množici **nizka** s pripadnostjo 0.7, **srednja** s pripadnostjo 0.3 in **visoka** s pripadnostjo 0. Temu izračunu, ki iz dejanskih vrednosti določene vhodne spremenljivke, (primer za $T=3 °C$) izračuna vrednosti pripadnosti, recemo *mehčanje* (ang. *fuzzification*). Pripadnostne funkcije so običajno funkcije ene same spremenljivke, $\mu_i(T)$ ali $\mu_i(V)$, kjer i definira linguistični izraz **nizka**, **srednja**, **visoka** ali **šibek**, **močan**. Na ta način mehki sistemi delujejo na posameznih vhodih ločeno in jih potem preko pravil medsebojno sestavljamo z AND ali OR operatorji; glej poglavje 12.1.2.

V mehkem sistemu procesiramo naprej vrednosti pripadnosti. To je nelinearno transformiranje vhodnih spremenljivk. Na ta način lahko izgubimo določen del informacije. Na primer, na sliki 12.1 lahko vidimo, da za *temperaturo*, ki je enaka $10 °C$ ali $15 °C$ ali za vrednosti na tem intervalu, dobimo enako vrednost pripadnosti. Druga pomembna lastnost je, da je vsota pripadnosti na sliki 12.1 enaka 1, to pomeni, da pri določeni vrednosti velja $\mu_{\text{nizka}}(T) + \mu_{\text{srednja}}(T) + \mu_{\text{visoka}}(T) = 1$, in $\mu_{\text{šibek}}(V) + \mu_{\text{močan}}(V) = 1$, ali bolj splošno



Slika 12.2. Pripadnostne funkcije so lahko gladke (levo), kar ne zmanjša informacije in je bolj primerno za učenje ali pa so lahko posebne (singleton) kar predstavlja ostre množice (desno).

$$\sum_{i=1}^M \mu_i(u) = 1 \quad \text{za vse } u , \quad (12.2)$$

kjer M definira število pripadnostnih funkcij za linguistično spremenljivko u . Čeprav ni zahtevano, da so pripadnostne funkcije normirane, je to največkrat izpolnjeno, saj olajša interpretacijo; glej poglavje 12.3.3.

Pripadnostne funkcije na sliki 12.1 so trikotne in trapezne. To vodi do izgube informacije tam, kjer imajo te pripadnostne funkcije odvod enak nič. Hkrati pa te funkcije niso zvezno odvedljive in zato imamo z njim lahko določene probleme pri učenju na osnovi podatkov. Zato se pogosto uporabljajo gladke, normirane funkcije, kot na primer Gaussove, Bellove in druge, da se temu izognemo.

Včasih imamo določene spremenljivke, ki zahtevajo ostro predstavitev. Takrat uporabimo pravokotne pripadnostne funkcije. Če je pripadnost določena samo za eno točko, potem dobimo posebno pripadnostno funkcijo (singleton) kot je to prikazano na sliki 12.2 desno. Poseben pripadnostne funkcije imajo vrednost 1 pri določeni vrednosti neodvisne spremenljivke (v tem primeru pri $0\text{ }^{\circ}\text{C}$ in $100\text{ }^{\circ}\text{C}$) pri ostalih pa vrednost 0.

Ko smo izračunali pripadnosti za vse linguistične spremenljivke in njihove izraze, v naslednjem koraku izvedemo operacije po posameznih pravilih (AND in OR).

12.1.2 Logični operatorji

Mehki operatorji so razširitve klasičnih logičnih operatorjev. *Negacija* linguistične izjave $T = \text{nizka}$ je izračunana na način

$$\text{NOT}(\mu_i(T)) = 1 - \mu_i(T) . \quad (12.3)$$

Za *konjunkcijo* dveh linguističnih izjav, kot na primer $T = \text{nizka}$ in $V = \text{močan}$, obstaja več različnih operatorjev, ki jih imenujemo tudi *t-norme*. Najbolj osnovne t-norme so prikazane na sliki 12.3):

$$\text{Min: } \mu_i(T) \text{ AND } \mu_j(V) = \min(\mu_i(T), \mu_j(V)) , \quad (12.4a)$$

$$\text{Produkt: } \mu_i(T) \text{ AND } \mu_j(V) = \mu_i(T)\mu_j(V) , \quad (12.4b)$$

$$\text{Omejena razlika: } \mu_i(T) \text{ AND } \mu_j(V) = \max(0, \mu_i(T) + \mu_j(V) - 1) . \quad (12.4c)$$

Za *disjunkcijo* dveh linguističnih spremenljivk ravno tako obstaja več različnih operatorjev, ki jih imenujemo tudi *t-konorme ali s-norme*. Najbolj osnovne t-konorme so:

$$\text{Max: } \mu_i(T) \text{ OR } \mu_j(V) = \max(\mu_i(T), \mu_j(V)) , \quad (12.5a)$$

$$\text{Algebraična vsota: } \mu_i(T) \text{ OR } \mu_j(V) = \mu_i(T) + \mu_j(V) - \mu_i(T)\mu_j(V) , \quad (12.5b)$$

$$\text{Omejena vsota: } \mu_i(T) \text{ OR } \mu_j(V) = \min(1, \mu_i(T) + \mu_j(V)) . \quad (12.5c)$$

Za namene klasifikacije je zelo pogosta izbira min in max operatorja. Za aproksimacijske probleme pa sta bolj primerna produkt in algebraični produkt, zaradi svoje gladkosti in odvedljivosti.

Vse t-norme in t-konorme se lahko uporabijo na poljubno velikem številu linguističnih izjav z gnezdenjem operatorjev na naslednji način:

$$A \text{ AND } B \text{ AND } C = (A \text{ AND } B) \text{ AND } C = A \text{ AND } (B \text{ AND } C) .$$

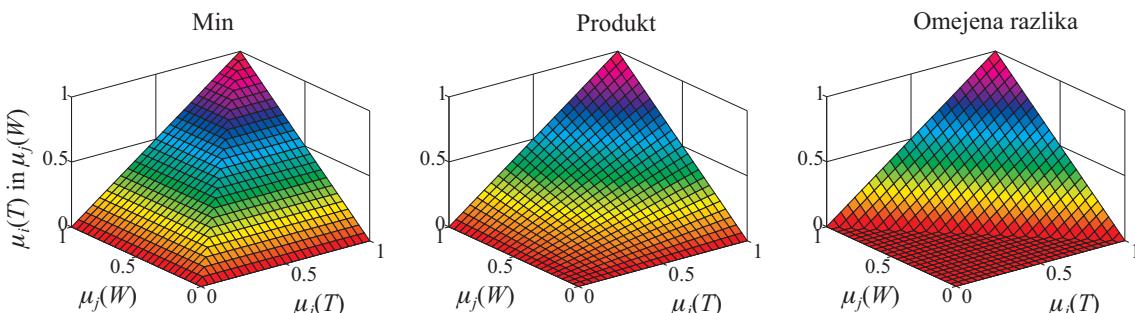
V sistemih kot so mehke-nevronske mreže so linguistične izjave večkrat sestavljene samo z AND operatorji, kot je to v pravilih R_1-R_6 v gornjem primeru. V taki *konjunktivni obliki* lahko izrazimo vse vhodno-izhodne relacije.

12.1.3 Stopnja izpolnjenosti pravila

Z logičnimi operatorji lahko sestavimo posamezne linguistične izraze v linguistične izjave in sesestavimo premiso pravila. Na primer, *temperatura* je lahko $T = 3^\circ\text{C}$ in *veter* $W = 11 \text{ m/s}$. To ustreza naslednjim vrednostim pripadnosti: $\mu_{\text{nizka}}(T) = 0.7$, $\mu_{\text{srednja}}(T) = 0.3$, $\mu_{\text{visoka}}(T) = 0$, $\mu_{\text{šibek}}(V) = 0.4$ in $\mu_{\text{močan}}(V) = 0.6$; glej sliko 12.1. V pravilu R_1 izračunamo konjunkcijo med $\mu_{\text{nizka}}(T) = 0.7$ in $\mu_{\text{močan}}(V) = 0.6$. To rezultira v vrednosti 0.6, če uporabimo operator min, 0.42 za operator produkta in 0.3 za operator omejene razlike. Izhod mehkega sistema je na ta način odvisen tudi od izbire operatorjev.

Kombinacija pripadnosti vseh linguističnih izrazov določa *stopnjo izpolnjenosti pravila* ali *aktivacijo pravila*, ker izraža kako dobro se premlisa ujemata z danimi vrednostmi vhodnih spremenljivk (v tem primeru $T = 3^\circ\text{C}$ in $V = 11 \text{ m/s}$).

Za celoten mehki sistem so pomembne samo stopnje izpolnjenosti, ki so večje od nič. Za strogo lokalne pripadnostne funkcije, kot so trikotne na sliki 12.1, se aktivira samo del pravil pri določenem vhodu. Za $T = 3^\circ\text{C}$ in $V = 11 \text{ m/s}$ imata pravili R_3 in R_6 nično aktivacijo, ker je $\mu_{\text{visoka}}(T) = 0$. Paziti moramo, da s pravili izpolnimo celoten vhodni prostor, da se izognemo situacijam, ko pri določeni vhodnih vrednostih ni aktivirano nobeno pravilo. V primeru nestrogo lokalnih pripadnostnih funkcij ta problem odpade, saj imamo vedno avtomatično izpolnjena *vsa* pravila, čeprav z zelo majhnimi vrednostmi.



Slika 12.3. Prikaz različnih t-norm (operatorji konjunkcije).

12.1.4 Akumulacija in ostrenje

Potem, ko je izračunana stopnja izpoljenosti posameznega pravila moramo določiti prispevke posameznih posledničnih delov in jih sestaviti, da dobimo izhod mehkega sistema. Temu rečemo *akumulacija*. Navadno je izhod mehkega sistema mehka množica, ki jo je potrebno za nadaljnje delo transformirati v ostro obliko. To imenujemo *ostrenja* (ang. *defuzzification*). Ta seveda ni potrebna v primeru, ko imamo na posledični strani že ostro vrednost, ali pa če gre za rezultat, ki služi kvalitativnim ocenam.

12.2 Tipi mehkih sistemov

V tem poglavju si bomo ogledali tri različne tipe mehkih sistemov: linguistični ali Mamdanijev, poseben ali singleton in mehki sistem Takagi-Sugeno.

12.2.1 Lingvistični mehki sistemi

Linguistični mehki sistem, ali tudi *Mamdanijev* mehki sistem [79], zapisuje pravila mehkega sistema v obliki

$$R_i : \text{ IF } u_1 = A_{i1} \text{ AND } u_2 = A_{i2} \text{ AND } \dots u_p = A_{ip} \text{ THEN } y = B_i ,$$

kjer so u_1, \dots, u_p vhodi v mehki sistem, ki jih zapišemo v vhodni vektor \underline{u} , y je izhod, indeks $i = 1, \dots, M$ gre preko vseh M pravil mehkega sistema, A_{ij} predstavlja mehko množico za spremenljivko u_j v pravilu i in B_i je mehka množica izhodne spremenljivke pri pravilu i . Primer v poglavju 12.1 je tega tipa. V primeru procesiranja tovrstnega sistema je potrebno:

mehčanje → agregacija → aktivacija → akumulacija → ostrenje

Z mehčanjem na osnovi pripadnostnih funkcij preslikamo določeno natančno vrednost spremenljivke v pripadnost. Z agregacijo sestavimo posamezne linguistične izjave v stopnjo aktivacije pravila glede na operatorje med njimi. Ti koraki so enaki pri vseh tipih mehkih sistemov. Zadnja dva koraka pa sta odvisna od tipa mehkega sistema.

Koraki mehkega sklepanja.

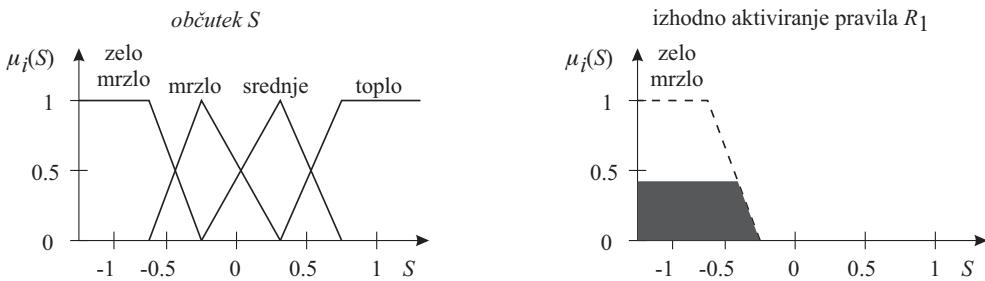
- V koraku *mehčanja* se izračuna stopnja pripadnosti za vse linguistične izjave. Zapišemo jih z $\mu_{ij}(u_j)$, $i = 1, \dots, M$, $j = 1, \dots, p$.
- V koraku *agregacije* se te stopnje pripadnosti znotraj enega pravila sestavijo glede na zahtevane operatorje. Če je mehki sistem zapisan v konjunkcijski obliki in uporabimo operator produkta za t-normo, je stopnja aktivacije pravila i enaka

$$\mu_i(\underline{u}) = \mu_{i1}(u_1) \cdot \mu_{i2}(u_2) \cdot \dots \cdot \mu_{ip}(u_p) . \quad (12.6)$$

- V koraku *aktivacije* se stopnje aktivacije pravil prenesejo na izhodno mehko množico. Na primer, to lahko storimo tako, da odrežemo ustrezno pripadnostno funkcijo, ki je izhod tega pravila, pri vrednosti, ki jo definira stopnja izpoljenosti pravila

$$\mu_i^{\text{akt}}(\underline{u}, y) = \min(\mu_i(\underline{u}), \mu_i(y)) , \quad (12.7)$$

kjer je $\mu_i(y)$ izhodna pripadnostna funkcija ustrezne mehke množice B_i in $\mu_i(\underline{u})$ je stopnja izpoljenosti pravila R_i . Slika 12.4 prikazuje pripadnostne funkcije in izhodno izpoljenost pravila R_1 iz primera v poglavju 12.1.3, kjer je izpoljenost prvega pravila enaka 0.42.



Slika 12.4. Pripadnostne funkcije za linguistično spremenljivko *občutek S* (levo) in izračun izhodne aktivacije (desno). Črno področje predstavlja izhodno aktivacijo za pravilo R_1 in je dobljeno z odrezom izhodne pripadnostne funkcije pri stopnji izpoljenosti pravila 0.42.

- V *akumulaciji* sestavimo vse prispevke izhodnih aktivacij. To lahko storimo tako, da izračunamo maksimum vseh izhodnih aktivacij

$$\mu^{\text{aku}}(\underline{u}, y) = \max_i [\mu_i^{\text{akt}}(\underline{u}, y)] . \quad (12.8)$$

Akumulacija rezultira v eni mehki množici, ki je mehki izhod sistema. To je lahko že končni rezultat, če ne potrebujemo ostre vrednosti.

- Če je za izhod mehkega sistema zahtevana ostra vrednost, potem moramo izvesti še postopek *ostrenja*. Najpogostejsi način za ta korak je uporaba metode *težišča*, ki jo izvedemo na naslednji način is,

$$\hat{y} = \frac{\int_{y_{\min}}^{y_{\max}} y \mu^{\text{aku}}(\underline{u}, y) dy}{\int_{y_{\min}}^{y_{\max}} \mu^{\text{aku}}(\underline{u}, y) dy} , \quad (12.9)$$

kjer je $\mu^{\text{aku}}(\underline{u}, y)$ mehka množica, ali drugače, akumulacija izhodnih aktivacij.

Slika 12.5 opisuje celoten postopek mehkega sklepanja (*ang. fuzzy inference*) z *min* in *max* operatorjema za konjunkcijo in disjunkcijo, z *max* operatorjem za akumulacijo in težiščno metodo za ostrenje.

12.2.2 Posebni mehki sistemi

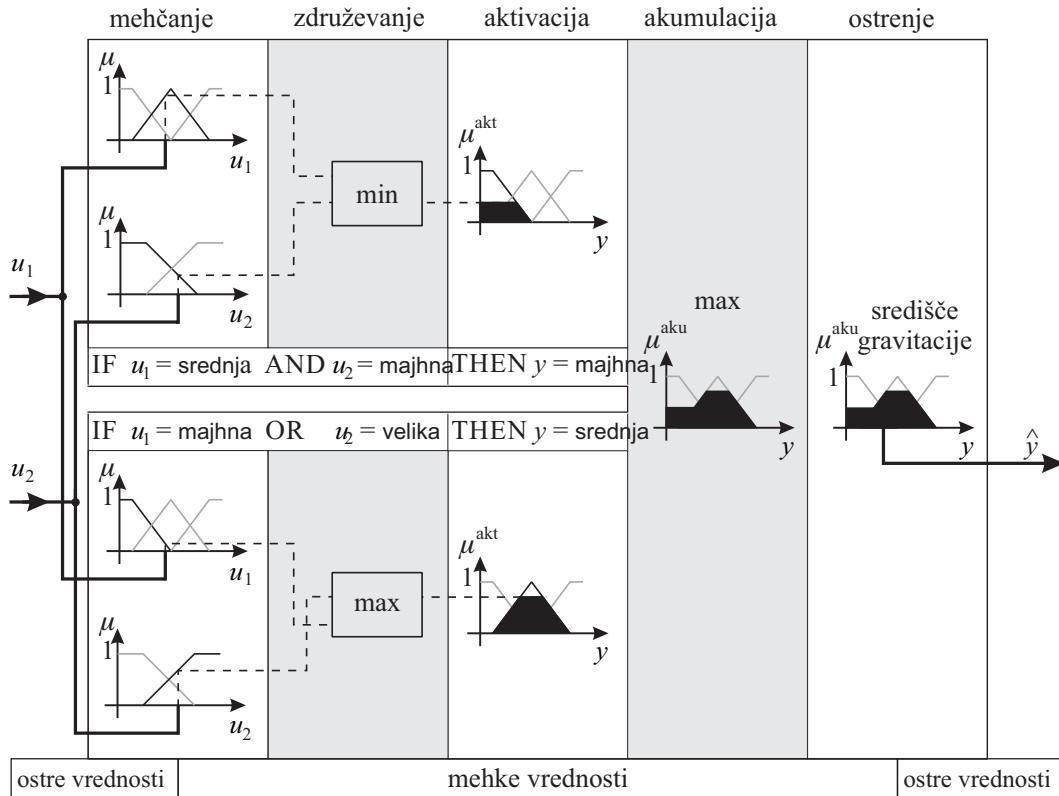
Posebni mehki sistem odpravlja in poenostavlja izračune pri koraku ostrenja. To bistveno zmanjša čase računanja. Zaradi tega so pogosti v praktičnih aplikacijah.

Pravilo posebnega mehkega sistema ima naslednjo obliko:

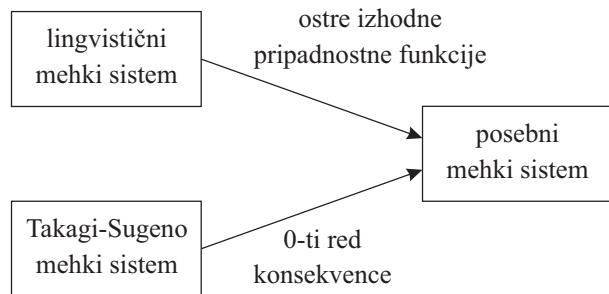
$$R_i : \text{ IF } u_1 = A_{i1} \text{ AND } u_2 = A_{i2} \text{ AND } \dots u_p = A_{ip} \text{ THEN } y = s_i ,$$

kjer je s_i realna vrednost na posledični strani pravila in jo imenujemo *singleton* pravila i . Določa položaj trivialne izhodne pripadnostne funkcije; glej sliko 12.2. Z uporabo tega tipa se poenostavijo postopki računanja, saj se ostrenje poenostavi v enostavno uteženo vsoto. Izhod posebnega mehkega sistema je definiran kot

$$\hat{y} = \frac{\sum_{i=1}^M s_i \mu_i(\underline{u})}{\sum_{i=1}^M \mu_i(\underline{u})} , \quad (12.10)$$



Slika 12.5. Sklepanje v primeru mehkega sistema z dvema vhodoma in dvema praviloma.



Slika 12.6. Posebni mehki sistem kot poseben primer linguističnega ali Takagi-Sugeno mehkega sistema.

kjer je M število pravil in $\mu_i(\underline{u})$ je stopnja izpolnjenosti pravila R_i glede na enačbo 12.6. To pomeni, da je mehki izhod utežena vsota posameznih singletonov, kjer so uteži enake stopnjam izpolnjenosti pravila. Normalizacijski imenovalec v enačbi 12.10 je prisoten v vseh oblikah mehkih sistemov in skrbi za pravo interpretacijo posledičnih delov pravil.

Vidimo lahko, da je imenovalec v enačbi 12.10 enak 1 takrat, ko so pripadnostne funkcije normirane na vrednost 1 (glej enačbo 12.2) in je množica pravil kompletна. Množica pravil je kompletна, če je vsaka kombinacija vhodnih lingvističnih oznak zapisana z enim pravilom. To je največkrat izpolnjeno. V tem primeru se pravilo za izračun izhoda posebnega mehkega sistema še poenostavi na

$$\hat{y} = \sum_{i=1}^M s_i \mu_i(\underline{u}) . \quad (12.11)$$

12.2.3 Mehki sistem Takagi-Sugeno

Leta 1985 sta Takagi in Sugeno [110] predlagala nov tip mehkega sistema z naslednjim zapisom pravil:

$$R_i : \text{ IF } u_1 = A_{i1} \text{ AND } \dots \text{ AND } u_p = A_{ip} \text{ THEN } y = f_i(u_1, u_2, \dots, u_p) .$$

Ta zapis je razširitev posebnih mehkih sistemov. Medtem, ko so posebni mehki sistemi posebna vrsta lingvističnih mehkih sistemov s posebnim tipom pripadnostnih funkcij, funkcije $f_i(\cdot)$ na posledični strani definitivno ne moremo predstaviti na ta način. Lingvistično interpretabilna je samo premisa Takagi-Sugeno mehkega sistema. Kvaliteta mehkega sistema Takagi-Sugeno pride do izraza pri modeliranju *dinamičnih sistemov*, kjer je njihova interpretabilnost bistveno večja kot pri vseh ostalih splošnih aproksimatorjih.

Če je izbira funkcije $f_i(\cdot)$ trivialna in izberemo kar konstanti s_i , potem dobimo posebni mehki sistem. V tem primeru ga imenujemo kar mehki model Takagi-Sugeno *ničtega reda*, ker je ničti red v Taylorjevi vrsti funkcije $f_i(\cdot)$ kar konstanta. Pogosto srečamo mehki model Takagi-Sugeno *prve vrste ali prvega reda*. To pomeni, da posledični del *linearna* funkcija vhodov

$$y = w_{i0} + w_{i1}u_1 + w_{i2}u_2 + \dots + w_{ip}u_p . \quad (12.12)$$

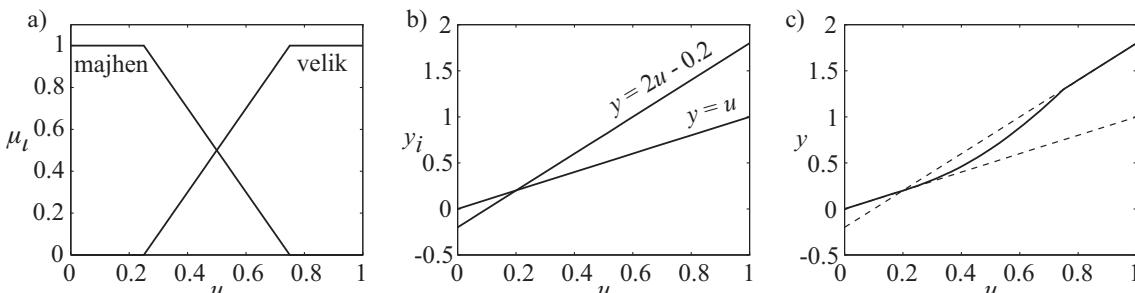
Izhod iz mehkega sistema Takagi-Sugeno lahko izračunamo kot

$$\hat{y} = \frac{\sum_{i=1}^M f_i(\underline{u}) \mu_i(\underline{u})}{\sum_{i=1}^M \mu_i(\underline{u})} , \quad (12.13)$$

kar je enostavna razširitev za izhod posebnega mehkega sistema v enačbi 12.10. Slika 12.7 prikazuje izračun za enodimensionalni mehki sistem Takagi-Sugeno z dvema praviloma:

$$\begin{aligned} R_1 : & \text{ IF } u = \text{majhen} \text{ THEN } y = u , \\ R_2 : & \text{ IF } u = \text{velik} \text{ THEN } y = 2u - 0.2 . \end{aligned}$$

Vse kar velja za posebne mehke sisteme lahko razširimo tudi na Takagi-Sugeno mehke sisteme s tem, da s_i zamenjamo s funkcijo $w_{i0} + w_{i1}u_1 + w_{i2}u_2 + \dots + w_{ip}u_p$. Takagi-Sugeno mehki modeli so doživelji veliko uporabo in so zelo raziskani predvsem zaradi svoje odlične interpretabilnosti.



Slika 12.7. Enodimensionalni mehki sistem Takagi-Sugeno z dvema praviloma: a) pripadnostni funkciji, b) linearni rezultati obih posledičnih delov, c) izhod mehkega sistema (polna črta) in linearnih funkcij (črtkana). Opozorimo naj na to, da je imenovalec v enačbi 12.13 enak 1, ker je vsota pripadnosti v vseh točkah prostora enaka 1.

12.3 Nevro-mehke mreže

V tem poglavju si bomo ogledali nevro-mehke mreže, ki temeljijo na strukturi posebnih mehkih modelov. V poglavju 13 pa smo prikazani nevro-mehki modeli Takagi-Sugeno. Nevro-mehke mreže so mehki modeli, ki niso načrtovani samo na osnovi eksperimentnega znanja, temveč z učenjem na osnovi podatkov. Obstaja veliko različnih metod učenja mehkih modelov, ki omogočajo neposredno optimizacijo mehkega modela. Na začetku je bilo učenje mehkih modelov realizirano tako, da je bil mehki model pretvorjen v nevronsko strukturo in potem naučen z optimizacijskimi algoritmi nevronskega mrež.

V tem poglavju si bomo ogledali, katere komponente posebnega mehkega modela lahko optimiramo na osnovi podatkov in kako učinkovito lahko to izvedemo, ne da bi vplivali na slabšo interpretabilnost modela. Možnost vnosa predznanja v mehki model je ena od najpomembnejših lastnosti tega pristopa in prednost pred ostalimi arhitekturami. Seveda pa obstaja določen kompromis med interpretabilnostjo in natančnostjo. Če natančnost modela ni prvi cilj modeliranja, potem je izbira mehkih modelov zelo dobra, saj nam model nudi veliko možnost interpretacije in vpogleda v funkciranje.

Za uporabo mehkih modelov lahko navedemo predvsem dva najpomembnejša razloga:

- *Uporaba apriornega znanja.* Omogoča izboljšati natančnost modela in zmanjša potrebno količino podatkov in definirati model na področju, kjer nimamo izmerjenih podatkov.
- *Izboljšano razumevanje delovanja procesa.* Na osnovi mehkih modelov lahko bolje razumemo različna obnašanja procesa v različnih delovnih točkah.

12.3.1 Mehke bazne funkcije

Nespecifični mehki model lahko zapišemo v bazni formulaciji. Če je mehki model zapisan v konjunktivni obliki in za t-normo uporabimo produkt, potem je stopnja izpolnjenosti pravila

$$\mu_i(\underline{u}) = \prod_{j=1}^p \mu_{ij}(u_j). \quad (12.15)$$

Funkcija $\mu_i(\underline{u})$ predstavlja *večdimenzionalno* pripadnostno funkcijo, ki je s produktno konstrukcijo sestavljena iz več enodimensinalnih pripadnostnih funkcij $\mu_{ij}(u_j)$. Slika 12.8 prikazuje, kako lahko enodimensionalno trikotno in Gaussovo pripadnostno funkcijo sestavimo v dvodimensionalno pripadnostno funkcijo z uporabo operatorja produkta in minimuma. Kot je prikazano na sliki 12.8d, je večdimenzionalna pripadnostna funkcija lahko gladka samo takrat, kadar so gladke vse enodimensinalne pripadnostne funkcije, ki jo sestavljajo, in je uporabljen gladek operator.

Mehki modeli konstruirani na ta način imajo pravila, ki so konjunkcija vseh možnih kombinacij enodimensinalnih pripadnostnih funkcij. Na ta način je tak mehki model zelo podvržen problemu velikega števila dimenzij. Interpretacija mehkega modela z večdimenzionalnimi baznimi funkcijami je naslednja

$$\hat{y} = \sum_{i=1}^M s_i \Phi_i(\underline{u}) \quad \text{za} \quad \Phi_i(\underline{u}) = \frac{\mu_i(\underline{u})}{\sum_{j=1}^M \mu_j(\underline{u})}. \quad (12.16)$$

Imenovalec v enačbi 12.16 poskrbi za to, da je izhod mehkega modela vedno utežena vsota singlonov, ker je vsota uteži $\Phi_i(\underline{u})$, $i = 1, \dots, M$ vedno enaka 1 za vsak vhodni vektor \underline{u} . Če

ima mehki model kompletni nabor pravil in so enodimenzionalne pripadnostne funkcije normirane (enotska particija) po vsaki dimenziji, potem je tudi bazne funkcije $\Phi_i(\underline{u})$ tvorijo enotsko praticijo, ker je $\sum_{j=1}^M \mu_j(\underline{u}) = 1$.

12.3.2 Optimizacija mehkega modela

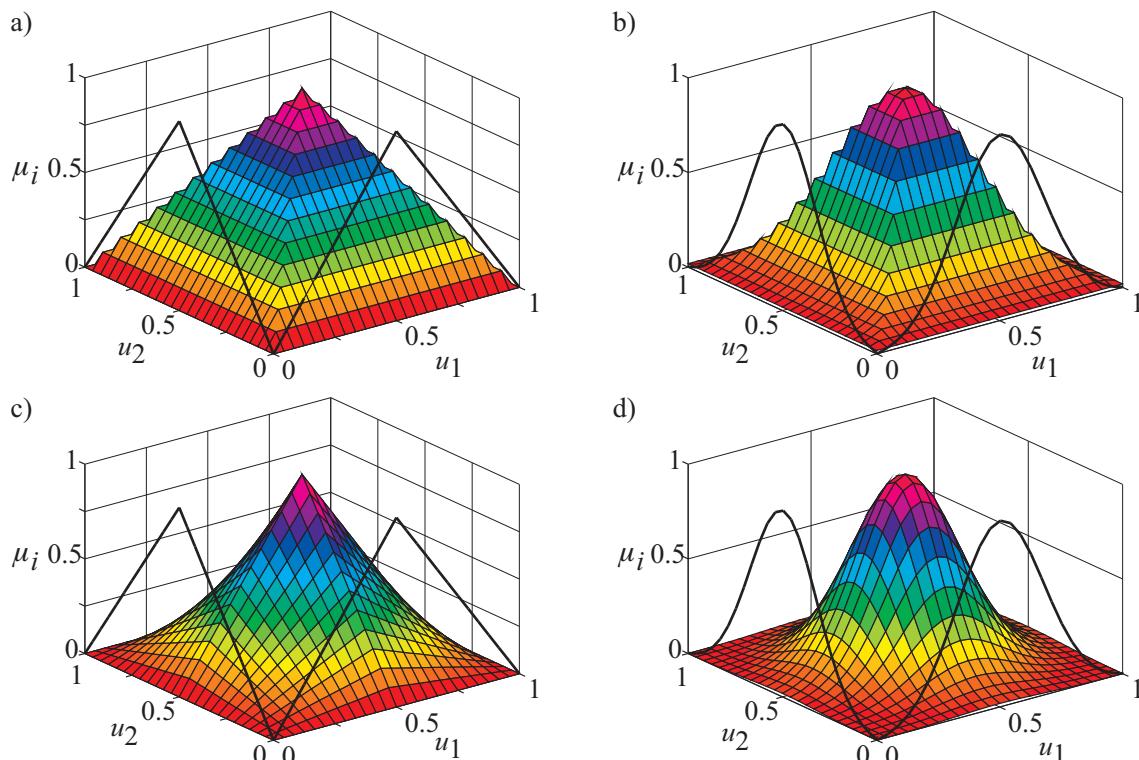
V mehkih modelih lahko optimiramo veliko število različnih parametrov. Glavno pravilo pa ostaja, da stvari, ki jih lahko določimo z znanjem o procesu ne poskušamo optimirati.

Optimiranje posledičnih parametrov. Parametri posledičnega dela mehkega modela ustrezaajo izhodnim utežem pri formulaciji modela z baznimi funkcijami. Zaradi tega jih je relativno enostavno oceniti. Za posebni mehki model in za modele Takagi-Sugeno, ki imajo linearne izhodne funkcije $f_i(\cdot)$ so ti parametri linearni. Zaradi tega jih lahko enostavno ocenimo z metodo najmanjših kvadratov .

Pogosto optimiramo parametre tako, da so parametri premise (parametri na skritem nivoju pri nevronskih mrežah) kar konstantni in ekvidistantno razdelijo vhodne spremenljivke, optimiramo pa samo parametre v izhodnem delu pravila.

Optimiranje parametrov premise. Parametri v premisi pravil so tisti, ki definirajo pripadnostne funkcije, njihove položaje in širino. Ustrezajo parametrom na skritem nivoju nevronskih mrež in so nelinearni. Zaradi tega se poskušamo izogniti njihovi optimizaciji, če je možno, in optimiramo samo linearne parametre v posledičnem delu.

Postavlja se vprašanje kdaj je smiselno uporabiti nelinearno lokalno ali globalno optimizacijsko tehniko za optimiranje nelinearnih parametrov. Ena od metod, ki se pogosto srečuje v



Slika 12.8. Konstrukcija večdimenzionalne pripadnostne funkcije: a) trikotna pripadnostna funkcija z operatorjem min, b) Gaussova pripadnostna funkcija z operatorjem min, c) trikotna pripadnostna funkcija z operatorjem produkta, d) Gaussova z operatorjem produkta.

takih primerih, je metoda genetskih algoritmov (GA) s katero lahko dobro optimiramo parameter v premisi. Uporabiti jo je smiselno takrat, ko je znanje o procesu zares majhno ali ga sploh nimamo. Ne glede na to, kako izberemo pripadnostne funkcije, pa je potrebno biti pazljiv pri njihovem normiranju in s tem povezanim efektom; glej poglavje 12.3.3.

Optimizacija strukture pravil. Optimizacija strukture pravil je zelo pomembna, ker na ta način lahko zmanjšamo problem dimenzij. Optimizacija strukture pravil je kombinacijski problem, ki ga lahko rešujemo z uporabo izbire linearnih podmnogic ali z uporabo nelinearne globalne optimizacije.

12.3.3 Interpretacija mehkih modelov

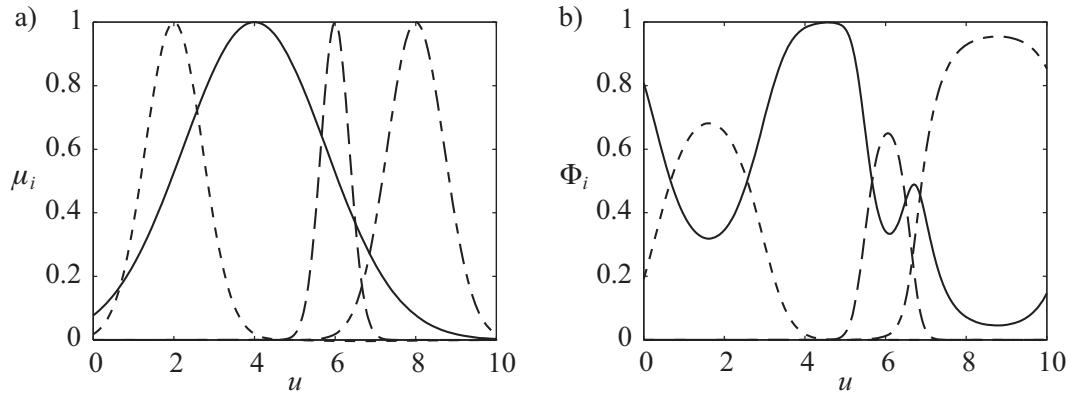
Glavna razlika med mehkimi sistemi in ostalimi nelinearnimi aproksimatorji je v možnosti interpretacije na osnovi pravil, ki so zelo razumljiva načinu človekovega razmišljenja. Zato je to eden od najpomembnejših vidikov pri ocenjevanju kvalitete mehkega modela.

Na interpretabilnost mehkega sistema lahko vpliva:

- *Število pravil.* Če je število pravil preveliko, je interpretabilnost vprašljiva. V primeru velikega števila vhodnih spremenljivk je to navadno res, ker poskušamo izpolniti vse možne kombinacije na vhodni strani. To lahko delno odpravimo samo z manjšo granulacijo vhodov, na račun manjše kompleksnosti.
- *Število linguističnih izjav v premisi pravil.* Če pravila vsebujejo več linguističnih izjav, jih težko razumemo. Zgornja meja je okoli tri.
- *Normiranje vhodnih pripadnostnih funkcij (enotske particije).*

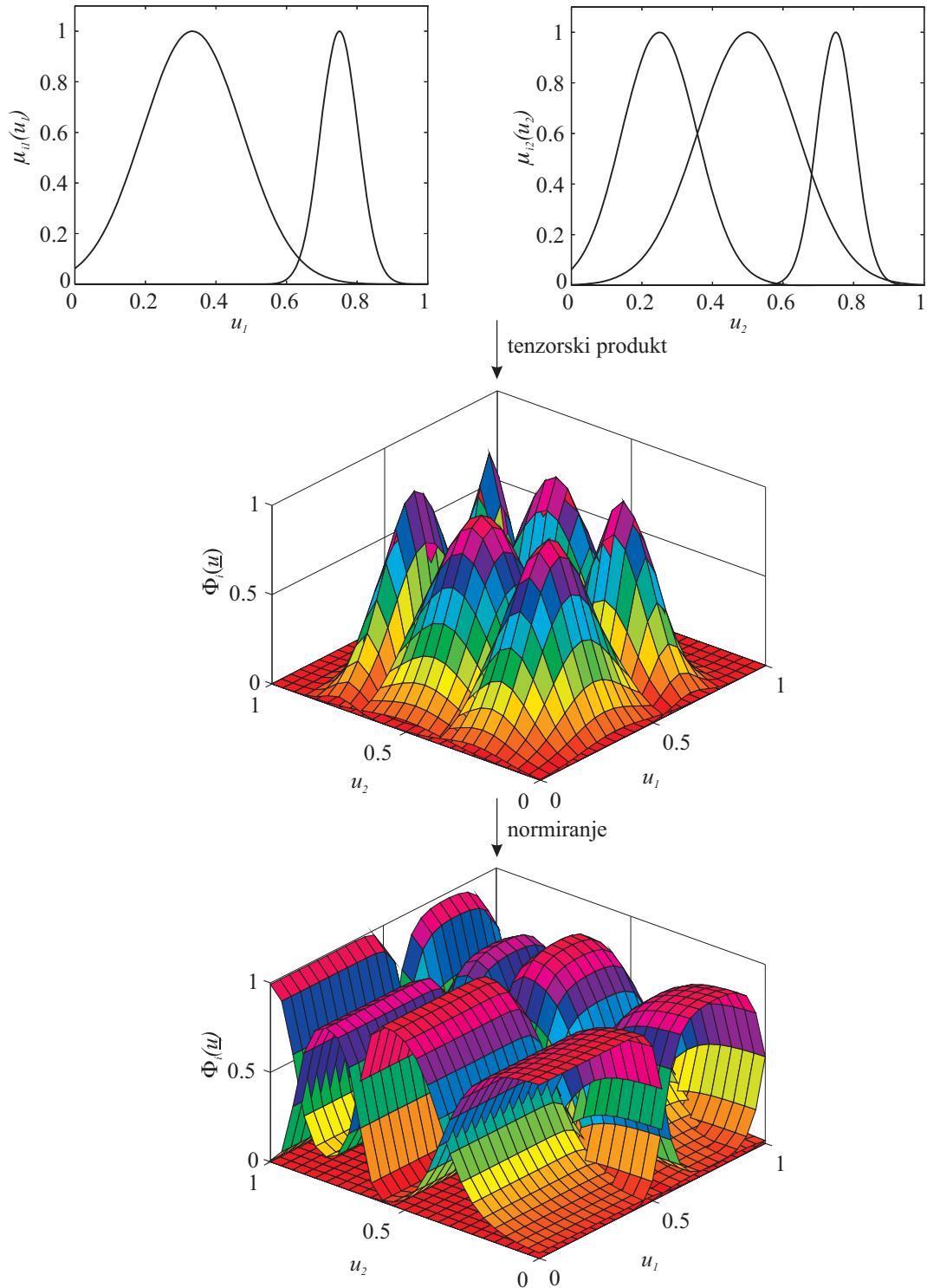
Slika 12.9 kaže nezaželen efekt normiranja Gaussovin pripadnostnih funkcij, ki nimajo enake širine. Kar vodi v nelokalno naravo normirane pripadnostne funkcije. Četudi pripadnostnih funkcij eksplisitno ne normiramo, se normiranje izvede z imenovalcem v enačbi 12.16. Slike 12.10 in 12.11 prikazujeta stranski efekt normiranja, ki se pojavi tako pri normiranih kot tudi pri nenormiranih pripadnostnih funkcijah.

Slika 12.12 kaže neželen efekt reaktivacije pripadnostne funkcije, ki se pojavi, če so funkcije različnih standardnih deviacij. Če imajo enake širine, se to ne more zgoditi. Zgodi pa se zunaj vhodne učne domene in je nezaželeno le v primeru ekstrapolacije.



Slika 12.9. a) Gaussove pripadnostne funkcije z različnimi standardnimi deviacijami. b) Normirana Gaussova pripadnostna funkcija, ki ima enotsko particijo, ki ima vsoto 1. Druga pripadnostna funkcija (polna črta) ima večjo standardno deviacijo in zaradi tega postane dominantna za $u \rightarrow -\infty$ in $u \rightarrow \infty$. Zaradi tega postane nelokalna. Pravilo, ki jo vsebuje ima dominantni vpliv ne samo okoli njenega centra $u = 4$ ampak tudi okoli $u = 0$. To je nezaželen vpliv normiranja pripadnostnih funkcij v enačbi 12.16. Isti efekt pa lahko zaznamo tudi, če uporabimo nenormirane funkcije (a).

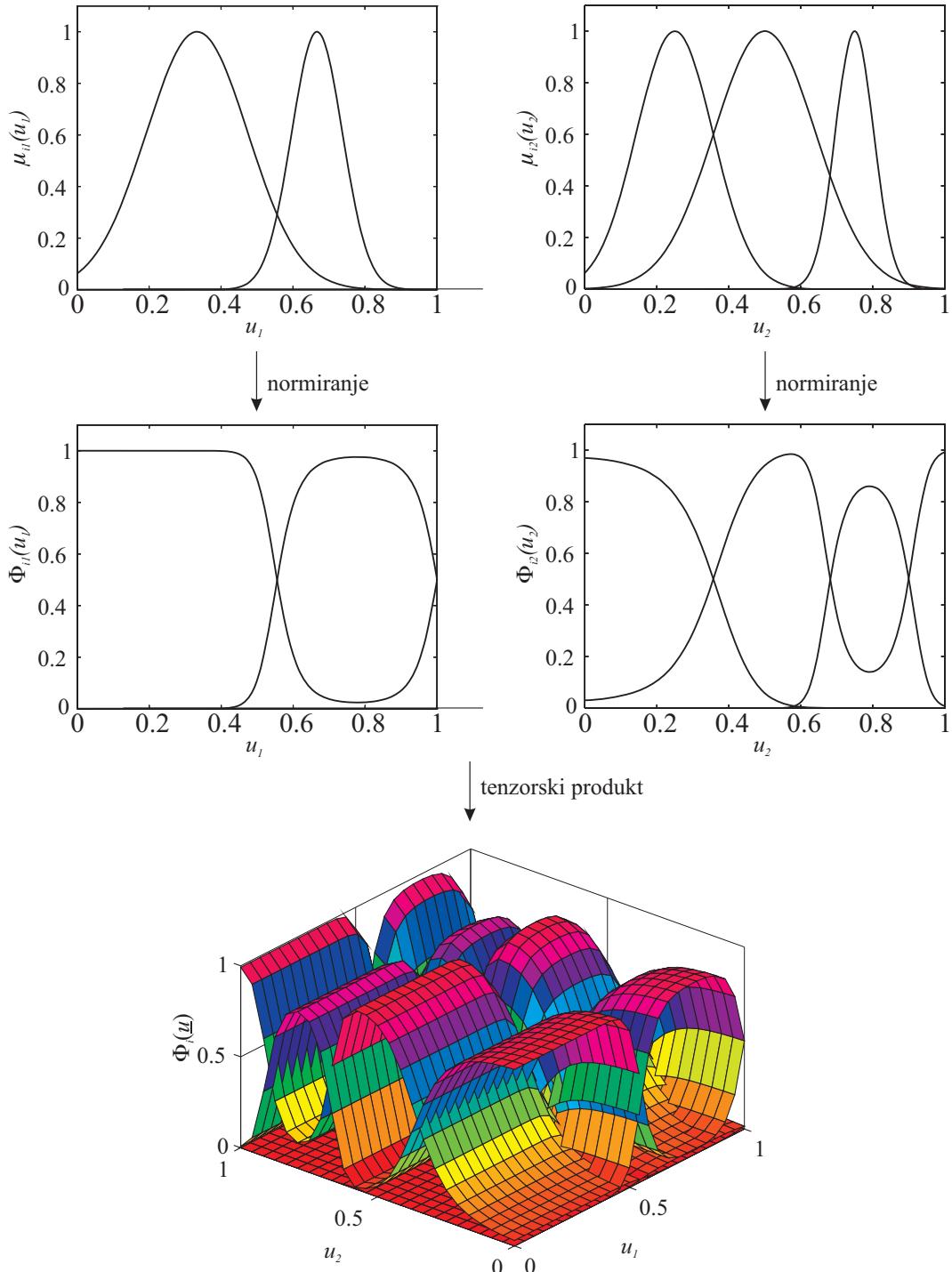
Gornja diskusija predstavlja določene omejitve za mehke modele, kar je cena za odlično interpretabilnost, ki jo dobimo z mehkimi modeli.



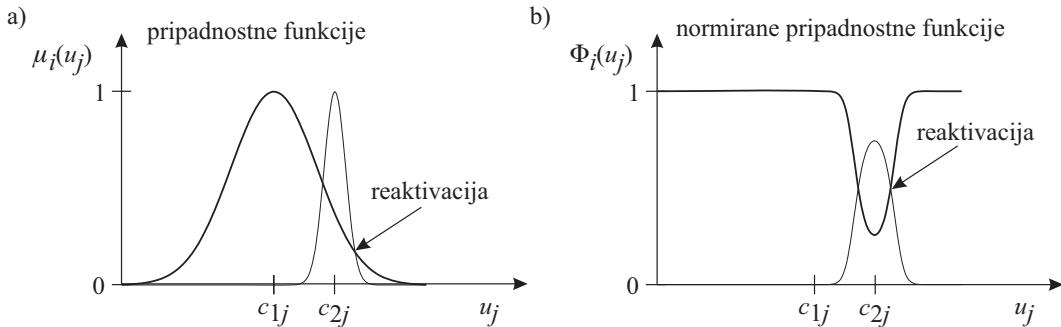
Slika 12.10. Stranski efekt normiranja se pojavi, če normiramo večdimenzijsko pripadnostno funkcijo, da dobimo bazne funkcije.

12.3.4 Simulacijski primer

Aproksimacijski primer, ki smo ga že obravnavali v poglavju 11.2.5 v primeru večnivojskega perceptronja, obravnavamo tudi v primeru mehkega sistema. Prikazuje lastnosti aproksimacije z posebnimi nevro-mehkimi modeli.



Slika 12.11. Stranski efekt normiranja se pojavi tudi ko enodimenzionalne pripadnostno funkcijo normiramo, da dobimo normirane enodimenzionalne pripadnostne funkcije. Ta efekt se pojavi, ko formiramo tensorski produkt. Rezultat je enak kot na sliki 12.10. Zaporedje operacij normiranja in tensorskega produkta je nepomembno.



Slika 12.12. Reaktivacija bazne funkcije se zgodi, če imamo : a) Gaussovo pripadnostno funkcijo različnih širin, b) normirano pripadnostno funkcijo.

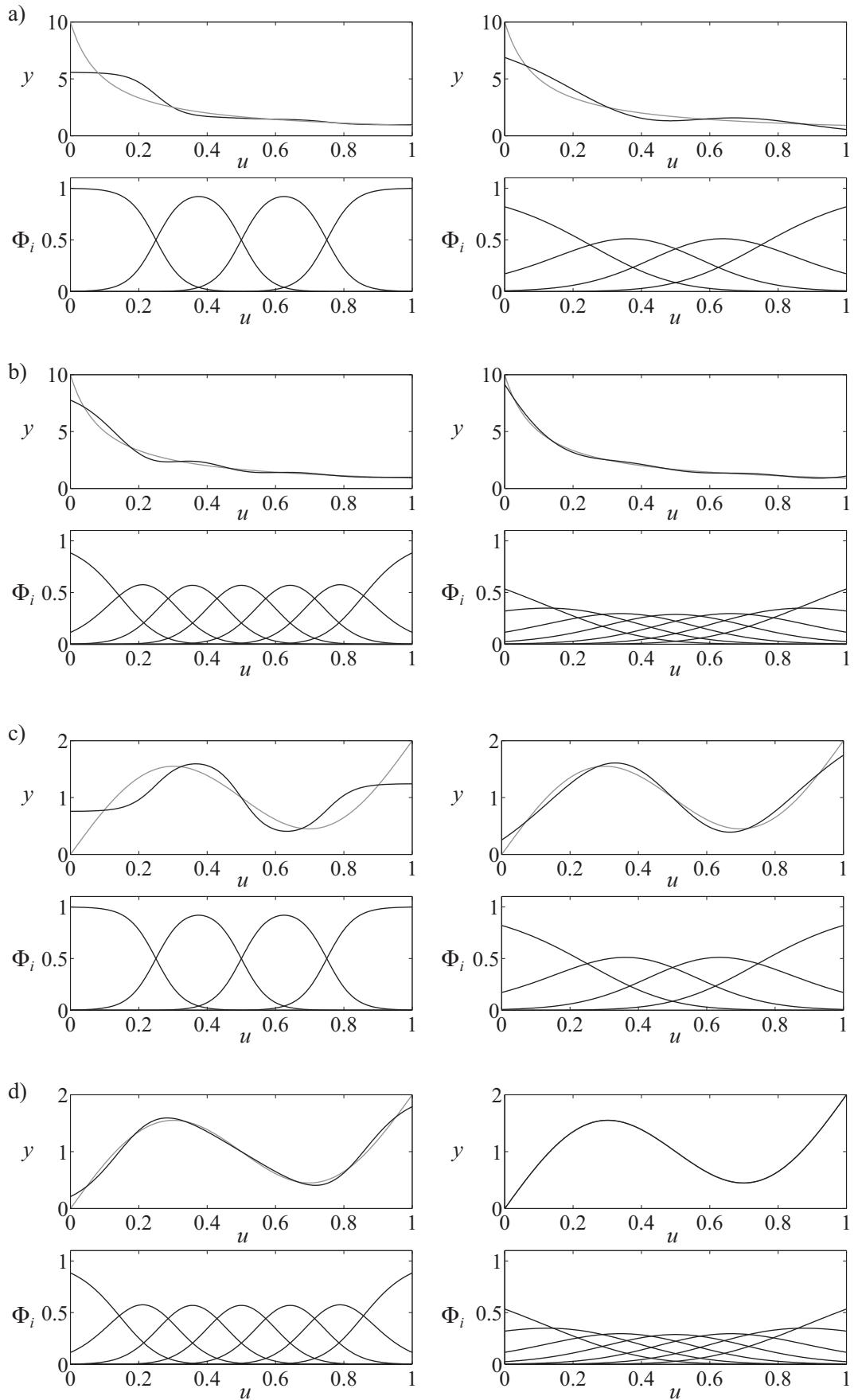
Pripadnostne funkcije so ekvidistančno razdeljene po vhodnem prostoru. Rezultati nevromehkega modela na sliki 12.13 so prikazani za štiri in sedem pravil in s pripadnostnimi funkcijami z različnimi širinami. Vidimo, da so rezultati veliko boljši kot pri večplastnem perceptronu. Manjša širina pripadnostnih funkcij je dobra izbira v lokalem smislu, slabo pa vpliva samo na gladkost aproksimacije.

12.4 Povzetek

Mehki modeli omogočajo vnos kvalitativnega znanja v obliki pravil. Ločimo lahko tri osnovne mehke modele. Linguistični mehki modeli so zelo transparentni, saj so tako vhodne spremenljivke kot tudi izhodne zapisane z mehkimi množicami. V nasprotju z posebnimi in mehkimi modeli Takagi-Sugeno, kjer je izhod zapisan z enačbo. V primeru posebnih modelov je to samo konstantna vrednost, v primeru mehkega modela Takagi-Sugeno pa je posledični del običajno linearna kombinacija vhodnih spremenljivk kar pa ni nujno, saj so izhodne spremenljivke v linearji kombinaciji polynomia neodvisne. V tabeli 12.1 je primerjava med njihovimi lastnostmi.

Najpomembnejši lastnosti mehkih modelov sta:

- možnost vnosa apriori znanja pred in med identifikacijo;
- interpretacija modela, ki ga dobimo z identifikacijo.



Slika 12.13. Aproksimacija funkcije $y = 1/(u + 0.1)$ (a, b) in $y = \sin(2\pi u) + 2u$ (c, d) z nevro-mehkim modelom s štirimi (a, c) ali sedmimi (b, d) pravili. Standardne deviacije pripradnostnih funkcij pred normiranjem so enake 0.1 (a, b) in 0.2 (c, d).

Tabela 12.1. Primerjava med Mamdanijevim, posebnim mehkim modelom in mehkim modelom Takagi-Sugeno.

Lastnosti	Mamdanijev	Nespecifični	Takagi-Sugeno
Interpolacija	0	+	0
Ekstrapolacija	+	+	++
Lokalnost	0	+	+
Natančnost	-	0	++
Gladkost	+	+	0
Občutljivost na šum	+	+	++
Optimizacija parametrov	--	++*/--**	++*/--**
Strukturna optimizacija	--	0	++
Sprotna adaptacija	--	+	++
Hitrost učenje	--	+*/--**	++*/--**
Hitrost izračuna modela	--	-	-
Problem dimenzij	-	-	0
Interpretacija	++	+	0
Vključevanje omejitev	0	0	0
Uporba apriori znanja	+	0	0
Uporaba	+	++	0

* = linearna optimizacija, ** = nonlinearna optimizacija,
 ++ / -- = zelo pozitivno / zelonegativno.

13. Lokalno linearni nevro-mehki modeli

V tem poglavju bodo predstavljeni lokalno linearni nevro-mehki modeli, ki se drugače imenujejo tudi mehki modeli Takagi-Sugeno. Predstavljeni so tudi algoritmi za identifikacijo parametrov mehkih modelov na osnovi podatkov. Glavni problem pri njihovi konstrukciji je delitev prostora vhodno-izhodnih podatkov in s tem povezani algoritmi. Ko enkrat razdelimo prostor na ustrezne podprostore, dobimo strukturo premise mehkih pravil. Ocenjevanje parametrov modela v posledičnem delu je običajno dokaj enostavno, ker je posledični del po definiciji linearen v parametrih.

13.1 Osnovna ideja

Mrežna struktura lokalno linearnih nevro-mehkih modelov je prikazana na sliki 13.1. Vsak nevron realizira en *lokalni linearni model* (LLM) in pridruženo *veljavnostno funkcijo*, ki definira področje veljavnosti lokalnega linearnega modela. Veljavnostne funkcije tvorijo enotsko particijo, normirane so tako, da je

$$\sum_{i=1}^M \Phi_i(\underline{u}) = 1 , \quad (13.1)$$

za katerikoli vhodni vektor $\underline{u} = [u_1 \ u_2 \ \dots \ u_p]^T$. To lastnost je pomembna za pravilno interpretacijo veljavnostnih funkcij $\Phi_i(\cdot)$, kar zagotavlja, da je skupni prispevek lokalnih linearnih modelov enak 1.

Izhodi lokalni linearni modeli so zaradi konstantnega člena afini modeli in ne linearni modeli. Kljub temu jih v literaturi imenujejo kar linearni. Funkcija v premisi pravila je naslednja

$$\hat{y}_i = w_{i0} + w_{i1}u_1 + w_{i2}u_2 + \dots + w_{ip}u_p , \quad (13.2)$$

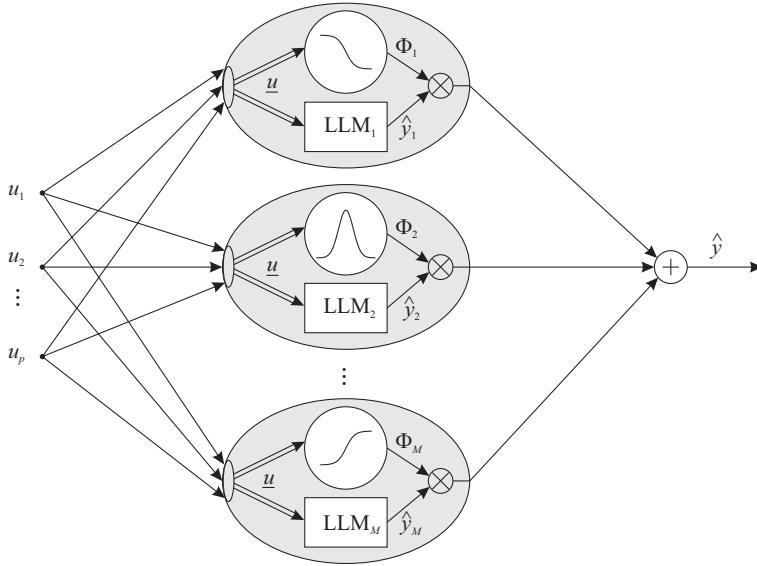
kjer w_{ij} predstavlja parametre LLM za i -ti lokalni model ali nevron.

Izhod lokalnega linearnega modela je potem enak

$$\hat{y} = \sum_{i=1}^M \underbrace{(w_{i0} + w_{i1}u_1 + w_{i2}u_2 + \dots + w_{ip}u_p)}_{\hat{y}_i} \Phi_i(\underline{u}) . \quad (13.3)$$

Na ta način je izhod mreže izračunan kot utežena vsota izhodov lokalnih linearnih modelov, kjer so veljavnostne funkcije $\Phi_i(\cdot)$ interpretirane kot uteži, ki so odvisne od delovne točke. Mreža interpolira med različnimi lokalnimi linearnimi modeli na osnovi uteži veljavnostnih funkcij. Uteži w_{ij} so linearni parametri mreže. Njihovo ocenjevanje je obravnavano v poglavju 13.2.

Veljavnostne funkcije so običajno normirane Gaussove funkcije $\mu_i(\underline{u})$, če so še osno ortogonalne, potem so veljavnostne funkcije enake



Slika 13.1. Struktura mreže za statični lokalno linearni nevro-mehki model z M nevroni in p vhodi.

$$\Phi_i(\underline{u}) = \frac{\mu_i(\underline{u})}{\sum_{j=1}^M \mu_j(\underline{u})}, \quad (13.4)$$

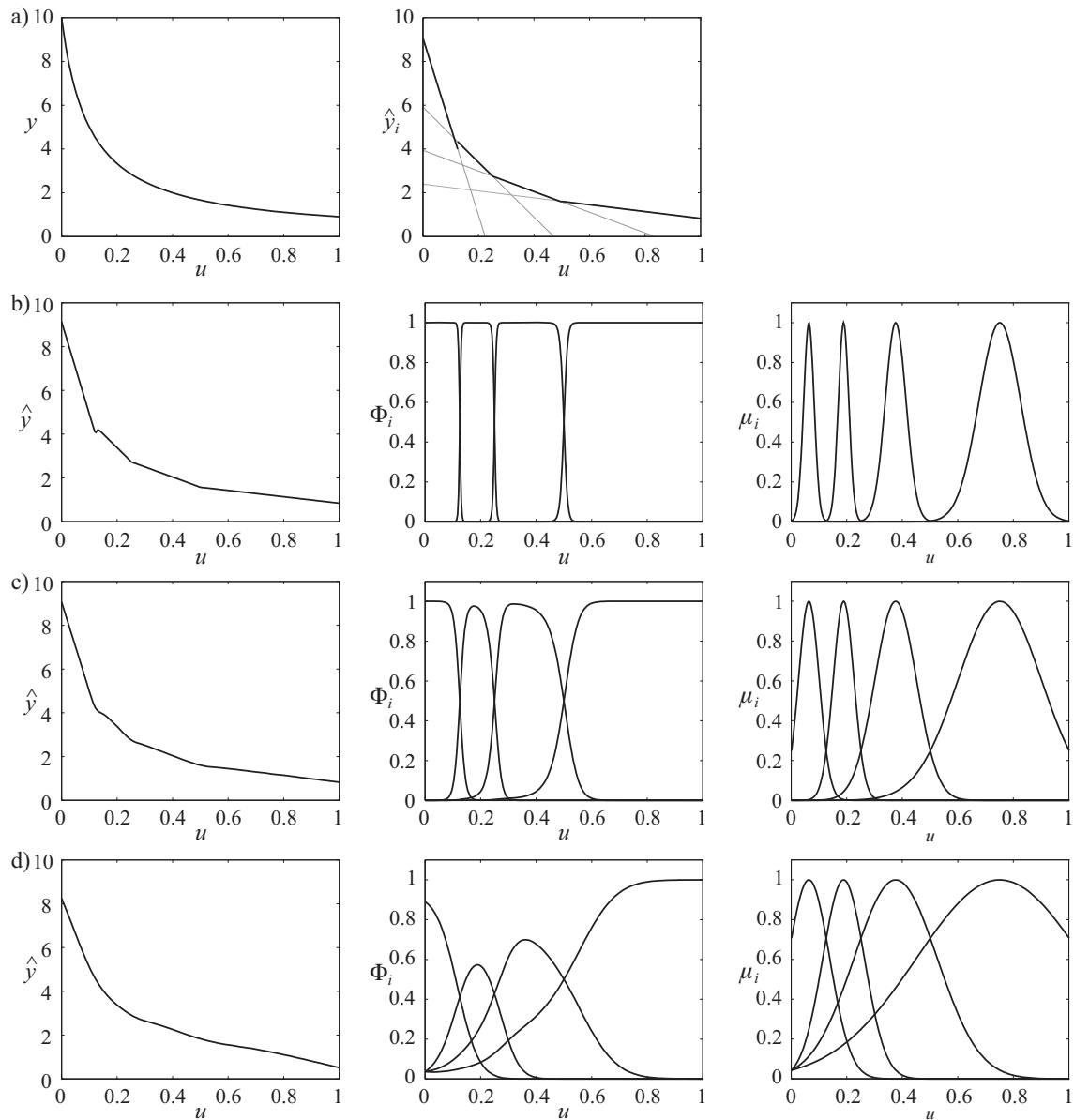
$$\begin{aligned} \mu_i(\underline{u}) &= \exp \left(-\frac{1}{2} \left(\frac{(u_1 - c_{i1})^2}{\sigma_{i1}^2} + \dots + \frac{(u_p - c_{ip})^2}{\sigma_{ip}^2} \right) \right) \\ &= \exp \left(-\frac{1}{2} \frac{(u_1 - c_{i1})^2}{\sigma_{i1}^2} \right) \cdot \dots \cdot \exp \left(-\frac{1}{2} \frac{(u_p - c_{ip})^2}{\sigma_{ip}^2} \right). \end{aligned} \quad (13.5)$$

Normirane Gaussove funkcije $\Phi_i(\cdot)$ so odvisne od središč c_{ij} in standardne deviacije ali širine σ_{ij} . Ta dva parametra sta nelinearna in ustrezata parametrom na skritem nivoju nevronskih mrež. Veljavnostne funkcije lahko včasih imenujemo tudi aktivacijske funkcije, ker definirajo aktivnost lokalnega linearnega modela, oziroma utež njegovega prispevka k izhodu celotnega modela.

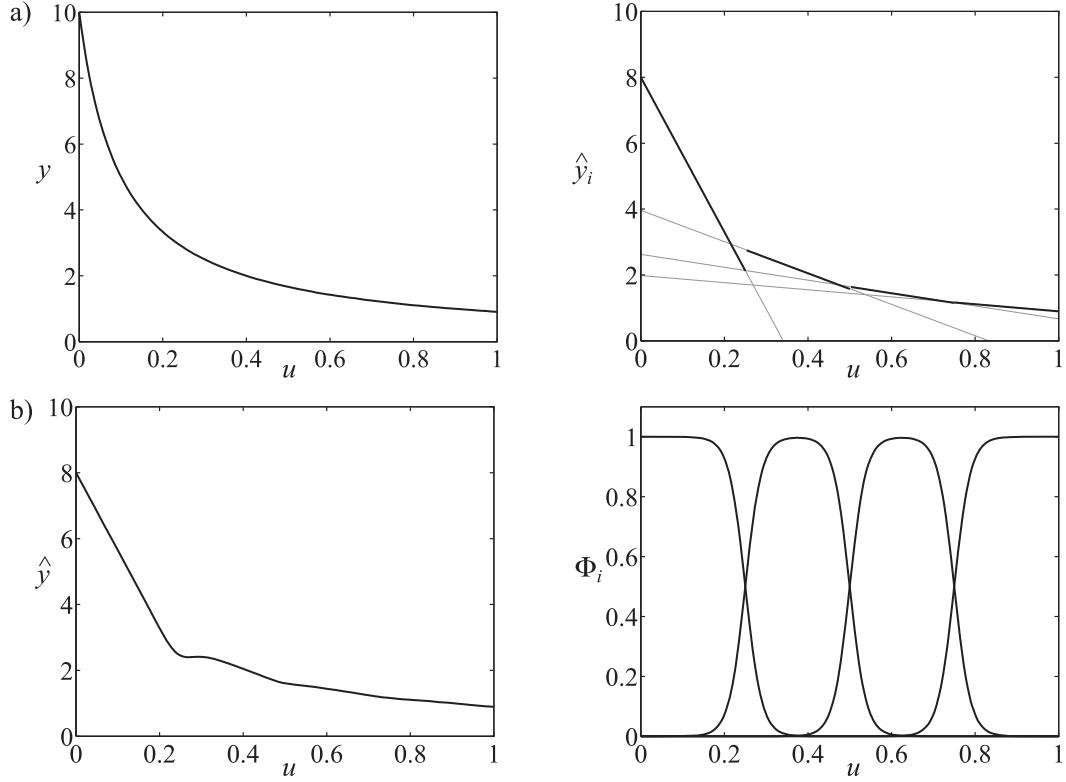
V naslednjem podpoglavlju bo predstavljena struktura lokalnega linearnega modela.

13.1.1 Prikaz lokalnih linearnih nevro-mehkih modelov

Slika 13.2 prikazuje aproksimacijo nelinearne funkcije z lokalnim linearnim modelom. Nelinearno funkcijo na sliki 13.2a (levo) aproksimiramo z mrežo štirih nevronov. Vsak nevron predstavlja lokalni linearni model, kar je prikazano na sliki 13.2a (desno). Prikazani so trije primeri z različnimi veljavnostnimi funkcijami. V vseh primerih so centri na enakih položajih, različne pa so širine funkcij. Lahko vidimo, da majhne standardne deviacije na sliki 13.2b vodijo do ostrih prelomov med lokalnimi modeli. Srednja deviacija poveča gladkost izhoda modela, kar vidimo na sliki 13.2c. Če pa je deviacija prevelika, potem njihove maksimalne vrednosti po normiranju pada; glej sliko 13.2d. Na primer, druga veljavnostna funkcija na sliki 13.2d (desno) je manjša od 0.6 za vse vhode u . Ustrezeni lokalni linearni model prispeva manj kot 60 % k končnemu izhodu modela. Večje prekrivanje veljavnostnih funkcij prispeva k nelokalnosti in slabši interpretabilnosti. Za dober model je potrebno najti pravi kompromis pri izberi središč in širin pripadnostnih funkcij. Na ta način naredimo kompromis med gladkostjo in lokalnostjo.



Slika 13.2. Aproksimacija z lokalnimi linearnimi modeli: a) funkcija, ki jo aproksimiramo (levo) in lokalni linearni modeli (desno); b) majhna, c) srednja, d) velika standardna deviacijo pripadnostnih funkcij.



Slika 13.3. Lokalni linearni model kot na sliki 13.2 c, a z ekvidistantnimi pripadnostnimi funkcijami.

Veljavnostne funkcije na sliki 13.2 so dobro pozicionirane. V območju okoli 0, kjer je velika krivina funkcije, ki jo aproksimiramo, so Φ_i bolj zgoščene kot v območju okoli 1, kjer je funkcija skoraj linearna. Za primerjavo je na sliki 13.3 model z ekvidistantno razdelitvijo funkcij. Kvaliteta aproksimacije je v tem primeru bistveno slabša kot pri modelu na sliki 13.2c, ki pa ima primerljivo gladkost. Dva lokalna modela na sliki 13.3, na intervalu med $0.5 < u < 1$, sta zelo podobna in jih lahko združimo brez poslabšanja aproksimacije. Modela na intervalu $0 < u < 0.5$ pa sta zelo različna in dovoljujeta samo grobo aproksimacijo. Glavni cilj dobrega algoritma je v optimalni postavitevi pripadnostnih funkcij.

Slika 13.4 prikazuje lokalne linearne modele za dve vhodni spremenljivki. Veljavnostne funkcije so dvodimenzionalne, lokalni linearni modeli pa so ravnine. Razširitev na višje dimenzijske je enostavna: veljavnostne funkcije so večdimenzionalne, lokalni modeli pa hiperravnine.

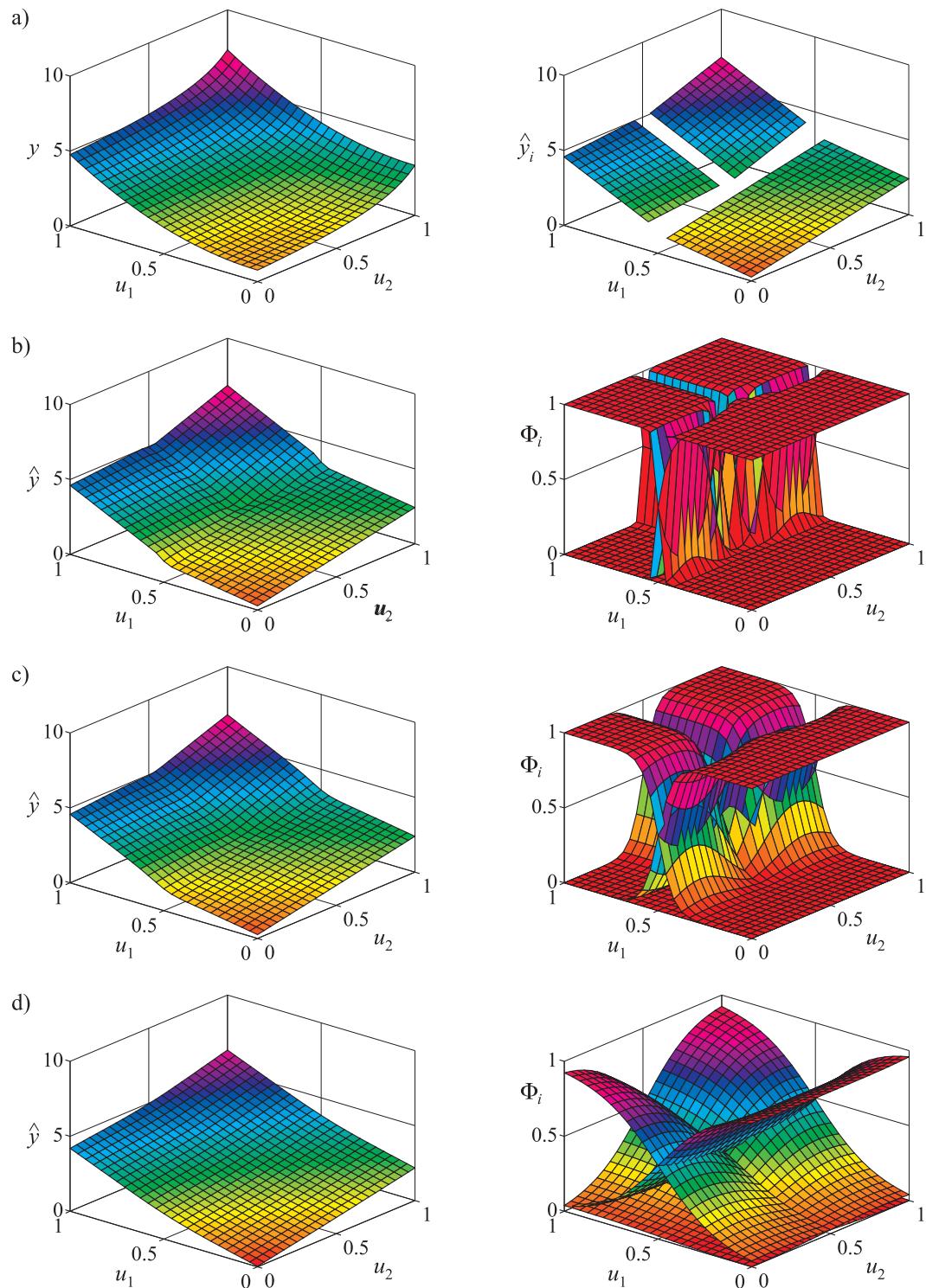
13.1.2 Interpretacija afinega člena pri lokalno linernih modelih

Člen w_{i0} v enačbi 13.3 je neodvisen od vhodnih spremenljivk in določa delovno točko ali enosmerno komponento pri meritvah v dinamičnih sistemih. Za interpretacijo zapišimo naslednjo obliko izhoda mehkaga modela

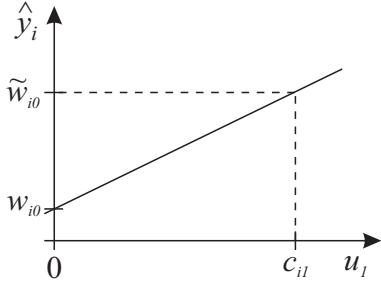
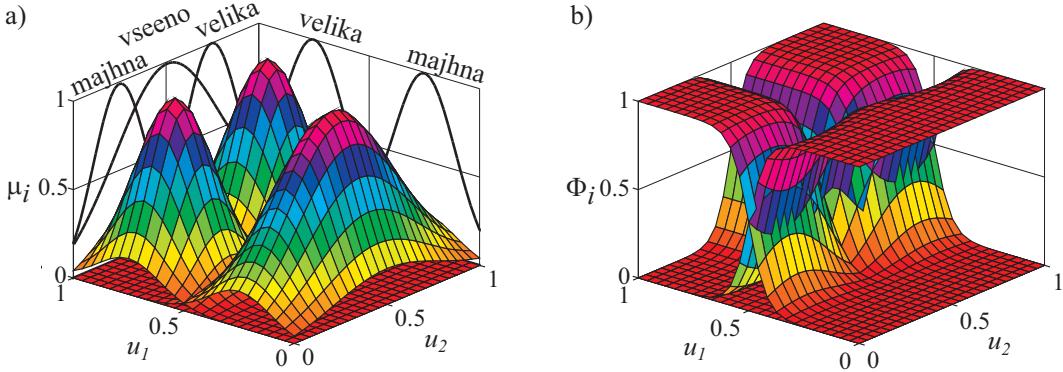
$$\hat{y} = \sum_{i=1}^M (\tilde{w}_{i0} + w_{i1}(u_1 - c_{i1}) + \dots + w_{ip}(u_p - c_{ip})) \Phi_i(\underline{u}), \quad (13.6)$$

pri čemer smo trasformirali afini člen v \tilde{w}_{i0} . Slika 13.5 prikazuje interpretacijo tega člena

$$\tilde{w}_{i0} = \hat{y}_i(\underline{u} = \underline{c}_i), \quad (13.7)$$



Slika 13.4. Učinkovanje lokalnih linearnih modelov: a) funkcija, ki jo aproksimiramo (levo) in lokalni linearni modeli (desno); b) majhna, c) srednja, d) velika standardna deviacija Gaussovin pripadnostnih funkcij.

Slika 13.5. Afini člen pri LLM w_{i0} in \tilde{w}_{i0} za model z enim vhodom u_1 .Slika 13.6. Prehod od pripadnostnih funkcij do veljavnostnih funkcij: a) s t-normo (operator produkta) enodimensijske Gaussove pripadnostne funkcije μ_{ij} sestavimo v večdimenzijsko pripadnostno funkcijo μ_i . b) z normalizacijo dobimo večdimenzijsko veljavnostno funkcijo Φ_i .

kjer je \hat{y}_i izhod i -tega lokalnega linearnega modela. Relacija med \tilde{w}_{i0} v enačbi 13.6 in w_{i0} v enačbi 13.3 je enaka

$$\tilde{w}_{i0} = w_{i0} + w_{i1}c_{i1} + \dots + w_{ip}c_{ip}. \quad (13.8)$$

Obe formulaciji v enačbah 13.3 in 13.6 sta ekvivalentni, le da je zaradi boljše pogojenosti ocena parametrov veliko bolj numerično robustna v primeru enačbe 13.6.

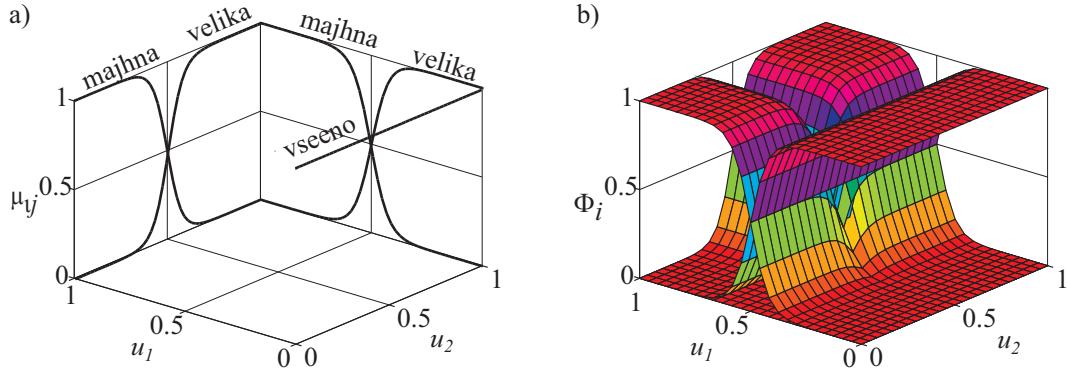
13.1.3 Interpretacija modela v obliki Takagi-Sugeno modela

Pod določenimi pogoji je mreža na sliki 13.1, zapisana z enačbo 13.3, ekvivalentna mehkemu modelu Takagi-Sugeno. Pripadnostne funkcije morajo biti Gaussove in za t-normo moramo izbrati produkt. Poleg tega pa morajo biti veljavnostne funkcije osno ortogonalne, da dovoljujejo popolno projekcijo na vhodne osi.

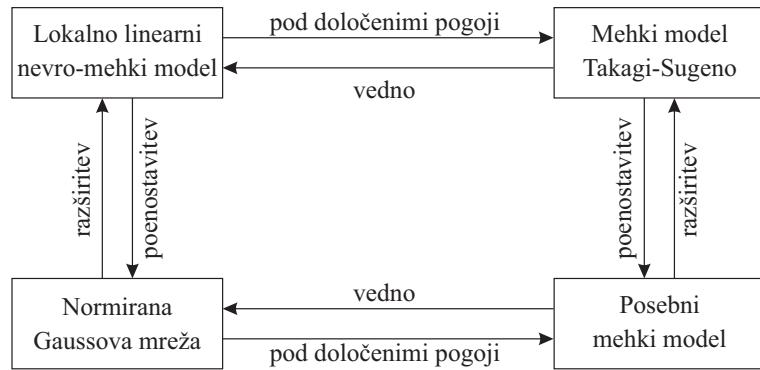
Slika 13.6 prikazuje Takagi-Sugeno mehki model s tremi pravili:

$R_1 :$	IF $u_1 = \text{majhna}$	THEN $y = w_{10} + w_{11}u_1 + w_{12}u_2 ,$
$R_2 :$	IF $u_1 = \text{velika}$ AND $u_2 = \text{majhna}$	THEN $y = w_{20} + w_{21}u_1 + w_{22}u_2 ,$
$R_3 :$	IF $u_1 = \text{velika}$ AND $u_2 = \text{velika}$	THEN $y = w_{30} + w_{31}u_1 + w_{32}u_2 .$

Baza pravil pokriva celoten vhodni prostor, čeprav ni kompletna, to pomeni, da ne vsebuje vseh možnih kombinacij pripadnostnih funkcij (majhna/majhna, majhna/velika, velika/majhna, velika/velika). Prvo pravilo vsebuje samo eno lingvistično izjavo glede spremenljivke u_1 , spremenljivka u_2 pa lahko zavzame poljubno vrednost. To lahko verificiramo z obliko ustreznih veljavnostnih funkcij na sliki 13.6b. Odvisna je od u_1 in neodvisna od u_2 .



Slika 13.7. Z normiranimi Gaussovimi pripadnostnimi funkcijami dobimo podoben (ne enak) rezultat kot je na sliki 13.6b.



Slika 13.8. Relacija med lokalnimi nevronskimi mrežami in mehkimi modeli.

Pri mehkih modelih Takagi-Sugeno je veljavnostna funkcija odvisna od središč c_{ij} in standardnih deviacij σ_{ij} pripadnostnih funkcij in od strukture pravil. Optimizacija parametrov v *strukturi premise* in *parametriih premise* je obravnavana v podpoglavlju 13.3. Parametre lokalnih linearnih modelov w_{ij} , ki jih imenujemo tudi *parametri posledičnega dela*, in njihovo estimacijo obravnavamo v podpoglavlju 13.2. Na sliki 13.8 je prikazana relacija med lokalnimi nevronskimi mrežami in mehkimi modeli.

13.2 Optimizacija parametrov lokalno linearnih modelov

Ocena parametrov lokalno linearnih modelov je linearni optimizacijski problem pod pogojem, da poznamo veljavnostne funkcije. To je v nadaljevanju tudi predpostavljeno. V naslednjem podpoglavlju pa si bomo ogledali tudi to, kako pridemo do optimalnega položaja in širine pripadnostnih funkcij in s tem tudi veljavnostnih funkcij.

Ločimo dva osnovna pristopa pri določanju parametrov lokalno linearnih modelov: *globalno* in *lokalno* ocenjevanje parametrov. Globalno ocenjevanje predstavlja enostavno uporabo metode najmanjših kvadratov. Tudi v primeru lokalnega ocenjevanja uporabimo metodo najmanjših kvadratov s tem, da pri oceni zanemarimo prekrivanje veljavnostnih funkcij in upoštevamo samo lokalno značilnost modela. [20, 86] V nadaljevanju bosta predstavljena oba pristopa. Na koncu pa bomo predstavili še metodo uteženih najmanjših kvadratov.

13.2.1 Globalno ocenejanje parametrov modela

V tem primeru so *vsi* linearni parametri modela ocenjeni hkrati z eno samo optimizacijo z metodo najmanjših kvadratov. Vektor parametrov vsebuje vseh $n = M(p + 1)$ parameterov lokalnega linearnega modela iz enačbe 13.3 z M nevroni in p vhodi:

$$\underline{w} = [w_{10} \ w_{11} \ \cdots \ w_{1p} \ w_{20} \ w_{21} \ \cdots \ w_{2p} \ \cdots \ w_{M0} \ w_{M1} \ \cdots \ w_{Mp}]^T. \quad (13.10)$$

Regresijska matrika je v tem primeru enaka \underline{X} in vsebuje vrednosti *spremenljivk regresorja* (*regresorje*) za vse učne podatke $i = 1, \dots, N$. Za N podatkov je regresijska matrika enaka

$$\underline{X} = \begin{bmatrix} \underline{X}_1^{(\text{sub})} & \underline{X}_2^{(\text{sub})} & \cdots & \underline{X}_M^{(\text{sub})} \end{bmatrix}, \quad (13.11)$$

z regresijskimi podmatrikami $\underline{X}_i^{(\text{sub})}$

$$\underline{X}_i^{(\text{sub})} = \begin{bmatrix} \Phi_i(\underline{u}(1)) & u_1(1)\Phi_i(\underline{u}(1)) & u_2(1)\Phi_i(\underline{u}(1)) & \cdots & u_p(1)\Phi_i(\underline{u}(1)) \\ \Phi_i(\underline{u}(2)) & u_1(2)\Phi_i(\underline{u}(2)) & u_2(2)\Phi_i(\underline{u}(2)) & \cdots & u_p(2)\Phi_i(\underline{u}(2)) \\ \vdots & \vdots & \vdots & & \vdots \\ \Phi_i(\underline{u}(N)) & u_1(N)\Phi_i(\underline{u}(N)) & u_2(N)\Phi_i(\underline{u}(N)) & \cdots & u_p(N)\Phi_i(\underline{u}(N)) \end{bmatrix}.$$

Izhod modela $\hat{\underline{y}}$ je enak

$$\hat{\underline{y}} = \underline{X}\underline{w}, \quad (13.12)$$

kjer je $\hat{\underline{y}} = [\hat{y}(1) \ \hat{y}(2) \ \cdots \ \hat{y}(N)]^T$. Če ocenjujemo parametre po enačbi 13.6, potem odštejemo delovno točko in moramo vse u_j v matriki \underline{X} nadomestiti z $u_j - c_{ij}$.

Z globalnim ocenejanjem parametrov, optimiramo kriterijsko funkcijo glede na parametre modelov:

$$I = \sum_{j=1}^N e^2(j) \longrightarrow \min_{\underline{w}}, \quad (13.13)$$

kjer je $e(j) = y(j) - \hat{y}(j)$ izhodni pogrešek, to je razlika med izhodom procesa $y(j)$ in izhodom modela $\hat{y}(j)$. Globalno optimalne parametre lahko izračunamo v primeru, če je $N \geq M(p + 1)$

$$\hat{\underline{w}} = (\underline{X}^T \underline{X})^{-1} \underline{X}^T \underline{y}, \quad (13.14)$$

kjer $\underline{y} = [y(1) \ y(2) \ \cdots \ y(N)]^T$ vsebuje merjene izhode procesa.

Čeprav je globalno ocenejanje parametrov modela z metodo najmanjših kvadratov zelo učinkovita pot za oceno posledičnih parametrov, je računsko zelo zahtevna, saj njena kompleksnost raste s tretjo potenco glede na število parametrov in na ta način glede na število nevronov M

$$\mathcal{O}(M^3(p+1)^3) \approx \mathcal{O}(M^3 p^3). \quad (13.15)$$

Lokalno ocenejanje parametrov modela z metodo najmanjših kvadratov, ki bo predstavljeno v nadaljevanju, je veliko bolj numerično robustno in računsko manj zahteven postopek.

13.2.2 Lokalno ocenjevanje parametrov modela

Pri tem načinu ocenjevanja parametrov obravnavamo podatke znotraj določenega področja polnoma ločeno in zanemarimo interakcije z ostalimi lokalnimi modeli. Kot smo že ugotavljali, to poveča napako pristranskosti modela. V tem primeru gre širina pripadnostne funkcije $\sigma \rightarrow 0$ in nimamo nobene interakcije med lokalnimi modeli. Izhod modela je v takem primeru odvisen samo od enega modela.

Lokalna ocena se izvede za vsakega od M lokalnih modelov posebej in vsakič ocenimo $p + 1$ parameterov določenega lokalnega modela. Vektor parametrov za vsakega od $i = 1, \dots, M$ modelov je enak

$$\underline{w}_i = [w_{i0} \ w_{i1} \ \dots \ w_{ip}]^T. \quad (13.16)$$

Ustrezna regresijska matrika pa

$$\underline{X}_i = \begin{bmatrix} 1 & u_1(1) & u_2(1) & \dots & u_p(1) \\ 1 & u_1(2) & u_2(2) & \dots & u_p(2) \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & u_1(N) & u_2(N) & \dots & u_p(N) \end{bmatrix}. \quad (13.17)$$

Če želimo parametre oceniti v obliki iz enačbe 13.6, kjer prosti člen definira delovno točko, potem moramo opazovati samo odmike od središč pripadnostnih funkcij, in vse u_j zamenjati z $u_j - c_{ij}$.

Opozorimo naj, da so vse regresijske matrike v tem primeru enake ($i = 1, \dots, M$), ker elementi matrike \underline{X}_i niso odvisni od i . Vseeno pa je dobro obdržati indeks i zaradi preglednosti. Izhod lokalnega modela določimo z \hat{y}_i

$$\hat{y}_i = \underline{X}_i \underline{w}_i, \quad (13.18)$$

kjer je $\hat{y}_i = [\hat{y}_i(1) \ \hat{y}_i(2) \ \dots \ \hat{y}_i(N)]^T$ izhod lokalnega modela in je veljaven samo na področju, kjer je veljavnostna funkcija $\Phi_i(\cdot)$ večja od določene vrednosti, ki je zadost blizu 1. To so podatki, ki so blizu središča $\Phi_i(\cdot)$ in so relevantni za oceno parametrov \underline{w}_i . Glede na to je seveda ena od možnih dobrih rešitve tudi uporaba uteženih najmanjših kvadratov, kjer je utež enaka veljavnostni funkciji

$$I_i = \sum_{j=1}^N \Phi_i(\underline{w}(j)) e^2(j) \longrightarrow \min_{\underline{w}_i}, \quad (13.19)$$

in je $e(j) = y(j) - \hat{y}(j)$ pogrešek med izhodom meritve in modelom.

Podatke utežimo z matriko dimenzije $N \times N$, ki jo imenujemo utežna matrika

$$\underline{Q}_i = \begin{bmatrix} \Phi_i(\underline{w}(1)) & 0 & \dots & 0 \\ 0 & \Phi_i(\underline{w}(2)) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \Phi_i(\underline{w}(N)) \end{bmatrix}, \quad (13.20)$$

in dobimo rešitev uteženih najmanjših kvadratov za parametre posledičnega dela i -tega lokalnega modela v primeru predoločenega sistema enačb ($N \geq p + 1$)

$$\hat{\underline{w}}_i = (\underline{X}_i^T \underline{Q}_i \underline{X}_i)^{-1} \underline{X}_i^T \underline{Q}_i \underline{y}. \quad (13.21)$$

Na ta način izračunamo ocene za vse lokalne modele ($i = 1, \dots, M$). Glavna prednost tega izračuna je v zmanjšani kompleksnosti in dobri pogojenosti regresijske matrike. Računska kompleksnost raste linearno s številom modelov.

$$\mathcal{O}(M(p+1)^3) \approx \mathcal{O}(Mp^3). \quad (13.22)$$

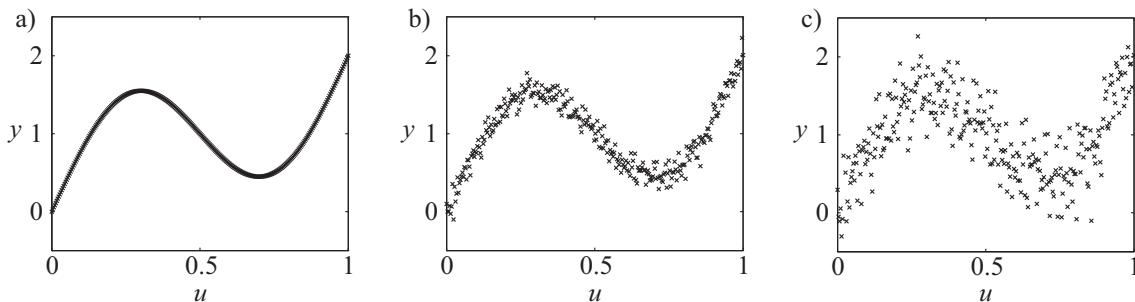
Na ta način vidimo, da je računska zahtevnost precej nižja kot v primeru globalne estimacije. Seveda pa je cena za to slabša kvaliteta modela, saj smo zanemarili interakcije med lokalnimi modeli.

13.2.3 Primerjava med globalnim in lokalnim ocenjevanjem parametrov modela

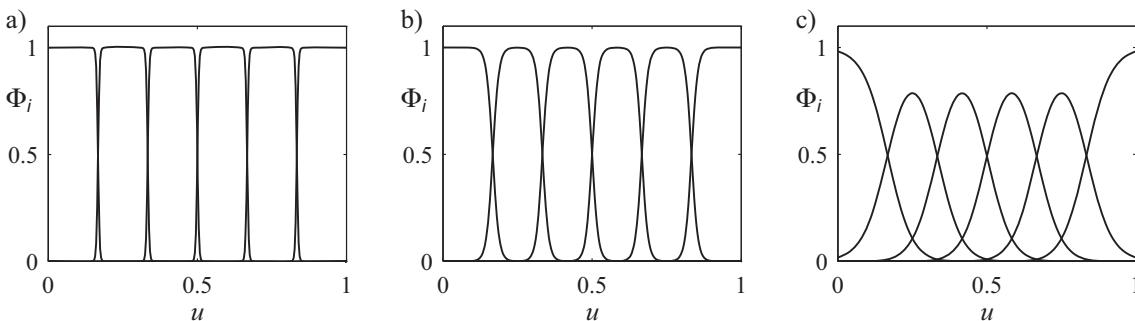
Na enostavnem primeru si bomo ogledali primerjavo med globalnim in lokalnim ocenjevanjem parametrov modela. Nelinearna funkcija na sliki 13.9a je aproksimirana z nevro-mehkim modelom s šestimi ($M = 6$) lokalimi modeli in ekvidistančno razdeljenimi Gaussovimi pripadnostnimi funkcijami. Imamo 300 učnih vzorcev. Funkciji je dodan beli šum z različnimi normalnimi distribucijami $\sigma_n = 0, 0.1, 0.3$; glej sliko 13.9a–c. Za pripadnostne funkcije smo izbrali tri različne širine $\sigma = 0.125/M, 0.25/M, 0.5/M$; glej sliko 13.10a–c.

Tabla 13.1 prikazuje rezultate, ki jih dobimo z globalnim in lokalnim ocenjevanjem parametrov modela. Globalna in lokalna ocena dasta enake rezultate, če se veljavnostne funkcije malo prekrivajo; glej sliko 13.11. Če se prekrivanje povečuje, pa se razlika povečuje.

Za majhne vrednosti šuma je globalna estimacija bistveno primernejša kot lokalna, kot to vidimo na sliki 13.12. Ta efekt je intuitivno enostavno razložiti, saj je interakcija med lokalnimi



Slika 13.9. Učni podatki za funkcijo $y = \sin(2\pi u) + 2u$ z različnimi vrednostmi šuma: a) $\sigma_n = 0$, b) $\sigma_n = 0.1$, c) $\sigma_n = 0.3$.



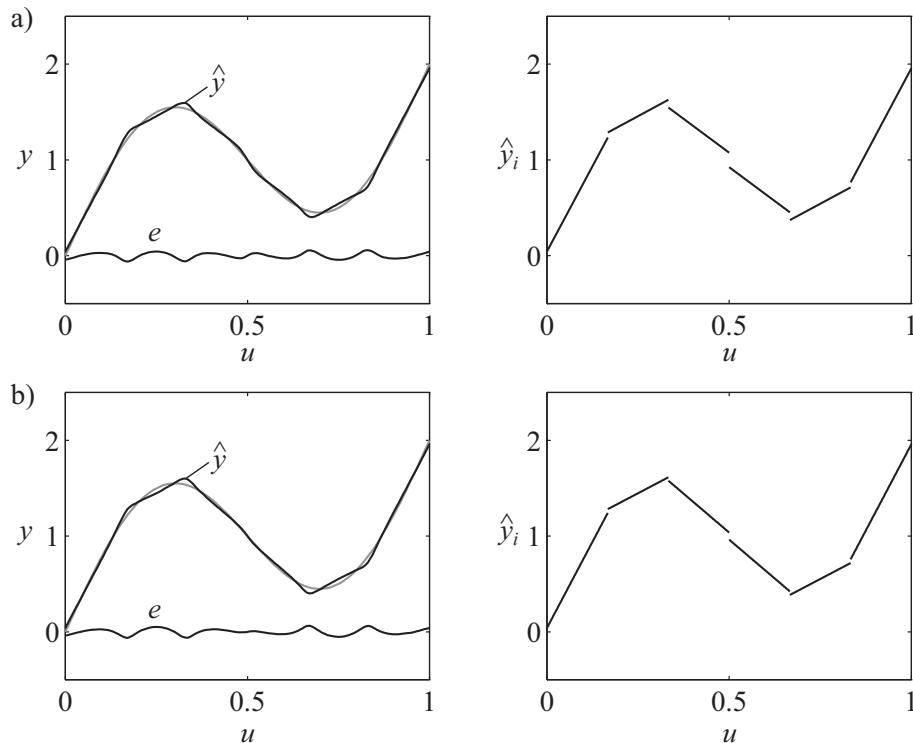
Slika 13.10. Veljavnostne funkcije za nevro-mehki model z različnimi prekrivanji: a) $\sigma = 0.125/M$, b) $\sigma = 0.25/M$, c) $\sigma = 0.5/M$ pri $M = 6$.

Tabela 13.1. Primerjava med globalno in lokalno estimacijo za različne stopnje šuma in različne širine pripadnostnih funkcij.

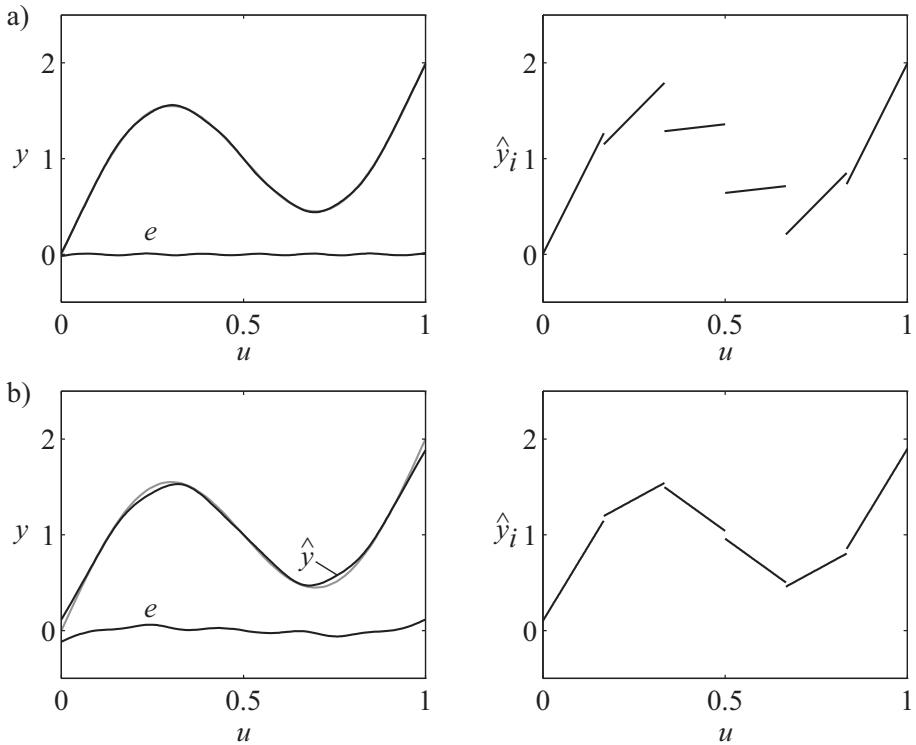
100 · RMSE: globalno / lokalno	$\sigma = 0.125/M$	$\sigma = 0.25/M$	$\sigma = 0.5/M$
šum $\sigma_n = 0.0$	2.84 / 2.85	2.92 / 3.06	0.67 / 3.81
šum $\sigma_n = 0.1$	3.35 / 3.33	3.39 / 3.43	2.00 / 3.69
šum $\sigma_n = 0.3$	6.04 / 5.92	5.94 / 5.69	5.70 / 4.46
nominalno število parametrov	12 / 12	12 / 12	12 / 12

RMSE = koren srednje kvadratične napake (root mean square error)

Srednja kvadratična napaka se izračuna po $\frac{1}{N} \sum_{i=1}^N e^2(i)$, če je napaka $e(i) = y(i) - \hat{y}(i)$ izračunana na učnih podatkih.



Slika 13.11. Izhod procesa in modela (levo) in lokalni linearni modeli (desno) za a) globalno in b) lokalno estimacijo pri $\sigma_n = 0$ in $\sigma = 0.125/M$.



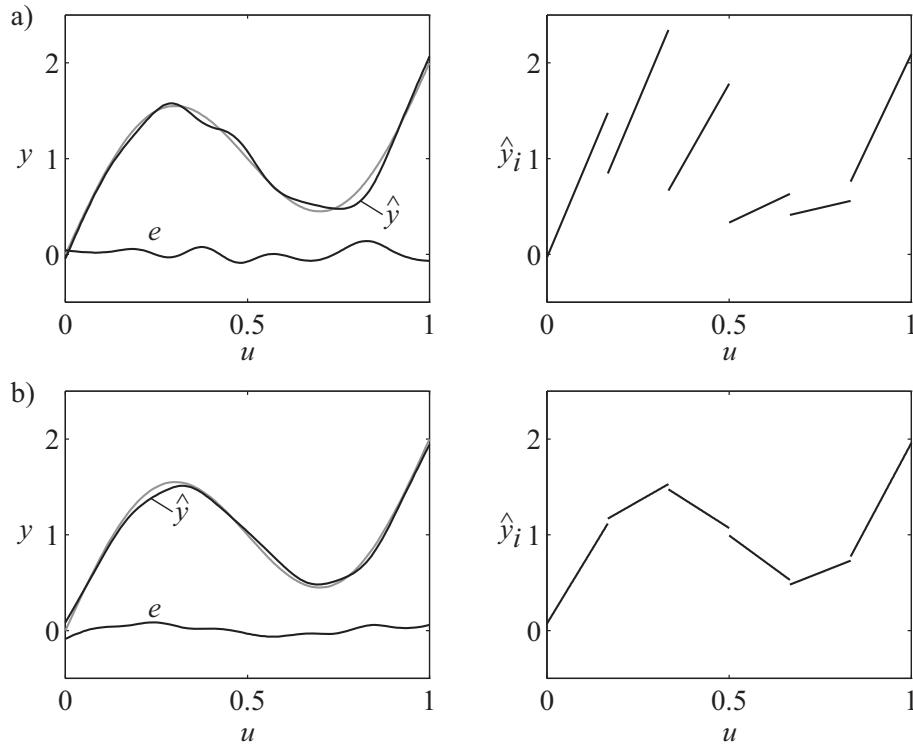
Slika 13.12. Izhod procesa in modela (levo) in lokalni linearni modeli (desno) za a) globalno in b) lokalno estimacijo pri $\sigma_n = 0$ in $\sigma = 0.5/M$.

modeli, ki jo upoštevamo z globalno estimacijo, večja pri večjih σ . Lokalna estimacija zanemari te interakcije in je zato njena aproksimacija slabša. Ne glede na to, da lokalna estimacija rezultira v bistveno slabši aproksimaciji, ima ta pristop bistveno prednost pri interpretaciji modela. Lokalni linearni modeli dobro predstavljajo lokalno obnašanje funkcije. Z analizo parametrov lokalno linearnih modelov si to lahko bolje predstavimo. V primeru lokalne estimacije lokalni modeli dobro opisujejo funkcijo na določenem območju, v primeru globalne estimacije pa dobimo dobro aproksimacijsko napako, lokalni modeli pa ne dajo nobene informacije o lokalnem obnašanju funkcije, ki jo aproksimiramo. Na primer tretji in četrti lokalni model imata pozitiven odvod, čeprav ima funkcija visok negativen odvod. Samo v interakciji s sosednjim lokalnim modelom dobimo pravo informacijo.

Glede na posamezne aplikacije lahko srečamo veliko različnih kriterijev, ki so poleg aproksimacije še pomembni pri odločitvi, katero tehniko ocenjevana izberemo. Pogosto je interpretabilnost lokalnih linearnih modelov zelo pomemben kriterij. V tem smislu je lokalno ocenjevanje v veliki prednosti.

Izkaže se, da so lastnosti lokalnega ocenjevanja večkrat prednost. V nadaljevanju so naštete prednosti:

- *Hitro učenje.*
- *Interpretacija.*
- *Možnost sprotnega učenja z rekurzivno identifikacijo.*
- *Visoka fleksibilnost.* Zaradi manjšega števila parametrov lahko določene lokalne model parametriramo linearno druge pa nelinearno in njihove parametre ločeno ocenjujemo. Lahko pa izbiramo tudi popolnoma različne strukture.



Slika 13.13. Izhod procesa in modela (levo) in lokalni linearni modeli (desno) za a) globalno in b) lokalno estimacijo pri $\sigma_n = 0.3$ in $\sigma = 0.5/M$.

13.3 Struktturna optimizacija premise pravil

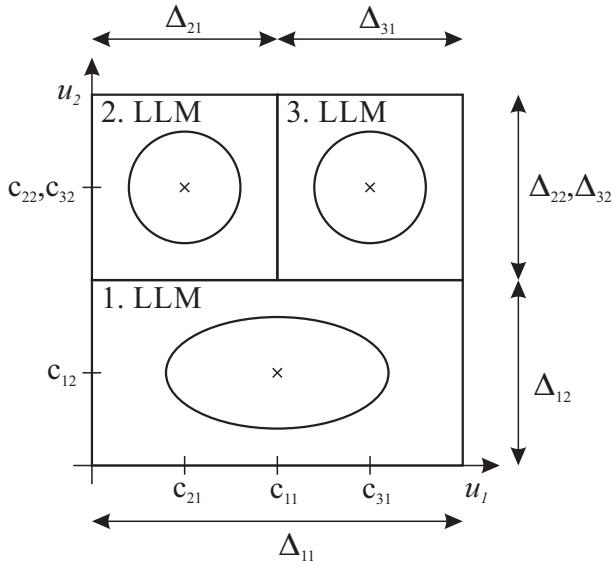
Linearne parametre na posledični strani pravil smo ocenjevali ob predpostavki, da smo predhodno določili pripadnostne funkcije in posledično tudi veljavnostne funkcije. V osnovi lahko ločimo dve strategiji za določevanje pripadnostnih funkcij: nastavljanje na osnovi apriornega znanja in na osnovi podatkov. Lahko pa ti dve starategiji tudi kombiniramo.

Število pripadnostnih funkcij in njihovi parametri, središča c_{ij} in standardne deviacije σ_{ij} , definirajo razdelitev vhodnega prostora. Na primer, tri pripadnostne funkcije na sliki 13.6 razdelijo vhodni prostor na tri pravokotna področja na sliki 13.14. Razširitev na višje dimenzije vhodnega prostora je enostavna; pravokotniki v tem primeru postanejo hiperpravokotniki. Če normiramo Gaussove pripadnostne funkcije, potem njihova središča c_{ij} določajo središča pravokotnikov, standardne deviacije σ_{ij} pa določajo velikost stranic v vsaki dimenziji.

Razdeljevanje vhodnega prostora in iskanje optimalnih rojev in njihovih pripadnostnih funkcije predstavlja že način iskanja strukture v premisi pravil za vhodne spremenljivke; glej sliko 13.1.3. V tem primeru z izbiro števila rojev, njihovih položajev in oblik že določimo število pravil, strukturo premise in njene pripadnostne funkcije.

Identifikacija parametrov pripadnostnih funkcij je problem nelinearne optimizacije. Obstajajo naslednje osnovne strategije za določitev teh parametrov:

- *Razdelitev mreže.* Število pripadnostnih funkcij na vhodu je določeno s predznanjem. Pri tem pristopu nastane problem dimenzij. Mrežo lahko uporabimo tam, kjer je zadost podatkov, drugod pa ne. Pristop je omejen na probleme majhnih dimenzij.
- *Rojenje vhodnega prostora.* Pripadnostne funkcije dobimo z rojenjem vhodnega prostora in so postavljene glede na distribucijo podatkov. Pristop ne upošteva narave procesa.



Slika 13.14. Obstaja enostavna relacija med particijami vhodnega prostora in veljavnostnimi funkcijami Φ_i . Kogi in elipse predstavljajo konture večdimensionalne pripadnostne funkcije μ_i (t.j., pred normiranjem); glej sliko 13.6 .

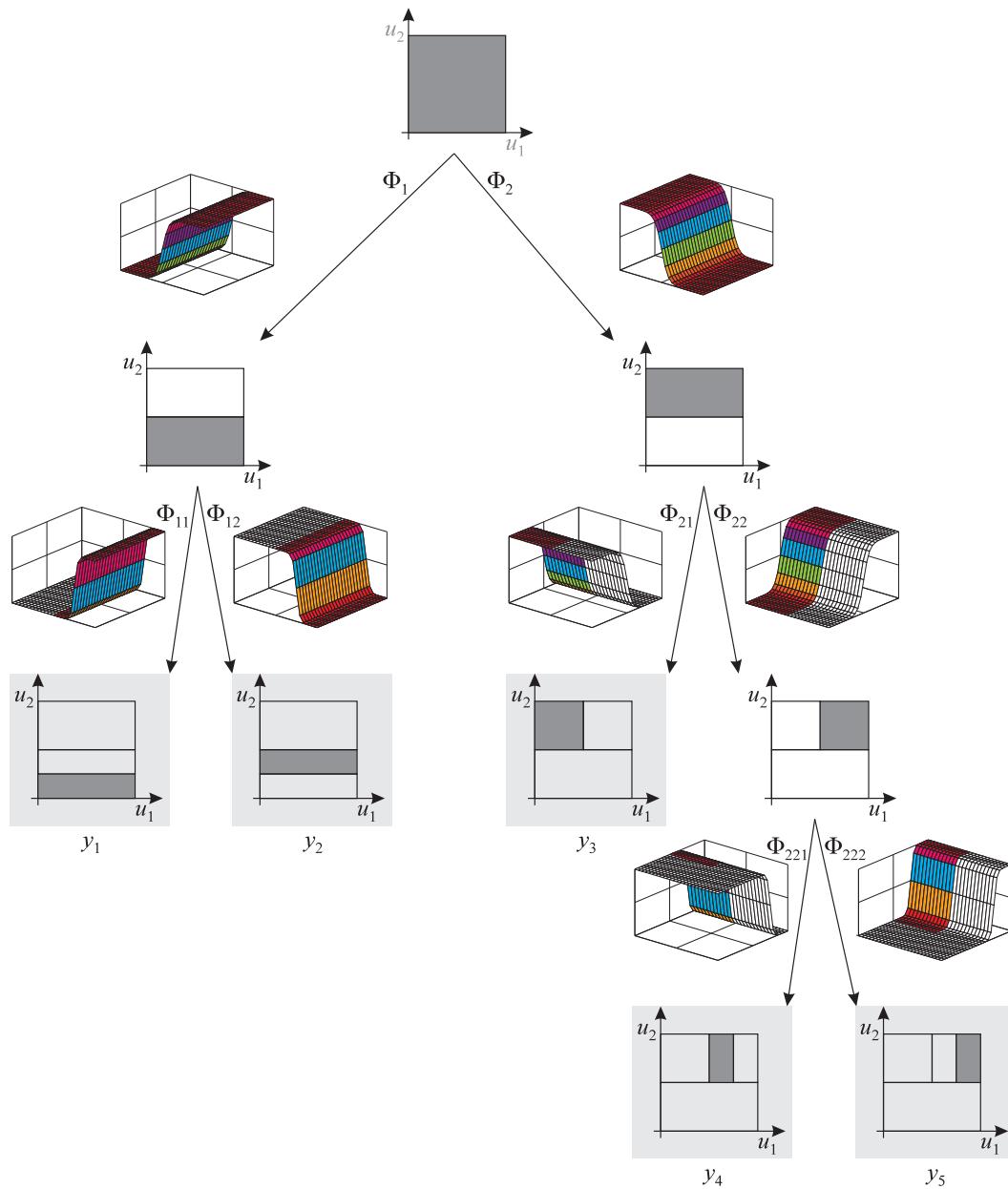
- *Nelinearna lokalna optimizacija.* Originalno so pripadnostne funkcije in parametri posledičnega dela pravil optimirani simulatano. Običajno se vseeno stvari rešuje v dveh korakih: z nelinearno lokalno optimizacijo nastavimo parametre v premisi pravil in z globalnimi najmanjšimi kvadri ocenimo posledične parametre. Tak pristop uporablja algoritem ANFIS (adaptive neuro-fuzzy inference system) [56, 57, 58]. Pristop je računsko zahteven, daje pa zelo dobre rezultate. Algoritem optimira zelo veliko število parametrov in pogosto naletimo na problem preparametrizacije.
- *Genetski algoritmi.* So eden od možnih načinov optimizacije strukture v premisi, čeprav so računsko zelo zahtevni [8, 112].
- *Projekcija rojev.* Zelo pogosto se v optimizaciji strukture mehkih modelov uporablja mehko rojenje Gustafson-Kessel, s pomočjo katerega lahko najdemos roje, ki so hiperelipsoidnih oblik, kar je običajna oblika, če se lotevamo aproksimacijskih problemov. Rojem so pridružne njihove kovariančne matrike. Če kovarične matrike razstavimo na njihove glavne komponente, lahko pod določenimi pogoji dobimo tudi model podatkov v obliki različnih hiperravin [4, 6, 5, 40, 71, 88, 94, 119].
- *Hevristično konstruiranje.* Najbolj razširjeni algoritmi za konstrukcijo so algoritmi, ki povečujejo kompleksnost modela z iteracijami učenja. Začno z enim pravilom, potem pa znižujejo resolucijo vhodnega prostora in dodajo pravila na področja, kjer modeli ne ustrezajo učni množici. Metode lahko rezultirajo v zelo fleksibilne strukture, ki lahko dosežejo skoraj poljubno razdelitev prostora [86, 87], ali pa malo manj fleksibilne razdelitve, ki so osno ortogonalne in neortogonalne [30, 59, 60, 62, 108, 109, 110].

V nadaljevnu si bomo zelo na kratko ogledali algoritem lokalnega linearne drevesa (LOLIMOT). To je osno ortogonalni, hevristični, inkrementalni algoritem za konstrukcijo modela.

13.3.1 Algoritem lokalnega linearne modela drevesa (LOLIMOT)

LOLIMOT je inkrementalni algoritem z drevesno strukturo, ki razdeljuje vhodni prostor z osno ortogonalno delitvijo. V vsaki iteraciji algoritem doda k celotni strukturi modela eno novo pra-

vilo ali en nov lokalno linearni model (LLM). Novi model se doda na tistem delu vhodnega prostora, kjer je model, ki ima najslabšo kriterijsko funkcijo. Zaradi tega algoritem spada v razred *inkrementalnih* ali *rastočih* algoritmov. Vpeljuje hevristično iskanje strukture v premise pravil. V vsaki iteraciji se izračuna pripadnostna funkcija, ki ustreza delu vhodnega prostora, ki smo ga na novo dobili z razdeljevanjem podprostorov. Hkrati pa ocenimo tudi parametre posledičnega dela pripadajočega lokalnega linearnega modela.



Slika 13.15. Delovanje algoritma LOLIMOT v prvih nekaj iteracijah za dvodimensionalni problem ($p = 2$).

13.4 Povzetek

Predstavili smo lokalno linearni nevro-mehki model in načine ocenjevanja posledičnih parametrov. Pregledali smo prednosti in slabosti globalne in lokalne ocene parametrov in na kratko preleteli mogoče načine za strukturno optimizacijo premise pravil v lokalno linearnih nevromehkih modelih.

Del IV

Dinamični modeli

14. Identifikacija linearnih dinamičnih sistemov

Ime *identifikacija sistemov* se običajno nanaša na identifikacijo linearnih *dinamičnih* sistemov. Razumevanje osnovnih konceptov in terminologije identifikacije linearnih sistemov je nujno potrebno za razumevanje nelinearne identifikacije, ker najbolj pogosto uporabljeni metodi temeljijo na lokalnih linearnih linearnih modelih. Bolj natančna študija identifikacije linearnih sistemov pa je podana v [11, 31, 53, 54, 61, 78, 107].

V tem poglavju si bomo ogledali kratek pregled linearne identifikacije. V poglavju 14.3 je podana terminologija, ki jo uporabljamo v primeru različnih struktur, razlaga osnovnih konceptov prediktorjev in metod za oceno linearnih modelov na osnovi podatkov. V poglavju 14.4 sta podana kratek pregled modelov časovnih vrst in klasifikacija linearnih modelov v dva razreda: razred modelov s povratno zanko v poglavju 14.5 in razred modelov brez povratne zanke v poglavju 14.6.

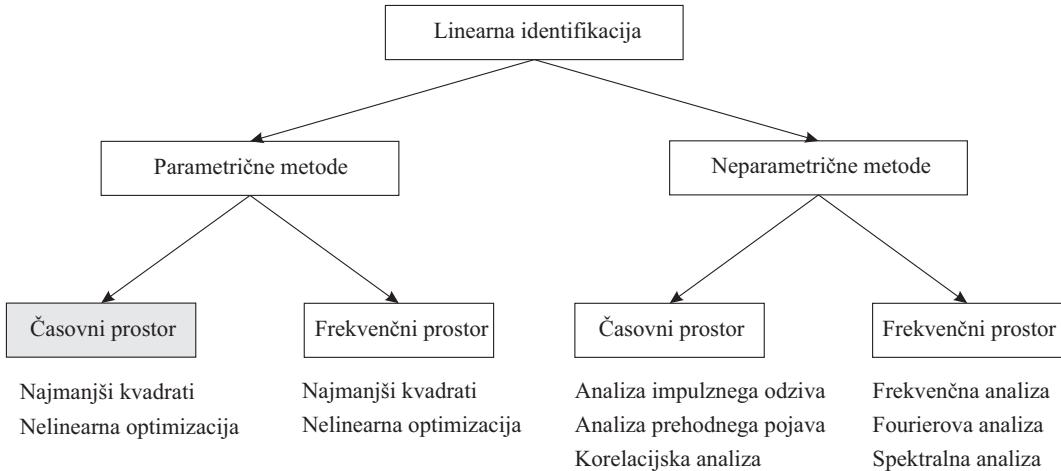
14.1 Pregled metod identifikacije linearnih sistemov

Slika 14.1 prikazuje pregled metod identifikacije linearnih sistemov. Pri tem moramo ločiti med modelom in tipom metode, ki jo uporabimo. Modele lahko delimo na parametrične in neparametrične:

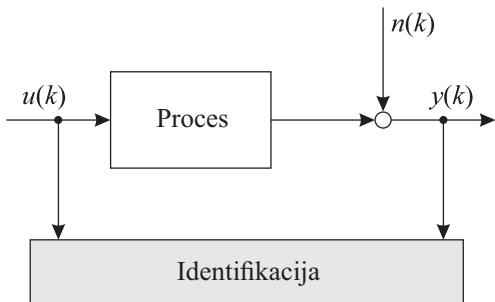
- *Parametrični model* s končnim številom parametrov eksaktно opiše obnašanje procesa. Tipičen primer je model v obliki diferenčne ali diferencialne enačbe. Pogosto imajo parametri tudi fizikalni pomen.
- *Neparametrični model* navadno zahteva neskončno število parametrov za eksakten opis procesa. Tipičen primer je impulzni odziv procesa.

Naprej lahko razdelimo metode na:

- *Parametrične metode*, ki določijo relativno majhno število parametrov modela. Parametre običajno dobimo z optimizacijo določenega kriterija. Tipičen primer je ocenjevanje parametrov z linearno regresijo. Parametrične modele lahko uporabimo tudi za aproksimacijo neparametričnih modelov z zmanjšanim, končnim številom parametrov. Tipičen je primer impulznega odziva, ki ga aproksimiramo s končnim impulznim odzivom.
- *Neparametrične metode* so bolj fleksibilne kot parametrične metode. Tipičen primer je opis s Fourierovo funkcijo, ki je ne moremo zapisati s končnim številom parametrov. Čeprav se v praktičnih implementacijah uporabi končno število parametrov pri opisu teh funkcij, je to število vedno veliko, ne glede na strukturo modela. Število parametrov je največkrat odvisno od števila podatkov ali kvantizacije.



Slika 14.1. Pregled metod identifikacije linearnih sistemov. V tem poglavju obravnavamo samo metode v črno oseňenih blokih. Poudarimo, da govorimo o metodah identifikacije. Neparametrični model impulznega odziva lahko dobimo tudi s parametrično metodo, če predpostavimo končno število parametrov modela.



Slika 14.2. Vhod in izhod z motnjo na izhodu procesa, ki ga merimo za uporabo pri identifikaciji.

14.2 Signali vzbujanja

Vhodni signal procesa $u(k)$ ima zelo pomembno vlogo pri identifikaciji sistema. To je edini način za pridobitev informacije o sistemu. V realnih primerih je vhodni signali vedno omejeni, navzdol z vrednostjo minimuma u_{\min} in navzgor z maksimumom u_{\max} . Hkrati pa je vedno omejen tudi čas izvajanja meritev. Razen teh omejitev, ki so lastne vsem procesom, lahko načrtovalec v idealnem primeru prosto načrtuje vhodni signal. To se lahko zgodi samo takrat, ko proces ni v normalnem delovanju. Če je naprava v normalnem načinu delovanja, so navadno možna le majhna odstopnaja od normalnega obratovanja.

Slika 14.2 kaže proces, ko vse motnje, ki vplivajo na izhod procesa, združimo v šum $n(k)$. Vse motnje, ki dejansko vplivajo na vhod ali na določena stanja procesa, se lahko transformirajo na izhod procesa tako, da določen signal motnje filtriramo s filtrom ustrezne frekvenčne karakteristike. Na šum ne moremo vplivati in je vhodni signal v proces edina prostostna stopnja za načrtovalca. Z vhodnim signalom lahko načrtovalec spreminja razmerje med koristnim signalom in šumom, ki vpliva na kvaliteto ocene parametrov. Zato je smiselno, da proces vzbujamo med skrajnjima mejama u_{\min} in u_{\max} .

Frekvenčni spekter vhodnega signala določa, pri kateri frekvenci bo vhodni signal vzbujal proces. Ocena modela je na ta način boljša v frekvenčnem področju, ki ga signal močneje vzbuja. Če je signal sinusne oblike, potem vzbujamo samo eno frekvenco in lahko dobimo dober model samo pri tej frekvenci. Iz tega sledi, da je za načrtovanje vzbujanja odločilnega pomena upo-

raba modela. Če je pomembno področje nizkih frekvenc, potem bomo sistem vzbujali z nizkimi frekvencami, če pa naj deluje pri natančno določeni frekvenci, potem je smiselno izbrati signal sinusne oblike s to frekvenco. Če modeliramo za namene vodenja, je pomembno imeti dobro oceno modela procesa okoli Nyquistove frekvence (to je pri faznem zamiku -180°). Pri splošni uporabi modela je dobra izbira psevdonaključni beli šum, ki vzbuja vse frekvence enakomerno. Pri tem je potrebno poudariti, da visoke frekvence nimajo tako pomembne vloge, še posebno, če ustrezeno izberemo čas vzorčenja T_0 . Čeprav je zelo naravno, da izberemo vzorčenje kolikor je mogoče hitro, pa se v praksi navadno izbere čas vzorčenja tako, da je ta relativen glede na čas izravnave. Običajno vzamemo čas vzorčenja enak $T_0 = T_s/12$ do $T_0 = T_s/10$, kjer je T_s čas izravnave.

Primer 14.2.1. Vzbujalni vhodni signali

Naslednja slika kaže nekaj tipičnih vzbujalnih signalov in ustrezne izhodne signale za sistem prvega reda z ojačenjem $K = 1$ in časovno konstanto $T = 8$ s, ki ga vzorčimo s časom vzorčenja $T_0 = 1$ s in dobimo diferenčno enačbo ($a_1 = \exp(-T_0/T)$, $b_1 = K(1 - a_1)$)

$$y(k) = 0.1175u(k-1) + 0.8825y(k-1). \quad (14.1)$$

Proces vzbujamo z vhodnimi signali na sliki 14.3–14.7, in vzamemo 100 meritev. Te vzorce vzamemo in identificiramo ARX model prvega reda; glej podpoglavlje 14.3.1. Procesu dodamo beli šum z varianco 0.01. Šumni filter ima imenovalec, ki je enak imenovalcu procesa $1/A$, da se prilagodimo predpostavkam za primer ARX modela. Ker sta struktura modela in struktura procesa enaki, je pristranskost modela enaka nič (glej poglavje 8.2) in je celotna napaka samo zaradi šuma.

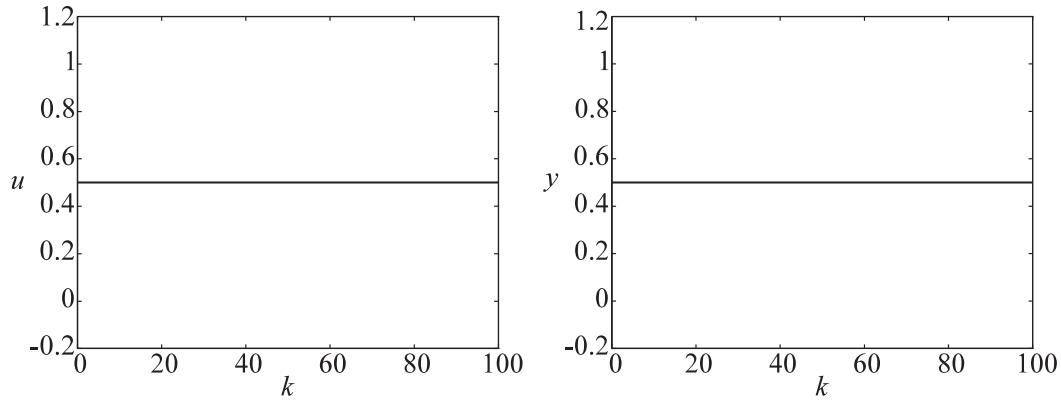
Rezultati identifikacije v primeru različnih vzbujanj so zbrani v tabeli 14.1. Primerjava je sledеča:

- *Konstanta.* Primerna za identifikacijo samo enega parametra pri frekvenci nič. V tem primeru smo identificirali enosmerno ojačenje K , ki je v diskretnem času dano z $K = b_1/(1 - a_1)$. Signal ni primeren za identifikacijo dinamičnih sistemov, ker ne vzbuja dinamike. Parametrov b_1 in a_1 ne moremo oceniti vsakega posebej, dobro ocenimo samo razmerje $b_1/(1 - a_1)$.
- *Impulz.* Ni primeren za identifikacijo, ker je posebno ocena ojačenja zelo nenatančna.
- *Stopnica.* Zelo primerna za identifikacijo. Ocenimo dinamiko in zelo dobro ocenimo nizko frekvenčno obnašanje procesa, ker so nizke frekvence prevladujoče.
- *Pravokotni pulzi.* Zelo primerni za identifikacijo. Hkrati lahko s spremenjanjem frekvence pulzov vzbujamo določen frekvenčni pas bolj močno. Za signal na sliki 14.6 je ocena časovne konstante zelo natančna.
- *PRBS (psevdonaključni binarni signal).* Zelo primeren za identifikacijo, ker posnema beli šum v diskretnem času z determinističnim signalom in zaradi tega enakomerno vzbuja širok pas frekvenc.

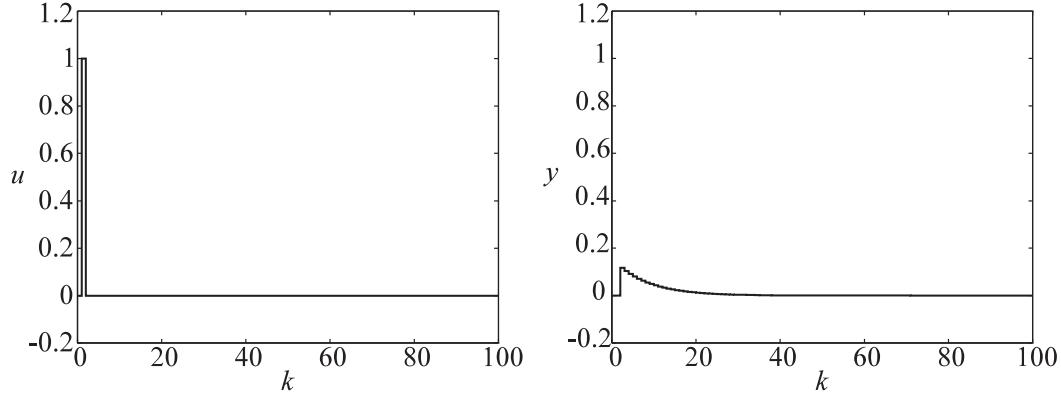
14.3 Splošne strukture modelov

V tem podpoglavlju si bomo ogledali splošno linearino strukturo modela iz katere lahko s poenostavtvami pridemo do vseh ostalih struktur. Splošnega modela običajno ne uporabljamo.

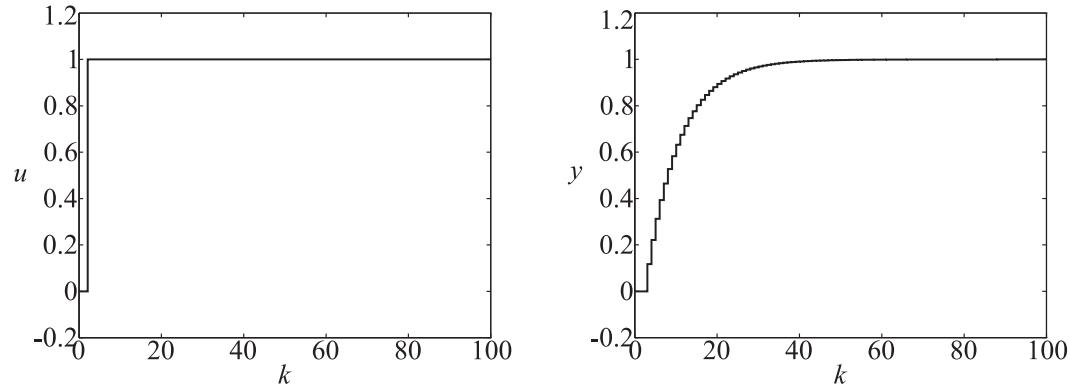
Izhod linearnega sistema $y(k)$ v časovnem trenutku k lahko dobimo s filtriranjem vhodnega signala $u(k)$ skozi prenosno funkcijo procesa $G(q)$ (q predstavlja operator premika, to je $q^{-1}x(k) = x(k-1)$, in je časovni ekvivalent operatorju $z = e^{j\omega}$ v frekvenčnem prostoru>):



Slika 14.3. Vzbujanje s konstantnim signalom (levo) in izhod procesa brez motnje (desno).



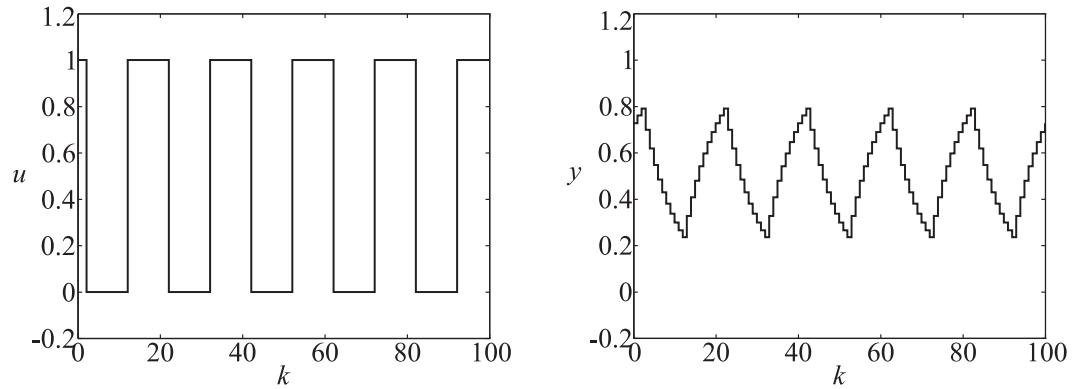
Slika 14.4. Vzbujanje z impulznim signalom (levo) in izhod procesa brez motnje (desno).



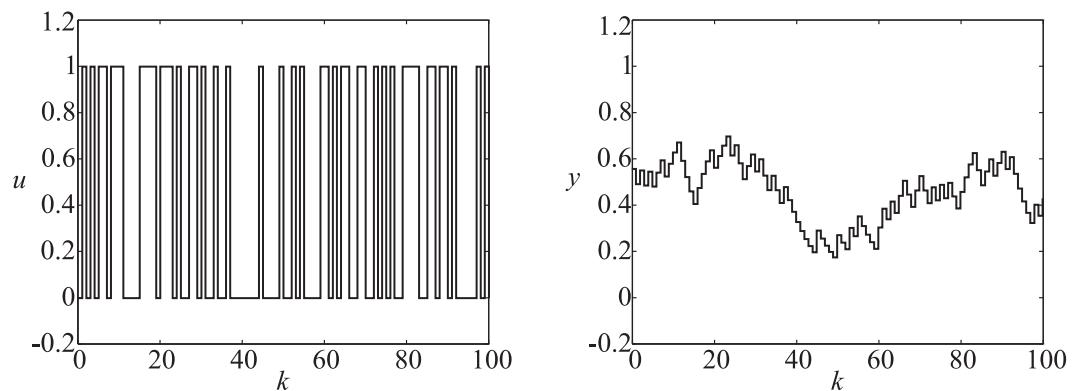
Slika 14.5. Vzbujanje s stopničastim signalom (levo) in izhod procesa brez motnje (desno).

Tabela 14.1. Identifikacija procesa pri različnih signalih vzbujanja.

Vhodni signal	b_1	a_1	K	T [s]
Konstanta	0.2620	-0.7392	1.0048	3.3098
Impulz	0.0976	-0.8570	0.6826	6.4800
Stopnica	0.1220	-0.8780	0.9998	7.6879
Pravokotni pulzi	0.1170	-0.8834	1.0033	8.0671
Psevdonaključni binarni signal	0.1201	-0.8796	0.9980	7.7964
Pravi parametri procesa	0.1175	-0.8825	1	8



Slika 14.6. Vzbujanje s pravokotnimi pulzi (levo) in izhod procesa brez motnje (desno).



Slika 14.7. Vzbujanje s PRBS (pseudonaključni binarni signal) (levo), ki je deterministična aproksimacija belega šuma v diskretnem času in izhod procesa brez motnje (desno).

$$y(k) = G(q)u(k) = \frac{\tilde{B}(q)}{\tilde{A}(q)}u(k). \quad (14.2)$$

V splošnem lahko prenosna funkcija procesa $G(q)$ vsebuje števec $\tilde{B}(q)$ in imenovalec $\tilde{A}(q)$. Poleg tega determinističnega dela procesa, pa lahko modeliramo še nedeterministični del. S filtriranjem belega šuma $v(k)$ skozi linearni filter $H(q)$ lahko modeliramo katerokoli karakteristiko šuma (barvni šum). Na ta način lahko dobimo poljubni barvni šum $n(k)$, ki ga zapišemo

$$n(k) = H(q)v(k) = \frac{\tilde{C}(q)}{\tilde{D}(q)}v(k). \quad (14.3)$$

Splošni linearni model, ki opisuje deterministični del in stohastični del procesa, lahko sedaj zapišemo tako, da združimo oba dela (glej sliko 14.8a)

$$y(k) = G(q)u(k) + H(q)v(k). \quad (14.4)$$

Filter $G(q)$ se imenuje *vhodna prenosna funkcija ali prenosna funkcija procesa*, ker povezuje vhod $u(k)$ z izhodom $y(k)$, filter $H(q)$ pa se imenuje *prenosna funkcija šuma*, ker povezuje vhodni beli šum $v(k)$ z izhodom procesa $y(k)$. Prenosni funkciji $G(q)$ in $H(q)$ lahko razstavimo na njuna števca in imenovalca; glej sliko 14.8b. Za nadaljnjo analizo je dobro ločiti tudi skupni imenovalec $A(q)$ iz $G(q)$ in $H(q)$; glej slike 14.8c in 14.8d. Splošno obliko linearnega modela lahko zapišemo

$$y(k) = \frac{B(q)}{F(q)A(q)}u(k) + \frac{C(q)}{D(q)A(q)}v(k), \quad (14.5)$$

ali ekvivalentno kot

$$A(q)y(k) = \frac{B(q)}{F(q)}u(k) + \frac{C(q)}{D(q)}v(k). \quad (14.6)$$

S posebnimi predpostavkami za vrednosti polinomov A, B, C, D in F dobimo pogosto uporabljeni tipe linearnih modelov. Preden se lotimo obravnave posameznih modelov, si oglejmo nekaj splošnih pojmov in splošnih lastnosti splošnega linearnega modela.

14.3.1 Terminologija in klasifikacija

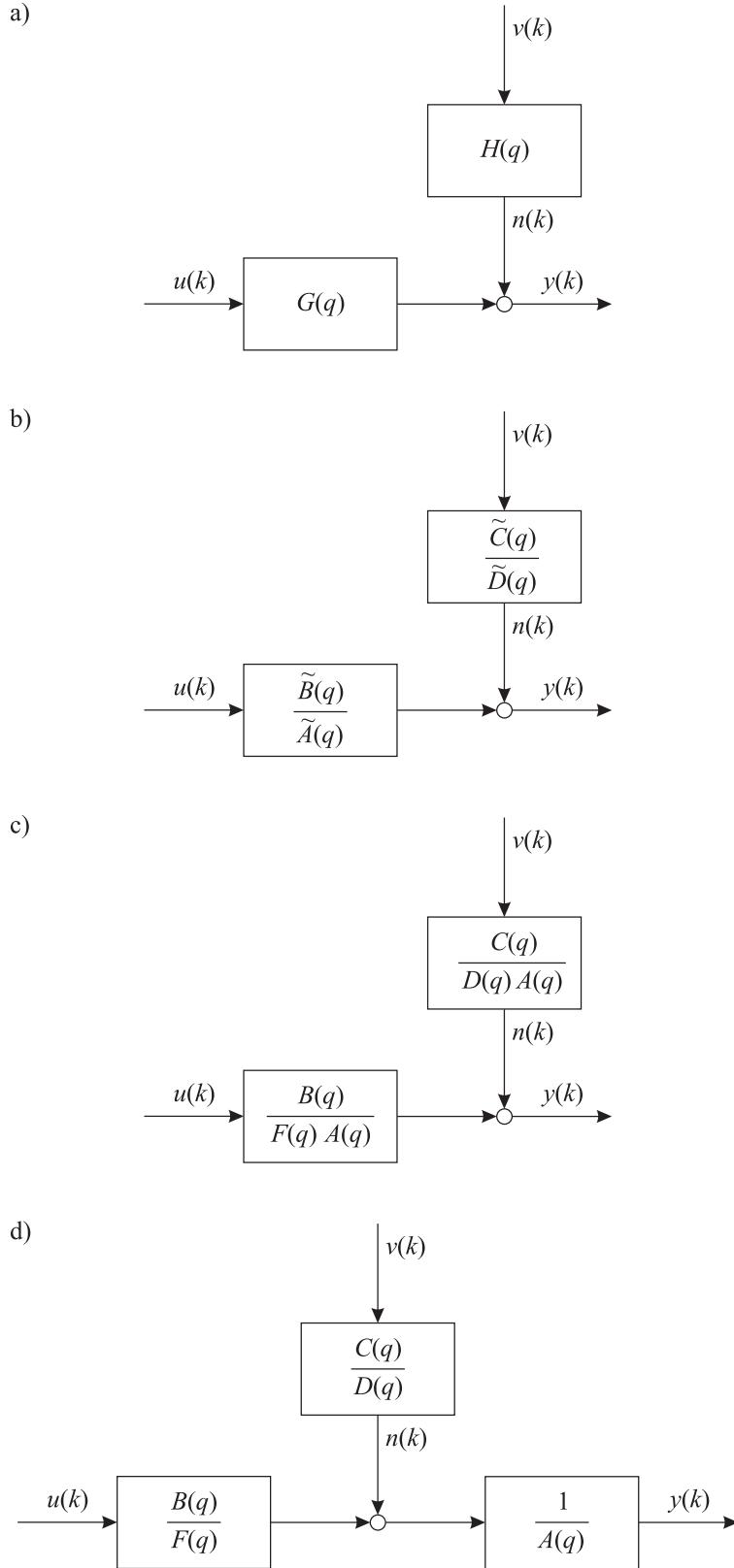
Najprej poglejmo, kako iz splošnega modela dobimo model za modeliranje časovne vrste, kjer nimamo informacije o vhodu v proces ali pa ga ne znamo določiti iz množice vseh možnih spremenljivk (ekonomski modeli). Tak model dobimo, če v splošnem modelu na sliki 14.8 in enačbi 14.6 eliminiramo vhod tako, da je $u(k) = 0$. V takem primeru postane model popolnoma stohastičen. Model časovne vrste je prikazan na sliki 14.9, kot nasprotje popolnoma determinističnega modela na sliki 14.10.

Model časovne vrste, ki ima samo imenovalec (slika 14.11) imenujemo model AR (*avtoregresijski model, ang. AutoRegressive model*) in ga zapišemo kot:

$$y(k) = \frac{1}{D(q)}v(k). \quad (14.7)$$

Model časovne vrste, ki pa ima samo števec (slika 14.11) pa se imenuje model MA (*model drsečega povprečja, ang. Moving Average*) in ga zapišemo kot

$$y(k) = C(q)v(k). \quad (14.8)$$



Slika 14.8. Splošni linearni model

Model časovne vrste, ki pa ima tako števec kot imenovalec v prenosni funkciji (slika 14.11) pa se imenuje model ARMA (*avtoregresijski model drsečega povprečja, ang. AutoRegressive Moving Average*) in ga zapišemo kot

$$y(k) = \frac{C(q)}{D(q)}v(k). \quad (14.9)$$

Razumljivo je, da modeli, ki temeljijo le na časovni vrsti, brez upoštevanja ustreznega vhodnega signala, ne morejo biti zelo natančni. Bolj natančne modele lahko dobimo z vpeljavo enega ali več vhodov v model. Ta vhod $u(k)$ imenujemo *zunanji vhod* (ang. *exogenous input*). Glede na to dobijo imena modelov časovnih vrst, ki imajo dodan zunanji vhod, še oznako "X" za "eXogenous" vhod.

Slika 14.12 daje pregled najbolj pomembnih linearnih dinamičnih vhodno-izhodnih modelov, ki so na kratko obravnavani v nadaljevanju. Vsi modeli na levi strani slike 14.12 so označeni z AR in spadajo v razred modelov s posplošenim pogreškom ali razred modelov z *napako enačbe*. Njihova glavna skupna značilnost je ta, da imajo skupni imenovalec $A(q)$ tako v determinističnem delu, kot tudi v stohastičnem delu modela. Vsi modeli na desni strani pa pripadajo razredu modelov z *izhodnim pogreškom*, ki je definiran s šumom, ki je neodvisen od determinističnega dela modela.

Model ARX, *avtoregresijski model z zunanjim vhodom* na sliki 14.12, je razširjeni AR model:

$$y(k) = \frac{B(q)}{A(q)}u(k) + \frac{1}{A(q)}v(k). \quad (14.10)$$

Pojem "avtoregresiven" se tu nanaša na prenosno funkcijo med vhodom $u(k)$ in izhodom $y(k)$ in tudi na prenosno funkcijo šuma med $v(k)$ in $y(k)$. Pomembno je, da imata deterministični in stohastični del enaka imenovalca.

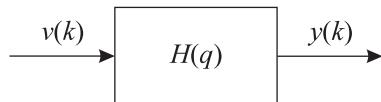
Model ARMAX, *avtoregresijski model drsečega povprečja z zunanjim vhodom* na sliki 14.12, je razširjen ARMA model:

$$y(k) = \frac{B(q)}{A(q)}u(k) + \frac{C(q)}{A(q)}v(k). \quad (14.11)$$

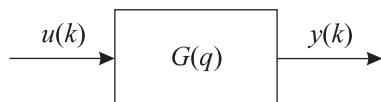
Kot pri modelu ARX, sta tudi v primeru modela ARMAX imenovalca vhodne in šumne prenosne funkcije enaka. Vendar je v tem primeru šumna prenosna funkcija bolj fleksibilna, z dodanimi parametri.

Model ARARX, *avtoregresijski z zunanjim vhodom* na sliki 14.12), je razširitev modela AR:

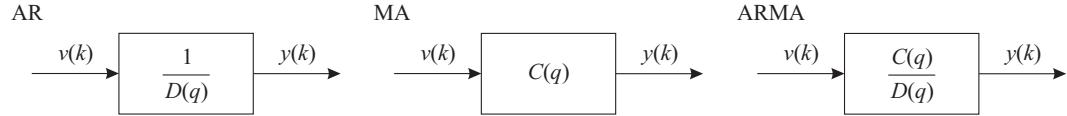
$$y(k) = \frac{B(q)}{A(q)}u(k) + \frac{1}{D(q)A(q)}v(k). \quad (14.12)$$



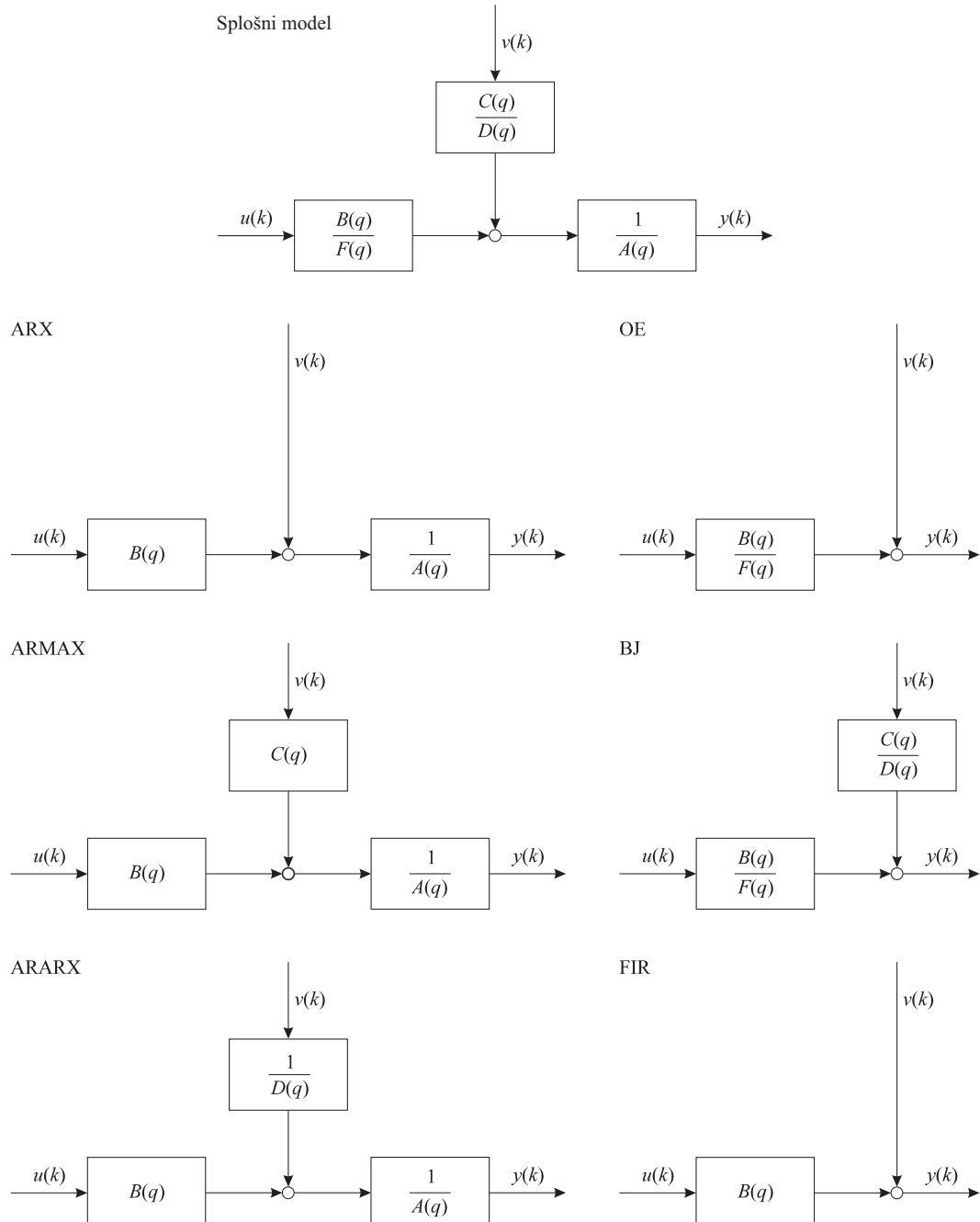
Slika 14.9. Splošni linearni model časovne vrste. Vhod modela $v(k)$ je beli šum, nimamo pa determinističnega vhoda $u(k)$.



Slika 14.10. Splošni linearni deterministični model. Vhod modela $u(k)$ je deterministični signal, nimamo pa stohastičnega signala $v(k)$.



Slika 14.11. Pregled modelov časovnih vrst: avtoregresijski model (AR), model drsečega povprečja (MA) in avtoregresijski model drsečega povprečja (ARMA).



Slika 14.12. Pregled najpomembnejših linearnih dinamičnih modelov.

To je model ARX z dodano fleksibilnostjo v imenovalcu prenosne funkcije šuma. V tem primeru je namesto filtra drsečega povprečja $C(q)$, kot je to v primeru modela ARMAX, dodan avtoregresijski filterski člen $1/D(q)$ in dobimo model ARARX.

Enako je z razširitvijo dobljen tudi model ARARMAX, *avtoregresijski model drsečega povprečja z zunanjim vhodom*, ki ga zapišemo kot

$$y(k) = \frac{B(q)}{A(q)}u(k) + \frac{C(q)}{D(q)A(q)}v(k). \quad (14.13)$$

Ta model je zelo kompleksen in se redko uporablja.

Vsi modeli, ki imajo v imenu AR, imajo skupni polinom $A(q)$ kot imenovalec v vhodni in prenosni funkciji šuma. Imenujemo jih tudi **modeli s pospološenim pogreškom**, ker napako tvorimo iz vhoda in iz izhoda, ali modeli **napake enačbe**. Skupni imenovalec ustreza dejstvu, da šum običajno ne vstopa v proces direktno na izhodu $y(k)$, ampak vstopa pred filtrom $1/A(q)$. Ta predpostavka je seveda smiselna le takrat, ko se to zares zgodi in je frekvenčna karakteristika šuma modulirana z dinamiko procesa. Če pa nastane šum v prvi vrsti zaradi meritve, potem vstopa direktno na izhodu in je *model z izhodnim pogreškom* bolj realističen.

Najbolj enostaven model **izhodnega pogreška** je model OE (*ang. output-error model - OE*) na sliki 14.12 in ga lahko zapišemo z enačbo

$$y(k) = \frac{B(q)}{F(q)}u(k) + v(k). \quad (14.14)$$

V nasprotju z modelom ARX, imamo pri modelu OE nefiltriran beli šum, ki se prišteva izhodnemu signalu brez kakršnegakoli filtra. To pomeni, da predpostavljam direkten vpliv na izhod procesa. Terminologija v primeru teh modelov ne sledi terminologiji, ki smo je predstavili na začetku.

Enostavni model OE lahko razširimo s filtriranjem belega šuma skozi ARMA filter. Na ta način dobimo model *Box-Jenkins-BJ*, ki je na sliki 14.12 in ga zapišemo kot

$$y(k) = \frac{B(q)}{F(q)}u(k) + \frac{C(q)}{D(q)}v(k). \quad (14.15)$$

Model BJ lahko primerjamo z modelom ARARMAX, tako kot tudi model OE lahko primerjamo z modelom ARX. Vhodno-izhodna prenosna funkcija in prenosna funkcija šuma sta popolnoma različni in sta zato neodvisni. V posebnih primerih modela BJ sta $C(q) = 1$ in $D(q) = 1$.

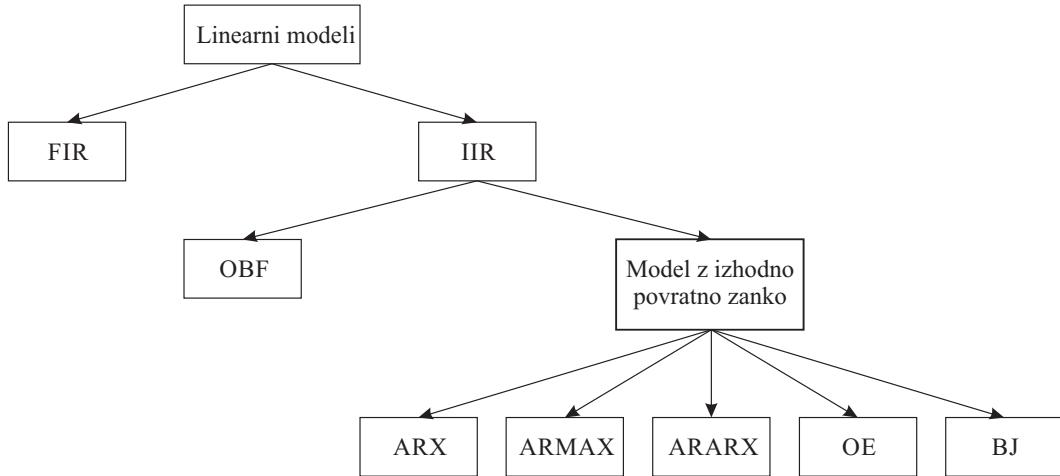
Na koncu predstavimo še model izhodnega pogreška, ki je precej drugačen od ostalih, to je *model končnega impulznega odziva* (*ang. Finite Impulse-Response model - FIR*), ki je prikazan na sliki 14.12 in ga zapišemo kot

$$y(k) = B(q)u(k) + v(k). \quad (14.16)$$

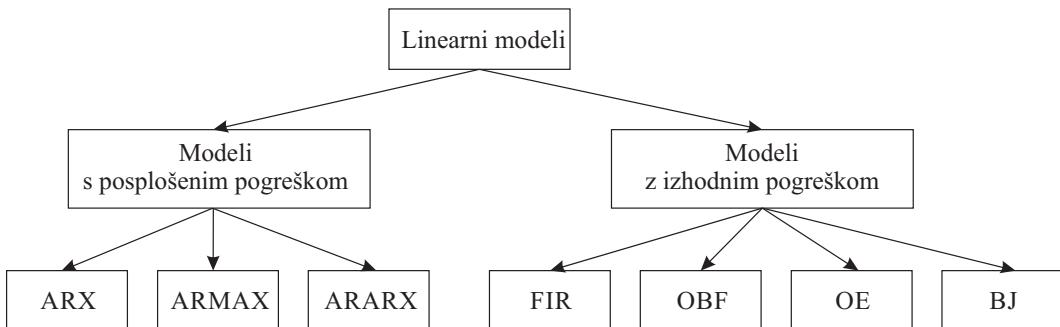
FIR model je OE ali ARX model brez kakršnekoli povratne zanke, to pomeni, da je $F(q) = 1$ ali $A(q) = 1$.

Pomembno je, da si zapomnimo štiri osnovne strukture: ARX, ARMAX, OE in FIR, na katerih slonijo koncepti, ki jih bomo obravnavali v nadaljevanju. Slike 14.13–14.16 prikazujejo opisane linearne modela z drugega vidika. Tabela 14.2 povzame poenostavitve, ki vodijo od glavnega modela do posebnih struktur.

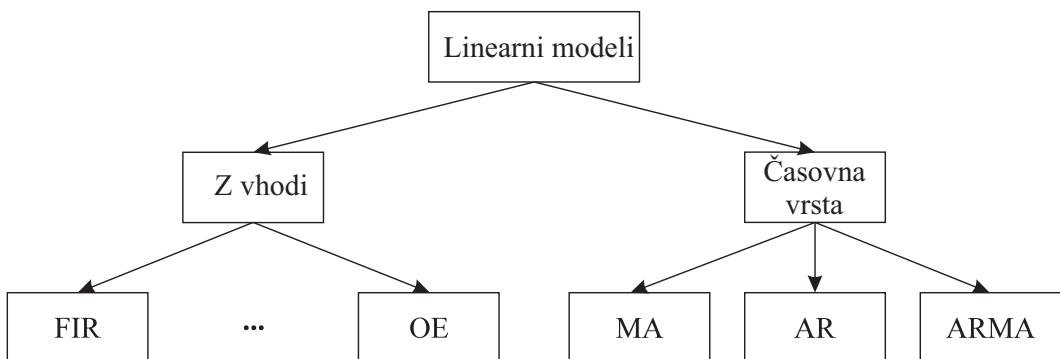
V naših zapisih smo zaradi enostavnosti zapisa predpostavili procese in modele brez mrtvega časa. Ne glede na to pa lahko v vsaki enačbi vhod $u(k)$ zamenjamo z zakasnejmin vhodnim



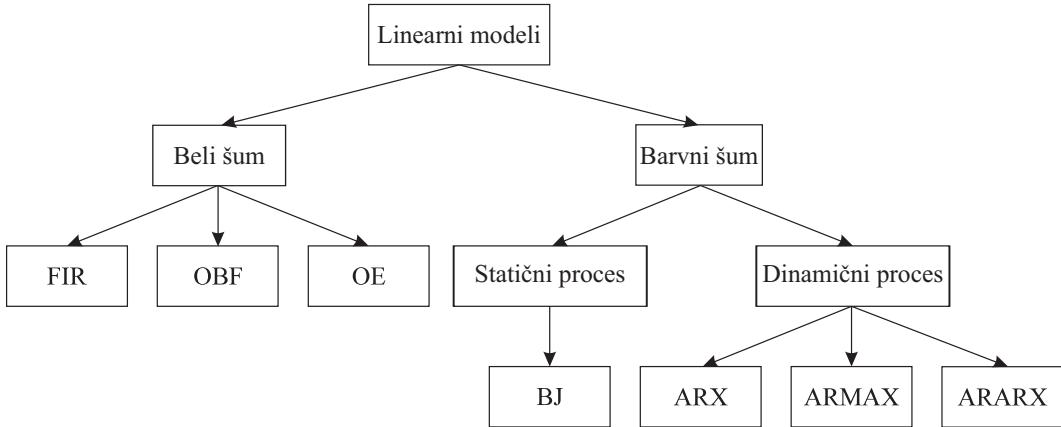
Slika 14.13. Razdelitev linearnih modelov glede na model končnega (FIR) in neskončnega impulznega odziva (IIR) (OBF - ang. ortonormal basis-function model).



Slika 14.14. Razdelitev linearnih modelov na modele s posplošenim in modele z izhodnim pogreškom.



Slika 14.15. Razdelitev linearnih modelov na modele z vhodin in modele časovnih vrst.



Slika 14.16. Razdelitev linearnih modelov glede na lastnosti šuma.

Tabela 14.2. Pogosti linearni modeli.

Struktura modela	Enačba modela
MA	$y(k) = C(q) v(k)$
AR	$y(k) = 1/D(q) v(k)$
ARMA	$y(k) = C(q)/D(q) v(k)$
ARX	$y(k) = B(q)/A(q) u(k) + 1/A(q) v(k)$
ARMAX	$y(k) = B(q)/A(q) u(k) + C(q)/A(q) v(k)$
ARARX	$y(k) = B(q)/A(q) u(k) + 1/D(q)A(q) v(k)$
ARARMAX	$y(k) = B(q)/A(q) u(k) + C(q)/D(q)A(q) v(k)$
OE	$y(k) = B(q)/F(q) u(k) + v(k)$
BJ	$y(k) = B(q)/F(q) u(k) + C(q)/D(q) v(k)$
FIR	$y(k) = B(q) u(k) + v(k)$

signalom $u(k-d)$, kjer je $d T_0$ mrtvi čas. Hkrati smo tudi predpostavili, da obravnavani procesi nimajo direktne poti med vhodom in izhodom. Ta predpostavka je izpolnjena za večino realnih procesov.

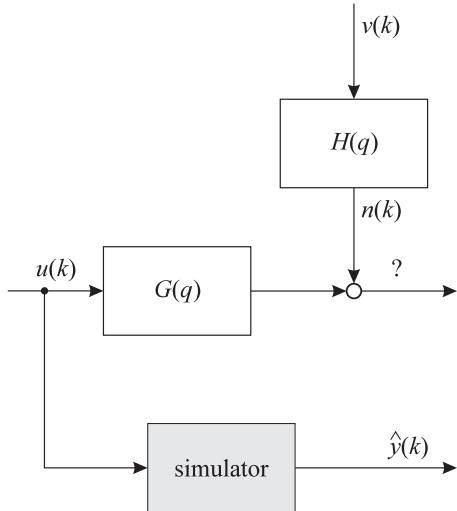
14.3.2 Optimalni prediktor

Najpogostejsa uporaba modelov je v napovedovanju obnašanja procesa v prihodnosti. Pri tej napovedi lahko ločimo dva različna primera: *simulacija* in *predikcija*. Če izračunavamo izhod modela na vhodni signal procesa brez merjenja vmesnih vrednosti izhoda procesa, potem to imenujemo *simulacija*. Če pa so izhodi procesa znani, oziroma so merljivi med izračunavanjem napovedi, potem govorimo o *predikciji*. Slike 14.17 in 14.18 prikazujeta shemi za simulacijo in predikcijo.

Simulacija. Iz enačbe 14.17 sledi, da je simulacija popolnoma deterministična:

$$\hat{y}(k) = G(q)u(k) . \quad (14.17)$$

Zaradi tega je model $H(q)$ popolnoma irelevanten za simulacijo. Pripomnimo lahko, da model šuma $H(q)$ vpliva na oceno parametrov prenosne funkcije $G(q)$ in na ta način posredno vpliva na kvaliteto simulacije, čeprav $H(q)$ ni direktno vključen v enačbi 14.17.



Slika 14.17. Za simulacijo so potrebni samo vhodi, nobene informacije o izhodu niso na voljo.

Ker ne poznamo izhoda procesa, nimamo nobene informacije o motnjah. Če želimo dobiti vsaj nekaj vpogleda v motnje, je običajno najbolje, da generiramo beli šum $w(k)$ ustrezne variance, ga filtriramo preko filtra $H(q)$ in dodamo determinističnemu izhodu modela

$$\hat{y}(k) = G(q)u(k) + H(q)w(k). \quad (14.18)$$

Enačba 14.18 prinaša samo boljši kvalitativni vpogled v obnašanje procesa, ker je šum $w(k)$ drugačen od šuma $v(k)$ na sliki 14.17.

Predikcija. V nasprotju s simulacijo, pa lahko v primeru predikcije uporabimo informacijo o zakasnjenjih izmerjenih vrednostih izhodnega signala procesa. To pomeni, da *optimalni prediktor* kombinira vhod in zakasnjenje merjene izhode procesa. Optimalni *linearni* prediktor lahko definiramo kot linearno kombinacijo filtriranih vhodov in izhodov procesa

$$\begin{aligned} \hat{y}(k|k-1) = & s_0u(k) + s_1u(k-1) + \dots + s_{ns}u(k-ns) \\ & + t_1y(k-1) + \dots + t_{nt}y(k-nt), \end{aligned} \quad (14.19)$$

ali

$$\hat{y}(k|k-1) = S(q)u(k) + T(q)y(k). \quad (14.20)$$

Opozorimo naj, da je v filtru $T(q)$ člen $t_0 = 0$, ker izhod $y(k)$ ni na voljo, ko delamo predikcijo iz $k-1$ časovnega trenutka ($\hat{y}(k|k-1)$). Za večino realnih procesov pa velja tudi $s_0 = 0$, ker vhod ne more v trenutku vplivati na izhod procesa, kar pomeni, da je model strogo pravi (*ang. strictly proper*).

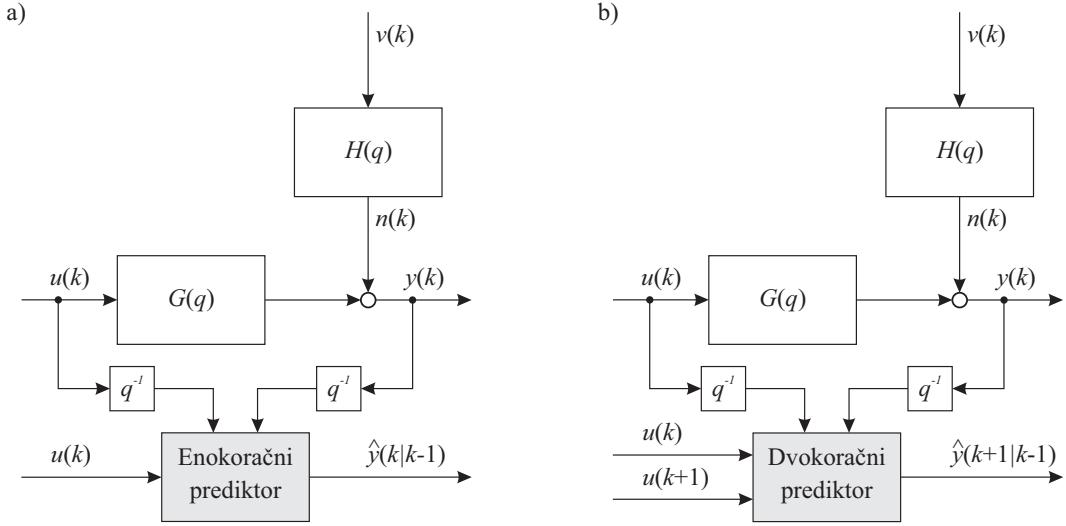
V literaturi je pokazano, da je mogoče določiti optimalni prediktor, ki da najmanjšo kvadratično napako predikcije (varianco predikcijske napake) tako, da je [78]

$$\hat{y}(k|k-1) = \frac{G(q)}{H(q)}u(k) + \left(1 - \frac{1}{H(q)}\right)y(k), \quad (14.21)$$

ali

$$H(q)\hat{y}(k|k-1) = G(q)u(k) + (H(q) - 1)y(k). \quad (14.22)$$

To pomeni, da sta $S(q) = G(q)/H(q)$ in $T(q) = 1 - 1/H(q)$.



Slika 14.18. a) Enokoračna predikcija in b) dvokoračna predikcija. Zapis „ $|k - 1$ ” pomeni, da je informacija dostopna v časovnem trenutku $k - 1$. Za predikcijo potrebujemo poleg vhodov tudi zakasnjene izhode procesa. Če je predikcijski horizont l velik, informacija o prejšnjih izhodih procesa nima več posebnega pomena. Če gre $l \rightarrow \infty$, predikcija limitira k simulaciji, slika 14.17 .

Optimalni prediktor mora vso informacijo dobiti iz preteklih signalov. Na ta način bi moral biti predikcijski pogrešek, razlika med izhodom procesa $y(k)$ in predikcijo $\hat{y}(k|k-1)$, enak belemu šumu $v(k)$, ker je to edina stvar, ki je nenapovedljiva:

$$v(k) = y(k) - \hat{y}(k|k-1) . \quad (14.23)$$

To enačbo lahko vstavimo v enačbo, ki je bločno prikazana na sliki 14.17 in dobimo:

$$y(k) = G(q)u(k) + H(q)(y(k) - \hat{y}(k|k-1)) . \quad (14.24)$$

Če iz te enačbe izrazimo $\hat{y}(k|k-1)$, dobimo optimalni prediktor iz enačbe 14.21. Optimalni prediktor rezultira v residualu belega šuma, kar lahko izkoristimo za testiranje strukture modela.

14.3.3 Modeli predikcijskega pogreška PEM

Običajno uporabimo optimalni prediktor za oceno kvalitete modela. Predikcijski pogrešek je razlika med izhodom procesom in enokoračno predikcijo modela pri enakem vhodu

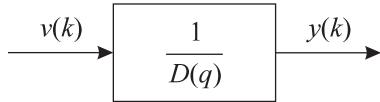
$$e(k) = y(k) - \hat{y}(k|k-1) . \quad (14.25)$$

V nadaljevanju smo izraz *predikcijski pogrešek* uporabili kot sinonim za *enokoračni* predikcijski pogrešek. Z uporabo optimalnega prediktorja iz enačbe 14.21 dobimo predikcijski pogrešek, ki je enak

$$e(k) = \frac{1}{H(q)}y(k) - \frac{G(q)}{H(q)}u(k) . \quad (14.26)$$

Večina identifikacijskih algoritmov temelji na minimizaciji kriterijske funkcije, ki je funkcija predikcijskega pogreška (enokoračne predikcije), to pomeni, da je kriterijska funkcija naslednja

$$J = \sum_{i=1}^N e^2(i) . \quad (14.27)$$



Slika 14.19. Model AR.

14.4 Modeli časovnih vrst

Časovna vrsta je model razvijajočega se signala, ki na osnovi svojih zakasnjenih vrednosti napoveduje prihodnje vrednosti. Časove vrste se lahko razvijajo po času, lahko pa tudi prostorsko. Dobri primeri za uporabo časovnih vrst so napovedi v ekonomiji (valutna razmerja, borzni indeks, gospodarska rast, zaposlenost) ali v geologiji (razširjanje potresnega vala). Običajno jih uporabljamo takrat, ko so vhodni signali neznani ali pa preveč kompleksi, da bi jih lahko vključili v model. Zato je tudi zelo razumljivo, da je tipična uporaba časovnih vrst predvsem na netehniških področjih. Na področju tehniških sistemov so relacije znotraj procesov bistveno bolje razumljive in znane in je seveda v takih primerih bolj smiselno uporabiti modele z determinističnimi vhodi in po možnosti dodati tudi stohastične elemente.

Kvaliteta modela, ki temelji na časovni vrsti, je navadno manjša od modelov, ki imajo deterministične vhode, vseeno pa lahko v določenih primerih zgradimo model, ki bo dal dobro kratkoročno napoved (enokoračno).

Model časovne vrste zapišemo kot filtriranje signala belega šuma za katerega pa ne poznamo lastnosti; glej sliko 14.9. Zapišemo ga kot

$$y(k) = H(q)v(k) = \frac{C(q)}{D(q)}v(k), \quad (14.28)$$

kjer je $y(k)$ izhod modela in $v(k)$ je umetni signal belega šuma.

14.4.1 Model AR

Model AR (avtoregresijski model časovne vrste) na sliki 14.19 je zelo pogosto uporabljen, saj lahko njegovo frekvenčno karakteristiko moduliramo s samo nekaj linearnimi parametri. Zelo pogosto ga uporabljamo za modeliranje rahlo dušenih nihanj pri zelo ošumljenih podatkih. Oscilacije so predstavljene s parom konjugirano kompleksnih polov v funkciji $1/D(q)$.

Model časovnega odziva dobimo kot filtriranje belega šuma $v(k)$ in ga lahko zapišemo kot:

$$y(k) = \frac{1}{D(q)}v(k). \quad (14.29)$$

Model AR lahko zapišemo tudi v obliki diferenčne enačbe

$$y(k) = -d_1y(k-1) - \dots - d_my(m) + v(k). \quad (14.30)$$

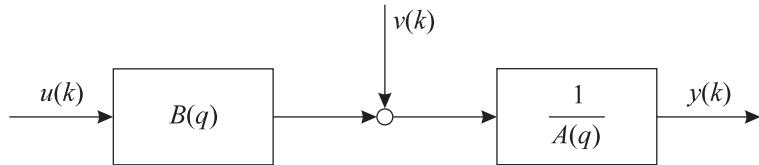
Optimalni prediktor je v primeru model AR enak

$$\hat{y}(k|k-1) = (1 - D(q))y(k). \quad (14.31)$$

Napaka predikcije je v tem primeru enaka

$$e(k) = y(k) - \hat{y}(k|k-1) = y(k) - (1 - D(q))y(k) = D(q)y(k). \quad (14.32)$$

Z uporabo metode predikcijskega pogreška lahko ocenimo parametre modela z metodo najmanjših kvadratov, ki jo lahko enostavno rešimo.



Slika 14.20. Model ARX.

14.5 Modeli z izhodno povratno zanko

V nadaljevanju si bomo ogledali linearne modele z izhodno povratno zanko in metode za ocenjevanje njihovih parametrov. Ti modeli so daleč najbolj uporabni.

14.5.1 Model ARX

Model ARX (avtoregresijski model z zunanjim vhodom) je daleč najbolj pogost linearji dinamični model. Navadno najprej poskusimo sistem modelirati z modelom ARX in, če rezultat ni zadovoljiv, potem poiskušamo z bolj kompleksnimi zgradbami. Kar pa se redko zgodi, ker je njegova zgradba zelo blizu realnim procesom. Popularnost modela ARX je treba pripisati tudi zelo enostavnemu načinu ocene parametrov. Parametre preprosto ocenimo z metodo najmanjših kvadratov, ker je napaka predikcije linearna v parametrih. Hkrati pa obstaja tudi zanesljiva rekurzivna možnost izračuna. Model ARX je prikazan na sliki 14.20 in ga zapišemo kot

$$A(q)y(k) = B(q)u(k) + v(k) . \quad (14.33)$$

Optimalni ARX prediktor lahko v primeru modela ARX zapišemo s polinomoma prenosne funkcije kot

$$\hat{y}(k|k-1) = B(q)u(k) + (1 - A(q))y(k) , \quad (14.34)$$

ali pa ga zapišemo v obliki diferenčne enačbe kot

$$\begin{aligned} \hat{y}(k|k-1) &= b_1u(k-1) + \dots + b_mu(k-m) \\ &\quad - a_1y(k-1) - \dots - a_my(k-m) , \end{aligned} \quad (14.35)$$

ob predpostavki, da je $\text{red}(A) = \text{red}(B) = m$.

Prediktor modela ARX je stabilen (ne vsebuje povratne zanke) celo, če ima polinom $A(q)$ nestabilne pole in je model ARX nestabilen. Zaradi tega lahko nestabilne procese modeliramo z modelom ARX. To je lastnost vseh modelov s posplošenim pogreškom, ki imajo polinom $A(q)$ samo v števcu optimalnega prediktorja in je zaradi tega prediktor stabilen, čeprav ima npr. $A(q)$ nestabilne pole.

Glede na enačbo 14.34 je predikcijski pogrešek za model ARX enak

$$e(k) = A(q)y(k) - B(q)u(k) . \quad (14.36)$$

Člen $A(q)y(k)$ deluje kot filter na korelirano motnjo. Merjeni izhod $y(k)$ lahko razdelimo na dva dela: deterministični izhod procesa $y_u(k)$ in motnjo $n(k)$. To pomeni, da je $y(k) = y_u(k) + n(k)$. Ker je $n(k) = (1/A(q))v(k)$ in je $v(k)$ beli šum, dobimo $A(q)y(k) = A(q)y_u(k) + v(k)$. To pomeni, da filter $A(q)$ v enačbi 14.36 povzroči, da je $e(k)$ beli šum.

Kot vidimo na sliki 14.20 je ena od lastnosti modela ARX ta, da prihaja motnja belega šuma $v(k)$ v proces pred imenovalcem procesa $A(q)$. Kar z drugimi besedami pomeni, da ima

model ARX tudi model šuma, ki je enak $1/A(q)$. Šum ima tako enako dinamiko kot proces, kar lahko upravičimo, če motnja prihaja v proces blizu njegovega vhoda. Seveda pa bi v tem primeru morali upoštevati tudi števec $B(q)$. V realnih procesih redko srečamo takšen primer šuma. Bistveno bolj realistične so motnje, ki jih predpostavlja model OE.

Slika 14.21 prikazuje tri različne konfiguracije modela ARX. Vse tri slike predstavljajo enak model ARX v različnih interpretacijah. Polinomi procesa so označeni z $B(q)$ in $A(q)$, modela pa s strešico $\hat{B}(q)$ in $\hat{A}(q)$.

Slika 14.21a predstavlja najbolj običajno konfiguracijo. Pogrešek $e(k)$ za model ARX se imenuje *pospološeni pogrešek ali pogrešek enačbe*, ker je razlika v enačbi enaka $e(k) = \hat{A}(q)y(k) - \hat{B}(q)u(k)$. Ime "pogrešek enačbe" poudari dejstvo, da ni to razlika med izhodom procesa $y(k)$ in $\hat{B}(q)/\hat{A}(q)u(k)$, ki se imenuje *izhodni pogrešek*. Glede na sliko 14.21a je razumljivo, da je v primeru enakosti med modelom in procesom, to je $\hat{B}(q) = B(q)$ in $\hat{A}(q) = A(q)$, pospološeni pogrešek ali pogrešek enačbe enak $e(k) = \hat{A}(q)n(k) = A(q)n(k)$. Glede na strukturo v primeru modela ARX, kjer je motnja s filtrom $1/A(q)$ filtriran beli šum, je potem pospološeni pogrešek $e(k)$ enak belemu šumu (ker velja da je $n(k) = 1/A(q)v(k)$).

Slika 14.21b prikazuje drugo konfiguracijo modela ARX, ki temelji na enačbi predikcije. S prediktorjem, v primeru modela ARX v enačbi 14.34, dobimo enak pospološeni pogrešek kot na sliki 14.21a. Slika 14.21c predstavlja konfiguracijo modela ARX v obliki izhodnega pogreška (OE). V tem primeru lahko vidimo, da je pospološeni pogrešek $e(k)$ filtrirana verzija izhodnega pogreška $e_{OE}(k)$. Povejmo, da $e_{OE}(k)$ in $\hat{y}_{OE}(k)$ označujeta izhodni pogrešek in izhod modela OE in sta različna od predikcijskega pogreška $e(k)$ in predikcije izhoda $\hat{y}(k)$ pri modelu ARX. Slike 14.21a, b in c z različnimi interpretacijami istega modela ARX smo predstavili zato, da osvetlimo relacije med različnimi strukturami.

Metoda najmanjših kvadratov. Popularnost modela ARX je v veliki meri posledica možnosti enostavne ocene parametrov z metodo najmanjših kvadratov. Za N podatkovnih vzorcev, ki jih imamo na voljo, lahko model ARX zapišemo v naslednji vektorsko-matrični obliku, ki ima $N - m$ enačb. Pri tem je $k = m + 1, \dots, N$, kjer je \hat{y} vektor izhoda modela, \underline{y} pa je vektor izhoda procesa, ki naj se mu izhod modela čim bolj približa:

$$\hat{y} = \underline{X}\underline{\theta}, \quad (14.37)$$

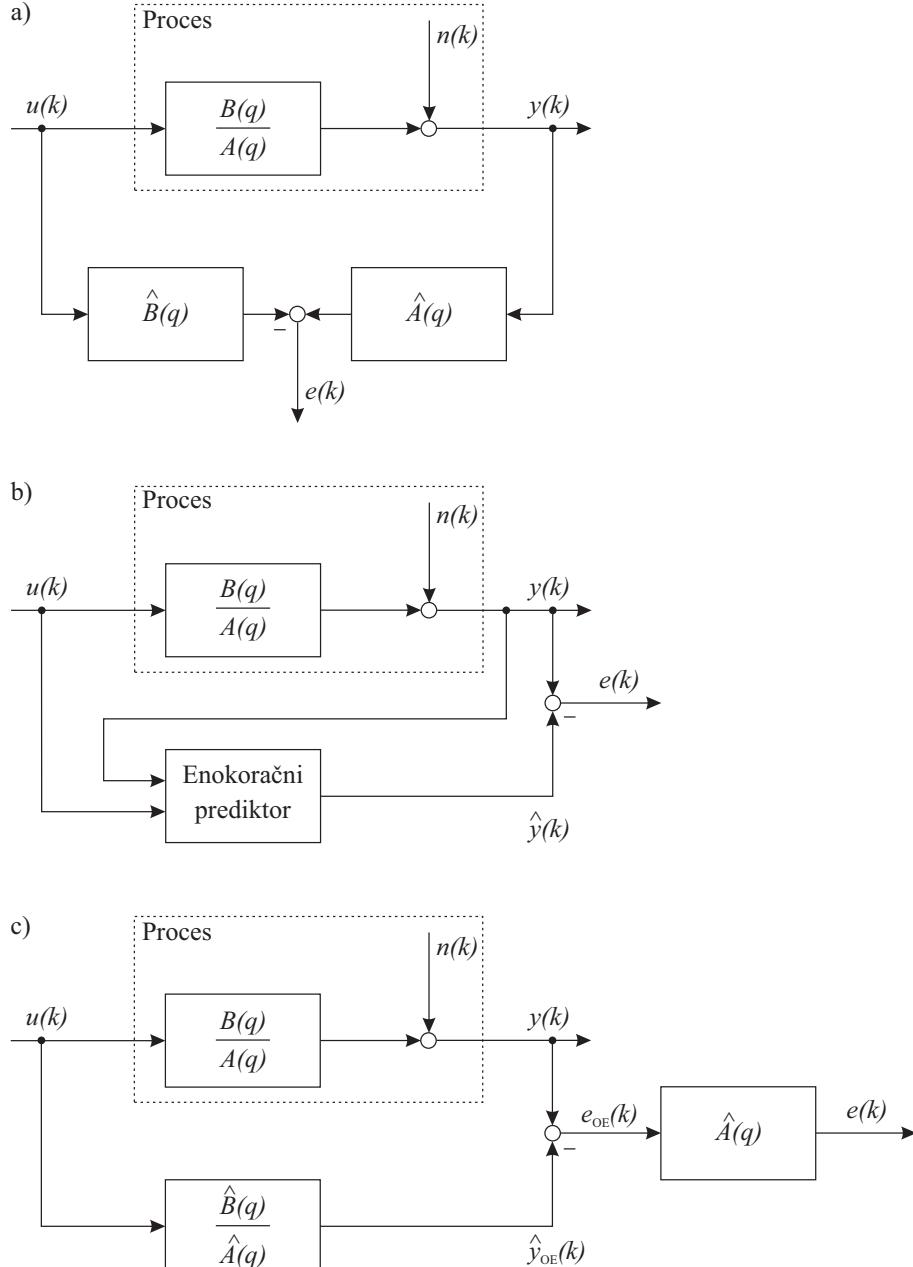
kjer je

$$\hat{y} = \begin{bmatrix} \hat{y}(m+1) \\ \hat{y}(m+2) \\ \vdots \\ \hat{y}(N) \end{bmatrix}, \quad \underline{y} = \begin{bmatrix} y(m+1) \\ y(m+2) \\ \vdots \\ y(N) \end{bmatrix}, \quad \underline{\theta} = \begin{bmatrix} b_1 \\ \vdots \\ b_m \\ a_1 \\ \vdots \\ a_m \end{bmatrix}, \quad (14.38)$$

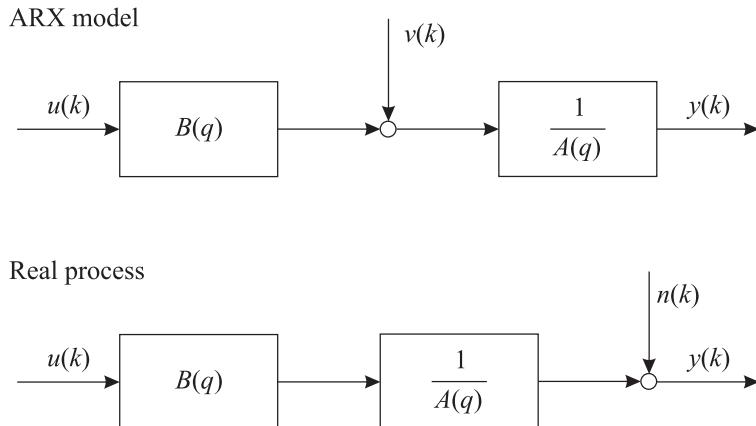
$$\underline{X} = \begin{bmatrix} u(m) & \cdots & u(1) & -y(m) & \cdots & -y(1) \\ u(m+1) & \cdots & u(2) & -y(m+1) & \cdots & -y(2) \\ \vdots & & \vdots & \vdots & & \vdots \\ u(N-1) & \cdots & u(N-m) & -y(N-1) & \cdots & -y(N-m) \end{bmatrix}. \quad (14.39)$$

Če minimiziramo kvadratično kriterijsko funkcijo v enačbi 14.27, dobimo optimalne parametre modela ARX z metodo najmanjših kvadratov (glej poglavje 4)

$$\hat{\underline{\theta}} = (\underline{X}^T \underline{X})^{-1} \underline{X}^T \underline{y}. \quad (14.40)$$



Slika 14.21. Različne sheme modela ARX: a) konfiguracija v obliki s pospoljenim pogreškom, b) konfiguracija v obliki prediktorja, c) psevdo-paralelna konfiguracija s filtriranjem signala pogreška. Vse konfiguracije realizirajo enak model ARX.



Slika 14.22. Model ARX predpostavlja šumni model s prenosno funkcijo $1/A(q)$, čeprav je aditivni šumom na izhodu modela $n(k)$ veliko bolj realističen. Šum $n(k)$ je lahko beli šum ali pa barvni šum $n(k) = C(q)/D(q)v(k)$.

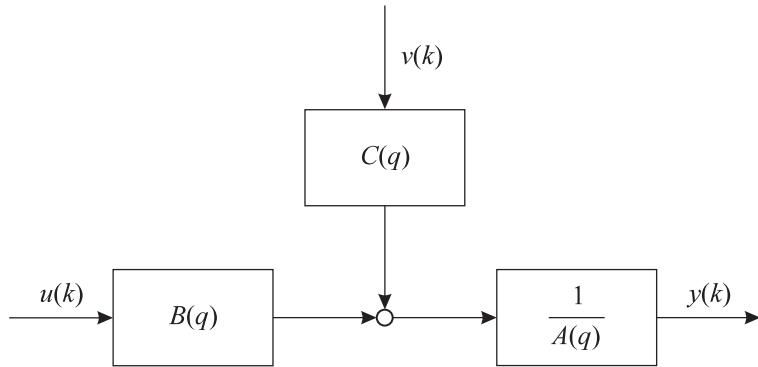
Za izračun ocen v enačbi 14.40 moramo zagotoviti nesingularnost matrike $\underline{X}^T \underline{X}$. Potreben pogoj za to je, da vhodni signal $u(k)$ neprestano vzbuja sistem (*ang. persistent excitation*). Prva velika prednost modela ARX je njegova struktura, ki je linearna v parametrih. Druga prednost pa je ta, da z uporabo linearne optimizacije v enem koraku pridemo do globalnega optimuma. Glavna pomanjkljivost modela ARX pa je v tem, da je šum, ki ga model predpostavlja in ima prenosno funkcijo $1/A(q)$, zelo nerealističen. Aditivni izhodni šum je bistveno bolj pogost. S tem povezani problemi so obravnavani v nadaljevanju.

Pristranskost in konsistenco. Model ARX in druge bolj realistične strukture modelov so prikazane na sliki 14.22. Zaradi nerealistične obravnave šuma v zapisu modela ARX naletimo na nekaj težav. Zelo enostavno lahko pokažemo, da so v primeru, ko proces ne izpolnjuje predpostavk o lastnostih in izvoru šuma, ocenjeni parametri *pristranski* (*ang. biased*) in *nekonsistentni* (*ang. non-consistent*). **Pristranskost pomeni, da parametri sistemsko odstopajo od svojih optimalnih vrednosti**, t.j. parametri so sistematično napačni. **Nekonsistentnost pa pomeni, da ta sistemski napaka tudi v primeru, ko gre število učnih vzorcev N proti neskončnosti, ne gre proti nič.**

Še slabše pa je to, da meje zaupanja, ki jih izračunamo iz kovariančne matrike ocenjenih parametrov (glej poglavje 4) lahko pokažejo, da je ocena zelo natančna, čeprav je pristranskost zelo velika. Kar kaže na to, da metoda zavaja [31]. Razlog za to neželjeno obnašanje leži v tem, da izpeljava različnih teoremov o metodi najmanjših kvadratov v poglavju 4 predpostavlja deterministično regresijsko matriko \underline{X} . Lahko pa hitro ugotovimo, da temu ni čisto tako, saj, kot smo videli v enačbi 14.39, regresijska matrika \underline{X} vsebuje merjeni izhod procesa $y(k)$, ki je nedeterminističen zaradi motnje. Kovariančne matrike v primeru modela ARX torej ne smemo računati po enačbi 4.26 in posledično tudi meje zaupanja ne.

Ker je konsistentnost verjetno najpomembnejša lastnost vsake metode ocenjevanja parametrov, je bilo razvitih veliko strategij za odpravo nekonsistence pri oceni parametrov za model ARX. Ideja večine je v tem, da ohranimo linearnost v parametrih modela ARX, ker je to njegova največja prednost. Prva takšna strategija je metoda dodatnih spremenljivk, instrumentalnih spremenljivk. Druga strategija pa predlaga delo s korelacijami merjenih signalov namesto obravnave samih merjenih signalov.

Alternativne strategije za preseganje problema konsistentnosti ocen parametrov pa gredo v smeri bolj kompleksnih modelov, kot sta model ARMAX ali model OE, ki sta nelinearna v



Slika 14.23. Model ARMAX.

parametrih. Nadaljnji razvoj je šel v smeri ocenjevanja nelinearnih parametrov s ponavljajočo uporabo linearne metode najmanjših kvadratov za oceno nelinearnih parametrov.

14.5.2 Model ARMAX

Model ARMAX (avtoregresijski model drsečega povprečja z zunanjim vhodom) je drugi najbolj popularen model po modelu ARX. Nekateri regulatorji, kot je minimalno variančni regulator, temeljijo na modelu ARMAX in izkoriščajo informacijo, ki jo prinaša šumni model. V primerjavi z modelom ARX, je model ARMAX veliko bolj prilagodljiv, ker vsebuje razširjeni model šuma. Z dodanim šumnim modelom postane model ARMAX nelinearen v parametrih. Razvite so bile zelo učinkovite večkoračne linearne metode najmanjših kvadratov za oceno parametrov modela ARMAX. Model ARMAX je prikazan na sliki 14.23 in ga opisuje enačba

$$A(q)y(k) = B(q)u(k) + C(q)v(k) . \quad (14.41)$$

Optimalni prediktor v primeru modela ARMAX je enak

$$\hat{y}(k|k-1) = \frac{B(q)}{C(q)}u(k) + \left(1 - \frac{A(q)}{C(q)}\right)y(k) . \quad (14.42)$$

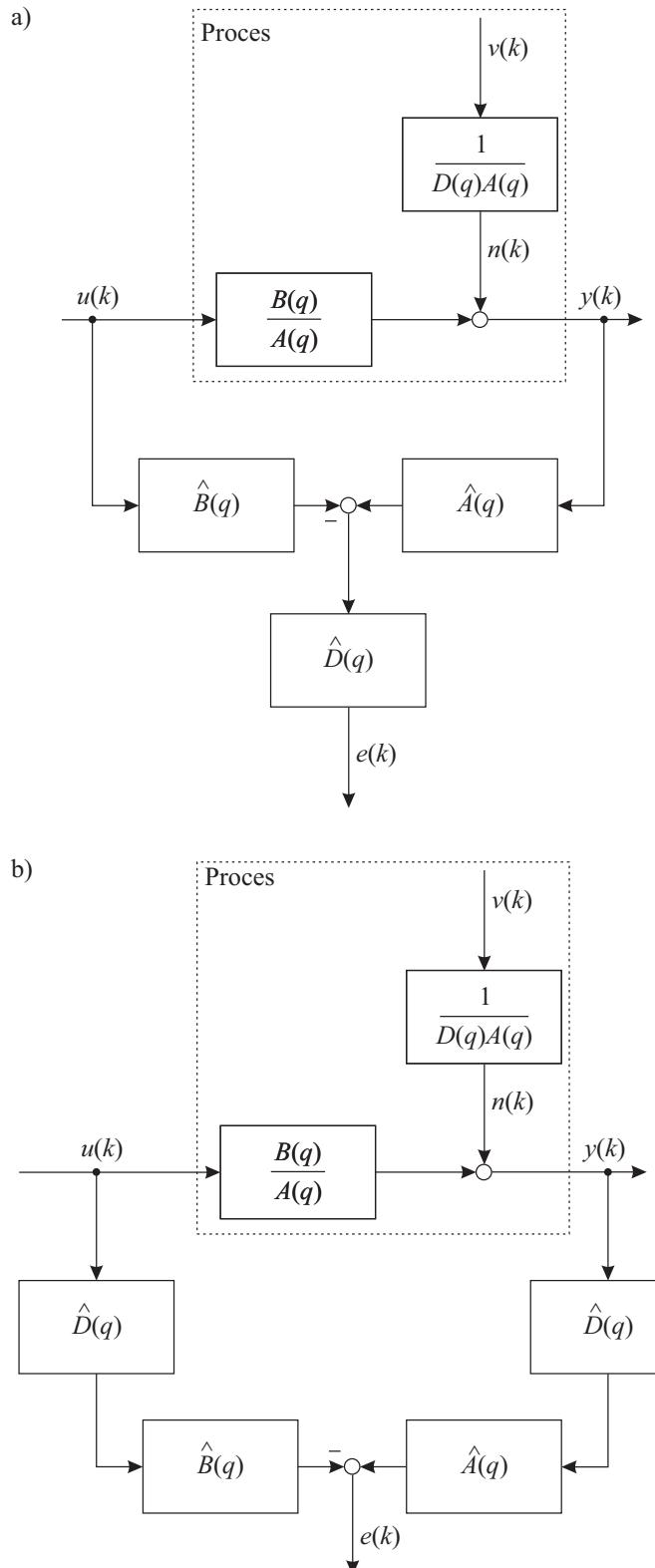
Prediktor v primeru modela ARMAX je stabilen tudi za nestabilni polinom $A(q)$ in s tem nestabilni model ARMAX. Pri tem pa mora biti $C(q)$ stabilen.

Z upoštevanjem enačbe 14.42 dobimo pogrešek predikcije modela ARMAX, ki je enak

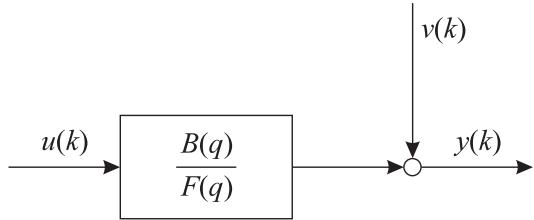
$$e(k) = \frac{A(q)}{C(q)}y(k) - \frac{B(q)}{C(q)}u(k) . \quad (14.43)$$

Če pogledamo gornje enačbe, lahko vidimo, da je model ARMAX razširitev modela ARX, ki vnaša filter $C(q)$. Če je $C(q) = 1$, se model ARMAX poenostavi v model ARX. Zaradi dodatnega filtra $C(q)$ je ARMAX model zelo fleksibilen. Na primer, pri $C(q) = A(q)$, lahko ARMAX model posnema model OE.

Ker šumni filter $C(q)/A(q)$ vsebuje imenovalec, ki je enak vhodnemu delu prenosne funkcije, tudi model ARMAX spada med modele s posplošenim pogreškom. To lahko ugotovimo tudi iz modela ARMAX konfiguracije na sliki 14.24, glej tudi sliko 14.21. Če je $\hat{A}(q) = A(q)$, $\hat{B}(q) = B(q)$ in $\hat{C}(q) = C(q)$, je residual $e(k)$ enak belemu šumu.



Slika 14.24. Model ARMAX v konfiguraciji modela s posplošenim pogreškom.



Slika 14.25. Model OE.

Ocenjevanje parametrov modela ARMAX. Pogrešek v enačbi 14.43 za model ARMAX je nelinearen v svojih parametrih zaradi filtriranja z $1/C(q)$. Vseeno pa lahko pogrešek izrazimo z naslednjo *psevdolinearno enačbo*:

$$C(q)e(k) = A(q)y(k) - B(q)u(k), \quad (14.44)$$

ki jo lahko naprej zapišemo kot

$$e(k) = A(q)y(k) - B(q)u(k) + (1 - C(q))e(k). \quad (14.45)$$

To naprej rezultira v naslednji diferenčni enačbi:

$$\begin{aligned} e(k) &= a_1y(k-1) + \dots + a_my(k-m) \\ &- b_1u(k-1) - \dots - b_mu(k-m) \\ &- c_1e(k-1) - \dots - c_me(k-m). \end{aligned} \quad (14.46)$$

Zgornja enačba formalno predstavlja linearno regresijo. Ker pa so pogreški $e(k-i)$ nemerljivi in jih je potrebno izračunati iz predhodne ocene modela kot residue, tudi parametre imenujemo psevdolinearni parametri. Zaradi tega enačbi 14.45 in 14.46 dovoljujeta dva različna pristopa k oceni parametrov modela. Najenostavnejši je nelinearna optimizacija, druga pa izkoristi psevdolinearni pogrešek.

14.5.3 Model OE

Poleg modelov ARX in ARMAX je model OE (*model z izhodnim pogreškom, ang. Output-Error model*) tretji najbolj pogost model. Je najenostavnejši predstavnik razreda modelov z izhodnim pogreškom. Model predpostavlja aditivno šumno motnjo na izhodu modela in ne v sredini modela, kot je to pri modelih s posplošenim pogreškom. Modeli OE so pogosto bližje realnim problemom in se pogosto bolje izkažejo kot modeli s posplošenim pogreškom. Ker pa šumni model ne vsebuje imenovalca $1/A(q)$, so vse oblike modelov z izhodnim pogreškom nelinearne v parametrih in jih je posledično težje oceniti. Model OE je predstavljen na sliki 14.25 in ga zapišemo z enačbo

$$y(k) = \frac{B(q)}{F(q)}u(k) + v(k). \quad (14.47)$$

Običajna notacija na področju linearne identifikacije sistemov je, da v primeru modela z izhodnim pogreškom imenovalec zapišemo s polinomom $F(q)$, imenovalec modelov s posplošenim pogreškom, kot so modeli ARX, ARMAX in ARARX, pa s polinomom $A(q)$ [78]. To je samo način, da poudarimo razlike med modeli.

Optimalni prediktor v primeru modela OE je kar simulator, ker ne uporablja nobenega izmerjenega izhodnega signala $y(k)$:

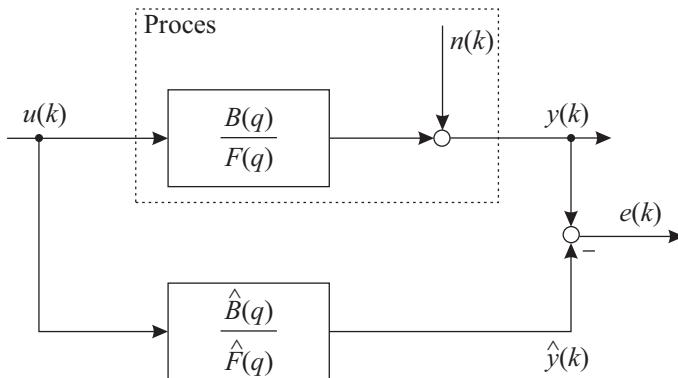
$$\hat{y}(k|k-1) = \hat{y}(k) = \frac{B(q)}{F(q)}u(k). \quad (14.48)$$

Pripomnimo naj, da lahko v tem primeru v notaciji $|k-1$ izpustimo, ker model OE ne temelji na izmerjenih signalih za izračun predikcije. Lahko tudi ugotovimo, da je prediktor v primeru modela OE nestabilen, če je polinom $F(q)$ nestabilen. Zaradi tega ga ne moremo uporabljati za modeliranje nestabilnih procesov. Enako velja za vse modele z izhodnim pogreškom.

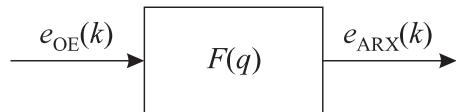
Ob upoštevanju enačbe 14.48 je napaka predikcije za model OE enaka

$$e(k) = y(k) - \frac{B(q)}{F(q)}u(k). \quad (14.49)$$

Slika 14.26 prikazuje model OE paralelno s procesom. Predikcijski pogrešek modela OE je razlika med izhodom procesa in simuliranim izhodom modela. Predpostavljamo, da je motnja $n(k)$ beli šum.



Slika 14.26. Model OE paralelno procesu.



Slika 14.27. Odnos med residuali modela ARX in modela OE. Residuali modela ARX so filtrirani residuali modela OE preko filtra $F(q)$.

Slika 14.27 prikazuje pogrešek modela OE in modela ARX. Glede na enačbe za model ARX in model OE lahko residualne modela ARX dobimo s filtriranjem residualov modela OE

$$e_{\text{ARX}}(k) = F(q)e_{\text{OE}}(k). \quad (14.50)$$

Če predpostavimo, da je $\hat{F}(q) = F(q)$ in $\hat{B}(q) = B(q)$ in ker je $n(k)$ beli šum ($n(k) = v(k)$), potem je tudi $e_{\text{OE}}(k) = v(k)$ beli šum. Pogrešek $e_{\text{ARX}}(k) = F(q)v(k)$ pa je koreliran. Če pa je procesni šum koreliran tako, da je $n(k) = 1/F(q)v(k)$, potem je $e_{\text{OE}}(k) = 1/F(q)v(k)$ koreliran, ker je $e_{\text{ARX}}(k) = v(k)$ beli šum. Ta relacija dovoljuje, da parametre modela OE ocenjujemo tako, da ponavljamo linearno metodo najmanjših kvadratov in filtriranje, čeprav so parametri nelinearni.

Osvetlimo trditev, da je predikcija izhoda modela OE nelinearna v svojih parametrih (glej enačbo 14.48)

$$\begin{aligned}\hat{y}(k) &= b_1 u(k-1) + \dots + b_m u(k-m) \\ &\quad - f_1 \hat{y}(k-1) - \dots - f_m \hat{y}(k-m).\end{aligned}\quad (14.51)$$

Primerljivo z modelom ARX je merjeni izhod v enačbi 14.35 zamenjan z napovedjo (ali simulacijo, kar je v primeru modela OE enako) izhoda v enačbi 14.51. V tem leži vzrok za nelinearnost parametrov enačbe 14.51. Napoved izhoda modela $\hat{y}(k-i)$ je odvisna od prejšnjih napovedi. Tako se v členih $\hat{y}(k-i)$ pojavljajo parametri modela, kar seveda vodi v nelinearnost. Če želimo to odpraviti, potem lahko v enačbi 14.51 zamenjamo izhode modela $\hat{y}(k-i)$ z izmerjenimi izhodi procesa $y(k-i)$. Na ta način se model OE poenostavi v model ARX, ki pa je seveda linearen v parametrih.

Parametre modela OE lahko ocenimo z nelinearno optimizacijo ali pa s ponavljačno metodo najmanjših kvadratov in filtriranjem. Na ta način izkoristimo podobnost z modelom ARX.

Algoritem 12 Psevdokoda ponavljačih se najmanjših kvadratov s filtriranjem za oceno parametrov modela OE

- 1: Ocenjevanje parametrov modela ARX $F(q)y(k) = B(q)u(k) + v(k)$ za učno množico podatkov $\{\underline{u}(k), \underline{y}(k)\}$ na osnovi enačbe

$$\hat{\underline{\theta}}_{\text{ARX}} = (\underline{X}^T \underline{X})^{-1} \underline{X}^T \underline{y} \quad (14.52)$$

kjer so parameteri vektorja $\hat{\underline{\theta}}$ definirani kot f_i in b_i .

2: **repeat**

- 3: Filtriranje vhoda $u(k)$ in izhoda procesa $y(k)$ skozi ocenjeni filter $\hat{F}(q)$:

$$u^F(k) = \frac{1}{\hat{F}(q)} u(k) \quad \text{in} \quad y^F(k) = \frac{1}{\hat{F}(q)} y(k) \quad (14.53)$$

- 4: Ocenjevanje parametrov OE f_i in b_i po metodi ocenjevanja parametrov modela ARX na osnovi filtriranih spremenljivk $u^F(k)$ in $y^F(k)$.

- 5: **until** Parametri f_i in b_i ne konvergirajo.
-

14.5.4 Simulacijski primer

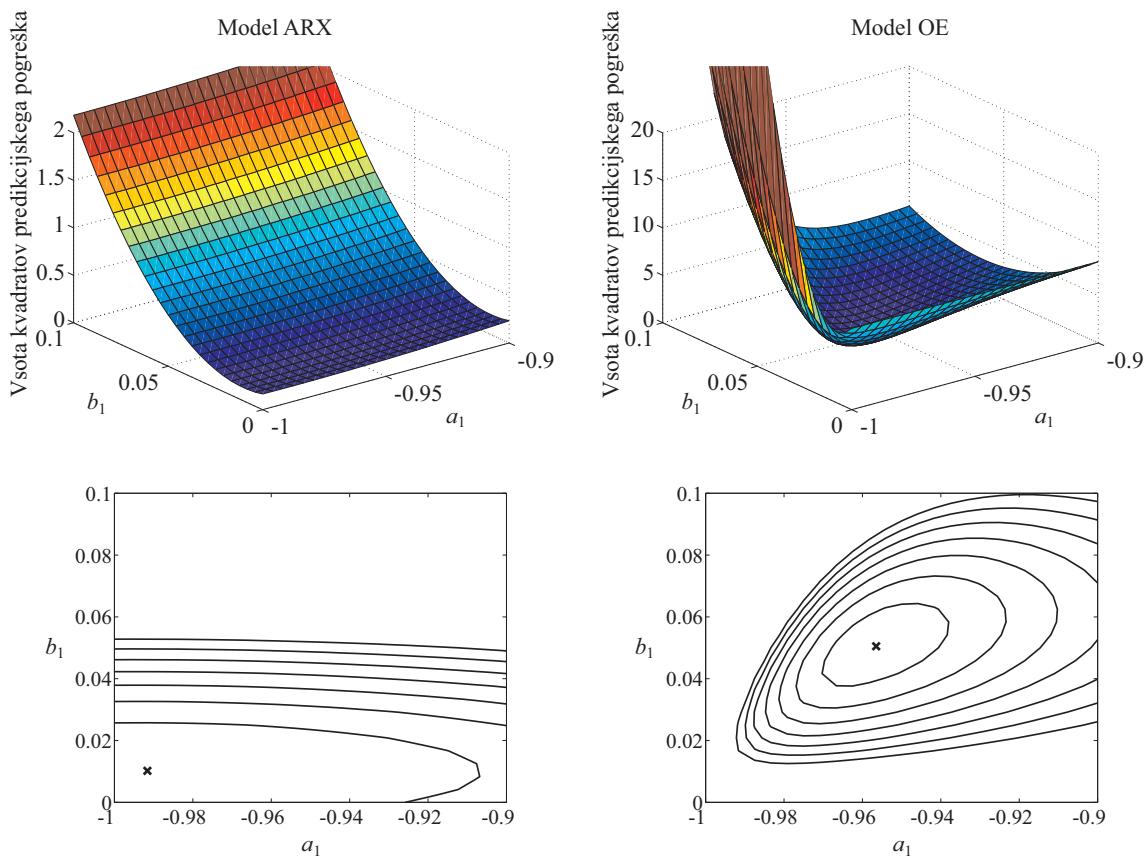
Predpostavimo naslednji proces drugega reda z ojačenjem $K = 1$ in časovnima konstantama $T_1 = 10\text{s}$ in $T_2 = 5\text{s}$:

$$G(s) = \frac{1}{(10s+1)(5s+1)}. \quad (14.54)$$

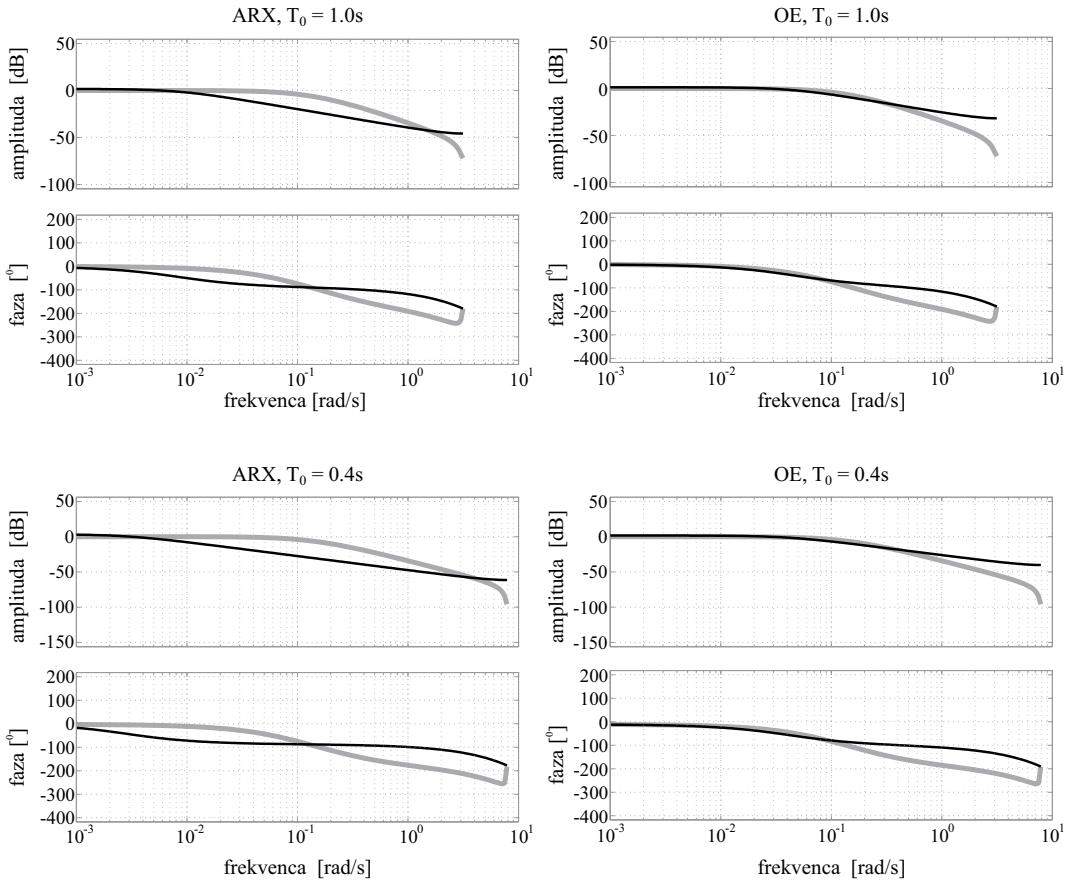
Modelirali smo ga z ARX in OE modelom prvega reda, da bi lahko prikazali lastnosti teh dveh modelov v primeru zmanjšane kompleksnosti, kar je zelo pogosto v praktičnih primerih identifikacije. Za vhodni signal je uporabljen PRBS s 255 vzorci, ki dobro vzbuja celotno frekvenčno področje procesa. Signale pa smo vzorčili s časom vzorčenja $T_0 = 1\text{s}$. Signali so brez motenj.

Slika 14.28 kaže kriterijski funkciji za model ARX in model OE. Za model ARX je v kriterijski funkciji *pogrešek enokoračne predikcije* in zato je funkcija parabola. To lahko vidimo iz eliptičnih kontur te funkcije. V primeru modela OE pa nastopa v kriterijski funkciji *simulacijski pogrešek*, ki ima za posledico kriterijsko funkcijo, ki je odvisna od parametrov. Razlog za močno naraščanje kriterijske funkcije za $a_1 \rightarrow 1$ je v tem, da $a_1 = 1$ predstavlja mejno vrednost, kjer model postane nestabilen.

Optimalni parametri v primeru modela ARX in modela OE so precej različni. Poleg razlike v njihovi vrednosti se razlikujejo tudi v natančnosti ocene (variance v primeru ošumljenih podatkov), ker so oblike kriterijskih funkcij v okolici minimuma zelo različne. Za model OE lahko



Slika 14.28. Kriterijski funkciji za model ARX in model OE (zgoraj) in njuni konturni diagrami (spodaj).



Slika 14.29. Frekvenčna karakteristika za model ARX in model OE pri različnih časih vzorčenja T_0 . Siva črta predstavlja karakteristiko procesa, črna pa modela.

ocenimo oba parametra s podobno natančnostjo, ker je kriterijska funkcija enako občutljiva na oba parametra. To lahko dobro vidimo iz konturnega diagrama, ki je približno krožen okoli optimuma. Za model ARX pa so konture elipse, ki pokažejo, da je kriterijska funkcija zelo občutljiva v smeri parametra b_1 in skoraj neobčutljiva na parameter a_1 . Zaradi tega lahko pričakujemo, da je za model ARX model parameter b_1 ocenjen z veliko stopnjo natančnosti, a_1 pa z nizko.

Razlog za različne optimalne vrednosti parametrov v primeru modela ARX in modela OE lahko najbolje razložimo v frekvenčnem prostoru; glej sliko 14.29 zgoraj. Model ARX na račun boljše aproksimacije v visokofrekvenčnem področju žrtvuje aproksimacijo v srednjefrekvenčnem področju. Model OE pa enakomerno porazdeli natančnost po celotnem frekvenčnem področju. Ker je znižan red modela vedno nenatančen pri visokih frekvencah, je v takih primerih bolj smiselno izbrati model OE, ker je ocena modela ARX pri visokih frekvencah še vedno v splošnem slaba.

Za višje frekvence vzorčanja model ARX poudari visoke frekvence in zelo poslabša natančnost pri nizkih in srednjih frekvencah. Zelo enostaven razlog za to je v tem, da hitrejše vzorčenje potisne Nyquistovo frekvenco na sliki 14.29 proti desni in zaradi tega se natančnost modela ARX preseli še k višjim frekvencam. Na spodnji sliki 14.29 je čas vzorčanja povečan za faktor 2.5, kar rezultira v še večji degradaciji modela ARX. Na primer, ojačenji modela OE pri $T_0 = 1$ s in $T_0 = 0.4$ s sta 1.15 in 1.25. Ojačenji pri enakih vzorčenjih pa sta za model ARX enaki 1.2 in 1.5 pri dejanskem ojačenju $K = 1$.

14.6 Modeli brez povratne zanke

V prejšnjem poglavju smo obravnavali linearne modele s povratno zanko, ki so daleč najpogostejsi tipi modelov. Njihova značilnost so prenosne funkcije, kjer ocenjujemo tako parametre števca kot tudi imenovalca. Prenosne funkcije omogočajo enostaven in učinkovit opis dinamike procesa, preko polov in ničel.

V nadaljevanju se bomo posvetili modelom, ki nimajo povratne zanke. Na ta način so ti modeli manj fleksibilni in zato pogosto zahtevajo veliko večje število parametrov. Na drugi strani pa imajo ti modeli tudi določene prednosti.

Za modele brez povratne zanke velja naslednje:

- ta razred modelov spada med *modele z izhodnim pogreškom* in v razred *modelov, ki so linearni v parametrih*. V prejšnjih izvajanjih smo ugotovili, da je samo model ARX linearen v parametrih in je zato tako pogosto v uporabi, čeprav so predpostavke glede šuma v njegovem primeru pogosto neupravičene. Zaradi tega dobimo nekonsistentne ocene parametrov modela. Vsega tega v primeru modelov FIR ni.
- *Stabilnost* je zagotovljena v vsakem primeru, ker ocenjeni parametri ne določajo polov modela.
- Modeli so zelo *enostavni*. Struktura modela je mnogo enostavnejša kot v primeru modelov s povratno zanko. Ti modeli so primerni takrat, ko ne zahtevamo velike natančnosti, ampak enostavnost modela. Pogosto se uporabljajo pri aplikacijah adaptivnega filtriranja v komunikacijah.

V nadaljevanju si bomo ogledali model s končnim impulznim odzivom (*ang. finite impulse response*).

14.6.1 Model FIR

Model končnega impulznega odziva (FIR) je najenostavnjeji linearni model. Vsi ostali modeli rezultirajo v neskončnem impulznem odzivu, ker imajo vsebovano povratno zanko z izhoda modela. FIR model je enostaven filter drsečega povprečja vhodnega signala. Na ta način je izhod modela utežena vsota zakasnejenih vhodov kar je prikazano na sliki 14.30.

V polinomski obliki lahko model FIR zapišemo kot

$$y(k) = B(q)u(k) + v(k), \quad (14.55)$$

kar prikazuje tudi slika 14.31. V obliki diferenčne enačbe pa dobimo

$$y(k) = b_1u(k-1) + b_2u(k-2) + \dots + b_mu(k-m) + v(k). \quad (14.56)$$

Beli šum je dodan izhodu modela, kot to definira enačba 14.55. Zaradi tega spada model FIR v razred modelov z izhodnim pogreškom. Optimalni prediktor je definiran na naslednji način

$$\hat{y}(k|k-1) = B(q)u(k), \quad (14.57)$$

in je linearen v parametrih. Ker nimamo povratne zanke v modelu, je mogoče imeti linearno parametriran model z izhodnim pogreškom. Ker v prediktorju nastopajo samo vhodi, je optimalni enokoračni prediktor enak simulaciji, tako kot v primeru modela OE.

Motivacija za model FIR izhaja iz dejstva, da lahko vsak izhod linearnega sistema zapišemo kot konvolucijsko vsoto:

$$y(k) = \sum_{i=1}^m g_i u(k-i) = g_1 u(k-1) + g_2 u(k-2) + \dots + g_m u(k-m), \quad m \rightarrow \infty, \quad (14.58)$$

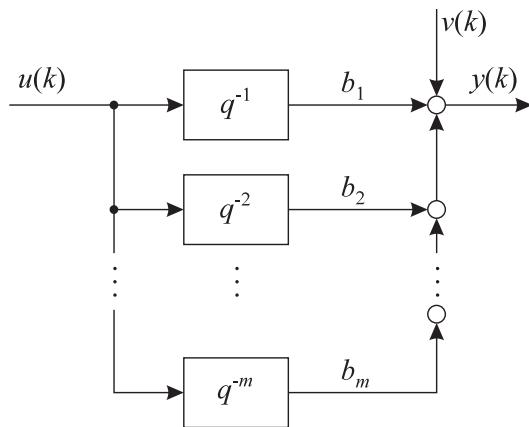
kjer so g_i uteži impulznega odziva. Prvi člen $g_0 u(k) = 0$, ker predpostavimo, da proces nima direktne poti z vhoda na izhod in je strogo pravi. Lahko vidimo, da je model FIR samo aproksimacija konvolucijske vsote, saj je število členov v modelu FIR končno. To pomeni, da vzamemo samo prvih $m+1$ členov neskončne vrste. Ker gredo pri stabilnih sistemih koeficienti g_i proti nič, če gre $i \rightarrow \infty$, je tak aproksimator mogoč. Že v primeru mejno stabilnih sistemov je taka aproksimacija nemogoča, saj gre g_i proti konstantni vrednosti.

Model FIR je linearen v parametrih. Ocenimo jih lahko z metodo najmanjših kvadratov. Vektor parametrov in regresijska matrika sta enaka

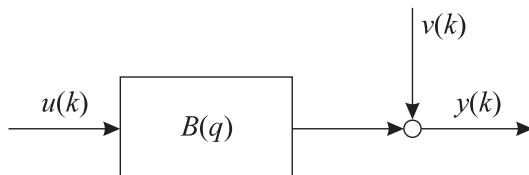
$$\underline{\theta} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}, \quad \underline{X} = \begin{bmatrix} u(m) & u(m-1) & \cdots & u(1) \\ u(m+1) & u(m) & \cdots & u(2) \\ \vdots & \vdots & & \vdots \\ u(N-1) & u(N-2) & \cdots & u(N-m) \end{bmatrix}. \quad (14.59)$$

Enostavnost, linearnost v parametrih in struktura izhodnega pogreška so prednosti modela FIR. Slabost pa je predvsem v velikem številu koeficientov, ki so potrebni za ustrezno aproksimacijo. To je razvidno iz enačbe 14.58, kjer moramo izbrati zadostno število členov g_i , $i = 1, \dots, m$, da bo aproksimacija primerno natančna.

Kako velik naj bo m si lahko razložimo na naslednjem primeru. Čas vzorčenja naj bo izbran na 1/5 najpočasnejše časovne konstante T procesa. Potem je pri približnem času umiritev T_{95} (čas potreben, da se proces ustali v 95% končne vrednosti ali $T_{95} = 3T$) odziv vzorčen 15 krat. To pomeni, da je primerna izbira za m enaka 15. Če model uporabljamamo samo za vodenje, bi lahko m izbrali še manjši, saj v statičnem področju ne potrebujemo take natančnosti. V primeru simulacije pa bi izbira pod $m = 15$ zelo poslabšala kvaliteto modela.



Slika 14.30. Model FIR v filtrski predstavitev.



Slika 14.31. Model FIR v polinomski predstavitev.

14.7 Nekaj pomembnih vidikov identifikacije

V nadaljevanju si bomo na kratko pogledali nekaj pomembnih vidikov pri identifikaciji, ki jih do sedaj zaradi enostavnosti nismo obravnavali.

14.7.1 Konsistentnost

Za konsistentnost ocenjavanja parametrov modela moramo izbrati za model procesa prenosno funkcijo $\hat{G}(q)$, ki ima strukturo enako kot proces. Drugače ne moremo popolnoma opisati procesa $G(q)$.

V poglavju 14.5.1 smo obravnavali konsistentnost modela ARX. Parametri modela ARX so ocenjeni konsistentno samo takrat, kadar lahko šum ustrezno modeliramo s prenosno funkcijo $1/A(q)$. To se v praksi zgodi zelo redko. Če so modeli šumov pravilno stukturirani in so na ta način sposobni opisati dejanski šum, potem so parametri v takih primerih ocenjeni konsistentno. Ker pa je seveda verodostojno poznavanje šuma redko, je v praksi bolj pomembno to, da poskušamo uporabiti metode, ki kljub neustreznem modeliranju šuma dajo konsistentno oceno parametrov.

Vse strukture modelov, ki imajo neodvisno vhodno prenosno funkcijo $G(q)$ in prenosno funkcijo šuma $H(q)$, dovoljujejo konsistentno oceno parametrov prenosne funkcije tudi, če je model šuma neustrezen. "Neodvisno parametrirana" prenosna funkcija pomeni, da $G(q)$ in $H(q)$ ne vsebuje skupnih parametrov. V primeru razreda modelov z izhodnim pogreškov so parametri neodvisni, to je v primeru modela OE ali modela BJ. V primeru modelov s posplošenim pogreškom, to je modela v primeru ARX in modela ARMAX, so parametri odvisni, ker imata $G(q)$ in $H(q)$ skupni polinom $A(q)$. Neodvisnost parametrov je osnovna prednost modela OE in modela BJ pred modelom ARX in modelom ARMAX. Je pa seveda jasno, da je parametre modela OE in modela BJ težje oceniti.

14.7.2 Relacija med modelom šuma in filtriranjem

Predfiltriranje ali vključitev pravega šumnega modela je ekvivalenten problem. Če predikcijski pogrešek v enačbi 14.26 napako filtriramo z filtrom $L(q)$, potem dobimo

$$e_F(k) = \frac{L(q)}{H(q)} (y(k) - G(q)u(k)) . \quad (14.60)$$

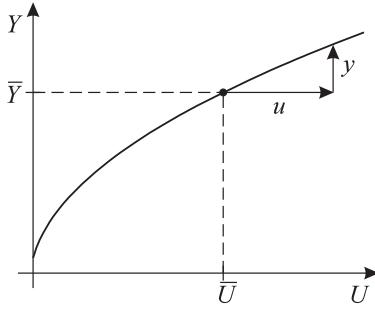
Vpliv šumnega modela $H(q)$ lahko izničimo z izbiro filtra $L(q)$.

To relacijo lahko intuitivno razložimo takole. Pri frekvencah, kjer je amplitudni del šumne prenosne funkcije majhen, lahko pričakujemo majhne vrednosti šuma, kar pomeni, da je razmerje med koristnim signalom in šumom veliko. Zaradi tega je potrebno to področje v identifikaciji bolj utežiti, saj so podatki v tem podočju zelo dobri. Na drugi strani pa pri velikih vrednostih amplitudnega dela šumega filtra pričakujemo nizko razmerje med signalom in šumom. Tako področje je potem manj primerno za identifikacijo modela in je zato manj uteženo. Torej lahko namesto šumnega modela uporabimo filter, ki poveča ali zmanjša uteževanje posameznega frekvenčnega področja. Če je šumnii model nizkopasovni, je potem ustezni filter visokopasovni in obratno.

Če povzamemo, potem lahko efekt obstoječega šumnega modela $H(q)$ *odstranimo* z dodatnim filtriranjem predikcijskega pogreška

$$L(q) = H(q) . \quad (14.61)$$

Ta relacija nam pomaga razumeti algoritem **ponavljanjočih najmanjših kvadratov za ocenjevanje** parametrov za model OE v poglavju 14.5.3. Lahko ga razložimo tudi takole. Po modelu ARX je predikcijski pogrešek enak



Slika 14.32. Podatki za linearno identifikacijo morajo ležati blizu delovne točke \bar{U} , \bar{Y} .

$$e(k) = A(q)(y(k) - G(q)u(k)) . \quad (14.62)$$

Filteriranje s filtrom $L(q) = 1/A(q)$ pokrajša učinek šumnega modela, pri modelu ARX in vodi do predikcijskega pogreška modela OE

$$e_F(k) = y(k) - G(q)u(k) . \quad (14.63)$$

Seveda pa moramo ocenjevanje modela ARX in filtriranja s filtrom $L(q)$ ponavljati do konvergencije, ko začetni parametri imenovalca $A(q)$ modela ARX konvergirajo k imenovalcu $F(q)$ modela OE.

14.7.3 Delovna točka

Podatki za linearno identifikacijo so tipično merjeni okoli delovne točke \bar{U} in \bar{Y} . Slika 14.32 kaže delovno točko, ki leži na statični karakteristiki procesa. Če želimo aproksimirati proces z linearnim modelom, potem moramo proces vzbujati tako, da so odstopanja od delovne točke u in y zadosti majhna glede na stopnjo nelinearnosti procesa pri določeni delovni točki \bar{U} in \bar{Y} . Pri identifikaciji linearnega modela nastane problem, ker merimo $U(k)$ in $Y(k)$, za identifikacijo pa potrebujemo odstopanja od delovne točke $u(k)$ in $y(k)$. Če zapišemo deviacije od delovne točke procesa kot $u(k) = U(k) - \bar{U}$ in $y(k) = Y(k) - \bar{Y}$, dobimo naslednji diferenčni enačbi [53]:

$$\begin{aligned} (Y(k) - \bar{Y}) + a_1(Y(k-1) - \bar{Y}) + \dots + a_m(Y(k-m) - \bar{Y}) &= \\ = b_1(U(k-1) - \bar{U}) + \dots + b_m(U(k-m) - \bar{U}) , \end{aligned} \quad (14.64)$$

ki jih lahko zapišemo tudi kot

$$\begin{aligned} Y(k) = & - a_1 Y(k-1) - \dots - a_m Y(k-m) \\ & + b_1 U(k-1) + \dots + b_m U(k-m) \\ & + \underbrace{(1+a_1+\dots+a_m)\bar{Y} - (b_1+\dots+b_m)\bar{U}}_C . \end{aligned} \quad (14.65)$$

Odmik (*ang. offset*) C vsebuje informacijo o delovni točki \bar{U} , \bar{Y} . Imamo tri možnosti za obravnavo tega odmika:

- odstranitev vrednosti delovne točke iz podatkov in procesiranje odmikov,
- ocenjevanje odmika,
- razširitev modela.

- Če poznamo delovno točko, potem lahko deviacije izračunamo eksplisitno kot

$$u(k) = U(k) - \bar{U}, \quad y(k) = Y(k) - \bar{Y}, \quad (14.66)$$

in potem $u(k)$ in $y(k)$ uporabimo v identifikaciji.

- Če delovna točka ni poznana, jo lahko aproksimiramo z $U(k)$ in $Y(k)$

$$\bar{U} = \frac{1}{N} \sum_{i=1}^N U(i), \quad \bar{Y} = \frac{1}{N} \sum_{i=1}^N Y(i), \quad (14.67)$$

in potem to aproksimirano delovno točko uporabimo v prvem pristopu.

- Odmik lahko izračunamo eksplisitno, če izvedemo oceno parametrov na osnovi enačbe 14.65.

Za model ARX se regresijski vektor razširi v

$$\underline{x} = [U(k-1) \ \dots \ U(k-m) \ -Y(k-1) \ \dots \ -Y(k-m) \ 1]^T, \quad (14.68)$$

$$\underline{\theta} = [b_1 \ \dots \ b_m \ a_1 \ \dots \ a_m \ C]^T. \quad (14.69)$$

Ravnotako se razširitev izvede pri ostalih strukturah modelov. Oceniti moramo še en dodaten parameter, kar se v regresijskem vektorju odraža kot konstanta 1.

- Odmik lahko eliminiramo z diferenciacijo podatkov. To lahko storimo s predfiltriranjem podatkov $U(k)$ in $Y(k)$ s filtrom

$$L(q) = 1 - q^{-1}, \quad (14.70)$$

kar povzroči, da dobimo razlike med sosednjimi vzorci $U(k) - U(k-1)$ in $Y(k) - Y(k-1)$, ali na način, da razširimo šumni filter z integratorjem

$$\tilde{H}(q) = \frac{1}{1 - q^{-1}} H(q). \quad (14.71)$$

Največja pomanjkljivost tega pristopa je v tem, da ta pristop ojači visoke frekvence. To pa povzroči težave pri ocenjevanju parametrov modela.

14.8 Povzetek

Splošna navodila pri linearni identifikaciji bi lahko strnili v naslednje:

- če imamo malo informacije o procesu, potem začnemo s strukturo modela ARX, ki ga enostavno ocenimo z metodo najmanjših kvadratov.
- Če to ne da zadosti dobrih rezultatov, potem nadgradimo strukturo v model ARMAX ali model OE.
- Razširitev na ostale strukture je smiselna samo, če z enostavnimi strukturami ne moremo doseči želene kvalitete modela.
- Predfiltriranje ali posebne modele šuma lahko izkoristimo, da oblikujemo podatke glede na frekvenčno karakteristiko modela.
- Identifikacija modela iz zaprtozančnih podatkov je smiselna samo takrat, ko je edini cilj modeliranja vodenje.

Za razširitev na nelinearne strukture sta model ARX in model OE zelo pomembna. Modeli FIR so ravno tako uporabni v nelinearni identifikaciji. Bolj kompleksni modeli, kot npr. v model ARMAX, model ARARX in model BJ se redko uporabljajo v nelinearni identifikaciji. Razlog za to je v kompleksnosti identifikacije. Fleksibilnost in variančna napaka sta v nelinearnih primerih precejšnja omejitev. Zaradi tega se v nelinearnih problemih navadno uporablja samo zelo enostavne strukture.

15. Identifikacija nelinearnih dinamičnih sistemov

V tem poglavju je predstavljen pregled konceptov identifikacije nelinearnih dinamičnih sistemov. Obravnavani so osnovni pristopi in njihove lastnosti, ki so neodvisni od izbire arhitekture modela in moderni pristopi z nevno-mehkimi modeli.

Najprej je v podpoglavlju 15.1 prikazan prehod z linearne na nelinearno identifikacijo. Potem pa je v podpoglavlju 15.2 prikazan princip identifikacije z zunanjim dinamikom. V poglavju 15.3 je podan pregled načrtovanja meritev v primeru nelinearnih procesov in v 15.4 problematika določevanja dinamičnega reda iz meritev. Na koncu je podan povzetek.

15.1 Razširitev linearne identifikacije na nelinearno identifikacijo

Zaradi enostavnosti bodo vsi problemi v tem poglavju formulirani za sistem z enim vhodom in enim izhodom (*ang. single-input single-output* - SISO). Razširitev na multivariabilne sisteme je enostavna. Nadalje smo predpostavili, da imata vhod in izhod enako število zakasnitev (red dinamike) m in da sistem nima direktne povezave med vhodom in izhodom, kar pomeni, da vhod $u(k)$ ne more direktno vplivati na izhod $y(k)$. Predpostavili smo tudi sistem brez mrtvega časa. Vse te predpostavke za naše nadaljevanje nimajo posebnega pomena, gre samo za to, da poenostavimo zapis.

Najenostavnejši linearni vhodno-izhodni model v diskretnem času je model ARX (*autoregresivni model z zunanjim vhodom*, *ang. AutoRegressive with eXogenous input*) ali model s poslošenim pogreškom; glej poglavje 14.5.1, [78]. Optimalno predikcijo izhoda modela m -tega reda dobimo z modelom ARX kot sledi

$$\hat{y}(k) = b_1 u(k-1) + \dots + b_m u(k-m) - a_1 y(k-1) - \dots - a_m y(k-m) . \quad (15.1)$$

Ta princip lahko iz linearnega zapisa razširimo v nelinearni model NARX (nelinearni model ARX) na enostaven način tako, da zamenjamo linearne relacije v enačbi 15.1 z neznano nelinearno funkcijo $f(\cdot)$ in dobimo

$$\hat{y}(k) = f(u(k-1), \dots, u(k-m), y(k-1), \dots, y(k-m)) . \quad (15.2)$$

Če govorimo o modelu, ki ga bomo uporabili za namene vodenja, potem lahko stvari še dodatno poenostavimo v manj splošno obliko

$$\hat{y}(k) = b_1 u(k-1) + \tilde{f}(u(k-2), \dots, u(k-m), y(k-1), \dots, y(k-m)) , \quad (15.3)$$

ki je izbrana zato, ker je afina (linearna z dodano enosmerno komponento) v regulirnem signalu $u(k-1)$. Iz tega zapisa lahko direktno dobimo inverzni regulirni zakon kot

$$u(k) = [r(k+1) - \tilde{f}(u(k-1), \dots, u(k-m+1), y(k), \dots, y(k-m+1))] / b_1 , \quad (15.4)$$

kjer je $r(k+1)$ referenčni signal za naslednji časovni trenutek.

Primerjava med identifikacijo linearnih in nelinearnih modelov kaže na to, da se problem *ocene parametrov*, b_i in a_i razširi na problem *aproksimacije funkcije* $f(\cdot)$. Tak razred nelinearnih modelov imenujemo razred modelov z *zunanjo dinamiko*.

Model NARX, ki je zapisan v enačbi 15.2, in druge strukture modelov lahko opišejo zelo velik razred procesov [14], [75]. Imajo pa seveda tudi določene omejitve, kar si bomo ogledali v naslednjem poglavju.

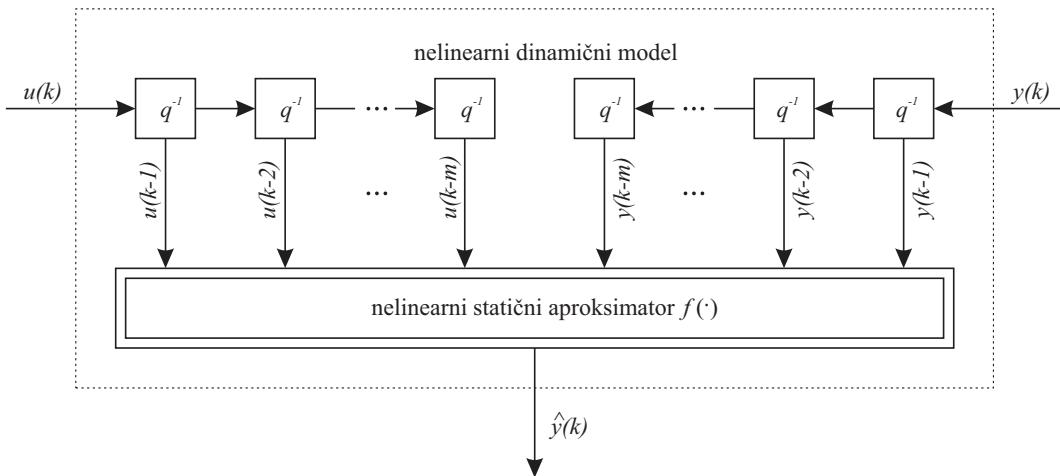
15.2 Model z zunanjim dinamikom

Strategija zunanje dinamike je najpogosteji pristop za identifikacijo nelinearnih dinamičnih modelov. Temelji na vhodno-izhodnem modelu, ki je predstavljen v enačbi 15.2. Ime "zunanja dinamika" izvira iz preprostega dejstva, da lahko običajno tak nelinearni sistem razdelimo na dva ločena dela: na filter stanj vhodnega in izhodnega signala in na nelinearni statični del; glej sliko 15.1. V principu lahko za aproksimacijo nelinearnega statičnega modela $f(\cdot)$ izberemo katerokoli arhitekturo. Problem pa seveda nastane, ker mora aproksimator procesirati preslikavo, ki je lahko zelo visokega reda, vsaj v primeru sistemov, ki so višjega reda. Običajno so filtri realizirani kar z enostavno zakasnitvijo q^{-1} . V tem primeru se taka struktura imenuje TDL (ang. *Tapped-Delay Lines*) in če izberemo za aproksimator funkcije $f(\cdot)$ nevronsko mrežo, potem se tak model imenuje TDNN (*Time-Delayed Neural Network*) [77, 89]. Večina lastnosti modelov z zunanjim dinamikom je neodvisna od izbire aproksimatorjev. Te splošne lastnosti si bomo ogledali v naslednjem podpoglavlju.

15.2.1 Metoda modeliranja z zunanjim dinamikom

Razumevanje metode modeliranja z zunanjim dinamikom in njihove lastnosti nam omogočajo, da lahko na ta način obravnavamo veliko število najrazličnejših praktičnih problemov. Strukturo filtra in statičnega dela na sliki 15.1 je v vsakem primeru potrebno ustreznou izbrati. To pa lahko storimo samo, če dobro razumemo metodo.

Relacija med vhodno-izhodnim prostorom in vhodnim prostorom aproksimatorja. Ena od pomembnih stvari, ki jo je potrebno dobro spoznati je, kako vhodni signal $u(k)$ in izhod



Slika 15.1. Model z zunanjim dinamikom: model lahko razdelimo na statični aproksimator in na zunanji niz filtrov, ki je v tem primeru realiziran s strukturo TDL.

$y(k)$ s časom določata distribucijo podatkov v vhodnem prostoru aproksimatorja $f(\cdot)$, ki ga določata vhod in izhod procesa. Za primere prvega reda lahko ta prostor tudi vizualiziramo, kar pa seveda ni možno v primeru višjih dimenzij. Za primer si bomo pogledali Hammersteinov model, ki ga določata statična nelinearnost v seriji s procesom prvega reda

$$y(k) = 0.1 \arctan(u(k-1)) + 0.9 y(k-1). \quad (15.5)$$

Proces prikazuje slika 15.4a levo. Za vhodne signale, ki so stopničaste oblike, kot je prikazano na sliki 15.2a, je vhodno-izhodni prostor zapolnjen s podatki, ki so v vertikalnih linijah pri konstantnem $u(k)$. Za signale sinusne oblike z nizko in visoko frekvenco pa je distribucija podatkov prikazana na slikah 15.2b in 15.2c. Opazimo lahko naslednje:

- Vseh vhodov aproksimatorja ne moremo izbirati poljubno. Samo vhod $u(k)$ izberemo direktno, vsi ostali vhodi aproksimatorja pa so posledica tega.
- Nižja, ko je frekvence vhodnega signala, bližje so podatki statični nelinearnosti (ravnotežni točki) procesa.
- Naravno je tudi, da je število (distribucija) točk v bližini statične nelinearnosti bistveno večja kot je izven ravnotežnih točk.
- Potreben je zelo dinamičen vhodni signal, da lahko s podatki pokrijemo široko območje vhodnega prostora.

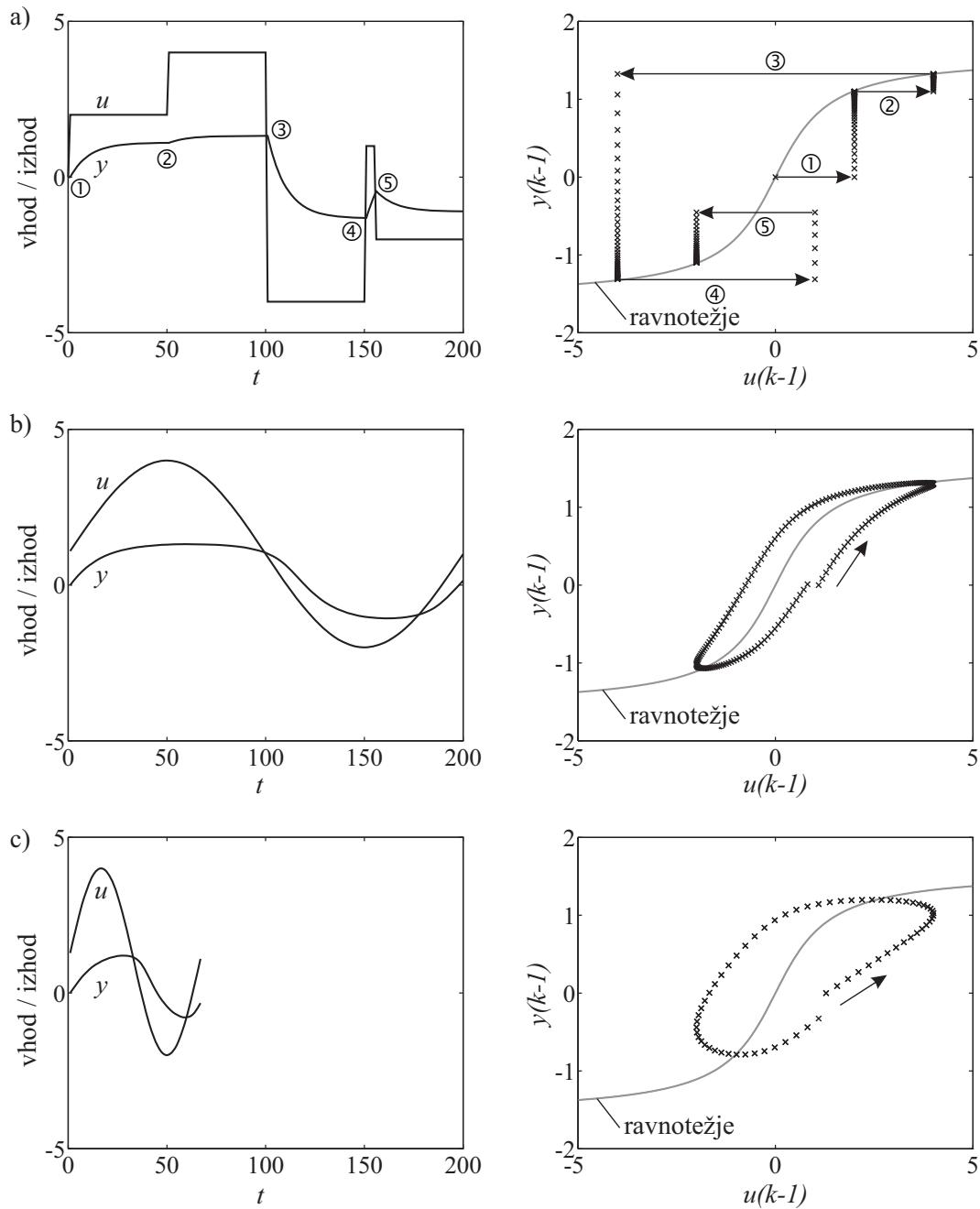
Vse to kaže na to, da mora biti signal vzbujanja $u(k)$ izbran izjemno pazljivo, da na ta način pridobimo čim več informacije o sistemu.

Za dinamične sisteme višjega reda postane tudi vhodni prostor aproksimatorja večdimenzionalen, distribucija podatkov pa ostane bolj ali manj enaka. Vseeno pa je treba pripomniti, da so za dimenzijske sisteme, kjer je $m \geq 2$, zakasnjeni izhodi $y(k-i)$ za zakasnitve $i = 1, 2, \dots, m$, zelo korelirani. To je prikazano na sliki 15.3a. S povečevanjem časa vzorčenja se ta korelacija zmanjšuje, vendar so podatki vedno razporejeni okoli diagonale vhodnega prostora $y(k-1)-y(k-2)-\dots-y(k-m)$; glej sliko 15.3b. To je posledica izbire časa vzorčenja, ki mora biti primerno majhen, da lahko opišemo spremembe izhoda procesa zadosti natančno, predvsem prehodne pojave. Posledično pa seveda sledi, da je $y(k-i) \approx y(k-i-1)$. Podobno velja tudi za vhodni signal. Iz vsega tega sledi, da so v primeru pristopa z zunanjim dinamikom velika področja vhodnega prostora, ki so brez podatkov. V principu jih ne moremo doseči, ne glede na izbiro vhodnega signala $u(k)$.

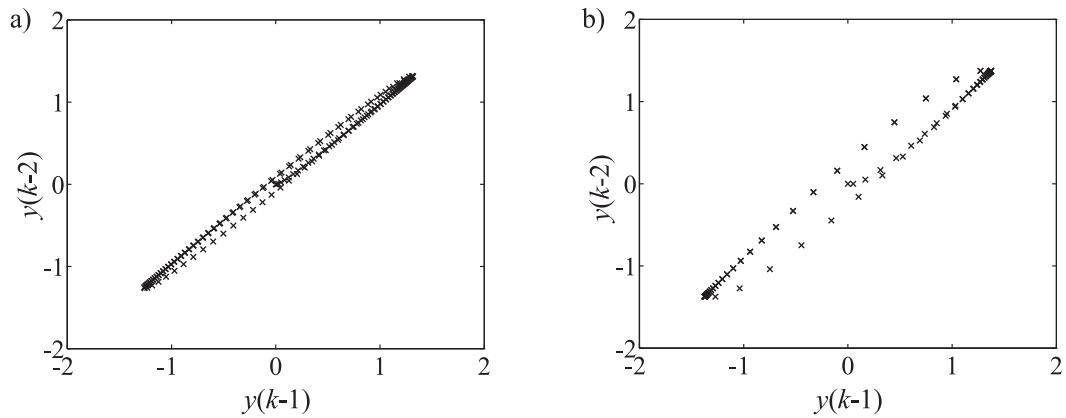
Hiperpovršina enokoračne predikcije. Zanima nas oblika hiperpovršine funkcije $f(\cdot)$ v enačbi 15.2, ki jo dejansko modeliramo z nelinearnim statičnim aproksimatorjem. Za sistem prvega reda jo lahko vizualiziramo kot $y(k) = f(u(k-1), y(k-1))$. Funkcija $f(\cdot)$ je enokoračni prediktor, ker preslika prejšnje vrednosti vhodov in izhodov v trenutni izhod modela. Slika 15.4 primerja enokoračne predikcije za Hammersteinov in Wienerjev model prvega reda. Hammersteinov model na sliki 15.4a levo je zapisan v enačbi 15.5. Wienerjev model pa je dobljen tako, da statično nelinearnost prestavimo na konec, za dinamični blok, kot je to prikazano na sliki 15.4a desno. Na ta način je enačba enaka

$$y(k) = \arctan(0.1 u(k-1) + 0.9 \tan(y(k-1))). \quad (15.6)$$

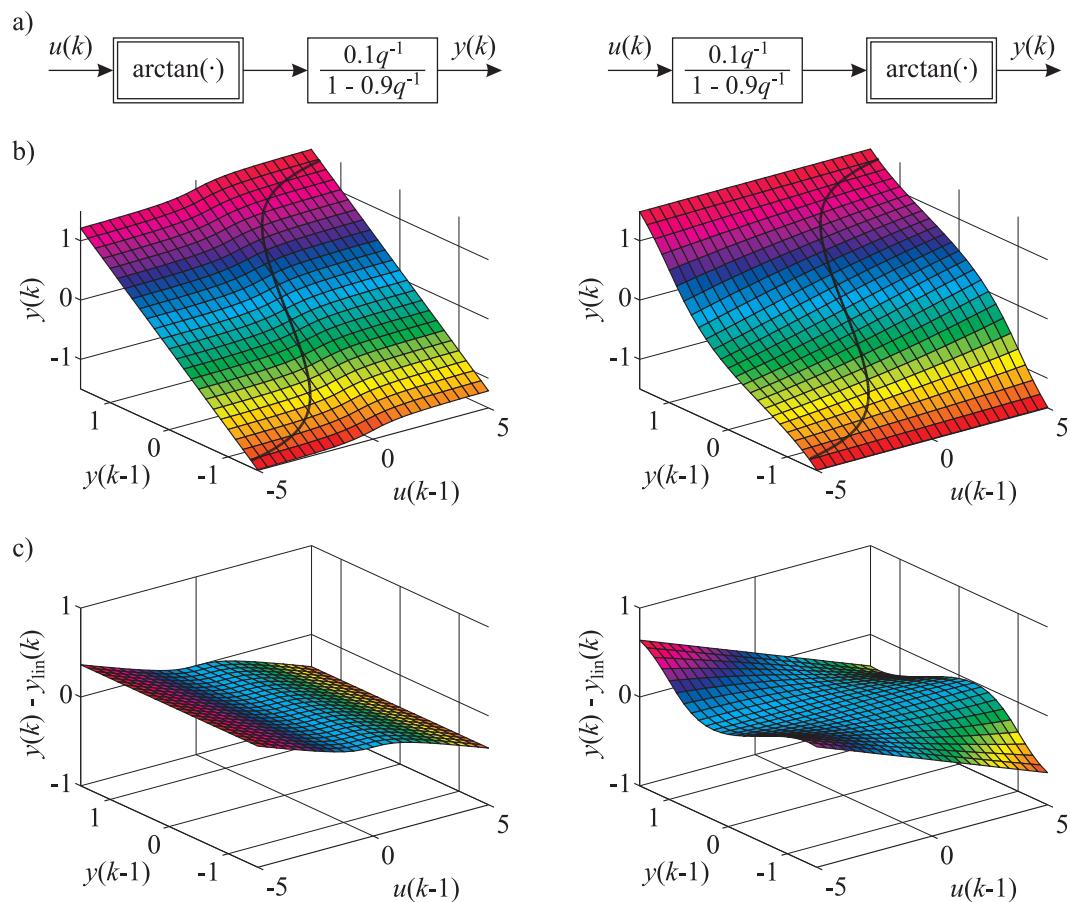
Predstavitev enokoračne predikcije linearnega sistema je površina, kjer so gradienti definirani z linearnimi parametri b_i in $-a_i$. Za primer Hammersteinovega in Wienerjevega sistema so enokoračne predikcije in predikcijska površina predstavljeni na sliki 15.4b. Na ta način je predstavljena vsa informacija o sistemu. Ta zapis predstavlja lineariziran model v delovni točki (u_0, y_0) iz česar sledi



Slika 15.2. Odvisnost med vhodno-izhodnimi podatki sistema (levo) in vhodni prostor aproksimatorja pri pristopu z zunanjim dinamikom (desno): a) vzbujanje s signali z obliko stopnice, b) sinusni signal vzbujanja z nizko frekvenco, c) sinusni signal vzbujanja z visoko frekvenco. Priporočimo, da točke, ki so daleč od ravnotežnega stanja lahko dosežemo samo z velikimi vhodnimi stopnicami. Območje nad statično nelinearnostjo predstavlja dinamiko z zmanjšanjem vhodnega signala, območje pod njo, pa predstavlja dinamiko, ko smo vhodni signal povečali.



Slika 15.3. Korelacija med zakasnjenimi izhodnimi spremenljivkami za sistem drugega reda za a) majhen in b) velik čas vzorčenja.



Slika 15.4. Hammersteinov (levo) in Wienerjev (desno) model: a) bločna shema modela, b) prikaz enokoračne predikcije s površino in staticne krivulje z debelo črto, c) površina, ki prikazuje razlike med meritvami in izhodom modela.

$$\Delta y(k) = b_1 \Delta u(k-1) - a_1 \Delta y(k-1), \quad (15.7)$$

kjer sta $\Delta u(k) = u(k) - u_0$ in $\Delta y(k) = y(k) - y_0$ odmika od delovne točke, odvoda pa dobimo na naslednji način:

$$\begin{aligned} b_1 &= \left. \frac{\partial f(u(k-1), y(k-1))}{\partial u(k-1)} \right|_{(u_0, y_0)}, \\ a_1 &= - \left. \frac{\partial f(u(k-1), y(k-1))}{\partial y(k-1)} \right|_{(u_0, y_0)}. \end{aligned} \quad (15.8)$$

V primeru Hammersteinovega modela, iz enačbe 15.5, sta odvoda enaka

$$\begin{aligned} b_1 &= \left. \frac{\partial f(u(k-1), y(k-1))}{\partial u(k-1)} \right|_{(u_0, y_0)} = \frac{0.1}{1 + u_0^2(k-1)}, \\ a_1 &= - \left. \frac{\partial f(u(k-1), y(k-1))}{\partial y(k-1)} \right|_{(u_0, y_0)} = 0.9. \end{aligned} \quad (15.9)$$

Vidimo lahko, da model izkazuje močno nelinearnost, saj se ojačenje spreminja med 1 pri $u_0 = 0$ in $1/26$ pri $u_0 = \pm 5$. Kljub temu pa sta predikcijski površini le rahlo nelinearni. To ni posledica konkretnega primera, temveč dejstva, da zapis v smislu predikcijske površine vedno rezultira v le rahko nelinearni karakteristiki. V nadaljevanju bomo razložili zakaj je temu tako. Enokoračna predikcijska površina v primeru Wienerjevega modela je bolj nelinearna kot površina v primeru Hammersteinovega modela. Modela imata enaki statični karakteristiki in različne izven ravnotežne regije; glej sliko 15.4b.

Če od enokoračnih površin Hammersteinovega in Wienerjevega modela odštejemo enokoračno predikcijsko površino linearne dela modela $y(k) = 0.1 u(k-1) + 0.9 y(k-1)$ dobimo površino, ki predstavlja razliko in je enaka nelinearnosti. To je prikazano na sliki 15.4c. Očitno je v primeru Hammersteinovega modela ta razlika nelinearna samo v dimenziji spremenljivke $u(k-1)$ in linearna v spremenljivki $y(k-1)$ medtem, ko je v primeru Wienerjevega modela nelinearna v obeh vhodih modela $u(k-1)$ in $y(k-1)$. To je seveda v skladu z njuno naravo, saj v primeru Wienerjevega modela nelinearnost vpliva na obe spremenljivki.

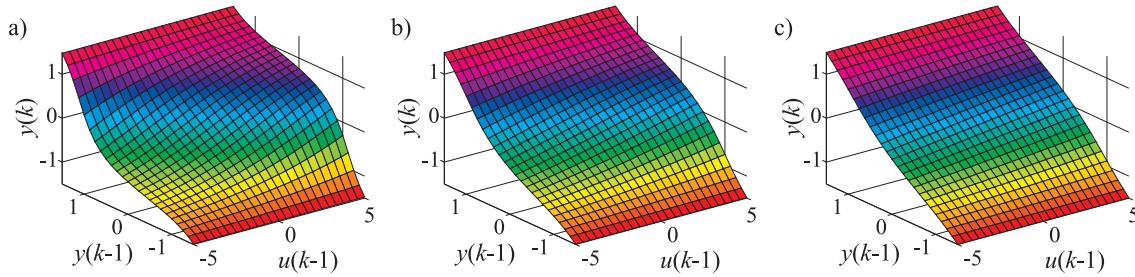
Iz gornjega lahko povzamemo, da je Hammersteinov model po svoji zgradbi veliko preprostejši, ima pa enako opisno moč kot Wienerjev model. Hammersteinov model ima šibkejo nelinearno enokoračno predikcijo in je nelinearen samo v eni spremenljivki, $u(k-i)$, $i = 1, 2, \dots, m$, in je popolnoma linearen v zakasnjeni spremenljivki izhoda modela $y(k-i)$. To seveda močno zmanjša težave pri dimenziji problema.

Pomemben zaključek tega izvajanja je ta, da imajo tudi zelo nelinearni sistemi relativno šibko enokoračno predikcijsko površino ali funkcijo $f(\cdot)$. To dejstvo močno vpliva na široko uporabo *lokalno linearnih modelov z zunanjim dinamikom*.

Učinek časa vzorčenja. V nadaljevanju bomo osvetlili dejstvo relativno šibke nelinearnosti v primeru predstavitev z enokoračno predikcijo tudi takrat, ko gre za sisteme, ki imajo močne nelinearnosti. Funkcija, ki jo modeliramo in predstavlja nelinearni zapis sistema, $f(\cdot)$ je enokoračni prediktor. Če seveda želimo narediti napoved za več korakov v prihodnost, moramo narediti večkratno predikcijo. Za sistem prvega reda je enokoračna predikcija dana z

$$y(k+1|k) = f(u(k), y(k)). \quad (15.10)$$

Dvakoračno napoved pa določimo tako, da uporabimo enokoračno napoved in dobimo



Slika 15.5. Enokoračna predikcijska površina za Wienerjev sistem na sliki 15.4 (desno) za različne čase vzorčenja: $T_{95}/T_0 = \text{a)} 15, \text{b)} 30, \text{c)} 60$.

$$y(k+2|k) = f(u(k+1), y(k+1)) = f[u(k+1), f(u(k), y(k))] . \quad (15.11)$$

To pomeni, da so funkcije gnezdenje. S povečevanjem koraka predikcije se povečuje število gnezdenj funkcijskih funkcij. Zaradi enostavnosti bomo predpostavili, da je vhod konstanten, kot je to v primeru stopničaste funkcije vzbujanja, to je, $u(k) = u(k+1) = \dots = u(k+h)$. Potem enačba 15.11 postane $y(k+2|k) = f(u(k), f(u(k), y(k)))$, izhod modela pa lahko za h korakov v prihodnost določimo s h -kratnim gnezdenjem funkcije $f(\cdot)$

$$y(k+h|k) = \underbrace{f(\dots f(u(k), f(u(k), y(k))) \dots)}_{h\text{-krat}}, \quad (15.12)$$

za oceno odziva na stopnico za $h = T_{95}/T_0$ korakov v prihodnost, kjer je T_{95} čas umiritve v 5% interval končne vrednosti in je T_0 čas vzorčenja.

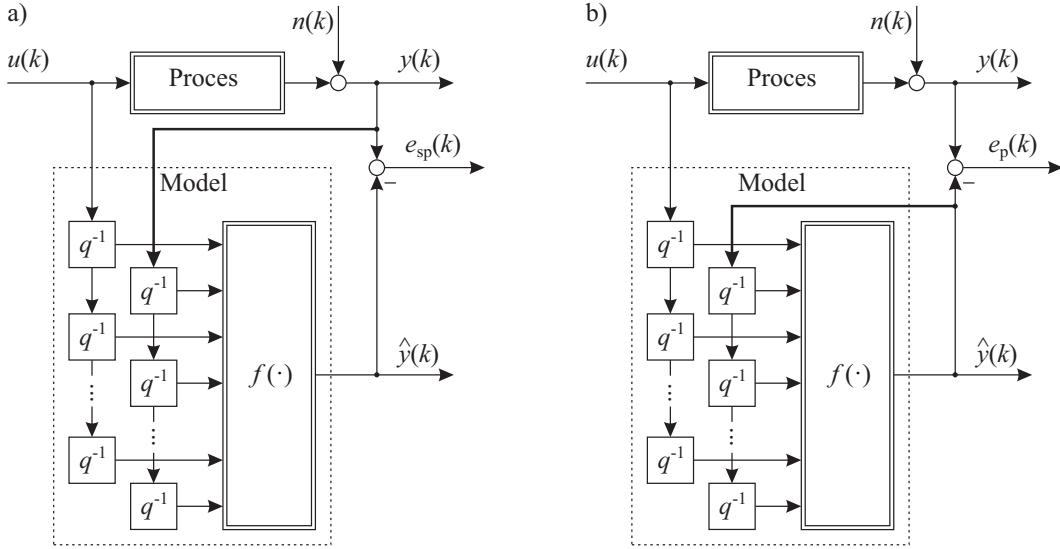
Z večkratnim gnezdenjem funkcije $f(\cdot)$, ki je samo rahlo nelinearna lahko dobimo močno nelinearno obnašanje. Pri manjših časih vzorčenja T_0 je enokoračna predikcijska površina bolj linearna. Ti odnosi so predstavljeni na sliki 15.5, kjer je enokoračna predikcijska površina v primeru Wienerjevega modela prikazana pri različnih časih vzorčenja. Razmerja so podobna tudi pri drugih sistemih in kažejo na splošne lastnosti.

15.2.2 Serijsko-paralelni in paralelni modeli

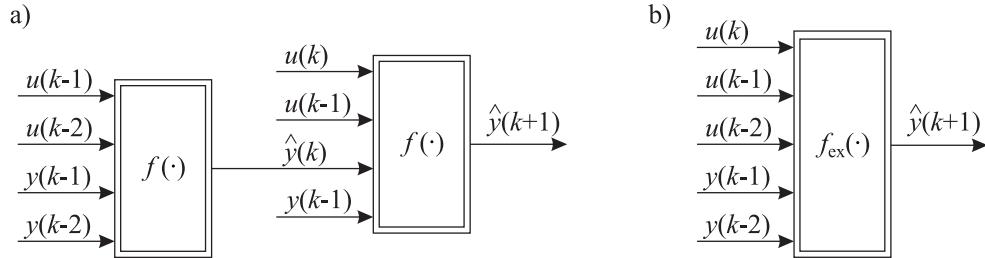
Ravno tako kot v primeru linearnih modelov, lahko nelinearne modele uporabimo v dveh različnih konfiguracijah: za *napoved ali predikcijo* in za *simulacijo*. *Predikcija* pomeni, da na osnovi *vhodov* in *izhodov* procesa s pomočjo modela napovedemo izhod procesa za en ali več korakov v prihodnosti. Pogoj v tem primeru je, da lahko izhod procesa merimo med delovanjem. V nasprotju s tem pa *simulacija* pomeni, da izhod modela napovedemo samo na osnovi *vhodov procesa*. V tem primeru rečemo, da izhod simuliramo. Govorimo o predikcijskem in simulacijskem modelu. Na sliki 15.6 je primerjava konfiguracij modelov za predikcijo (a) in simulacijo (b). V klasični literaturi o identifikaciji [31] se modeli za enokoračne napovedi imenujejo *serijsko-paralelni modeli*, konfiguracija simulacijskih modelov pa *paralelni modeli*.

Klasične aplikacije napovedovanja so v primeru vremena, napovedovanja na borzi in npr. napovedovanja porabe energije, kjer lahko trenutna stanja merimo. Tudi v primeru vodenja različnih sistemov je napovedovanje ključnega pomena za načrtovanje prediktivnih sistemov vodenja ali minimalno variančnih regulatorjev.

Simulacija je zahtevana takrat, ko izhoda med delovanjem sistema ne moremo meriti. Ta primer srečamo, ko delamo analizo delovanja brez prisotnosti realnega procesa ali pa nam model nadomesti senzor na sistemu. Ravno tako nam lahko simulacijski izhod nominalnega modela služi



Slika 15.6. a) Enokoračna napoved in serijsko-paralelni model. b) Simulacija in paralelni model.



Slika 15.7. Dvokoračna predikcija a) s kaskadno aplikacijo enokoračne napovedi in b) z razširjenim prediktorjem, ki izvede dvokoračno predikcijo v enem koraku.

za primerjavo s trenutnim izmerjenim izhodnim signalom za zaznavanje in dignostiko napak v delovanju. Iz razlike lahko s tehnikami residualov izločimo napako in jo diagnosticiramo.

Konfiguraciji, ki sta prikazani na sliki 15.6, se ne razlikujeta samo v fazi delovanja, ampak tudi v fazi učenja. Model lahko učimo tako, da minimiziramo kriterijsko funkcijo, ki je odvisna od pogreške $e(k)$. Za serijsko-paralelne modele pogrešek $e_{sp}(k)$ imenujemo *posplošeni vzhodno-izhodni pogrešek ali pogrešek enačbe*, za paralelne modele pa *izhodni pogrešek* $e_p(k)$. Tudi to je v skladu s klasično terminologijo na področju linearne identifikacije [78].

Za model drugega reda je enokoračna predikcija izračunana glede na zakasnjene izhode procesa

$$\hat{y}(k) = f(u(k-1), u(k-2), y(k-1), y(k-2)), \quad (15.13)$$

medtem ko je v primeru simulacijskega modela izračunana glede na zakasnjene izhode modela kot

$$\hat{y}(k) = f(u(k-1), u(k-2), \hat{y}(k-1), \hat{y}(k-2)). \quad (15.14)$$

Enokoračni predikcijski model je popolnoma *odprtozančen*, simulacijski pa *povratnozančen model*. Zato lahko model v enačbi 15.13 imenujemo tudi odprtozančni model, model v enačbi 15.14 pa zaprtozančni model. Primer večkoračne predikcije lahko rešimo na več načinov, slika 15.7. Običajni pristop je na sliki 15.7a, kjer je enokoračna predikcija uporabljenata zaporedno ali serijsko. Drugi pristop pa je kombinacija zapisa v enačbi 15.13 in 15.14, ker je izhod procesa

$y(k-1)$ uporabljen, vrednost trenutnega izhoda modela pa ocenjena $\hat{y}(k)$, ker $y(k)$ ni na voljo. V splošnem je za h -koračno predikcijo potrebno uporabiti izhode modela, če je $h > m$. To pomeni, da je za velike korake predikcije razlika med predikcijo in simulacijo vedno manjša. Alternativne konfiguracije za predikcijo so na sliki 15.7b, kjer napovedujemo hkrati dva koraka v prihodnosti. Ta način zahteva drugačen aproksimator $f_{\text{ex}}(\cdot)$ z dodanim vhodom $u(k)$. Povečuje se dimenzija in več različnih prediktorjev je potrebnih za oceno določenega koraka v prihodnosti. Zaradi tega je pristop b) nepraktičen.

Teža simulacijskega modela je bistveno večja od predikcijskega in ga je tudi bistveno težje dobro oceniti. To je predvsem zaradi njegove paralelne strukture, ki vnaša povratno zanko v model. Priporočimo pa naj, da model, ki ga uporabljamo v paralelni konfiguraciji ni nujno tudi naučen v tej konfiguraciji.

15.2.3 Nelinarni razredi dinamičnih vhodno-izhodnih modelov

Nelinearne modele označimo z "N" pred razredom linearnega modela, [78].

Vse nelinearne modele lahko zapišemo v obliki vhodno-izhodne relacije

$$\hat{y}(k) = f(\underline{\varphi}(k)) , \quad (15.15)$$

kjer so v regresijskem vektorju $\underline{\varphi}(k)$ zapisani zakasnjeni in trenutni vhodi, zakasnjeni izhodi modela in zakasnjeni predikcijski pogreški. Ločimo med modeli s povratno zanko in modeli brez povratne zanke.

Modeli s povratno zanko. Tako kot za linearne sisteme so tudi za nelinearne sisteme najbolj pogosti povratnozančni modeli. Regresijski vektor $\underline{\varphi}(k)$ vsebuje zakasnjlene vhode in izhode procesa ali modela, in morebiti tudi napake v predikciji. Tri najbolj pogoste strukture linearnih modelov so model ARX, model ARMAX in model OE, [78]. Analogno temu se na nelinearnem področju pojavijo modeli z naslednjimi regresorji:

$$\text{NARX : } \underline{\varphi}(k) = [u(k-1) \ \dots \ u(k-m) \ y(k-1) \ \dots \ y(k-m)]^T , \quad (15.16a)$$

$$\text{NARMAX : } \underline{\varphi}(k) = [u(k-1) \ \dots \ u(k-m) \ y(k-1) \ \dots \ y(k-m) \\ e(k-1) \ \dots \ e(k-m)]^T , \quad (15.16b)$$

$$\text{NOE : } \underline{\varphi}(k) = [u(k-1) \ \dots \ u(k-m) \ \hat{y}(k-1) \ \dots \ \hat{y}(k-m)]^T . \quad (15.16c)$$

Lahko vidimo, da model NARX učimo v serijsko-paralelni konfiguraciji (slika. 15.6 a) in model NOE v paralelni konfiguraciji. Model NARMAX zahteva oboje, izhode procesa $y(k-i)$ in izhode modela $\hat{y}(k-i)$, ki so zajeti v napakah predikcije $e(k-i)$. Ugotovimo lahko, da eksplicitno modeliranje šuma pri modelu NARMAX zahteva dodatne vhode za aproksimator $f(\cdot)$. Ker imamo v primeru nelinearnih sistemov inherentno problem visoke dimenzionalnosti, se bolj kompleksne strukture modelov kot so model NBJ ali model NARARX, pravzaprav zelo redko uporabljajo. Zaradi enostavnosti sta najbolj uporabljeni model NARX ali model NOE.

Ena od pomembnih izbir pri modelih s povratno zanko je izbira reda modela m . Red ima odločilni vpliv na opisne zmožnosti modela. Pri tem si lahko pomagamo z metodo Lipschitzevih koeficientov, metodo PCA ali pa smo prepuščeni izbiri s poskušanjem. Največkrat s poskušamo pomagati z uporabo apriornega znanja. To je seveda zelo dolgotrajen postopek posebno takrat, ko ima model zakasnitve na vhodnem in izhodnem signalu in po možnosti tudi mrtvi čas. V nadaljevanju si bomo ogledali, kako si pri izbiri reda pomagamo z uporabo zgoraj omenjenih dveh metod, [41].

Tabela 15.1. Problemi, ki se pojavijo pri modelu NARX in modelu NOE, ko je aproksimator $f(\cdot)$ linearno ali nelinearno parametriran.

	NARX	NOE
Linearno parameteriran model $f(\cdot)$	Linearna optimizacija	Nelinearna optimizacija
Nelinearno parameteriran model $f(\cdot)$	Nelinearna optimizacija	Nelinearna optimizacija

Tabela 15.2. Učenje modela NARX in modela NOE glede na namen uporabe

	NARX (serijsko-paralelni model)	NOE (paralelni model)
Model za enokoračno predikcijo	×	neobčutljive
Model za simulacijo	kot aproksimacija	×

Naslednji pomanjkljivosti metod s povratno zanko sta, da v splošnem stabilnosti ne moremo dokazati in da moramo stacionarno stanje izračunati iterativno z reševanjem nelinearne enačbe:

$$y_0 = f\left(\underbrace{u_0, \dots, u_0}_{m-krat}, \underbrace{y_0, \dots, y_0}_{m-krat}\right), \quad (15.17)$$

kjer je (u_0, y_0) delovna točka.

Na drugi strani pa so modeli s povratno zanko zelo kompaktni in dobro opišejo proces. Regresijski vektor $\varphi(k)$ vsebuje majhno število regresorjev kar omogoča nizko kompleksnost pri aproksimaciji $f(\cdot)$. To je seveda v primeru nelinearnih sistemov še bolj pomembno kot v primeru linearnih.

Poglejmo si prednosti in slabosti modela NARX. Prva slabost je nerealistično obravnavanje šuma, ki vodi do pristranskosti v ocenah parametrov, do velike občutljivosti na čas vzorčenja (prevelika hitrost vzorčenja poslabša kvaliteto modela) in prevelikega poudarka pri ocenjevanju visokofrekvenčnih lastnosti modela. Na drugi strani omogoča struktura NARX linearno optimiranje parametrov, ker je enačba napake linearna v parametrih. To seveda velja samo, če je posledična stran linearno parametrirana (LLM, polinomi). Za nelinearno parametrirane aproksimatore je potrebno uporabiti (MLP) nelinearno optimizacijo. Tudi v teh primerih je učenje modela NARX enostavnejše, ker ni potrebno upoštevati povratne zanke pri izračunu odvodov. Lastnosti so zbrane v tabeli 15.1.

Tabela 15.2 ilustrira uporabo modela NARX in modela NOE glede na namen uporabe modela. Če želimo model uporabiti za enokoračno predikcijo, potem je najprimernejši model NARX, ki minimizira napako enokoračne predikcije. Če model uporabljamemo za simulacijo, potem je situacija bolj nejasna. Na eni strani je model NOE tisti, ki optimira simulacijsko napako, na drugi strani pa je enostavnejše naučiti model NARX, še posebno, če je $f(\cdot)$ linearno parametrirana; glej tabelo 15.1. Zaradi tega lahko uporabimo obe metodi. Model NOE bolje zasleduje cilje modeliranja, NARX pa je aproksimator, ki optimira enokoračno predikcijo in je enostavnejši. Najpogosteje učimo model NARX najprej kot inicializacijo, potem pa optimiramo model NOE.

Učenje modela NOE je neodvisno od izbranega aproksimatorja in vedno zahteva nelinearno optimizacijo z izračunom odvodov. Nelinearna optimizacija je potrebna zaradi povratnozorančne strukture. V nasprotju z modelom NARX se lahko pri modelu NOE pojavi problem *akumulacije pogreška* kar lahko vodi do slabe natančnosti in celo v nestabilnost. To se zgodi, ko se majhni pogreški v predikciji zbirajo in naraščajo glede na povratno zanko (v modelu se povratno vračajo

napovedani signali, ki imajo svoje pogreške). Pri modelu NOE lahko zmanjšamo ta efekt med učenjem modela. Pri modelu NARX pa se ta efekt pojavi, ko ga uporabimo v simulaciji.

Akumulacija pogreška je resen problem pri modelu NARX, saj lahko na eni strani z učenjem zmanjšujemo pogrešek enokoračne predikcije, istočasno pa narašča pogrešek v primeru simulacijske uporabe modela. Lahko se celo zgodi, da enokoračna predikcija izkazuje dobre lastnosti, model pa postane lokalno nestabilen.

Modeli brez povratne zanke. Če ni nobene povratne zanke, potem regresorski vektor $\underline{\varphi}(k)$ vsebuje samo zakasnjene ali filtrirane vhodne signale. Število zahtevanih regresorjev je v primeru modelov brez povratne zanke veliko večje kot obratno, kar je v primeru nelinearnosti in problema dimenzij lahko kritično. Zato je smiselno uporabljeni le aproksimatorje, ki lahko dobro delajo z velikimi dimenzijami.

Nelinearni model končnega impulznega odziva. Modeli s povratno zanko imajo veliko slabosti, zaradi tega se poskuša določene probleme reševati z uporabo nelinearnih modelov končnega impulznega odziva (NFIR), ker ne vnašajo nobene povratne zanke. Regresijski vektor za NFIR model je sestavljen samo iz zakasnjenih vhodov procesa:

$$\text{NFIR} : \underline{\varphi}(k) = [u(k-1) \ u(k-2) \ \cdots \ u(k-m)]^T. \quad (15.18)$$

Cena odprtozančne strukture modela je veliko število zakasnitev m , ki so potrebne, da opisemo dinamiko procesa. Teoretično bi morali imeti $m \rightarrow \infty$. V praksi pa moramo izbrati število zakasnitev glede na čas izravnave procesa. Tipično v območju časa izravnave T_{95} od $10T_0$ do $50T_0$, kjer je T_0 čas vzorčenja. Ker model nima povratne zanke, je njegova stabilnost zagotovljena. Stacionarno stanje enostavno izračunamo na ta način, da v enačbi 15.18 postavimo $u(k-i) = u_0$. Čeprav so določene lastnosti tega modela zanimive, je uporab malo, ker zahteva previsok red m . To pa močno poveča njegovo kompleksnost pri kateremkoli aproksimatorju.

15.2.4 Omejitve nelinearnih vhodno-izhodnih modelov

Vhodno-izhodni modeli lahko opišejo zelo velik razred različnih sistemov [75], vendar niso tako splošni kot nelinearni modeli stanj, kjer lahko nelinearnosti realiziramo na stanjih. Omejeni so na nelinearnosti, ki so invertibilne. Ne moremo jih uporabiti v primerih, kjer nastopata histereza in zračnost (*ang. backlash*), ker so v teh primerih odločilnega pomena tudi nemerljiva stanja.

Zadnjo omejitev lahko prikažemo na primeru, prikazanem na sliki 15.8. Procesa Wienerjevega tipa z neinvertibilno nelinearnostjo ne moremo modelirati z vhodno-izhodnim modelom, ker stanje $x(k)$ ni merljivo.

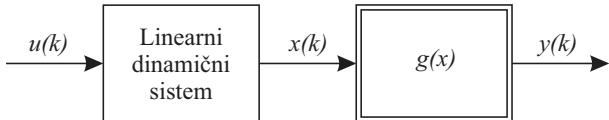
Predpostavimo dinamični sistem na sliki 15.8, kjer je Wienerjev proces definiran z enačbo

$$x(k) = b_1 u(k-1) - a_1 x(k-1), \quad y(k) = g(x(k)). \quad (15.19)$$

Izhod Wienerjevega procesa pa zapišemo kot

$$\begin{aligned} y(k) &= g(b_1 u(k-1) - a_1 x(k-1)) \\ &= g(b_1 u(k-1) - a_1 g^{-1}(y(k-1))), \end{aligned} \quad (15.20)$$

kjer je $g^{-1}(\cdot)$ inverz funkcije $g(\cdot)$. Če $g^{-1}(\cdot)$ obstaja, potem lahko enačbo 15.20 zapišemo kot nelinearni vhodno-izhodni model prvega reda glede na enačbo 15.2. Drugače ga ne moremo formulirati kot vhodno-izhodni model. Lahko ga seveda aproksimiramo, vendar ne s poljubno natančnostjo. Zahteva invertibilnosti funkcije $g(\cdot)$ je precej stroga zahteva, ki omeji uporabo tega tipa modela na monotone nelinearnosti $g(\cdot)$.



Slika 15.8. Wienerjevega sistema ne moremo modelirati z vhodno-izhodnim modelom, če $g(\cdot)$ ni invertibilna.

15.3 Vzbujanje sistemov

Med najpomembnejše korake pri identifikaciji sistemov sodi načrtovanje ustreznih meritev. Posebno pomembno je vzbujanje sistema z ustreznimi signali vzujanja. Ta korak je še toliko bolj pomemben v primeru nelinearnih sistemov, ker so nelinearni sistemi bistveno bolj kompleksni in morajo zato meritve vsebovati bistveno več informacij. Pri tem je seveda zelo pomembno apriorno znanje na osnovi katerega lahko primerno načrtamo meritve.

Ne glede na arhitekturo ali strukturo modela je gornja meja natančnosti modela odvisna od kvalitete podatkov, ki jih uporabimo za identifikacijo. V primeru linearnih sistemov je pogost vzbujalni signal psevdonakljčni binarni signal (PRBS), katerega parametre določimo glede na pričakovano dinamiko procesa oziroma njegove frekvenčne lastnosti. Pri nelinearnih sistemih pa je poleg ustreznega frekvenčnega vzbujanja potrebno izbrati tudi ustrezeno amplitudno vzbujanje, da dobimo informacijo po celotnem delovnem področju sistema. To pomeni, da vsak nelinearni sistem zahteva svoje načrtovanje meritev in je to bisteno bolj zahtevno kot v primeru linearnih sistemov.

V nadaljevanju si bomo ogledali nekaj splošnih stvari v zvezi z vzbujanjem sistemov, ki veljajo ne glede na arhitekturo ali strukturo modela in jih je vedno potrebno upoštevati.

Najprej si poglejmo, zakaj je PRBS neprimeren vzbujalni signal v primeru nelinearne dinamike sistema. Predpostavimo nelinearni Hammersteinov model z vhodom na intervalu med $[-4, 4]$ in časovno konstanto 16 s , ki ga zapišemo z diferenčno enačbo

$$y(k) = 0.06 \arctan(u(k-1)) + 0.94 y(k-1), \quad (15.21)$$

kjer je čas vzorčenja $T_0 = 1\text{ s}$. Slika 15.9a kaže PRBS v časovnem prostoru in rezultirajoči vhodni prostor aproksimatorja, ki ga določata $u(k-1)$ in $y(k-1)$ in njuna porazdelitev. Ta razdelitev je dobra za določitev ravnine, kar bi bilo primerno, če bi obravnavali linearni sistem. V linearjem primeru bi identificirali parametra b_1 in a_1 linearnega modela

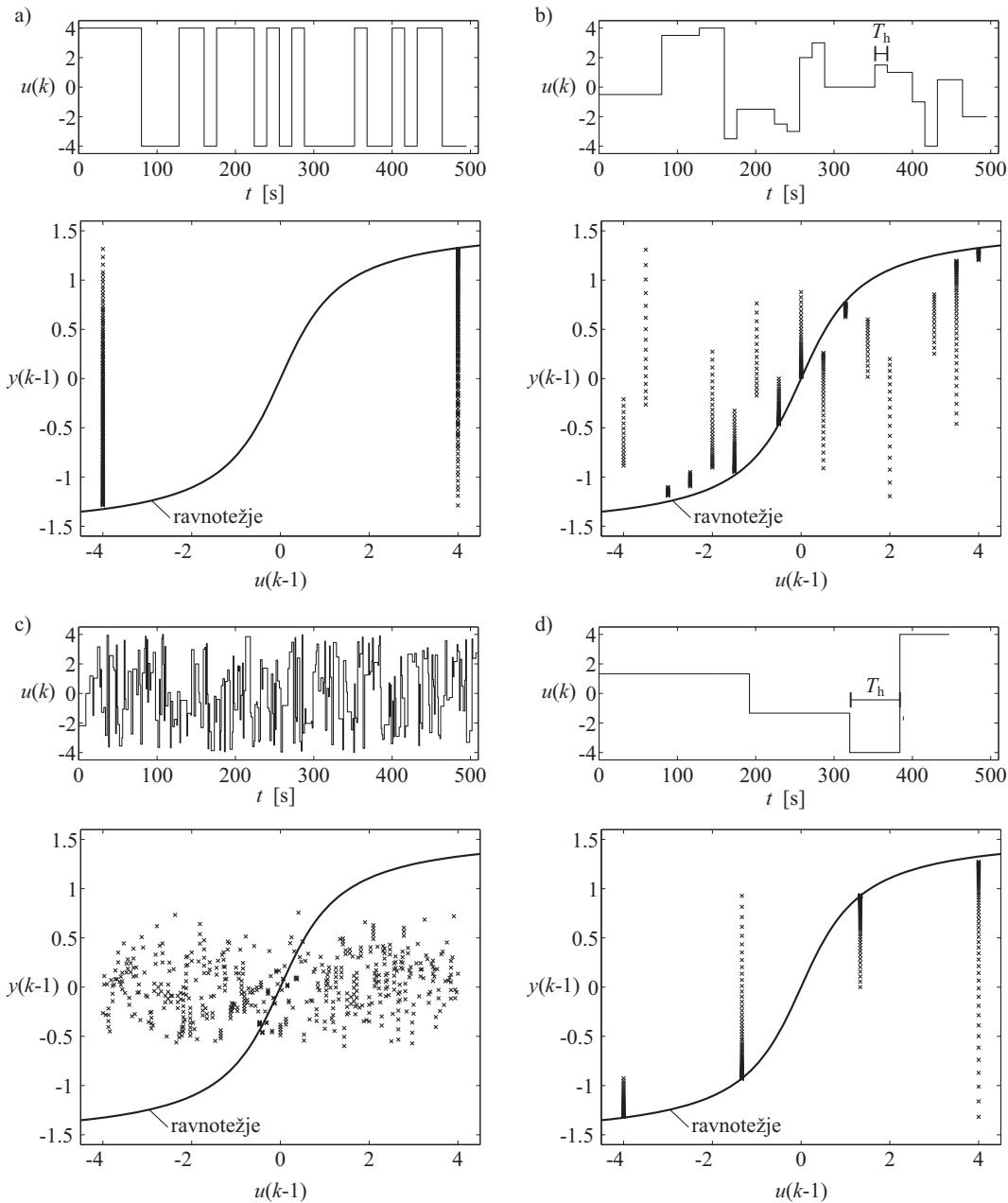
$$y(k) = b_1 u(k-1) - a_1 y(k-1)$$

. Zaradi tega morajo imeti podatki čim večjo variabilnost v smeri $u(k-1)$ in $y(k-1)$. Taka distribucija vodi k najmanjši možni varianci napake pri oceni parametrov b_1 in a_1 . To seveda dosežemo s PRBS, ki preklaplja med minimalno in maksimalno vrednostjo -4 in 4 po spremenljivki $u(k-1)$. S tem zavzame tudi celoten prostor spremenljivke $y(k-1)$ med -1.4 in 1.4 . Podatki so primerni za identifikacijo ravnine, niso pa primerni za identifikacijo nelinearne površine, ker v vmesnih točkah intervala, med -4 in 4 , nimamo nobene informacije.

Enostavna rešitev tega problema je generiranje amplitudno moduliranega PRBS (APRBS). Signal je predstavljen na sliki 15.9b.

Poleg minimalne in maksimalne amplitude in dolžine trajanja signala je potrebno v primeru vzbujanja nelinearnih sistemov določiti še čas zadržanja, to je minimalni čas, v katerem signal zadrži enako amplitudo. Primeri so na sliki 15.9b, c in d.

Najbolj primerno je, da je minimalni čas zadržanja APRBS približno enak dominantni (največji) časovni konstanti procesa:



Slika 15.9. Vzbujanje sistemov z nelinearno dinamiko: a) binarni PRBS, b) APRBS z ustreznim časom zadržanja, c) APRBS s kratkim časom zadržanja, d) APRBS s predolgom časom zadržanja.

$$T_h \approx T_{\max}. \quad (15.22)$$

Poleg bolj splošnih navodil, ki so bila dana zgoraj, dodajmo še nakaj navodil za načrtovanje signalov za vzbujanje:

1. *Namen modela.* Najprej je potrebno ugotoviti za kakšen namen gradimo model. Ali ga bomo uporabili za vodenje, diagnosticiranje napak, za napovedovanje ali za optimizacijo. Iz tega lahko ugotovimo potrebno natančnost v različnih delovnih režimih in zahtevana frekvenčna območja. Na primer, za namene vodenja je najbolj pomembna natančnost okoli lomne frekvence, napake pri nizkih frekvencah pa enostavno odpravimo z integrirno akcijo regulatorja. Napake nad lomno frekvenco pa so izven območja izvršnih členov in zaprtozančnega frekvenčnega območja.
2. *Maksimalna dolžina učne množice.* Z večjo učno množico bo, pri ustrezni porazdelitvi podatkov, model bolj natančen. V večini realni primerov pa je čas meritev omejen. Lahko pa je velikost množice odvisna tudi od omejitev strojne opreme.
3. *Lastnosti različnih vhodnih signalov.* Za vsak vhodni signal sistema je potrebno ugotoviti ali je potrebno izvesti statične ali dinamične meritve. Z drugimi besedami, ugotoviti moramo funkcionalnost vhodne spremenljivke. Če je vhodna spremenljivka regulirni signal, bodo zanesljivo potrebne dinamične meritve. V drugih primerih pa so mogoče zadosti že statične, kot je to v primeru počasi spreminjačih se merjenih motenj v regulacijskih sistemih.
4. *Območje vhodnih signalov.* Proces moramo vzbujati po celotnem delovnem področju, ki lahko nastopi med obratovanjem sistema. Učni podatki morajo vsebovati podatke med limitnimi vrednostmi vhodnih signalov, ker je interpolacijsko delovanje bistveno bolj pomembno kot ekstrapolacijsko.
5. *Enakomerna porazdelitev podatkov.* Še posebno pri modeliranju za vodenje je potrebno zagotoviti, da so izhodni podatki enakomerno porazdeljeni. Na ta način vsebujejo enako količino informacije o vseh delovnih točkah.
6. *Dinamične lastnosti.* Dinamika signalov mora biti primerno izbrana, da lahko vzbujamo različne dinamike v različnih delovnih točkah.

Zelo veliko predhodnega znaja je potrebno za dobro načrtovanje identifikacijskih signalov. V resnici so določena osnovna znanja o sistemih vedno na voljo. Na primer, informacija o približni statistični karakteristiki in informacija o dominantni časovni konstanti. Če je obnašanje sistema popolnoma neznano, lahko za začetek napravimo nekaj odzivov na stopnico v različnih delovnih točkah.

V praksi je operater procesa tisti, ki omeji čas in tip meritev, ki jih bomo izvedli. Običajno pa lahko upazujemo delovanje procesa samo v normalnem načinu delovanja, brez aktivnega načina pridobivanja informacije (izbranih vzbujalnih signalov). V takih primerih je potrebno biti izjemno previden, saj je težko dobiti dober model za ekstrapolacijo. Pomagamo si z vnašanjem apriornega znanja v model (fizikalne enačbe), detekcijo ekstrapolacije in ustrezno ravnanje v takem primeru (robustni paralelni nadomestni model). Dober napotek je tudi ta, da v bolj pogostih delovnih točkah napravimo več meritev in bolj kompleksen model, ki nam bo dal bolj natančen model. Pomembno je razumeti, da velika gostota podatkov v enem področju sili model, da "porabi" velik del prostostnih stopenj svoje kompleksnosti za opis tega področja. To je lahko tudi nezaželeno, če gostota podatkov ni v pravem področju. To se lahko zgodi, če so podatki zajeti med normalnim delovanjem in ne z aktivnim vzbujanjem. Take podatke lahko v kriterijski funkciji utežimo in jim damo manjši pomen.

15.4 Ocenjevanje reda sistema

V primeru nelinearnih modelov z zunanjim dinamikom je določevanje reda sistema enakovredno določevanju ustreznih vhodov za funkcijo $f(\cdot)$ v enačbi 15.15. Določevanje reda je na ta način določevanje vhodov in je zato procedura, ki je razložena v nadaljevanju, uporabna tako v primeru dinamičnih kot tudi v primeru statičnih sistemov. Pomembno je razumeti, da sta zakasnjeni vhod $u(k-i)$ in zakasnjeni izhod $y(k-i)$ lahko dva vhoda v funkcijo $f(\cdot)$ in da vsebujejo lastnosti, ki povzročijo, da je določevanje reda bistveno bolj zahtevno. Na primer, $y(k-1)$ in $y(k-2)$ sta zelo korelirana (kar prinaša redundanco v informaciji), ampak sta oba relativantna.

Problem določevanja reda na osnovi avtomatske detekcije še ni ustrezno rešen. Največkrat se uporablja metoda na osnovi poizkušanja in vnosa apriori znanj. Z enostavnimi odzivi na stopnico v različnih delovnih točkah lahko aproksimativno določimo red sistema (oscilacije določajo vsaj drugi red, itd.) He in Asada [41] sta predlagala metodo Lipschitzevih koeficientov, ki bo predstavljena v nadaljevanju.

15.4.1 Struktturna identifikacija na osnovi metode Lipschitzevih koeficientov

Metoda temelji na podatkih in ne zahteva nobene predpostavke o arhitekturi ali strukturi modela. Edina omejitev metode je gladko obnašanje izhoda procesa, ki je skupna tudi modelom z zunanjim dinamikom in splošnim identifikacijskim modelom. Osnovna ideja metode je predstavljena na sliki 15.10a. Najprej je potrebno je definirati ustrezone vhode funkcije

$$y = f(\varphi_1, \varphi_2, \dots, \varphi_n), \quad (15.23)$$

iz množice potencialnih vhodov $\varphi_1, \varphi_2, \dots, \varphi_o$ ($o > n$), ki je na voljo. Če so φ_i določeni fizikalni vhodni signali, potem enačba 15.23, opisuje problem statične aproksimacije, če pa so zakasnjeni vhodni in izhodni signali, potem je to model z zunaj dinamiko.

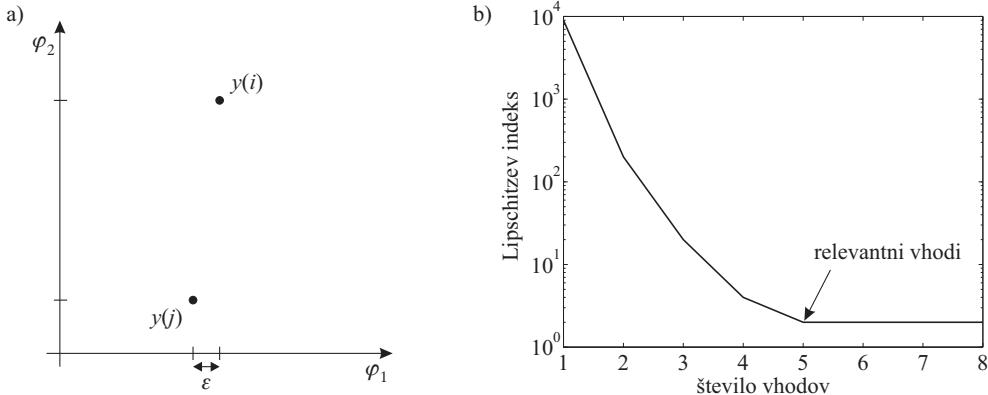
Ideja je naslednja. Če je funkcija v enačbi 15.23 odvisna samo od $n-1$ vhodov, v resnici pa je funkcija n vhodov, potem množica učnih podatkov vsebuje vsaj dve točki, ki sta si zelo blizu (lahko sta celo identični) v prostoru dimenzije $n-1$, se pa zelo razlikujeta glede na vhod n . Ta primer je na sliki 15.10a pri $n=1$. Dve točki i in j sta si blizu v vhodnem prostoru, ki ga določa spremenljivka φ_1 , sta pa daleč narazen, če točki pogledamo v vhodnem prostoru, ki ga določata spremenljivki φ_1 in φ_2 . Ker sta ti dve točki blizu v prostoru, ki ga definira $n-1$ vhodov (v našem primeru φ_1), bi bilo pričakovati, da sta ob predpostavki gladke funkcije pripadajoča $y(i)$ in $y(j)$ tudi blizu skupaj. Če en ali več vhodov manjka, potem je naravno, da imata $y(i)$ in $y(j)$ popolnoma drugačne vrednosti. V takem primeru lahko sklepamo, da $n-1$ vhodov ni zadosti in moramo vključiti še n -ti vhod v obravnavo.

V [41] je definiran Lipschitzev kvocient kot indeks, ki je velik, če manjka en ali več vhodov (večji, če manjka več vhodov) in majhen v obratnem primeru. Z uporabo Lipschitzevega indeksa na sliki 15.10b smo dobili rezultat $n=5$ in $o=8$. Na ta način je pravo število ($n=5$) določeno tam, kjer Lipschitzev indeks neha padati.

Lipschitzev kvocient v enodimensionalnem primeru (vhod φ) definiramo kot

$$l_{ij} = \frac{|y(i) - y(j)|}{|\varphi(i) - \varphi(j)|} \quad \text{za } i = 1, \dots, N, j = 1, \dots, N \text{ in } i \neq j, \quad (15.24)$$

kjer je N število meritev v učni množici. Pripomnimo naj, da je na ta način definiranih $N(N-1)$ Lipschitzevih kvocientov, izračunati pa jih moramo $N(N-1)/2$, ker velja $l_{ij} = l_{ji}$. Ker v enačbi 15.24 končna diferenca predstavlja aproksimacijo odvoda $df(\varphi)/d\varphi$, jo moramo omejiti z



Slika 15.10. a) Dve točki sta si blizu v prostoru spremenljivke φ_1 , če dodamo še spremenljivko φ_2 pa imata lahko zelo različni izhodni vrednosti $y(i)$ in $y(j)$, če je funkcija odvisna od obeh spremenljivk, če pa je $y(i) \approx y(j)$, potem je funkcija odvisna samo od φ_1 . b) Lipschitzev indeks detektira primer, ko so vsi relevantni vhodi vključeni.

maksimalnim naklonom funkcije $f(\cdot)$, če je $f(\cdot)$ gladka. Za večdimenzionalni primer je potrebno enačbo 15.24 razširiti na

$$l_{ij}^{(n)} = \frac{|y(i) - y(j)|}{\sqrt{(\varphi_1(i) - \varphi_1(j))^2 + \dots + (\varphi_n(i) - \varphi_n(j))^2}}, \quad (15.25)$$

za $i = 1, \dots, N$, $j = 1, \dots, N$ in $i \neq j$. Zgornji indeks (n) v $l_{ij}^{(n)}$ definira število vhodov.

Lipschitzev indeks je definiran kot maksimalni Lipschitzev kvocient

$$l^{(n)} = \max_{i,j,(i \neq j)} (l_{ij}^{(n)}) , \quad (15.26)$$

ali, kot je predlagano v [41], kot geometrijsko povprečje c (kjer je c načrtovalski parameter) največjih Lipschitzevih kvocientov, da zagotovimo manjšo odvisnost indeksa od šuma. Če je n pramajhen in vsi relevantni vhodi niso vključeni, potem je Lipschitzev indeks visok zaradi situacije na sliki 15.10a. Če vključimo vse relevantne vhode, potem tudi maksimalni Lipschitzev koeficient v enačbi 15.26 ostane konstanten.

Ta strategija zahteva ureditev vhodov po vrsti. Na primer, za nelinearni dinamični sistem lahko izberemo vhode $\varphi_1 = u(k-1)$, $\varphi_2 = y(k-1)$, $\varphi_3 = u(k-2)$, $\varphi_4 = y(k-2)$ in tako dalje. To lahko vodi do tega, da najdemo red sistema, ni pa možno ugotoviti mrtvih časov.

Pri tej metodi lahko ugotovimo dve pomembni slabnosti. Prva je občutljivost na šum, druga pa porazdelitev podatkov. Vpliv šuma lahko zmanjšamo z ustreznim $c > 1$, vendar to zmanjša občutljivost detekcije reda, ker povpreči med različnimi podatki. Ne glede na vse to, je metoda zelo uporabna, ker ni zelo računsko zahtevna, sploh v primeru manjših kompleksnosti.

15.4.2 Struktturna identifikacija z uporabo metode PCA

PCA je v osnovi statična metoda in jo je treba za analizo dinamičnih sistemov razširiti. Za opis dinamike sistema je potrebno vhodni matriki podatkov dodati še ustrezno število časovno zamaknjjenih vhodnih podatkov

$$\underline{X}_l(k) = [\underline{X}(k), \underline{X}(k-1), \dots, \underline{X}(k-l)] . \quad (15.27)$$

Poiskati moramo tisto število časovno zamaknjjenih vhodnih podatkov l , ki v kovariančni matriki ne dodaja nobenih novih relacij med podatki. Kar pomeni, da je optimalno število s katerim

lahko opišemo dinamiko izmerjenih podatkov za eno manjše in enako $l - 1$. Na ta način je matrika podatkov sestavljena iz zamaknjenih vhodnih matrik $\underline{X}_i(k)$, $i = 1, \dots, l$. S primernim številom l (red sistema) zagotovimo, da se tako statične kot tudi dinamične komponente pojavijo v prostoru podatkov. Število l določamo iterativno. Najprej obravnavamo statičen primer ($l = 0$) in določimo število statičnih linearnih relacij, ki je enako razlik med številom vseh spremenljivk (stolpci $\underline{X}(l)$) in številom določenih glavnih komponent. V naslednjem koraku vzamemo $l = 1$ ter določimo število na novo odkritih relacij. To dobimo tako, da od vseh spremenljivk odštejemo število glavnih komponent in število v predhodnih korakih odkritih kolinearnih relacij r_l , kjer je z indeksom l definirano število časovnih zakasnitev. Predpostavimo, da imamo primer podatkovne matrike $\underline{X}(k)$ (statični podatki) v kateri imamo r_0 kolinearnih relacij. Če dodamo tej matriki matriko zakasnjenih podatkov dobimo novo podatkovno matriko, ki je definirana kot

$$\underline{X}_1(k) = [\underline{X}(k), \underline{X}(k-1)] .$$

Število kolinearnih stolpcev v tej matriki je enako $r_1 = m - s - 2r_0$, kjer je m število stolpcev v matriki $\underline{X}_1(k)$, s je število glavnih komponent matrike $\underline{X}_1(k)$, kjer upoštevamo vse komponente, ki imajo od nič različne lastne vrednosti in r_0 število kolinearnih stolpcev v matriki $\underline{X}(k)$. Ker so statistične lastnosti matrike $\underline{X}(k)$ enake lastnostim premaknjene matrike $\underline{X}(k-1)$, je v sestavljeni matriki $\underline{X}_1(k)$ število kolinearnih stolpcev r_0 dvojno.

Z dodajanjem časovno zamaknjenih podatkov dobimo matriko

$$\underline{X}_2(k) = [\underline{X}(k), \underline{X}(k-1), \underline{X}(k-2)] .$$

Število kolinearnih stolpcev v tej matriki je enako $r_2 = m - s - 2r_1 - 3r_0$, kjer je m število stolpcev v matriki $\underline{X}_2(k)$, s je število glavnih komponent matrike $\underline{X}_2(k)$ in r_1 število kolinearnih stolpcev v matriki $\underline{X}_1(k)$. Ker so statistične lastnosti matrike $\underline{X}_1(k)$ enake lastnostim premaknjene matrike $[\underline{X}(k-1), \underline{X}(k-2)]$, je v sestavljeni matriki $\underline{X}_2(k)$ to število kolinearnih stolpcev r_1 dvojno. Število kolinearnih stolpcev r_0 pa se potroji, ker imamo v matriki $\underline{X}_2(k)$ tri podmnožice stolpcev z enakimi statističnimi lastnostmi, ki so v matrikah $\underline{X}(k)$, $\underline{X}(k-1)$ in $\underline{X}(k-2)$.

Splošno zapišemo število novih kolinearnih stolpcev z naslednjo formulo, ki velja za $l \geq 1$

$$r_l = m - s - \sum_{i=1}^l (i+1) r_{l-i} . \quad (15.28)$$

V primeru $l = 0$ je število kolinearnih relacij enako $r_0 = m - s$. Iterativni postopek izračunavanja kolinearnih stolpcev končamo, ko je število kolinearnih relacij r_l enako nič. Število zakasnitev, ki predstavlja optimalno strukturo zapisa dinamičnega sistema je v tem primeru enako $l - 1$.

Estimacija parametrov z uporabo metode PCA. V primeru, ko je število glavnih komponent z od nič različnimi lastnimi vrednostmi enako $s = m - 1$, takrat lastni vektor, ki ima lastno vrednost ($\lambda_{sm} = 0$) enako nič, definira linearno relacijo med podatki

$$\underline{X}_l^T \underline{X}_l - \left(\sum_{i=1}^s \lambda_{li} \underline{p}_{li} \underline{p}_{li}^T + \lambda_{lm} \underline{p}_{lm} \underline{p}_{lm}^T \right) = 0 . \quad (15.29)$$

To pomeni, da se podatki ne širijo v smeri lastnega vektorja \underline{p}_{lm} , ki ima lastno vrednost enako nič. Vektor \underline{p}_{lm} je normalni vektor na hiperravnino podatkov, ki jo lahko zapišemo z naslednjo enačbo

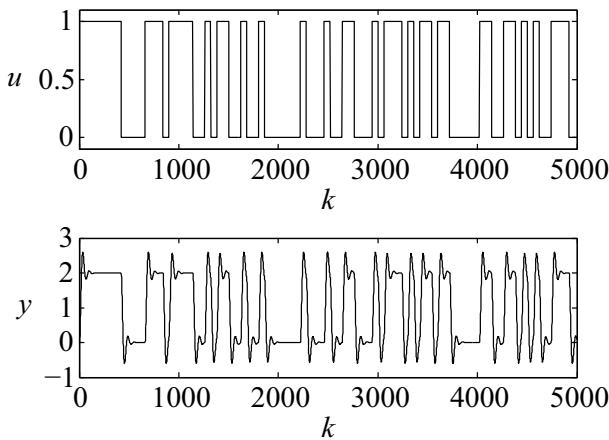
$$\underline{p}_{lm}^T \underline{x} = 0 . \quad (15.30)$$

Enačba 15.30 je implicitni linearni model, ki povezuje podatke v matriki \underline{X}_l . Podatki, ki jih obravnavamo so enakovredni, med njimi ne ločimo vhodnih in izhodnih spremenljivk. Če poznamo vhode in izhode, lahko iz enačbe 15.30 eksplisitno izrazimo izhodno spremenljivko in dobimo linearno enačbo, ki definira izhodno spremenljivko kot linearno kombinacijo vhodnih spremenljivk.

Primer strukturne identifikacije sistema drugega reda z metodo PCA. Primer strukturne identifikacije (določevanje reda sistema) iz izmerjenih podatkov bomo prikazali na primeru sistema drugega reda, ki je dan z diferenčno enačbo

$$y(k+2) = 1.92y(k+1) - 0.93y(k) + 0.02u(k).$$

Sistem vzbujamo s psevdonaključnim binarnim signalom $u(k)$. Oba signala sta prikazana na sliki 15.11. Pri analizi podatkov bomo rezultate zapisali v tabelo 15.3. Lastni vredno-



Slika 15.11. Vhodni in izhodni signal v primeru strukturne identifikacije z metodo PCA.

sti v primeru matrike podatkov brez premikov $\underline{X}_0(k)$ sta različni od nič, zato v matriki ni kolinearnih stolpcev. Ravno tako dobimo v primeru matrike $\underline{X}_1(k)$, ki ima štiri stolpce ($m = 4$), štiri od nič različne lastne vrednosti. Pri $l = 2$ dobimo lastne vrednosti enake $A_2 = [3.8941, 0.1776, 0.0075, 0.0027, 0.0024, 0.0000]$, kar pomeni, da je $s = 5$. Za premik $l = 3$ je število od nič različnih lastnih vrednosti enako $s = 6$ in v primeru $l = 4$ je enako $s = 7$. Pri tej vrednosti premika tudi zaključimo z dodajanjem zakasnjenih podatkov. Iz tabele 15.3

m	l	s	r_l
2	0	2	$r_0 = m - s = 0$
4	1	4	$r_1 = m - s - 2r_0 = 0$
6	2	5	$r_2 = m - s - 2r_1 - 3r_0 = 1$
8	3	6	$r_3 = m - s - 2r_2 - 3r_1 - 4r_0 = 0$
10	4	7	$r_4 = m - s - 2r_3 - 3r_2 - 4r_1 - 5r_0 = 0$

Tabela 15.3. Postopek strukturne identifikacije z uporabo metode PCA.

ugotovimo, da z dodajanjem matrike, ki je zakasnjena za tri vzorce ($l = 3$) ne odkrijemo nobenih novih relacij v podatkih in zato lahko rečemo, da je dinamični sistem, ki ga obravnavamo, drugega reda.

V primeru dveh zakasnitev ($l = 2$) imamo eno kolinearno relacijo med stolpcji ($r_2 = 1$) v matriki $\underline{X}_2(k)$. Residuum $\underline{E}_r^T \underline{E}_r^T$ lahko zapišemo v obliki razcepa s singularnimi vrednostmi

$$\underline{E}_r^T \underline{E}_r = \underline{\lambda}_{26} \underline{p}_{26} \underline{p}_{26}^T ,$$

potem lastni vektor, ki pripada lastni vrednosti z vrednostjo 0, opisuje model procesa. To lahko zapišemo z naslednjo enačbo

$$\underline{p}_{26}^T \underline{x} = 0 . \quad (15.31)$$

V našem primeru dobimo:

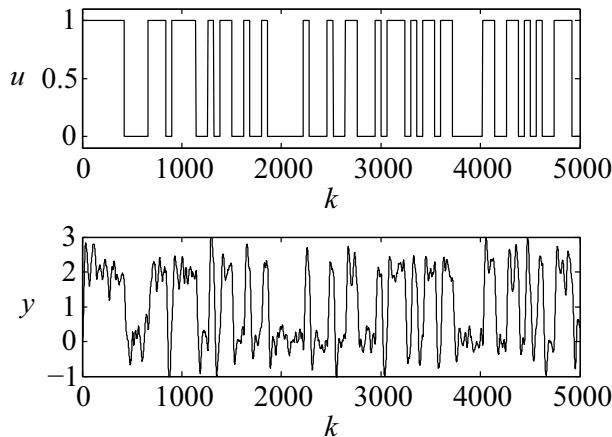
$$-0.4244y(k) + 0.8149y(k-1) + 0.0085u(k-2) - 0.3947y(k-2) = 0 .$$

Če v gornji enačbi eksplizitno izrazimo spremenljivko $y(k)$, dobimo zapis modela podatkov, ki je v obravnavanem (idealnem) primeru naslednji

$$y(k+2) = 1.9201y(k+1) - 0.9300y(k) + 0.0200u(k) .$$

Vidimo, da smo dobili model procesa, ki ustreza diferenčni enačbi s pomočjo katere smo generirali podatke.

Poglejmo si sedaj isti primer le, da bomo predpostavili šum meritve izhodnih podatkov $\mathcal{N}(0, 0.01)$. Meritve so v tem primeru prikazane na sliki 15.12. S postopkom opazovanja ko-



Slika 15.12. Vhodni in izhodni signal z dodanim šumom v primeru strukturne identifikacije z metodo PCA.

linearnih relacij v zakasnjениh podatkovnih matrikah pridemo do enake tabele kot v primeru podatkov brez šuma (tabela 15.3). Identificirana diferenčna enačba modela pa je v tem enaka

$$y(k+2) = 1.9447y(k+1) - 0.9535y(k) + 0.0177u(k) .$$

15.5 Povzetek

Obravnavali smo metodo zunanje dinamike pri identifikaciji nelinearnih dinamičnih modelov. Predstavlja enostavno razširitev linearne vhodno-izhodnega modela, ki je široko uporaben. Najpogostejsi nelinearni dinamični strukturi sta model NARX in model NOE. Za oceno parametrov modela NARX optimiramo enokoračno predikcijo, ki omogoča uporabo linearne optimizacije, če je model linearen v parametrih. V primeru modela NOE optimiramo simulacijski pogrešek, ki zahteva uporabo nelinearne optimizacije.

Za strukturno identifikacijo lahko uporabimo metodo Lipschitzevih koeficientov ali pa metodo, ki temelji na uporabi PCA.

Del V

Odločanje na osnovi modela

16. Adaptivno vodenje

V štiridesetih letih prejšnjega stoletja se je v tehniški literaturi pojavil izraz *adaptivnost*. V tem času je postal jasno, da so adaptivni sistemi poseben razred nelinearnih sistemov. Nelinearnost vnaša v sistem določene lastnosti, ki jih ni moč opaziti pri linearnih sistemih. Težave pri iskanju splošno veljavne definicije adaptivnih sistemov, kot tudi pri iskanju splošnih tehnik za analizo in sintezo, lahko pripisemo prav temu dejству.

Načrtovalne tehnike v teoriji sistemov so tesno povezane s stabilnostnimi značilnostmi sistema. Stabilnostne razmere je relativno enostavno ugotoviti v primeru linearnih sistemov, saj za to obstaja širok nabor metod. Posledica tega je tudi množica različnih tehnik načrtovanja vodenja na področju linearnih sistemov. Nasprotno pa na področju nelinearnih sistemov ne poznamo splošnih metod načrtovanja vodenja, saj lahko stabilnost ugotavljam le na posameznih primerih in ne splošno.

Vprašanja vodenja v splošnem, posebno pa vprašanja avtomatskega vodenja postajajo vse pomembnejša v modernem življenju, saj postajajo sistemi, ki nas obdajajo vse bolj zahtevni in med seboj vse bolj povezani. Zahteve vodenja, kot kvalitativnega vpliva na obnašanje sistema z zahtevo po približevanju zaželenemu obnašanju, so vedno večje. Na ta način postaja klasična teorija nezadostna, saj ne zagotavlja primerenega obnašanja v primeru sistemov, katerih značilnosti se s časom spreminjajo in so obdani s spreminjačim se okoljem. Posledica tega je pojav teorije adaptivnega vodenja.

Pojem *adaptacija* se pojavi prvič v biologiji in predstavlja *koristno prilagoditev organizma na spremembe v njegovem okolju*. Navdahnjena s to idejo sta leta 1957 Drenick in Shabender vpeljala termin *adaptivni sistem* na področje teorije vodenja [90]. Po njuni definiciji so adaptivni tisti sistemi, ki nadzorujejo svoje obnašanje in korigirajo lastna stanja in parametre v smislu ugodnejšega obnašanja.

Z razvojem adaptivne teorije se je pojavilo veliko število novih definicij, ki vsaka na svoj način razjasnjujejo bistvo adaptivnega vodenja. Posplošenje teh definicij, ne da bi pri tem izgubili del informacije, ni prineslo uspeha, kar lahko intuitivno pripisemo nelinearni naravi sistemov, katerih značilnosti skušamo opredeliti. S predstavljivijo nekaterih definicij skušamo predstaviti naravo adaptivnih sistemov.

Definicija 1 *Adaptivni sistem je sistem, ki neprestano nadzira svoje obnašanje in v smislu približevanja optimalni rešitvi izbranega kriterija spreminja svoje parametre in stanja.*

Definicija 2 *Adaptivno vodenje je inteligentno povratnozančno vodenje, ki spreminja svoje parametre in stanja v spreminjačem okolju tako, da deluje optimalno v smislu zadovoljevanja izbranega kriterija.*

Definicija 3 *Leta 1959 Bellman in Kalaba predstavita termin adaptivno vodenje v smislu vodenja sistema, o katerem nimamo popolne informacije. Po tej definiciji predstavlja adaptivno vodenje zadnjo stopnjo v razvoju vodenja procesov.*

V smislu vodenja lahko govorimo o treh osnovnih konceptih, ki so odvisni od informacije, ki jo imamo o sistemu.

- V primeru, ko imamo o procesu, ki je predmet vodenja, popolno informacijo, takrat lahko govorimo o deterministično vodljivem procesu.
- Ko so v procesu prisotni elementi, ki so neznani in jih lahko matematično zapišemo kot naključne spremenljivke z znano distribucijo, takrat govorimo o stohastično vodljivem procesu.
- Tretja stopnja so sistemi, kjer je informacija še manjša. V tem primeru mora regulator izboljševati svoje obnašanje na temelju opazovanja izhodov sistema (adaptivni algoritem). Z zadostno informacijo, postanejo tudi odločitve regulatorja pravilne v smislu izboljševanja obnašanja sistema. Te sisteme definirata Bellman in Kalaba kot adaptivno vodljive sisteme.

16.1 Principi adaptivnega vodenja

Živi organizem, kot prototip adaptivnega sistema, mora obvladovati spremembe v svojem okolju za svoje preživetje, rast in razvoj. Dobro poznavanje okolja omogoča organizmu predvidevanje sprememb, ki so ključne za njegovo preživetje. Vendar sta razumevanje in nadzorovanje okolja dve različni dejavnosti, zato predvidevanje sprememb na temelju poznavanja sistema ni vedno zadostno za pravilno dejstvo organizma, ki bi vodila k optimalni rešitvi. Vidimo, da obstaja zelo tesna zveza med razumevanjem ali *identifikacijo* okolja in *odločanjem ali vodenjem*.

Implikacijo zgornjega razmišljanja na področju avtomatskega vodenja procesov je prvi podal Feldbaum leta 1965. Predpostavimo, da imamo regulator A in proces B , za katerega načrtujemo vodenje. Problem lahko definiramo kot *determinističen*, *stohastičen* ali *adaptiven* glede na informacijo o A in B , ki je dostopna. Če so, na primer, karakteristike procesa B znane, potem lahko načrtamo izbrano strategijo vodenja z eno od znanih klasičnih metod. Na drugi strani je potrebno v primeru neznanih karakteristik sistema B le-te najprej identificirati. V tem primeru mora algoritem vodenja simultano reševati dva problema, ki sta tesno povezana, a različnega značaja. Najprej mora na osnovi zbranih informacij, determinirati značilnosti sistema. Nakar mora na osnovi teh informacij determinirati način ukrepanja, ki je potrebno za uspešno vodenje. Prvi problem je problem identifikacije sistema in drugi problem vodenja. Celoten algoritem imenujemo *dualno vodenje*.

Da bi zagotovili primerno obnašanje sistema, moramo v primeru spremembe parametrov procesa in spremembe okolja, v katerem dinamični sistem deluje, zagotoviti kompenzacijo spremenjenih parametrov procesa s pomočjo parametrov regulatorja. Kompenzacijo lahko dosežemo s spremenjanjem parametrov regulatorja pri nespremenjeni strukturi regulatorja ali pa s spremenjanjem same strukture regulatorja. V prvem primeru govorimo lahko o *parametrično adaptivnem sistemu*, v drugem pa o *struktурно adaptivnem sistemu*. V nadaljevanju se bomo posvetili obravnavi *parametrično adaptivnih sistemov*.

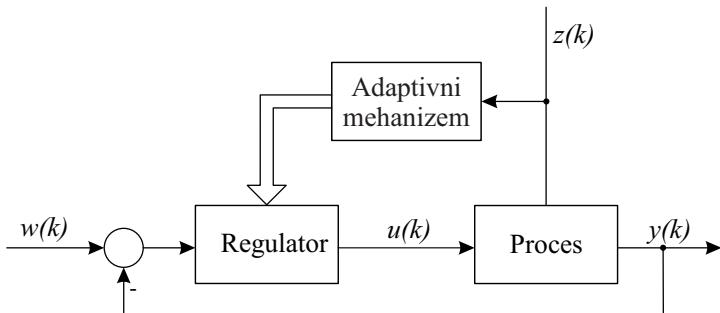
16.2 Razdelitev adaptivnih sistemov glede na vodljivostno shemo

V tem delu si bomo ogledali najpomembnejše strukture adaptivnih sistemov. Glavna lastnost, ki označuje adaptivne sisteme je pridobivanje informacije o neznanem procesu med njegovim delovanjem in spremenjanje vodljivostnih zakonov glede na te informacije. Lahko rečemo, da splošno sprejemljive definicije *adaptivnih sistemov* pravzaprav ni. Zato lahko večino adaptivnih sistemov razdelimo glede na njihovo strukturo. Razdelimo jih na *odprtozančene adaptivne sisteme* in *zaprtozančne adaptivne sisteme*. Adaptivni algoritem je lahko v zaprti zanki ali pa v odprtih, brez povratne informacije.

16.2.1 Odprtozančni adaptivni sistemi

Odprtozančni adaptivni sistem temelji na predpostavki, da je mogoče spreminjanje obnašanja procesa dognati s pomočjo meritev signalov, ki so prisotni v sistemu. Če je mogoče glede na to informacijo načrtati vodenje sistema, potem je implementacija odprtozančnega adaptivnega sistema upravičena. Za realizacijo odprtozančnega adaptivnega sistema je potrebno poznati naravo vpliva merjnih signalov na obnašanje sistema. Prvi primeri zgornjega algoritma so znani že iz leta 1950 pod imenom *razvrščanje ojačanj* (ang. *gain scheduling*). Princip se je razširil tudi v smeri nastavljanja drugih parametrov. Princip temelji na dejstvu, da lahko za vsako delovno območje ali vrednost določenega merjenega signala, izračunamo parametre regulatorja. Vrednosti parametrov so shranjene v pravilnostni tabeli. Ko se pojavi določena vrednost merjenega signala, lahko na osnovi te vrednosti iz pravilnostne tabele določimo vrednost pripadajočih parametrov regulatorja.

Prednost odprtozančnih adaptivnih sistemov je v hitri reakciji na spremembe procesa, ki so predvidene že vnaprej in znane in jih tako ni potrebno identificirati v času eksperimenta. Pomanjkljivosti pa so v tem, da so zanemarjene vse nepredvidene spremembe v sistemu, hkrati pa je število shranjenih parametrov, ki so predvideni za določene pogoje delovanja, lahko zelo veliko. Princip je tudi omejen na procese, kjer se parametri počasi spreminjajo. Značilno shemo odprtozančnega adaptivnega sistema prikazuje slika 16.1, kjer je $z(k)$ merljivi signal, ki da informacijo o pogojih delovanja in na osnovi katerega izračunamo ali izberemo predefinirane parametre regulatorja.



Slika 16.1. Odprtozančni adaptivni sistem.

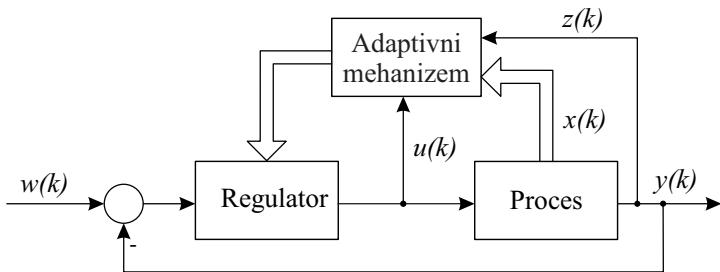
16.2.2 Zaprtozančni adaptivni sistemi

V primeru, ko sprememb v delovanju procesa ne moremo enolično zaznati z meritvijo eksternih signalov, ki nastopajo v sistemu, je potrebno problem reševati z zaprtozančnim adaptivnim principom. Struktura zaprtozančnega adaptivnega sistema je prikazana na sliki 16.2.

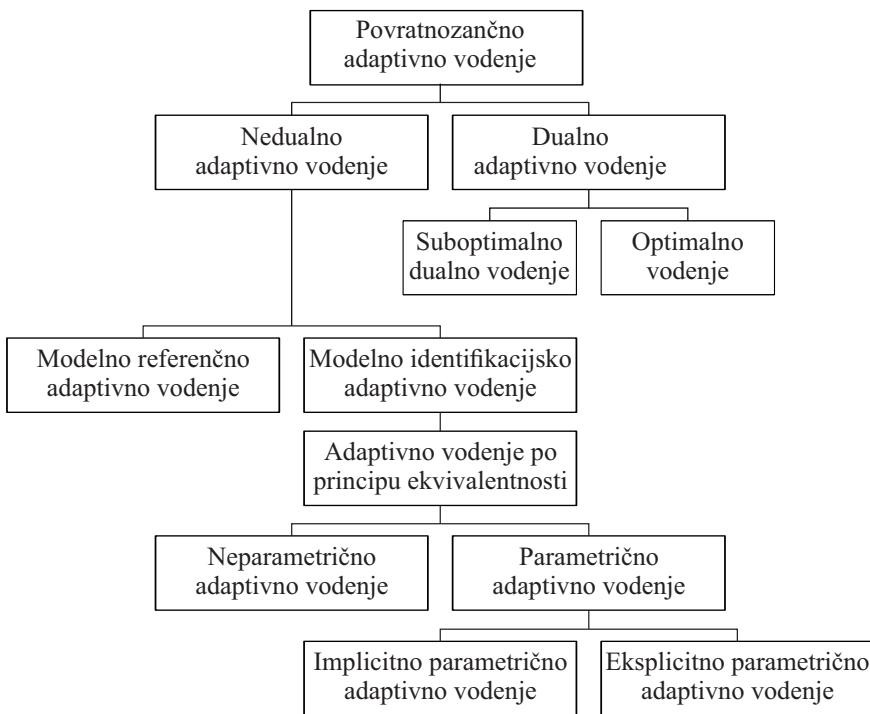
Zaprtozančne adaptivne sisteme lahko naprej delimo na *dualne* in *nedualne*. Delitev zaprtozančnih adaptivnih sistemov, ki bodo predmet naše obravnave, prikazuje slika 16.3.

Dualno adaptivno vodenje. Princip dualnega adaptivnega vodenja sloni na optimizaciji izbranega performančnega kriterija pri predpostavki, da imamo popolno informacijo o internih in eksternih spremenljivkah sistema in da hkrati optimiramo in popolnoma poznamo tudi obnašanje sistema v prihodnosti. Princip imenujemo aktivno učenje ali *optimalno dualno vodenje* (Feldbaum, 1965).

Načrtovanje optimalnega dualnega vodenja je zelo kompleksno in ga je mogoče realizirati le numerično, z uporabo t.i. dinamičnega programiranja (Bellman, 1961). Ta princip načrtovanja



Slika 16.2. Zaprtozančni adaptivni sistem.



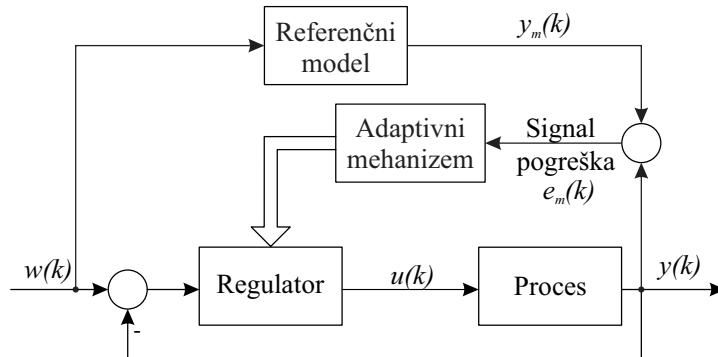
Slika 16.3. Delitev zaprtozančnih adaptivnih sistemov.

regulatorja predpostavlja, da poznamo obnašanje procesa v prihodnosti in predpostavlja, da so apriori znane stohastične lastnosti procesa. Tem zahtevam v realnem primeru ni možno zadovoljiti, zato za praktično uporabo ta pristop ni primeren.

S suboptimalnim dualnim adaptivnim vodenjem poiškušamo preseči pomanjkljivosti optimalnega principa s tem, da optimiramo kriterijsko funkcijo regulatorja. Celoten princip rezultira v postopku aktivnega učenja.

Nedualno adaptivno vodenje. Nedualni principi adaptivnega vodeneja temeljijo na optimizaciji določene kriterijske funkcije v kateri so vsebovane le trenutne in pretekle vrednosti signalov, ki nastopajo v sistemu in so lahko stanja sistema, ocenjena stanja sistema in vhodno-izhodni signali sistema. Postopek načrtovanja je tesno povezan s principom separacije (*ang. separation*) in principom ekvivalentnosti (*ang. certainty equivalence principle*), saj načrtujemo vodenje na podlagi identificiranega procesa in pri tem predpostavimo ekvivalenco med identificiranim modelom in dejanskim procesom. Glede na ta princip lahko nedualne adaptivne sisteme naprej delimo na modelno-referenčne adaptivne sisteme (MRAC) in na modelno-identifikacijske adaptivne sisteme (MIAC) [55].

Modelno-referenčno adaptivno vodenje. Modelno-referenčno adaptivno vodenje je koncipirano tako, da zaprtozančni odziv sistema sledi želeni referenci, ki je dana z referenčnim modelom. Princip zahteva merljivo referenco $w(t)$, ki je hkrati vhod v referenčni model. Parametre regulatorja nastavljamo v skladu z adaptivnim zakonom tako, da minimiziramo pogrešek med izhodom referenčnega modela in izhodom procesa. Adaptivni zakonom dobimo na osnovi optimalne teorije. Minimizacija skupaj s predpostavko določenih omejitev vodi do adaptivnega algoritma, ki je hkrati nelinearen in časovno spremenljiv. Zakon adaptacije je v primeru modelno-referenčnih adaptivnih sistemov razdeljen v tri dele: primerjava zaprtozančnega delovanja z referenčnim modelom, izračun parametrov regulatorja glede na adaptivni zakon, ki ga dobimo na osnovi optimalne teorije in sprememba parametrov regulatorja. Značilna shema modelno-referenčnega adaptivnega vodenja je predstavljena na sliki 16.4.



Slika 16.4. Shema modelno-referenčnega adaptivnega vodenja.

Poznamo več pristopov k modelno-referenčnemu adaptivnemu vodenju, ki se razlikujejo predvsem glede na adaptivni zakon. Znana sta predvsem gradientni princip in globalno stabilni princip na osnovi Lyapunove teorije.

Primer: direktni adaptivni modelno referenčni regulator po Lyapunovu. Adaptivni modelno referenčni regulator spada v kategorijo direktnih adaptivnih regulatorjev. Namens teh regulatorjev je zagotoviti asimptotično sledenje izhoda procesa izhodu referenčnega modela. Modelno referenčni adaptivni regulatorji so globalno stabilni, če veljajo naslednje predpostavke o procesu: proces mora biti fazno minimalen, vpliv motenj ni prisoten, nimamo nemodelirane dinamike procesa, so linearni, časovno invariantni, znan pa mora biti tudi predznak ojačenja procesa. V praksi je včasih težko izpolniti te predpostavke. K sreči lahko nekatere predpostavke obidemo za ceno večje kompleksnosti regulacijskega algoritma. Kompleksni algoritmi niso preveč zaželenjeni za implementacijo v praksi, saj zahtevajo od inženirjev precejšnje poznavanje matematike. Adaptivno modelno referenčni regulator predstavljen v tem poglavju je preprost za implementacijo, hkrati pa zagotavlja stabilno delovanje, saj je njegov algoritem adaptacije regulirnega zakona zasnovan tako, da zadovolji kriterijem stabilnosti. Algoritem je izpeljan iz Lyapunovega stabilnostnega kriterija.

Pri izpeljavi regulacijskega zakona predpostavljamo, da imamo proces prvega reda, ki ga lahko zapišemo v obliki prenosne funkcije kot

$$G_p(s) = \frac{Y_p(s)}{U(s)} = \frac{b}{s+a}, \quad (16.1)$$

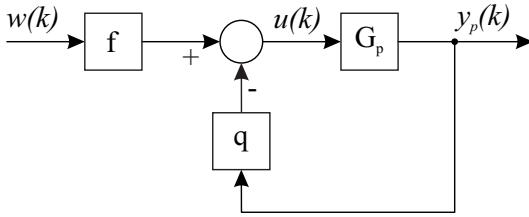
kjer je $Y_p(s)$ izhod procesa in $U(s)$ vhod v proces. Želimo, da izhod procesa sledi referenčnemu modelu prvega reda s prenosno funkcijo

$$G_r(s) = \frac{Y_r(s)}{W(s)} = \frac{b_r}{s + a_r} , \quad (16.2)$$

kjer je $Y_r(s)$ izhod referenčnega modela in $W(s)$ referenca. Za zagotavljanje sledenja procesa referenčnemu modelu moramo določiti vrednosti odprtozančnega ojačanja f in ojačanja povratne zanke q , ki nastopata v spodnjem regulacijskem zakonu:

$$u(k) = fw(k) - qy_p(k) . \quad (16.3)$$

Bločna shema regulacijskega zakona dana na sliki 16.5.



Slika 16.5. Osnovna bločna shema za adaptivni modelno referenčni regulator.

Zaprtovančna prenosna funkcija procesa G_p reguliranega z regulacijskim zakonom v enačbi 16.3 je

$$G_w(s) = \frac{Y_p(s)}{W(s)} = \frac{fb}{s + a + bq} . \quad (16.4)$$

Kot smo omenili na začetku želimo, da izhod našega procesa sledi izhodu referenčnega modela. Torej mora biti napaka med referenčnim modelom in našo zaprtovančno prenosno funkcijo čim manjša. Napako določimo, tako da odštejemo diferencialni enačbi obeh prenosnih funkcij. Diferencialna enačba za referenčni model se glasi

$$\dot{y}_r(t) + a_r y_r(t) = b_r w(t) , \quad (16.5)$$

za zaprtovančno prenosno funkcijo pa

$$\dot{y}_p(t) + (a + bq)y_p(t) = fbw(t) . \quad (16.6)$$

Če enačbi odštejemo, dobimo

$$\dot{y}_p - \dot{y}_r + (a + bq)y_p - a_r y_r = fbw - b_r w . \quad (16.7)$$

Vsaki strani enačbe prištejemo $a_r y_p$ in upoštevamo, da je napaka $e = y_p - y_r$

$$\dot{e} + a_r e = (fb - b_r)w - ((a + bq) - a_r)y_p . \quad (16.8)$$

Za parametre uvedemo pomožni spremenljivki

$$\tilde{b} = (fb - b_r) , \quad (16.9)$$

$$\tilde{a} = (a + bq) - a_r . \quad (16.10)$$

Adaptivni zakon za ojačanja f in q dobimo s pomočjo pozitivno definitne Lyapunove funkcije

$$V(e, \tilde{a}, \tilde{b}) = e^2 + \frac{1}{\gamma_f} \tilde{b}^2 + \frac{1}{\gamma_q} \tilde{a}^2 , \quad (16.11)$$

ki ima odvod enak

$$\dot{V}(e, \tilde{a}, \tilde{b}) = -2a_r e^2 + 2\tilde{b}we - 2\tilde{a}y_p e + \frac{2}{\gamma_f} \tilde{b}\dot{\tilde{b}} + \frac{2}{\gamma_q} \tilde{a}\dot{\tilde{a}}, \quad (16.12)$$

kjer smo pri odvajjanju napake upoštevali enačbo 16.8. Da zagotovimo stabilnost mora veljati $\dot{V}(e, \tilde{a}, \tilde{b}) \leq 0$. Ob predpostavki časovne invariantnosti procesa nas pripelje to do naslednjega zakona adaptacije

$$\dot{\tilde{b}} = -\frac{\gamma_f}{b} ew, \quad (16.13)$$

$$\dot{\tilde{a}} = \frac{\gamma_q}{b} ey_p, \quad (16.14)$$

ker je $\dot{\tilde{b}} = \dot{f}b + f\dot{b}$ in $\dot{\tilde{a}} = \dot{q}b + q\dot{b}$ in ker predpostavimo počasne spremembe parametrov procesa ($\dot{b} \approx 0$, je $\dot{\tilde{b}} = \dot{f}b$ in $\dot{\tilde{a}} = \dot{q}b$). Zakon adaptacije za ojačenje f in q je potem enak

$$\dot{f} = -\frac{\gamma_f}{b} ew, \quad (16.15)$$

$$\dot{q} = \frac{\gamma_q}{b} ey_p. \quad (16.16)$$

Zakon adaptacije lahko poenostavimo, če poznamo predznak ojačenja oziroma parametra b , v naslednjo obliko:

$$\dot{f} = -\gamma_f^* sign(b)ew, \quad (16.17)$$

$$\dot{q} = \gamma_q^* sign(b)ey_p, \quad (16.18)$$

kjer sta γ_f^* in γ_q^* poljubni pozitivni konstanti.

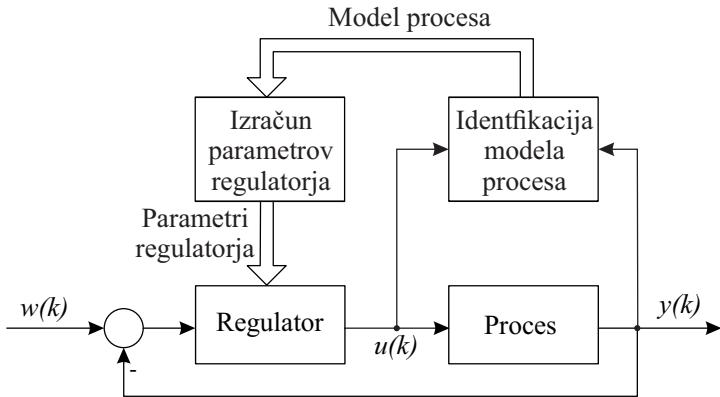
Predstavljeni regulator je globalno stabilen v okolici delovne točke, kjer je opis procesa z linearnim modelom zadovoljiv. Če imamo opravka z nelinearnim procesom, katerega želimo regulirati skozi celotno delovno območje, pa je potrebno regulator razširiti. To najlažje storimo z vpeljavo koncepta mehke logike.

Modelno-identifikacijsko adaptivno vodenje. Modelno identifikacijsko adaptivno vodenje (MIAC) imenujemo včasih tudi samoorganizirajoče ali samonastavljivo adaptivno vodenje. Značilne za ta princip so tri faze: pridobivanje informacije o trenutnem obnašanju procesa, optimiranje določene kriterijske funkcije in prilagajanje regulatorja. Glede na način pridobivanja informacije lahko ločimo več različnih tipov modelno identifikacijskih adaptivnih sistemov. Parametri regulatorja se izračunavajo tako, da izpolnimo določene performančne kriterije ali zaprtozančno obnašanje. Prilagajanje regulatorja pomeni zamenjavo starih parametrov regulatorja z novimi izračunanimi vrednostmi.

Bistvo pristopa je v doseganju optimalnega modeliranja procesa in optimalnega obnašanja zaprtozančnega sistema ne glede na vhodni signal in na pogoje delovanja. Značilno strukturo prikazuje slika 16.6.

Adaptivni mehanizem sestoji iz identifikacije procesa in procedure za načrtovanje regulatorja. Identifikacija procesa temelji izključno na vhodno-izhodnih podatkih procesa. Parametre regulatorja izračunamo na osnovi ocenjenih parametrov procesa s tem, da zanemarimo odstopanja ocen od dejanskih parametrov. Algoritmi uporabljeni za identifikacijo in načrtovanje regulatorja se izračunavajo sprotno (*ang. on-line*) in v realnem času (*ang. in real-time*).

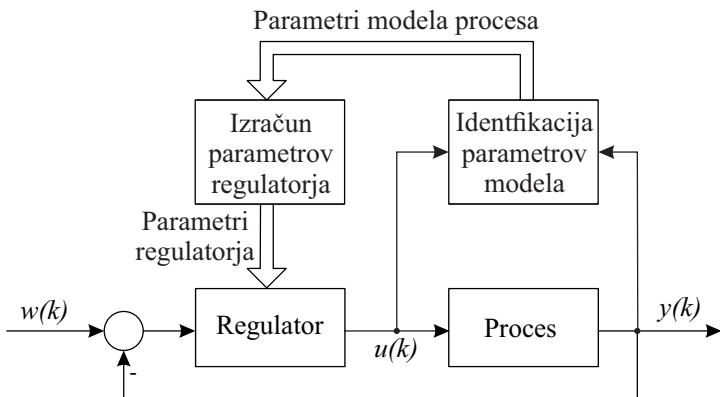
Zgoraj omenjeni princip modelno-identifikacijskega adaptivnega vodenja temelji na principu separacije. O principu separacije govorimo, če je ocenjevanje parametrov procesa ločeno od načrtovanja regulatorja.



Slika 16.6. Shema modelnega identifikacijskega adaptivnega vodenja.

Princip ekvivalentnosti temelji na principu separacije in predpostavlja enakost ocenjenih parametrov procesa ali spremenljivk stanja in dejanskih parametrov procesa ali dejanskih spremenljivk stanja. Za izračun parametrov regulatorja predpostavimo enakost modela procesa in dejanskega procesa.

Adaptivno vodenje, ki temelji na principu ekvivalentnosti lahko naprej delimo na parametrično adaptivno in neparametrično adaptivno vodeneje. Razdelitev temelji na identifikaciji, ki je v prvem primeru v parametrični obliki, v drugem primeru pa v neparametrični obliki (npr. impulzni odziv). Struktura parametričnih adaptivnih sistemov je prikazana na sliki 16.7.



Slika 16.7. Shema parametričnega adaptivnega sistema.

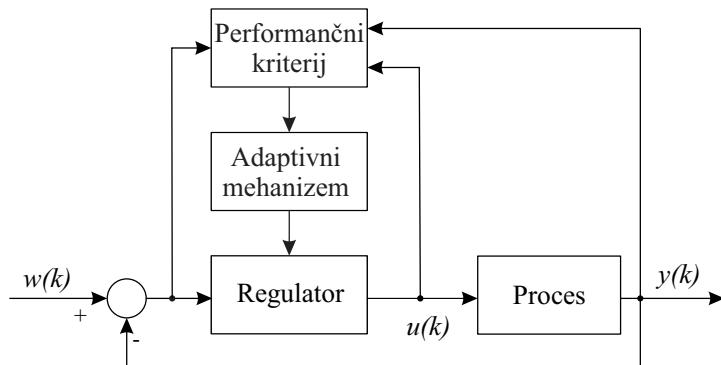
Princip ekvivalence je teoretično upravičen v primeru konstantnih parametrov procesa, pri kvadratični kriterijski funkciji estimiranih parametrov in pri belem šumu na procesu. Pri stohastičnih spremembah parametrov procesa je princip veljaven samo v primeru, če so spremembe parametrov statistično neodvisne. Povzamemo lahko, da princip ekvivalentnosti splošno ne velja, čeprav se pogosto uporablja pri načrtovanju adaptivnega vodenja.

16.3 Razdelitev adaptivnih sistemov glede na njihove funkcionalne podsklope

Druga razdelitev adaptivnih sistemov, ki jo podajata Narendra in Annawamy [90] je zelo groba razdelitev in temelji na funkcionalnih podsklopih adaptivnega sistema. Glede na to delimo adaptivno vodenje na *direktno* in *indirektno*.

16.3.1 Direktno adaptivno vodenje

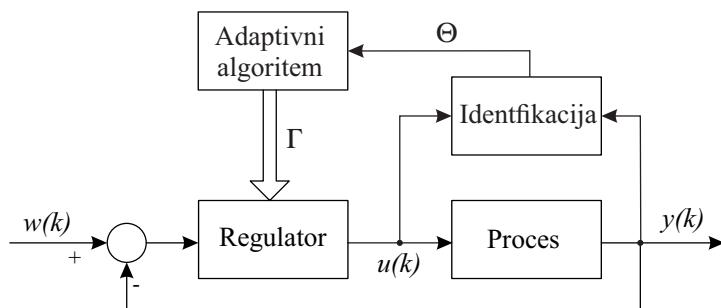
Prvi pristop imenujemo *direktno adaptivno vodenje*. Ta pristop je možen v nekaterih primerih, ko lahko modificiramo sistem direktno tako, da modificiramo parametre regulatorja glede na določene performančne kriterije, katerim mora zadostiti celoten sistem. Princip direktnega adaptivnega vodenja prikazuje slika 16.8.



Slika 16.8. Bločna shema direktnega adaptivnega sistema.

16.3.2 Indirektno adaptivno vodenje

Drugi pristop imenujemo *indirektno vodenje*. V tem primeru identificiramo parametre procesa s pomočjo ene od metod sprotne identifikacije (*ang. on-line*) in na osnovi teh parametrov modificiramo parametre vodenja. Posplošeno shemo indirektnega parametrično adaptivnega vodenja predstavlja slika 16.9. Vektor $\underline{\theta}$ predstavlja vektor parametrov procesa in vektor $\underline{\Gamma}$ predstavlja vektor parametrov regulatorja.



Slika 16.9. Bločna shema indirektnega adaptivnega sistema.

16.4 Mehko adaptivno vodenje

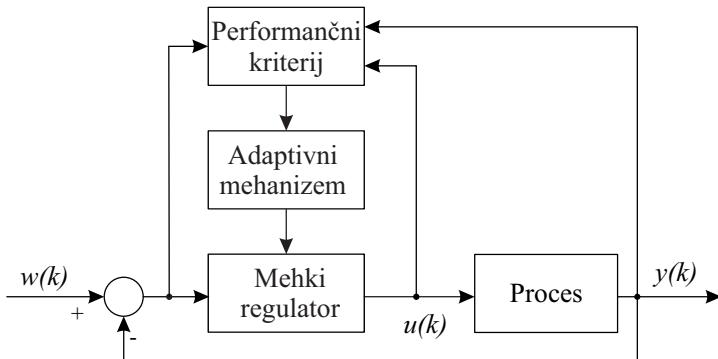
Razvoj mehke logike v poznih 70ih letih je izval veliko število aplikacij in razširitev osnovnega mehkega regulatorja, ki ga je predlagal Mamdani. Osnovni princip načrtovanja mehkih regulatorjev temelji na enostavnem pogojevanju v obliki *if ... then* stavkov, ki je zelo blizu človekovemu načinu razmišljanja. Celoten mehanizem načrtovanja temelji na principu *poizkusi in popravi*

(ang. *trial and fail*), ker splošne metode za načrtovanje trenutno še ne obstajajo. Da bi rešili problem načrtovanja in druge probleme, ki so povezani z nelinearno dinamiko in spremenljivimi parametri, se je pojavil mehki adaptivni pristop.

Enako kot klasično adaptivno vodenje lahko tudi mehko adaptivno vodenje razdelimo na *odprtozančno* in *povratnozančno mehko vodenje*. Glede na strukturo pa ga delimo na *direktno* in *indirektno mehko adaptivno vodenje*. Prvi pristop na tem področju sta predlagala Procyk in Mamdani in sodi v skupino direktnega mehkega adaptivnega vodenja. Nato so se pojavili tudi pristopi, ki sodijo v skupino indirektnega mehkega adaptivnega vodenja. Med seboj se razlikujejo predvsem v različnih mehkih identifikacijskih metodah. Strukturno gledano pa temeljijo na inverznem modeliranju procesa in njegovim vključevanjem v vodljivostno shemo. Tak pristop predlagajo Czogala in Pedrycz [22], Moore in Harris [84].

16.4.1 Direktno mehko adaptivno vodenje

Direktni mehki adaptivni sistem lahko načrtamo v obliki samonastavljevega (ang. *self-regulating, self-adjusting*) vodenja. Značilna shema direktnega mehkega adaptivnega sistema sestoji iz pomožnega dela, ki je lahko referenčni model, pomožni regulator, nadzornik ali mehanizem za nastavljanje parameterov mehkega regulatorja v direktni veji. Shema direktnega mehkega adaptivnega vodenja je prikazana na sliki 16.10.



Slika 16.10. Bločna shema direktnega mehkega adaptivnega vodenja.

Prvi mehki samonastavljeni sistem ali (ang. *Self-Organizing Controller* - SOC), ki sta ga predlagala Procyk in Mamdani temelji na modifikaciji pravilnostne tabele glede na model procesa, ki je podan v zelo grobi obliki, zato tudi govorimo o direktnem principu mehkega adaptivnega vodenja. Metoda z nastavljanjem pripadnostnih funkcij glede na določen kriterij je bila razvita kasneje. Direktni mehki adaptivni sistem lahko načrtamo tudi v smislu mehkega nastavljanja klasičnih PID (PI, PD) regulatorjev s pomočjo mehke logike. V tem primeru nastavljamo parameterje regulatorja glede na določeno kriterijsko funkcijo, določene performančne kriterije kot so: čas vzpona, prenihaj, nastavitevni čas, pogrešek v ustaljenem stanju, itd. Če povzamemo zgornje lahko rečemo, da je mogoče samonastavljeni mehki regulator (STFC) načrtati kot regulator, ki se prilagaja s spremjanjem mehkih pravil, pripadnostnih funkcij, skalirnih faktorjev vhodnih in izhodnih signalov mehkega reglatorja.

Primer: direktni mehki adaptivni modelno-referenčni regulator po Lyapunovu. Podobno kot v primeru MRAC lahko naredimo izpeljavo tudi za mehki modelno-referenčni regulator (ang. *Direct Fuzzy Model-Reference Adaptive Control* - DFMRAC). Zaradi preprostosti

izpeljave ponovno predpostavljamo, da je proces, ki ga želimo regulirati, prvega reda. Pri direktnem mehkem modelno-referenčnem regulatorju razdelimo delovno področje na več manjših mehkih podpodročij oziroma rojev na katerih lahko proces zadovoljivo dobro opišemo z linearnim modelom. Področja lahko določimo ročno ali pa s pomočjo metod rojenja. Za vsako podpodročje definiramo svoje ojačenje f_i in q_i . Dobimo vektorja ojačenj

$$\underline{f} = [f_1, f_2, \dots, f_c]^T , \quad (16.19)$$

$$\underline{q} = [q_1, q_2, \dots, q_c]^T . \quad (16.20)$$

Nelineareni proces lahko opišemo z mehkim modelom kot uteženo vsoto linearnih modelov, ki veljajo vsak na svojem mehkem podpodročju. Linearni model opišemo z diferencialno enačbo prvega reda

$$\dot{y}_{p,i} = -a_{i,p}y_p + b_{i,p}u , \quad (16.21)$$

kjer je y_p izhod procesa in $\dot{y}_{p,i}$ odvod izračunan glede na i -ti lokalni model. Celoten odvod \dot{y}_p je enak vsoti

$$\dot{y}_p = \sum_{i=1}^c \beta_i \dot{y}_{p,i} , \quad (16.22)$$

kjer je c število rojev oziroma lokalnih linearnih modelov in β_i normirana pripadnost k posameznemu roju. Izhod mehkega modela lahko zapišemo v matrični obliki kot

$$\dot{y}_p = -\underline{\beta}^T \underline{a} y_p + \underline{\beta}^T \underline{b} u , \quad (16.23)$$

kjer je $\underline{\beta} = [\beta_1, \beta_2, \dots, \beta_c]^T$, $\underline{a} = [a_1, a_2, \dots, a_c]^T$ in $\underline{b} = [b_1, b_2, \dots, b_c]^T$. Podobno, kot pri mehkem prediktivno funkcijskem regulatorju lahko tudi tukaj govorimo o globalnem linearinem modelu, katerega trenutni parametri so

$$\tilde{a} = \underline{\beta}^T \underline{a} , \quad (16.24)$$

$$\tilde{b} = \underline{\beta}^T \underline{b} . \quad (16.25)$$

Za izpeljavo adaptivnega algoritma DFMRAC predpostavimo naslednji regulacijski zakon:

$$u = \frac{1}{\underline{\beta}^T \underline{b}} (\underline{\beta}^T \underline{f}^* w - \underline{\beta}^T \underline{q}^* y_p) . \quad (16.26)$$

Regulacijskega zakona, ki je podan z enačbo 16.26, ni mogoče realizirati, saj običajno ne poznamo vektorja parametrov \underline{b} . Tako kot v linearinem primeru zapišemo enačbo za zaprtozančni sistem

$$\dot{y}_p = -\underline{\beta}^T \underline{a} y_p + \underline{\beta}^T \underline{f}^* w - \underline{\beta}^T \underline{q}^* y_p . \quad (16.27)$$

Referenčni model podamo v obliki mehkega modela kot

$$\dot{y}_r = -\underline{\beta}^T \underline{a}_r y_r + \underline{\beta}^T \underline{b}_r w , \quad (16.28)$$

$$\underline{a}_r = [a_{r,1}, a_{r,2}, \dots, a_{r,c}]^T , \quad (16.29)$$

$$\underline{b}_r = [b_{r,1}, b_{r,2}, \dots, b_{r,c}]^T . \quad (16.30)$$

Ko odštejemo diferencialni enačbi 16.27 in 16.28, dobimo

$$\dot{e} + \underline{\beta}^T \underline{a}_r e = \underline{\beta}^T (\underline{f}^* - \underline{b}_r) w - \underline{\beta}^T (\underline{q}^* + \underline{a} - \underline{a}_r) y_p . \quad (16.31)$$

Uvedemo naslednje pomožne spremenljivke, za bolj strnjen zapis enačb

$$\begin{aligned}\tilde{\underline{b}} &= \underline{f}^* - \underline{b}_r, \\ \tilde{\underline{a}} &= \underline{q}^* + \underline{a} - \underline{a}_r.\end{aligned}\quad (16.32)$$

Enačba 16.31 se tako poenostavi v

$$\dot{e} + \underline{\beta}^T \underline{a}_r e = \underline{\beta}^T \underline{b} w - \underline{\beta}^T \underline{a} y_p. \quad (16.33)$$

Definiramo pozitivno definitno Lyapunovo funkcijo

$$V(e, \tilde{\underline{a}}, \tilde{\underline{b}}) = e^2 + \sum_{i=1}^c \frac{1}{\gamma_{bi}} \tilde{b}_i^2 + \sum_{i=1}^c \frac{1}{\gamma_{ai}} \tilde{a}_i^2, \quad (16.34)$$

ki ima odvod po času enak:

$$\dot{V}(e, \tilde{\underline{a}}, \tilde{\underline{b}}) = 2e\dot{e} + 2 \sum_{i=1}^c \frac{1}{\gamma_{bi}} \dot{\tilde{b}}_i \tilde{b}_i + 2 \sum_{i=1}^c \frac{1}{\gamma_{ai}} \dot{\tilde{a}}_i \tilde{a}_i. \quad (16.35)$$

Če upoštevamo pogoje za stabilnost po Lyapunovu ($V(e, \tilde{\underline{a}}, \tilde{\underline{b}}) > 0$ in $\dot{V}(e, \tilde{\underline{a}}, \tilde{\underline{b}}) \leq 0$), pridemo do naslednjih enačb

$$\begin{aligned}ew \sum_{i=1}^c \beta_i \tilde{b}_i + \sum_{i=1}^c \frac{1}{\gamma_{bi}} \dot{\tilde{b}}_i \tilde{b}_i &= 0, \\ ey_p \sum_{i=1}^c \beta_i \tilde{a}_i + \sum_{i=1}^c \frac{1}{\gamma_{ai}} \dot{\tilde{a}}_i \tilde{a}_i &= 0,\end{aligned}\quad (16.36)$$

iz katerih izrazimo odvod $\dot{\tilde{b}}_i$ in $\dot{\tilde{a}}_i$

$$\begin{aligned}\dot{\tilde{b}}_i &= -\gamma_{bi} ew \beta_i, \\ \dot{\tilde{a}}_i &= \gamma_{ai} ey_p \beta_i.\end{aligned}\quad (16.37)$$

Ob predpostavki časovne invariantnosti procesa velja

$$\begin{aligned}\dot{\tilde{b}} &= \dot{\underline{f}}^*, \\ \dot{\tilde{a}} &= \dot{\underline{q}}^*.\end{aligned}\quad (16.38)$$

Tako dobimo adaptivni zakon za ojačenji \dot{f}_i^* in \dot{q}_i^*

$$\begin{aligned}\dot{f}_i^* &= -\gamma_{bi} ew \beta_i, \\ \dot{q}_i^* &= \gamma_{ai} ey_p \beta_i, \quad i = 1, \dots, c.\end{aligned}\quad (16.39)$$

Ojačenja \dot{f}_i^* in \dot{q}_i^* , $i = 1, \dots, c$, sestavlajo vektorja ojačenj

$$\begin{aligned}\underline{f}^* &= [f_1^*, \dots, f_k^*]^T, \\ \underline{q}^* &= [q_1^*, \dots, q_k^*]^T.\end{aligned}\quad (16.40)$$

Regulacijski zakon iz enačbe 16.26 lahko zapišemo sedaj v naslednji obliki

$$u = \underline{\beta}^T \underline{f} w - \underline{\beta}^T \underline{q} y_p, \quad (16.41)$$

kjer sta vektorja \underline{f} in \underline{q} enaka

$$\begin{aligned}\underline{f} &= \frac{\underline{f}^*}{\underline{\beta}^T \underline{b}}, \\ \underline{q} &= \frac{\underline{q}^*}{\underline{\beta}^T \underline{b}}.\end{aligned}\quad (16.42)$$

Glede na enačbo 16.39 in 16.42 je algoritom adaptacije za ojačenji f_i in q_i enak

$$\begin{aligned}\dot{f}_i &= -\frac{\gamma_{bi}}{\underline{\beta}^T \underline{b}} ew \beta_i, \quad \gamma_{bi} > 0, \\ \dot{q}_i &= \frac{\gamma_{ai}}{\underline{\beta}^T \underline{b}} ey_p \beta_i, \quad \gamma_{ai} > 0.\end{aligned}\quad (16.43)$$

V primeru, da poznamo predznačke parametrov b_i lahko zgornji algoritom adaptacije implementiramo. V večini primerov teh vrednosti ne poznamo. Zato moramo zakon adaptacije prilagoditi temu. Podobno, kot smo to naredili za linearen primer, lahko tudi v primeru mehkega regulatorja adaptivni zakon poenostavimo.

V primeru, da je vektor parametrov \underline{b} konstanten in poznamo predznaček parametra b_i , ki je za vse b_i enak ($sign(b_i) = b_*$, $i = 1, \dots, c$), lahko adaptivni zakon poenostavimo v slednjo obliko

$$\begin{aligned}\dot{\underline{f}} &= -\gamma_f sign(b_*) ew \underline{\beta}, \\ \dot{\underline{q}} &= \gamma_q sign(b_*) ey_p \underline{\beta}.\end{aligned}\quad (16.44)$$

V primeru, da se vektor parametrov \underline{b} spreminja oziroma, da imamo opravka z neznano dinamiko, ki je nismo zajeli v samem modelu, lahko pride do velikih težav pri stabilnosti algoritmov adaptacije. Za doseganje večje stabilnosti in robustnosti adaptivnega algoritma vanj običajno uvedemo faktor puščanja (*ang.leakagetterm*). S tem popolnoma integracijski značaj adaptivnega algoritma v enačbi 16.44 spremenimo v integracijski značaj s puščanjem. S tem se izognemo pobegu parametrov.

En od možnih robustnih rešitev je tudi dinamična normalizacija. Normalizacijski signal m zagotavlja, da bo $\frac{\eta_s}{m} \in \mathcal{L}_\infty$ in $\frac{\psi_f}{m} \in \mathcal{L}_\infty$, kjer je η_s napaka modela. Definiramo ga na naslednji način

$$\begin{aligned}m^2 &= 1 + n_s^2, \\ n_s^2 &= m_s + \underline{\psi}_f^T \underline{\psi}_f + u^2 + y_p^2, \\ \dot{m}_s &= -\delta_0 m_s + u^2 + y_p^2, \\ m_s(0) &= 0,\end{aligned}\quad (16.45)$$

kjer je $\delta_0 > 0$.

Ideja uvedbe faktorja puščanja v adaptivni zakon je v tem, da odvod Lyapunove funkcije postane negativen v prostoru parametrov, ko ti parametri presežejo določene mejne vrednosti. To dosežemo z modifikacijo adaptivnega zakona v naslednjo formo:

$$\begin{aligned}\dot{f}_i &= -\gamma_f sign(b_i) ew \beta_i - \gamma_f |em| \nu_0 f_i \beta_i, \\ \dot{q}_i &= -\gamma_q sign(b_i) ey_p \beta_i - \gamma_q |em| \nu_0 q_i \beta_i, \quad i = 1, \dots, c.\end{aligned}\quad (16.46)$$

Zgornji zakon je zapisan v skalarni obliki. Lahko pa ga zapišemo tudi v matrični obliki

$$\begin{aligned}\dot{\underline{f}} &= -\underline{\Gamma}_f ew \underline{\beta} - \underline{\Gamma}_f |em| \nu_0 \underline{F} \underline{\beta}, \\ \dot{\underline{q}} &= \underline{\Gamma}_q ey_p \underline{\beta} - \underline{\Gamma}_q |em| \nu_0 \underline{Q} \underline{\beta},\end{aligned}\quad (16.47)$$

kjer predpostavljamo, da je $sign(b_i) = sign(b)$ za vse $i = 1, \dots, c$ in

$$\begin{aligned}\underline{\Gamma}_f &= \gamma_f \text{sign}(b) \underline{I}_{k \times k}, \\ \underline{\Gamma}_q &= \gamma_q \text{sign}(b) \underline{I}_{k \times k},\end{aligned}\quad (16.48)$$

in sta \underline{F} in \underline{Q} diagonalni matriki, kjer so f_i , $i = 1, \dots, c$ in q_i , $i = 1, \dots, c$ diagonalni členi. Člen $|\epsilon m| \nu_0$ predstavlja člen puščanja. V literaturi obstajajo različni načini izbire tega člena. Najboljše rezultate dobimo z izbiro člena puščanja po ϵ -modifikaciji. Pri tem je spremenljivka ν_0 definirana kot uporabniško določena konstanta, m je normalizacijski signal in ϵ normirana napaka med referenčnim modelom y_r in izhodom procesa y_p . Adaptivni zakon iz enačbe 16.47 lahko tako predstavimo v bolj kompaktni obliki s naslednjo enačbo

$$\dot{\underline{\theta}} = \underline{\Gamma} \underline{\psi}_f \epsilon - \underline{\Gamma} \underline{\theta}_{diag} \underline{\beta} |\epsilon m| \nu_0, \quad (16.49)$$

kjer je

$$\underline{\Gamma} = \begin{bmatrix} \underline{\Gamma}_f & \underline{0}_{k \times k} \\ \underline{0}_{k \times k} & \underline{\Gamma}_q \end{bmatrix} \quad (16.50)$$

in

$$\underline{\theta}_{diag} = \begin{bmatrix} \underline{F} \\ \underline{Q} \end{bmatrix}. \quad (16.51)$$

Pristop z ϵ -modifikacijo in dinamičnim normiranjem nam zagotavlja, da je

$$\begin{aligned}\epsilon, \epsilon n_s, \underline{\theta}, \dot{\underline{\theta}} &\in \mathcal{L}_\infty, \\ \epsilon, \epsilon n_s, \dot{\underline{\theta}} &\in \mathcal{S}(\nu_0 + \frac{\eta_s^2}{m^2}).\end{aligned}\quad (16.52)$$

Če je $n_s, \underline{\psi}_f \in \mathcal{L}_\infty$ in je prisotno dovolj vzbujanja $\underline{\psi}_f$, ki je neodvisno od η_s potem $\underline{\theta}$ eksponentno konvergira k setu napak

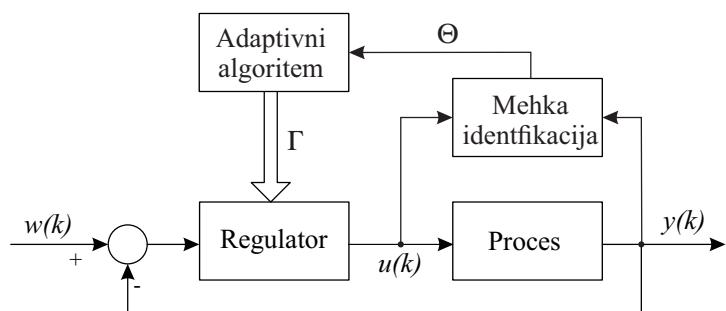
$$\mathcal{D}_\epsilon = \{\underline{\theta} \in \mathcal{R}^n, |\underline{\theta}| \leq c(\nu_0 + \bar{\eta})\}, \quad (16.53)$$

kjer je $c \geq 0$ in $\bar{\eta} = \sup_t \frac{\eta_s}{m}$. Iz enačbe 16.52 je prav tako razvidno, da so $\epsilon, \epsilon n_s, \dot{\underline{\theta}}$ omejeni v smislu \mathcal{L}_{2e} , ocena vektorja parametrov $\underline{\theta}$ pa je konsistentna v smislu srednjekvadratične vrednosti. Simbol \mathcal{L}_{2e} označuje linearni vektorski prostor funkcij $x(t)$ (razširjen Lebesgue prostor), katerih norma \mathcal{L}_{2e} je definirana kot $\|x\| = (\int_0^\tau x(t)^T x(t) dt)^{\frac{1}{2}}$ in obstaja za končen τ .

16.4.2 Indirektno mehko adaptivno vodenje

Indirektno mehko adaptivno vodenje temelji na identifikaciji procesa v mehki oblikih. Celoten algoritmom vodenja je v tem primeru sestavljen iz identifikacije procesa v mehki oblikih in izračuna regulirnega signala na osnovi mehkega modela. Princip indirektnega mehkega adaptivnega vodenja predstavlja sliko 16.11.

Glede na to v kakšni oblikih je mehki model, lahko indirektno mehko adaptivno vodenje delimo naprej na *mehko adaptivno vodenje na osnovi mehkega modela procesa* in *mehko adaptivno vodenje na osnovi inverznega mehkega modela procesa*. V našem delu smo največ prostora namenili indirektnim mehkim adaptivnim principom vodenja. Mehko adaptivno vodenje na osnovi mehkega modela procesa lahko realiziramo v oblikah prediktivnega algoritma. Princip temelji na izračunu regulirnega signala s pomočjo optimalne teorije glede na predikcijo izhoda procesa, ki jo dobimo s pomočjo mehkega modela. Mehko adaptivno vodenje na osnovi inverznega mehkega modela pa je mogoče realizirati v oblikah kompenzacijskoga regulirnega principa.



Slika 16.11. Bločna shema indirektnega mehkega adaptivnega vodenja.

17. Prediktivno vodenje

Prediktivno vodenje (*ang. Model Based Predictive Control - MBPC*) je strategija vodenja, ki temelji na eksplisitni uporabi modela procesa, ki ga vodimo. Model uporabljamo za napovedovanje (predikcijo) regulirane spremenljivke več korakov v prihodnost (*ang. long-range time horizon*). Področje prediktivnega vodenja zajema veliko različnih metod, ki so dobro imena predvsem po kraticah njihovih angleških imen: MAC (*ang. Model Algorithmic Control*) (Richalet et al., [101]), DMC (*ang. Dynamic Matrix Control*) (Cutler, [21]), GPC (*ang. Generalized Predictive Control*) (Clarke et al., [19]), EPSAC (*ang. Extended Prediction Self-Adaptive Control*) (De Keyser in Van Cauwenbergh, [24]), EHAC (*ang. Extended Horizon Adaptive Control*) (Ydstie, [3]), UPC (*ang. Unified Predictive Control*) (Soeterboek, [2]), MUSMAR (Mosca et al., [85]).

V začetku 80-let sta se neodvisno ena od druge pojavili prvi metodi prediktivnega vodenja: MAC in DMC. Skoraj istočasno so se nekatere raziskovalne skupine na področju adaptivnega vodenja začele ukvarjati z večkoračnimi prediktivnimi regulatorji, ki predstavljajo posloštev minimalnovariančnih regulatorjev. Ugotovimo lahko, da vse te metode in tudi mnogo drugih, predlaganih kasneje (npr. PFC, UPC), temeljijo na istih osnovnih principih. Razlikujejo se predvsem v obliki zapisa modela procesa.

S stališča možnosti uporabe pri industrijskih aplikacijah imajo metode prediktivnega vodenja pomembne prednosti pred ostalimi strategijami vodenja:

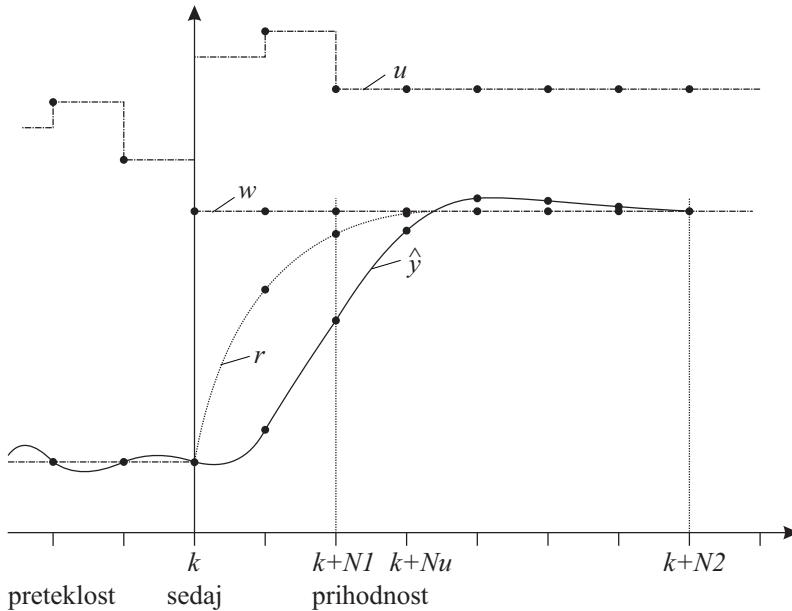
- primerne so za vodenje procesov z zahtevnejšo dinamiko, kot so procesi višjega reda, procesi z mrtvim časom in procesi z neminimalno fazo,
- splošen koncept omogoča vodenje tako univariabilnih kot multivariabilnih procesov,
- omogočajo predkompenzacijo merljivih motenj (*ang. feedforward compensation*),
- pri vodenju lahko upoštevamo omejitve velikosti reguliranega in regulirnega signala ter omejitve hitrosti spremembe regulirnega signala,
- precejšnja prostost pri načrtovanju vodenja, pri čemer gledamo na parametre načrtovanja kot na zahtevane specifikacije,
- prediktivni regulator lahko vnaprej generira ustrezni regulirni signal, če vnaprej poznamo potek referenčne trajektorije (*ang. looking-ahead*),
- ni eksplisitnega odvajanja, zato šum meritev ne povzroča problemov,
- ni eksplicitnega integriranja, zato ni problema integralskega pobega,
- osnovni princip je lahko razumljiv tudi za neeksperte.

Bistvene slabosti metod prediktivnega vodenja pa so:

- potrebujemo dober dinamični model procesa (konstrukcija modela lahko predstavlja celo do 80% celotnega potrebnega časa pri razvoju neke realne aplikacije prediktivnega vodenja),
- nekatere metode so računsko precej zahtevne, kar lahko postane problematično pri vodenju procesov s hitrejšo dinamiko.

17.1 Osnovni principi prediktivnega vodenja

Koncept prediktivnega vodenja je predstavljena na sliki 17.1:

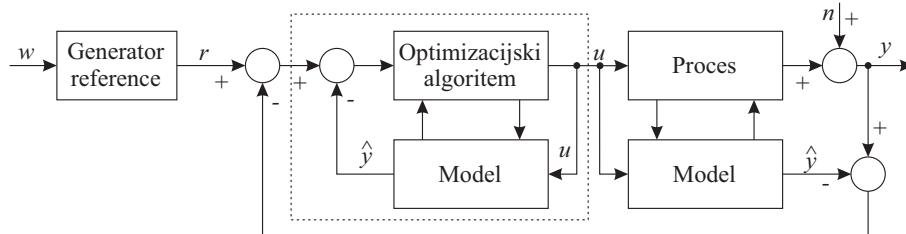


Slika 17.1. Osnovni princip prediktivnega vodenja.

- Ob vsakem diskretnem časovnem trenutku k izračunamo **napoved izhodnega signala procesa** $y(k + j)$ za določen horizont opazovanja v prihodnosti ($j = N_1, \dots, N_2$). Napovedane vrednosti označimo kot $\hat{y}(k + j|k)$ in predstavljajo j -koračno napoved modela, vrednosti N_1 in N_2 pa označujeta **spodnjo in zgornjo vrednost predikcijskega horizonta**. Spodnja in zgornja vrednost predikcijskega horizonta izhodnega signala določata **horizont ujemanja** (ang. *coincidence horizon*), znotraj katerega želimo doseči ujemanje izhodnega signala z referenčnim signalom. Napoved izhodnega signala procesa izračunamo s pomočjo modela procesa od časa k naprej, zato v argumentu to zapišemo za vertikalno črto $|k$. Napovedane vrednosti pa so seveda odvisne tudi od regulirnega scenarija v prihodnosti $u(k + j|k)$, $j = 0, \dots, N_u - 1$, ki ga **nameravamo** uporabiti od trenutka k naprej.
- Definiramo želeno **modelno-referenčno trajektorijo** (ang. *model-reference trajectory*) $r(k + j|k)$, $j = N_1, \dots, N_2$, s katero opišemo kako naj se odziv procesa obnaša od trenutne vrednosti $y(k)$ do predpisane referenčne vrednosti $w(k + N_2)$ (ang. *setpoint trajectory*).
- Vektor prihodnjih regulirnih signalov $(u(k + j|k), j = 0, \dots, N_u - 1)$ izračunamo z **minimizacijo ustreznih kriterijskih funkcij** tako, da minimiziramo napako med $r(k + j|k)$ in $\hat{y}(k + j|k)$, $j = N_1, \dots, N_2$. Pri tem pri nekaterih metodah uporabimo **strukturiranje** (ang. *structuring*) prihodnjih regulirnih signalov, kar bomo opisali kasneje.
- Za vodenje uporabimo samo prvi element $u(k|k)$ **optimalnega** vektorja regulirnih signalov $u(k + j|k)$, $j = 0, \dots, N_u - 1$.

V naslednjem diskretnem trenutku imamo na voljo nov merjeni izhod procesa in celotni postopek, opisan v štirih točkah, se ponovi. Ta princip se imenuje **strategija pomicnega horizonta** (ang. *receding horizon strategy*).

Osnovno shemo zaprtozančnega sistema vodenja si lahko ogledamo na sliki 17.2. V idealnem primeru, ko ni motenj in ko se model idealno ujema s procesom, rezultira shema v odprtozančno optimalno vodenje (v praksi imamo vedno odstopanja in je povratna zanka vedno delajoča).



Slika 17.2. Shema prediktivnega vodenja.

Iz opisanega koncepta lahko zaključimo naslednje pomembne elemente prediktivnega vodenja:

- napoved oz. predikcija na osnovi modela procesa,
- generiranje modelno-referenčne trajektorije,
- strukturiranje regulacijske akcije,
- algoritmizirano računanje najboljšega regulirnega scenarija.

Za vsakega od nastetih elementov lahko najdemos različne rešitve in po tem se posamezne metode prediktivnega vodenja razlikujejo med seboj. V nadaljevanju bomo predstavili nekaj najpogosteje uporabljenih različic. Še enkrat pa naj poudarimo, da vse metode prediktivnega vodenja temeljijo na istem osnovnem konceptu, ki je opisan na začetku tega podpoglavlja. Vse klasične metode prediktivnega vodenja temeljijo na linearnih modelih procesov.

17.2 Predikcijski modeli

Konceptualno lahko model vsakega realnega procesa razdelimo za dva aditivna podmodela:

- **model procesa**, ki opisuje relacije med merjenimi vhodi in merjenimi izhodi procesa,
- **model motenj**, s katerimi skušamo do določene mere opisati tisti prispevek merjenega izhoda procesa, ki ni zajet v modelu procesa. Pri aplikacijah na realnih napravah model motenj večinoma predstavlja vpliv nemerljivih vhodov v proces, motnje iz okolja, šum meritev, pa tudi napake, ki izvirajo iz nepopolnega modeliranja in povzročajo odstopanje med modelom in procesom.

Večina metod prediktivnega vodenja uporablja za napoved izhodnega signala $\hat{y}(k+j|k)$ eksplicitno le en model, ki zajema tako model procesa kot tudi model motenj. Zato bomo v nadaljevanju, zaradi enostavnosti, obravnavali le ta primer.

Opomba: do sedaj smo večkrat uporabili zapis $\hat{y}(k+j|k)$, ki ga interpretiramo kot j -koračno napoved izhodnega signala na podlagi informacije, ki jo imamo v trenutku k . V nadaljevanju bomo eksplicitno določilo " $|k$ " izpuščali (razen seveda v redkih primerih, kjer bomo želeli posebej poudariti zgornji pomen). Torej: $\hat{y}(k+j|k)$ in podobni izrazi bodo vedno pomenili napoved na osnovi informacije dostopne do trenutka k .

V strategiji prediktivnega vodenja lahko načelno uporabimo vsak model, s katerim je možno izračunati napoved izhodnega signala več korakov v prihodnost: modeli v obliki prenosne funkcije, zapisani v prostoru stanj ali neparametrični konvolucijski modeli; linearni ali nelinearni

modeli; modeli v obliki nevronskih mrež, asociativnih pomnilnikov ali mehki modeli; modeli s koncentriranimi parametri ali modeli s porazdeljenimi parametri; modeli s časovno nespremenljivimi parametri, modeli s spremenljivim ojačenjem ali popolnoma spremenljivimi parametri. Metodologija prediktivnega vodenja je tako široko odprta tudi za prihodnje razširitev in raziskovanje. V nadaljevanju bomo predstavili linearne modele, ki so v obstoječih metodah najpogosteje uporabljeni.

17.2.1 Model impulznega odziva

Model v obliki impulznega odziva procesa (imenovan tudi utežna funkcija, konvolucijski model ali FIR (*ang. Finite Impulse Response*) zapišemo v obliki

$$y(k) = \sum_{i=1}^{\infty} h_i u(k-i) \approx \sum_{i=1}^n h_i u(k-i) = H(q^{-1})u(k), \quad (17.1)$$

kjer so h_i vrednosti impluznega odziva procesa v posameznih časovnih trenutkih in

$$H(q^{-1}) = h_1 q^{-1} + h_2 q^{-2} + \dots + h_n q^{-n},$$

q^{-1} pa je operator časovnega zamika $q^{-1}u(k) = u(k-1)$.

Prednosti modela impulznega odziva so naslednje:

- ne potrebujemo skoraj nobenega predhodnega znanja o procesu (npr. reda procesa), primeren je tudi za procese z zahtevnejšo dinamiko, dobimo ga z neparametrično identifikacijo,
- enostaven izračun j -koračne napovedi izhodnega signala:

$$\hat{y}(k+j) = \sum_{i=1}^n h_i u(k+j-i) = H(q^{-1})u(k+j), \quad (17.2)$$

pri čemer upoštevamo, da so $u(k+j)$ poznani. Ker izračun predikcije ne zahteva rekurzije, je enostaven in manj občutljiv za numerične napake.

Slabosti modela impulznega odziva:

- neuporaben za nestabilne procese,
- zahteva veliko število parametrov, ki jih moramo poznati ali oceniti ($n = 30 \div 50$).

Ta model je osnova pri metodah prediktivnega vodenja IDCIM, EPSAC in tudi GPC ter UPC.

Primer 17.2.1. Primer j -koračne napovedi.

Predpostavimo, da je impulzni odziv zapisan s polinomom $H(q^{-1})$, kjer je $n = 4$. Zapišimo napoved izhodnega signala za $j = 2$:

$$\hat{y}(k+2) = h_1 u(k+1) + h_2 u(k) + h_3 u(k-1) + h_4 u(k-2). \quad (17.3)$$

Vidimo lahko, da moramo za izračun napovedi poznati tako vhodni signal v preteklosti, kot tudi vhodni signal v prihodnosti.

17.2.2 Model v obliki odziva na stopnico

Model v obliki odziva procesa na stopnico, ki je uporabljen pri metodi DMC, imenujemo tudi model FSR (*ang. Finite Step Response*). Zapišemo ga kot:

$$y(k) = y_0 + \sum_{i=1}^{\infty} g_i \Delta u(k-i) \approx y_0 + \sum_{i=1}^n g_i \Delta u(k-i) = y_0 + G(q^{-1})(1-q^{-1})u(k), \quad (17.4)$$

kjer so g_i vrednosti odziva na enotino stopnico procesa, $\Delta u(k)$ je sprememba vhodnega signala $\Delta u(k) = u(k) - u(k-1)$, n je red razvoja vrste in je dovolj velik, da so od n naprej vsi g_i enaki med seboj, y_0 je kumulativni prispevek vpliva vseh sprememb Δu pred n vzorci. Vrednost y_0 običajno postavimo na 0, ker jo lahko vzamemo kot delovno točko.

Napoved za j korakov v prihodnost izračunamo kot:

$$\hat{y}(k+j) = \sum_{i=1}^n g_i \Delta u(k+j-i) = G(q^{-1})(1-q^{-1})u(k+j). \quad (17.5)$$

Model v obliki odziva na stopnico ima enake prednosti in slabosti kot model v obliki impulznega odziva.

17.2.3 Model v obliki prenosne funkcije

Model v obliki prenosne funkcije je uporabljen pri metodah EPSAC, GPC, UPC, MUSMAR. Predstavimo ga lahko z diferenčno enačbo

$$y(k) + a_1 y(k-1) + \dots + a_{n_a} y(k-n_a) = b_1 u(k-d-1) + \dots + b_{n_b} u(k-d-n_b) \quad (17.6)$$

ali v obliki

$$y(k) = \frac{q^{-d} B(q^{-1})}{A(q^{-1})} u(k), \quad (17.7)$$

kjer je d mrtvi čas procesa ($d \geq 0$), polinoma $A(q^{-1})$ in $B(q^{-1})$ pa sta:

$$\begin{aligned} A(q^{-1}) &= 1 + a_1 q^{-1} + \dots + a_{n_a} q^{-n_a} \\ B(q^{-1}) &= b_1 + q^{-1} + \dots + b_{n_b} q^{-n_b}. \end{aligned} \quad (17.8)$$

Pri tem smo upoštevali, da je parameter $b_0 = 0$, kar pri realnih procesih praktično vedno velja. V primeru, če bi bil $b_0 \neq 0$ bi se izhod procesa skočno odzval na spremembo vhodnega signala, kar pa z vidika končnih moči ni možno. Ta predpostavka izvira tudi iz dejstva, da pri transformaciji zveznih procesov v diskretno domeno zaradi uporabe vzorcevalnikov vedno dobimo vsaj eno zakasnitev. Neparametrična modela FIR in FSR lahko obravnavamo kot podmnožico modela v obliki prenosne funkcije. Za model FIR pomeni to, da je $A(q^{-1}) = 1$ in da so koeficienti polinoma $B(q^{-1})$ elementi impulznega odziva $b_j = h_j$. Za model FSR pa velja $A(q^{-1}) = 1$, koeficienti polinoma $B(q^{-1})$ pa so $b_1 = g_1$, $b_i = g_i - g_{i-1}$, $\forall i \geq 2$.

Prednosti modela v obliki prenosne funkcije so naslednje:

- minimalno število parametrov, ki jih potrebujemo za opis linearnega procesa,
- model je uporaben za stabilne in nestabilne procese.

Slabosti modela v obliki prenosne funkcije:

- nujno poznavanje reda števca in imenovalca oz. ustrezeno ocenjevanje ali predpostavke,
- predikcija izhodnega signala procesa opisanega z modelom prenosne funkcije je bolj kompleksna.

Opozorimo naj, da smo pri tem modelu eksplizitno izrazili morebitni mrtvi čas procesa T_d , ($T_d = dT_0$). Tako eksplizitno obravnavo bomo uporabljali praktično v celotnem poglavju. V vseh ostalih poglavjih mrtvi čas izpuščamo zaradi kompaktnosti in preglednosti matematičnih izrazov.

Če v skladu s principom ekvivalence (*ang. certainty equivalence principle*) predpostavimo, da smo parametre modela idealno ocenili, torej, da točno poznamo parametre procesa ($\hat{B} = B$, $\hat{A} = A$, $\hat{d} = d$, zato zapišemo kar B , A , d), potem zapišemo predikcijo izhodnega signala kot:

$$\begin{aligned}\hat{y}(k+j) &= \frac{q^{-d}B(q^{-1})}{A(q^{-1})}u(k+j) \\ &= q^{-d}B(q^{-1})u(k+j) - q(A(q^{-1}) - 1)\hat{y}(k+j-1).\end{aligned}\quad (17.9)$$

Pri tem je $q(A(q^{-1}) - 1) = a_1 + a_2q^{-1} + \dots + a_{n_a}q^{-(n_a-1)}$, ker je $A(q^{-1})$ monični polinom. Tako lahko vse $\hat{y}(k+j)$ za $j \geq 1$ izračunamo rekurzivno po enačbi 17.10 s pričetkom pri $j = 1$:

$$\hat{y}(k+1) = q^{-d}B(q^{-1})u(k+1) - q(A(q^{-1}) - 1)\hat{y}(k), \quad (17.10)$$

pri čemer v napovedi vhod nastopa dejansko z $u(k)$ in njegovimi zakasnittvami. Ker na splošno velja, da je $b_0 = 0$, člen pri $u(k+1)$ odpade. Prediktor v enačbi 17.21 deluje paralelno procesu.

Ta prediktor zaradi rekurzivnega računanja ni uporaben, ker se v praksi ne moremo izogniti razlikam med modelom in procesom. Že najmanjša odstopanja ocenjenih parametrov modela od dejanskih parametrov procesa ali motnje na izhodu procesa povzročijo napačno predikcijo. Možna izboljšava je, da predikcijo $\hat{y}(k)$ v enačbi 17.21 zamenjamo z izmerjenim izhodom procesa $y(k)$. Pri tem uvedemo Diofantsko enačbo:

$$\frac{1}{A(q^{-1})} = E_j(q^{-1}) + q^{-j} \frac{F_j(q^{-1})}{A(q^{-1})}, \quad (17.11)$$

kjer sta polinoma $E_j(q^{-1})$ stopnje $(j-1)$ in $F_j(q^{-1})$ stopnje (n_a-1) , ki sta enolično določena z j in $A(q^{-1})$. Rešitev enačbe poiščemo z dolgim deljenjem polinomov ali rekurzivnim algoritmom, ki bo opisan kasneje. Enačbo 17.11 vstavimo v enačbo 17.7 in na levi strani zamenjamo izhod modela $y(k)$ z oceno izhodnega signala $\hat{y}(k)$. Dobimo naslednjo enačbo:

$$\hat{y}(k) = q^{-d}B(q^{-1})E(q^{-1})u(k) + q^{-d} \frac{B(q^{-1})}{A(q^{-1})}u(k)q^{-j}F_j(q^{-1}), \quad (17.12)$$

in naprej

$$\hat{y}(k) = q^{-d}B(q^{-1})E(q^{-1})u(k) + q^{-j}F_j(q^{-1})y(k). \quad (17.13)$$

Enačbo 17.13 z leve in desne pomnožimo s q^j in dobimo naslednji zapis j -koračnega prediktorja:

$$\hat{y}(k+j) = q^{-d}E_j(q^{-1})B(q^{-1})u(k+j) + F_j(q^{-1})y(k). \quad (17.14)$$

17.2.4 Model z zapisom v prostoru stanj

Model z zapisom v prostoru stanj je uporabljen pri metodi PFC. Opišemo ga z naslednjima enačbama:

$$x(k+1) = \underline{A} \underline{x}(k) + \underline{b} u(k), \quad (17.15)$$

$$y(k) = \underline{c}^T \underline{x}(k), \quad (17.16)$$

kjer so \underline{x} stanja procesa, \underline{A} sistemski matrični vektor, \underline{b} vhodni vektor in \underline{c}^T izhodni vektor. Ker pri realnih procesih v glavnem ne zasledimo direktnih povezav med vhodom in izhodom procesa, jih v zgornjem zapisu nismo upoštevali.

Predikcijo izhodnega signala izračunamo z rekurzijo stanj sistema po enačbi 17.16 in uporabo enačbe 17.16. Dobimo naslednji zapis za j -koračni prediktor:

$$\hat{y}(k+j) = \underline{c}^T \underline{x}(k+j) = \underline{c}^T \left(\underline{A}^j \underline{x}(k) + \sum_{i=1}^j \underline{A}^{i-1} \underline{b} u(k+j-i) \right). \quad (17.17)$$

Model v tej obliki je prilagojen za analizo notranje strukture procesa. Enak koncept je uporabljen tudi za multivariabilne sisteme. Računska zahtevnost je večja in če stanja niso direktno merljiva, moramo uporabiti opazovalnike stanj.

17.2.5 Ostali modeli

Koncept prediktivnega vodenja poleg teh osnovnih linearnih modelov, dopušča tudi uporabo drugih modelov, posebno nelinearnih modelov, ki so linearni v prihodnjih regulirnih signalih $u(k+j)$, $j > 0$.

17.3 Napoved izhoda na osnovi modela v obliki prenosne funkcije

V tem poglavju si bomo še enkrat natančneje pogledali izračun napovedi izhoda modela na osnovi modela v obliki prenosne funkcije. Enačbo, ki zapisuje to napoved smo imenovali tudi prediktor.

Predpostavimo, da imamo model procesa v obliki prenosne funkcije Z-prostoru

$$G_p(z) = \frac{q^{-d}B(z)}{A(z)}. \quad (17.18)$$

Predikcijo izhodnega signala izračunamo v časovnem prostoru. Zato namesto operatorja z^{-1} uporabljamo časovni zamik q^{-1} in k za definiranje določenega časovnega trenutka. Dobimo naslednji zapis

$$\hat{y}(k) = \frac{q^{-d}B(q^{-1})}{A(q^{-1})} u(k). \quad (17.19)$$

Če želimo izračunati predikcijo izhodnega signala za j korakov v prihodnosti, potem moramo v argumentu namesto k zapisati $k+j$. Na ta način dobimo

$$\hat{y}(k+j) = \frac{q^{-d}B(q^{-1})}{A(q^{-1})} u(k+j). \quad (17.20)$$

Polinom $A(q^{-1}) = 1 + a_1 q^{-1} + a_2 q^{-2} + \dots + a_{n_a} q^{-(n_a)}$ je moničen in reda n_a .

Če s polinomom $A(q^{-1})$ pomožimo celotno enačbo 17.20 in prenesemo predikcije izhoda, ki so manjše od j na desno stran, dobimo predikcijo izhodnega signala pri horizontu j kot

$$\hat{y}(k+j) = q^{-d}B(q^{-1})u(k+j) - q(A(q^{-1}) - 1)\hat{y}(k+j-1),$$

pri čemer je sedaj $q(A(q^{-1}) - 1) = a_1 + a_2q^{-1} + \dots + a_{n_a}q^{-(n_a-1)}$. Tako lahko vse $\hat{y}(k+j)$ za $j \geq 1$ izračunamo rekurzivno po enačbi 17.20 s pričetkom pri $j = 1$:

$$\hat{y}(k+1) = q^{-d}B(q^{-1})u(k+1) - q(A(q^{-1}) - 1)\hat{y}(k), \quad (17.21)$$

pri čemer vhodni signal nastopa z $u(k)$ in zakasnitvami, zaradi vrednosti člena $b_0 = 0$. Prediktor v enačbi 17.20 je struktурno gledano paralelen procesu. Napovedi izhodnega signala se povratnozančno uporabljajo za izračun naslednjih napovedi. Ta struktura preko povratne zanke akumulira napake, ki se pojavi zaradi odstopanja ocenjenih parametrov modela od dejanskih parametrov procesa ali motnje na izhodu procesa. Zaradi tega je dejansko ta struktura neuporabna.

Prediktor izboljšamo tako, da $\hat{y}(k)$ nadomestimo z izmerjenim izhodom procesa $y(k)$. Izračuna predikcije se lahko lotimo tako, da v enačbi 17.19 zapišemo člen $\frac{1}{A(q^{-1})}$ kot vsoto polinoma in ostanka pri deljenju, ki je ulomljena racionalna funkcija

$$\frac{1}{A(q^{-1})} = E_j(q^{-1}) + q^{-j}\frac{F_j(q^{-1})}{A(q^{-1})}. \quad (17.22)$$

Polinoma E_j in F_j dobimo tako, da rešimo Diofantsko enačbo

$$E_j(q^{-1})A(q^{-1}) + q^{-j}F_j(q^{-1}) = 1. \quad (17.23)$$

Poskušajmo ugotoviti kakšna morata biti reda the dveh polinomov, da dobimo določen sistem, ki nam da enoumno rešitev. Če predpostavimo, da je red polinoma E_j enak n_e in red polinoma F_j enak n_f , potem imamo na levi strani polinom, ki je reda $\max((n_e + n_a), (j + n_f))$. To pomeni, da imamo $\max((n_e + n_a), (j + n_f)) + 1$ enačb na osnovi katerih moramo izračunati neznane parametre za polinoma E_j in F_j , ki jih je skupaj $n_e + 1 + n_f + 1$.

Reda n_e in n_f izberemo tako, da velja $n_e + n_a = j + n_f$. Potem velja, da imamo $n_e + n_a + 1$ enačb, na osnovi katerih moramo določiti $n_e + 1 + n_f + 1$ parametrov. Za rešitev mora biti število enačb enako številu parametrov, kar pomeni, da je $n_f = n_a - 1$ in $n_e = j - 1$.

To pomeni, da je polinom $E_j(q^{-1})$ stopnje $(j - 1)$, polinom $F_j(q^{-1})$ pa stopnje $(n_a - 1)$. Stopnji sta določeni s korakom napovedi j in redom procesa, ki je določen s polinomom $A(q^{-1})$.

Če sedaj izraz iz enačbe 17.22 uporabimo v enačbi 17.19, dobimo

$$\hat{y}(k+j) = q^{-d}E_j(q^{-1})B(q^{-1})u(k+j) + q^{-j}F_j(q^{-1})\frac{q^{-d}B(q^{-1})}{A(q^{-1})}u(k+j), \quad (17.24)$$

in če predpostavimo, da lahko merimo izhode procesa in zamenjamo $\hat{y}(k)$ z $y(k)$, dobimo

$$\hat{y}(k+j) = q^{-d}E_j(q^{-1})B(q^{-1})u(k+j) + F_j(q^{-1})y(k). \quad (17.25)$$

Ugotovimo, da za vsak horizont predikcije j lahko izračunamo enolična polinoma E_j in F_j , ki določata predikcijo izhodnega signala pri tem horizontu. Polinoma lahko določimo z dolgim deljenjem polinomov ali rekurzivnim algoritmom. V nadaljevanju si bomo ogledali izračun napovedi z izenačevanjem koeficientov polinomov.

17.3.1 Primer izračuna napovedi izhodnega signala

Predpostavimo, da imamo model procesa, ki ga lahko zapišemo na naslednji način, s prenosno funkcijo v Z-prostoru

$$G_p(z) = \frac{Y(z)}{U(z)} = \frac{0.2z^{-1} - 0.4z^{-2}}{(1 - 0.5z^{-1})(1 - 0.8z^{-1})}, \quad (17.26)$$

kjer je števec $B(z^{-1})$ reda $n_b = 2$, imenovalec $A(z^{-1})$ pa je ravno tako reda $n_a = 2$. Zaradi preglednosti smo izbrali primer, kjer je čista zakasnitev d enaka nič.

Zapišimo Diofantsko enačbo za horizont j , kjer je red E_j enak $n_e = j - 1$ in red F_j enak $n_f = n_a - 1$. Polinoma E_j in F_j dobimo tako, da rešimo Diofantsko enačbo. Za naš primer dobimo splošno rešitev:

$$(e_0 + e_1q^{-1} + \dots + e_{j-1}q^{j-1}) (1 + a_1q^{-1} + a_2q^{-2}) + q^{-j} (f_0 + f_1q^{-1}) = 1, \quad (17.27)$$

kjer sta $a_1 = -1.3$ in $a_2 = 0.4$. Če izvedemo množenje in seštejemo člene z enakimi potencami pri $j = 1$, dobimo

$$e_0 + (e_0a_1 + f_0)q^{-1} + (e_0a_2 + f_1)q^{-2} = 1. \quad (17.28)$$

Z izenačevanjem koeficientov polinoma na levi in desni strani dobimo:

$$e_0 = 1, f_0 = -a_1 = 1.3, f_1 = -a_2 = -0.4.$$

To pomeni, da je polinom

$$E_1(q^{-1}) = 1$$

in

$$F_1(q^{-1}) = 1.3 - 0.4q^{-1}.$$

V primeru horizonta $j = 2$, dobimo

$$e_0 + (e_0a_1 + e_1)q^{-1} + (e_0a_2 + e_1a_1 + f_0)q^{-2} + (e_1a_2 + f_1)q^{-3} = 1. \quad (17.29)$$

Po izenačevanju koeficientov polinoma dobimo:

$$e_0 = 1, e_1 = -e_0a_1 = 1.3, f_0 = -e_0a_2 - e_1a_1 = 1.29, f_1 = -e_1a_2 = -0.52.$$

Polinoma sta enaka

$$E_2(q^{-1}) = 1 + 1.3q^{-1}$$

in

$$F_2(q^{-1}) = 1.29 - 0.52q^{-1}.$$

V primeru horizonta $j = 3$ pa dobimo

$$E_3(q^{-1}) = 1 + 1.3q^{-1} + 1.29q^{-2}$$

in

$$F_3(q^{-1}) = 1.157 - 0.516q^{-1}.$$

Ob upoštevanju enačbe 17.25 dobimo naslednje predikcije za izhod procesa

$$\begin{aligned}\hat{y}(k+1) &= E_1(q^{-1})B(q^{-1})u(k+1) + F_1(q^{-1})y(k), \\ \hat{y}(k+2) &= E_2(q^{-1})B(q^{-1})u(k+2) + F_2(q^{-1})y(k), \\ \hat{y}(k+3) &= E_3(q^{-1})B(q^{-1})u(k+3) + F_3(q^{-1})y(k).\end{aligned}\quad (17.30)$$

Če zapišemo vrednosti za naš primer, dobimo

$$\begin{aligned}\hat{y}(k+1) &= (0.2q^{-1} - 0.4q^{-2}) u(k+1) + (1.3 - 0.4q^{-1}) y(k), \\ \hat{y}(k+2) &= (0.2q^{-1} - 0.14q^{-2} - 0.52q^{-3}) u(k+2) + (1.29 - 0.52q^{-1}) y(k), \\ \hat{y}(k+3) &= (0.2q^{-1} - 0.14q^{-2} - 0.26q^{-3} - 0.516q^{-4}) u(k+3) + (1.157 - 0.516q^{-1}) y(k),\end{aligned}$$

ali drugače

$$\begin{aligned}\hat{y}(k+1) &= 0.2u(k) - 0.4u(k-1) + 1.3y(k) - 0.4y(k-1), \\ \hat{y}(k+2) &= 0.2u(k+1) - 0.14u(k) - 0.52u(k-1) + 1.29y(k) - 0.52y(k-1), \\ \hat{y}(k+3) &= 0.2u(k+2) - 0.14u(k+1) - 0.26u(k) - 0.516u(k-1) + 1.157y(k) - 0.516y(k-1).\end{aligned}$$

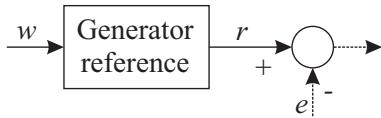
Če napoved zapišemo še v matrični obliki, kjer ločimo na desno strani prihodnje vrednosti vhodnega signala in pretekle vrednosti, dobimo naslednji zapis strukturiranega prediktorja

$$\begin{bmatrix} \hat{y}(k+1) \\ \hat{y}(k+2) \\ \hat{y}(k+3) \end{bmatrix} = \begin{bmatrix} 0.2 & 0 & 0 \\ -0.14 & 0.2 & 0 \\ -0.26 & -0.14 & 0.2 \end{bmatrix} \begin{bmatrix} u(k) \\ u(k+1) \\ u(k+2) \end{bmatrix} + \begin{bmatrix} -0.4 \\ -0.52 \\ -0.516 \end{bmatrix} u(k-1) + \begin{bmatrix} 1.3 & -0.4 \\ 1.29 & -0.52 \\ 1.157 & -0.516 \end{bmatrix} \begin{bmatrix} y(k) \\ y(k-1) \end{bmatrix}.$$

17.4 Modelno-referenčna trajektorija

Vse metode prediktivnega vodenja predpostavlja poznavanje napovedi referenčnega signala v prihodnosti (*ang. setpoint trajectory*). Pri tem je referenčni signal včasih res znan vnaprej, v ostalih primerih pa ga moramo napovedati. Običajno poleg splošnih zahtev vodenja kot so stabilnost in izregulacija pogreška v ustaljenem stanju, želimo predpisati tudi na kakšen način naj se izhod procesa približuje referenčnemu signalu. Zato določimo tako imenovano modelno-referenčno trajektorijo (*ang. model-reference trajectory*), $r(k+j)$, $j = 1, \dots, N_2$, ki predstavlja internou referenco. Z modelno referenčno trajektorijo tako predpišemo željeno zaprtozančno obnašanje.

Princip prikazuje slika 17.3.



Slika 17.3. Modelno-referenčna trajektorija.

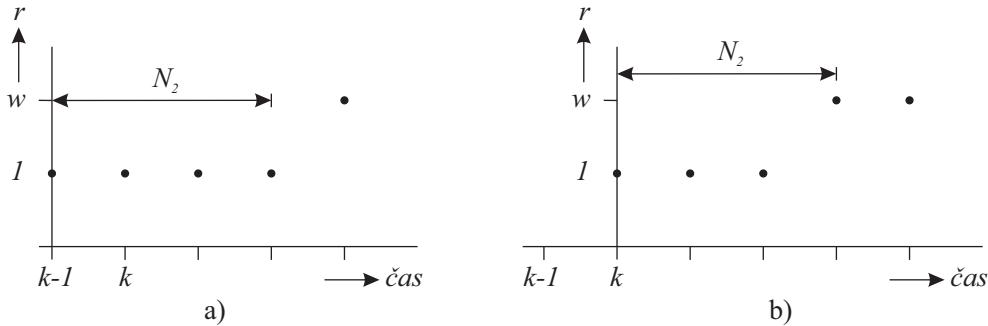
Modelno-referenčna trajektorija je v splošnem lahko poljubna in ni omejena na postopek generiranja preko referenčnega modela.

Obravnavajmo sedaj različni situaciji glede poznavanja ali ne poznavanja vnaprejšnjega poteka referenčnega signala. Zaradi enostavnosti privzemimo, da proces nima mrtvega časa ($d = 0$) in, da je modelno-referenčna trajektorija enaka referenčnemu signalu)

- Referenčni signal $w(k+j)$ je znan vnaprej za vse $j = 1, \dots, N_2$. V primeru robotskih sistemov in šaržnih procesov, kjer je zahtevano sledilno delovanje, pogosto naletimo na tak primer. Zaradi lastnosti prediktivnih metod, da stalno opazujejo razmere v prihodnosti (*ang. looking-ahead feature*), bo prediktivni regulator povzročil regulacijsko akcijo vnaprej (še pred dejansko

spremembo referenčnega signala) in na ta način kompenziral mrtvi čas in velike časovne zasnivne procesa.

Sprememba referenčnega signala ob času $k + N_1$ je znana že v trenutku k in prediktivni regulator se na to spremembo odzove z izračunom ustreznega regulirnega signala $u(k)$.



Slika 17.4. Referenčni signal je znan vnaprej.

Slika 17.4a prikazuje situacijo v trenutku $k - 1$. Sprememba referenčnega signala ob $k + N_2$ je izven predikcijskega horizonta in zato ni upoštevana v predikciji poteka oblikovane referenčne trajektorije:

$$\underline{r}(k+j) = [1, 1, \dots, 1]^T. \quad (17.32)$$

Slika 17.4b prikazuje situacijo v trenutku k . Sprememba referenčnega signala je sedaj znotraj horizonta N_2 in vektor \underline{r} tako postane:

$$\underline{r}(k+j) = [1, 1, \dots, w(k+N_2)]^T. \quad (17.33)$$

Izbira modelne referenčne trajektorije preko vektorja $\underline{r}(k+j)$ direktno vpliva na izračun $u(k)$, kar ob neprimerni izbiri lahko vodi v fazno-neminimalen zaprtozančni sistem (tudi, če je proces fazno minimalen). Temu se lahko izognemo na dva načina: i) izberemo $N_u = 1$, ii) izberemo $N_2 = N_u$.

- Referenčni signal $w(k+j)$ **ni znan vnaprej**. Kot najboljšo možno napoved referenčnega signala vzamemo njegovo trenutno vrednost: $w(k+j) = w(k)$ in oblikovana referenčna trajektorija tako postane:

$$\underline{r}(k+j) = [w(k), w(k), \dots, w(k)]^T. \quad (17.34)$$

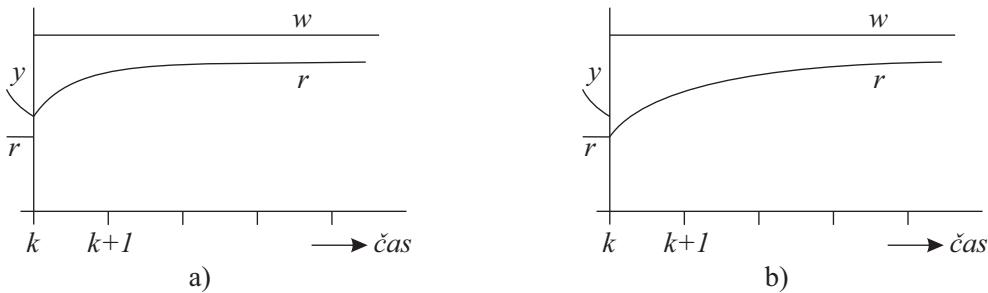
V tem primeru ni nevarnosti pojava nezaželenega fazno-neminimalnega učinka.

Neodvisno od tega, ali je referenčni signal znan vnaprej ali ne, pa način generiranja oblikovane referenčne trajektorije igra zelo pomembno vlogo. Kot smo že omenili, je trajektorija lahko poljubno zaporedje vrednosti, ki predstavljajo na primer želeno trajektorijo gibanja robota, ali na primer parabolo, ki se včasih uporablja pri sledilnih sistemih. Za večino sistemov v procesnem vodenju pa običajno uporabljamo obliko trajektorije prvega reda s konstanto a_r

$$r(k+j) = a_r r(k+j-1) + (1-a_r) w(k+j), \quad j = 1, \dots, N_2, \quad 0 \leq a_r < 1. \quad (17.35)$$

Začetno točko določimo lahko na dva načina (slika 17.5):

- uporabimo trenutno merjeno vrednost izhodnega signala: $r(k) = y(k)$



Slika 17.5. Dva načina generiranja oblikovane referenčne trajektorije.

b) uporabimo trenutno vrednost trajektorije $r(k)$

Prvi način povzroči v sistemu dodatno povratno zanko, katere vpliv na celotni zaprtozančni sistem zelo težko predvidimo, ker ne nastane kot rezultat optimizacije kriterijske funkcije. V nekaterih primerih lahko ta dodatna povratna zanka privede celo do nestabilnosti celotnega sistema.

Poleg oblike trajektorije prvega reda v literaturi pogosto najdemo še drugo možnost, ki se priporoča predvsem za spreminjačoče se referenčne signale. Modelno-referenčno trajektorijo v tem primeru lahko zapišemo kot

$$w(k+j) - r(k+j) = a_r^j (w(k) - y(k)) . \quad (17.36)$$

V obeh primerih je parameter referenčnega modela a_r . Ta parameter določa hitrost odziva zaprtozančnega sistema in ga izbiramo na intervalu ($0 \leq a_r < 1$).

Izbira oblike in načina generiranja modelno-referenčne trajektorije je torej popolnoma prosta in prepuščena načrtovalcu, zavedati pa se moramo, da z njo vplivamo ne samo na sledilno delovanje zaprtozančnega sistema, pač pa tudi na regulacijsko delovanje in na robustnost.

17.5 Strukturiranje regulacijskega zakona

Napovedani izhodni signal $\hat{y}(k+j)$, $j = 1, \dots, N_2$ je odvisen od predvidenega regulirnega signala v prihodnosti $u(k+j)$, $j = 0, \dots, N_2 - 1$. Konceptualno lahko izhodni signal razdelimo na vsoto:

$$\hat{y}(k+j) = y_p(k+j) + y_v(k+j) \quad j = 1, \dots, N_2 , \quad (17.37)$$

kjer je $y_p(k+j)$ prosti odziv sistema, če je vsiljen vhodni signal v prihodnosti enak nič, $y_v(k+j)$ pa je vsiljeni odziv sistema na vhodni signal, ki je različen od nič $\Delta u(k+j) \neq 0$, $j > 0$.

Opomba: enačbe za izračun napovedi izhodnega signala večinoma vsebujejo odvisnost od spremembe vhodnega signala $\Delta u(k+j)$ in ne od absolutnih vrednosti $u(k+j)$.

Prosti odziv sistema $y_p(k+j)$ izračunamo tako, da v enačbe za napoved izhodnega signala postavimo $\Delta u(k+j) = 0$, $j > 0$. Vsiljen odziv $y_v(k+j)$ pa zapišemo v obliki:

$$y_v(k+j) = g_j \Delta u(k) + g_{j-1} \Delta u(k+1) + \dots + g_1 \Delta u(k+j-1) , \quad (17.38)$$

kjer so g_i koeficienti odziva procesa na enotino stopnico.

Definirajmo vektor prihodnjih napak kot:

$$\underline{E} = [e(k+1), e(k+2), \dots, e(k+N_2)]^T , \quad (17.39)$$

kjer je $e(k+j) = r(k+j) - \hat{y}(k+j)$. S pomočjo enačbe 17.37 in enačbe 17.38 lahko zapišemo

$$e(k+j) = r(k+j) - y_p(k+j) - g_j \Delta u(k) + g_{j-1} \Delta u(k+1) + \dots + g_1 \Delta u(k+j-1), \quad j = 1, \dots, N_2 \quad (17.40)$$

ali z zapisom v matrični obliki

$$\underline{E} = \underline{E}_0 - \underline{G} \cdot \underline{\Delta u}, \quad (17.41)$$

kjer so

$$\underline{E}_0 = [r(k+1) - y_p(k+1), \dots, r(k+N_2) - y_p(k+N_2)]^T, \quad (17.42)$$

$$\underline{G} = \begin{bmatrix} g_1 & 0 & 0 & \dots & 0 \\ g_2 & g_1 & 0 & \dots & 0 \\ g_3 & g_2 & g_1 & \dots & 0 \\ \vdots & \vdots & \dots & \ddots & 0 \\ g_{N_2} & g_{N_2-1} & g_{N_2-2} & \dots & g_{N_1} \end{bmatrix} \quad (17.43)$$

in

$$\underline{\Delta u}(k) = [\Delta u(k), \Delta u(k+1), \dots, \Delta u(k+N_2-1)]^T. \quad (17.44)$$

Kakšenkoli kriterij izberemo za minimizacijo vektorja prihodnjih napak \underline{E} (kriterije za optimizacijo predstavljamo v naslednjem podoglavlju), bo ta vedno funkcija N_2 neodvisnih spremenljivk (elementi vektorja $\Delta u(k)$). Za vrednosti predikcijskega horizonta $N_2 = 10 \div 30$ postane očitno, da je optimizacija računsko zahtevna in časovno potratna.

To je eden od glavnih razlogov za strukturiranje prihodnjega regulirnega vektorja $\Delta u(k)$. Strukturiranje pomeni redukcijo števila prostostnih stopenj z apriori določanjem nekaterih relacij med prihodnjimi vrednostmi regulirnega signala (elementi vektorja $\Delta u(k)$).

V literaturi lahko najdemo drug, mogoče še bolj fundamentalen razlog za uvedbo strukturiranja, ki ga predlagajo raziskovalci s področja robustnosti (De Keyser, [63]). Strukturiranje ima zelo ugoden vpliv na robustnost in na splošne učinke vodenja s prediktivnimi metodami. Če ne uporabimo strukturiranja (kar pomeni, da vse prihodnje vrednosti regulirnega vektorja $\Delta u(k)$ lahko zavzamejo poljubne vrednosti), postane regulirni signal bogat z visokimi frekvencami, ki so nezaželjene pri realnih problemih vodenja. Ta pojav lahko opazimo zlasti pri netočnih modelih procesov. V skrajnem primeru dobimo celo nestabilno zaprto zanko. V nadaljevanju bomo podali nekaj najpogosteje uporabljenih tehnik strukturiranja regulirnega signala.

17.5.1 Regulirni horizont

Pri konstantnem referenčnem signalu je, po prehodnem pojavu, konstanten tudi (velja za proporcionalne procese) regulirni signal. Zato je smiseln vpeljati **regulirni horizont** N_u ($N_u < N_2$) od katerega naprej bo regulirni signal ostal konstanten: $\Delta u(k+j) = 0$ za $j \geq N_u$. To daje regulirni vektor dimenzije N_u namesto N_2 :

$$\underline{\Delta u}(k) = [\Delta u(k), \Delta u(k+1), \dots, \Delta u(k+N_u-1)]^T. \quad (17.45)$$

Poudarimo naj, da dejanski regulirni signal $u(k+j)$ ne bo konstanten, kar je rezultat tako imenovanega principa pomicnega horizonta (*ang. receding horizon*). S stališča računske zahtevnosti povzroči omenjena tehnika izpustitev zadnjih $N_2 - N_u$ stolpcov v matriki odzivov na stopnico \underline{G} v enačbi 17.43. S tem dosežemo zmanjšanje dimenzije, ker je regulirni horizont N_u v splošnem veliko manjši od predikcijskega horizonta N_2 . Tipične vrednosti so: $N_2 = 10 \div 30$ in $N_u = 1 \div 5$.

Skrajni primer je $N_u = 1$, ko računamo samo en element vektorja $\underline{\Delta u}(k)$, to je $u(k)$. Veliko simulacijskih študij in tudi aplikacij poroča o presenetljivih zaključkih, da je za veliko število

primerov vodenja ta strategija, kljub limitni skalarni obliki vektorja $\Delta \underline{u}(k)$, zelo primerna in daje dobre rezultate. Celo v primeru spreminjajočega se referenčnega signala. Ta princip je uporabljen pri metodah DMC, EPSAC, GPC.

17.5.2 Princip grupiranja

Pri tem principu ostane regulirni horizont N_u enake dimenzije kot horizont N_2 , pri tem pa razdelimo celoten predikcijski horizont N_2 na nekaj segmentov in zahtevamo, da se regulirni signal znotraj vsakega segmenta ne spreminja (je odsekoma konstanten). Rezultat tega je izpuščanje ustreznih stolpcov v matriki \underline{G} , kar zmanjšuje računsko kompleksnost problema. Za način, kako najbolje izbrati segmente, pa še vedno ni splošno veljavnih napotkov.

17.5.3 Razvoj po baznih funkcijah

Vedno, ko govorimo o strukturiranju prihodnjega regulacijskega zakona, se moramo zavedati, da s tem na nek način apriorno omejimo obliko prihodnjega regulirnega vektorja. Zato moramo vedno razmisljiti, kako to vpliva na točnost samega vodenja. Skrajno enostaven primer $N_u = 1$ dostikrat ni zadovoljiva rešitev, posebno pri regulacijskih sistemih za sledilno delovanje. Naslednja možnost je, da strukturiramo prihodnji regulirni signal kot linearne kombinacije vnaprej izbranih linearne neodvisnih baznih funkcij:

$$u(k+j) = \sum_{i=1}^{N_B} u_i(k) \cdot B_i(j), \quad j = 0, \dots, N_2 - 1. \quad (17.46)$$

Izbira baznih funkcij B_i je odvisna od narave procesa (red in vrsta procesa) ter od referenčnega signala. Pogoste so polinomske bazne funkcije:

$$B_0(j) = 1, \quad B_1(j) = j, \quad B_2(j) = j^2, \dots \quad (17.47)$$

Izračun regulirnega signala se tako skrči na izračun N_B uteži μ_i , $i = 1, \dots, N_B$. Ta način je uporabljen pri metodi PFC.

V primeru problema sledenja referenci, ki ima obliko rampe, lahko pričakujemo, da bo tudi regulirni signal te oblike (velja v primeru procesov, ki imajo proporcionalni značaj). Zato regulirni signal zapišemo na naslednji način

$$u(k+j) = u(k + jT_s) = u_0(k) + u_1(k)jT_s. \quad (17.48)$$

V tem primeru je napoved izhodnega signala odvisna od vrednosti $u_0(k)$ in $u_1(k)$.

17.6 Algoritmiziran izračun regulirnega signala

17.6.1 Kriterijska funkcija

Zahteve za vodenje so običajno podane v inženirski obliki s določenim prenihajem, časom zaksnitve in časom vzpona zaprtozančnega sistema. Ker so relacije med temi specifikacijami in rezultirajočimi parametri regulatorja v splošnem zapletene in nelinearne, nikoli ne uporabljamo kriterijskih funkcij, ki bi eksplicitno temeljile na omenjenih inženirskih kriterij. Za kriterijske funkcije, ki jih želimo minimizirati, zato uporabljamo matematično bolj primerne funkcije, katerih minimum znamo poiskati analitično ali z učinkovitimi optimizacijskimi algoritmi.

Navadno želimo minimizirati odstopanje napovedanega izhoda procesa od predvidene modelno-referenčne trajektorije. Najenostavnejši primer kriterijske funkcije je:

$$J = \sum_{j=N_1}^{N_2} (r(k+j) - \hat{y}(k+j))^2 . \quad (17.49)$$

Parameter N_1 je spodnji horizont predikcije, ki označuje začetek predikcije. Skupaj s parametrom N_2 , ki označuje zgornji predikcijski horizont (krajše tudi predikcijski horizont), tvori tako imenovani **interval ujemanja** (*ang. coincidence interval*). S tem imenom poudarimo, da želimo da se napovedani izhod procesa znotraj intervala čim bolj ujema z modelno-referenčno trajektorijo. Z optimizacijo iščemo torej tak vektor prihodnjih regulirnih signalov $\Delta u(k)$, ki bo rezultiral v minimumu kriterijske funkcije znotraj intervala ujemanja. Ostali časovni trenutki nas ne zanimajo in niso zajeti v optimizaciji.

Vpliv intervala ujemanja je intuitivno jasen. Velik interval vodi k bolj umirjenemu in gladkemu vodenju. Pri procesih z mrtvim časom je očitno, da ni smiseln postaviti N_1 na manjšo vrednost, kot je mrtvi čas procesa. Podobno je pri fazno neminimalnih procesih.

Kriterijsko funkcijo v enačbi 17.49 lahko razširimo, kot to opisuje naslednja enačba:

$$J = \sum_{j=N_1}^{N_2} \alpha_j (r(k+j) - \hat{y}(k+j))^2 . \quad (17.50)$$

Vektor uteži $\underline{\alpha} = [\alpha_{N_1}, \dots, \alpha_{N_2}]^T$ omogoča dodatne možnosti vplivanja za zaprtozančno obnašanje. Na primer, z izbiro $\alpha_j = \lambda_f^{N_2-j}$ dobimo $\underline{\alpha} = [\dots, \lambda_f^2, \lambda_f, 1]^T$, kjer je λ_f faktor pozabljanja. To ustreza eksponencialnem uteževanju preko intervala ujemanja. Za $0 < \lambda_f < 1$ velja: regulacijski pogrešek daleč v prihodnosti je bolj pomemben kot v bližnji prihodnosti. Taka izbira daje bolj umirjeno in gladko regulacijsko akcijo, manjšo porabo energije in robusten zaprtozančni sistem. Za $\lambda_f > 1$ pa velja: regulacijski pogrešek v bližnji prihodnosti je bolj pomemben kot daleč naprej. Regulacijska akcija je agresivnejša.

Možen način za reševanje zgoraj opisanih problemov je tudi uvedba razširjene oblike kriterijske funkcije, ki eksplicitno vključuje tudi vrednosti regulirnega signala:

$$J = \sum_{j=N_1}^{N_2} (r(k+j) - \hat{y}(k+j))^2 + \lambda \sum_{j=0}^{N_u-1} (u(k+j))^2 , \quad (17.51)$$

kjer je λ utežni faktor ($\lambda \geq 0$). V zapisu 17.51 nastopata dve načelno nasprotujoči si zahtevi: zmanjšanje odstopanja med napovedanim izhodom procesa in modelno referenčno trajektorijo ter zmanjšanje regulacijske akcije. Utežni faktor nastopa torej kot mehanizem za doseganje kompromisa med obema zahtevama. Večja vrednost za λ daje večjo pomembnost varianci regulirne veličine v kriterijski funkciji, kar vodi v manj agresivno regulacijsko akcijo. Istočasno pa sledenje modelno referenčni trajektoriji postane manj pomembno in v splošnem rezultira v počasnejši odziv sistema. Uvedba utežnega faktorja λ v kriterijsko funkcijo pa ima dve pomanjkljivosti:

1. kljub temu, da je vpliv λ na zaprtozančni sistem jasen, je nemogoče vnaprej določiti ustrezno vrednost, ki bo zagotavljala želeno obnašanje. Ustrezne vrednosti parametra λ zato običajno določamo s poskušanjem.
2. Za procese 0-te vrste je rezultat uvedbe utežnega faktorja pogrešek v ustaljenem stanju pri konstantni referenci in konstantnih motnjah.

Iz omenjenih razlogov nekatere prediktivne metode (npr. DMC, GPC) uporabljajo v kritevni funkciji spremembe regulirnih signalov:

$$J = \sum_{j=N_1}^{N_2} (r(k+j) - \hat{y}(k+j))^2 + \lambda \sum_{j=0}^{N_u-1} (\Delta u(k+j))^2 , \quad (17.52)$$

ki na ta način v regulirni sistem implicitno vpeljejo integrirno delovanje.

17.7 Nastavitevna pravila za parametre prediktivnega vodenja

V tem podoglavlju bomo za splošno prediktivno vodenje podali groba priporočila za izbiro ustreznih parametrov. Ta priporočila (Soeterboek, [2]) se primerna za grobo nastavitev (inicjalizacijske) vrednosti parametrov, ki jih je potrebno naknadno še fino uglasiti, za vsako aplikacijo posebej. Nastavitevni parametri so: spodnji predikcijski horizont N_1 , zgornji predikcijski horizont N_2 , regulirni horizont N_u in še nekateri drugi.

17.7.1 Nastavitevna pravila za nadkritično dušene procese:

- $N_1 = d + 1$. Najboljše sledilno delovanje dobimo z minimizacijo napake predikcije čez celoten horizont predikcije.
- $N_2 = \frac{5T+T_m}{T_s}$, kjer je T dominantna časovna konstanta, T_m je mrtvi čas, T_s pa je čas vzorčenja.
- $N_u = n_a$. Ta vrednost za N_u daje kompromisno rešitev med robustnostjo in učinkom.

Če s temi grobimi nastavtvami ne dosežemo predpisanega obnašanja zaprtozančnega sistema, si skušamo pomagati z naslednjimi priporočili za fino uglaševanje parametrov glede na simptome:

- Varianca regulirnega signala je prevelika. Če je regulacijska akcija preveč živahna samo ob spremembah reference, moramo spremeniti modelno referenčno trajektorijo. Varianco regulirnega signala lahko zmanjšamo tudi z naslednjima ukrepoma: i) zmanjšamo N_u (zaprtozančni sistem postane počasnejši in bolj robusten, izločanje motenj pa je slabše); ii) povečamo utežni faktor λ .
- Prepočasna izregulacija izhodne, vhodne motnje ali motnje obremenitve (*ang. load disturbances*). Hitrejšo izregulacijo dosežemo na tri načine: i) zmanjšamo λ , ii) če je $N_u = 1$, poskušamo povečati N_u .
- Sledenje referenčnemu signalu je prepočasno. Hitrejše sledenje lahko dosežemo z naslednjimi ukrepi: i) zmanjšamo λ . ii) povečamo N_u , če je $N_u < n_a$. Vsi ti ukrepi pa zmanjšajo robustnost sistema.
- Premajhna robustnost sistema (pri večjem odstopanju modela od realnega procesa lahko postane zaprtozančni sistem nestabilen). Ker je v praksi vsaj minimalno odstopanje vedno prisotno, je to vprašanje precej aktualno. Robustnost lahko povečamo tako, da: i) pazimo da N_u ni mnogo večji od reda procesa n_a , ii) povečamo λ .

17.7.2 Nastavitevna pravila za slabo dušene procese:

Večina zgoraj naštetih priporočil še vedno velja, razlikujejo pa se naslednja:

- $N_2 = \text{int}(2\omega_s/\omega_b)$, kjer je ω_s frekvanca vzorčenja in ω_b pasovna širina procesa.

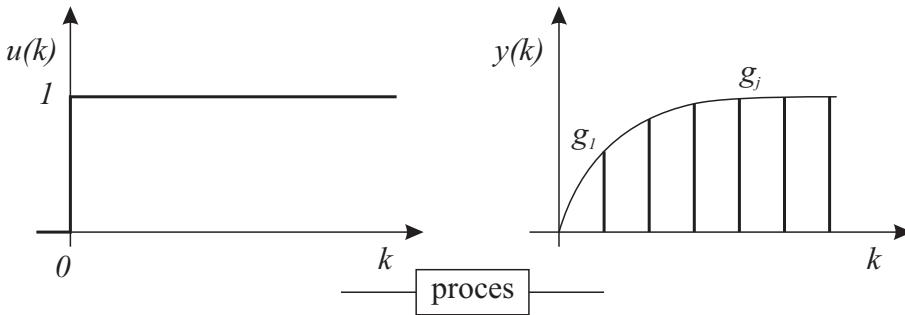
Ostala priporočila so enaka tistim za nadkritično dušene procese.

17.8 Prediktivno vodenje z dinamično matriko (DMC)

Algoritmom prediktivnega vodenja z dinamično matriko (*ang. Dynamic Matrix Control - DMC*) je predlagal Cutler, [21]. V procesni industriji je bil ta algoritmom velikokrat uspešno uporabljen. Osnovni algoritmom je narejen za multivariabilne sisteme, na tem mestu pa se bomo omejili na njegovo univariabilno izvedenko. Algoritmom DMC temelji na modelu odziva (linearnega, časovno nespremenljivega) procesa na enotino stopnico, ki ga zapišemo z diskretno konvolucijsko enačbo

$$y(k) = \sum_{i=1}^{\infty} g_i \Delta u(k-i) + n(k), \quad (17.53)$$

kjer je $y(k)$ izhodni signal procesa, $\Delta u(k)$ sprememba regulirnega signala ozziroma vhoda procesa, g_i so koeficienti, ki ustrezajo vrednostim odziva na stopnico in $n(k)$ motnja, ki deluje ob času k . Grafična ilustracija tega neparametričnega modela procesa, ki ga navadno določimo z identifikacijo, je prikazana na sliki 17.6.



Slika 17.6. Model v obliki odziva na stopnico.

Vektor prihodnjih sprememb regulirnega signala $\Delta u(k)$ računamo z minimizacijo kriterijske funkcije v enačbi 17.54 glede na vektor $\Delta u(k+j)$

$$J = \sum_{j=N_1}^{N_2} (r(k+j) - \hat{y}(k+j))^2 + \lambda \sum_{j=0}^{N_u-1} (\Delta u(k+j))^2. \quad (17.54)$$

ob predpostavki, da so za $j \geq N_u$, vse spremembe regulirnega signala enake $\Delta u(k+j) = 0$. Za lažjo obravnavo izračuna regulirnega signala razdelimo predikcijo izhodnega signala na prosti odziv sistema $y_p(k+j)$ in na vsiljeni odziv sistema $y_v(k+j)$:

$$\hat{y}(k+j) = y_p(k+j) + y_v(k+j). \quad (17.55)$$

Prosti odziv sistema je napovedano obnašanje sistema pri vhodnih signalih $\Delta u(k+j) = 0$ za $j = 1, \dots, \infty$, vsiljeni odziv pa je odziv sistema na vhodni signal $\Delta u(k+j) \neq 0$ za $j \geq 0$:

$$\hat{y}(k+j) = \sum_{i=1}^j g_i \Delta u(k+j-i) + \sum_{i=j+1}^{\infty} g_i \Delta u(k+j-i) + n(k+j). \quad (17.56)$$

Če predpostavimo, da je najboljša ocena motnje v prihodnosti $n(k+j)$ kar trenutna vrednost $n(k)$:

$$n(k+j) = n(k) = y(k) - \sum_{i=1}^{\infty} g_i \Delta u(k-i) , \quad (17.57)$$

dobimo napoved izhodnega signala:

$$\hat{y}(k+j) = \sum_{i=1}^j g_i \Delta u(k+j-i) + \sum_{i=j+1}^{\infty} g_i \Delta u(k+j-i) + y(k) - \sum_{i=1}^{\infty} g_i \Delta u(k-i) , \quad (17.58)$$

ozziroma v kompaktni obliki

$$\hat{y}(k+j) = G_j(q^{-1}) \Delta u(k+j) + p_j , \quad (17.59)$$

kjer je $G_j(q^{-1})$ polinom koeficientov odziva na stopnico

$$G_j(q^{-1}) = g_1 q^{-1} + \dots + g_j q^{-j} ,$$

p_j pa predstavlja prosti odziv sistema na stopnico:

$$\begin{aligned} p_j &= y(k) + \sum_{i=j+1}^{\infty} g_i \Delta u(k+j-i) - \sum_{i=1}^{\infty} g_i \Delta u(k-i) \\ &= y(k) + \sum_{i=1}^{\infty} (g_{j+i} - g_i) \Delta u(k-i) . \end{aligned} \quad (17.60)$$

V primeru, ko so vsi $\Delta u(k+j) = 0$ za $j = 0, \dots, N_2$, postane enačba 17.59 enaka: $\hat{y}(k+j) = p_j$, torej je p_j ocena prostega odziva procesa na enotino stopnico.

V primeru asimptotično stabilnih procesov konvergirajo koeficienti g_i proti konstantni vrednosti, zato lahko zgornjo enačbo 17.60 enostavneje zapišemo:

$$p_j = y(k) + \sum_{i=1}^N (g_{j+i} - g_i) \Delta u(k-i) , \quad (17.61)$$

kjer je N število, za katerega velja

$$g_{j+i} - g_i \cong 0 , \quad i > N, \quad j = N_1, \dots, N_2 . \quad (17.62)$$

Če proces ni asimptotično stabilen, število N ne obstaja in izrazov p_j ne moremo izračunati.

Z upoštevanjem enačbe 17.59 lahko zapišemo matrično obliko izhoda modela

$$\underline{Y} = \underline{G} \underline{\Delta u} + \underline{P} , \quad (17.63)$$

kjer je $\underline{Y} = [\hat{y}(k+N_1), \hat{y}(k+N_1+1), \dots, \hat{y}(k+N_2)]^T$ vektor napovedi izhoda modela med horizontom N_1 in N_2 , $\underline{\Delta u} = [\Delta u(k), \Delta u(k+1), \dots, \Delta u(k+N_u-1)]^T$ je vektor prihodnjih sprememb regulirnega signala, $\underline{P} = [p_{N_1}, p_{N_1+1}, \dots, p_{N_2}]^T$ vektor prostega odziva procesa v prihodnosti, \underline{G} pa je tako imenovana dinamična matrika (*ang. dynamic matrix*), ki je sestavljena iz vrednosti odziva na stopnico

$$\underline{G} = \begin{bmatrix} g_{N_1} & \dots & g_1 & 0 & \dots & \dots & 0 \\ g_{N_1+1} & \dots & g_2 & g_1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{N_2} & \dots & \dots & \dots & \dots & \dots & g_{N_2-N_u+1} \end{bmatrix} . \quad (17.64)$$

Če upoštevamo, da lahko referenčni signal v prihodnosti med horizontoma N_1 in N_2 zapišemo kot $\underline{R} = [r(k + N_1), r(k + N_1 + 1), \dots, r(k + N_2)]^T$, potem dobimo naslednji zapis kriterijske funkcije iz enačbe 17.54

$$J = (\underline{R} - \underline{Y})^T (\underline{R} - \underline{Y}) + \lambda \Delta \underline{u}^T \Delta \underline{u}. \quad (17.65)$$

Z upoštevnjem enačbe 17.63 dobimo naslednji zapis kriterijske funkcije

$$J = (\underline{R} - \underline{G} \Delta \underline{u} - \underline{P})^T (\underline{R} - \underline{G} \Delta \underline{u} - \underline{P}) + \lambda \Delta \underline{u}^T \Delta \underline{u}. \quad (17.66)$$

Če zapišemo $\underline{E}_0 = \underline{R} - \underline{P}$ in rešimo kriterijsko funkcijo z odvajanjem po vektorju $\Delta \underline{u}$, dobimo **analitični regulirni zakon**

$$\Delta \underline{u}(k) = (\underline{G}^T \underline{G} + \lambda \underline{I})^{-1} \underline{G}^T \underline{E}_0. \quad (17.67)$$

Na ta način lahko rešimo optimalni problem vodenja v primeru, ko nimamo omejitev sistemskih spremenljivk. Rešitev daje N_u prihodnjih vrednosti za regulirni signal, za vodenje pa uporabimo le prvi element $\Delta u(k)$, v naslednjem časovnem trenutku pa celotni postopek (meritev novih signalov procesa, optimizacija in regulacija) ponovimo v skladu s strategijo pomicnega horizonta (*ang. receding horizon*).

Eksistenza analitične rešitve je pomembna prednost klasičnih algoritmov prediktivnega vodenja pred ostalimi metodami, ker je računsko bistveno manj kompleksna in časovno potratna, ne zahteva velike procesorske moči, in ni tako numerično problematična kot reševanje z optimizacijskimi algoritmi. Pri reševanju optimizacijskega problema ima pomembno vlogo pogojenost matrike $(\underline{G}^T \underline{G} + \lambda \underline{I})$, ki določa kako natančna je rešitev. Slaba pogojenost te matrike daje kriterijsko funkcijo J z dolgimi dolinami in slabo izraženim globalnim minimumom, kjer se iterativni optimizacijski algoritmi slabše obnesejo kot zgornji način reševanja, kjer z uporabljenim reguliracijskim principom rešimo problem slabe pogojenosti matrike $\underline{G}^T \underline{G}$.

Za primer reševanja vodenja je zelo pomembno tudi dejstvo, da je za določen konkretni primer število potrebnih korakov za konvergenco konstantno v vsakem koraku vodenja. To dejstvo za iterativne optimizacijske metode večinoma ne velja, ker je potrebno število iteracij odvisno od trenutne vrednosti kriterijske funkcije. Pri vodenju realnih procesov moramo namreč poiskati rešitev optimizacijskega problema znotraj predpisanega intervala vzorčenja.

V primeru omejitev procesnih spremenljivk moramo optimizacijski problem reševati numerično z ustreznimi metodami kvadratičnega programiranja.

17.9 Posplošen prediktivni regulator (GPC)

Posplošen prediktivni regulator (*ang. Generalized Predictive Control - GPC*) je razvil D.W. Clarke s sodelavci (Clarke et al., [19]) kot nadgradnjo pospoljenega minimalno variančnega regulatorja GMV z namenom, da bi odpravil njegove slabosti. GPC regulator temelji na ARIMAX modelu (včasih imenovanem tudi CARIMA)

$$y(k) = q^{-d} \frac{B(q^{-1})}{A(q^{-1})} u(k) + \frac{C(q^{-1})}{\Delta A(q^{-1})} v(k). \quad (17.68)$$

Poglejmo si bistvene korake pri razvoju j -koračnega prediktorja ARIMAX. Polinoma $C(q^{-1})$ običajno ne poznamo in ga ocenimo s polinomom $T(q^{-1})$, ki predstavlja šumni filter. Enačbo 17.68 zapišemo za časovni trenutek $k + j$ in preoblikujemo v:

$$\frac{\Delta A(q^{-1})}{T(q^{-1})}y(k+j) = q^{-d}\frac{B(q^{-1})}{T(q^{-1})}\Delta u(k+j) + v(k+j). \quad (17.69)$$

Za sistematično izpeljavo napovedi izhoda procesa $\hat{y}(k+j)$ (v smislu minimalno variančnega prediktorja) rešimo Diofantsko enačbo:

$$\frac{T(q^{-1})}{\Delta A(q^{-1})} = E_j(q^{-1}) + q^{-j}\frac{F_j(q^{-1})}{\Delta A(q^{-1})} \text{ ali } 1 = \frac{\Delta A(q^{-1})}{T(q^{-1})}E_j(q^{-1}) + q^{-j}\frac{F_j(q^{-1})}{T(q^{-1})}, \quad (17.70)$$

katero rešitev sta polinoma $E_j(q^{-1})$, ki je reda $j-1$ in $F_j(q^{-1})$ reda n_a+1 . Polinoma sta enolično določena z imenovalcem procesa $A(q^{-1})$ ter trenutkom predikcije j . V enačbi 17.70 smo upoštevali:

$$\begin{aligned} T(q^{-1}) &= t_0 + t_1q^{-1} + t_2q^{-2} + \dots t_{n_t}q^{-n_t}, \\ E_j(q^{-1}) &= e_0 + e_1q^{-1} + e_2q^{-2} + \dots e_{j-1}q^{-(j-1)}, \\ F_j(q^{-1}) &= f_0 + f_1q^{-1} + f_2q^{-2} + \dots f_{n_a+1}q^{-(n_a+1)}. \end{aligned}$$

Če enačbo 17.69 pomnožimo z E_j , dobimo:

$$\frac{\Delta A(q^{-1})}{T(q^{-1})}E_j(q^{-1})y(k+j) = q^{-d}E_j(q^{-1})\frac{B(q^{-1})}{T(q^{-1})}\Delta u(k+j) + E_j(q^{-1})v(k+j) \quad (17.71)$$

in z upoštevanjem enačbe 17.70 sledi

$$\left(1 - q^{-j}\frac{F_j(q^{-1})}{T(q^{-1})}\right)y(k+j) = q^{-d}E_j(q^{-1})\frac{B(q^{-1})}{T(q^{-1})}\Delta u(k+j) + E_j(q^{-1})v(k+j). \quad (17.72)$$

S preureditvijo enačbe 17.72 dobimo minimalno variančni prediktor

$$\hat{y}(k+j) = q^{-d}E_j(q^{-1})\frac{B(q^{-1})}{T(q^{-1})}\Delta u(k+j) + \frac{F_j(q^{-1})}{T(q^{-1})}y(k) + E_j(q^{-1})v(k+j). \quad (17.73)$$

V nadaljevanju bomo neznano vrednost belega šuma $v(k+j)$ nadomestili z njegovo najboljšo možno oceno, to je srednjo vrednostjo $E\{v(k+j)\} = 0$. V enačbi 17.73 je $\hat{y}(k+j)$ funkcija znanih trenutnih in preteklih vrednosti signalov ter tudi prihodnjih regulirnih signalov, ki jih iščemo. Za razdelitev med preteklimi in prihodnjimi vrednostmi regulirnih signalov rešimo Diofantsko enačbo:

$$\frac{E_j(q^{-1})B(q^{-1})}{T(q^{-1})} = G_j(q^{-1}) + q^{-j}\frac{\Gamma_j(q^{-1})}{T(q^{-1})}. \quad (17.74)$$

Predikcija izhodnega signala postane tako enaka

$$\hat{y}(k+j) = G_j(q^{-1}) \underbrace{\Delta u(k+j-d)}_{\substack{\text{neznani} \\ \text{prihodnji}}} + \underbrace{\frac{\Gamma_j(q^{-1})}{T(q^{-1})}\Delta u(k) + \frac{F_j(q^{-1})}{T(q^{-1})}y(k)}_{\substack{\text{znani} \\ \text{trenutni in pretekli}}}, \quad (17.75)$$

kjer sta

$$G_j(q^{-1}) = g_1q^{-1} + \dots + g_{j-d}q^{-(j-d)}, \quad (17.76)$$

$$\Gamma_j(q^{-1}) = \gamma_0 + \gamma_1q^{-1} + \dots + \gamma_{n_\gamma}q^{-n_\gamma}, \quad n_\gamma = \max(n_b + d - 2, n_t - 1), \quad (17.77)$$

$G_j(q^{-1})$ predstavlja prvih $j-d$ koeficientov odziva na stopnico za sistem s prenosno funkcijo $\frac{B(q^{-1})}{\Delta A(q^{-1})}$. Zaradi oblike polinoma $B(q^{-1}) = b_1q^{-1} + \dots + b_{n_b}q^{-n_b}$, ($b_0 = 0$), je očitno, da je tudi $g_0 =$

0. Prvi člen enačbe 17.75 predstavlja vsiljen odziv (odziv sistema na vhodni signal v prihodnosti - ob predpostavki, da pred časom $k = 0$ ni bilo vzbujanja), druga dva člena pa predstavlja prosti odziv sistema (odziv sistema na vhodne signale iz preteklosti - ob predpostavki, da od trenutka $k - 1$ naprej ni vzbujanja). Podobno kot pri Cutlerjevi metodi DMC, lahko torej zapišemo:

$$\hat{y}(k+j) = G_j(q^{-1})\Delta u(k+j-d) + p_j , \quad (17.78)$$

kjer je p_j

$$p_j = \Gamma_j(q^{-1})\Delta u_f(k) + F_j y_f(k) , \quad (17.79)$$

$$\Delta u_f(k) = \frac{\Delta u(k)}{T(q^{-1})} , \quad (17.80)$$

$$y_f(k) = \frac{y(k)}{T(q^{-1})} . \quad (17.81)$$

Izpeljavo lahko naprej nadaljujemo podobno kot pri DMC metodi, vendar v praksi ocene prostega odziva p_j ne računamo po enačbi 17.79, pač pa direktno po enačbi 17.75, pri čemer si pomagamo z že omenjeno Diofantovim enačbo.

17.9.1 Izpeljava regulacijskega zakona

Predikcijo izhoda procesa iz enačbe 17.75 zapišimo v vektorski obliki:

$$\underline{\hat{y}}(k) = \underline{G}\underline{\Delta u}(k) + \underline{\Gamma}\underline{\Delta u}_f^-(k) + \underline{F}\underline{y}_f(k) , \quad (17.82)$$

kjer veljajo naslednji zapisi

$$\begin{aligned} \underline{\hat{y}}(k) &= [\hat{y}(k+N_1), \hat{y}(k+N_1+1), \dots, \hat{y}(k+N_2)]^T , \\ \underline{y}_f(k) &= [y_f(k), y_f(k-1), \dots, y_f(k-n_a)]^T , \\ \underline{\Delta u}(k) &= [\Delta u(k), \Delta u(k+1), \dots, \Delta u(k+N_u-1)]^T , \\ \underline{\Delta u}_f^-(k) &= [\Delta u(k-1), \Delta u(k-2), \dots, \Delta u(k-d-n_b)]^T , \end{aligned}$$

kjer je $\underline{\hat{y}}(k)$ vektor napovedanih izhodov procesa, $\underline{y}_f(k)$ je vektor trenutnega in preteklih filtriranih izhodov procesa, $\underline{\Delta u}(k)$ je vektor trenutnega in prihodnjih sprememb regulirnega signala, $\underline{\Delta u}_f^-(k)$ je vektor preteklih sprememb regulirnega signala in:

$$\underline{G} = \begin{bmatrix} g_{N_1} & \dots & g_1 & 0 & \dots & \dots & 0 \\ g_{N_1+1} & \dots & g_2 & g_1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ g_{N_2} & \dots & \dots & \dots & \dots & \dots & g_{N_2-N_u+1} \end{bmatrix} , \quad (17.83)$$

$$\underline{\Gamma} = \begin{bmatrix} \Gamma_{N_1} \\ \Gamma_{N_1+1} \\ \vdots \\ \Gamma_{N_2} \end{bmatrix} , \quad \underline{F} = \begin{bmatrix} F_{N_1} \\ F_{N_1+1} \\ \vdots \\ F_{N_2} \end{bmatrix} . \quad (17.84)$$

Za izpeljavo regulirnega zakona moramo optimirati naslednjo kriterijsko funkcijo

$$J = \sum_{j=N_1}^{N_2} (\hat{y}(k+j) - r(k+j))^2 + \lambda \sum_{j=0}^{N_u-1} (\Delta u(k+j))^2 , \quad (17.85)$$

ob predpostavki, da so vse spremembe regulirnega signala za $j \geq N_u$ enake $\Delta u(k+j) = 0$. Cenilko lahko v vektorsko-matrični obliki zapišemo kot:

$$\begin{aligned} J &= (\hat{y}(k) - \underline{r}(k))^T (\hat{y}(k) - \underline{r}(k)) + \lambda \Delta \underline{u}^T(k) \Delta \underline{u} \\ &= (\underline{G} \Delta \underline{u} + \underline{\Gamma} \Delta \underline{u}_f^- + \underline{F} \underline{y}_f - \underline{r})^T (\underline{G} \Delta \underline{u} + \underline{\Gamma} \Delta \underline{u}_f^- + \underline{F} \underline{y}_f - \underline{r}) + \lambda \Delta \underline{u}^T \Delta \underline{u}, \end{aligned} \quad (17.86)$$

kjer je $\underline{r}(k)$ vektor predikcije referenčnega signala v prihodnosti:

$$\underline{r}(k) = [r(k+N_1), r(k+N_1+l), \dots, r(k+N_2)]^T.$$

Analitično rešitev kriterijske funkcije v enačbi 17.86 za vektor prihodnjih sprememb regulirnega signala $\Delta \underline{u}(k)$ dobimo po standardnem postopku:

$$\begin{aligned} \frac{\partial J}{\partial \Delta \underline{u}} &= 2\underline{G}^T \underline{G} \Delta \underline{u} - 2\underline{G}^T (\underline{\Gamma} \Delta \underline{u}_f^- + \underline{F} \underline{y}_f - \underline{r}) + 2\lambda \Delta \underline{u} = 0, \\ \Delta \underline{u} &= (\underline{G}^T \underline{G} + \lambda \underline{I})^{-1} \underline{G}^T (\underline{r} - \underline{\Gamma} \Delta \underline{u}_f^- - \underline{F} \underline{y}_f). \end{aligned} \quad (17.88)$$

Rešitev je vektor prihodnjih sprememb regulirnega signala. Za vodenje pa uporabimo samo prvi element vektorja $\Delta \underline{u}(k)$ in v naslednjem časovnem trenutku celoten postopek ponovimo z novimi meritvami po principu pomicnega horizonta. Za časovno nespremenljive procese ima regulator GPC konstantno ojačenje $(\underline{G}^T \underline{G} + \lambda \underline{I})^{-1} \underline{G}^T$, spreminja se samo vektorji $\Delta \underline{u}_f^-(k)$, $\underline{y}_f(k)$ in $\underline{r}(k)$. Zaprtovančna strategija vodenja je pri GPC regulatorju zajeta pri načrtovanju (izračun matrik \underline{G} , \underline{r} , in \underline{F}), implementaciji (konstrukcija vektorjev preteklih signalov) ter pri uporabi principa pomicnega horizonta.

17.9.2 Primer razvoja osnovnega prediktivnega regulatorja

Tipična kriterijska funkcija upošteva odstopanje napovedi reguliranega signala od želene reference na izbranem časovnem intervalu, ki ga imenujemo interval ujemanja ($N_2 - N_1$). Ta interval se običajno ne prične takoj, temveč šele v bližnji prihodnosti, saj bo trenutna vrednost regulirnega signala vplivala na regulirani signal šele v prihodnosti (N_1). Poleg odstopanja reguliranega signala od želene reference pa upošteva kriterijska funkcija še stroške, ki so potrebni za sledenje referenčni vrednosti, izražajo pa se z odstopanjmi od regulirnega signala od njegove ustaljene vrednosti.

Oglejmo si prediktivno vodenje procesa prvega reda z ojačenjem $K = 1$ in s časovno konstanto $T = 1.96$ s oziroma z diskretno prenosno funkcijo (pri času vzorčenja $T_s = 1$ s)

$$G_p(z) = \frac{Y(z)}{U(z)} = \frac{0.4z^{-1}}{1 - 0.6z^{-1}}. \quad (17.89)$$

Če pomnožimo prenosno funkcijo z operatorjem $\Delta = 1 - z^{-1}$ v števcu in imenovalcu, dobimo

$$(1 - 0.6z^{-1})(1 - z^{-1})Y(z) = 0.4\Delta U(z), \quad (17.90)$$

ozioroma v časovnem prostoru

$$y(k) = 1.6y(k-1) - 0.6y(k-2) + 0.4\Delta u(k-1). \quad (17.91)$$

Izbrano kriterijsko funkcijo, ki zajema interval ujemanja med $N_1 = 1$ in $N_2 = 3$ ter regulirni horizont $N_u = 2$, lahko zapišemo v naslednji obliki

$$J = \sum_{j=1}^3 \left(y(k+j) - r(k+j) \right)^2 + \lambda \sum_{j=0}^1 \left(\Delta u(k+j) \right)^2, \quad (17.92)$$

kjer je λ parameter, ki uravnava razmerje med kvaliteto vodenja in za to potrebnimi stroški. Iz enačbe 17.91 sledijo predikcije reguliranega signala

$$\begin{aligned} \hat{y}(k+1) &= 1.6y(k) - 0.6y(k-1) + 0.4\Delta u(k), \\ \hat{y}(k+2) &= 1.6\hat{y}(k+1) - 0.6y(k) + 0.4\Delta u(k+1) = \\ &= 1.96y(k) - 0.96y(k-1) + 0.4\Delta u(k+1) + 0.64\Delta u(k), \\ \hat{y}(k+3) &= 1.6\hat{y}(k+2) - 0.6\hat{y}(k+1) + 0.4\Delta u(k+2) = \\ &= 2.1760y(k) - 1.1760y(k-1) + 0.4\Delta u(k+2) + 0.64\Delta u(k+1) + 0.784\Delta u(k), \end{aligned}$$

oziroma v matrični obliki ob upoštevanju dejstva, da se regulirna veličina po intervalu ne spreminja več ($\Delta u(k+2) = 0$)

$$\begin{bmatrix} \hat{y}(k+1) \\ \hat{y}(k+2) \\ \hat{y}(k+3) \end{bmatrix} = \begin{bmatrix} 0.4 & 0 \\ 0.64 & 0.4 \\ 0.7840 & 0.64 \end{bmatrix} \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \end{bmatrix} + \begin{bmatrix} 1.6 & -0.6 \\ 1.96 & -0.96 \\ 2.1760 & -1.1760 \end{bmatrix} \begin{bmatrix} y(k) \\ y(k-1) \end{bmatrix}. \quad (17.93)$$

Skrajšani zapis te enačbe je

$$\underline{y}^+ = \underline{G}\Delta\underline{u}^+ + \underline{F}\underline{y}^-,$$

kjer oznaka $+$ pomeni, da gre za neznane vrednosti signala v prihodnosti in oznaka $-$, da gre za znane pretekle vrednosti signalov. Če bi imeli sistem višjega reda, bi se na desni strani zgornje enačbe pojavil še izraz $\underline{\Gamma}\Delta\underline{u}^-$, ki ga bomo zaradi popolnosti pri izpeljavi regulacijske zakonitosti upoštevali. V našem primeru seveda velja $\underline{\Gamma} = \underline{0}$.

Kriterijsko funkcijo v enačbi 17.92 lahko zapišemo v obliki

$$\begin{aligned} J &= (\underline{y}^+ - \underline{r}^+)^T (\underline{y}^+ - \underline{r}^+) + \lambda \Delta\underline{u}^{+T} \underline{u}^+ = \\ &= (\underline{G}\Delta\underline{u}^+ + \underline{\Gamma}\Delta\underline{u}^- + \underline{F}\underline{y}^- - \underline{r}^+)^T \cdot (\underline{G}\Delta\underline{u}^+ + \underline{\Gamma}\Delta\underline{u}^- + \underline{F}\underline{y}^- - \underline{r}^+) + \lambda \Delta\underline{u}^{+T} \Delta\underline{u}^+, \end{aligned}$$

kjer vektor \underline{r}^+ zajema tri prihodnje vrednosti referenčnega signala. Poiščimo vektor prihodnjih vrednosti reguliranega signala, ki minimizira kriterijsko funkcijo

$$\frac{\partial J}{\partial \Delta\underline{u}^+} = 2\underline{G}^T (\underline{G}\Delta\underline{u}^+ + \underline{\Gamma}\Delta\underline{u}^- + \underline{F}\underline{y}^- - \underline{r}^+) + 2\lambda \Delta\underline{u}^+ = 0$$

iz česar sledi

$$\Delta\underline{u}^+ = (\underline{G}^T \underline{G} + \lambda \underline{I})^{-1} \underline{G}^T (\underline{r}^+ - \underline{\Gamma}\Delta\underline{u}^- + \underline{F}\underline{y}^-). \quad (17.95)$$

Po vstavitevi matrik \underline{G} in \underline{F} iz enačbe 17.93, $\underline{\Gamma}$, ki je v našem primeru enak nič in izbranem $\lambda = 0.1$, lahko zapišemo prvo vrstico enačbe 17.95 v naslednji obliki.

$$\Delta u(k) = -2.6648y(k) + 1.1485y(k-1) + 0.9374r(k+1) + 0.4390r(k+2) + 0.1400r(k+3).$$

Ta enačba predstavlja regulirni zakon. Opazimo lahko, da ta zakon zahteva poznavanje prihodnjih vrednosti referenčnega signala. Če le teh ne poznamo, jih v praksi zamenjamo z njegovo trenutno vrednostjo. V tem primeru izhod optimalno sledi zakasnjemu referenčnemu signalu. Splošna značilnost prediktivnega regulatorja in prednost pred regulatorjem PID je v bolj gladkem regulirnem signalu in odzivu z manj prenihaja.

17.10 Splošni časovo-zvezni predikcijski regulator (CGPC)

V tem delu je predstavljen splošni časovno-zvezni predikcijski regulator (*ang. Continuous-time Generalized Predictive Control* - CGPC), ki je eden od pomembnejših algoritmov prediktivnega vodenja. Obravnavamo ga predvsem zaradi njegovega zapisa v zveznem časovnem prostoru.

Algoritem splošnega predikcijskega regulatorja (GPC) je razvit v diskretnem časovnem prostoru. CGPC predstavlja njegov časovno-zvezni ekvivalent. Lastnosti obeh so zelo podobne.

CGPC algoritmom zagotavlja stabilno delovanje za široko množico najrazličnejših dinamičnih procesov. Procesi, ki jih lahko ustrezeno vodimo s CGPC regulatorjem, so lahko z zakasnitvijo, fazno neminimalni ali višjega reda.

17.10.1 Osnovni CGPC algoritmom

Predpostavimo absolutno pravi, linearji, časovno nespremenljivi sistem z enim vhodom in enim izhodom (*ang. Single-Input Single-Output* - SISO)

$$Y(s) = \frac{B(s)}{A(s)}U(s) + \frac{C(s)}{A(s)}V(s), \quad (17.96)$$

kjer so $A(s), B(s)$ in $C(s)$ polinomi v Laplacovem prostoru. $Y(s)$ je izhod sistema, $U(s)$ vhod in $V(s)$ motnja. $C(s)$ predstavlja števec prenosne funkcije šumnega filtra. Določimo ga s pomočjo pospoljene metode najmanjših kvadratov ali pa ga definira načrtovalec. Red $C(s)$ je odvisen od karakteristike motnje.

V zgornji enačbi nismo eksplisitno vključili časovne zakasnitve. Zakasnitev je lahko aproksimativno vključena v polinoma $A(s)$ in $B(s)$ s pomočjo ene od metod aproksimacije mrtvega časa (*Padéjeva ali Taylorjeva vrsta*). Zaradi tega lahko sisteme z zakasnitvijo opišemo na enak način kot sisteme brez zakasnitve le, da se v tem primeru zviša red sistema.

Izpeljava algoritma temelji na predikciji izhodnega signala. V splošnem dobimo predikcijo časovne funkcije na osnovi njene trenutne vrednosti in vrednosti njenih odvodov v tem trenutku.

Predikcija izhodnega signala. Kot pove ime, temelji prediktivno vodenje na napovedi predikciji (*predvidevanju*) izhodnega signala $y(t)$. Za razliko od minimalno-variančnega, kjer je napoved izračunana samo za en korak, temelji regulator CGPC na nizu predvidenih vrednosti.

Napoved funkcije ali signala dobimo z razvojem v *Maclaurinovo vrsto*

$$\hat{y}(t+T) = y(t) + \sum_{k=1}^{N_y} y_k(t) \frac{T^k}{k!}, \quad (17.97)$$

kjer je

$$y_k(t) = \frac{d^k y(t)}{dt^k}, \quad (17.98)$$

N_y predstavlja red napovedi in T čas pri katerem računamo napoved vrednosti funkcije. Vidimo, da je prihodnja vrednost funkcije odvisna od trenutne vrednosti funkcije in trenutnih vrednosti odvodov funkcije. Govorimo o aproksimaciji prihodnje vrednosti signala $y(t)$. Napaka pri aproksimaciji je odvisna od časa T in reda predikcije N_y .

V enačbi 17.97 potrebujemo za izračun napovedi odvode izhodnega signala $y(t)$, ki pa v splošnem niso znani.

Izračun odvodov. Množenje z Laplacovim operatorjem s ustrezza odvajanjju v časovnem prostoru. Zato lahko $k - ti$ odvod izhodnega signala po času zapišemo v Laplacovem prostoru kot

$$Y_k(s) = s^k Y(s) = \frac{s^k B(s)}{A(s)} U(s) + \frac{s^k C(s)}{A(s)} V(s) . \quad (17.99)$$

Če prenosno funkcijo drugega sumanda na desni strani enačbe razdelimo na dva dela, dobimo

$$\frac{s^k C(s)}{A(s)} = E_k(s) + \frac{F_k(s)}{A(s)} , \quad (17.100)$$

$$\begin{aligned} red(F_k) &= red(A) - 1 , \\ red(E_k) &= k - 1 , \end{aligned}$$

kjer prenosna funkcija $\frac{F_k(s)}{A(s)}$ predstavlja *absolutno pravi* del in $E_k(s)$ njen ostanek.

Absolutno prava prenosna funkcija. Absolutno prava je prenosna funkcija $G(s) = \frac{B(s)}{A(s)}$ takrat, kadar za njen relativni red velja

$$\rho \geq 1 , \text{ kjer je } \rho = red(A) - red(B) .$$

K -ti odvod izhoda $y(t)$ lahko sedaj zapišemo s pomočjo zapisa $Y_k^*(s)$ in ustrezne napake $E_k^*(s)$

$$Y_k(s) = Y_k^*(s) + E_k^*(s) , \quad (17.101)$$

$$Y_k^*(s) = \frac{s^k B(s)}{A(s)} U(s) + \frac{F_k(s)}{A(s)} V(s) , \quad (17.102)$$

$$E_k^*(s) = E_k(s) V(s) . \quad (17.103)$$

Če vstavimo v enačbo 17.102 izraz za $V(s)$ iz enačbe 17.96, dobimo

$$Y_k^*(s) = \frac{E_k(s) B(s)}{C(s)} U(s) + \frac{F_k(s)}{C(s)} Y(s) , \quad (17.104)$$

kjer $Y_k^*(s)$ predstavlja Laplaceov transform k -tega odvoda signala $y(t)$. Prenosna funkcija $\frac{F_k(s)}{C(s)}$ je absolutno prava. Relativni red prenosne funkcije $\frac{E_k(s) B(s)}{C(s)}$ je $\rho - k$ in jo je zato možno realizirati pri pogoju $k \leq \rho$.

Odvode izhodnega signala nadomestimo z njihovimi ocenjenimi vrednostmi

$$Y_k^*(s) = \frac{E_k(s) B(s)}{C(s)} U(s) + \frac{F_k(s)}{C(s)} Y(s) . \quad (17.105)$$

V primeru $k > \rho$ lahko prenosno funkcijo $\frac{E_k(s) B(s)}{C(s)}$ zapišemo v obliki

$$\frac{E_k(s) B(s)}{C(s)} = H_k(s) + \frac{G_k(s)}{C(s)} , \quad (17.106)$$

$$\begin{aligned} red(H_k) &= k - \rho , \\ red(G_k) &= n - 2 , \\ red(A) &= n . \end{aligned}$$

Sedaj lahko enačbo 17.104 zapišemo v obliki

$$Y_k^*(s) = H_k(s)U(s) + \frac{G_k(s)}{C(s)}U(s) + \frac{F_k(s)}{C(s)}Y(s) . \quad (17.107)$$

Prenosni funkciji $\frac{G_k(s)}{C(s)}$ in $\frac{F_k(s)}{C(s)}$ sta absolutno pravi in ju je mogoče realizirati. Zato lahko zgornjo enačbo zapišemo v obliki, ki predstavlja realizabilni del in ostanek

$$Y_k^*(s) = H_k(s)U(s) + Y_k^0(s) , \quad (17.108)$$

$$Y_k^0(s) = \frac{G_k(s)}{C(s)}U(s) + \frac{F_k(s)}{C(s)}Y(s) , \quad (17.109)$$

kjer je $Y_k^0(s)$ del, ki ga lahko realiziramo. Enačbo 17.108 lahko v časovnem prostoru zapišemo

$$y_k^*(t) = \underline{h}_k \underline{u} + y_k^0(t) , \quad (17.110)$$

$$\underline{u} = [u(t) \ u_1(t) \dots u_{k-\rho}(t)]^T , \quad (17.111)$$

$$\underline{h}_k = [h_0 \ h_1 \dots h_{k-\rho}] , \quad (17.112)$$

$$u_k(t) = \frac{d^k u(t)}{dt^k} , \quad (17.113)$$

kjer je \underline{h}_k vrstični vektor, ki vsebuje koeficiente polinoma $H_k(s)$ in \underline{u} stolpični vektor, ki vsebuje odvode vhodnega signala $u(t)$. Vektor \underline{h}_k sestavlja Markovi parametri odprtozančnega sistema $\frac{B(s)}{A(s)}$. Dobimo jih kot količnik pri deljenju polinoma $s^k B(s)$ s polinomom $A(s)$.

Markovi parametri Markovi parametri so koeficienti polinoma, ki jih dobimo z deljenjem polinoma $B(s)$ s polinomom $A(s)$

$$\frac{B(s)}{A(s)} = M(s) + \frac{R(s)}{A(s)} .$$

$M(s)$ predstavlja Markov polinom sestavljen iz Markovih koeficientov

$$M(s) = m_\delta s^\delta + \dots + m_0 \text{ za } \delta = \text{red}(B) - \text{red}(A) > 0 .$$

Polinom $R(s)$ predstavlja ostanek pri deljenju, za katerega velja $\text{red}(R) = \text{red}(A) - 1$.

Red vodenja. Analogno diskretnemu tudi v primeru zveznega predikcijskega algoritma predpostavimo stopnjo razvoja regulirnega signala. Predpostavimo, da so odvodi regulirnega signala, ki so višji od reda N_u , enaki nič

$$u_k(t) = 0 \ \forall k > N_u . \quad (17.114)$$

Parameter N_u imenujemo red vodenja. Njegov vpliv na regulacijo sistema bo razložen kasneje. Z uvedbo reda vodenja zmanjšamo dimenzijo vektorja \underline{u} . Njegova dimenzija je sedaj $(N_u + 1) \times 1$

$$\underline{u} = [u(t) \ u_1(t) \dots u_{N_u}(t)]^T . \quad (17.115)$$

Z uvrstitevijo enačbe 17.110, ki opisuje oceno odvodov, v enačbo 17.97, dobimo izraz za napoved izhodnega signala $y(t)$ za čas T . Dobimo izraz v matrični obliki

$$y^*(t + T) = \underline{T}_{N_y} \underline{H} \underline{u} + \underline{T}_{N_y} \underline{Y}^0 , \quad (17.116)$$

kjer je

$$\underline{T}_{N_y} = \left[1 \ T \ \frac{T^2}{2!} \ \dots \ \frac{T^{N_y}}{N_y!} \right] , \quad (17.117)$$

$$\underline{Y}^0 = [y(t) \ y_1^0(t) \ \dots \ y_{N_y}^0(t)]^T , \quad (17.118)$$

in je matrika \underline{H} reda $(N_y + 1) \times (N_u + 1)$

$$\underline{H} = \begin{bmatrix} 0 & 0 & \dots & 0 \\ h_1 & 0 & \dots & 0 \\ h_2 & h_1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ h_{N_y} & \dots & \dots & h_{(N_y - N_u)} \end{bmatrix} \quad (17.119)$$

sestavljeni iz Markovih koeficientov.

Interpretacija predikcije izhodnega signala. Za vsak čas t lahko prihodnji odziv linear-nega sistema razdelimo na tri dele

$$y(t+T) = y_u(t, T) + y_i(t, T) + v(t+T), \quad (17.120)$$

kjer je

- $y_u(t, T)$ je vrednost odziva sistema na vhodni signal $u(t)$ pri času T , pri začetnih pogojih enakih nič,
- $y_i(t, T)$ je odziv sistema na začetne pogoje sistema pri času T , pri vhodnem signalu enakem nič,
- $v(t+T)$ je prihodnja motnja.

Za vsak čas t je vrednost $y_i(t, T)$ natančno določena, vrednost $y_u(t, T)$ je odvisna od prihodnjega vhodnega signala, medtem, ko vrednosti $v(t+T)$ ne poznamo. Če zanemarimo vrednost motnje v zgornji enačbi, lahko gornjo enačbo zapišemo s pomočjo konvolucije

$$\tilde{y}(t+T) = h(T) * u(t+T) + y_i(t, T), \quad (17.121)$$

kjer je $h(T)$ impulzni odziv sistema pri času T .

Gornja enačba ima enako obliko kot enačba predikcije izhodnega signala v enačbi 17.116. Ugotovimo lahko, da izraz $\underline{T}_{N_y} \underline{Y}^0$ sestavlja vrednosti izhodnega signala $y(t)$ in njegovih odvodov pri času t , kar predstavlja odziv sistema na začetne pogoje $y_i(t)$. Izraz $\underline{T}_{N_y} \underline{H} u$ pa predstavlja odziv sistema na prihodnji vhodni signal, kar ustreza aproksimaciji izraza $h(T) * u(t+T)$.

Referenčna trajektorija Sledenje odsekoma zvezni referenčni vrednosti povzroča, zaradi hipnih sprememb njene vrednosti, konice regulirnega signala. Da bi se temu izognili, predpostavimo modelno-referenčno trajektorijo, ki je gladka na celotnem predikcijskem intervalu.

Ta pristop zmanjša prevzpone regulirnega signala $u(t)$, hkrati pa omogoča modelno-referenčni način vodenja, saj lahko s primerno izbiro parametrov CGPC sledimo modelno-referenčni trajektoriji, ki je izhod referenčnega modela.

Referenčno trajektorijo $w_r(t)$ definiramo kot izhod prenosne funkcije $\frac{R_n(s)}{R_d(s)}$ (referenčnega modela). Modelno-referenčno prenosno funkcijo lahko razvijemo v Maclarenovo vrsto

$$\frac{R_n(s)}{R_d(s)} = \lim_{N_y \rightarrow \infty} \sum_{i=0}^{N_y} r_i s^{-i}, \quad (17.122)$$

kjer so r_i Markovi parametri modelno-referenčne prenosne funkcije. Na enak način kot v primeru predikcije izhodnega signala lahko sedaj zapišemo predikcijo za modelno-referenčno trajektorijo $w_r^*(t, T)$

$$w_r^*(t, T) = \underline{T}_{N_y} \underline{w}, \quad (17.123)$$

kjer je

$$\underline{w} = \underline{r}[w(t) - y(t)] , \quad (17.124)$$

$$\underline{r} = [r_0 \ r_1 \dots r_{N_y}]^T \quad (17.125)$$

in je r_k k -ti Markov parameter prenosne funkcije modela $\frac{R_n(s)}{R_d(s)}$ in nastopa izraz razlike $w(t) - y(t)$ zaradi premika izhodišča sistema v točko $y(t)$.

Lahko povzamemo, da nekatere metode uvajajo filtriranje referenčnega signala, da bi se s tem izognili koničam regulirnega signala. Modelno-referenčna prenosna funkcija se implicitno odraža v ojačenju regulatorja.

Regulirni zakon. Splošni zvezni predikcijski regulator temelji na optimizaciji kriterijske funkcije J , ki je funkcija pogreška med prihodnjim izhodnim signalom $y(t+T)$ in prihodnjim referenčnim signalom $w(t+T)$ glede na spremenljiv časovni odmik T . Časovni odmik T je definiran glede na izhodišče, ki ga določa trenutni čas t .

Za vsak časovni trenutek t predpostavimo pomožne spremenljivke, na osnovi katerih minimiziramo kriterijsko funkcijo. Predpostavimo vhodni pomožni signal $u_r^*(t, T)$, izhodni pomožni signal $y_r^*(t, T)$ in referenčni pomožni signal $w_r^*(t, T)$ pri spremenljivem časovnem odmiku T in konstantnem času t . Definiramo jih tako, da so za $T = 0$ enaki prvotnim

$$y_r^*(t, 0) = y(t+T) - y(t) = 0 , \quad (17.126)$$

$$u_r^*(t, 0) = u(t) . \quad (17.127)$$

Predikcija izhodnega signala je odvisna od vhodnega signala $u(t)$ in njegovih odvodov. Če so le-ti znani lahko izračunamo predikcijo izhodnega signala. Algoritem regulatorja CGPC tvori regulirni signal in njegove odvode tako, da je odstopanje predikcije signala $y(t)$ od referenčne trajektorije minimalno. Iskanje optimalnega regulirnega signala in njegovih odvodov je ekvivalentno iskanju predikcije vhodnega signala $u_r^*(t, T)$, saj lahko regulirni signal zapišemo kot $u(t) = u_r^*(t)$. Do rešitve pridemo z minimizacijo kriterijske funkcije

$$J = \int_{T_1}^{T_2} (y_r^*(t, T) - w_r^*(t, T))^2 dT + \lambda \int_0^{T_2-T_1} (u_r^*(t, T))^2 dT , \quad (17.128)$$

kjer je

$$y_r^*(t, T) = y^*(t+T) - y(t) , \quad (17.129)$$

$$u_r^*(t, T) = \sum_{k=0}^{N_u} u_k(t) \frac{T^k}{k!} = u_r^*(t) , \quad (17.130)$$

$$w_r^*(t, T) = \sum_{k=0}^{N_u} w_k(t) \frac{T^k}{k!} , \quad (17.131)$$

kjer sta $u_k(t)$ in $w_k(t)$ vrednosti k -tega odvoda vhodnega, oziroma referenčnega signala, v časovnem trenutku t .

Glede na izbrano kriterijsko funkcijo lahko povzamemo naslednje ugotovitve:

- Minimalni horizont predikcije T_1 , ki je vključen v kriterijsko funkcijo kot spodnja meja kriterijskega integrala je lahko vedno enak nič. V tem primeru integriramo na intervalu $[0, T_2]$. Če ima sistem zakasnitev enako T_d , potem je vrednost pod integralom na intervalu $[0, T_d]$ enaka nič. Zato lahko spodnjo mejo integriranja enačimo z zakasnitojico sistema ($T_1 = T_d$).

- Z izbiro zgornje meje integracije vplivamo na dinamiko izhodnega signala.
- Predikcija vhodnega signala $u_r^*(t, T)$ ima vpliv na predikcijo izhodnega signala $y_r^*(t, T)$.
- Vpliv motnje je indirektno vključen v kriterijsko funkcijo preko trenutene vrednosti izhodnega signala $y(t)$.

Izpeljava regulirnega zakona Kriterijsko funkcijo lahko sedaj zapišemo v matrični obliki

$$J = \int_{T_1}^{T_2} \left(\underline{T}_{N_y} \underline{H} \underline{u} + \underline{T}_{N_y} \underline{Y}_0 - \underline{T}_{N_y} \underline{w} \right)^2 dT + \lambda \int_0^{T_2 - T_1} (\underline{u}^T \underline{T}_{N_u}^T \underline{T}_{N_u} \underline{u}) dT. \quad (17.132)$$

Z zadružitvijo obeh integralov dobimo

$$J = \int_{T_1}^{T_2} \left(\underline{u}^T \underline{H}^T \underline{T}_{N_y}^T + \underline{Y}_0^T \underline{T}_{N_y}^T - \underline{w}^T \underline{T}_{N_y}^T \right) \left(\underline{T}_{N_y} \underline{H} \underline{u} + \underline{T}_{N_y} \underline{Y}_0 - \underline{T}_{N_y} \underline{w} \right) dT \quad (17.133)$$

$$+ \lambda \underline{u}^T \underline{T}_{N_u}^T \underline{T}_{N_u} \underline{u} dT. \quad (17.134)$$

Izraz pod integralom predstavlja Hamiltonovo funkcijo

$$\mathcal{H}(\underline{u}(t), \underline{w}(t), \lambda, t) = \underline{u}^T \underline{H}^T \underline{T}_y \underline{H} \underline{u} + (\underline{Y}_0)^T \underline{T}_y \underline{H} \underline{u} - \underline{w}^T \underline{T}_y \underline{H} \underline{u} \quad (17.135)$$

$$+ \underline{u}^T \underline{H}^T \underline{T}_y \underline{Y}_0 + (\underline{Y}_0)^T \underline{T}_y \underline{Y}_0 - \underline{w}^T \underline{T}_y \underline{Y}_0 \quad (17.136)$$

$$- \underline{u}^T \underline{H}^T \underline{T}_y \underline{w} - (\underline{Y}_0)^T \underline{T}_y \underline{w} + \underline{w}^T \underline{T}_y \underline{w} \quad (17.137)$$

$$+ \lambda \underline{u}^T \underline{T}_u \underline{u}, \quad (17.138)$$

kjer velja naslednja substitucija

$$\underline{T}_y = \int_{T_1}^{T_2} \underline{T}_{N_y}^T \underline{T}_{N_y} dT, \text{ reda}(N_y + 1) \times (N_y + 1), \quad (17.139)$$

$$\underline{T}_u = \int_{T_1}^{T_2} \underline{T}_{N_u}^T \underline{T}_{N_u} dT, \text{ reda}(N_u + 1) \times (N_u + 1). \quad (17.140)$$

Matriki \underline{T}_y in \underline{T}_u sedaj nastopata kot konstantni matriki in ne več kot funkciji časovnega odmika T . Potreben pogoj optimalnosti je izpolnjen z zadovoljtvijo tretje Euler-Lagrangove enačbe

$$\frac{\partial \mathcal{H}(\underline{u}(t), \lambda, t)}{\partial \underline{u}} = 0, \quad (17.141)$$

kar rezultira v

$$\frac{\partial \mathcal{H}(\underline{u}(t), \lambda, t)}{\partial \underline{u}} = 2 \underline{H}^T \underline{T}_y \underline{H} \underline{u} + 2 \underline{H}^T \underline{T}_y \underline{H} \underline{Y}_0 - 2 \underline{H}^T \underline{T}_y \underline{w} + 2 \lambda \underline{T}_u \underline{u}. \quad (17.142)$$

Izpolnjen je tudi zadosten pogoj optimalnosti, ki ga opisuje Legandre-Clebschova enačba, saj velja

$$\frac{\partial^2 \mathcal{H}(\underline{u}(t), \lambda, t)}{\partial \underline{u}^2} = 2 \lambda \underline{T}_u \geq 0, \quad (17.143)$$

ker je matrika \underline{T}_u v enačbi 17.140 pozitivno definitna.

Optimalni regulirni zakon tako sledi

$$\underline{u} = (\underline{H}^T \underline{T}_y \underline{H} + \lambda \underline{T}_u)^{-1} \underline{H}^T \underline{T}_y (\underline{w} - \underline{Y}_0) \quad (17.144)$$

ali

$$\underline{u} = \underline{K}(\underline{w} - \underline{Y}_0) , \quad (17.145)$$

$$\underline{K} = (\underline{H}^T \underline{T}_y \underline{H} + \lambda \underline{T}_u)^{-1} \underline{H}^T \underline{T}_y , \quad (N_y - \rho + 1) \times (N_y + 1) . \quad (17.146)$$

Optimalni vhodni signal $u(t)$ nam opisuje le prva vrstica matrične enačbe (takrat nadomestimo v zgornji enačbi matriko \underline{K} z vektorjem \underline{k} , ki je enak prvi vrstici matrike \underline{K})

$$u(t) = \underline{k}(\underline{w} - \underline{Y}_0) \quad (17.147)$$

ali zapisano v Laplacovem prostoru

$$U(s) = \underline{k}r(W(s) - Y(s)) - \underline{k}\underline{Y}(s)_0 , \quad (17.148)$$

kjer je $\underline{Y}_0(s)$ Laplacov transform \underline{Y}_0

$$\underline{Y}_0(s) = \frac{\underline{G}\underline{S}_G(s)}{C(s)} U(s) + \frac{\underline{F}\underline{S}_F(s)}{C(s)} Y(s) \quad (17.149)$$

in sta \underline{G} in \underline{F} matriki koeficientov polinomov $G_k(s)$ oziroma $F_k(s)$ reda $(N_y + 1) \times (n - 1)$ oziroma $(N_y + 1) \times n$. \underline{S}_G in \underline{S}_F sta pripadajoča s vektorja

$$\underline{S}_G = [s^{n-2} \ s^{n-3} \ \dots \ s \ 1]^T , \quad (17.150)$$

$$\underline{S}_F = [s^{n-1} \ s^{n-2} \ \dots \ s \ 1]^T . \quad (17.151)$$

Tedaj lahko zapišemo končno obliko regulirnega zakona v Laplacovem prostoru

$$U(s) = g(W(s) - Y(s)) - \frac{G_0(s)}{C(s)} U(s) - \frac{F_0(s)}{C(s)} Y(s) , \quad (17.152)$$

kjer je

$$g = \underline{k}r , \quad (17.153)$$

$$G_0(s) = \underline{k}\underline{G}\underline{S}_G , \quad (17.154)$$

$$F_0(s) = \underline{k}\underline{F}\underline{S}_F \quad (17.155)$$

in je g skalar, $G_0(s)$ in $F_0(s)$ pa sta polinoma spremenljivke s .

Prenosna funkcija regulatorja. Prenosno funkcijo regulatorja lahko zapišemo v naslednji obliki

$$G_r(s) = \frac{gA(s)}{A(s) + L_0(s)} , \quad (17.156)$$

kjer je

$$L_0 = \frac{G_0}{C} + \frac{F_0}{C} B . \quad (17.157)$$

Stopnja polinoma v imenovalcu je v splošnem enaka stopnji polinoma $A(s)$, relativni red prenosne funkcije regulatorja pa je $\rho = 0$. V primeru, ko ima prenosna funkcija procesa pole v koordinatnem izhodišču, se ti krajšajo s korenji imenovalca prenosne funkcije regulatorja tako, da dobimo v direktni veji povratnozančnega sistema integrirno delovanje (I_1). Na ta način se zmanjša red regulatorja.

Krajšanje ničel odprtozančnega sistema. Pri izbiri uteži $\lambda = 0$, reda vodenja N_u in reda predikcije N_y tako, da velja relacija $N_u = N_y - \rho$, se ničle prenosne funkcije procesa krajšajo s poli regulatorja. V tem primeru lahko zaprtozančni odziv sistema podamo z enačbo

$$Y(s) = \frac{g}{Z_0(s) + g} W(s), \quad (17.158)$$

kjer $Z_0(s)$ podaja naslednja relacija

$$A(s) + L_0(s) = Z_0(s)B(s). \quad (17.159)$$

Krajšanje polov in ničel zaprtozančne prenosne funkcije povzroči v tem primeru nestabilnost, če je prenosna funkcija procesa fazno neminimalna, saj ničle, ki se krajšajo, ležijo na desni strani ravnine s .

Izkaže pa se, da do krajšanja ne pride samo pri natančno izpolnjeni relaciji $N_u = N_y - \rho$, ampak tudi pri približno izpolnjeni relaciji. Če želimo, da ne pride do krajšanja, mora biti razlika $N_y - N_u$ prizorno velika.

Primer 17.10.1. Preprost primer naj pokaže krajšanje ničel in polov. Predpostavimo prenosno funkcijo

$$G(s) = \frac{B(s)}{A(s)} = \frac{s+1}{s(2s^2+1)} \quad (17.160)$$

$$C(s) = 0.2s^2 + s + 1. \quad (17.161)$$

Parametri regulatorja naj bodo $N_y = 6, N_u = 4, T_1 = 0, T_2 = 1$ in $\lambda = 0$. Pri danih parametrih dobimo, s pomočjo enačbe 17.156, naslednje pole in ničle regulatorja:

$$p_1 = 0, \quad (17.162)$$

$$p_2 = -1, \quad (17.163)$$

$$p_3 = -22.5000, \quad (17.164)$$

$$z_1 = 0, \quad (17.165)$$

$$z_{1,2} = \pm j0.7071. \quad (17.166)$$

Ugotovimo lahko, da se pol in ničla v izhodišču krajšata. Vseeno pa obdrži direktna veja v vodljivostni shemi integrabilni značaj, zaradi pola v izhodišču, ki ga prispeva prenosna funkcija procesa.

Zaprtozančni poli in ničle so naslednji:

$$p_{1,2} = -11.2500 \pm j7.3101, \quad (17.167)$$

$$p_3 = -1.0000, \quad (17.168)$$

$$z_1 = -1.0000. \quad (17.169)$$

Zaradi relacije med N_u in N_y pride pri zaprtozančni prenosni funkciji do krajšanja ničel in polov. V tem primeru sta to p_3 in z_1 .

Če za isti primer izračunamo regulator pri spremenjenem parametru $N_u = 1$, dobimo naslednje vrednosti korenov zaprtozančne prenosne funkcije

$$p_{1,2} = -2.6646 \pm j2.4431, \quad (17.170)$$

$$p_3 = -0.9945, \quad (17.171)$$

$$z_1 = -1.0000. \quad (17.172)$$

V tem primeru se korena minimalno razlikujeta.

Modelno-referenčni način delovanja. Predpostavimo parametre regulatorja $N_u = N_y - \rho$ in utež $\lambda = 0$. Referenčni model definira prenosna funkcija

$$G_m(s) = \frac{R_n(s)}{R_d(s)}. \quad (17.173)$$

Zaprtozančni odziv podaja v tem primeru enačba 17.158, ki jo z upoštevanjem zgornje enačbe lahko zapišemo v drugi obliki

$$Y(s) = \frac{R_n(s)}{D(s)R_d(s)}W(s), \quad (17.174)$$

$$D(s) = \frac{Z_0(s) + g}{g}. \quad (17.175)$$

Izberimo referenčni model prvega reda

$$G_M(s) = \frac{r_{n0}}{r_{d1}s + 1}. \quad (17.176)$$

Red polinoma $D(s)$ je tedaj enak $red(D(s)) = \rho - 1$, kjer je ρ relativni red odprtozančnega sistema.

Sledenje referenčnemu modelu je odvisno od dinamike polinoma $D(s)$. Dinamiko polinoma $D(s)$ lahko spremojamo z izbiro minimalnega in maksimalnega horizonta predikcije T_1 in T_2 :

- Idealno sledenje modelu prvega reda je zagotovljeno v primeru, ko je $red(D(s)) = 0$ ali $\rho = 1$. V tem primeru je polinom $D(s)$ enak r_{0n} ne glede na izbiro horizontov predikcije, saj zahteva po optimalnosti predvideva odpravo pogreška med referenco in izhodnim signalom v stacionarnem stanju

$$E(s) = Y(s) - W(s), \quad (17.177)$$

$$E(s) = \left(\frac{R_n(s)}{D(s)R_d(s)} - 1 \right) W(s), \quad (17.178)$$

$$e_{ss}(t) = \lim_{t \rightarrow \infty} e(t) = \lim_{s \rightarrow 0} sE(s) = \lim_{s \rightarrow 0} s \frac{R_n(s) - R_d(s)D(s)}{R_d(s)D(s)} W(s), \quad (17.179)$$

$$e_{ss}(t) = \lim_{s \rightarrow 0} s \frac{r_{n0} - (r_{d1} + 1)d_0}{(r_{d1}s + 1)d_0} \frac{w_{ss}}{s} = 0. \quad (17.180)$$

Sledi

$$d_0 = r_{n0}, \quad (17.181)$$

$$Y(s) = \frac{1}{r_{d1}s + 1}W(s), \quad (17.182)$$

kar rezultira v idealnem sledenju modelu $G_m(s)$, kjer je $r_{n0} = 1$.

- V primeru, ko je relativni red odprtozančne prenosne funkcije $\rho > 1$ in s tem $red(D(s)) > 0$ je zagotovljeno dobro sledenje modelu, če je izbrana dinamika polinoma $D(s)$ zanemarljiva. Dinamika pa je zanemarljiva takrat, kadar pozicioniramo korene polinoma $D(s)$ primerno daleč od dominantnih korenov modelne prenosne funkcije. To lahko dosežemo z zmanjšanjem maksimalnega horizonta predikcije T_2 . Na ta način dobimo aproksimativno modelno-referenčno delovanje.

Modelno-referenčno delovanje dobimo v primeru izpolnjene relacije $N_u = N_y - \rho$. V primeru neenakosti $N_u \neq N - y - \rho$ pa se s pomočjo referenčnega modela izognemo koničam regulirnega signala $u(t)$ v času prehodnega pojava (pri redu vodenja $N_u = 0, 1, 2$).

Primer 17.10.2. Modelno-referenčni način delovanja osvetlimo s primerom. Predpostavimo prenosno funkcijo z relativnim redom $\rho = 3$

$$G(s) = \frac{B(s)}{A(s)} = \frac{1}{s(s^2 + 1)}, \quad (17.183)$$

$$C(s) = 0.2s^2 + s + 1. \quad (17.184)$$

Parametri regulatorja naj bodo $N_y = 6$, $N_u = 4$, $T_1 = 0$, $T_2 = 1$ in $\lambda = 0$. Odziv sistema naj sledi modelu

$$G_m(s) = \frac{1}{0.6s + 1}. \quad (17.185)$$

Algoritem regulatorja pozicionira pole zaprtozančnega sistema pri vrednostih

$$p_{1,2} = -9.1667 \pm j8.0881, \quad (17.186)$$

$$p_3 = -1.6667. \quad (17.187)$$

Zaprtozančna prenosna funkcija je tretjega reda. Vrednost zaprtozančnega pola p_3 je enaka vrednosti pola modela. Položaja ostalih dveh polov pa sta taka, da lahko njun vpliv na dinamiko, v primerjavi z dinamiko pola p_3 zanemarimo.

Regulirni signal ima v prehodnem pojavu konico, ki je posledica sledenja sistema, ki je tretjega reda, modelu, ki je prvega reda.

17.10.2 Vpliv parametrov regulatorja CGPC

Poglavlje opisuje izbiro in vpliv parametrov regulatorja CGPC na zaprtozančno obnašanje.

Minimalni horizont predikcije T_1 . Navadno izberemo minimalni horizont predikcije $T_1 = 0$. Če ima sistem zakasnitev, ki je enaka T_d , tedaj je nesmiselno postaviti $T_1 < T_d$, saj v tem času ne moremo vplivati na izhodni signal. Če zakasnitev ni znana, potem lahko izberemo tak spodnji horizont predikcije T_1 , da bo enak največji možni zakasnitvi.

V primeru fazno-neminimalnih sistemov pa izberemo T_1 tako, da se izognemo fazno neminalnemu delu izhodnega signala.

Maksimalni horizont predikcije T_2 . Maksimalni horizont predikcije vpliva na čas izravnave izhodnega signala. Z izbiro večjega parametra T_2 dobimo počasnejše odzive izhodnega signala (večji čas izravnave T_{iz}). Z manjšim parametrom T_2 pa dosežemo večjo dinamiko izhodnega signala in s tem manjši čas izravnave.

V primeru modelno-referenčnega načina delovanja je izbran T_2 približno enak času izravnave modela.

Red predikcije N_y . Predikcija izhodnega signala je aproksimirana z Maclarenovo vrsto N_y -tega reda. V smislu zmanjševanja napake pri aproksimaciji predikcije izhodnega signala na časovnem intervalu $[T_1, T_2]$, je potrebno izbrati primeren red predikcije. Velikost parametra N_y je odvisna od časovnega intervala, v katerem računamo predikcijo. Red predikcije izberemo tako, da velja relacija

$$\sum_{i=N_y+1}^{\infty} y_i(t) \frac{T_2^i}{i!} < \epsilon, \quad (17.188)$$

kjer je ϵ poljubno majhno realno število. Ker višji odvodi $y^{(i)}(t)$ v splošnem niso znani, je ta ocena le teoretična. V praksi problem rešujemo interaktivno.

Ena od metod iskanja primerenega parametra N_y je s pomočjo odprtozančnega odziva na stopnico. Pri tem je potrebno izbrati tak N_y , da bo aproksimacija predikcije v območju $0 < T < T_2$ zadovoljiva.

Izbira parametra N_y je odvisna od območja predikcije in dinamike sistema. V primeru sistemov z zmerno dinamiko lahko za območje predikcije $0 < T < 5$ izberemo $N_y = 6$, kar praktično zadosti pogojem aproksimacije. V primeru kompleksnejše dinamike pa so vrednosti parametrov višje ($N_y = 10 \div 30$).

Red vodenja N_u . Z redom vodenja N_u vplivamo na predikcijo regulirnega signala $u_r^*(t, T)$. Vrednost $N_u = 0$, na primer, zagotavlja konstantno vrednost predikcije $u_r^*(t, T)$ v celotnem intervalu $T_1 < T < T_2$. Na osnovi enačbe 17.130 lahko ugotovimo, da je ta vrednost v primeru $N_u = 0$ enaka $u_r(t, T) = u(t)$ na celotnem predikcijskem intervalu. Medtem, ko je v primeru $N_u = 1$ predikcija $u_r^*(t, T) = u(t) + u_1(t)T$, kar predstavlja linearno naraščajočo funkcijo ($u_1(t) = \frac{du(t)}{dt}$) na intervalu predikcije. Omejevanje predikcije $u_r^*(t, T)$ pomeni posredno tudi omejevanje regulirnega signala $u(t)$. Rezultat tega je večja dejavnost regulirnega signala $u(t)$ pri večjih vrednostih parametra N_u in obratno. Ta pa direktno vpliva na dinamiko izhodnega signala $y(t)$.

Red vodenja pravzaprav določa stopnjo razvoja Maclarenove vrste pri predikciji signala $u(t + T)$. Iz tega lahko povzamemo, da s povečevanjem parametra dosežemo vrednost, kjer je aproksimacija zadovoljiva in nadaljnje spremjanje ne prinese bistvenih sprememb.

S simulacijo smo ugotovili, da je za veliko množico sistemov primerna izbira parametra $N_u = 0$.

Ugotovimo lahko, da se s povečevanjem parametra N_u poli zaprtozančnega sistema oddaljujejo od izhodišča koordinatnega sistema ravnine s .

Utežni faktor λ . Parameter λ je utežni faktor kriterijske funkcije, ki jo opisuje enačba 17.128. Z izbiro uteži določimo vpliv vhodnega signala na minimizacijo kriterijske funkcije.

- Osnovno načrtovanje regulatorja CGPC predpostavlja vrednost utežnega faktorja $\lambda = 0$. Ta izbira eliminira drugi integral v enačbi 17.128 in daje prednost sledenju izhodnega signala referenci (optimalna strategija zahteva minimalno ploščino, ki jo oklepata izhodni signal in referenca znotraj predikcijskega intervala) pred optimizacijo vhodnega signala. Pograšek v stacionarnem stanju je v tem primeru enak nič (tako načrtan regulator je v primeru proporcionalnih procesov PI tipa, v primeru integralskih procesov pa P tipa).
- Z izbiro uteži $\lambda > 0$ upoštevamo v kriterijski funkciji tudi prispevek vhodnega signala $u(t)$. Oceni dodamo še ploščino, ki jo vhodni signal $u(t)$ znotraj predikcijskega intervala oklepa z absciso. Z večanjem utežnega faktorja λ povečujemo vpliv vhodnega signala na kriterijsko funkcijo, na račun slabšega sledilnega delovnja. Značaj regulatorja je v tem primeru proporcionalen (regulator P).

17.11 Prediktivno funkcijski regulator (PFC)

V nadaljevanju si bomo ogledali več različnih tipov prediktivnega funkcijskega regulatorja (*ang. Predictive Functional Controller - PFC*). PFC je metodologija, ki se je močno uveljavila v

praksi zaradi enostavnosti in transparentnosti. Prediktivno funkcionalni regulator je vpeljal R. Chalet, [100]. Algoritem je doživel veliko uspešnih aplikacij v petrokemiji in podobni industriji [96, 99, 100]. Uspeh prediktivno funkcionalnega regulatorja lahko pripisemo njegovi enostavnosti, razumljivosti in dobremu delovanju. Algoritem je bil sprva razvit za linearne sisteme z enim vhodom in enim izhodom. Izkaže se, da je za preprost PFC algoritem možno dokazati, da je to dejansko regulator PI, oziroma regulator PI s Smithovim prediktorjem v primeru, ko upoštevamo mrtvi čas procesa [26], [27]. Kasneje je bil algoritem PFC razširjen za vodenje nelinearnih sistemov [18, 28] in multivariabilnih sistemov, [47, 92].

Za regulacijo nelinearnih sistemov je zelo priročen prediktivno funkcionalni regulator na osnovi mehkega modela (*ang. Fuzzy Predictive Functional Control - FPFC*), [115, 76]. Pristop temelji na ideji, da lahko nelinearen proces, z gladko in invertibilno nelinearnostjo, aproksimiramo z množico lokalnih linearnih modelov v obliki Takagi-Sugeno mehkega modela [111, 116].

Strategija PFC temelji na uporabi modela procesa za predikcijo izhoda procesa na določenem horizontu, [13]. Cilj pristopa je določiti tak regulirni signal, da se ujemata razlika med napovedjo referenčne vrednosti $y_r(k + H)$ in trenutno vrednostjo izhoda procesa $y_p(k)$ in razlika med napovedjo izhoda modela $y_m(k + H)$ in $y_m(k)$:

$$y_r(k + H) - y_p(k) = y_m(k + H) - y_m(k). \quad (17.189)$$

Horizont, kjer se načrtuje ujemanje, imenujemo horizont ujemanja H (*ang. coincidence horizon*).

V nadaljevanje bo zaradi enostavnosti prikazana izpeljava PFC algoritma na linearinem procesu prvega reda, ki je v zveznem prostoru podan s prenosno funkcijo

$$Y_m(s) = \frac{K_m}{T_m s + 1} U(s). \quad (17.190)$$

Model v enačbi 17.190 naprej zapišemo v diskretnem prostoru kot

$$y_m(k + 1) = a_m y_m(k) + b_m u(k), \quad (17.191)$$

kjer je $y_m(k)$ izhod modela, $u(k)$ pa vhod v model oziroma proces. Zveza med časovno konstanto in ojačenjem ter parametromi a_m in b_m je sledenča:

$$a_m = e^{-\frac{T_s}{T_m}}, \quad (17.192)$$

$$b_m = K_m(1 - a_m), \quad (17.193)$$

kjer je T_s čas vzorčenja.

Za izpeljavo regulacijskega zakona je potrebno podati še modelno-referenčno trajektorijo. Podana je z diferenčno enačbo 17.194, kar zagotavlja sledenje referenci $w(k)$

$$y_r(k + 1) = a_r y_r(k) + b_r w(k). \quad (17.194)$$

Referenčni model definira želeno obnašanje celotnega zaprtozančnega sistema. Da zagotovimo sledenje referenci, mora biti ojačenje tega modela enako ena. Na podlagi tega pogoja dobimo vrednost parametra b_r

$$b_r = (1 - a_r). \quad (17.195)$$

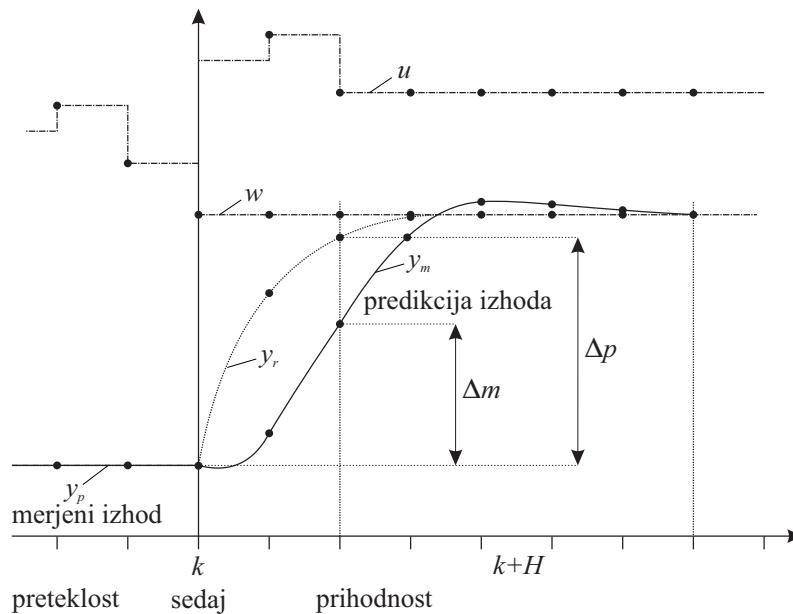
Eračbo 17.196, z upoštevanjem pogoja za ojačenje modela, zapišemo v naslednji obliki:

$$y_r(k + 1) = a_r y_r(k) + (1 - a_r) w(k). \quad (17.196)$$

Regulacijski zakon razvijemo na osnovi pogoja v enačbi 17.189. PFC predvideva horizont ujemanja H ($N_1 = N_2 = H$) in utežni faktor regulirnega signala $\lambda = 0$. V horizontu ujemanja naj bi izhod procesa sovpadal z referenčnim odzivom. To pomeni, da morata biti sprememba izhoda modela procesa in sprememba referenčnega modela od koraka k do koraka $k + H$ enaki, kar prikazuje slika 17.7.

$$\Delta_p = \Delta_m . \quad (17.197)$$

Spremembo referenčnega modela zapišemo kot



Slika 17.7. Princip regulatorja PFC.

$$\Delta_p = y_r(k + H) - y_r(k) , \quad (17.198)$$

spremembo modela procesa pa kot

$$\Delta_m = y_m(k + H) - y_m(k) . \quad (17.199)$$

Izhod modela procesa, oziroma referenčnega modela, pri koraku $k + H$ dobimo tako, da rekurzivno izračunamo predikcijo

$$\begin{aligned} y_m(k+1) &= a_m y_m(k) + b_m u(k) \\ y_m(k+2) &= a_m y_m(k+1) + b_m u(k+1) = \\ &= a_m^2 y_m(k) + a_m b_m u(k) + b_m u(k) \\ &\vdots \\ y_m(k+H) &= a_m y_m(k+H-1) + b_m u(k+H-1) = \\ &= a_m^H y_m(k) + a_m^{H-1} b_m u(k) + a_m^{H-2} b_m u(k+1) + \dots + b_m u(k+H-1) . \end{aligned} \quad (17.200)$$

Pri reševanju optimizacijskega problema predpostavimo konstanten prihodnji regulacijski signal (*ang. mean level control*). Ob predpostavki konstantnega prihodnjega regulacijskega signala

$u(k) = u(k+1) = \dots = u(k+H-1)$ lahko zapišemo predikcijo za $k+H$ -ti korak, iz enačbe 17.200, v naslednji obliki

$$y_m(k+H) = a_m^H y_m(k) + a_m^{H-1} b_m u(k) + a_m^{H-2} b_m u(k) + \dots + b_m u(k). \quad (17.201)$$

Če upoštevamo, da je

$$(1 + a_m + \dots + a_m^{H-2} + a_m^{H-1})(1 - a_m^H) = (1 - a_m^H),$$

se enačba 17.201 poenostavi v:

$$y_m(k+H) = a_m^H y_m(k) + \frac{b_m}{1 - a_m^H} (1 - a_m^H) u(k). \quad (17.202)$$

Podobno enačbo dobimo za predikcijo izhoda referenčnega modela, kjer smo upoštevali konstantno prihodnjo referenco ($w(k) = w(k+1) = \dots = w(k+H-1)$):

$$y_r(k+H) = a_r^H y_r(k) + (1 - a_r^H) w(k). \quad (17.203)$$

Kot vidimo, je vodenje na podlagi modela praktično odprtozančno vodenje procesa. Saj optimiramo izhod modela glede na referenčni model. Zaprta zanka je vključena v regulacijski algoritem posredno s tem, da so začetne vrednosti referenčnega modela enake trenutnim vrednostim procesa. Torej, da je izhod referenčnega modela v k -tem trenutku enak izhodu procesa ($y_r(k) = y_p(k)$). Spremembni Δ_p in Δ_m sedaj zapišemo kot:

$$\Delta_p = a_r^H y_r(k) + (1 - a_r^H) w(k) - y_p(k), \quad (17.204)$$

$$\Delta_m = a_m^H y_m(k) + \frac{b_m}{1 - a_m^H} (1 - a_m^H) u(k) - y_m(k). \quad (17.205)$$

Če sedaj rešimo enačbo 17.197 ob upoštevanju, da je $y_r(k) = y_p(k)$, dobimo naslednji regulirni zakon

$$u(k) = \frac{(1 - a_r^H)(w(k) - y_p(k))}{\frac{b_m}{1 - a_m^H}(1 - a_m^H)} + \frac{y_m(k)}{\frac{b_m}{1 - a_m^H}}. \quad (17.206)$$

Če definiramo dve novi ojačenji

$$K_m = \frac{b_m}{1 - a_m^H} \quad (17.207)$$

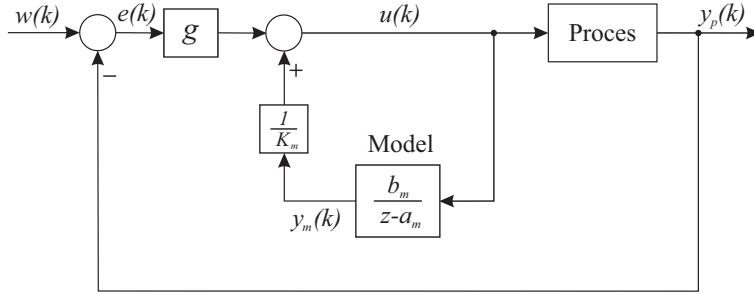
in

$$g = \frac{1 - a_r^H}{K_m(1 - a_m^H)}, \quad (17.208)$$

lahko enačbo 17.206 zapišemo v bolj pregledni obliki kot:

$$u(k) = g(w(k) - y_p(k)) + \frac{1}{K_m} y_m(k). \quad (17.209)$$

Regulacijski zakon iz enačbe 17.209 lahko ponazorimo z bločno shemo na sliki 17.8.



Slika 17.8. Bločna shema regulatorja PFC.

17.12 Regulator PFC z modelom prvega reda za sisteme z mrtvim časom

Regulacijski zakon, ki ga podaja enačba 17.209 je primeren za vodenje sistemov, ki jih lahko zadovoljivo dobro opišemo z modelom prvega reda brez zakasnitve. V praksi pa se pogosto pojavljajo procesi, ki imajo znaten mrtvi čas. Za regulacijo takšnih procesov je potrebno regulacijski zakon nadgraditi. Pri tem si pomagamo s konceptom Smithovega prediktorja. Pri regulaciji sistema z mrtvim časom, bi bilo najbolj idealno, če bi lahko sistem razdelili na dva dela. Prvi del bi bila prenosna funkcija brez zakasnitve, drugi del pa prenosna funkcija čiste zakasnitve, kot je prikazano na sliki 17.9. Nezakasnjen izhod procesa \tilde{y}_p bi peljali nazaj na vhod regulatorja. Regulacijski pogrešek $e(k)$ bi bil v tem primeru definiran kot:

$$e(k) = w(k) - \tilde{y}_p(k) . \quad (17.210)$$

Ker nezakasnjenega izhoda $\tilde{y}_p(k)$ ni mogoče izmeriti, si pomagamo z modelom procesa.

Zaradi odstopanj med modelom in procesom je nezakasnjeni izhod procesa \tilde{y}_p najlažje oceniti tako, da predpostavimo enakost razlike med izhodom in nezakasnjenim izhodom modela in razlike med izhodom in nezakasnjenim izhodom procesa. Na ta način delno odpravimo napako, ki jo povzroči odstopanje med modelom in procesom. Predpostavka velja ob dobrem modelu, še posebno je pomembna ocena zakasnitve modela D_m .

Enakost razlike med izhodom in nezakasnjenim izhodom modela in razlike med izhodom in nezakasnjenim izhodom procesa zapišemo kot:

$$y_p(k) - \tilde{y}_p(k) = y_m(k) - \tilde{y}_m(k) . \quad (17.211)$$

Ocenjena nezakasnjeni vrednost izhoda procesa je tako enaka

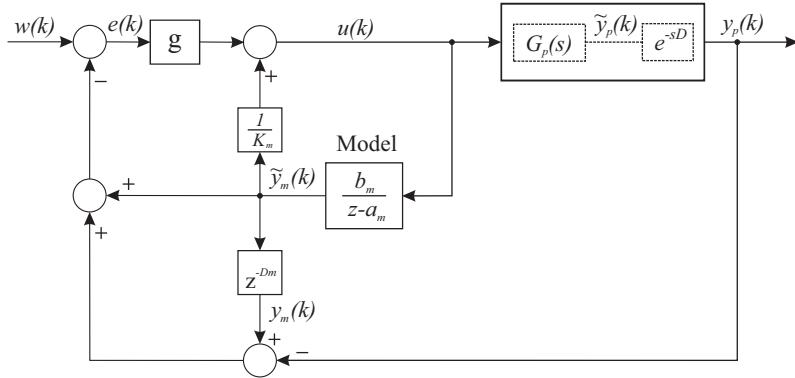
$$\tilde{y}_p(k) = y_p(k) - y_m(k) + \tilde{y}_m(k) . \quad (17.212)$$

Vodenje procesa z mrtvim časom lahko realiziramo tako, da ocenimo nezakasnjeni izhod procesa $\tilde{y}_p(k)$ in celotni regulacijski zakon iz enačbe 17.209 izvedemo na tem signalu. Regulacijski zakon pa se spremeni v naslednjo obliko:

$$u(k) = g (w(k) - \tilde{y}_p(k)) + \frac{1}{K_m} \tilde{y}_m(k) \quad (17.213)$$

ali

$$u(k) = g (w(k) - y_p(k) + y_m(k) - \tilde{y}_m(k)) + \frac{1}{K_m} \tilde{y}_m(k) . \quad (17.214)$$



Slika 17.9. Bločna shema regulatorja PFC z upoštevanjem zakasnitve.

17.13 Regulator PFC v prostoru stanj

Za vodenje procesov višjega reda in multivariabilnih procesov preprost regulator PFC z modelom prvega reda ni primeren. Za take primere razvijemo prediktivni regulator na osnovi modela v prostoru stanj. Izpeljava regulacijskega zakona je enaka kot pri regulatorju PFC z modelom prvega reda. Model procesa v prostoru stanj zapišemo:

$$\underline{x}_m(k) = \underline{A}_m \underline{x}_m(k) + \underline{B}_m \underline{u}(k) , \quad (17.215)$$

$$\underline{y}_m(k) = \underline{C}_m \underline{x}_m(k) . \quad (17.216)$$

V prostoru stanj je podan tudi referenčni model

$$\underline{x}_r(k) = \underline{A}_r \underline{x}_r(k) + \underline{B}_r \underline{w}(k) , \quad (17.217)$$

$$\underline{y}_r(k) = \underline{C}_r \underline{x}_r(k) , \quad (17.218)$$

ki mora imeti ojačenje enako ena. Zaradi tega morajo matrike \underline{A}_r , \underline{B}_r in \underline{C}_r izpolnjevati naslednji pogoj

$$\underline{C}_r(\underline{I} - \underline{A}_r)^{-1} \underline{B}_r = \underline{I} . \quad (17.219)$$

Napoved v časovnem trenutku k za H korakov v prihodnosti dobimo kot sledi

$$\underline{x}_m(k+H) = \underline{A}_m^H \underline{x}_m(k) + (\underline{A}_m^H - \underline{I})(\underline{A}_m - \underline{I})^{-1} \underline{B}_m \underline{u}(k) , \quad (17.220)$$

$$\underline{y}_m(k+H) = \underline{C}_m \underline{x}_m(k+H) . \quad (17.221)$$

Podobno zapišemo napoved za referenčno trajektorijo

$$\underline{x}_r(k+H) = \underline{A}_r^H \underline{x}_r(k) + (\underline{A}_r^H - \underline{I})(\underline{A}_r - \underline{I})^{-1} \underline{B}_r \underline{w}(k) , \quad (17.222)$$

$$\underline{y}_r(k+H) = \underline{C}_r \underline{x}_r(k+H) . \quad (17.223)$$

Rešimo enačbo 17.197, ki definira enakost

$$\Delta_m = \underline{C}_m (\underline{A}_m^H \underline{x}_m(k) + (\underline{A}_m^H - \underline{I})(\underline{A}_m - \underline{I})^{-1} \underline{B}_m \underline{u}(k)) - \underline{y}_m(k) \quad (17.224)$$

in

$$\Delta_p = \underline{C}_r (\underline{A}_r^H \underline{x}_r(k) + (\underline{A}_r^H - \underline{I})(\underline{A}_r - \underline{I})^{-1} \underline{B}_r \underline{w}(k)) - \underline{y}_p(k) . \quad (17.225)$$

Regulirni zakon dobimo v naslednji obliki

$$\underline{u}(k) = G_0^{-1}(\underline{y}_r(k+H) + \underline{y}_m(k) - \underline{y}_p(k) - \underline{C}_m \underline{A}_m^H \underline{x}_m(k)) , \quad (17.226)$$

kjer je

$$\underline{y}_r(k+H) = \underline{C}_r(\underline{A}_r^H \underline{x}_r(k) + (\underline{A}_r^H - \underline{I})(\underline{A}_r - \underline{I})^{-1} \underline{B}_r \underline{w}(k)) \quad (17.227)$$

in G_0

$$G_0 = \underline{C}_m(\underline{A}_m^H - \underline{I})(\underline{A}_m - \underline{I})^{-1} \underline{B}_m . \quad (17.228)$$

Če izberemo referenčni model tako, da je $\underline{C}_r = \underline{I}$ in $\underline{B}_r = (\underline{I} - \underline{A}_r)$, se regulacijski zakon (enačba 17.226) poenostavi. Predikcijo referenčnega signala na ta način lahko zapišemo kot

$$\underline{y}_r(k+H) = \underline{A}_r^H \underline{y}_r(k) + (\underline{I} - \underline{A}_r^H) \underline{w}(k) . \quad (17.229)$$

Rešimo enačbo 17.197 ob upoštevanju $\underline{y}_p(k) = \underline{y}_r(k)$, kjer je

$$\Delta_m = \underline{C}_m(\underline{A}_m^H \underline{x}_m(k) + (\underline{A}_m^H - \underline{I})(\underline{A}_m - \underline{I})^{-1} \underline{B}_m \underline{u}(k)) - \underline{y}_m(k) \quad (17.230)$$

in

$$\Delta_p = \underline{A}_r^H \underline{y}_r(k) + (\underline{I} - \underline{A}_r^H) \underline{w}(k) - \underline{y}_p(k) . \quad (17.231)$$

Dobimo poenostavljen regulacijski zakon

$$\underline{u}(k) = \underline{G}(\underline{w}(k) - \underline{y}_p(k)) + G_0^{-1} \underline{y}_m(k) - G_0^{-1} \underline{C}_m \underline{A}_m^H \underline{x}_m(k) , \quad (17.232)$$

kjer je

$$\underline{G} = G_0^{-1}(\underline{I} - \underline{A}_r^H) \quad (17.233)$$

in G_0

$$G_0 = \underline{C}_m(\underline{A}_m^H - \underline{I})(\underline{A}_m - \underline{I})^{-1} \underline{B}_m . \quad (17.234)$$

Kot vidimo iz enačbe 17.226 in 17.232 regulator obstaja le, če je vrednost izraza G_0 različna od nič. To pa velja za stabilne, vodljive in spoznavne procese.

17.14 Regulator PFC v prostoru stanj za sisteme z mrtvim časom

Podobno kot pri regulatorju PFC z modelom prvega reda, lahko tudi za regulator PFC v prostoru stanj uvedemo koncept Smithovega prediktorja za regulacijo sistemov z mrtvim časom. V primeru sistema z mrtvim časom $\underline{y}_p(k)$ predstavlja vektor merjenih izhodov procesa, ki pa je zaradi mrtvega časa zakasnjen. Za oceno nezakasnjenega izhoda procesa uporabimo koncept Smithovega prediktorja. Z upoštevanjem enačbe 17.212, za nezakasnjeno oceno izhoda procesa, se enačba 17.232 preoblikuje regulirni zakon

$$\underline{u}(k) = \underline{G}(\underline{w}(k) - \tilde{\underline{y}}_m(k) + \underline{y}_m(k) - \underline{y}_p(k)) + G_0^{-1} \underline{y}_m(k) - G_0^{-1} \underline{C}_m \underline{A}_m^H \underline{x}_m(k) , \quad (17.235)$$

kjer je

$$\underline{G} = G_0^{-1}(\underline{I} - \underline{A}_r^H) \quad (17.236)$$

in

$$G_0 = \underline{C}_m(\underline{A}_m^H - \underline{I})(\underline{A}_m - \underline{I})^{-1} \underline{B}_m . \quad (17.237)$$

17.15 Regulator PFC z neparametričnim modelom

Podobno kot smo v zgornjem poglavju izpeljali prediktivno funkcionalni regulacijski zakon za parametrične modele, to lahko storimo tudi za neparametrični model procesa. Model procesa lahko zapišemo na osnovi konvolucije

$$y(k) = \sum_{\tau=0}^{\infty} h(\tau)u(k-\tau), \quad (17.238)$$

kjer je $h(\tau)$ diskretna utežna funkcija procesa s prenosno funkcijo $H(z)$. Če uporabimo kot vhodni signal v proces enotin Diracov impulz

$$u(k) = \begin{cases} 1, & k = 0 \\ 0, & k \neq 0 \end{cases}, \quad (17.239)$$

je izhod procesa enak $h(k)$. Impulzni odziv procesa predstavlja neparametrični model procesa. Če poznamo impulzni odziv procesa $h(k)$, lahko odziv na stopnico $g(k)$ tega procesa preprosto izračunamo tako, da seštejemo vrednosti $h(k)$ do časovnega trenutka k

$$g(k) = \sum_{\tau=0}^k h(\tau).$$

Primer 17.15.1. Imamo stopnico, ki jo lahko zapišemo kot

$$u(k) = \begin{cases} 1, & k \geq 0 \\ 0, & k < 0 \end{cases}. \quad (17.240)$$

Če poskušamo izračunati konvolucijo za $k = 3$ dobimo

$$g(3) = \sum_{\tau=0}^n h(\tau)u(3-\tau) = h(0)u(3) + h(1)u(2) + h(2)u(1) + \dots + h(n)u(3-n). \quad (17.241)$$

Ker je vrednost stopnice za negativne argumente enaka 0, drugače pa enaka 1, dobimo

$$g(3) = h(0) + h(1) + h(2) + h(3). \quad (17.242)$$

V praksi je zelo težko pridobiti dober impulzni odziv procesa. V primeru, ko je pri meritvi prisoten šum, pa je impulzni odziv kot neparametrični model procesa praktično neuporaben. Zato se v praksi kot neparametrični model raje uporablja odziv na stopnico. Enačbo 17.238 v primeru odziva na enotino stopnico preoblikujemo tako, da impulzni odziv sistema $h(k)$ ustreznno nadomestimo z odzivom na enotino stopnico $g(k)$. Enačbo za odziv na enotino stopnico dobimo s seštevanjem vrednosti $h(k)$

$$g(k) = \sum_{\tau=0}^k h(\tau). \quad (17.243)$$

Zgornjo enačbo lahko zapišemo v naslednji obliki

$$g(k) = \sum_{\tau=0}^{k-1} h(\tau) + h(k) = g(k-1) + h(k) \quad (17.244)$$

in iz nje izrazimo $h(k)$

$$h(k) = g(k) - g(k-1). \quad (17.245)$$

S tem smo dobili relacijo med odzivom procesa na enotino stopnico in odzivom na enotin impulz. Z upoštevanjem tega se enačba 17.238 preoblikuje v naslednjo obliko

$$y(k) = \sum_{\tau=0}^{\infty} (g(\tau) - g(\tau-1)) u(k-\tau). \quad (17.246)$$

Če gornjo enačbo zapišemo tako, da v drugem členu spremenimo zapis za sumacijo, dobimo

$$y(k) = \sum_{\tau=0}^{\infty} g(\tau) u(k-\tau) - \sum_{\tau=-1}^{\infty} g(\tau) u(k-\tau-1). \quad (17.247)$$

Če upoštevamo, da je $g(-1) = 0$ in združimo člene z istim $g(\tau)$, pridemo do naslednje enačbe

$$y(k) = \sum_{\tau=0}^{\infty} g(\tau) \Delta u(k-\tau), \quad (17.248)$$

kjer je $\Delta u(k-\tau) = u(k-\tau) - u(k-\tau-1)$ in $g(\tau)$ vrednost odziva procesa na enotino stopnico v časovnem trenutku τ . Prišli smo do konvolucijske enačbe, ki opisuje obnašanje procesa in jo bomo naprej uporabljali pri razvoju regulatorja.

Enako kot pri osnovnem algoritmu PFC, tudi v primeru konvolucijskega pristopa najprej definiramo inkrement model Δ_m

$$\Delta_m = y_m(k+H) - y_m(k). \quad (17.249)$$

Izhod modela $y_m(k)$ zapišemo s konvolucijo kot

$$y_m(k) = \sum_{\tau=0}^{\infty} g(\tau) \Delta u(k-\tau), \quad (17.250)$$

napoved model pri horizontu ujemanja H pa je enaka

$$y_m(k+H) = \sum_{\tau=0}^{\infty} g(\tau) \Delta u(k+H-\tau). \quad (17.251)$$

Izraz za $y_m(k+H)$ lahko razbijemo na dva dela

$$y_m(k+H) = \sum_{\tau=0}^H g(\tau) \Delta u(k+H-\tau) + \sum_{\tau=H+1}^{\infty} g(\tau) \Delta u(k+H-\tau). \quad (17.252)$$

Če pri drugem členu začnemo seštevanje od ena naprej, se izraz spremeni v

$$y_m(k+H) = \sum_{\tau=0}^H g(\tau) \Delta u(k+H-\tau) + \sum_{\tau=1}^{\infty} g(\tau+H) \Delta u(k-\tau). \quad (17.253)$$

Inkrement modela Δ_m lahko z upoštevanjem enačbe 17.253 in 17.250 zapišemo kot

$$\Delta_m = \sum_{\tau=0}^H g(\tau) \Delta u(k+H-\tau) + \sum_{\tau=1}^{\infty} (g(\tau+H) - g(\tau)) \Delta u(k-\tau), \quad (17.254)$$

ker lahko sumacijo pri konvoluciji za $y_m(k)$ začnemo pri $\tau = 1$ zaradi $g(0) = 0$. V primeru asimptotično stabilnih procesov se odziv za vrednosti $\tau > N_1$ ustali. To pomeni, da je razlika $g(\tau+H) - g(\tau)$ za $\tau > N_1$ približno enaka nič. Zato nam zadnjega dela izraza ni potrebno seštevati do neskončnosti, ampak samo do $\tau = N_1$. Enačba 17.254 se lahko preoblikuje v:

$$\Delta_m = \sum_{\tau=0}^H g(\tau) \Delta u(k+H-\tau) + \sum_{\tau=1}^{N_1} (g(\tau+H) - g(\tau)) \Delta u(k-\tau). \quad (17.255)$$

Če upoštevamo konstanten prihodnji regulirni signal ($u(k) = u(k+1) = \dots = u(k+H)$) bodo difference vhodnega signala $\Delta u(k+H-\tau)$ enake nič, razen difference $\Delta u(k)$. Zato lahko izraz v enačbi 17.255 poenostavimo v obliko

$$\Delta_m = g(H) \Delta u(k) + \sum_{\tau=1}^{N_1} (g(\tau+H) - g(\tau)) \Delta u(k-\tau). \quad (17.256)$$

Za izpeljavo regulirnega zakona moramo rešiti še enačbo $\Delta_p = \Delta_m$, ki jo lahko zapišemo v naslednji obliki:

$$a_r^H y_p(k) + (1 - a_r^H) w(k) - y_p(k) = g(H) \Delta u(k) + \sum_{\tau=1}^{N_1} (g(\tau+H) - g(\tau)) \Delta u(k-\tau). \quad (17.257)$$

Dobimo regulacijski zakon regulatorja PFC z neparametričnim modelom:

$$\Delta u(k) = \frac{(1 - a_r^H)}{g(H)} (w(k) - y_p(k)) - \frac{1}{g(H)} \sum_{\tau=1}^{N_1} (g(\tau+H) - g(\tau)) \Delta u(k-\tau). \quad (17.258)$$

Regulacijski zakon v enačbi 17.258 podaja spremembo regulacijskega signala. Regulirni signal dobimo tako, da to spremembo prištejemo k prejšnji vrednosti regulirnega signala

$$u(k) = u(k-1) + \Delta u(k). \quad (17.259)$$

17.16 Povzetek

V tem poglavju smo predstavili osnovni koncept prediktivnega vodenja, ki določa regulacijsko akcijo z optimizacijo kriterijske funkcije, pri čemer po principu pomicnega horizonta uporablja napoved regulirane veličine za določen horizont v prihodnost na osnovi modela procesa. Sistematično smo obdelali vse posamezne elemente prediktivnega vodenja: modele procesov, modelno-referenčno trajektorijo, strukturiranje regulacijskega zakona in algoritmiziran izračun regulirnega signala. Osnovni koncept smo obdelali splošno, v nadaljevanju pa smo obravnavali nekaj osnovnih metod prediktivnega vodenja. Opisali smo dva najbolj znana predstavnika: regulator DMC, ki temelji na neparametričnem modelu v obliki odziva na stopnico, regulator GPC, ki temelji na modelu v obliki prenosne funkcije, regulator CGPC, ki je primer prediktivnega regulatorja v zveznem časovnem prostoru in več modifikacij njenostavnejšega prediktivnega regulatorja, regulatorja PFC. Ta je zaradi svoje enostavnosti in učinkovitosti zelo primeren tudi za vodenje nelinearnih procesov na osnovi lokalnih linearnih modelov.

A. Vektorji in matrike

V tem dodatko so zbrane osnove odvajanja vektorjev in matrik.

A.1 Odvodi vektorjev in matrik

Odvod m -dimenzionalnega stolpnega vektorja $\underline{x} = [x_1 \ x_2 \ \dots \ x_m]^T$ glede na skalar θ je enak

$$\frac{\partial \underline{x}}{\partial \theta} = \begin{bmatrix} \frac{\partial x_1}{\partial \theta} \\ \frac{\partial x_2}{\partial \theta} \\ \vdots \\ \frac{\partial x_m}{\partial \theta} \end{bmatrix}. \quad (\text{A.1})$$

Odvod matrike \underline{X} dimenzije $l \times m$ glede na skalar θ pa je definiran kot

$$\frac{\partial \underline{X}}{\partial \theta} = \begin{bmatrix} \frac{\partial X_{11}}{\partial \theta} & \frac{\partial X_{12}}{\partial \theta} & \dots & \frac{\partial X_{1m}}{\partial \theta} \\ \frac{\partial X_{21}}{\partial \theta} & \frac{\partial X_{22}}{\partial \theta} & \dots & \frac{\partial X_{2m}}{\partial \theta} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial X_{l1}}{\partial \theta} & \frac{\partial X_{l2}}{\partial \theta} & \dots & \frac{\partial X_{lm}}{\partial \theta} \end{bmatrix}. \quad (\text{A.2})$$

Kadar imamo odvod glede na n -dimenzionalni vektor $\underline{\theta} = [\theta_1 \ \theta_2 \ \dots \ \theta_n]^T$, takrat pa je najbolje, če najprej definiramo operator odvajanja:

$$\frac{\partial}{\partial \underline{\theta}} = \begin{bmatrix} \frac{\partial}{\partial \theta_1} \\ \frac{\partial}{\partial \theta_2} \\ \vdots \\ \frac{\partial}{\partial \theta_n} \end{bmatrix}. \quad (\text{A.3})$$

Z uporabo (A.3) je odvod skalarja x glede na n -dimenzionalne vektorje $\underline{\theta}$ definiran kot

$$\frac{\partial x}{\partial \underline{\theta}} = \begin{bmatrix} \frac{\partial x}{\partial \theta_1} \\ \frac{\partial x}{\partial \theta_2} \\ \vdots \\ \frac{\partial x}{\partial \theta_n} \end{bmatrix}. \quad (\text{A.4})$$

Odvod m -dimenzionalnega vektorja \underline{x}^T glede na n -dimenzionalni stoljni vektor $\underline{\theta}$ je definiran z zunanjim produktom in je enak

$$\frac{\partial \underline{x}^T}{\partial \underline{\theta}} = \frac{\partial}{\partial \underline{\theta}} \cdot \underline{x}^T = \begin{bmatrix} \frac{\partial x_1}{\partial \theta_1} & \frac{\partial x_2}{\partial \theta_1} & \dots & \frac{\partial x_m}{\partial \theta_1} \\ \frac{\partial x_1}{\partial \theta_2} & \frac{\partial x_2}{\partial \theta_2} & \dots & \frac{\partial x_m}{\partial \theta_2} \\ \vdots & \vdots & & \vdots \\ \frac{\partial x_1}{\partial \theta_n} & \frac{\partial x_2}{\partial \theta_n} & \dots & \frac{\partial x_m}{\partial \theta_n} \end{bmatrix}. \quad (\text{A.5})$$

Vektorski odvod skalarnega produkta je enak:

$$\frac{\partial}{\partial \underline{\theta}} (\underline{x}^T \underline{y}) = \frac{\partial \underline{x}^T}{\partial \underline{\theta}} \cdot \underline{y} + \frac{\partial \underline{y}^T}{\partial \underline{\theta}} \cdot \underline{x}, \quad (\text{A.6})$$

kjer je \underline{x}^T m -dimenzionalni vrstni vektor, \underline{y} je m -dimenzionalni stolpni vektor in $\underline{\theta}$ je n -dimenzionalni stolpni vektor.

Če eden od vektorjev v (A.6) ni odvisen od parametra $\underline{\theta}$ drugi pa je odvisen od $\underline{\theta}$, dobimo:

$$\frac{\partial}{\partial \underline{\theta}} (\underline{z}^T \underline{\theta}) = \underline{z} \quad (\text{A.7})$$

in

$$\frac{\partial}{\partial \underline{\theta}} (\underline{\theta}^T \underline{z}) = \underline{z}, \quad (\text{A.8})$$

kjer vrstice in stolpci v vektorjih \underline{z}^T in \underline{z} niso odvisni od vektorja $\underline{\theta}$. Ustrezen zapis sledi, če vektor \underline{z} zamenjamo z matriko \underline{Z} :

$$\frac{\partial}{\partial \underline{\theta}} (\underline{Z}^T \underline{\theta}) = \underline{Z}, \quad (\text{A.9})$$

$$\frac{\partial}{\partial \underline{\theta}} (\underline{\theta}^T \underline{Z}) = \underline{Z}. \quad (\text{A.10})$$

Na koncu si poglejmo še odvod kvadratične form $\underline{\theta}^T \underline{Z} \underline{\theta}$, ki je zelo pogost in pomemben izraz, kjer je matrika \underline{Z} dimenzijs $n \times n$ matrix:

$$\begin{aligned} \frac{\partial}{\partial \underline{\theta}} \left(\underbrace{\underline{\theta}^T \underline{Z}}_{\underline{x}^T} \underbrace{\underline{\theta}}_{\underline{y}} \right) &= \frac{\partial}{\partial \underline{\theta}} (\underline{\theta}^T \underline{Z}) \cdot \underline{\theta} + \frac{\partial}{\partial \underline{\theta}} (\underline{\theta}^T) \cdot (\underline{\theta}^T \underline{Z})^T \\ &= \underline{Z} \underline{\theta} + \underline{I} \underline{Z}^T \underline{\theta} = (\underline{Z} + \underline{Z}^T) \underline{\theta}. \end{aligned} \quad (\text{A.11})$$

V prvem koraku je uporabljena (A.6), tako kot sledi. V drugem koraku je $\partial \underline{\theta}^T / \partial \underline{\theta}$, kar je enako enotski matriki \underline{I} glede na (A.5). Zadnji zapis v (A.11) pa lahko poenostavimo v $2\underline{Z} \underline{\theta}$, če je \underline{Z} simetrična matrika.

A.2 Gradient, Hessova in Jacobijeva matrika

Gradient *gradient* \underline{g} je prvi odvod skalarne funkcije glede na vektor. Gradient funkcije $f(\underline{\theta})$ je odvisen od n -dimenzionalnega vektorja parametrov $\underline{\theta} = [\theta_1 \ \theta_2 \ \dots \ \theta_n]^T$. Glede na te parametre ga lahko zapišemo kot n -dimenzionalni vektor:

$$\underline{g} = \frac{\partial f(\underline{\theta})}{\partial \underline{\theta}} = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_n \end{bmatrix} = \begin{bmatrix} \partial f(\underline{\theta}) / \partial \theta_1 \\ \partial f(\underline{\theta}) / \partial \theta_2 \\ \vdots \\ \partial f(\underline{\theta}) / \partial \theta_n \end{bmatrix}. \quad (\text{A.12})$$

Geometrijsko gledano je gradient smer največje strmine funkcije $f(\underline{\theta})$. Stacionarne točke (lokal-nem ali globalnem) $f(\underline{\theta})$ so definirane tako, da je v teh točkah gradient funkcije enak nič.

Hessova matrika \underline{H} je drugi odvod skalarne funkcije glede na vektor parametrov. Hessova matrika skalarne funkcije $f(\underline{\theta})$ je odvisna od n -dimenzionalnega vektorja parametrov $\underline{\theta} = [\theta_1 \ \theta_2 \ \dots \ \theta_n]^T$ in je definirana kot $n \times n$ -dimenzionalna matrika:

$$\begin{aligned}\underline{H} &= \frac{\partial^2 f(\underline{\theta})}{\partial \underline{\theta}^2} = \begin{bmatrix} H_{11} & H_{12} & \cdots & H_{1n} \\ H_{21} & H_{22} & \cdots & H_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ H_{n1} & H_{n2} & \cdots & H_{nn} \end{bmatrix} \\ &= \begin{bmatrix} \frac{\partial^2 f(\underline{\theta})}{\partial \theta_1^2} & \frac{\partial^2 f(\underline{\theta})}{\partial \theta_1 \partial \theta_2} & \cdots & \frac{\partial^2 f(\underline{\theta})}{\partial \theta_1 \partial \theta_n} \\ \frac{\partial^2 f(\underline{\theta})}{\partial \theta_2 \partial \theta_1} & \frac{\partial^2 f(\underline{\theta})}{\partial \theta_2^2} & \cdots & \frac{\partial^2 f(\underline{\theta})}{\partial \theta_2 \partial \theta_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(\underline{\theta})}{\partial \theta_n \partial \theta_1} & \frac{\partial^2 f(\underline{\theta})}{\partial \theta_n \partial \theta_2} & \cdots & \frac{\partial^2 f(\underline{\theta})}{\partial \theta_n^2} \end{bmatrix}. \quad (\text{A.13a})\end{aligned}$$

Geometrijsko gledano Hessova matrika predstavlja ukrivljenost (konveksnost ali konkavnost) funkcije $f(\underline{\theta})$, t.j., za linearno funkcijo $f(\underline{\theta})$ je Hessova matrika enaka nič, za kvadratično funkcijo $f(\underline{\theta})$ pa je Hessova matrika konstantna. Hessova matrika je simetrična. Če je pozitivno definitna (konveksna), potem je stacionarna točka funkcije minimum, če pa je negativno definitna (konkavna) je stacionarna točka maksimum.

Jakobijeva matrika je prvi odvod vektorske funkcije glede na vektor. Jakobijeva matrika N -dimenzionalne funkcije $\underline{f}(\underline{\theta}) = [f_1(\underline{\theta}) \ f_2(\underline{\theta}) \ \cdots \ f_N(\underline{\theta})]^T$ je odvisna od n -dimenzionalnega vektorja parametrov $\underline{\theta} = [\theta_1 \ \theta_2 \ \dots \ \theta_n]^T$ na naslednji način tako, da dobimo $N \times n$ -dimenzionalno matriko:

$$\begin{aligned}\underline{J} &= \frac{\partial \underline{f}(\underline{\theta})}{\partial \underline{\theta}} = \begin{bmatrix} (\partial f_1(\underline{\theta}) / \partial \underline{\theta})^T \\ (\partial f_2(\underline{\theta}) / \partial \underline{\theta})^T \\ \vdots \\ (\partial f_N(\underline{\theta}) / \partial \underline{\theta})^T \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} & \cdots & J_{1n} \\ J_{21} & J_{22} & \cdots & J_{2n} \\ \vdots & \vdots & & \vdots \\ J_{N1} & J_{N2} & \cdots & J_{Nn} \end{bmatrix} \\ &= \begin{bmatrix} \frac{\partial f_1(\underline{\theta})}{\partial \theta_1} & \frac{\partial f_1(\underline{\theta})}{\partial \theta_2} & \cdots & \frac{\partial f_1(\underline{\theta})}{\partial \theta_n} \\ \frac{\partial f_2(\underline{\theta})}{\partial \theta_1} & \frac{\partial f_2(\underline{\theta})}{\partial \theta_2} & \cdots & \frac{\partial f_2(\underline{\theta})}{\partial \theta_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_N(\underline{\theta})}{\partial \theta_1} & \frac{\partial f_N(\underline{\theta})}{\partial \theta_2} & \cdots & \frac{\partial f_N(\underline{\theta})}{\partial \theta_n} \end{bmatrix}. \quad (\text{A.14a})\end{aligned}$$

Jakobijevu matriko uporabimo, na primer pri izračunu gradijenta vektorja napake po posameznih elementih $\underline{e} = [e_1 \ e_2 \ \cdots \ e_N]^T$, kjer je N število meritev. Jakobijeva matrika vsebuje transponirane gradijente posameznih elementov vektorja \underline{e} glede na vektor parametrov $\underline{\theta}$, t.j.,

$$J = [\underline{g}_1 \ \underline{g}_2 \ \cdots \ \underline{g}_N]^T$$

kjer je \underline{g}_i gradijet e_i .

B. Mehka logika

V dodatku o mehki logiki so predstavljeni osnovni pojmi mehke logike. Definirana je mehka množica, kot osnovni gradnik, opredeljene so njene lastnosti, predstavljeni so operatorji nad mehkimi množicami in mehko sklepanju.

B.1 Mehke množice

Teorija mehkih množic se je pojavila v sedemdesetih letih, čeprav so se podobne ideje pojavljale že prej. Pojavi se kot refleksija na eksaktno klasično logiko, ki v določenih pogledih ne zadostuje. Najprej bomo predstavili osnovno definicijo mehkih množic in osnovne lastnosti mehkih množic.

B.1.1 Definicija mehke množice

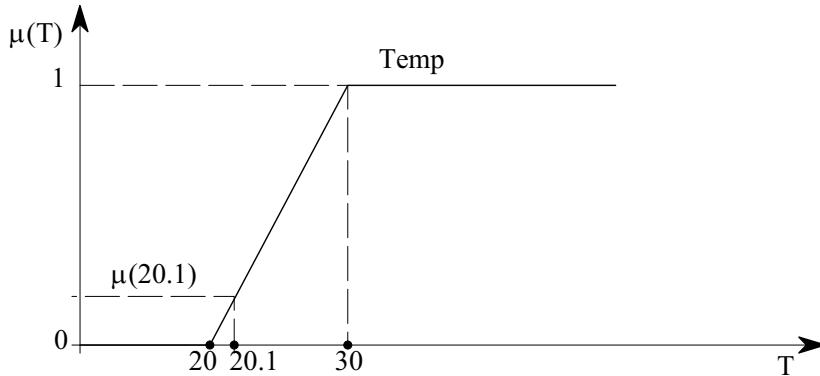
Klasična teorija množic definira pripadnost $\mu_A(x)$ elementa x določeni klasični množici A , ki je podmnožica univerzalne množice X , na naslednji način

$$\mu_A(x) = \begin{cases} 1, & x \in A \\ 0, & x \notin A \end{cases}. \quad (\text{B.1})$$

Glede na gornjo definicijo klasične množice lahko povzamemo, da je element x ali element množice A ($\mu_A(x) = 1$) ali pa ni element te množice ($\mu_A(x) = 0$). Klasične množice imenujemo tudi ostre množice. Način ostre predstavitve določenega problema pa mnogokrat ni v skladu z naravo problema. V naravi so predvsem zelo pomembni zvezni prehodi med različnimi stanji ali v naši terminologiji, med različnimi pripadnostmi določeni množici.

Za primer si oglejmo problem sobne temperature. Naj bo množica A , množica v kateri so vrednosti sobne temperature, ki so za bivanje prijetne. V tem primeru je jasno, da je zelo težko potegniti določeno mejo, kjer bi ločili temperaturo, ki je za bivanje prijetna in tisto, ki je neprijetna. Z veliko verjetnostjo lahko trdimo, da so sobne temperature, ki so višje od $30^\circ C$, neprijetne za večino ljudi. Kot prijetno sobno temperaturo pa lahko označimo temperaturo $20^\circ C$. Z izbiro temperatur za katere lahko z veliko verjetnostjo rečemo, da bodo imele predvidljiv vpliv na počutje ljudi, smo določili meje našega problema. Potrebno je poudariti, da je določevanje domene in mejnih vrednosti zelo subjektivno.

Kakšen vpliv na človeka pa imajo temperature, ki so v neposredni okolici mejnih vrednosti? Izberimo naprimer vrednost temperature $20.1^\circ C$. V tem primeru je temperatura v neposredni okolici temperature, ki je primera, zato lahko trdimo, da spremembra temprature za $0.1^\circ C$ ne more bistveno vplivati na pripadnost določeni množici. V primeru klasične teorije množic se pripadnost določeni množici spremeni ostro, kar pa mnogokrat ne ustrezata naravi problema. Pripadnost mehki množici, ki označuje neprimernost sobne temperature pa se povečuje od vrednosti npr. 0 pri $20^\circ C$ do pripadnosti 1 pri temperaturi $30^\circ C$, kjer lahko z gotovostjo trdimo, da je temperatura neprimerna. Slika prikazuje mehko množico neprimerne temperature.



Slika B.1. Mehka množica za neprimerno temperaturo.

Mehka množica je torej množica elementov, ki z določeno pripadnostjo pripadajo množici. Pripadnost množici izrazimo z realno mero na intervalu $[0, 1]$.

Definicija mehke množice. Mehka množica A je množica elementov, ki pripadajo tej množici s pripadnostmi, ki so vrednosti na realnem intervalu $\mu_A(x) \in [0, 1]$. Mehko množico A lahko zapišemo v obliki vsote

$$A = \sum_{i=1}^m \frac{\mu_A(x_i)}{x_i}, \quad (\text{B.2})$$

kjer je $\mu_A(x)$ **pripadnostna funkcija** in X je domena spremenljivke. Zapis vsote je simboličen in pomeni, da množico sestavljajo elementi z njihovimi pripadnostmi. Zapis z ulomkom pomeni, da ustreza elementu x_i pripadnost $\mu_A(x_i)$. Za primer si oglejmo zapis mehke množice neprimerne temperature $Temp$, ki je predstavljena na sliki B.1

$$Temp = \left(\frac{0}{20}, \frac{0.5}{25}, \frac{1}{30}, \frac{1}{40} \right). \quad (\text{B.3})$$

Naslednji od možnih zapisov je klasičen zapis mehke množice

$$A = \{x \in X, \mu_A(x) \in (0, 1]\}. \quad (\text{B.4})$$

Mehke množice z odstopanjem. Mehke množice z odstopnjem so definirane kot mehke množice, ki jim dodamo dopolnilo. Če definiramo mehko množico $A = \text{primerna temperatura}$, lahko tej izjavi dodamo dopolnilo **zelo**, kar nam da novo izjavo $A_h = \text{zelo primerna temperatura}$.

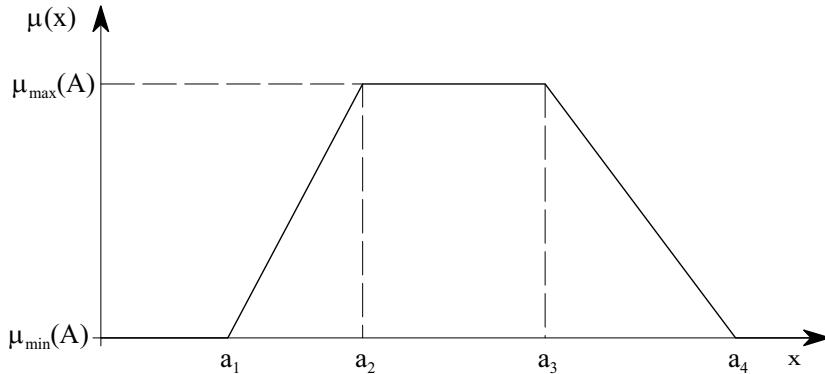
B.1.2 Lastnosti mehkih množic

Pripadnostne funkcije mehkih množic, ki jih največkrat srečujemo so: trikotniške, trapezne in zvončaste oblike. Na sliki B.2 je predstavljena trapezna oblika pripadnostne funkcije mehke množice in njeni parametri s katerimi jo definiramo.

Zapis določene oblike lahko enoumno definiramo z izbiro določenega števila parametrov. V primeru trapezne oblike pripadnostne funkcije mehke množice A , ki jo prikazuje slika B.2, lahko množico enoumno definiramo s šestimi parametri

$$A = \{a_1, a_2, a_3, a_4, \mu_{\min}(A), \mu_{\max}(A)\}. \quad (\text{B.5})$$

Vrednost $\mu_{\max}(A)$ predstavlja zgornji nivo pripadnosti mehki množici A , ki je definirana kot



Slika B.2. Trapezna pripadnostna funkcija.

$$\mu_{\max}(A) = \max_{x \in X} \mu_A(x) . \quad (\text{B.6})$$

Vrednost $\mu_{\min}(A)$ pa predstavlja spodnji nivo pripadnosti mehki množici A , ki je definiran kot

$$\mu_{\min}(A) = \min_{x \in X} \mu_A(x) . \quad (\text{B.7})$$

Vrednosti spodnjega nivoja in zgornjega nivoja pripadnosti morata ustreznati pogoju

$$0 \leq \mu_{\min}(A), \quad \mu_{\max}(A) \leq 1 . \quad (\text{B.8})$$

Če je vrednost $\mu_{\min}(A) = 0$ in $\mu_{\max}(A) = 1$, je taka množica *normalna mehka množica*, drugače pa jo imenujemo *subnormalna mehka množica*. V nadaljevanju se bomo omejili na normalne mehke množice.

Nosilec (*ang. support*) mehke pripadnostne funkcije je interval, ki je definiran kot podmnožica množici X

$$\text{supp}(A) = \{x \in X \mid \mu_A(x) > 0\} . \quad (\text{B.9})$$

Kar pomeni, da je nosilec mehke množice podmnožica, kjer je pripadnostna funkcija večja od nič. Nosilec mehke množice na sliki B.2 je enak intervalu

$$\text{supp}(A) = [a_1, a_4] . \quad (\text{B.10})$$

V primeru normalne mehke pripadnosti lahko definiramo jedro (*ang. kernel*), ki ga na splošno definiramo kot

$$\text{kern}(A) = \{x \in X \mid \mu_A(x) = 1\} . \quad (\text{B.11})$$

Jedro mehke množice je interval, kjer je pripadnostna funkcija mehke množice enaka 1. V primeru na sliki B.2 je jedro enako

$$\text{kern}(A) = [a_2, a_3] . \quad (\text{B.12})$$

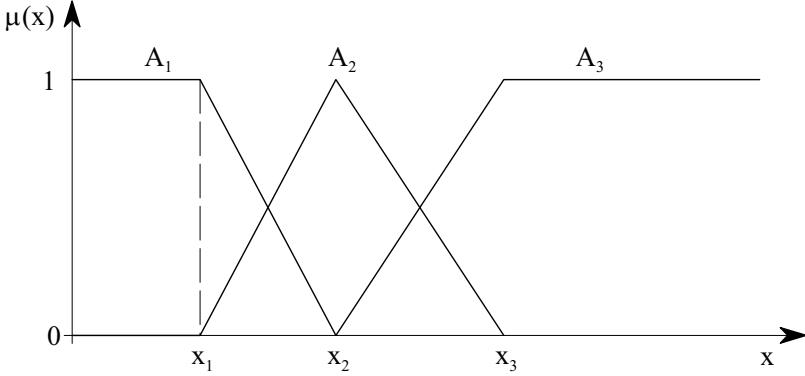
Definirajmo še α_{cut} množico, ki jo zapišemo kot

$$\alpha_{cut} = \{x \in X \mid \mu_A(x) > \alpha\} . \quad (\text{B.13})$$

V primeru uporabe mehke logike v vodenju celotno domeno določene spremenljivke razdelimo na več mehkih množic. Tak nabor mehkih množic tvori *mehko spremenljivko* ali *mehko particijo*. Predpostavimo, da imamo m mehkih množic na domeni X . Nabor mehkih množic $A_m(X) = (A_1, \dots, A_j, \dots, A_m)$ mora zadostovati pogoju

$$\forall x \in X, \sum_{j=1}^m \mu_{A_j} = 1 , \quad (\text{B.14})$$

pri čemer velja, da je $A_j \neq 0$ in $A_j \neq X$. Pogoj dovoljuje, da mehka spremenljivka, ki je sestavljena iz konveksnih in normalnih mehkih množic vsebuje največ dvakratna prekrivanja mehkih množic. Primer mehke particije ali mehke spremenljivke je na sliki B.3.



Slika B.3. Primer mehke particije ali mehke spremenljivke.

Pomembna lastnost mehkih množic je tudi *mera mehkosti*. Mero mehkosti definiramo, kot preslikavo

$$f : A_m(X) \rightarrow \mathcal{R} . \quad (\text{B.15})$$

Pri tem je $A_m(X)$ množica vseh mehkih množic na domeni X ali mehka particija in \mathcal{R} množica pozitivnih realnih števil. Funkcija f priredi vrednost $f(A)$ za vsako množico A prostora X . Če je mera mehkost enaka $f(A) = 0$, potem je to klasična množica. Med množicami lahko uvedemo relacijo večje in manjše ostrine. Mehka množica A_i je mehkejša od množice A_j , če velja $f(A_i) \leq f(A_j)$. Gornje lahko zapišemo matematično v naslednji obliki

$$f(A_i) \leq f(A_j) \rightarrow A_i \angle A_j . \quad (\text{B.16})$$

Mehka množica je maksimalno mehka v primeru, ko je stopnja pripadnosti enaka 0.5 na celotnem intervalu $x \in X$.

Mera mehkosti je lahko podana na več načinov. Lahko jo podamo z enačbo

$$f(A) = - \sum_{x \in X} (f_A(x) \log_2 f_A(x) + (1 - f_A(x)) \log_2 (1 - f_A(x))) . \quad (\text{B.17})$$

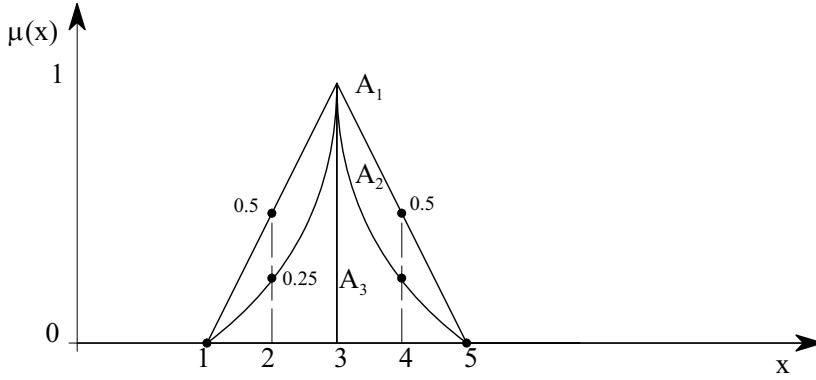
Funkcijo normaliziramo tako, da jo spravimo v okvir področja $0 \leq f(A) \leq 1$. V tem primeru je potrebno izraz B.17 deliti s številom elementov v prostoru X . Za primer izračunajmo mero mehkosti po definiciji iz enačbe B.17 za mehke množice, ki so na sliki B.4 in jih lahko podamo na naslednji način:

$$A_1 = \left(\frac{0}{1}, \frac{0.5}{2}, \frac{1}{3}, \frac{0.5}{4}, \frac{0}{5} \right), \quad A_2 = \left(\frac{0}{1}, \frac{0.25}{2}, \frac{1}{3}, \frac{0.25}{4}, \frac{0}{5} \right), \quad A_3 = \left(\frac{0}{1}, \frac{0}{2}, \frac{1}{3}, \frac{0}{4}, \frac{0}{5} \right) . \quad (\text{B.18})$$

Mere mehkosti izračunane po definiciji B.17 imajo naslednje vrednosti

$$f(A_1) = -2(0.5 \log_2 0.5 + 0.5 \log_2 0.5) = 2 , \quad (\text{B.19})$$

$$f(A_2) = -2(0.25 \log_2 0.25 + 0.75 \log_2 0.75) = -2(0.25(-2) + 0.75 \log_2 0.75) \geq 2 . \quad (\text{B.20})$$



Slika B.4. Primerjava mehkosti različnih mehkih množic.

Glede na definicijo v enačbi B.16 ugotovimo, da je mehka množica A_1 mehkejša od množice A_2 .

Druga oblika mere mehkosti je mehko razdalja Minkowskega, ki je podana z naslednjo enačbo

$$f(A_i) = \left(\sum_{x \in X} |f_{A_i}(x) - f_{A_j}(x)|^w \right)^{\frac{1}{w}}, \quad (\text{B.21})$$

kjer je A_j najbližja mehka množica. Za splošno mehko razdaljo Minkowskega velja, da je parameter $w \in [0, 1]$.

Za mehke množice na sliki B.4 izračunajmo še mehko razdaljo Minkowskega pri vrednosti parametra $w = 1$. Dobimo naslednje vrednosti mere mehkosti

$$\begin{aligned} f(A_1) &= \sum_{x \in X} |f_{A_1}(x) - f_{A_2}(x)|^1 = (0 + 0.25 + 0 + 0.25 + 0) = 0.5, \\ f(A_2) &= \sum_{x \in X} |f_{A_2}(x) - f_{A_3}(x)|^1 = (0 + 0.25 + 0 + 0.25 + 0) = 0.5, \\ f(A_3) &= \sum_{x \in X} |f_{A_3}(x) - f_{A_1}(x)|^1 = (0 + 0.5 + 0 + 0.5 + 0) = 1. \end{aligned} \quad (\text{B.22})$$

Iz gornjih vrednosti je razvidno, da sta mehki množici A_1 in A_2 mehkejši od množice A_3 .

Predstavitev mehke množice. Mehko množico lahko predstavimo na več različnih načinov. Najpogosteje srečujemo naslednji dve predstavitvi:

- funkcionalna predstavitev mehke množice (*ang. function presentation*), ki temelji na predstavitvi mehke množice v obliki

$$\mu_A(x) = f(x). \quad (\text{B.23})$$

Množico A zapišemo s funkcionalno predstavitev na naslednji način

$$A = \left\{ \mu(x) = \frac{1}{1 + (x - 4)^2}, \quad 1 \leq x \leq 7 \right. . \quad (\text{B.24})$$

Mehka množica A je predstavljena na sliki B.5.

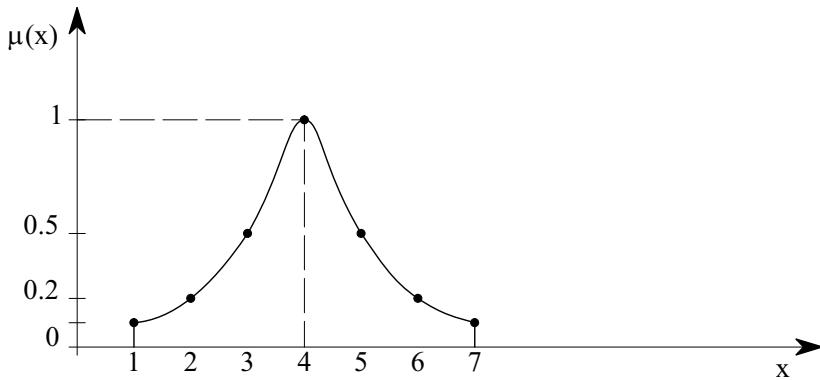
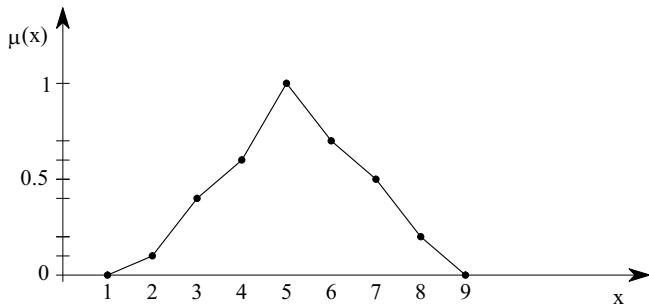
- predstavitev mehke množice v obliki urejenih parov (*ang. paired presentation*) je definirana v obliki

$$\mu_B(x) = \left(\frac{\mu(x_1)}{x_1}, \frac{\mu(x_2)}{x_2}, \dots, \frac{\mu(x_n)}{x_n} \right) . \quad (\text{B.25})$$

Primer predstavitve mehke množice B

$$\mu_B(x) = \left(\frac{0}{1}, \frac{0.1}{2}, \frac{0.4}{3}, \frac{0.6}{4}, \frac{1}{5}, \frac{0.7}{6}, \frac{0.5}{7}, \frac{0.2}{8}, \frac{0}{9} \right) \quad (\text{B.26})$$

v obliki urejenih parov je na sliki B.6.

Slika B.5. Mehka množica A .Slika B.6. Mehka množica B .

B.1.3 Mehki logični operatorji

Prav tako kot definiramo operatorje v klasični logiki, so definirani operatorji tudi v mehki logiki. Poznamo operatorje *preseka*, *unije* in *komplementa*. Operator *preseka* imenujemo tudi T ali *and* operator. Operator *unije* pa S ali *or* operator.

V prostoru X , ki ga sestavljajo mehke množice A , B in C izvajamo določene operacije in na ta način tvorimo nove množice istega prostora. Simbolično lahko operacijo nad mehkimi množicami definiramo kot *kompozitum*, ki ga zapišemo kot $A \circ B$. Operacija kompozitura lahko predstavlja mehki presek, mehko unijo ali kakšno drugo mehko operacijo. V tem poglavju si bomo ogledali najpomembnejše lastnosti posameznih operatorjev.

Splošni T operator. Splošni T operator, ki ga imenujemo tudi trikotna norma ali mehki presek je definiran na karteziskem produktu

$$[0, 1] \times [0, 1] \rightarrow [0, 1]. \quad (\text{B.27})$$

Operator T zapišemo na naslednji način

$$\mu_{A \cap B}(x) = T(\mu_A(x), \mu_B(x)) \quad (\text{B.28})$$

in mora zadostiti naslednjim pogojem:

T-1 robni pogoji

$$T(0, a) = 0, \quad (\text{B.29})$$

$$T(a, 1) = a, \quad (\text{B.30})$$

T-2 komutativnost

$$T(a, b) = T(b, a), \quad (\text{B.31})$$

T-3 monotonost

$$T(a, b) \leq T(c, d), \quad a \leq c, b \leq d, \quad (\text{B.32})$$

T-4 asociativnost

$$T(T(a, b), c) = T(a, T(b, c)), \quad (\text{B.33})$$

T-5 idempotencija

$$T(a, a) = a, \quad (\text{B.34})$$

T-6 zveznost funkcije

$$T(\mu_A(x), \mu_B(x)) \text{ je zvezna funkcija.} \quad (\text{B.35})$$

V nadaljevanju bodo opisani nekateri najpomembnejši T operatorji:

Algebraični T operator

$$T_A(a, b) = ab, \quad (\text{B.36})$$

Zadehov ali originalni T operator

$$T_Z(a, b) = \min(a, b), \quad (\text{B.37})$$

Lukasiewiczev T operator

$$T_L(a, b) = \max(0, a + b - 1), \quad (\text{B.38})$$

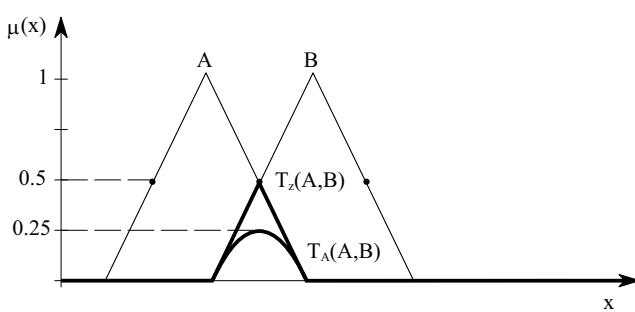
Drastični ali Webrow T operator

$$T_W(a, b) = \begin{cases} a, & b = 1 \\ b, & a = 1 \\ 0, & a \notin 1, b \notin 1 \end{cases}. \quad (\text{B.39})$$

Za primer si oglejmo primerjavo med Zadehovim in algebraičnim T operatorjem. Iz mehkih množic A in B tvorimo novo množico enkrat na osnovi Zadehovega T operatorja in enkrat na osnovi algebraičnega T operatorja. Primerjava je prikazana na sliki B.7:

$$T_Z(\mu_A(x), \mu_B(x)) = \min(\mu_A(x), \mu_B(x)), \quad (\text{B.40})$$

$$T_A(\mu_A(x), \mu_B(x)) = \mu_A(x)\mu_B(x). \quad (\text{B.41})$$



Slika B.7. Kompozitum mehkih množic A in B na osnovi Zadehovega in algebraičnega T operatorja.

Operator T je lahko *parametrični* ali *neparametrični*. Zgoraj našteti operatorji so neparametrični. Parametrični operatorji vsebujejo parameter, ki zavzame določeno vrednost. Na osnovi tega parametra pa se spreminja tudi njegova funkcionalnost. V nadaljevanju bomo našteli nekaj pomembnejših parameterskih T operatorjev

Hamacherjev T operator

$$T_H(a, b, \gamma) = \frac{ab}{\gamma + (1 - \gamma)(a + b - ab)} , \quad (\text{B.42})$$

Yagerjev T operator

$$T_Y(a, b, w) = 1 - \min(1, ((1 - a)^w + (1 - b)^w)^{\frac{1}{w}}) , \quad (\text{B.43})$$

Duboisjev T operator

$$T_{DP}(a, b, \alpha) = \frac{ab}{\max(a, b, \alpha)} . \quad (\text{B.44})$$

Če natančneje opazujemo posamezne operatorje ugotovimo, da z izbiro določenih vrednosti parametrov dobimo neketere neparametrične operatorje. Na primer pri vrednosti parametra $\gamma = 1$ postane *Hamacherjev T operator* enak *algebraičnemu T operatorju*

$$\begin{aligned} \gamma = 1, \quad T_H(a, b, \gamma) &= \frac{ab}{\gamma + (1 - \gamma)(a + b - ab)} , \\ T_H(a, b, 1) &= ab , \\ T_H(a, b, 1) &= T_A(a, b) . \end{aligned} \quad (\text{B.45})$$

Splošno lahko katerikoli T operator uvrstimo med dve skrajni meji, ki sta definirani z *drastičnim ali Webrovim T operatorjem* $T_W(a, b)$, kot spodnjo mejo in *Zadehovim ali originalnim T operatorjem* $T_Z(a, b)$, kot zgornjo mejo, kar lahko matematično zapišemo

$$T_W(a, b) \leq T(a, b) \leq T_Z(a, b) . \quad (\text{B.46})$$

V primeru reševanja inverznih problemov je potrebno vpeljati inverzni T operator. Splošni inverzni T operator mora zadostiti pogojem, ki so podani v naslednji definiciji.

Definicija 4 Operator $\propto: [0, 1] \times [0, 1] \rightarrow [0, 1]$ definiramo tako, da izpolnjuje naslednje zahteve

1. $x \propto \max(y, z) > x \propto y, x \propto z ,$
2. $x \propto T(x, y) < y ,$
3. $x \propto (x \propto y) > y .$

Zadehov T operator ima pripadojoči inverzni \propto operator, ki je podan z naslednjo relacijo

$$a \propto b = \begin{cases} 1, & a < b \\ b, & a \geq b \end{cases} . \quad (\text{B.47})$$

Splošni S operator. Splošni S operator (t.i. trikotna ko-norma) ali splošna mehka unija je ravno tako kot splošeni T operator definiran na karteziskem produktu in mora zadostiti pogojem monotonosti, komutativnosti in asociativnosti. S operator je definiran na naslednji način

$$\mu_{A \cup B}(x) = S(\mu_A(x), \mu_B(x)) \quad (\text{B.48})$$

in mora zadostiti naslednjim aksiomom:

S-1 robni pogoji

$$S(0, a) = a , \quad (\text{B.49})$$

$$S(a, 1) = 1 , \quad (\text{B.50})$$

S-2 komutativnost

$$S(a, b) = S(b, a), \quad (\text{B.51})$$

S-3 monotonost

$$S(a, b) \leq S(c, d), \quad a \leq c, \quad b \leq d, \quad (\text{B.52})$$

S-4 asociativnost

$$S(S(a, b), c) = S(a, S(b, c)), \quad (\text{B.53})$$

S-5 idempotencija

$$S(a, a) = a, \quad (\text{B.54})$$

S-6 zveznost funkcije

$$S(\mu_A(x), \mu_B(x)) \text{ je zvezna funkcija.} \quad (\text{B.55})$$

V nadaljevanju bodo opisani nekateri najpomembnejši S operatorji:

algebraični S operator

$$S_A(a, b) = a + b - ab, \quad (\text{B.56})$$

Zadehov ali originalni S operator

$$S_Z(a, b) = \max(a, b), \quad (\text{B.57})$$

Lukasiewiczev S operator

$$S_L(a, b) = \min(1, a + b), \quad (\text{B.58})$$

drastični ali Webrov S operator

$$S_W(a, b) = \begin{cases} a, & b = 0 \\ b, & a = 0 \\ 1, & a \notin 0, b \notin 0 \end{cases}. \quad (\text{B.59})$$

Za primer si oglejmo primerjavo med Zadehovim in algebraičnim S operatorjem. Iz mehkih množic A in B tvorimo novo množico enkrat na osnovi Zadehovega S operatorja in enkrat na osnovi algebraičnega S operatorja, kar prikazuje slika B.8

$$T_Z(\mu_A(x), \mu_B(x)) = \max(\mu_A(x), \mu_B(x)), \quad (\text{B.60})$$

$$T_A(\mu_A(x), \mu_B(x)) = \mu_A(x)\mu_B(x). \quad (\text{B.61})$$

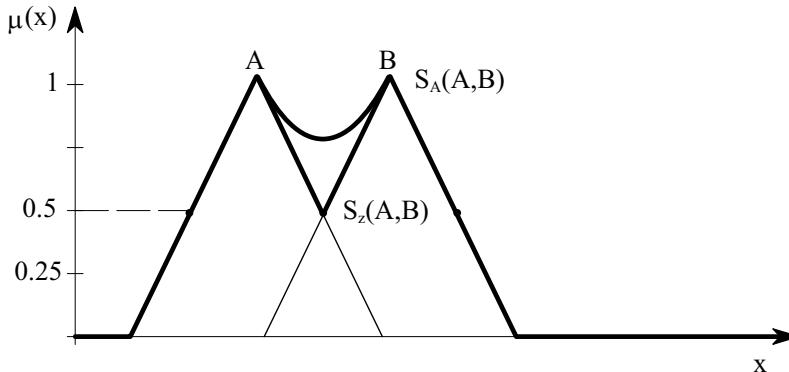
Operator S je lahko definiran v *parametrični* ali *neparametrični* obliki. Zgoraj našteti operatorji so neparametrični. Parametrični operatorji vsebujejo parameter, ki zavzame določeno vrednost. V nadaljevanju bodo našteti nekateri pomembnejši parametrični operatorji:

Hamacherjev S operator

$$S_H(a, b, \gamma) = \frac{a + b - (1 - \gamma)ab}{1 - (1 - \gamma)ab}, \quad \gamma \geq 0, \quad (\text{B.62})$$

Yagerjev S operator

$$S_Y(a, b, w) = \min(1, (a^w + b^w)^{\frac{1}{w}}), \quad w \geq 1, \quad (\text{B.63})$$



Slika B.8. Kompozitum mehkih množic A in B na osnovi Zadehovega in Algebraičnega S operatorja.

Duboisjev S operator

$$S_{DP}(a, b, \alpha) = \frac{a + b - ab - \min(a, b, 1 - \alpha)}{\max(1 - a, 1 - b, \alpha)}, \quad 0 \leq \alpha \leq 1. \quad (\text{B.64})$$

Če natančneje opazujemo posamezne operatorje ugotovimo, da z izbiro določenih vrednosti parametrov dobimo neketere neparametrične operatorje. Pri $\alpha = 0$ postane *Duboisjev S operator* enak *algebraičnemu S operatorju*

$$\begin{aligned} \alpha = 0, \quad S_{DP}(a, b, \alpha) &= \frac{a + b - ab - \min(a, b, 1 - \alpha)}{\max(1 - a, 1 - b, \alpha)}, \\ S_{DP}(a, b, 0) &= a + b - ab, \\ S_{DP}(a, b, 0) &= S_A(a, b). \end{aligned} \quad (\text{B.65})$$

Splošni S operator lahko uvrstimo med dve skrajni meji, ki sta definirani z *Webrovo drastično vsoto* $S_W(a, b)$, kot spodnjo mejo in *Zadehovim ali originalnim S operatorjem* $S_Z(a, b)$, kot zgornjo mejo. To lahko matematično zapišemo kpt

$$S_Z(a, b) \leq S(a, b) \leq S_W(a, b). \quad (\text{B.66})$$

O *komplementarni dualnosti* operatorjev T in S lahko govorimo takrat kadar velja naslednja relacija:

$$T(a, b) = c(S(c(a), c(b))), \quad (\text{B.67})$$

kjer je c mehki komplement, ki bo natančneje opisan v nadaljevanju.

Splošni mehki komplement. Komplement mehke množice A simbolično zapišemo kot \overline{A} . Definiran je z naslednjimi aksiomi:

c-1 robni pogoji

$$c(1) = 0, \quad c(0) = 1, \quad (\text{B.68})$$

c-2 monotonost

$$a, b \in [0, 1], \quad a < b, \quad c(a) \geq c(b), \quad (\text{B.69})$$

c-3 zveznost

$$c(x) \text{ je zvezna funkcija na domeni } x, \quad (\text{B.70})$$

c-4 involucija

$$c(c(a)) = a, \forall a \in [0, 1]. \quad (\text{B.71})$$

Parameterski mehki komplement lahko zapišemo v obliki

$$c_w(a) = (1 - a^w)^{\frac{1}{w}}. \quad (\text{B.72})$$

Če izberemo vrednost parametra $w = 1$ dobimo standardni mehki komplement. Pri izbiri $w > 1$ imamo konveksno in pri $w < 1$ konkavno obliko mehkega komplementa.

B.2 Mehko sklepanje

V tem delu bodo najprej definirani osnovni gradniki mehkega sklepanja, to so *mehke izjave*. Sestavljanje mehkih izjav vodi do *mehkih pravil* in do *mehkih tabel*, ki so dejansko množice paralelnih pravil. Večji del pa bo posvečen *mehki implikaciji*, ki je temeljna operacija mehkega sklepanja.

B.2.1 Mehka izjava in predikat

V klasični logiki poznamo *izjave* in *predikate*. Izjave so spremenljivke v izjavnem računu, predikati pa spremenljivke v predikatnem računu. Za primer zapišemo izjavi x in y

$$x : a \text{ is } rdeč \quad (\text{B.73})$$

$$y : a \text{ is } okrogel. \quad (\text{B.74})$$

Z izjavnim računom sestavljamo nove izjave v obliki funkcij $f(x, y)$.

Predikatni račun v prostoru P definira predikatno funkcijo $p(x)$, kjer je x element predikatnega prostora. Predikatna funkcija je pravzaprav izjava o določenem elementu prostora P , ki jo lahko zapišemo

$$p(x) : x \text{ is } okrogel. \quad (\text{B.75})$$

Če izberemo določen element, na primer $x = a$, dobimo

$$p(a) = \begin{cases} 1, & a \text{ is } okrogel \\ 0, & drugje \end{cases}, \quad (\text{B.76})$$

Definicija mehkega predikata. Mehki predikat definiramo kot predikat, kjer je predikatna funkcija $p(x)$ *mehka pripadnostna funkcija*. Mehko pripadnostno funkcijo zapišemo navadno kot funkcijo $\mu(x)$. Zapis mehkega predikata je definiran z naslednjim zapisom

$$\mu(x) : x \text{ is } A, \quad (\text{B.77})$$

kjer je A *mehki segment* ali *lingvistična labela* in x mehka spremenljivka. Mehke izjave povezujejo mehke spremenljivke in lingvistične labele in so hkrati osnovni gradniki mehkega sklepanja, ki jih lahko povezujemo med seboj z uporabo logičnih mehkih T in S operatorjev.

Do sedaj smo predpostavljeni izjave, ki so temeljile na enaki domeni. Če predpostavimo izjave, ki temeljijo na različnih domenah dobimo rezultat operacij med temi izjavami v obliki mehkih relacij. Kot primer lahko zapišemo izjavo f

$$f : x_1 \text{ is } A_1 \text{ and } x_2 \text{ is } A_2, \quad (\text{B.78})$$

kjer imata elementa x_1 in x_2 pripadnosti $\mu_{A_1}(x_1)$ in $\mu_{A_2}(x_2)$. Izjavo f lahko predstavimo z mehko relacijo F , ki ima naslednjo pripadnostno funkcijo

$$F : \mu_F(x_1, x_2) = T(\mu_{A_1}(x_1), \mu_{A_2}(x_2)). \quad (\text{B.79})$$

B.2.2 Mehka pravila

Osnovo mehkih pravil predstavlja *implikacija*, ki jo v klasični logiki zapišemo z izrazom

$$A \rightarrow B \quad (\text{B.80})$$

in predstavlja matematični zapis lingvističnega stavka ali *mehkega pravila*

$$\text{if } A \text{ then } B . \quad (\text{B.81})$$

Izraz predstavlja *mehko implikacijo*, če sta A in B mehki množici. Z logičnega stališča pomeni gornja trditev to, da dobimo na izhodu množico B povsem zanesljivo samo takrat, kadar je na vhodu množica A . Če imamo na vhodu \bar{A} , potem na izhodu lahko imamo B ali pa tudi ne. Če predpostavimo, da pripada mehki množici A prostor U in mehki množici B prostor V , potem mehka pravila predstavlja mehko relacijo R v okviru kartezijjskega produkta $U \times V$. Simbolično torej lahko zapišemo mehko relacijo med mehko množico A in B v obliki

$$\mathbf{R} : A \rightarrow B , \quad (\text{B.82})$$

kjer je pripadnostna funkcija

$$\mu_R(u, v) = T_{u,v}(\mu_A(u), \mu_B(v)) . \quad (\text{B.83})$$

Če želimo iz določene mehke relacije izračunati neznano mehko množico, potem je to inverzen problem, ki ga lahko zapišemo na naslednji način

$$B = A \circ R , \quad (\text{B.84})$$

$$B = A \circ (A \rightarrow B) , \quad (\text{B.85})$$

kjer je pripadnostna funkcija enaka

$$\mu_B(v) = S_u(T(\mu_A(u), \mu_R(u, v))) . \quad (\text{B.86})$$

Definicija. Predpostavimo mehko relacijo $R : G_1 \times G_2 \Rightarrow [0, 1]$ in pa mehko množico $G_1 \Rightarrow [0, 1]$. Mehko množico G_2 lahko dobimo s pomočjo izraza

$$\mu_{G_2}(x_2) = S_{x_1}(T(\mu_{G_1}(x_1), \mu_R(x_1, x_2))) . \quad (\text{B.87})$$

V splošnem lahko rečemo, da pravilo sestavlja *premisa* ali *pogoj*, ki je lahko sestavljen iz več izrazov med seboj povezanih z logičnimi *and* ali *or* operatorji in *konsekvenca*. Tako pravilo lahko zapišemo na naslednji način

$$\text{if } x_1 \text{ is } A_1 \text{ and } x_2 \text{ is } A_2 \text{ then } y \text{ is } B . \quad (\text{B.88})$$

Matematično lahko gornje pravilo zapišemo v naslednji obliki

$$R = I(T(A_1, A_2), B) , \quad (\text{B.89})$$

kjer T označuje splošni operator, ki je na mestu operatorja preseka, I pa predstavlja splošno funkcijo implikacije. Če predpostavimo mehke množice A_1, A_2 in B s pripadnostnimi funkcijami $\mu_{A_1}(x_1), \mu_{A_2}(x_2)$ in $\mu_B(y)$, dobimo pripadnostno funkcijo mehke relacije R z naslednjo enačbo

$$\mu_R(x_1, x_2, y) = I(T(\mu_{A_1}(x_1), \mu_{A_2}(x_2)), \mu_B(y)) . \quad (\text{B.90})$$

V nadaljevanju bodo najprej predstavljeni različni tipi mehkih implikacij, nato pa posamezni primeri mehke implikacije.

B.2.3 Mehka implikacija

V nadaljevanju bomo za implikacijo uporabili simbol $I(a, b)$, kjer je $a, b \in [0, 1]$.

Tipi mehkih implikacij. Ravno tako kot v primeru ostalih operatorjev iz klasične logike je tudi operator mehke implikacije mogoče realizirati na več različnih načinov. V tem poglavju bodo predstavljeni najpomembnejši tipi mehkih implikacij:

1. **klasična mehka implikacija ali S-implikacija**

$$I(a, b) = S(c(a), b) , \quad (\text{B.91})$$

2. **mehka implikacija na osnovi kvantne logike**

$$I(a, b) = S(c(a), T(a, b)) , \quad (\text{B.92})$$

kjer sta S in T dualni operaciji,

3. **mehka implikacija residua**

$$I(a, b) = \begin{cases} 1, & a \leq b \\ 0, & a = 1 \wedge b = 0 \\ \in [0, 1], & drugje \end{cases} , \quad (\text{B.93})$$

4. **mehka implikacija kot konjunkcija**

$$I(a, b) = T(a, b) . \quad (\text{B.94})$$

Ta tip implikacije se pogosto uporablja pri mehkem vodenju.

5. **Mehka implikacija kot disjunkcija**

$$I(a, b) = S(a, b) . \quad (\text{B.95})$$

V nadaljevanju bomo našeli nekaj najpomembnejših mehkih implikacij, ki ustrezajo zgoraj navedenim tipom:

1. **Kleene-Dienesova implikacija**

$$I(a, b) = \max(1 - a, b) , \quad (\text{B.96})$$

predstavlja tip *S-implikacije*.

2. **Implikacija po Goguenu**

$$I(a, b) = \begin{cases} 1, & a = 0 \\ \min(\frac{b}{a}, 1), & drugje \end{cases} , \quad (\text{B.97})$$

predstavlja tip *residualne implikacije*.

3. Mamdanijeva implikacija

$$I(a, b) = \min(a, b) , \quad (\text{B.98})$$

4. Larsenova implikacija

$$I(a, b) = a.b , \quad (\text{B.99})$$

predstavljata tip *implikacije s konjunkcijo*.

5. Zadehova implikacija

$$I_Z(a, b) = \max(1 - a, \min(a, b)) , \quad (\text{B.100})$$

6. Lukasiewiczeva implikacija

$$I_L(a, b) = \min(1, 1 - a + b) , \quad (\text{B.101})$$

7. Gödelova implikacija

$$I_{Gö}(a, b) = \begin{cases} 1, & a \leq b \\ b, & \text{drugje} \end{cases} , \quad (\text{B.102})$$

8. Kleene-Dienes-Lukasiewiczeva implikacija

$$I_{KDL}(a, b) = 1 - a + a.b , \quad (\text{B.103})$$

9. Willmottova implikacija

$$I_{Wm}(a, b) = \min(\max(1 - a, b), \max(a, 1 - a), \max(1 - b, b)) . \quad (\text{B.104})$$

1. M.R. Anderberg. *Cluster Analysis for Applications*. Probability and Mathematical Statistics. Academic Press, New York, 1971.
2. Soeterboek A.R.M. *Predictive control - A unified approach*. Prentice Hall, New York, 1992.
3. Ydstie B. Extended horizon adaptive control. In *Proc. 9th IFAC World Congress*, pages 133–137, Budapest, Hungary, 1984.
4. R. Babuška. *Fuzzy Modeling and Identification*. PhD thesis, Dept. of Control Engineering, Delft University of Technology, Delft, The Netherlands, 1996.
5. R. Babuška and H.B. Verbruggen. An overview of fuzzy modeling for control. *Control Engineering Practice*, 4(11):1593–1606, 1996.
6. R. Babuška and H.B. Verbruggen. Fuzzy set methods for local modelling and identification. In R. Murray-Smith and T.A. Johansen (Eds.), editors, *Multiple Model Approaches to Modelling and Control*, chapter 2, pages 75–100. Taylor & Francis, London, 1997.
7. T. Bäck, U. Hammel, and H.-P. Schwefel. Evolutionary computation: Comments on the history and current state. *IEEE Transactions on Evolutionary Computation*, 1(1):3–17, April 1997.
8. T. Bernd, A. Kroll, and H. Schwarz. Approximation nichtlinearer dynamischer Prozesse mit optimierten funktionalen Fuzzy-Modellen. In *7. Workshop "Fuzzy Control" des GMA-UA 1.4.2*, Dortmund, Germany, 1997.
9. J.C. Bezdek. *Pattern Recognition with Fuzzy Objective Function*. Plenum Press, New York, 1981.
10. C.M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, 1995.
11. G.E.P. Box and G.M. Jenkins. *Times Series Analysis, Forecasting and Control*. Holden-Day, San Francisco, 1970.
12. M.A. Branch and A. Grace. *MATLAB Optimization Toolbox User's Guide, Version 1.5*. The MATHWORKS Inc., Natick, MA, 1998.
13. E F Camacho and C Bordons. *Model Predictive Control*. Springer-Verlag, London, 2004.
14. S. Chen and S.A. Billings. Representations of nonlinear systems: The NARMAX model. *International Journal of Control*, 49(3):1013–1032, 1983.
15. S. Chen, S.A. Billings, and W. Luo. Orthogonal least squares methods and their application to nonlinear system identification. *International Journal of Control*, 50(5):1873–1896, 1989.
16. S. Chen, C.F.N. Cowan, and P.M. Grant. Orthogonal least-squares learning algorithm for radial basis function networks. *IEEE Transactions on Neural Networks*, 2(2), March 1991.
17. X. Chen and M. Bushnell. *Efficient Branch and Bound Search with Applications to Computer-Aided Design*. Kluwer Academic Publishers, Boston, 1996.
18. D W Clarke. *Advances in model-based predictive control*. Oxford Sci., Oxford, U.K., 1994.
19. Tuffs P.S. Clarke D.W., Mohtadi C. Generalized predictive control-part 1, part2. *Automatica*, 24:137–160, 1987.
20. W.S. Cleveland, S.J. Devlin, and E. Grosse. Regression by local fitting: Methods, properties, and computational algorithms. *Journal of Econometrics*, 37:87–114, 1996.
21. Cutler C.R. Dynamic matrix control -a computer control algorithm. In *Proceedings ACC*, San Francisco, USA, 1982.
22. Pedrycz W. Czogala E. On identification in fuzzy systems and its applications in control problems. *North Holland Publishing Company*, 6(1):257–273, 1981.
23. L. Davis. *Handbook of Genetic Algorithms*. van Nostrand Reinhold, New York, 1991.
24. Van Cauwenberghhe A. De Keyser R. A self-tuning multistep predictors. *Automatica*, (17):167–174, 1985.
25. McAvoy T.J. Dong, D. Batch tracking via nonlinear principal component analysis. *AICHE Journal*, 42, 8, 1996.
26. Dejan Dovžan. *Študija samonastavljivih algoritmov za prediktivno funkcijski regulator*. Diplomska naloga, Faculty of electrical engineering, Ljubljana, sept. 2008.
27. Dejan Dovzan and Igor Skrjanc. Control of mineral wool thickness using predictive functional control. *Robotics and Computer-Integrated Manufacturing*, 0(0):0, 2011.
28. F J Doyle III, T A Ogunnaike, and R K Pearson. Nonlinear model-based control using second-order Volterra models. *Automatica*, 31(5):697–714, 1995.
29. N.R. Draper and H. Smith. *Applied Regression Analysis*. Probability and Mathematical Statistics. John Wiley & Sons, New York, 1981.

30. S. Ernst. Hinging hyperplane trees for approximation and identification. In *IEEE Conference on Decision and Control (CDC)*, pages 1261–1277, Tampa, USA, 1998.
31. P. Eykhoff. *System Identification*. John Wiley & Sons, London, 1974.
32. M. Frean. The upstart algorithm: A method for constructing and training feedforward neural networks. *Neural Computation*, 2(2):198–209, 1990.
33. S.I. Gallant. *Neural Networks Learning and Expert Systems*. MIT Press, Cambridge, 1987.
34. S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58, 1992.
35. M. Gen. *Genetic Algorithms and Engineering Design*. Wiley, New York, 1997.
36. D.E. Goldberg. *Genetic Algorithms in Search, Optimization & Machine Learning*. Addison-Wesley, Reading, 1989.
37. R. Haber. Nonlinearity tests for dynamic processes. In *IFAC Symposium on Identification and System Parameter Estimation*, pages 409–414, York, UK, 1985.
38. M.T. Hagan and M.B. Menhaj. Training feedforward networks with the Marquardt algorithm. *IEEE Transactions on Neural Networks*, 5(6):989–993, November 1994.
39. E. Hänsler. *Statistische Signale*. Springer, Berlin, 1991.
40. R.J. Hathaway and J.C. Bezdek. Switching regression models and fuzzy clustering. *IEEE Transactions on Fuzzy Systems*, 1(3):195–204, 1993.
41. X. He and H. Asada. A new method for identifying orders of input-output models for nonlinear dynamic systems. In *American Control Conference (ACC)*, pages 2520–2524, San Francisco, USA, 1993.
42. J. Heitkötter and D. Beasley. The hitch-hiker's guide to evolutionary computation: A list of frequently asked questions (FAQ).
43. H. Hjalmarsson, M. Gevers, and F. de Bruyne. For model-based control design, closed-loop identification gives better performance. *Automatica*, 32(12):1659–1673, 1996.
44. J.H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, 1975.
45. F. Höppner, F. Klawonn, and R. Kruse. *Fuzzy-Clusteranalyse*. Vieweg, Braunschweig, 1997.
46. K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Computation*, 2:359–366, 1989.
47. S Blažič I Škrjanc and S Oblak. An approach to predictive control of multivariable time-delayd plant: Stability and design issues. *ISA transactions*, 43:585–595, 2004.
48. L. Ingber. Very fast simulated re-annealing. *Mathematical and Computer Modelling*, 12:967–973, 1989.
49. L. Ingber. Simulated annealing: Practice versus theory. *Mathematical and Computer Modelling*, 18(11):29–57, 1993.
50. L. Ingber. Adaptive simulated annealing (ASA): Lessons learned. *Mathematical and Computer Modelling*, 25(1):33–54, 1995.
51. L. Ingber and B. Rosen. Genetic algorithms and very fast simulated reannealing: A comparison. *Mathematical and Computer Modelling*, 16(11):87–100, 1992.
52. R. Isermann. Required accuracy of mathematical models of linear time invariant controlled elements. *Automatica*, 7:333–341, 1971.
53. R. Isermann. *Identifikation dynamischer Systeme - Band 1, 2. ed.* Springer, Berlin, 1992.
54. R. Isermann. *Identifikation dynamischer Systeme - Band 2, 2. ed.* Springer, Berlin, 1992.
55. D. Matko Isermann R, K.-H. Lachmann. *Adaptive Control Systems*. Prentice Hall, New York, Toronto, Sydney, 1992.
56. J.-S.R. Jang. *Neuro-Fuzzy Modeling: Architectures, Analyses, and Applications*. PhD thesis, EECS Department, Univ. of California at Berkeley, Berkeley, USA, 1992.
57. J.-S.R. Jang. ANFIS: Adaptive-network-based fuzzy inference systems. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(3):665–685, 1993.
58. J.-S.R. Jang, C.T. Sun, and E. Mizutani. *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. Prentice Hall, Englewood Cliffs, 1997.
59. T.A. Johansen. *Operating Regime Based Process Modeling and Identification*. PhD thesis, Dept. of Engineering Cybernetics, Norwegian Institute of Technology, Trondheim, Norway, 1994.
60. T.A. Johansen. Identification of non-linear system structure and parameters using regime decomposition. *Automatica*, 31(2):321–326, 1995.
61. R. Johansson. *System Modeling & Identification*. Prentice Hall, Englewood Cliffs, 1993.
62. R.D. Jones and coworkers. Nonlinear Adaptive Networks: A little Theory, a few Applications. Technical Report 91-273, Los Alamos National Lab., New Mexico, 1991.
63. Dumortier F.A.G. Keyser de R.M.C., van der Velde G.A. A comparative study of self-adaptive long-range predictive control methods. *Automatica*, 24(2):149–163, 1988.
64. S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.
65. T. Kohonen. *Self-Organizing Maps, 2nd edition*. Springer, Berlin, 1997.
66. M. Kortmann. *Die Identifikation nichtlinearer Ein- und Mehrgrößensysteme auf der Basis nichtlinearer Modellansätze*. Reihe 8: Mess-, Steuerungs- und Regelungstechnik, Nr. 177. VDI-Verlag, Düsseldorf, 1989.
67. B. Kosko. Fuzzy systems as universal approximators. *IEEE Transactions on Computers*, 43:1329–1333, 1994.

68. J.R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, 1992.
69. J.R. Koza. *Genetic Programming II: Automatic Discovery of Reusable Programs*. MIT Press, Cambridge, MA, 1994.
70. J.R. Koza. *Genetic Programming III: Darwinian Invention and Problem Solving*. Morgan Kaufmann, San Francisco, CA, 1999.
71. A. Kroll. *Fuzzy-Systeme zur Modellierung und Regelung komplexer technischer Systeme*. Reihe 8: Mess-, Steuerungs- und Regelungstechnik, Nr. 612. VDI-Verlag, Düsseldorf, 1997.
72. F. Kursawe and H.-P. Schwefel. Optimierung mit Evolutionären Algorithmen. *Automatisierungstechnische Praxis*, 39(9):10–17, 1997.
73. E.L. Lawler and E.D. Wood. Branch-and-bound methods: A survey. *Journal of Operations Research*, 14, 1966.
74. Y. Lee, S. Oh, and M. Kim. The effect of initial weights on premature saturation in backpropagation learning. In *International Joint Conference on Neural Networks*, volume 1, pages 765–770, Seattle, USA, 1991.
75. I.J. Leontaritis and S.A. Billings. Input-output parametric models for nonlinear systems, part 1: Deterministic nonlinear systems. *International Journal of Control*, 41:303–344, 1985.
76. M Lepetić, I Škrjanc, H G Chiacchiarini, and D Matko. Predictive functional control based on fuzzy model: magnetic suspension system case study. *Engineering Applications of Artificial Intelligence*, 16:425–430, 2003.
77. A.U. Levin and K.S. Narendra. Identification using feedforward networks. *Neural Computation*, 7(2):349–357, 1995.
78. L. Ljung. *System Identification: Theory for the User*. Prentice Hall, Englewood Cliffs, 1987.
79. E.H. Mamdani. Application of fuzzy logic to approximate reasoning using linguistic systems. *Fuzzy Sets & Systems*, 26:1182–1191, 1977.
80. Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, Berlin, 1992.
81. A.J. Miller. *Subset Selection in Regression*. Statistics and Applied Probability. Chapman & Hall, New York, 1990.
82. M.L. Minsky and S.A. Pappert. *Perceptrons*. MIT Press, Cambridge, MA, 1969.
83. L.G. Mitten. Branch-and-bound methods: General formulation and properties. *Journal of Operations Research*, 18, 1970.
84. Harris C.J. Moore C.G. Indirect adaptive fuzzy control. *Int.J.Control*, 56(2):441–468, 1992.
85. Manfredi C. Mosca E., Zappa G. Multistep horizon self-tuning controllers: the musmar approach. In *Proceedings 9th IFAC World Congress*, Budapest, Hungary, 1984.
86. R. Murray-Smith. *A Local Model Network Approach to Nonlinear Modeling*. PhD thesis, University of Strathclyde, Strathclyde, UK, 1994.
87. R. Murray-Smith and T.A. Johansen. Local learning in local model networks. In R. Murray-Smith and T.A. Johansen (Eds.), editors, *Multiple Model Approaches to Modelling and Control*, chapter 7, pages 185–210. Taylor & Francis, London, 1997.
88. Y. Nakamori and M. Ryoke. Identification of fuzzy prediction models through hyperellipsoidal clustering. *IEEE Transactions on Systems, Man, and Cybernetics*, 24(8):1153–1173, 1994.
89. K.S. Narendra and K. Parthasarathy. Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, 1(1):4–27, March 1990.
90. Annaswamy A.M. Narendra K.S. *Stable Adaptive Systems*. Prentice-Hall, Englewood Cliffs, New Jersey, 1989.
91. McGregor J.F. Nomikos, P. Monitoring batch processes using multiway principal component analysis. *AIChE Journal*, 40, 8, 1994.
92. S Oblak and I Škrjanc. Multivariable fuzzy predictive functional control of a mimo nonlinear system. In *Intelligent Control, 2005. Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation*, pages 1029 –1034, june 2005.
93. A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. Electrical & Electronic Engineering. McGraw-Hill, New York, 1991.
94. M.-K. Park, S.-H. Ji, E.-T. Kim, and M. Park. Identification of Takagi-Sugeno fuzzy models via clustering and hough transform. In H. Hellendoorn and D. Driankov, editors, *Fuzzy Model Identification: Selected Approaches*, chapter 3, pages 91–161. Springer, Berlin, 1997.
95. V. Peterka. Bayesian approach to system indetification. In P. Eykhoff, editor, *Trends and Progress in System Identification*. Pergamon Press, Oxford, 1991.
96. S Joe Qin and Thomas A Badgwell. A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733 – 764, 2003.
97. I. Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog-Verlag, Stuttgart, 1975.
98. G.V. Reklaitis, A. Ravindran, and K.M. Ragsdell. *Engineering Optimization – Methods and Applications*. John Wiley & Sons, London, 1983.
99. J Richalet. Industrial application of model based predictive control. *Automatica*, 29(5):1251–1274, 1993.
100. J Richalet, A Rault, J L Testud, and J Papon. Model predictive heuristic control: Application to industrial processses. *Automatica*, 14(5):413–428, 1978.

101. J. Richalet, A. Rault, J.L. Testud, and J. Papon. Model predictive heuristic control: Applications to industrial processes. *Automatica*, 14:413–428, 1978.
102. H. Ritter, T. Martinetz, and K. Schulten. *Neuronale Netze*. Addison-Wesley, Bonn, 1991.
103. D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. In D.E. Rumelhart and J.L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, volume 1, chapter 8. MIT Press, Cambridge, 1986.
104. W.S. Sarle. Frequently asked questions (FAQ) of the neural network newsgroup.
105. L.E. Scales. *Introduction to Non-Linear Optimization*. Computer and Science Series. Macmillan, London, 1985.
106. H.-P. Schwefel. *Numerical Optimization of Computer Models*. John Wiley & Sons, Chichester, 1988.
107. T. Söderström and P. Stoica. *System Identification*. Prentice Hall, New York, 1989.
108. M. Sugeno and G.T. Kang. Structure identification of fuzzy model. *Fuzzy Sets & Systems*, 28(1):15–33, 1988.
109. M. Sugeno and K. Tanaka. Successive identification of a fuzzy model and its application to prediction of a complex system. *Fuzzy Sets & Systems*, 42:315–334, 1991.
110. T. Takagi and M. Sugeno. Fuzzy identification of systems and its application to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, 15(1):116–132, 1985.
111. T. Takagi and M. Sugeno. A fuzzy identification of systems and its applications to modelling and control. *IEEE Trans. on Syst., Man and Cyber.*, 15(1):116–132, 1985.
112. M. Tanaka, J. Ye, and T. Tanino. Identification of nonlinear systems using fuzzy logic and genetic algorithms. In *IFAC Symposium on System Identification*, pages 301–306, Copenhagen, Denmark, 1994.
113. A.N. Tikhonov and V.Y. Arsenin. *Solutions of Ill-Posed Problems*. Wiley, New York, 1977.
114. P.M.J. Van den Hof. Closed-loop issues in system identification. In *IFAC Symposium on System Identification*, pages 1651–1664, Fukuoka, Japan, 1997.
115. I Škrjanc and D Matko. Fuzzy predictive functional control in the state space domain. *Journal of Intelligent and Robotic Systems*, 31(1–3):283–297, 2001.
116. L X Wang. *Adaptive fuzzy systems and control: Design and Stability Analysis*. Prentice-Hall, Engelwood Cliffs, NJ, 1994.
117. L.-X. Wang and J.M. Mendel. Fuzzy basis functions, universal approximation, and orthogonal least-squares learning. *IEEE Transactions on Neural Networks*, 3(5):807–814, September 1992.
118. S.M. Weiss and C.A. Kulikowski. *Computer Systems that Learn*. Morgan Kaufmann Publishers, San Francisco, 1991.
119. Y. Yoshinari, W. Pedrycz, and K. Hirota. Construction of fuzzy models through clustering techniques. *Fuzzy Sets & Systems*, 54:157–165, 1993.
120. L.A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.