Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Žiga Pušnik

Analiza prostora dopustnih rešitev v visokodimenzionalnih dinamičnih modelih bioloških sistemov

MAGISTRSKO DELO
ŠTUDIJSKI PROGRAM DRUGE STOPNJE
RAČUNALNIŠTVO IN INFORMATIKA

doc. dr. Miha Moškon
MENTOR

Ljubljana, 2018

IZJAVA O AVTORSTVU MAGISTRSKEGA DELA

Spodaj podpisani izjavljam, da sem avtor dela, da slednje ne vsebuje materiala, ki bi ga kdorkoli predhodno že objavil ali oddal v obravnavo za pridobitev naziva na univerzi ali drugem visokošolskem zavodu, razen v primerih, kjer so navedeni viri.
S svojim podpisom zagotavljam, da:

- sem delo izdelal samostojno pod mentorstvom doc. dr. Mihe Moškona,
- so elektronska oblika dela, naslov (slov., ang.), povzetek (slov., ang.) ter ključne besede (slov., ang.) identični s tiskano obliko in
- soglašam z javno objavo elektronske oblike dela v zbirki "Dela FRI".

— Žiga Pušnik, Ljubljana, september 2018.

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Žiga Pušnik

Analiza prostora dopustnih rešitev v visokodimenzionalnih dinamičnih modelih bioloških sistemov

POVZETEK

Sintezna in sistemska biologija sta interdisciplinarni znastveni vedi, ki združujeta področje biologije z matematiko in različnimi vedami inženirstva. Pri preučevanju obstoječih (sistemska biologija) in načrtovanju novih (sintezna biologija) bioloških sistemov se poslužujemo računalniškega modeliranja. Topologija opazovanega sistema je pogosto vnaprej znana. Kinetični parametri, ki so ključni za izvedbo simulacij pa so ponavadi nepoznani ali poznani zgolj delno. To se izkaže za problematično, saj enak model pri različnih naborih kinetičnih parametrov izkazuje bistveno drugačno delovanje. Pri analizi prostora parametrov se pogosto poslužujemo različnih metod iz družine metod za analizo občutljivosti. Žal se te metode izkažejo za problematične, ko je prostor dopustnih rešitev izrazito manjši od celotnega prostora rešitev in kadar ta prostor vsebuje večje število lokalnih ekstremov. V nalogi predstavimo metedologijo za analizo prostora kinetičnih parametrov visokodimenzionalnih bioloških sistemov, ki je primerna tudi za analizo sistemov, pri katerih obstoječe metode odpovedo. Predlagana metodologija temelji na vzorčenju prostora, združevanju dopustnih rešitev in zmanjševanju števila dimenzij v podatkih. Z uporabo metodologije lahko najdemo in analiziramo različne regije, ki se v prostoru dopustnih rešitev pojavljajo, identificiramo najrobustnejše območje ter dobimo občutek o velikosti in povezanosti prostora dopustnih rešitev kinetičnih parametrov. Metodologijo ovrednotimo na modelu biološkega represilatorja in modelu biološke pomnilne celice D s predpomenenjem. Izkaže se, da je prostor kinetičnih parametrov obeh modelov kompleksen, različni parametri pa imajo različno močan vpliv na odziv in delovanje sistema. Vpliv teh parametrov v naših analizah ovrednotimo in izpostavimo tiste, ki so za določeno dinamiko ključni. Pridobljeni rezultati imajo vpliv v širšem biološkem kontekstu, saj omogočajo določitev segmentov sistema, na katere se je potrebno pri nadaljnih raziskavah še posebej osredotočiti.

Ključne besede: modeliranje in simulacija, navadne diferencialne enačbe, represilator, pomnilna celica D s predpomnenjem, vzorčenje, genetski algoritmi, razvrščanje.

University of Ljubljana
Faculty of Computer and Information Science

Žiga Pušnik

Computational analysis of viable parameter spaces in high-dimensional dynamical models of biological systems

ABSTRACT

Synthetic and systems biology combine branches of biology, mathematics and engineering. Computational modelling can be applied to the study of the existing (system biology) and design of new biological systems (synthetic biology). While the topology of the biological system is in many cases well-known, the values of kinetic parameters that are required to perform computational simulations are often missing or known only partially. This can prove to be problematic since the same biological system can exhibit entirely different behaviour with different parameter values. The analysis of parameter space often applies different sensitivity analysis methods. These methods, however, prove to be problematic, when the viable parameter space is much smaller than the whole parameter space, and when the parameter space exhibits several local extrema. We propose a novel computational framework that can also deal with the systems in which existing methods prove to be inefficient. The methodology is based on the exploration of the high-dimensional viable parameter spaces with sampling, clustering and dimensionality reduction techniques. This methodology allows us to efficiently investigate the viable parameter space regions, evaluate the regions which exhibit the largest robustness and gather new insights regarding the size and connectivity of the viable parameter space. We evaluate the proposed computational framework on the repressilator model and the model of biological master-slave D flip-flop. We show that the viable parameter space of both models is complex. Furthermore, different kinetic parameters have a differently strong influence on system behaviour and its dynamics. We evaluate these parameters and show what kind of effects they exhibit. Our results should prove to be useful to the biologists to pinpoint the segments of the system, which need to be focused more precisely to achieve desired dynamics in the context of synthetic biology or to gather new knowledge in the context of systems biology.

Key words: modelling and simulation, ordinary differential equations, repressilator, master-slave D flip-flop, sampling, genetic algorithms, clustering.

ZAHVALA

Na prvem mestu bi se za strokovno pomoč in vodenje pri izdelavi naloge zahvalil mentorju doc. dr. Mihi Moškonu.

Velika zahvala gre tudi prof. dr. Mihi Mrazu, ki me je sprejel v Laboratorij za računalniške strukture in sisteme ter mi ponudil priložnost sodelovanja na mednarodnem študentskem tekmovanju iz sintezne biologije IGEM 2016.

Zahvaliti se moram tudi svoji družini, ki me je v času dosedanjega študija vseskozi podpirala.

Na koncu bi se rad zahvalil še svoji zaročenki, ki me je bodrila in prenašala pri premagovanju marsikaterega problema. Nina, hvala.

— Žiga Pušnik, Ljubljana, september 2018.

KAZALO

Povzetek i

Abstract iii

Zahvala v

1 Uvod 1

1.1 Motivacija . . . 1

1.2 Metodologija . . . 2

1.3 Prispevki znanosti . . . 3

1.4 Pregled naloge . . . 3

2 Gensko regulatorna omrežja 5

2.1 Opis gensko regulatornih omrežij . . . 6

2.2 Modeliranje gensko regulatornih omrežij . . . 9

3 Opis dinamičnih bioloških modelov 13

3.1 Represilator . . . 14

3.2 Model biološke pomnilne celice D s predpomnenjem . . . 16

4 Implementacija 21

4.1 Prostor kinetičnih parametrov . . . 23

4.2 Ocenjevanje kvalitete rešitev . . . 24

4.3 Vzorčenje . . . 29

4.3.1 Enakomerno vzorčenje . . . 30

4.3.2 Vzorčenje s simuliranim ohlajanjem . . . 30

4.3.3 Genetski algoritmi . . . 35

4.4 Razvrščanje 41
4.4.1 Hierarhično razvrščanje 42
4.4.2 Razvrščanje z voditelji 44
4.4.3 Izbira optimalnega števila razredov 45
4.5 Analiza prostora dopustnih rešitev 48
4.5.1 Ocena kvalitete razbitja 48
4.5.2 Prikaz visokodimenzionalnih podatkov 50

5 Rezultati **53**
5.1 Rezultati modela biološkega represilatorja 54
5.2 Rezultati modela biološke pomnilne celice D s predpomnenjem 63

6 Diskusija **73**

7 Zaključek **77**

A Priloga **83**
A.1 Izvorna koda 83

1 Uvod

1.1 Motivacija

Računalniško modeliranje je postalo nepogrešljivo pri preučevanju obstoječih (sistemska biologija) in načrtovanju novih bioloških sistemov (sintezna biologija). Pogost pristop pri modeliranju tovrstnih sistemov je uporaba navadnih diferencialnih enačb prvega reda (ang. *ordinary differential equations, ODEs*) [1–4], pri čemer je pomembno tako dobro poznavanje topologije obravnavanega sistema kot tudi natančnih vrednosti kinetičnih parametrov, ki opisujejo njegovo dinamiko. Prav tako so vrednosti kinetičnih parameterov ključne pri stohastičnem modeliranju [4, 5]. Vrednosti neznanih parametrov eksperimentalnim rezultatom pogosto približujemo z uporabo hevristik, kot so npr. genetski algoritmi [6]. V določenih primerih lahko različno parametrizirani modeli odražajo podobne rezultate. V takšnih primerih poskušamo identificirati robustno rešitev, ki se dejanskim parametrom biološkega sistema najbolj približa. Pri tem se poslužujemo različnih metod za analizo občutljivosti (ang. *sensitivity analysis*) [7].

Pri analizi bioloških modelov bi lahko ubrali sledeč pristop: za posamezen nabor kinetičnih parametrov bi v lokalni okolici perturbirali posamezen parameter in spremljali spremembo njegovega odziva. Takšen pristop bi lahko uvrstili med lokalne metode za analizo občutljivosti [8, 9]. Če nas zanima širše območje delovanja biološkega modela, moramo uporabiti metode iz družine metod za globalno analizo modelov [10]. Tovrstne metode pa se izkažejo za problematične, ko je prostor dopustnih rešitev, tj. rešitev, ki izkazujejo ustrezno dinamiko v kvalitativnem smislu, izrazito manjši od celotnega prostora rešitev. V našem primeru nas zanima predvsem, kako velik je prostor dopustnih rešitev, ali je povezan in katera območja delovanja izkazujejo najrobustnejšo delovanje. Zato v nalogi predlagamo drugačen pristop za analizo prostora dopustnih rešitev v visokodimenzionalnih dinamičnih modelih bioloških sistemov, ki je primerna, če vsebuje prostor dopustnih rešitev večje število lokalnih ekstremov. Ti so pogoj za nepovezanost prostora. S predlagano metodologijo lahko najdemo in analiziramo različne regije, ki se pojavljajo v prostoru rešitev, identificiramo najrobustnejše območje ter dobimo občutek o velikosti in povezanosti prostora dopustnih rešitev kinetičnih parametrov. Predlagano metodologijo ovrednotimo na bioloških modelih, ki odražajo kompleksnejšo dinamiko. Osredotočimo se na model biološke pomnilne celice D s predpomnenjem ter model biološkega represilatorja.

1.2 Metodologija

Analiza vpliva posameznih kinetičnih parametrov na odziv modela nam pove, za katere vrednosti parametrov bo biološki model izkazoval želeno delovanje. V kolikor nas zanima velikost in oblika prostora dopustnih rešitev, moramo ubrati drugačne pristope. Metodologija, ki smo jo v delu predlagali, je primerna, če prostor vsebuje večje število lokalnih ekstremov, kar je tudi pogoj za nepovezanost prostora dopustnih rešitev. S predlagano metodologijo lahko najdemo in analiziramo različne regije, ki se v prostoru dopustnih rešitev pojavljajo, identificiramo najrobustnejše območje ter dobimo občutek o velikosti in povezanosti prostora dopustnih rešitev kinetičnih parametrov.

Z uporabo hevristik, kot so npr. genetski algoritmi, najprej sestavimo prostor dopustnih rešitev, ki ga nato z uporabo različnih tehnik razvrščanja razdelimo na različne regije, ki so lahko tudi nepovezane. Na posameznih regijah lahko zato učinkovito izvedemo analize modelov s katerimi okarakteriziramo robustnost rešitev. Metodologija je

sestavljena iz več zaporednih sklopov. Ti so:

1. vzorčenje prostora dopustnih rešitev,
2. združevanje dopustnih rešitev v posamezne razrede,
3. analiza bioloških modelov znotraj posameznega razreda.

Za implementacijo predlagane metodologije se poslužujemo programskega jezika Python z uporabo ustreznih knjižnjic kot sta DEAP in NumPy. Ker je predlagana metodologija sestavljena iz več manjših sklopov, moramo znotraj posameznega sklopa izbrati najprimernejši algoritem. Tako pri sklopu vzorčenja analiziramo različne tehnike vzorčenja kot so enakomerno vzorčenje, vzorčenje s simuliranim ohlajanjem in vzorčenje z genetskimi algoritmi. Pri sklopu razvrščanja primerjamo hierarhično razvrščanje (ang. *hierarchical clustering*) in razvrščanje z voditelji (ang. *k-means*). Omenjeno metodologijo uporabimo za analizo prostora dopustnih rešitev na že obstoječih deterministčnih bioloških modelih represilatorja in pomnilne celice D s predpomnenjem. Na koncu metodologijo ovrednotimo in predlagamo izboljšave ter možnosti za nadaljnje delo.

1.3 Prispevki znanosti

Prispevki znanosti, ki jih v nalogi predvidevamo, so:

- razvoj nove metodologije za globalno analizo prostora rešitev bioloških modelov in validacija predlagane metodologije na modelih biološke pomnilne celice D s predpomnenjem ter represilatorja,
- poglobljena analiza prostora dopustnih rešitev modela biološkega represilatorja,
- poglobljena analiza prostora dopustnih rešitev modela biološke pomnilne celice D s predpomnenjem.

1.4 Pregled naloge

V poglavju 2 na kratko opišemo kaj so gensko regulatorna omrežja, kako delujejo, kakšne so njihove aplikacije in kako jih modeliramo. Nato v poglavju 3 predstavimo in opišemo modela biološke pomnilne celice D s predpomnenjem in biološkega represilatorja. Poglavje 4 je najobsežnejše, saj vsebuje opis predlagane metodologije. V tem poglavju primerjamo različne pristope vzorčenja prostora dopustnih rešitev ter analiziramo različne

algoritme za razvrščanje in določitev optimalnega števila razredov. Na koncu poglavja predstavimo različne metode za analizo in vizualizacijo prostora dopustnih rešitev. V poglavju 5 predstavimo rezultate in jih ovrednotimo. V poglavju 6 diskutiramo o primernosti predlagane metodologije ter izpostavimo njene prednosti in slabosti. V poglavju 7 podamo zaključke in smernice za nadaljnje delo.

2 Gensko regulatorna omrežja

Gensko regulatorna omrežja ali GRO predstavljajo skupek reakcij v celici biološkega organizma, ki uravnavajo izražanje genov [4, 11]. Prisotna so v celicah vseh živih bitij in so temelj za razvoj organizma, njegovo delovanje in odzivanje na okolje. Sintezna biologija se ukvarja z izgradnjo umetnih gensko regulatornih omrežij, kar ima veliko aplikacij na različnih področjih. Za nas so takšna omrežja zanimiva predvsem iz vidika procesiranja informacij.

V podpoglavju 2.1 na kratko opišemo gensko regulatorna omrežja iz biološkega vidika, ter predstavimo operaciji aktivacije in represije, ki predstavljata ključni del takšnih omrežij. Z uporabo aktivacije in represije lahko sestavimo tudi različna logična vrata, pomnilne celice, števce, oscilatorje ter druge elemente, ki se pojavljajo v digitalnih vezjih.

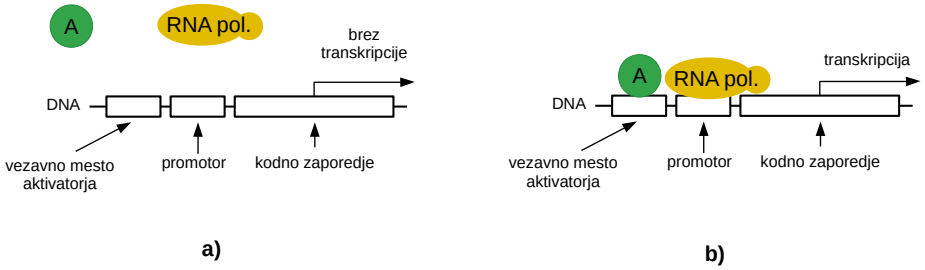
Pristope za modeliranje gensko regulatornih omrežij lahko razdelimo v dve glavni skupini. Ti sta stohastično in deterministično modeliranje [4, 11]. V nalogi se osredotočimo na deterministično modeliranje. V podpoglavju 2.2 predstavimo ter opišemo pristope in metode za deterministično modeliranje gensko regulatornih omrežij.

2.1 Opis gensko regulatornih omrežij

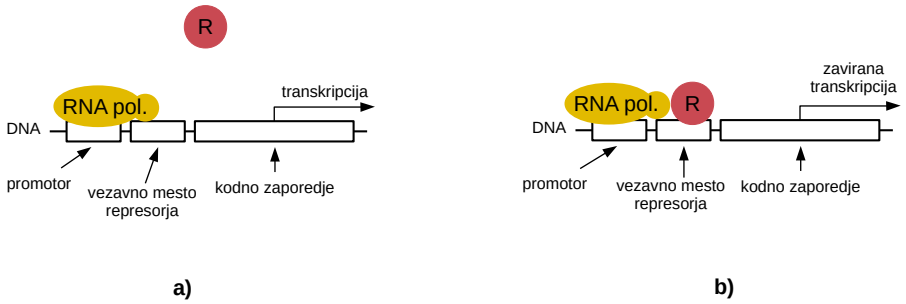
Glavni nosilec informacije v celicah organizma je deoksiribonukleinska kislina DNA [12]. Ta pri ljudeh vsebuje zapis o barvi kože, las in oči, določa spol, višino in vse ostale fiziološke značilnosti, ki nas določajo. Gre za dvojno vijačnico sestavljeno iz štirih osnovnih gradnikov, nukleotidov A, G, T in C. Zaporedje treh nukleotidov lahko predstavlja kodni zapis za aminokisline, iz katerih so sestavljeni proteini. V celici organizma imajo proteini pomembno vlogo, saj pospešujejo ali zavirajo kemijske reakcije, prenašajo molekule v jedro in iz jedra celice ter se odzivajo na zunanje dražljaje. DNA zaporedje, ki kodira protein se imenuje gen [12].

Postopek sinteze proteina iz kodnega DNA zaporedja je sestavljen iz večih korakov. Encim RNA polimeraza se veže na promotor gena in prične postopek genskega prepisovanja ali transkripcije, pri katerem se kodno DNA zaporedje prepiše v verigo ribonukleinske kisline oziroma v informacijsko RNA. To nato celični organel ribosom v postopku genskega prevajanja ali translacije prevede v protein [12].

Omeniti moramo tudi, da lahko proteini pospešujejo ali zavirajo postopek transkripcije. Protein, ki bodisi pospešuje ali zavira postopek genskega preprisovanja imenujemo transkripcijski faktor. Protein, ki pospešuje postopek transkripcije, imenujemo aktivator. Ta se v DNA zaporedju veže na vezavno mesto aktivatorja ter pospešuje vezavo RNA polimeraze na promotor. Če aktivator pospešuje postopek transkripcije, ga represor zavira. Ta se veže na vezavno mesto represorja in zavira vezavo RNA polimeraze na promotor in preprečuje postopek transkripcije. Potek genskega prepisovanja ob prisotnosti aktivatorja in odsotnosti represorja prikazujeta sliki 2.1 ter 2.2. Vendar pa nismo omejeni samo na eno vezavno mesto aktivatorja ali represorja na DNA kodnem zaporedju. V splošnem ima lahko DNA kodno zaporedje različno število vezavnih mest za različne transkripcijske faktorje. Povezani transkripcijski faktorji lahko tudi medsebojno sodelujejo in povečajo afiniteto vezave novih transkripcijskih faktorjev. Takšno obnašanje imenujemo kooperativnost.

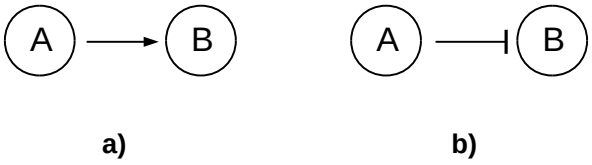


Slika 2.1 Prikaz genskega prepisovanja ob prisotnosti aktivatorja A. Slika a) prikazuje zmanjšano stopnjo transkripcije. Ob odsotnosti aktivatorja je afiniteta vezave RNA polimeraze na promotor nižja, zato je tudi stopnja transkripcije manjša. Slika b) prikazuje višjo stopnjo transkripcije ob prisotnosti aktivatorja. Ob vezavi aktivatorja na vezavno mesto se tudi afiniteta vezave RNA polimeraze na promotor poveča, kar zviša tudi stopnjo transkripcije.



Slika 2.2 Prikaz genskega prepisovanja pri odsotnosti represorja R. Slika a) prikazuje transkripcijo v odsotnosti represorja. Slika b) prikazuje zavirano stopnjo transkripcije ob vezavi represorja na vezavno mesto.

Če je protein A represor ali aktivator pri sintezi proteina B, rečemo tudi, da protein A pospešuje oziroma zavira produkcijo proteina B. Shematski prikaz aktivatorja in represorja je prikazan na sliki 2.3.



Slika 2.3 Shematski prikaz aktivacije in represije. Slika a) prikazuje aktivacijo proteina B. Protein A nastopa kot aktivator. Produkt genskega izražanja ob prisotnosti aktivatorja A je protein B. Slika b) prikazuje represijo proteina B. Protein A nastopa kot represor. Protein B se izraža ob odsotnosti represorja A.

Postopek genskega prepisovanja in prevajanja je ključen del centralne dogme molekularne biologije [12]. Za nas je zanimiv predvsem zaradi zmožnosti procesiranja informacij v celici organizma [4]. DNA zaporedje si lahko predstavljamo kot bralni pomnilnik, na katerem je zapisan program organizma. Proteine, ki se izražajo iz DNA zaporedja, si lahko predstavljamo kot operatorje oziroma izvrševalce delov programa ter kot nosilce končne informacije. Aktivatorji in represorji lahko tudi dostopajo do novih delov programa. Še več, z uporabo aktivacije in represije genov lahko zgradimo tudi različna logična vrata. Logična vrata *ALI* (ang. *OR gate*) bi lahko sestavili z dvema vezavnima mestoma različnih aktivatorjev A in B. V kolikor je na vezavno mesto aktivatorja vezan vsaj eden od proteinov, je stopnja transkripcije visoka, ob odsotnosti aktivatorjev pa je stopnja transkripcije nizka. Na podoben način bi lahko sestavili tudi logična vrata *IN* (ang. *AND gate*). Proteina A in B tvorita dimer, ki se veže na vezavno mesto aktivatorja. V kolikor sta prisotna oba proteina hkrati, bo stopnja transkripcije visoka, ob odsotnosti enega ali obeh proteinov pa je stopnja transkripcije nizka. Na podoben način bi lahko sestavili tudi ostala logična vrata. Z uporabo teh logičnih vrat bi lahko v biološkem sistemu realizirali tudi bolj zapletena digitalna vezja, kot so npr. pomnilne celice, oscilatorji in števci [4, 13].

Z realizacijo logičnih vrat ter ostalih operacij v celicah organizmov se ukvarja sintezna biologija. Ker pa takšni poskusi zahtevajo veliko časa, znanja in opreme, se z modeliranjem najprej prepričamo v teoretično delovanje sistema. Z modeliranjem se lahko prepričamo pri kakšnih parametrih bo sistem izkazoval želeno dinamiko, preverimo lahko kako vpliva perturbacija kinetičnih parametrov na njegovo delovanje in pri kakšnih začetnih pogojih bo sistem izkazoval optimalno delovanje.

2.2 Modeliranje gensko regulatornih omrežij

Pri modeliranju gensko regulatornih omrežij se velikokrat soočamo s kompleksnimi sistemi. To predstavlja veliko nevšečnost, saj v praksi velikokrat ne poznamo hitrosti vseh reakcij, hkrati pa je simuliranje kompleksnejših sistemov časovno potratno. Zato pri izgradnji modela tega poenostavimo do te mere, da je model čim bolj preprost, hkrati pa so rezultati simulacij veljavni in se ujemajo z dejanskim stanjem sistema. Pri tem se opiramo na določene predpostavke. Predpostavimo lahko, da poteka razgradnja vseh proteinov z enako hitrostjo, ali pa, da se stanje določene kemijske zvrsti po času ne spreminja in je v ravnovesnem stanju. Takšno predpostavko imenujemo kvazi-ravnovesni pristop.

Modeliranje gensko regulatornih omrežij delimo na dve glavni skupini. Imenujeta se stohastično in deterministično modeliranje [4, 11].

Pri stohastičnem modeliranju predstavlja naš model seznam vseh kemijskih reakcij sistema ter njihovih hitrosti. V časovnem koraku izberemo reakcijo, ki se bo izvedla, na podlagi njene verjetnosti. Ta verjetnost je odvisna od hitrosti kemijske reakcije, velikosti časovnega koraka, ter količine produktov, ki v reakciji nastopajo. Prednost takšnega modeliranja je, da v simulacije vnaša šum, zato so rezultati simulacij biološko relevantnejši. Slabost te metode je, da je sorazmeroma počasna. Glavni predstavnik algoritmov za stohastično modeliranje je algoritem SSA (ang. *stohastic simulation algorithm*) [4, 5].

Za razliko od stohastičnega modeliranja, je deterministično modeliranje veliko hitrejše, a v sistem ne vnaša dodatnega šuma. Pri determinističnem modeliranju predstavlja naš model sistem diferencialnih enačb prvega reda [11]:

$$\frac{d\vec{P}}{dt} = F\left(\vec{P}_t, t\right), \tag{2.1}$$

kjer predstavlja F sistem diferencialnih enačb, vektor $d\vec{P}$ količino vseh kemijskih zvrsti modela, $\frac{d\vec{P}}{dt}$ pa predstavlja njihovo spremembo oziroma njihov odvod po času.

Takšen sistem bi lahko reševali analitično, vendar se v praksi največkrat poslužujemo numeričnih pristopov. Najpreprostejša metoda za numerično reševanje diferencialnih enačb je Eulerjeva metoda [14]. V času t in stanju sistema $\vec{P}$ s sistemom diferencialnih enačb izračunamo spremembo kemijskih zvrsti, ter jo sorazmerno z velikostjo časovnega koraka dt prištejemo k trenutni količini kemijskih zvrsti:

$$\vec{P}_{t+dt} = \vec{P}_t + F\left(\vec{P}_t, t\right) * dt. \tag{2.2}$$

Ker gre za metodo prvega reda je napaka, ki jo pridelamo v časovnem koraku $t + dt$, sorazmerna s kvadratom časovnega koraka: $err \propto dt^2$ [14]. Če želimo torej zmanjšati napako, moramo zmanjšati tudi časovni korak. Ker pa se z manjšanjem časovnega koraka poveča časovna zahtevnost numerične metode, je Eulerjeva metoda za težje probleme nepraktična. Zato se poslužujemo numeričnih metod višjega reda. Ena takšnih metod je metoda Runge–Kutta 4. reda [14]:

$$\begin{aligned}
k_1 &= dt * F\left(\vec{P}_t, t\right), \\
k_2 &= dt * F\left(\vec{P}_t + \frac{k_1}{2}, t + \frac{dt}{2}\right), \\
k_3 &= dt * F\left(\vec{P}_t + \frac{k_2}{2}, t + \frac{dt}{2}\right), \\
k_4 &= dt * F\left(\vec{P}_t + k_3, t + dt\right), \\
\vec{P}_{t+dt} &= \vec{P}_t + \frac{k_1 + 2 * k_2 + 2 * k_3 + k_4}{6}.
\end{aligned} \tag{2.3}$$

Napaka, ki jo pridelamo v časovnem koraku $t + dt$ je sorazmerna s peto potenco velikosti časovnega koraka [14]. To pomeni, da si lahko v primerjavi z Eulerjevo metodo privoščimo veliko večji časovni korak, hkrati pa pridelamo manjšo napako pri numeričnem reševanju enačb.

Metoda Runge–Kutta 4. reda deluje dobro v praksi, vendar je njena slabost ta, da uporablja za numerično reševanje vedno enako velik časovni korak dt. Če je sprememba kemijske zvrsti po času nizka, bi si lahko privočili veliko večji korak za ceno enake napake, hkrati pa bi morali časovni korak dt povečati pri porastu spremembe kemijske zvrsti. Lahko bi tudi uporabili hitrejšo, a manj natančno metodo kadar je problem, ki ga rešujemo manj zahteven in počasnejšo, a zanesljivejšo metodo, kadar je problem težji.

V nalogi za realizacijo uporabimo programski jezik *Python*. Če v programski jezik *Python* uvozimo paket *integrate* knjižnjice *SciPy*, lahko z uporabo funkcije *odeint* numerično rešujemo sisteme diferencialnih enačb prvega reda. Funkcija *odeint* uporablja za numerično reševanje diferencialnih enačb metodo *LSODA* [15], ki z uporabo dinamičnega časovnega koraka in izmenjevanjem počasnih, a natančnih ter hitrih, toda nekoliko manj natančnih metod, optimizira natančnost in hitrost izvajanja numeričnega reševanja sis-

tema diferencialnih enačb prvega reda. Prednost metode *LSODA* je, da je hitra in hkrati natančna. Ker pa za določanje težavnosti problema uporablja Jakobijevo matriko [15], gre za matriko parcialnih odvodov po vseh prostih spremenljivkah, metoda *LSODA* odpove, če sistem diferencialnih enačb ni odvedljiv. Takrat se poslužujemo preprostejših metod, kot je npr. metoda Runge-Kutta 4. reda 2.3.

3 Opis dinamičnih bioloških modelov

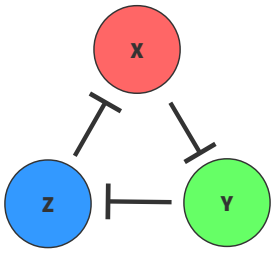
V nalogi se bomo osredotočili na dinamične biološke sisteme, ki izkazujejo kompleksnejšo dinamiko. To so oscilatorji in števci.

V sinhronih digitalnih vezjih je generator pravokotnega signala izrednega pomena, saj sinhronizira prehode pomnilnih celic iz nizkega v visoko stanje in obratno. Prav tako je za organizem ključen pomen motivov, ki izkazujejo oscilatorno dinamiko [16]. Narekujejo prilagajanje organizma na periodično spreminjanje razmer v okolju in regulirajo notranjo biološko uro organizma, ki uravnava biološke procese kot so celično signaliziranje, izražanje genov in metabolizem [17]. Ritme s periodo 24 h, ki uravnavajo dnevno-nočni cikel imenujemo cirkadiani ritmi [17].

V nadaljevanju opišemo modela dveh bioloških sistemov, na katera se bomo osredotočili v našem delu. To sta model represilatora (razdelek 3.1) in model biološke pomnilne celice D s predpomnjenjem (razdelek 3.2).

3.1 Represilator

Represilator je preprosto gensko regulatorno omrežje, ki med drugim predstavlja tudi eno izmed prvih delujočih aplikacij sintezne biologije [18]. Sestavljen je iz lihega števila represorjev, ki v povezani zanki zavirajo izražanje sosednjih proteinov [4]. Najpreprostejši model represilatorja je Goodwinov model, ki je sestavljen iz ene same negativne povratne zanke. Čeprav je za oscilatorno delovanje teoretično dovolj le en represor v negativni povratni zanki, se izkaže, da so v realnosti takšne oscilacije vselej dušene. Zato je tipičen represilator sestavljen iz treh represorjev. Slika 3.1 prikazuje shemo biološkega represilatorja s tremi represorji povezanimi v povratno zanko.



Slika 3.1 Shema tipičnega biološkega represilatorja s tremi represorji v povezani zanki, pri čemer so x, y in z proteini, ki zavirajo produkcijo sosednjih proteinov v zanki.

S sledečimi enačbami, lahko opišemo deterministično kinetiko represilatorja [4, 11]:

$$\frac{dm_x}{dt} = \alpha_{0_x} + \frac{\alpha_x}{1 + \left(\frac{[z]}{Kd}\right)^n} - [m_x] * \delta_{m_x}, \tag{3.1}$$

$$\frac{dm_y}{dt} = \alpha_{0_y} + \frac{\alpha_y}{1 + \left(\frac{[x]}{Kd}\right)^n} - [m_y] * \delta_{m_y}, \tag{3.2}$$

$$\frac{dm_z}{dt} = \alpha_{0_z} + \frac{\alpha_z}{1 + \left(\frac{[y]}{Kd}\right)^n} - [m_z] * \delta_{m_z}, \tag{3.3}$$

$$\frac{dx}{dt} = \beta_x * [m_x] - \delta_x * [x], \tag{3.4}$$

$$\frac{dy}{dt} = \beta_y * [m_y] - \delta_y * [y], \tag{3.5}$$

$$\frac{dz}{dt} = \beta_z * [m_z] - \delta_z * [z]. \tag{3.6}$$

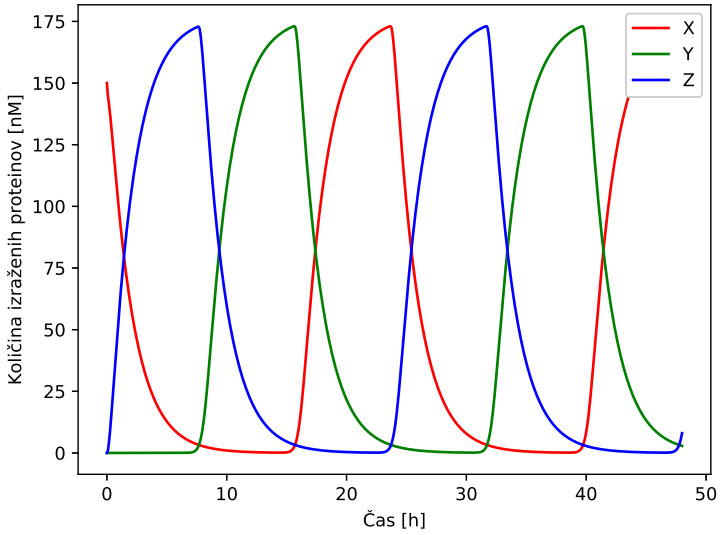
V tabeli 3.1 so opisane spremenljivke modela, kinetični koeficienti in ostali parametri, ki jih potrebujemo pri modeliranju biološkega represilatorja.

Tabela 3.1 Tabela enot in opisov spremenljivk, kinetičnih parametrov ter ostalih koeficientov preprostega biološkega represilatorja.

Faktor	Opis	Enota
x, y, z	koncentracija proteina	$[nM]$
$\frac{dx}{dt}$, $\frac{dy}{dt}$, $\frac{dz}{dt}$	sprememba proteina po času	$[nMs^{-1}]$
m_x, m_y, m_z	koncentracija informacijske RNA	$[nM]$
$\frac{dm_x}{dt}$, $\frac{dm_y}{dt}$, $\frac{dm_z}{dt}$	sprememba informacijske RNA po času	$[nMh^{-1}]$
n	Hillov koeficient	-
K_d	disociacijska konstanta represorja	$[nM]$
α_{0_x}, α_{0_y}, α_{0_z}	stopnja brezpogojnega izražanja informacijske RNA	$[nMh^{-1}]$
α_x, α_y, α_z	stopnja izražanja informacijske RNA ob prisotnosti represorja	$[nMh^{-1}]$
δ_{m_x}, δ_{m_y}, δ_{m_z}	stopnja degradacije informacijske RNA	$[h^{-1}]$
β_x, β_y, β_z	stopnja translacije	$[h^{-1}]$
δ_x, δ_y, δ_z	stopnja degradacije proteina	$[h^{-1}]$

V našem modelu biološkega represilatorja naredimo še dodatne predpostavke. Predpostavimo naslednje: vse informacijske RNA imajo enako stopnjo brezpogojnega izražanja ($\alpha_{0_x} = \alpha_{0_y} = \alpha_{0_z} = \alpha_0$), vse informacijske RNA imajo enako stopnjo izražanja ob prisotnosti represorja ($\alpha_x = \alpha_y = \alpha_z = \alpha$), vsi proteini imajo enako hitrost translacije ($\beta_x = \beta_y = \beta_z = \beta$), hitrost degradacije je enaka za vse informacijske RNA ($\delta_{m_x} = \delta_{m_y} = \delta_{m_z} = \delta_m$) ter za vse proteine ($\delta_x = \delta_y = \delta_z = \delta_p$).

Simulacije, ki jih poganjamo za model represilatorja so dolge 48 h, s časovnim korakom $dt = 0{,}001\ h = 3{,}6\ s$, z numerično metodo $LSODA$. Primer simulacijskih rezultatov sistema (3.1 - 3.6) prikazuje slika 3.2.



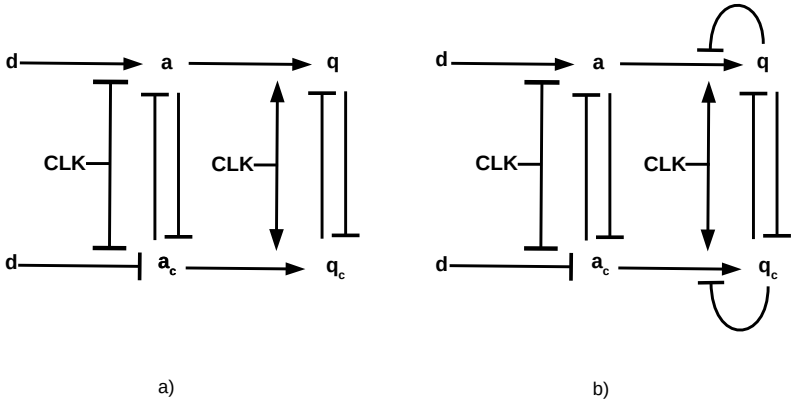
Slika 3.2 Simulacija represilatorja za $t = 48\ h$ s časovnim korakom 3,6 s pri začetnih pogojih $m_x = 0\ nM$, $x = 150\ nM$, $m_y = 0\ nM$, $y = 0\ nM$, $m_z = 0\ nM$, $z = 0\ nM$ s kinetičnimi parametri $\alpha = 10\ nMh^{-1}$, $\alpha_0 = 2 * 10^{-5}$ nMh^{-1}, $n = 8{,}9$, $\beta = 37{,}5\ h^{-1}$, $\delta_m = 4{,}14\ h^{-1}$, $\delta_p = 0{,}51\ h^{-1}$, $Kd = 2{,}94\ nM$. Oscilator prične takoj oscilirati, saj je začetna vrednost proteina x visoka, medtem ko sta vrednosti proteinov y in z nizki.

3.2 Model biološke pomnilne celice D s predpomnenjem

Pomnilna celica D (ang. *delay*) je v elektroniki preklopno vezje, ki ima dve stabilni stanji in lahko hrani en bit informacije. Izhod pomnilne celice D je odvisen od vhoda v prejšnji časovni enoti in je zakasnjen za eno urino periodo [13].

Preprosto pomnilno celico D lahko prožimo na visok ali nizek nivo urinega signala. Do težav pride, če urina perioda ni usklajena z vhodnim signalom ali s spremembo vhodnega signala pomnilne celice D, kar lahko povzroči nedefinirano delovanje. To je pomembno predvsem v bioloških sistemih, kjer perioda oscilacij ni konstanta in lahko variira glede na začetne pogoje, zunanje vplive in stohastičnost kemijskih reakcij. Zato vpeljemo pomnilno celico D s predpomnjenjem (ang. *Master-Slave*), kjer zaporedno vežemo dve celici D, pri čemer prožimo prvo celico npr. na nizek nivo urinega signala ter drugo na visok nivo urinega signala. Takšna pomnilna celica D se proži samo ob pozitivni urini fronti, zato je takšen način delovanja primeren tudi za dinamične biološke sisteme [13]. Slika 3.3 (a) prikazuje topologijo navadne pomnilne celice D s predpomnjenjem. Slika 3.3 (b) prikazuje topologijo izboljšane pomnilne celice D s predpomnjenjem, primerni za

izvedbo v gensko regulatornih omrežjih. Izhod izhodne pomnilne celice (ang. *Slave*) je omejen z negativno povratno zanko, kar dodatno regulira količino izhodnih proteinov q in q_c [13].



Slika 3.3 Slika a) prikazuje shemo pomnilne celice D s predpomnjenjem. d predstavlja vhodni protein, CLK kontrolni protein, a in a_c notranje stanje celice in q ter q_c izhodna proteina. Slika b) prikazuje shemo izboljšane pomnilne celice D, ki je primerna za izvedbo v bioloških sistemih.

Tudi tukaj naredimo podobne predpostavke kot pri modelu biološkega represilatorja, opisanem v podpoglavju 3.1. Zaradi kompleksnosti topologije pomnilne celice D s predpomnjenjem, deterministični model še dodatno poenostavimo, tako da predpostavimo neskončen Hillov koeficient oziroma neskončno kooperativnost transkripcijskih faktorjev. Z uporabo kvazi-ravnovesnega pristopa predpostavimo še, da se količina informacijske RNA po času spreminja veliko hitreje kot količina proteinov, zato lahko njen odvod po času enačimo z 0. Tako lahko spremembo produkcije proteina enostavno modeliramo z Heavisideovo skočno funkcijo:

$$H[x] = \begin{cases} 1, & \text{če } x \geq 0, \\ 0, & \text{sicer}, \end{cases} \tag{3.7}$$

pri čemer $x \geq 0$ predstavlja pogoj za izražanje proteina.

Z naslednjimi enačbami lahko opišemo deterministično kinetiko izboljšane pomnilne celice D s predpomnjenjem:

$$\frac{da}{dt} = \alpha_1 * \Theta(d - Kd_1) * \Theta(Kd_2 - CLK) + \alpha_2 * \Theta(Kd_3 - a_c) - \delta_1 * a, \tag{3.8}$$

$$\frac{da_c}{dt} = \alpha_1 * \Theta(Kd_1 - d) * \Theta(Kd_2 - CLK) + \alpha_2 * \Theta(Kd_3 - a) - \delta_1 * a_c, \tag{3.9}$$

$$\frac{dq}{dt} = \alpha_3 * \Theta(a - Kd_4) * \Theta(CLK - Kd_5) * \Theta(Kd_7 - q) \tag{3.10}$$

$$+ \alpha_4 * \Theta(Kd_6 - q_c) * \Theta(Kd_7 - q) - \delta_2 * q, \tag{3.11}$$

$$\frac{dq_c}{dt} = \alpha_3 * \Theta(a_c - Kd_4) * \Theta(CLK - Kd_5) * \Theta(Kd_7 - q_c) \tag{3.12}$$

$$+ \alpha_4 * \Theta(Kd_6 - q) * \Theta(Kd_7 - q_c) - \delta_2 * q_c. \tag{3.13}$$

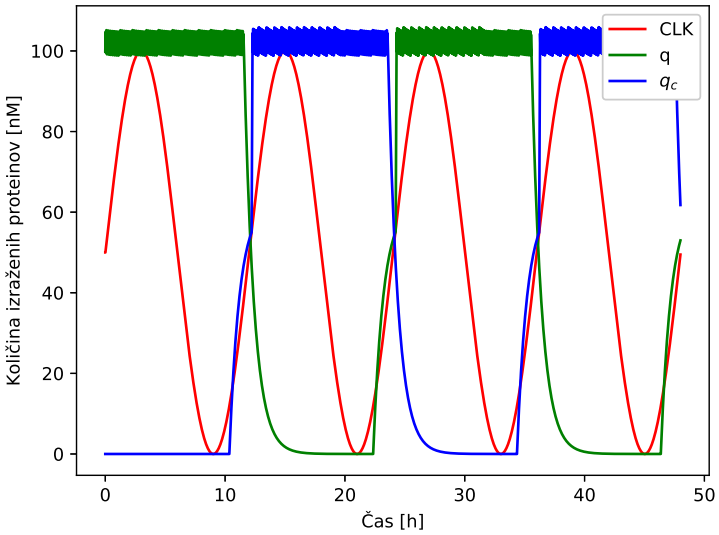
V tabeli 3.2 so opisane spremenljivke modela, kinetični koeficienti in ostali parametri, ki jih potrebujemo pri determinističnem modeliranju izboljšane pomnilne celice D s predpomnjenjem. Omeniti moramo, da so zaradi uporabe kvazi-ravnovesnega pristopa parametri $\alpha_1, \alpha_2, \alpha_3$ in α_4 sestavljeni, ter vsebujejo stopnjo transkripcije in degradacije informacijske RNA ter stopnjo translacije: $\alpha_i = \frac{\alpha_{m_i} * \beta_p}{\delta_{m_i}}$, kjer je α_i stopnja produkcije i-tega proteina, β_p stopnja genskega prevajanja i-tega proteina, α_{m_i} stopnja genskega prepisovanja informacijske RNA ter δ_{m_i} stopnja degradacije informacijske RNA.

Tabela 3.2 Tabela enot in opisov spremenljivk, kinetičnih parametrov ter ostalih koeficientov modela biološke D pomnilne celice v vezavi gospodar-suženj.

Faktor	Opis	Enota
CLK, d, a, a_c, q, q_c	koncentracija proteina	$[nM]$
$\frac{da}{dt}$, $\frac{da_c}{dt}$, $\frac{dq}{dt}$, $\frac{dq_c}{dt}$	sprememba proteina po času	$[nMh^{-1}]$
α_1, α_2, α_3, α_4	stopnja produkcije proteina	$[nMh^{-1}]$
Kd_1, Kd_2, Kd_3, Kd_4, Kd_5, Kd_6, Kd_7	inhibicijska/aktivacijska meja aktivatorja/represorja	$[nM]$
δ_1, δ_2	stopnja degradacije proteina	$[h^{-1}]$

Dolžina simulacije modela pomnilne celice D s predpomnenjem znaša 48 h s časovnim korakom $dt = 0{,}01\ h = 36\ s$. Urin signal s peridodo 12 h in amplitudo 100 nM predstavlja protein CLK, ki ga modeliramo z navadno sinusno funkcijo. Ker uporabimo v sistemu diferencialnih enačb (3.8 - 3.13) Heavisideovo funkcijo, metoda $LSODA$ odpove. Zato uporabimo za računanje vrednosti proteinov a, a_c, q ter q_c numerično metodo

Runge-Kutta 4. reda. Slika 3.4 prikazuje delovanje pomnilne celice D s predpomnjenjem, povezane z negativno povratno zanko, kjer je negiran izhod celice Q povezan nazaj na vhod celice D. Vezava predstavlja primer implementacije 1-bitnega števca.

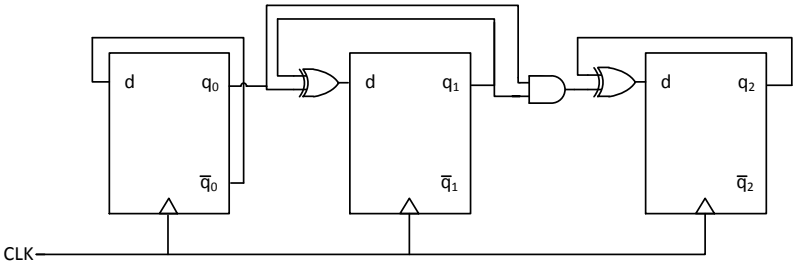


Slika 3.4 Deterministična simulacija pomnilne celice D s predpomnjenjem z negativno povratno zanko. Dolžina simulacije je 48 h s časovnim korakom 36 s ter dolžino urine periode 12 h in amplitudo 100 nM pri začetnih pogojih: $a = 0\ nM$, $a_c = 0\ nM$, $q = 100\ nM$, $q_c = 0\ nM$ ter sledečih kinetičnih parametrih: $\alpha_1 = 512{,}44\ nMh^{-1}$, $\alpha_2 = 179{,}23\ nMh^{-1}$, $\alpha_3 = 72{,}52\ nMh^{-1}$, $\alpha_4 = 634{,}34\ nMh^{-1}$, $K_{d_1} = 79{,}54\ nM$, $K_{d_2} = 7{,}75\ nM$, $K_{d_3} = 0{,}01895\ nM$, $K_{d_4} = 32{,}29\ nM$, $K_{d_5} = 11{,}91\ nM$, $K_{d_6} = 47\ nM$, $K_{d_A} = 100\ nM$, $\delta_1 = 4{,}11\ h^{-1}$, $\delta_2 = 1{,}17\ h^{-1}$.

Z n pomnilnimi celicami D lahko pomnimo 2^n bitov informacije, zato lahko z dodatno logiko realiziramo navaden n-bitni števec. Dodatna logika, ki jo potrebujemo je:

$$\begin{aligned} q_0 &\Leftarrow \overline{q_0}, \\ q_1 &\Leftarrow q_0 \oplus q_1, \\ q_2 &\Leftarrow (q_0 \wedge q_1) \oplus q_2, \\ &\ldots, \end{aligned} \tag{3.14}$$

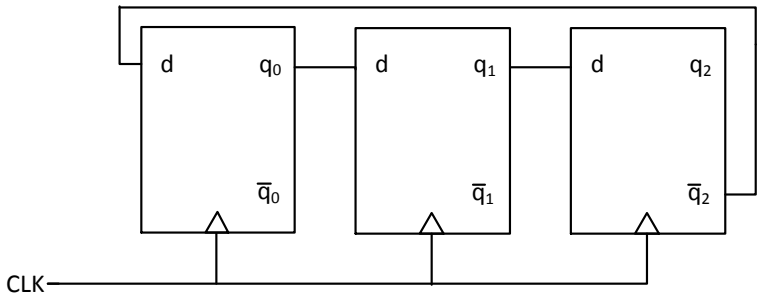
kjer je q_i notranje stanje pomnilne celice i, $\overline{q_i}$ pa njena negirana vrednost. Shema tribitnega števca je prikazana na sliki 3.5.



Slika 3.5 Logična shema navadnega tribitnega števca.

Glavni problem takšnega števca je v tem, da vpeljuje še dodatno logiko, ki otežuje realizacijo sistema v biološkem gostitelju, kot je npr. bakterija Escherichia coli, saj povečuje potrebo po številu ortogonalnih proteinov. Zato za realizacijo števca v bioloških sistemih v [13] predlagamo Johnsonov števec. Tak števec dobimo tako, da v kaskado povežemo n pomnilnih celic D, kjer je izhod celice i povezan na vhod celice $i+1$. Z negativno povratno zanko povežemo izhod izhodne pomnilne celice na vhod vhodne celice. Logična shema tribitnega Johnsonovega števca je prikazana na sliki 3.6. Takšen števec v neskončni zanki ponavlja $2n$ različnih kombinacij vrednosti [13]. Johnsonov števec s tremi biti bo tako ponavljal sledeče kombinacije vrednosti:

- $(0,0,0) \rightarrow (1,0,0) \rightarrow (1,1,0) \rightarrow (1,1,1) \rightarrow (0,1,1) \rightarrow (0,0,1) \rightarrow (0,0,0) \ldots$



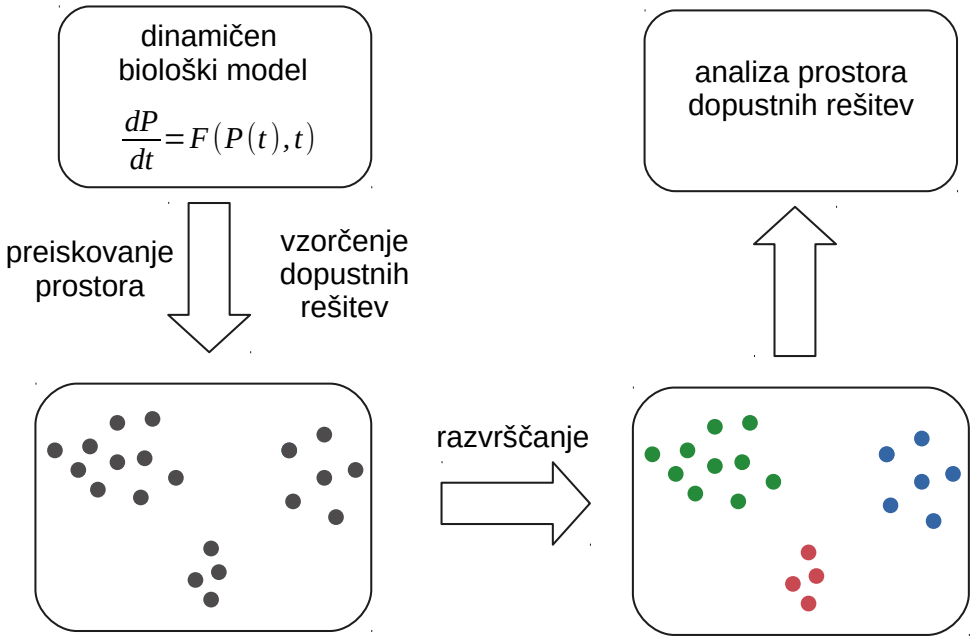
Slika 3.6 Logična shema tribitnega Johnsonovega števca.

Johnsonov števec predstavlja kompromis med kompleksnostjo vezja in modulom štetja, zato je primeren za izvedbo in uporabo v biološkem sistemu [13].

4 Implementacija

Pri analizi bioloških modelov nas pogosto zanima robustnost delovanja za podan nabor kinetičnih parametrov. Velikokrat želimo predvsem izvedeti, v kakšni meri in kako posamezen kinetičen parameter vpliva na želeni odziv sistema. V našem primeru obravnavamo modele bioloških sistemov, ki izkazujejo oscilatorno dinamiko, zato nas zanimata amplituda in perioda oscilacij. Analize bi se lahko lotili z uporabo lokalnih metod za analizo občutljivosti [8, 9]. Ker pa nas zanima širše območje delovanja biološkega modela, moramo uporabiti metode iz družine metod za globalno analizo občutljivosti [10]. Tovrstne metode se izkažejo za problematične, ko je prostor dopustnih rešitev izrazito manjši od celotnega prostora rešitev. V tem primeru nas zanima predvsem, kako velik je prostor dopustnih rešitev, ali je sestavljen iz večih regij in katera območja delovanja izkazujejo najoptimalnejše delovanje.

Metoda za analizo bioloških sistemov s slabo povezanim prostorom dopustnih rešitev, ki jo predlagamo v tej nalogi, je sestavljena iz večih korakov. Prostora dopustnih rešitev ne poznamo, zato ga moramo najprej preiskati in najdene dopustne rešitve ustrezno vzorčiti. Ker pričakujemo, da se bodo rešitve porazdelile okoli lokalnih ekstremov, te nato z algoritmom razvrščanja razvrstimo v posamezne razrede in ovrednotimo kvaliteto razbitja. Na koncu analiziramo prostor dopustnih rešitev nad posameznimi razredi. Zanima nas predvsem kvaliteta rešitev znotraj posameznih razredov, ki jo lahko kvantitativno vrednostimo z amplitudo in periodo oscilacij in njunim standardnim odklonom. Optimalen razred je močno zastopan in ima veliko število rešitev, ki so po prostoru čim bolj razpršene, njegovi osebki pa izkazujejo visoko povprečno amplitudo in periodo oscilacij ter nizek standardni odklon. Slika 4.1 prikazuje vizualizacijo predlagane metodologije.



Slika 4.1 Vizualen prikaz predlagane metodologije.

Prostor kinetičnih parametrov moramo najprej dobro poznati. V podpoglavju 4.1 opišemo kakšni kinetični parametri sestavljajo prostor naših bioloških modelov in kako so omejeni. Znati moramo ločiti med boljšimi in slabšimi rešitvami. V podpoglavju 4.2 opišemo kakšen kriterij uporabimo za ocenjevanje kvalitete dopustnih rešitev. V podpoglavju 4.3 opišemo različne pristope, s katerimi lahko preiskujemo in vzorčimo prostor dopustnih rešitev ter izberemo najprimernejši algoritem. V podpoglavju 4.4 opišemo različne algoritme razvrščanja, izberemo najprimernejšega in opišemo kako določamo kvaliteto razbitja.

4.1 Prostor kinetičnih parametrov

Ker obravnavamo biološke modele, moramo prostor kinetičnih parametrov omejiti tako, da bodo rešitve znotraj omejenega prostora biološko relevantne. Zato iščemo optimalno rešitev ali vsaj njen približek v omejenem n-dimenzionalnem prostoru kinetičnih parametrov.

$$\begin{aligned} \hat{K} &= [k_1, k_2, k_3, ...k_n], \\ k_{1min} &\leq k_1 \leq k_{1max}, \\ k_{2min} &\leq k_2 \leq k_{2max}, \\ k_{3min} &\leq k_3 \leq k_{3max}, \\ &... \\ k_{nmin} &\leq k_n \leq k_{nmax}. \end{aligned} \tag{4.1}$$

Tipi kinetičnih parametrov, ki jih v nalogi obravnavamo so: hitrost genskega prepisovanja ali transkripcije, hitrost genskega prevajanja ali translacije, hitrost produkcije proteina, hitrost degradacije informacijske RNA, hitrost degradacije proteina, Hillov koeficient in disociacijska konstanta aktivatorja ali represorja. Referenče vrednosti parametrov prikazuje tabela 4.1 [19]. Kinetični parametri se lahko znatno razlikujejo glede na vrsto organizma, vrsto kemijske zvrsti, zunanje vplive ter ostale dejavnike. Pri parametrih transkripcije, translacije in produkcije proteina vzamemo za minimalno vrednost k_{min}, ki jo kinetični parameter lahko zavzame, referenčno vrednost parametra pomnoženo s faktorjem 10^{-3}, za maksimalno vrednost k_{maks} pa vzamemo referenčno vrednost pomnoženo s faktorjem 10^3. Pri degradaciji proteinov ter informacijske RNA, Hillovem

koeficientu in disociacijski konstanti vzamemo za minimalno vrednost k_{min} referenčno vrednost parametra pomnoženo s faktorjem 10^{-2}, za maksimalno vrednost k_{maks} pa vzamemo referenčno vrednost pomnoženo s faktorjem 10^2.

Tabela 4.1 Tabela opisov kinetičnih parametrov in njihovih referenčnih vrednosti.

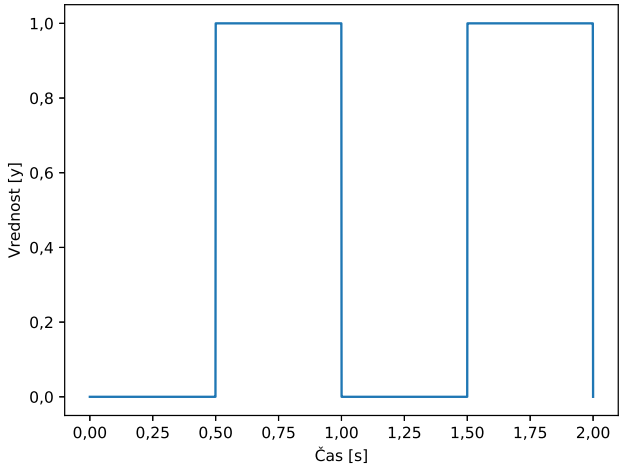
Opis	Referenčna vrednost	Enota
stopnja transkripcije	0,02	$[nMh^{-1}]$
stopnja translacije	0,075	$[h^{-1}]$
stopnja produkcije proteina	0,75	$[nMh^{-1}]$
stopnja degradacije RNA	2	$[h^-1]$
stopnja degradacije proteina	0,1	$[h^{-1}]$
Hillov koeficient	1	–
disociacijska konstanta	100	$[nM]$

4.2 Ocenjevanje kvalitete rešitev

Pri preiskovanju prostora potrebujemo kriterij, glede na katerega bomo ocenjevali kvaliteto posameznih rešitev. Takšen kriterij imenujemo cenovna funkcija. Preprosto cenovno funkcijo lahko opišemo z enačbo:

$$c(\hat{x}) = \frac{1}{|\hat{x}|}\left(\sum_{i=1}^{|\hat{x}|}(\hat{x}_i - h_i)^2\right), \tag{4.2}$$

kjer je $\hat{x}$ vhodni oscilatorni signal, h pa idealni oscilatorni signal, ki ga želimo doseči. Bolj se bo signal x prilegal idealnemu signalu h, boljša bo rešitev in nižja bo vrednost cenovne funkcije. Tipičen primer takšnega signala je pravokotna periodična funkcija (ang. *square wave function*), ki je prikazana na sliki 4.2. Takšna cenovna funkcija je občutljiva na amplitudo in periodo vhodnega signala, saj kaznuje vhodne signale, ki imajo sicer različno periodo in amplitudo od idealnega referenčnega signala, čeprav le-ti izkazujejo želeno oscilatorno dinamiko.



Slika 4.2 Periodična funkcija pravokotnega signala s periodo in amplitudo 1.

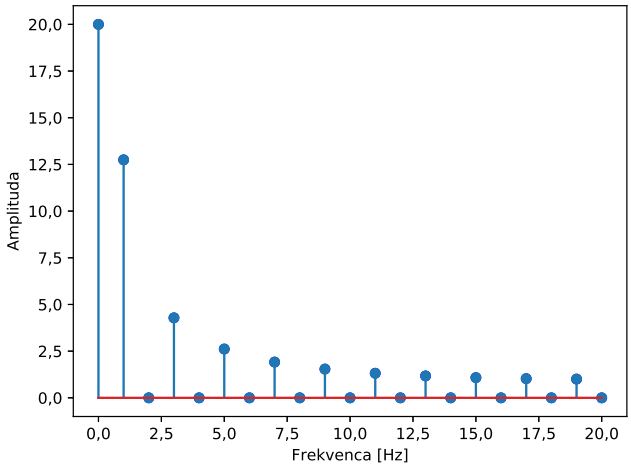
Takšna cenovna funkcija je torej primerna, če želimo doseči točno določeno amplitudo in periodo oscilacij. V primeru pomnilne celice D s predpomnenjem, se mora sprememba stanja prožiti na pozitivno fronto urinega signala, zato je v tem primeru ta cenovna funkcija primerna izbira. Za idealni signal h izberemo pravokotno periodično funkcijo s periodo 24 h in amplitudo 100 nM, saj želimo v primeru modela pomnilne celice D s predpomnjenjem doseči oscilatorno obnašanje s periodo 24 h in amplitudo 100 nM. Ker pa bi bilo nesmiselno primerjati idealen odziv h z odzivom modela biološke pomnilne celice za vsak časovni korak dt, signal najprej še vzorčimo tako, da obdržimo vsak 8. časovni korak. Ker ta znaša 36 s, bomo torej vzorčili s frekvenco 0,00347 Hz. To je približno 1 vzorec na vsakih 5 min ali natančneje 1 vzorec na vsakih 4,8 min. Nov signal, ki ga primerjamo z idealnim odzivom h, zato vsebuje 600 točk.

V primeru modela represilatorja nas zanima širše območje delovanja. Želimo optimizirati amplitudo in periodo oscilacij, vendar smatramo rešitev kot dobro, če le-ta izkazuje oscilatorno dinamiko ne glede na periodo in amplitudo oscilacij. Želimo torej takšno cenovno funkcijo, ki je neodvisna od amplitude ali periode oscilatornega signala. Če pa želimo takšno cenovno funkcijo sestaviti, moramo najprej poznati lastnosti idealnega oscilatornega signala. Za primer vzemimo periodično funkcijo pravokotnega signala prikazano na sliki 4.2. Takšna funkcija ima sledeče lastnosti:

- neskončno hitro pozitivno fronto,
- neskončno hitro negativno fronto,
- ničelno standardno deviacijo v visokem in nizkem stanju.

Zanimive lastnosti se pokažejo v frekvenčnem prostoru. Z uporabo Fourierjeve vrste lahko analitično določimo vse harmonike pravokotnega signala. Enačba (4.3) prikazuje izračun n-tega harmonika, kjer je a_n n-ti harmonik, A amplituda, T urina perioda, t_p pa širina pulza pravokotnega signala. Opazimo lahko, da so ničelni in neničelni harmoniki enakomerno razporejeni, ter da neničelni harmoniki sledijo racionalni funkciji $\frac{1}{n}$, kar prikazuje tudi slika 4.3. Pri signalih v bioloških sistemih je ta padec bolj linearen zaradi daljšega prehoda signala iz nizkega v visoko stanje in obratno. Poleg tega moramo v bioloških signalih upoštevati še nedeterminističnost kemijskih reakcij, zaradi česar je lahko razporeditev harmonikov še dodatno popačena, zato moramo pri izbiri cenovne funkcije upoštevati tudi stohastičnost signalov.

$$a_n = 2\frac{A}{n\pi} sin\left(n\pi\frac{t_p}{T}\right) \tag{4.3}$$



Slika 4.3 Odziv pravokotnega signala s periodo $1\ s$ in amplitudo 1 v frekvenčnem prostoru s frekvenco vzorčenja $40\ Hz$. Vzorce smo zajemali 1 sekundo s frekvenco vzorčenja $40\ Hz$, zato je najvišja frekvenca, ki jo lahko zaznamo enaka $20\ Hz$. Opazimo lahko, da vsi lihi harmoniki sledijo racionalni funkciji $\frac{1}{n}$, pri čemer n predstavlja n-ti harmonik. Vsi sodi harmoniki so ničelni. Ničti harmonik s frekvenco 0 ima neskončno periodo, zato si ga lahko predstavljamo kot povprečje signala.

Sestaviti želimo takšno cenovno funkcijo, ki bo neodvisna od amplitude in periode, a bo vseeno favorizirala oscilacije z višjo amplitudo nihanja in krajšim prehodom iz visokega v nizko stanje. Uporabimo lahko enačbo:

$$c\left(\hat{x}\right) = -|\gamma(\hat{x})| * \sum_{i=2}^{v} \left[\gamma(\hat{x})_i - \gamma(\hat{x})_{i-1} + \sigma(w(\gamma(\hat{x})_i))\right], \tag{4.4}$$

kjer je $\gamma(\hat{x})$ vektor vrhov signala v frekvenčnem prostoru, $w(\gamma(\hat{x})_i)$ okoliško okno i-tega harmonika velikosti 3 ter $\sigma(w(\gamma(\hat{x})_i))$ standardna deviacija vrha i v okoliškem oknu dolžine 3. Spremenljivka v predstavlja maksimalno število vrhov, ki jih v enačbi upoštevamo, v našem primeru je v enak 25. Ker želimo doseči visoke in ozke vrhove, izberemo minimalno širino okoliškega okna. Prvi del vsote v enačbi maksimizira amplitudo oscilacij, drugi del vsote pa hitrost prehoda med visokim in nizkim stanjem signala. Ker lahko takšna cenovna funkcija precenjuje signale z enim dominantim harmonikom z ne nujno najnižjo frekvenco, moramo enačbo pomnožiti še s faktorjem $|\gamma(\hat{x})|$. Tako ne kaznujemo oscilatornih signalov z večjim številom vrhov. Takšna cenovna funkcija je primerna za uporabo pri vzorčenju rešitev represilatorja, saj ne kaznuje oscilatornih signalov z drugačno amplitudo in periodo nihanja, a vseeno favorizira oscilacije z visoko amplitudo in hitrim odzivnim časom.

V nalogi za izračun frekvenčnega spektra uporabimo hitro Fourierjevo transformacijo (ang. *fast Fourier transform*) oziroma FFT [20]. Za razliko od navadne Fourierjeve transformacije, katere časovna kompleksnost je $O(n^2)$, je časovna komleksnost algoritma FFT $O(n \log n)$ [20]. Pri uporabi algoritma FFT moramo po Nyquist-Shannonovemu izreku signal vzorčiti vsaj z enkrat večjo frekvenco od največje frekvence, ki jo želimo zaznati v signalu [21]. Če želimo npr. zaznavati frekvence nižje od 20 Hz, moramo signal vzorčiti vsaj s frekvenco 40 Hz. Upoštevati moramo tudi, da vrne FFT rezultat v bazi kompleksnih števil. Velikost n-tega kompleksnega števila predstavlja velikost n-tega harmonika, kot kompleksnega števila pa fazo oziroma zamik po času. V našem primeru vzorčimo odziv modela represilatorja s periodo 5 min, zato ima najvišji harmonik, ki ga lahko zaznamo, periodo 10 min.

V primeru, da signal, ki ga analiziramo ne izkazuje oscilatorne dinamike, bo cenovna funkcija 4.4 enaka 0. V kolikor se bodo v signalu pojavila nihanja in bomo ta zaznali kot vrhove v frekvenčnem prostoru, pa bo cenovna funkcija različna od 0. V primeru, da imajo nižji harmoniki večjo amplitudo od višjih, bo vrednost cenovne funkcije negativna,

drugače pa bo vrednost cenovne funkcije pozitivna. Ker rešitev minimiziramo glede na vrednost cenovne funkcije 4.4, bomo odzive z negativno vrednostjo cenovne funkcije sprejeli, rešitve, katerih vrednost cenovne funkcije je enaka 0 ali pa je celo pozitivna, pa bomo kaznovali, saj signal ne izkazuje želene oscilatorne dinamike.

Kako pa se takšna cenovna funkcija obnese v praksi? Z genetskim algoritmom opisanim v podpoglavju 4.3.3, v desetih iteracijah z velikostjo populacije tisoč osebkov na primeru biološkega represilatorja vzorčimo osebke, v kolikor njihova vrednost cenovne funkcije pade v območje ±20% želene vrednosti enega izmed razredov. Vrednosti razredov smo eksperimentalno določili in znašajo: -10^3, $-2 * 10^3$, $-6 * 10^3$, $-12 * 10^3$ ter $-24 * 10^3$. Zanimata nas amplituda in perioda oscilacij, njihova povprečna vrednost ter standardni odklon. Tabela 4.2 prikazuje število osebkov, njihovo povprečno vrednost ter deviacijo amplitude in periode za različne vrednosti cenovne funkcije. Opazimo lahko, da se z nižanjem vrednosti cenovne funkcije veča število osebkov v posameznem razredu. Takšen pojav je pričakovan, saj se zaradi evolucijskega pritiska v genetskem algoritmu populacija osebkov boljša iz generacije v generacijo. Pomembno je, da se z nižanjem vrednosti cenovne funkcije večata povprečna amplituda ter perioda oscilacij. To pomeni, da cenovna funkcija optimizira tako amplitudo kot periodo oscilacij. Ker pa ima lahko cenovna funkcija enako vrednost za signale z različno periodo in amplitudo oscilacij, je standardni odklon amplitude in periode sorazmerno velik.

Tabela 4.2 Število osebkov, njihova povprečna vrednost ter deviacija amplitude in periode za različne vrednosti cenovne funkcije modela biološkega represilatorja.

Vrednost cenovne funkcije	Št. osebkov	Povprečje amplitude $[nM]$	Deviacija amplitude $[nM]$	Povprečje periode $[h]$	Deviacija periode $[h]$
-10^3	331	29,50	21,33	9,38	7,91
$-2 * 10^3$	281	42,32	25,11	8,37	6,24
$-6 * 10^3$	567	82,02	25,52	10,63	7,79
$-12 * 10^3$	782	121,09	30,78	10,13	6,43
$-24 * 10^3$	1123	138,29	39,83	20,88	7,81

4.3 Vzorčenje

V statistiki je vzorčenje postopek izbiranja osebkov iz populacije z namenom pridobivanja znanja o celotni populaciji. Vzorec je podmnožica osebkov, ki smo jih iz populacije izbrali. Čeprav je velikost vzorca veliko manjša od celotne populacije, lahko s statističnimi metodami zelo natančno ocenimo značilnosti celotne populacije.

V našem primeru populacijo predstavljajo vzorci znotraj zveznega in omejenega prostora dopustnih kombinacij kinetičnih parametrov $\hat{K}$. Točko v prostoru kinetičnih parametrov uvrščamo v prostor dopustnih rešitev, če zadosti naslednjemu pogoju:

$$c(\hat{x}) \leq c_{max}, \tag{4.5}$$

kjer predstavlja vektor $\hat{x}$ časovno evolucijo opazovanega proteina v biološkem modelu, $c(\hat{x})$ vrednost cenovne funkcije ter c_{max} mejo, ki je cenovna funkcija ne sme preseči. S cenovno funkcijo ocenjujemo kvaliteto oscilacij oziroma ali so oscilacije dušene, amplitudo in frekvenco oscilacij ter dolžino prehoda iz nizkega v visoko stanje.

V primeru modela biološke pomnilne celice D s predpomnenjem smatramo za dopustno rešitev osebek, čigar odziv ne odstopa od idealnega odziva h za več kot 30%. Ker vsebuje signal, ki ga opazujemo 600 točk, amplituda idealnega signala pa znaša 100 nM, smatramo za dopustno rešitev vsak odziv, katerega cenovna funkcija ne preseže vrednosti: $600 * 30^2 = 540000$.

V primeru modela biološkega represilatorja smo nekoliko manj specifični. Glede na tabelo 4.2, preseže povprečna amplituda vrednost 82 nM pri $-6 * 10^3$, kar je znaten dvig glede na $42nM$ pri vrednosti cenovne funkcije $-2 * 10^3$. Zato za maksimalno vrednost cenovne funkcije, pri kateri smatramo rešitev še za dopustno, določimo pri $-6 * 10^3$.

Ker pa takšen prostor dopustnih rešitev analitično težko opišemo, si pomagamo z vzorčenjem. Z različnimi metodami preiskujemo celoten omejen prostor kinetičnih parametrov, hkrati pa vsak nabor kinetičnih parametrov, ki izraža zadostno oscilatorno dinamiko, shranimo v vzorec. Množica teh vzorcev zato predstavlja prostor dopustnih rešitev kinetičnih parametrov.

4.3.1 Enakomerno vzorčenje

Preprosta metoda preiskovanja prostora in iskanja približka optimalne rešitve v prostoru je enakomerno vzorčenje, kjer vsako komponento prostora $\hat{S}$ razdelimo na i enako oddaljenih ekvidistančnih točk. Kartezični produkt točk v posameznih komponentah predstavlja možne kandidate za optimalno rešitev. Poudariti je potrebno, da število kandidatov k narašča eksponentno glede na število dimenzij n oziroma polinomsko glede na število ekvidistančnih točk i:

$$k = i^n. \tag{4.6}$$

Če bi želeli preiskati prostor parametrov biološkega modela z 20 kinetičnimi parametri razdeljenimi na 10^4 enakomerno razporejenih točk, bi morali preiskati 10^{80} možnih naborov kinetičnih parametrov. To je približno toliko, kolikor je vseh atomov v znanem vesolju [22]. Ker je takšno preiskovanje zelo nepraktično in postane za težje optimizacijske probleme računsko neobvladljivo, ga v nalogi ne obravnavamo in se osredotočimo na primernejši metodi preiskovanja prostora. Ti sta vzorčenje s simuliranim ohlajanjem (podpoglavje 4.3.2) ter genetski algoritmi (podpoglavje 4.3.3).

4.3.2 Vzorčenje s simuliranim ohlajanjem

Lokalno preiskovanje oziroma lokalna optimizacija je družina hevrističnih algoritmov, s katerimi iščemo rešitve optimizacijskih problemov, kot so problem trgovskega potnika, iskanje maksimalnega pretoka itd. [23]. Pri lokalnem preiskovanju začnemo preiskovanje v naključno generiranem stanju, ki ga ovrednotimo na podlagi optimizacijskega kriterija. V vsakem koraku v prostoru stanj generiramo in ocenimo soseščino. Če je najboljši sosed primernejši od trenutnega stanja, se pomaknemo v novo stanje in preiščemo njegovo okolico. Postopek ponavljamo, dokler ne pridemo do optimuma, stanja v katerem je celotna soseščina trenutnega stanja slabša. Težava takšnega preiskovanja je v tem, da lahko obtičimo v lokalnem optimumu. Delno lahko to premostimo tako, da preiskovanje poženemo večkrat in obdržimo najboljšo rešitev.

Kadar rešujemo težje optimizacijske probleme, smo ponavadi omejeni z računsko močjo. Takrat si ponovnega poganjanja ne moremo privoščiti, zato se velikokrat poslužujemo primernejših izpeljank lokalnega preiskovanja, kot je preiskovanje s simuliranim ohlajanjem [23, 24]. Ideja preiskovanja s simuliranim ohlajanjem izvira iz statistične

mehanike, kjer material s postopkom žarjenja najprej segrejemo in nato počasi ohlajamo. Pri visokih temeraturah imajo atomi v snovi več energije, zaradi česar se hitreje premikajo in se zato enakomerneje razporedijo. Z ohlajanjem se sistemu energija niža, v snovi pa se pričnejo tvoriti kristali pravilnih oblik. Tako obdelan material ima enakomernejšo razporeditev kristalov z manj nepravilnostmi [23, 24]. Tako kot pri lokalnem preiskovanju pri simuliranem ohlajanju preiskujemo soseščino, a sosede izbiramo naključno. Če je sosed boljši od trenutnega stanja, se pomaknemo v novo stanje, v nasprotnem primeru pa se v novo stanje premaknemo z neko verjetnostjo p. Ta verjetnost je odvisna od parametra T, ki predstavlja temperaturo. Ta je na začetku velika, skozi preiskovanje prostora pa se temperatura niža, zaradi česar se verjetnost izbire slabšega osebka manjša. Tako v začetni fazi preiskovanja preiščemo širšo okolico, šele nato pa lokalno optimiziramo rešitev. V začetni fazi je preiskovanje stohastično, v končni fazi pa je preiskovanje prostora bolj deterministične narave. Verjetnost, s katero izbiramo slabša stanja je:

$$p = e^{\left(-\frac{O(N)-O(C)}{T}\right)}, \tag{4.7}$$

kjer je e Eulerjevo število, $O(C)$ optimizacijski kriterij trenutnega stanja, $O(N)$ optimizacijski kriterij naključno izbranega soseda in T temperatura. V limiti, ko gre temperatura proti neskončno, je verjetnost izbire slabšega soseda enaka 1, če gre temperatura proti nič, pa je verjetnost izbire slabšega soseda 0. Parameter T sproti ohlajamo tako, da ga množimo s parametrom α, kjer je α pogosto enaka 0,95.

V našem primeru je stanje predstavljeno z vektorjem vrednosti kinetičnih parametrov visokodimenzionalnega dinamičnega modela nekega biološkega sistema. Prostor stanj je kartezični produkt vseh možnih vrednosti, ki jih lahko kinetični parametri biološkega modela zavzamejo. Ker je prostor stanj zvezen je tudi soseščina stanja neskončno velika. Zato v našem primeru generiramo soseščino nekoliko drugače. Naključno izberemo kinetični parameter k_i, ki ga bomo spremenili, ter ga naključno zmanjšamo ali povečamo za faktor $\delta * k_i$, kjer δ pogosto zavzame vrednosti med 0 in 1.

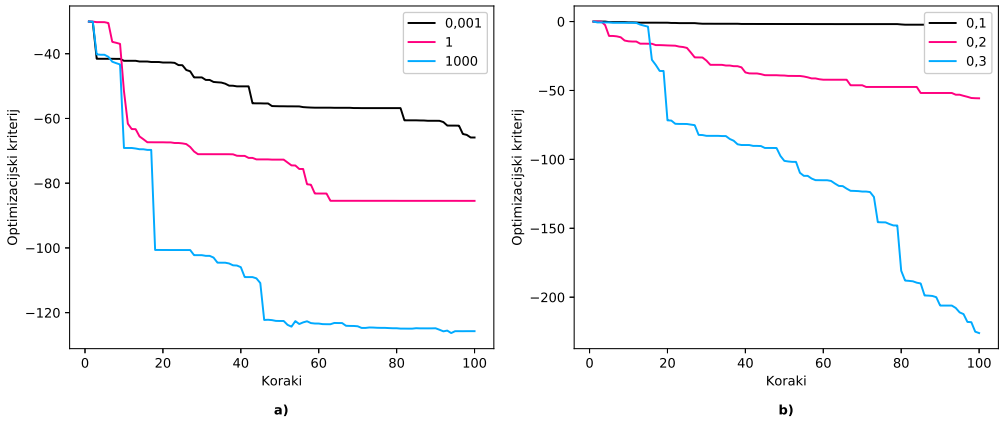
Ker želimo prostor dopustnih rešitev čim bolj temeljito in optimalno preiskati, uporabimo pristop Monte Carlo. Monte Carlo je skupina algoritmov, s katerimi lahko z generiranjem naključnih oziroma psevdonaključnih števil aproksimiramo zapletene matematične funkcije ter deterministične in stohastične dinamične modele [25]. Ena izmed aplikacij Monte Carlo metod je aproksimacija števila π, kjer v ravnini naključno generiramo točke znotraj kvadrata in računamo razmerje točk znotraj kvadrata včrtane

krožnice in vseh točk v kvadratu. Z večanjem števila naključno generiranih točk razmerje konvergira proti $\frac{\pi}{4}$ [26], kolikor je tudi razmerje ploščin kvadrata in včrtanega kroga. Simulirano ohlajanje je izpeljanka Metropolis–Hastingovega algortima. Gre za metodo Monte Carlo z Markovskimi verigami, razvito za generiranje vzorčnih stanj termodinamičnega sistema [27].

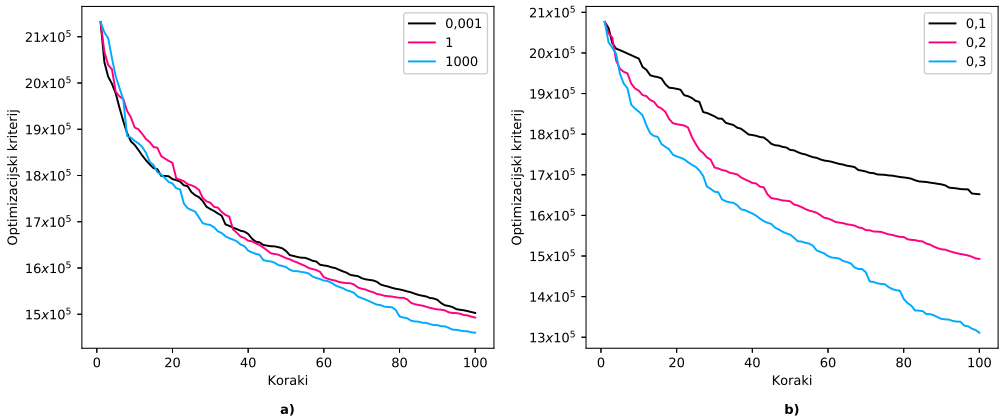
Ker iščemo dopustne regije rešitev znotraj omejenega visokodimenzionalnega prostora kinetičnih parametrov, predstavimo osebek kot vektor pozitivnih realnih števil. V iteracijah iz naključno generiranega stanja z algoritmom simuliranega ohlajanja preiskujemo prostor stanj. Prostor stanj preiskujemo tako dolgo, dokler na najdemo približka optimalne rešitve. Med potekom preiskovanja pa vzorčimo vse dopustne rešitve. Z večkratnim ponavljanjem preiskovanja se z višanjem števila iteracij povečuje tudi natančnost števila in oblika regij dopustnih rešitev.

Implementacija

Preiskovanje s simuliranim ohlajanjem smo implementirali v programskem jeziku *Python*. Za parameter hitrosti ohlajanja α smo izbrali vrednost 0,95. Preiskovanje prostora kinetičnih parametrov modela represilatorja in biološke pomnilne celice D s predpomnenjem poteka za 100 osebkov v sto korakih. Omenili smo, da generiramo soseščino z naključno izbiro parametra k_i, ki ga nato zmanjšamo ali povečamo za faktor $\delta * k_i$. Poglejmo, kako vplivata parametra začetne temperature T in bližine soseščine δ na kovergenco preiskovanja s simuliranim ohlajanjem. Da izključimo faktor naključnosti, pričnemo preiskovanje za različne vrednosti parametrov iz enakih, na začetku naključno generiranih osebkih. Slika 4.4 prikazuje padec cenovne funkcije za model represilatorja, slika 4.5 pa padec cenovne funkcije za model biološke pomnilne celice D s predpomnenjem.



Slika 4.4 Padec cenovne funkcije za model represilatorja. Slika a) prikazuje padec cenovne funkcije pri različnih začetnih temeraturah pri parametru velikosti izbire soseščine $\delta = 0{,}2$. Začetne temperature, pri katerih preiskujemo prostor, so: $0{,}001$, 1 in 1000. Slika b) prikazuje padec cenovne funkcije pri različnih velikostih koraka δ za izbiro soseščine pri začetni temperaturi 1. Velikosti koraka pri izbiri soseščine so: $0{,}1$, $0{,}2$ in $0{,}3$.



Slika 4.5 Padec cenovne funkcije za model biološke pomnilne celice D s predpomnjenjem. Slika a) prikazuje padec cenovne funkcije pri različnih začetnih temeraturah pri parametru velikosti izbire soseščine $\delta = 0{,}2$. Začetne temperature, pri katerih preiskujemo prostor, so: $0{,}001$, 1 in 1000. Slika b) prikazuje padec cenovne funkcije pri različnih velikostih koraka δ za izbiro soseščine pri začetni temperaturi 1. Velikosti koraka pri izbiri soseščine so: $0{,}1$, $0{,}2$ in $0{,}3$.

Pri modelu represilatorja lahko opazimo, da se z večanjem začetne temperature in koraka izbire soseščine viša konvergenca preiskovanja s simuliranim ohlajanjem. V primeru nižje začetne temperature preiskovanje izkazuje slabšo konvergenco kot v primeru veliko višje začetne temperature.

Pri modelu biološke pomnilne celice D s predpomnenjem dobimo podobne rezultate, le da izbira začetne temperature ne vpliva toliko na konvergenco preiskovanja. Z višanjem koraka izbire soseščine se prav tako veča hitrost konvergence preiskovanja.

Ker smo za ocenjevanje kvalitete rešitev modelov represilatorja in pomnilne celice izbrali različne cenovne funkcije, je tudi oblika konvergence drugačna. V primeru modela represilatorja je krivulja cenovne funkcije stopničaste oblike. Za ocenjevanje kvalitete rešitev represilatorja uporabimo cenovno funkcijo iz enačbe 4.4, s katero optimiziramo kvaliteto oscilacij v frekvenčnem prostoru. Če osebek ne izkazuje oscilacij, bo njegova vrednost cenovne funkcije enaka 0. Pri preiskovanju bo vrednost cenovne funkcije enaka 0 pri večini osebkov. V kolikor pri preiskovanju naletimo na oscilatorno dinamiko, le-to optimiziramo. V primeru biološke pomnilne celice D s prepomnenjem smo za cenovno funkcijo uporabili razliko kvadratov odziva osebka in idealnega signala 4.2. Konvergenca modela biološke pomnilne celice D je zato manj stopničasta, saj lahko odziv osebka vedno optimiziramo, ne glede na to, za koliko se odziv osebka razlikuje od idelanega oscilatornega signala.

Najnižja povprečna vrednost cenovne funkcije modela represilatorja, ki smo jo dosegli v 100 iteracijah, je približno -200. Za dopustno rešitev smatramo takšen osebek, čigar vrednost cenovne funkcije je nižja od $-6 * 10^3$. To pomeni, da nam je pri preiskovanju uspelo delno optimizirati le nekaj izmed sto osebkov. V primeru modela biološke pomnilne celice dosežemo v 100 iteracijah najnižjo povprečno vrednost cenovne funkcije okoli $15 * 10^5$. Osebek modela biološke pomnilne celice smatramo za dopustnega, če pade vrednost njegove cenovne funkcije pod mejo $54 * 10^4$. Glede na hitrost konvergence modelov represilatorja in biološke pomnilne celice D s predpomnenjem, lahko zaključimo, da je vzorčenje prostora s preiskovanjem s simuliranim ohlajanjem v našem primeru neučinkovito, saj je prostor kinetičnih parametrov preveč kompleksen.

Algoritem preiskovanja bi lahko sicer izboljšali tako, da ne bi ohlajali samo temperature, temveč tudi velikost koraka pri izbiri soseščine. Tako bi na začetku delali velike korake, na koncu pa bi rešitve še lokalno optimizirali. Vendar pa bi morali pravilno nastaviti tudi parametra velikosti koraka in hitrosti ohlajanja. Pri daljšem preiskovanju

z večjim številom osebkov, bi tudi s simuliranim ohlajanjem našli dopustno rešitev, a bi za reprezantativno vzorčenje prostora dopustnih rešitev porabili preveč časa. Zato se raje osredotočimo na potencialno primernejšo metodo vzorčenja prostora kinetičnih parametrov, tj. genetske algoritme.

4.3.3 Genetski algoritmi

Pri genetskih algoritmih rešujemo težke optimizacijske probleme, kjer s pomočjo optimizacijskega kriterija iščemo približek optimalne rešitve. Ker iščemo optimalno rešitev s pomočjo hevristike, žrtvujemo zagotovilo, da bomo našli optimalno rešitev v zameno za potencialno dobro rešitev v kratkem času [1]. Pri genetskih algoritmih se zgledujemo po naravni evoluciji, kjer se osebki znotraj populacije skozi več generacij s pomočjo reprodukcije in selekcije razvijajo in prilagajajo na okolje [1, 6]. Ločiti moramo med pojmoma genotip in fenotip. Pri genotipu gre za genetsko osnovo organizma oziroma njegov DNA, ki ga ta podeduje od svojih predhodnikov. Vidne in določljive lastnosti organizma, ki se pod vplivi iz okolja izražajo iz genotipa imenujemo fenotip. Selekcija naravne evolucije deluje izključno na fenotipu, saj imajo večjo verjetnost preživetja organizmi, ki so bolj prilagojeni na okolje.

Pri genetskih algoritmih je proces iskanja najprimernejšega osebka nekoliko bolj preprost kot v naravi. Vsak osebek znotraj populacije predstavlja kandidata za približek optimalne rešitve optimizacijskega problema. Skozi iteracije osebke križamo in jih mutiramo. Tako generiramo nove potomce, ki jih na koncu vsake iteracije ocenimo s cenovno funkcijo. Ker spustimo v naslednjo generacijo le delež najboljših kandidatov, lahko pričakujemo, da bo vsaka naslednja generacija izkazovala boljši odziv. Skozi iteracije več generacij lahko zato pričakujemo, da bo genetski algoritem konvergiral k približku optimalne rešitve [1].

V naslednjih odsekih opišemo na kakšne načine lahko predstavimo osebek, kako določimo začetno populacijo ter podrobneje predstavimo procese križanja, mutacije in selekcije. Na koncu še opišemo kakšne pristope smo ubrali pri reševanju našega problema.

Predstavitev osebka

Način predstavitve osebka je odvisen predvsem od problema, ki ga rešujemo. Če rešujemo problem, kjer predstavlja prostor rešitev zaporedje binarnih akcij ali odločitev DA in NE, je najprimernejša predstavitev vektor binarnih števil. Kadar iščemo kinetične parametre nekega modela, lahko predstavimo osebek kot vektor realnih števil, v primeru da je prostor zvezen, ali kot vektor celih števil, če je prostor diskreten. Če imamo podan nabor matematičnih operacij ter operandov in iščemo izraz, ki se bo čim bolj približal v naprej znani rešitvi, lahko osebek predstavimo v obliki drevesa. Listi drevesa predstavljajo operande, vozlišča pa predstavljajo operacije, ki jih nad operandi izvajamo [1]. V našem primeru iščemo optimalne kinetične parametre modelov represilatorja in pomnilne celice D s predpomnenjem, zato je osebek predstavljen kot točka v večdimenzionalnem prostoru z vektorjem pozitivnih realnih števil.

Začetna populacija

Za optimalno konvergenco genetskega algoritma mora biti začetna populacija dovolj velika in raznolika. Velikost populacije je odvisna od težavnosti problema, ki ga rešujemo. Ponavadi je populacija velika od nekaj deset do nekaj sto osebkov, v nekaterih primerih pa tudi več tisoč. Večja velikost populacije največkrat pomeni tudi hitrejšo konvergenco, ampak tudi večjo računsko kompleksnost. Če smo omejeni z računsko močjo in časom, iščemo kompromis med velikostjo populacije in hitrostjo konvergence genetskega algoritma. Začetna populacija mora biti tudi dovolj raznolika. V primeru kompleksnega prostora rešitev z večjim številom lokalnih ekstremov, lahko zaradi premajhne raznolikosti populacije genetski algoritem konvergira v lokalni ekstrem. Temu se izognemo tako, da osebke v začetni populaciji generiramo naključno.

Križanje in mutacija

Nedeterminističnost ima pri preiskovanju prostora pomembno vlogo, saj z naključnim iskanjem v okoliškem prostoru hitreje preiščemo prostor rešitev kot bi ga pri enakomernem vzorčenju. Mutacija je operacija, pri kateri posamezen osebek spremenimo oziroma mutiramo le za majhen faktor. Mutiran osebek se zato od starega ne bo veliko razlikoval in bo v prostoru rešitev tudi v njegovi soseščini. Z majhnimi modifikacijami bomo tako posamezne osebke lokalno optimizirali. Če je osebek predstavljen kot vektor realnih števil, lahko osebek mutiramo, tako da z določeno verjetnostjo p_m pomnožimo vsako

vrednost v genotipu z naključnim deležem njene vrednosti. V literaturi lahko zasledimo, da je pogosto uporabljena verjetnost mutacije reda 0,01 [1], vendar lahko ta variira glede na težavnost problema in velikost prostora, ki ga preiskujemo.

Pri križanju ustvarimo dva nova potomca iz dveh osebkov tako, da skombiniramo genotipa obeh staršev. Za razliko od mutacije se lahko novi osebki precej razlikujejo od starih tako v podobnosti genotipov kot tudi v rezultatu cenovne funkcije. Križanja se poslužujemo v upanju, da bodo novi kandidati podedovali dobre lastnosti obeh staršev. Tako lahko dobimo osebek, ki izkazuje veliko boljše obnašanje obeh staršev. Prav tako lahko generiramo tudi veliko slabši osebek, vendar slabim osebkom ponavadi ne uspe priti v naslednjo generacijo. Križanje lahko izvajamo na več različnih načinov. Pri predstavitvi osebkov z vektorjem števil sta najpogostejši metodi križanja enotočkovno in večtočkovno križanje. Pri enotočkovnem križanju izberemo naključno točko, pri kateri zamenjamo genotipa dveh osebkov. Tako pridobimo dva nova otroka, ki vsebujeta del genotipa obeh staršev. Večtočkovno križanje deluje v principu enako kot enotočkovno križanje, le da naključno izberemo večje število točk, pri katerih bomo izmenjaje kombinirali dele obeh staršev. Pri dvotočkovnem križanju razdelimo starša na tri dele in zamenjamo srednji del.

Optimizacijski kriterij

Optimizacijski kriterij oziroma cenovna funkcija je hevristika v genetskih algoritmih, s katero ocenjujemo kvaliteto posameznih osebkov na podlagi njihovega fenotipa. Na podlagi cenovne funkcije izločamo slabše osebke in skozi generacije ohranjamo boljše. Tako zagotovimo vedno boljšo populacijo in konvergenco k optimalni rešitvi. Cenovne funkcije se poslužujemo v vsaki iteraciji genetskega algoritma. Najprej z operacijama križanja in mutacije ustvarimo novo populacijo osebkov. To nato ocenimo s cenovno funkcijo, na podlagi katere bomo v novo generacijo spustili le delež najboljših osebkov.

Selekcija

Selekcija je operacija, s katero določamo, kateri osebki se bodo uvrstili v naslednjo generacijo. S selekcijo nad populacijo izvajamo evolucijski pritisk. Z večanjem evolucijskega pritiska se načeloma veča tudi konvergenca genetskega algoritma, hkrati pa se manjša raznolikost populacije, kar povečuje verjetnost da obtičimo v lokalnem ekstremu. Zato iščemo kompromis med močjo evolucijskega pritiska in stopnjo mutacije ter križanja.

Načinov, s katerimi uvrščamo osebke v naslednjo generacijo, je več. Najpogostejši so: izbira z ruletnim kolesom, izbira na podlagi ranka in turnirska izbira.

Pri izbiri z ruletnim kolesom osebek i uvrstimo v naslednjo generacijo na podlagi njegove verjetnosti $p(i)$. Ta verjetnost je odvisna od vrednosti njegove cenovne funkcije v primerjavi z vsoto vrednosti cenovnih funkcij vseh osebkov:

$$p(i) = \frac{f(i)}{\sum_i^N f(i)}, \tag{4.8}$$

kjer je N velikost populacije ter $f(i)$ vrednost cenovne funkcije i-tega osebka. Izbiro i-tega osebka lahko ponazorimo z ruletnim kolesom. Vsak osebek ima rezervirano določeno območje na ruletnem kolesu. Velikost tega območja je odvisna od verjetnosti $p(i)$. Večja bo ta verjetnost, večkrat bomo z ruletnim kolesom izbrali prav ta osebek. Ruletno kolo poženemo N-krat, dokler ne zapolnimo celotne populacije. Slabost ruletnega kolesa je, da ne deluje dobro, če imajo osebki podobne vrednosti cenovne funkcije, ali pa če ima en osebek bistveno višjo vrednost cenovne funkcije kot ostali osebki.

Zato lahko namesto vrednosti cenovne funkcije za izračun verjetnosti $p(i)$ upoštevamo le rank i-tega osebka. Takšna vrsta izbire deluje bolje od izbire z ruletnim kolesom, saj v primeru osebka s prevladujočo vrednostjo cenovne funkcije njegova verjetnost izbire ne bo prevelika. Če imajo osebki zelo podobno vrednost cenovne funkcije, bodo imeli boljši osebki vedno večjo verjetnost izbire, ne glede na to, kako majhna je razlika med vrednostmi cenovne funkcije.

Pri turnirski izbiri za izbiro osebka, ki se bo uvrstil v novo generacijo, organiziramo turnir, tako da izmed celotne populacije naključno izberemo k osebkov, med katerimi uvrstimo v naslednjo generacijo le najboljšega. Postopek ponavljamo, dokler ne zapolnimo celotne populacije. Tako kot pri izbiri z ruletnim kolesom in izbiri na podlagi ranka lahko tudi pri turnirski izbiri posamezen osebek izberemo večkrat.

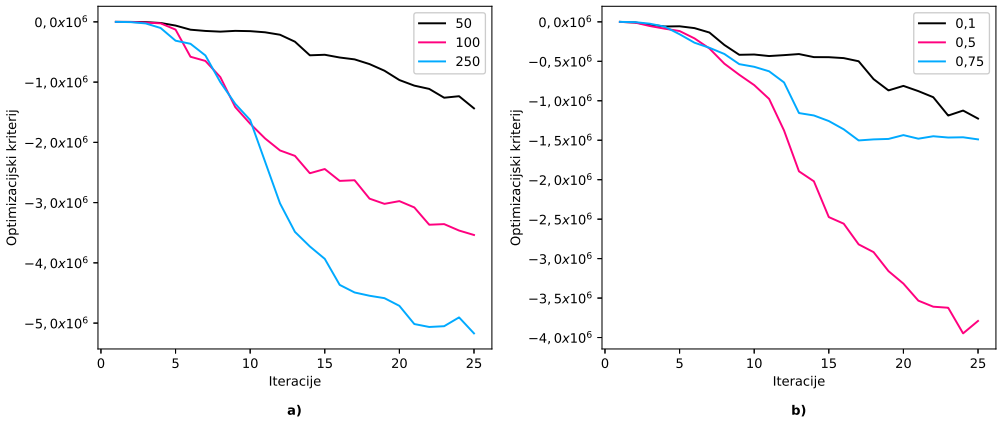
Implementacija

Za implementacijo preiskovanja in vzorčenja z genetskim algoritmov se poslužimo programskega jezika *Python* in knjižnjice *DEAP* (ang. *Distributed Evolutionary Algorithms in Python*) [28]. Gre za evolucijsko računsko ogrodje, ki vsebuje evolucijske algoritme, kot so: genetski algoritmi, genetsko programiranje in optimizacija z rojem delcev (ang. *particle swarm optimization*). Knjižnjica *DEAP* vsebuje vse zgoraj opisane operacije, kot

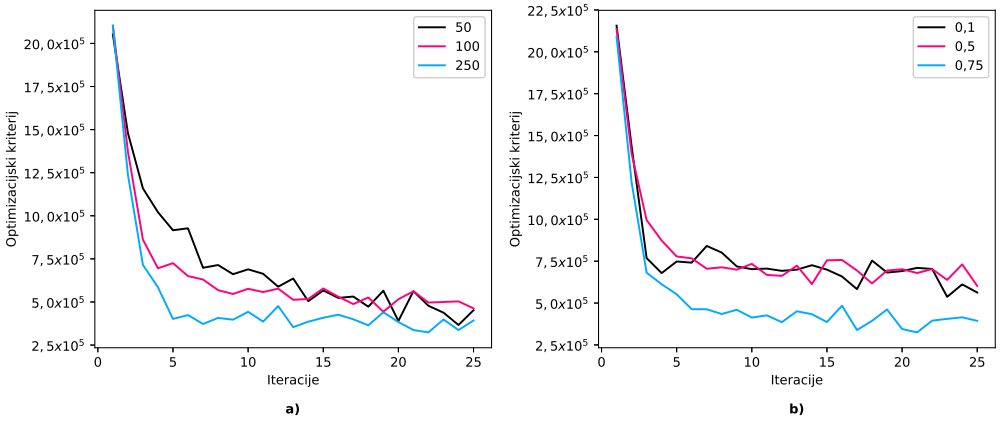
so križanje, mutacija in selekcija. Z uporabo *Python* knjižnjice *multiprocessing* lahko postopek evalvacije posameznih osebkov tudi paraleliziramo na večih jedrih procesorja.

Za križanje uporabimo dvotočkovno križanje, kjer z verjetnostjo 0,2 križamo dva sosednja osebka. Pri mutaciji osebku vsako koordinato z verjetnostjo p_m pomnožimo z naključno vrednostjo med m_1 in m_2. V našem primeru sta $m_1 = 0{,}8$ in $m_2 = 1{,}2$. Pri mutaciji tako osebek v pričakovanju, da najdemo optimalnejšo rešitev, le znatno spremenimo. Selekcijo realiziramo s turnirsko izbiro z velikostjo turnirja četrtine celotne velikosti populacije.

Poglejmo še kako na konvergenco genetskega algoritma vplivata velikost populacije in verjetnost mutacije posameznega kinetičnega parametra. Slika 4.6 prikazuje padec cenovne funkcije za model represilatorja, slika 4.7 pa padec cenovne funkcije za model biološke pomnilne celice D s predpomnenjem. Zaradi stohastičnosti genetskih algoritmov za posamezne velikosti populacij in verjetnosti mutacij preiskovanje poženemo štirikrat, rezultate pa nato povprečimo.



Slika 4.6 Padec cenovne funkcije za model represilatorja. Slika a) prikazuje padec cenovne funkcije pri različnih velikostih populacije pri verjetnosti mutacije 0,5. Za referenčne vrednosti populacije izberemo vrednosti: 50, 100 in 250. Slika b) prikazuje padec cenovne funkcije za različne verjetnosti mutacije kinetičnega parametra p_m pri velikosti populacije stotih osebkov. Verjetnosti mutacije kinetičnega parametra so: 0,1, 0,5 in 0,75.



Slika 4.7 Padec cenovne funkcije za model biološke pomnilne celice D s predpomnjenjem. Slika a) prikazuje padec cenovne funkcije pri različnih velikostih populacije pri verjetnosti mutacije $0{,}5$. Za referenčne vrednosti populacije izberemo vrednosti: 50, 100 in 250. Slika b) prikazuje padec cenovne funkcije za različne verjetnosti mutacije kinetičnega parametra pri velikosti populacije stotih osebkov. Verjetnosti mutacije kinetičnega parametra so: $0{,}1$, $0{,}5$ in $0{,}75$.

Pri modelu represilatorja lahko opazimo, da z večanjem populacije konvergenca genetskega algoritma narašča. Najoptimalnejšo konvergenco dobimo pri velikosti populacije 250 osebkov. Za verjetnost mutacije kinetičnega parametra p_m je najboljša izbira vrednost 0,5, saj dobimo takrat najhitrejšo konvergenco.

Pri modelu biološke pomnilne celice D s predpomnenjem so rezultati podobni, a nekoliko manj očitni. Tudi tukaj je naprimernejša izbira velikosti populacije 250 osebkov. Za razliko od represilatorja pa model pomnilne celice D konvergira bolje pri višji verjetnosti mutacije kinetičnega parametra, in sicer pri $p_m = 0{,}75$. Trend sicer nakazuje, da bi lahko izbrali še večjo stopnjo mutacije, vendar se z večanjem verjetnosti mutacije kinetičnega parametra veča tudi naključnost preiskovanja. To lahko posledično privede do slabše konvergence algoritma. Glede na literaturo [1], kjer je pogosto uporabljen red mutacije 0,01, je verjetnost mutacije v našem primeru občutno višja in je zato dodatno nismo višali.

Z večanjem velikosti populacije se pri obeh modelih veča tudi hitrost konvergence geneteskega algoritma. Za preiskovanje prostora dopustnih rešitev bi zato lahko izbrali še večjo velikost populacije, s čimer bi dosegli boljšo konvergenco. Ker pa z genetskimi algoritmi prostora ne vzorčimo enakomerno, bi se lahko zgodilo, da je število osebkov v okolici optimalne rešitve bistveno višje od števila osebkov v okolici manj optimalnih rešitev, ki

jih še uvrščamo med dopustne. Zato vzorčenje z genetskimi algoritmi poženemo večkrat na ne preveliki velikosti populacije osebkov.

Poglejmo si hitrost konvergence obeh modelov. Model represilatorja konvergira približno v 25 generacijah genetskega algoritma. Model biološke pomnilne celice D s predpomnenjem konvergira že približno v 5-ih generacijah genetskega algoritma. Za razliko od preiskovanja s simuliranim ohlajanjem je konvergenca preiskovanja z genetskim algoritmom hitra, zaradi česar so v našem primeru genetski algoritmi primernejši za vzorčenje prostora dopustnih rešitev. Oba modela bomo v nadaljevanju vzorčili z genetskim algoritmom z velikostjo populacije 250 osebkov. V primeru represilatorja izberemo za verjetnost mutacije p_m vrednost 0,5. V primeru modela biološke pomnilne celice izberemo za verjetnost mutacije p_m vrednost 0,75. Prostor vzorčimo tako, da poženemo preiskovanje obeh modelov 50-krat.

4.4 Razvrščanje

Razvrščanje ali gručenje (ang. *clustering*) je postopek razvrščanja elementov znotraj iste množice podatkov v razrede. Cilj razvrščanja je razporediti elemente v razrede, tako da so si elementi znotraj iste množice med seboj čim bolj podobni glede na določeno lastnost, elementi iz različnih razredov pa se med seboj čim bolj razlikujejo. Razvrstitev osebkov znotraj razredov imenujemo tudi razbitje. Podobnost med osebkoma a in b merimo z razdaljo $d(a,b)$. Najpogosteje uporabljeni sta Evklidska in Manhattanska razdalja [29].

Algoritmov za razvrščanje podatkov je ogromno. Ti se delijo v štiri glavne skupine: metode težišč (ang. *centroid-based clustering*), razvrščanje na podlagi povezanosti (ang. *connectivity-based clustering*), razvrščanje na podlagi gostote podatkov (ang. *density-based clustering*) in razvrščanje s porazdelitvami (ang. *distribution-based clustering*). Razvrščanje uvrščamo med metode nenadzorovanega strojnega učenja [29].

Ker v nalogi obravnavamo kompleksnejše dinamične modele, ki izražajo oscilatorno dinamiko, pričakujemo tudi pojav lokalnih ekstremov. Predvidevamo lahko, da se bo vzorec osebkov razporedil okoli lokalnih optimumov. Ti so lahko v prostoru kinetičnih parametrov zelo blizu, hkrati pa so lahko med seboj zelo oddaljeni. Zaradi slednjega razloga vzorce še dodatno razdelimo na razrede, za kar uporabimo različne algoritme razvrščanja.

V nalogi obravnavamo dva osnovna algoritma razvrščanja, to sta hierarhično razvrščanje

(ang. *hierarchical clustering*) in razvrščanje z voditelji (ang. *k-means*). Ker je v našem primeru prostor kinetičnih parametrov zvezen, za metriko podobnosti uporabimo kar Evklidsko razdaljo.

Algoritmov za razvrščanje na podlagi gostote podatkov v nalogi ne obravnavamo, saj želimo razvrstiti vse dopustne rešitve, ne samo tistih, ki zadoščajo minimalni gostoti [30]. Prav tako ne obravnavamo algoritmov razvrščanja s porazdelitvami, saj se te pogosto preveč prilagodijo podatkom [31].

4.4.1 Hierarhično razvrščanje

Hierarhično razvrščanje (ang. *hirearchical clustering*) je algoritem, ki spada v skupino razvrščanja na podlagi povezovanja. Osebke razvrščamo v razrede, ki jih nato združujemo v večje razrede glede na njihovo podobnost [29]. Za ocenjevanje podobnosti razredov R_a in R_b, lahko uporabimo naslednje razdalje:

- razdalja med najbližjima osebkoma obeh razredov (ang. *single linkage*):

$$d_{min} = \min_{a,b}\{d(a,b) \mid a \in R_a,\ b \in R_b\}, \tag{4.9}$$

- razdalja med najbolj oddaljenima osebkoma obeh razredov (ang. *complete linkage*),

$$d_{maks} = \max_{a,b}\{d(a,b) \mid a \in R_a,\ b \in R_b\}, \tag{4.10}$$

- povprečna razdalja vseh osebkov med razredoma (ang. *average linkage*):

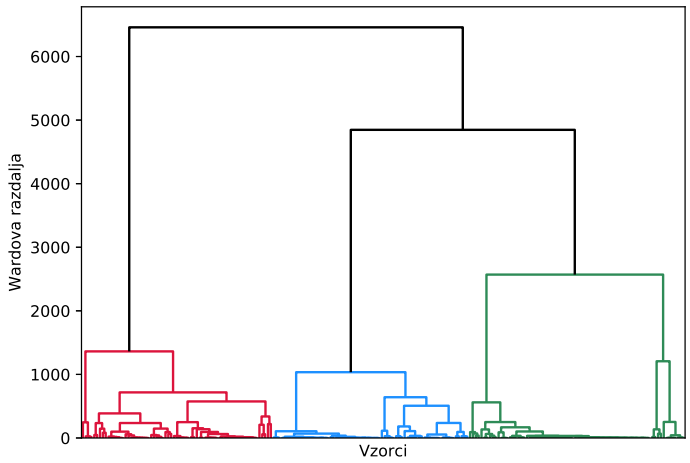
$$d_{pov} = \frac{1}{|R_a||R_b|}\left(\sum_{a\in R_a}\sum_{b\in R_b} d(a,b)\right) \tag{4.11}$$

- ter Wardova razdalja, ki predstavlja razliko napak med združenima razredoma R_{ab} ter vsoto obeh napak obeh posamičnih razredov R_a in R_b:

$$d_{ward} = \sum_{c\in R_{ab}} d(c, c_{ab}) - \left(\sum_{a\in R_a} d(a, c_a) + \sum_{b\in R_b} d(b, c_b)\right) \tag{4.12}$$

  kjer so c_a, c_b ter c_{ab} središčne točke razredov R_a, R_b in R_{ab}. Pri uporabi Wardove razdalje združujemo razrede, kjer so centri razredov zelo blizu, hkrati pa so osebki znotraj posameznih razredov dovolj oddaljeni od središčne točke svojega razreda.

Na začetku uvrstimo vsak osebek v svoj razred. V naslednjih korakih razrede hierarhično povezujemo v večje razrede glede na podobnost. Ta postopek ponavljamo, dokler ne ostane samo en razred, v katerega so razvrščeni vsi osebki. Rezultat takšnega razvrščanja je drevo. Koren drevesa predstavlja največji razred, ki vsebuje vse osebke, vozlišča drevesa predstavljajo podrazrede, v listih drevesa pa so posamezni osebki. Rezultat hirearhičnega razvrščanja lahko prikažemo z dendrogramom. V našem primeru uporabimo za razdaljo med posameznimi razredi Wardovo razdaljo. Primer takšnega dendrograma predstavlja slika 4.8.

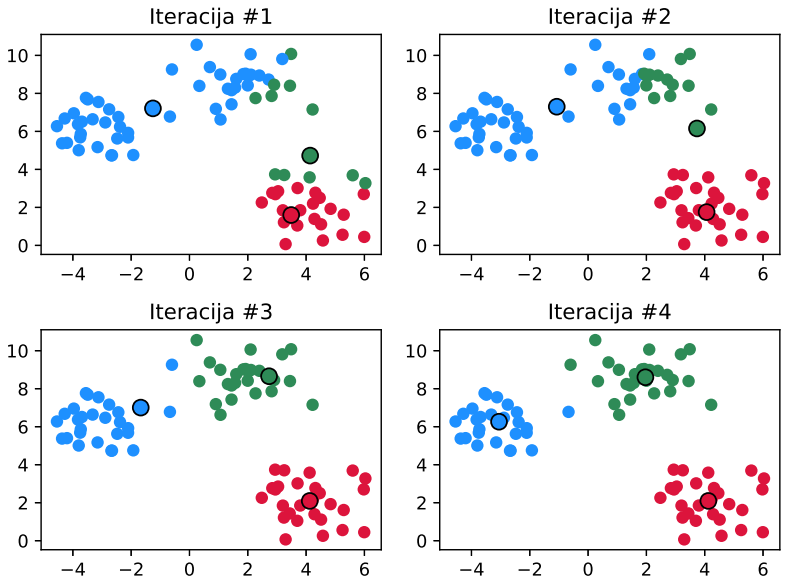


Slika 4.8 Grafičen prikaz hierarhičnega razvrščanja z dendrogramom na vzorčnih podatkih modela biološke pomnilne celice D s predpomnenjem. Vzorce smo pridobili z genetskim algoritmom. Zaradi velike časovne zahtevnosti in lepšega prikaza, smo razvrstili le delež najboljših vzorcev. Za metriko združevanja razredov smo uporabili Wardovo razdaljo. Iz dendrograma je razvidno, da bi bilo vzorce smiselno razvrstiti v tri skupine.

V najslabšem primeru združujemo le razred z enim osebkom v največji razred. Da izračunamo vse možne razdalje porabimo $O(n^2)$ časa, ker pa vedno največjemu razredu pridružimo le en osebek, moramo razdalje izračunavati $(n-1)$-krat, zaradi česar je časovna zahtevnost osnovne implementacije hirearhičnega razvrščanja enaka $O(n^3)$, kjer je n število osebkov. Z uporabo kopice lahko časovno kompleksnost zmanjšamo na $O(n^2 \log n)$, kar je še vedno nepraktično za velike količine podatkov. Dodatna slabost hirearhičnega razvrščanja je ta, da moramo sami razbrati optimalno število razredov. To postane pri velikih količinah podatkov nepraktično. Hirearhičnega razvrščanja za nadaljnjo analizo dopustnega prostora kinetičnih parametrov ne uporabimo, saj je zaradi prevelike časovne zahtevnosti neprimerno za uporabo na naših podatkih.

4.4.2 Razvrščanje z voditelji

Razvrščanje z voditelji (ang. *k-means*) je algoritem, ki spada v skupino težiščnih metod. Voditelji s pripadajočimi osebki predstavljajo posamezne razrede. Osebke v iteracijah razvrščamo k najbližjim voditeljem, tako vsak osebek razvrstimo v določen razred. V naslednji iteraciji za nove voditelje vzamemo kar težišča razredov. Postopek razvrščanja ponavljamo, dokler ta ne konvergira, oziroma se voditelji iz prejšnje iteracije še spreminjajo [29]. Prikaz razvrščanja z voditelji prikazuje slika 4.9.



Slika 4.9 Prve štiri iteracije algoritma razvrščanja z voditelji. Podatke smo generirali naključno po normalni porazdelitvi dveh spremenljivk s kovariančno matriko $\Sigma = \left[\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right]$ pri treh naključno izbranih aritmetičnih sredinah. Točke s črno obrobo predstavljajo voditelje razredov. Začetni voditelji so izbrani naključno. V prvi iteraciji razbitje ni optimalno. V naslednjih treh iteracijah algoritem razvrščanja konvergira k optimalni rešitvi.

Izbor začetnih voditeljev je lahko nakjučen, lahko pa poiščemo takšne voditelje, ki so med seboj najbolj oddaljeni, oziroma so čim bolj razpršeni. Če izbiramo voditelje naključno, se lahko zgodi, da večkratno poganjanje metode konvergira v različne rezultate. Zato v praksi razvrščanje poženemo večkrat za določeno število razredov k. Obdržimo tisti nabor voditeljev, ki minimizira vsoto kvadratov napak znotraj posameznih razredov (ang. *sum of squared errors*):

$$W_k = \sum_{i=1}^{k} \sum_{o \in R_i} (o - c_i)^2, \tag{4.13}$$

kjer je W_k vsota kvadratov napak znotraj posameznih razredov, k število razredov, R_i i-ti razred, o osebek razreda R_i ter c_i težišče razreda R_i.

Glavna slabost razvrščanja z voditelji je, da moramo sami določiti optimalno število razredov. Zato v praksi za različno število razredov k poženemo razvrščanje večkrat, ter na podlagi določene metrike izberemo najprimernejše število razredov.

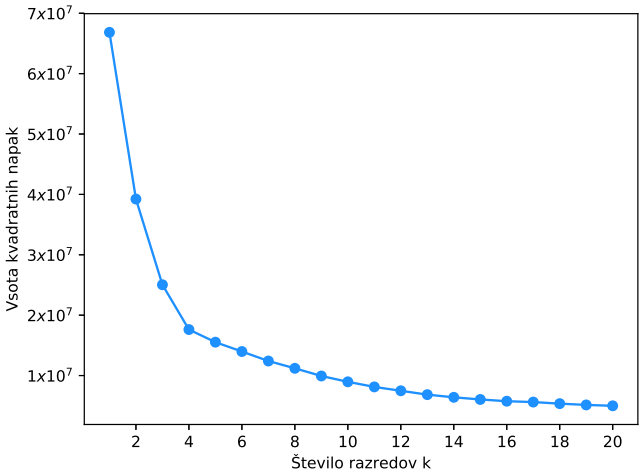
Za realizacijo razvrščanja z voditelji v programskem jeziku *Python* uporabimo paket *cluster KMeans* knjižnjice $scikit - learn$.

4.4.3 Izbira optimalnega števila razredov

Pri izbiri optimalnega števila razredov moramo biti pozorni, da ne izberemo premajhnega ali prevelikega števila razredov. V primeru razvrščanja s premajhnim številom razredov, bomo v isti razred razvrstili podatke, ki bi jih bilo smiselno razvrstiti v različne razrede. V primeru razvrščanja s prevelikim številom razredov se bodo podatki znotraj posamezne skupine razvrstili v večje število razredov kot bi bilo potrebno.

Komolčna metoda

Pri izbiri optimalnega števila razredov si lahko pomagamo z vsoto kvadratov napak W_k (glej enačbo 4.13). Ta z večanjem števila razredov strmo pada, nato pa se v določeni točki uravna. Točka, kjer se padec vsote kvadratov napak prevesi, nakazuje na optimalno število razredov. Graf, ki ga izrišemo, spominja na komolec, od tod tudi ime metode, komolčna metoda (ang. *elbow method*) [29, 32]. Primer rezultatov komolčne metode na vzorčnih podatkih represilatorja prikazuje slika 4.10.



Slika 4.10 Komolčna metoda na vzorčnih podatkih prostora dopustnih rešitev modela represilatorja. Za razbitje smo uporabili algoritem razvrščanja z voditelji. Iz grafa lahko razberemo, da je razbitje optimalno pri štirih razredih.

Glavna slabost komolčne metode je ta, da moramo sami razbrati optimalno število razredov razbitja. Zato metoda odpove v primeru, ko so podatki nekoliko bolj razpršeni, skupine pa niso tako izrazite. Takrat je komolec manj izrazit, zato ne moremo jasno določiti optimalnega števila razredov.

Statistika vrzeli

Pomankljivost komolčne metode se skriva v tem, da ne upošteva velikosti prostora razbitja [32]. Zato pri metodi statistika vrzeli (ang. *gap statistic*) primerjamo vrednost logaritma vsote kvadratnih napak W_k z referenčnim razbitjem. Za primer vzemimo razbitje na k razredov, kjer so podatki ter voditelji razredov enakomerno razpršeni po prostoru s p dimenzijami. Pričakovana vrednost $\log(W_k)$ je [32]:

$$\log(pn/12) - (2/p)\log(k) + C, \tag{4.14}$$

kjer je C konstanta. Zaradi člena $(2/p)\log(k)$ bo pričakovana vrednost referenčnega razbitja padala z večanjem števila razredov. Če naša množica podatkov izkazuje K jasno definiranih skupin, bo podobno kot pri komolčni metodi, vrednost $\log(W_k)$ strmo padala za $k \leq K$, pri $k > K$ pa padec ne bo tako izrazit. Zato bo pri $k = K$ vrzel med našim in referenčnim razbitjem največja [32]. Vrzel za število razredov k je definirana kot:

$$G(k) = E^*[\log(W_k)] - \log(W_k), \tag{4.15}$$

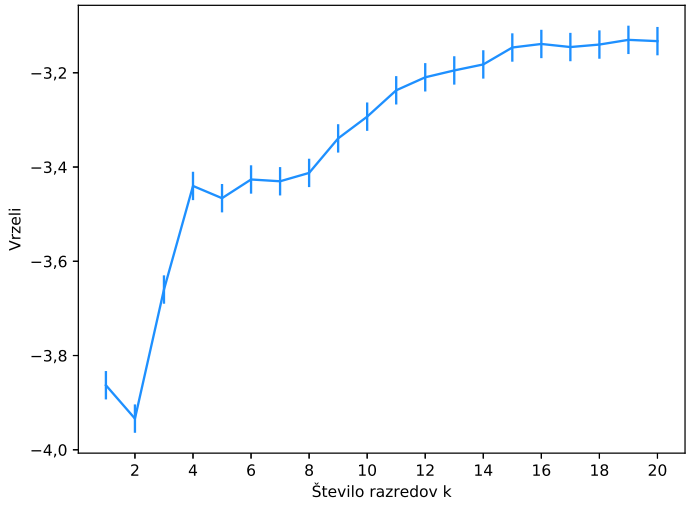
kjer je $E^*[\log(W_k)]$ upanje referenčne porazdelitve. Ker referenčne porazdelitve ne poznamo, vzorčimo podatke B-krat iz prostora rešitev po enakomerni porazdelitvi ter rezultate povprečimo. Vrzel lahko nato ocenimo kot:

$$G(k) = (1/B)\sum_b \log(W^*_{kb}) - \log(W_k). \tag{4.16}$$

Da pa števila razredov razbitja ne precenimo, določimo za optimalno število razredov najmanjši k, ki zadosti pogoju $G(k) \geq G(k+1) - s_{k+1}$, kjer je s_k ocena standardne napake, ki jo pridelamo z naključnim vzorčenjem:

$$\begin{aligned} \bar{l} &= (1/B)\sum_b \log(W^*_{kb}), \\ sd_k &= ((1/B)\sum_b (\log(W^*_{kb}) - \bar{l})^{1/2}, \\ s_k &= sd_k * \sqrt{1 + 1/B}. \end{aligned} \tag{4.17}$$

Slika 4.11 prikazuje graf statistike vrzeli na vzorčnih podatkih represilatorja.



Slika 4.11 Graf statistike vrzeli na podatkih represilatorja. Metoda napove optimalno število razredov pri $k = 1$, saj je vrzel enega razreda večja od vrzeli dveh razredov. Če analiziramo celotno krivuljo vrzeli, opazimo da vrzel najbolj naraste pri štirih razredih. To sovpada tudi s komolčno metodo 4.10. Vrzeli naraščajo z večanjem števila razredov, kar nakazuje, da skupine znotraj podatkov niso tako jasno definirane. V takšnih primerih moramo zato v obzir vzeti celotno krivuljo vrzeli.

V našem primeru vsebuje referenčno razbitje 250 osebkov, ki jih generiramo naključno po celotnem prostoru rešitev na enak način kot smo generirali začetno populacijo pri preiskovanju prostora z genetskim algoritmom. Referenčno porazdelitev vzorčimo desetkrat.

Metoda vrzeli poda v primerjavi s komolčno metodo bolj reprezentativne rezultate, zato jo uporabimo kot metriko pri izbiri optimalnega števila razredov prostora dopustnih rešitev bioloških modelov. Vendar moramo tudi pri tej metodi v primeru večje razpršenosti podatkov, kjer skupine niso jasno definirane, v obzir vzeti celotno krivuljo vrzeli.

4.5 Analiza prostora dopustnih rešitev

Pri analizi prostora dopustnih rešitev želimo v prvi vrsti vedeti ali je naše razbitje kvalitetno. Hkrati nas zanima kvaliteta rešitev znotraj posameznih razredov. Idealen razred je močno zastopan z visoko povprečno amplitudo in nizkim standardnim odklonom. Rešitve v razredu so po prostoru močno razpršene in je zato njihova varianca visoka. Pri oceni kvalitete rešitev znotraj razredov se bomo opirali na navedene lastnosti idealnega razreda.

Da si lahko visokodimenzionalne podatke predstavljamo, se moramo poslužiti različnih tehnik za vizualizacijo večdimenzionalnih podatkov, ki temeljijo na zmanjšanju dimenzionalnosti podatkov. Število dimenzij v podatkih lahko zmanjšamo z uporabo presekov in projekcij [29].

4.5.1 Ocena kvalitete razbitja

Pri ocenjevanju kvalitete pridobljenih razredov nas zanima predvsem, kako dobro so osebki razporejeni v posamezne razrede. Želimo si takšnih razredov, kjer so si elementi znotraj istega razreda dovolj podobni in je zato njihova povprečna razdalja do središčne točke majhna. Hkrati je zaželeno, da so posamezni razredi med seboj čim bolj oddaljeni. Če želimo oceniti kako dobro osebek ustreza posameznemu razredu, lahko uporabimo metodo silhuete.

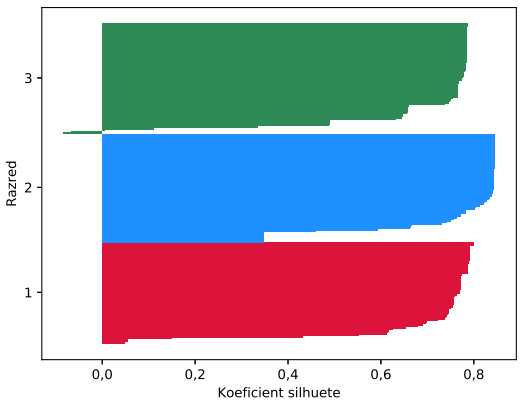
Metoda silhuete

Pri metodi silhuete ocenjujemo kvaliteto pripadnosti osebka znotraj posameznega razreda [29]. Če osebek o pripada razredu R_i in je njegova povprečna razdalja do vseh ostalih osebkov znotraj istega razreda enaka a, hkrati pa poiščemo tak razred R_j, kjer je povprečna razdalja do osebkov razreda R_j najmanjša glede na vse ostale razrede, potem lahko koeficient silhuete osebka o izrazimo z naslednjo enačbo:

$$s(o) = \frac{b - a}{max(a,b)}. \tag{4.18}$$

Iz enačbe 4.18 je razvidno, da bo vrednost $s(o)$ vedno med -1 in 1. Če je $s(o)$ negativna, potem si lahko osebek o interpretiramo kot osamelec znotraj svojega razreda, saj je povprečna razdalja znotraj njegovega razreda R_i večja od povprečne razdalje najbližjega razreda R_j.

Silhueto vseh osebkov lahko grafično prikažemo tako, da predstavimo razrede z različno barvo, osebke posameznega razreda pa razvrstimo glede na koeficient silhuete. Slika 4.12 prikazuje silhueto razbitja s tremi razredi pridobljenimi s hierarhičnim razvrščanjem 4.8.



Slika 4.12 Silhueta razbitja prikazanega na sliki 4.8. Opazimo lahko, da imajo nekateri osebki znotraj tretjega razreda negativno silhueto. Tretji razred ima tudi najnižjo povprečno vrednost silhuete. Rezultati silhuete sovpadajo z dendrogramom, saj ima tretji razred najvišjo Wardovo razdaljo.

Čeprav je silhueta metrika, ki kvantificira kako dobro se osebek o uvršča znotraj svojega razreda R_i, lahko uporabimo silhueto tudi kot mero za ocenjevanje kvalitete razbitja. Če ima veliko osebkov negativno vrednost silhuete, takšno razbitje ni kakovostno, saj bi

ti bolje pripadali drugim razredom. V takšnem primeru je morda smiselno uporabiti drugačno število razredov k.

4.5.2 Prikaz visokodimenzionalnih podatkov

Načeloma smo ljudje zelo intuitivni in se odlično znajdemo v dveh ali treh dimenzijah, če pa imamo opravka z visokomenzionalnimi podatki, si te veliko težje predstavljamo. Za predstavljivo vizualizacijo moramo število dimenzij v podatkih zmanjšati na tak način, da obdržimo glavne lastnosti, ki se v podatkih pojavljajo [29]. Najbolj preprost način zmanjševanja dimenzij so preseki. Pri preseku obdržimo le nekaj najpomembnejših dimenzij, ostale pa zanemarimo. Ker pa lahko pri preseku izgubimo velik delež informacije o obliki podatkov, so primernejši pristopi za zmanjševanje števila dimenzij različne projekcije. Dve pogosto uporabljeni metodi sta linearna diskriminantna analiza (ang. *linear discriminant analysis*) ali LDA [33] in metoda glavnih komponent (ang. *principal components analysis*) ali PCA [29, 34]. Ker metoda LDA maksimizira razliko aritmetičnih sredin med projeciranimi razredi in minimizira standardni odklon projeciranih razredov, je v našem primeru neprimerna, saj je pristranska. Razredi, ki smo jih pridobili z razvrščanjem bodo v projecirani ravnini metode LDA bolj narazen kot so v resnici. Metoda glavnih komponent je v našem primeru primernejša, saj maksimizira varianco projeciranih podatkov nad vsemi podatki hkrati.

Metoda glavnih komponent

Metoda glavnih komponent je metoda za zmanjševanje števila dimenzij, s katero projeciramo visokodimenzionalne podatke na nižjedimenzionalen prostor, ki ga sestavljajo ortogonalni vektorji glavnih komponent. Glavna komponenta je vektor, pri katerem bo varianca projeciranih podatkov maksimalna. Lepa lastnost te metode je, da v veliki meri ohranja obliko podatkov. Podatki, ki so blizu v celotnem prostoru, so zato blizu tudi v prostoru glavnih komponent. Tako lahko odkrivamo in analiziramo povezave in relacije, ki se pojavljajo v visokodimenzionalnih podatkih [29, 34].

Poglejmo postopek izračuna glavnih komponent na podatkih, predstavljenimi z matriko Y. Vrstice te matrike predstavljajo posamezne podatke, stolpci pa različne dimenzije. Matriko Y normaliziramo tako, da vsaki dimenziji odštejemo njeno povprečje:

$$X = [y_1 - \hat{y_1}, y_2 - \hat{y_2}, ...]. \tag{4.19}$$

Dobimo novo matriko X, katere kovariančna matrika je enaka:

$$C = X^T X. \tag{4.20}$$

Naredimo razcep kovariančne matrike C na lastne vektorje in lastne vrednosti:

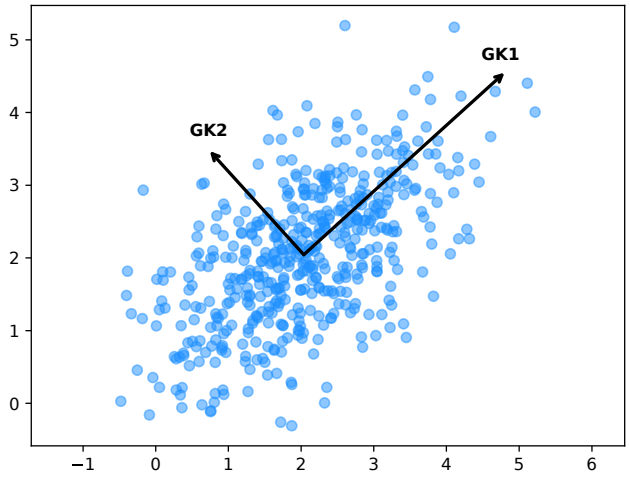
$$C = Q \Lambda Q^T. \tag{4.21}$$

Matrika lastnih vektorjev Q je ortogonalna in predstavlja glavne komponente naših podatkov. Diagonalna matrika lastnih vrednosti Λ pa predstavlja varianco, ki jo posamezna glavna komponenta opiše.

Ker pa imamo ponavadi veliko več podatkov kot je dimenzij, bo matrika X dolga in ozka, zato je v praksi nesmiselno izračunavati kovariančno matriko. Za določitev glavnih komponent uporabimo raje singularni razcep matrike X. Hitro lahko pokažemo, da predstavlja matrika desnih singularnih vrednosti glavne komponente, koreni singularnih vrednosti pa predstavljajo varianco, ki jo te komponente opišejo:

$$\begin{aligned} X &= U \Sigma V^T, \\ X^T X &= V \Sigma U^T U \Sigma V = V \Sigma^2 V^T. \end{aligned} \tag{4.22}$$

Če imamo opravka z večdimenzionalnimi podatki, obdržimo le nekaj glavnih komponent, ki opišejo največjo varianco podatkov. Tako bomo v projecirani ravnini še vedno obdržali velik delež variance. Ker pa je metoda glavnih komponent občutljiva na razpon podatkov, se pogosto predhodno poslužujemo standardizacije. Dimenzije standardiziramo tako, da imajo povprečje v točki 0 in je njihov standardni odklon enak 1. Na takšen način bo metoda glavnih komponent enakovredno vrednotila vse dimenzije. Prikaz metode glavnih komponent na naključno generiranih podatkih prikazuje slika 4.13. V tem primeru je problem trivialen, saj so orginalni podatki že v dveh dimenzijah.



Slika 4.13 Prikaz metode glavnih komponent v dveh dimenzijah. Podatke smo generirali naključno po normalni porazdelitvi dveh spremenljivk s kovariančno matriko $\Sigma = \begin{bmatrix} 1 & 3/5 \\ 3/5 & 1 \end{bmatrix}$ s povprečjem v točki $[2, 2]$. Vektorja $GK1$ in $GK2$ predstavljata dve glavni komponenti na kateri projeciramo podatke.

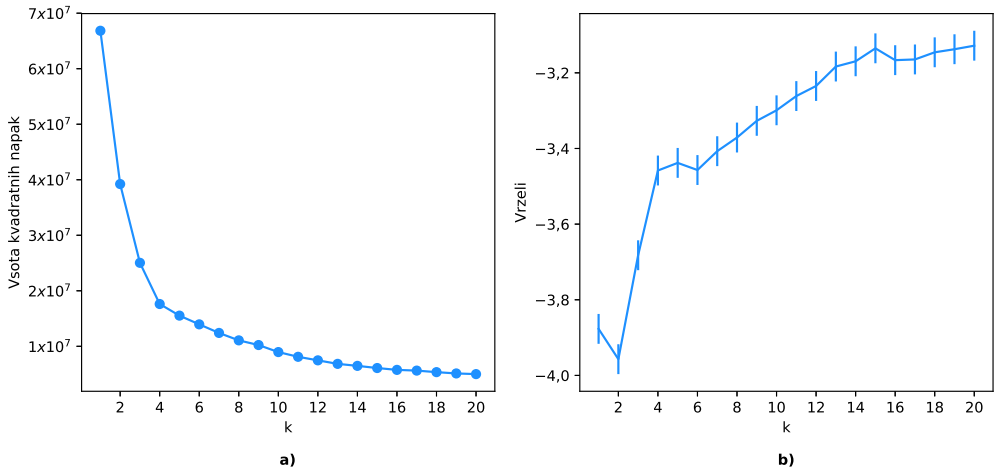
Za realizacijo metode glavnih komponent v programskem jeziku *Python* uporabimo paket *decomposition PCA* knjižnjice $scikit-learn$. Pred uporabo metode glavnih komponent podatke standardiziramo.

5 Rezultati

V prejšnjem poglavju smo predstavili in opisali predlagano metodologijo za analizo dinamičnih bioloških modelov z visokodimenzionalnim prostorom dopustnih rešitev. V tem poglavju opišemo rezultate, ki smo jih z metodologijo pridobili na dinamičnih bioloških modelih represilatorja in pomnilne celice D s predpomnenjem. Oba modela smo vzorčili z genetskim algoritmom, ki smo ga pognali 50-krat. Zanima nas, na koliko skupin moramo vzorce razdeliti, kakšno delovanje izkazujejo in kako vpliva posamezen kinetičen parameter na delovanje modela.

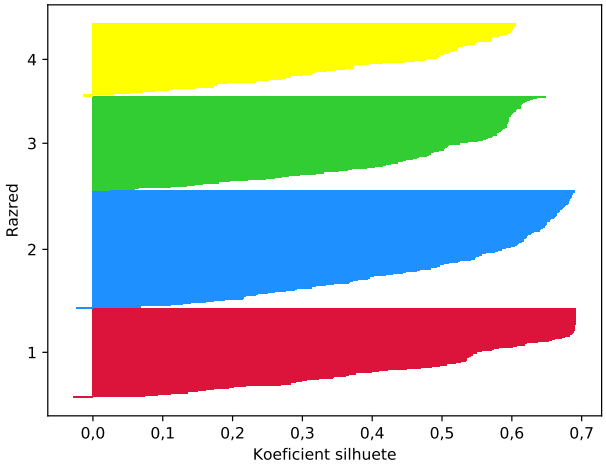
5.1 Rezultati modela biološkega represilatorja

Pri modelu biološkega represilatorja smo z vzorčenjem z genetskim algoritmom pridobili 22272 vzorcev. Poglejmo si, na koliko skupin bi bilo vzorce smiselno razdeliti. Slika 5.1 prikazuje krivulji komolčne metode in statistike vrzeli na podatkih represilatorja. Obe metodi nakazujeta na optimalno število razredov pri $k = 4$, vendar moramo biti pri statistiki vrzeli nekoliko bolj previdni. Krivulja statistike vrzeli nakazuje na en optimalen razred, saj je ta vrzel višja od vrzeli dveh razredov. Vrzel nato strmo naraste in se pri $k = 4$ ustali. Na podlagi obeh metod se torej odločimo za štiri razrede, v katere bomo razvrstili naše podatke.



Slika 5.1 Krivulji komolčne metode in statistike vrzeli na podatkih modela biološkega represilatorja. Slika a) prikazuje krivuljo komolčne metode, slika b) pa krivuljo statistike vrzeli. Pri obeh metodah smo predpostavili maksimalno število razredov $k = 20$.

Z algoritmom razvrščanja z voditelji smo razvrstili vzorce v štiri razrede. Dodatno moramo preveriti, če je tako razbitje ustrezno. Slika 5.2 prikazuje silhueto razredov danega razbitja. Najvišjo silhueto ima razred 1, sledijo mu razred 2, 3 in 4. Znotraj razredov ima negativno vrednost silhuete le majhen delež vseh vzorcev, zato lahko sklepamo, da je razbitje kvalitetno.



Slika 5.2 Silhueta štirih razredov razvrščanja z voditelji na vzorcih modela represilatorja.

Ugotovili smo že, da je razbitje na štiri razrede kvalitetno. Kakšno obnašanje pa izkazujejo posamezni vzorci znotraj teh razredov? Zanima nas kako je posamezen razred zastopan, kako osebki v prostoru v povprečju odstopajo od predstavnika razreda in kakšne so povprečne vrednosti cenovne funkcije, amplitude ter periode in njihovi standardni odkloni. Zanima nas tudi kakšno delovanje izkazujejo predstavniki razredov. Za predstavnike razredov vzemimo kar njihova težišča. Izkaže se, da imajo ti osebki bistveno slabšo vrednost cenovne funkcije od povprečja, za kar obstajata dve možni razlagi. Prvič, vzorci so se razvrstili po lokalnih ekstremih, te pa smo nato razvrstili v en razred. Povprečna vrednost cenovne funkcije je zato nizka. Ker pa je težišče razreda na sredini med lokalnimi ekstremi, je njegova vrednost cenovne funkcije višja. Drugič, optimalno delovanje izkazujejo rešitve na robu prostora dopustnih rešitev. Večina vzorcev se zato razporedi ravno ob robu in je zato povprečna vrednost cenovne funkcije nizka. Ostali vzorci višajo povprečno vrednost cenovne funkcije, hkrati pa v večji meri premikajo težišče razreda stran od roba rešitev, zaradi česar je vrednost cenovne funkcije težišča višja od povprečja. V prvem primeru, bi morali komolčna metoda in metoda statistika vrzeli nakazati na višje število razredov. Ker temu ni tako, lahko sklepamo, da je druga razlaga verjetnejša. Za nove predstavnike razredov zato izberemo težišča 10% najboljših osebkov znotraj posameznega razreda. Lastnosti posameznih razredov modela represilatorja prikazuje tabela 5.1. Najbolj zastopan je razred 2, sledijo mu razred 3, 1 in 4. Razred 2

ima tudi najboljšo vrednost cenovne funkcije, najvišjo povprečno amplitudo in periodo. Standardni odklon amplitude in periode je pri vseh razredih visok. To je pričakovano, saj smo sestavili takšno cenovno funkcijo, ki ne kaznuje signalov z različno amplitudo in periodo oscilacij, a vseeno favorizira signale z višjo amplitudo in periodo. Želeli bi si, da so razredi z nižjo povprečno vrednostjo cenovne funkcije po prostoru bolj razpršeni. Temu žal ni tako, saj imata najvišjo varianco razreda 3 in 4. Omeniti moramo, da lahko vse štiri razrede smatramo za kvalitetne, saj vsi izkazujejo visoko povprečno amplitudo in periodo oscilacij, vendar je na podlagi podatkov iz tabele 5.1 razred 2 najrobustnejši.

Tabela 5.1 Lastnosti posameznih razredov modela represilatorja.

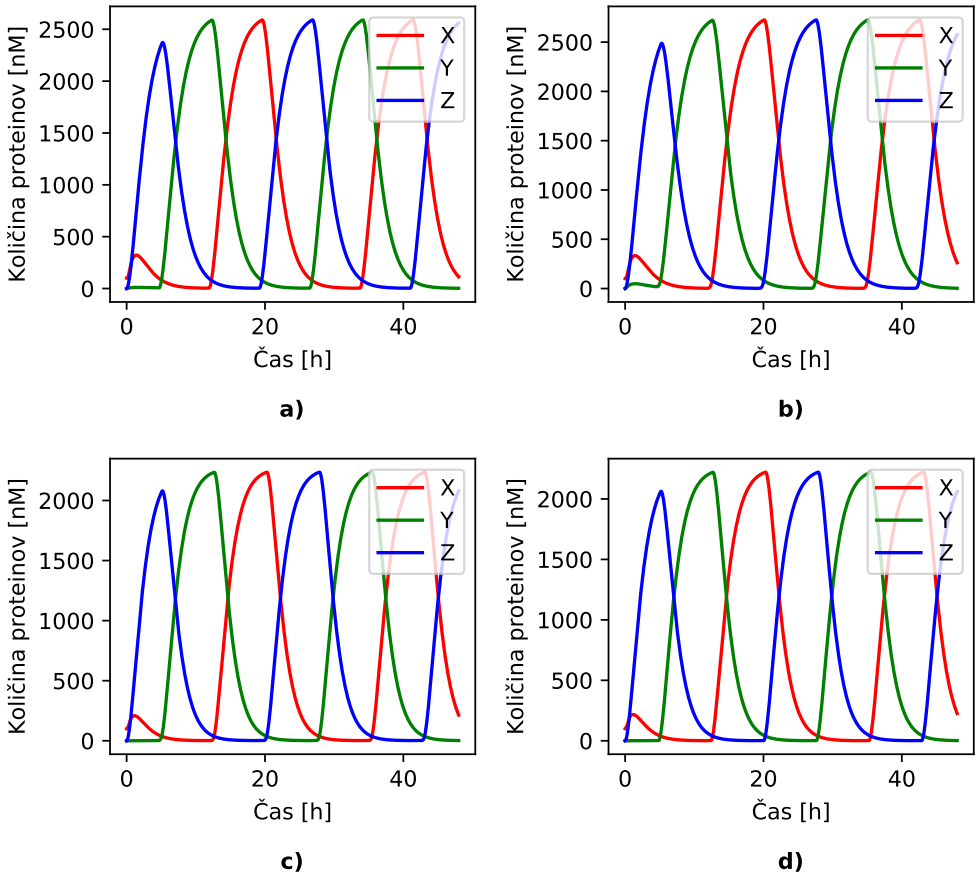
	Razred 1	Razred 2	Razred 3	Razred 4
Število osebkov	5336	7190	5788	3958
Varianca razreda	22,62	22,35	26,85	26,44
Povprečna vrednost cenovne funkcije	−1791348	−2019712	−1427360	−1405116
Deviacija cenovne funkcije	1951092	2078382	1570637	1607450
Povprečje amplitude $[nM]$	925	1018	742	702
Deviacija amplitude $[nM]$	718	753	568	565
Povprečje periode $[h]$	15,55	16,81	15,63	14,39
Deviacija periode $[h]$	10,18	9,14	9	10,7
Vrednost cenovne funkcije predstavnika razreda	−159365	−197158	−104259	−141946
Vrednost cenovne funkcije predstavnika najboljših 10%	−5129479	−4608700	−4540441	−4526905

Poglejmo si, kakšno delovanje izkazujejo predstavniki razredov in kakšni so njihovi kinetični parametri. Tabela 5.2 prikazuje kinetične parametre predstavnikov razredov. Opazimo lahko, da je hitrost izražanja ob odsotnosti represorja v vseh primerih visoka, medtem ko je hitrost brezpogojnega izražanja nizka. Hitrost genskega prevajanja je za predstavnike vseh razredov podobna in znaša približno 74 h^{-1}. Hitrosti degradacije informacijske RNA in proteinov sta v vseh primerih reda 10^{-1} h^{-1}. Hillov koeficient n in disociacijska konstanta K_d zavzemata različne vrednosti za različne predstavnike razredov, kar nakazuje, da na kvaliteto rešitve nimata občutnega vpliva.

Tabela 5.2 Kinetični parametri predstavnikov najboljših 10% osebkov razredov za model represilatorja.

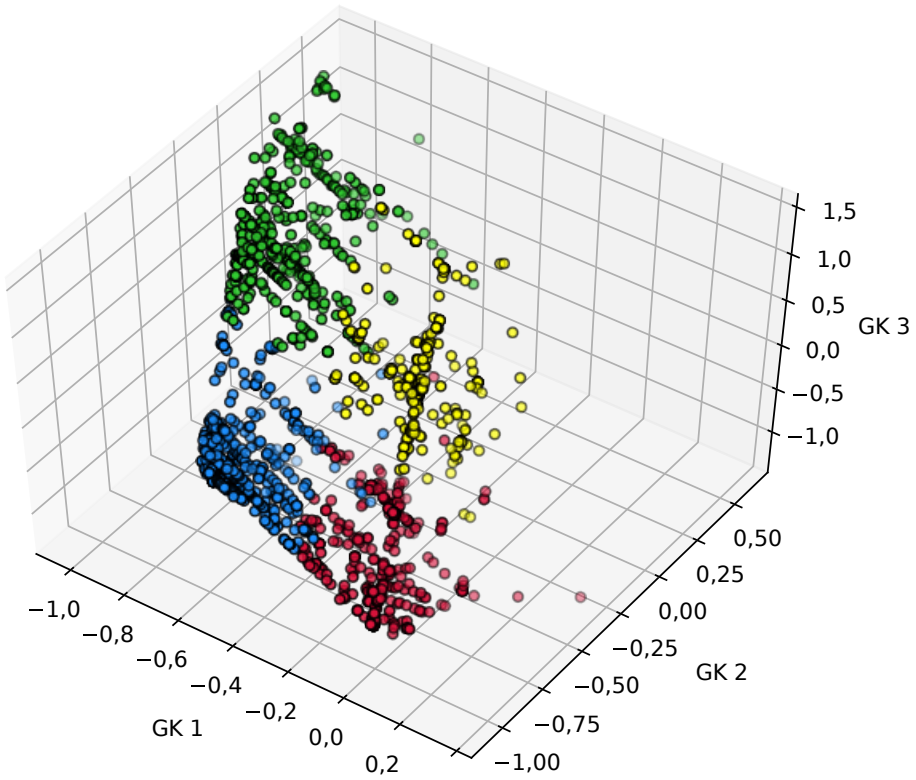
Parameter	Razred 1	Razred 2	Razred 3	Razred 4
α	19,85	19,92	19,9	19,85
α_0	$1{,}2 * 10^{-2}$	$2{,}33 * 10^{-3}$	$8{,}22 * 10^{-3}$	$1{,}39 * 10^{-3}$
n	91,26	17,52	17,97	93,82
β	74,34	74,76	74,32	73,68
δ_m	$7{,}4 * 10^{-1}$	$8 * 10^{-1}$	$8{,}5 * 10^{-1}$	$9{,}1 * 10^{-1}$
δ_p	$7{,}53 * 10^{-1}$	$6{,}67 * 10^{-1}$	$7{,}68 * 10^{-1}$	$7{,}13 * 10^{-1}$
K_d	96,7	97,7	41,47	44,96

Slika 5.3 prikazuje simulacijske rezultate predstavnikov razredov. Opazimo lahko, da je odziv predstavnikov vseh razredov podoben. Vsi predstavniki imajo približno enako periodo 20 h. Amplituda predstavnikov razredov 1 in 2 znaša približno 2500 nM, medtem ko znaša amplituda ostalih dveh predstavnikov razredov približno 2000 nM.



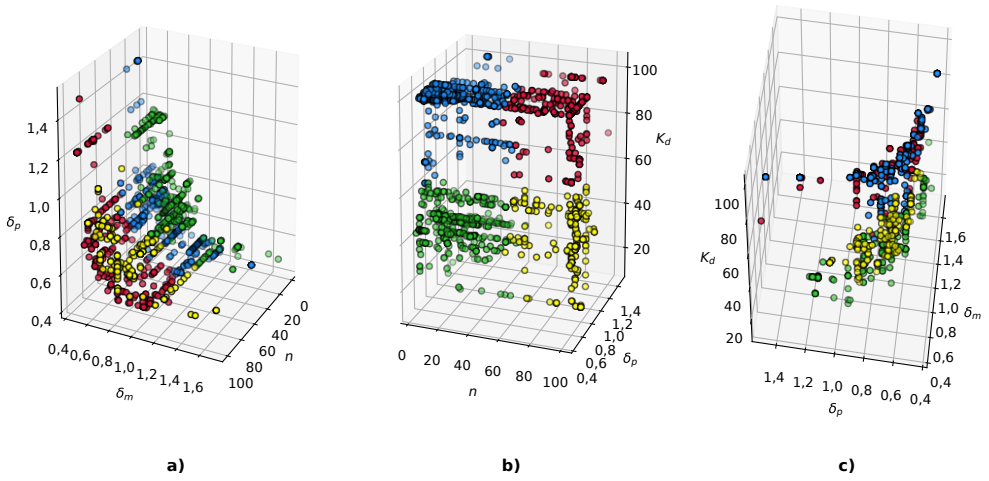
Slika 5.3 Simulacije modela represilatorja za predstavnike najboljših 10% osebkov znotraj posameznih razredov. Slika a) predstavlja razred 1, slika b) predstavlja razred 2, slika c) predstavlja razred 3, slika d) predstavlja razred 4.

Do sedaj smo določili optimalno število razredov za razvrščanje, ocenili kvaliteto razbitja ter analizirali posamezne razrede in njihove predstavnike. Kljub vsemu še vedno nimamo občutka, kako so vzorci modela biološkega represilatorja razporejeni po prostoru dopustnih rešitev. V ta namen podatke vizualiziramo, pri čemer za redukcijo dimenzij uporabimo metodo glavnih komponent. Vzorce standardiziramo in jih projeciramo na prve tri glavne komponente. Slika 5.4 prikazuje projekcijo osebkov modela represilatorja na ravnino prvih treh glavnih komponent. Opazimo lahko štiri jasno definirane razrede, ki so med sabo povezani. Osebki so v središču razredov nekoliko bolj zgoščeni. Prostor dopustnih rešitev biološkega modela represilatorja spominja v projekciji glavnih kompinent na nekakšno štiristrano prizmo.



Slika 5.4 Projekcija podatkov represilatorja na prve tri glavne komponente, kjer predstavlja rdeča barva osebke razreda 1, modra osebke razreda 2, zelena osebke razreda 3 in rumena osebke razreda 4. Zaradi lepše vizualizacije prikažemo le 10% najboljših osebkov znotraj posameznega razreda.

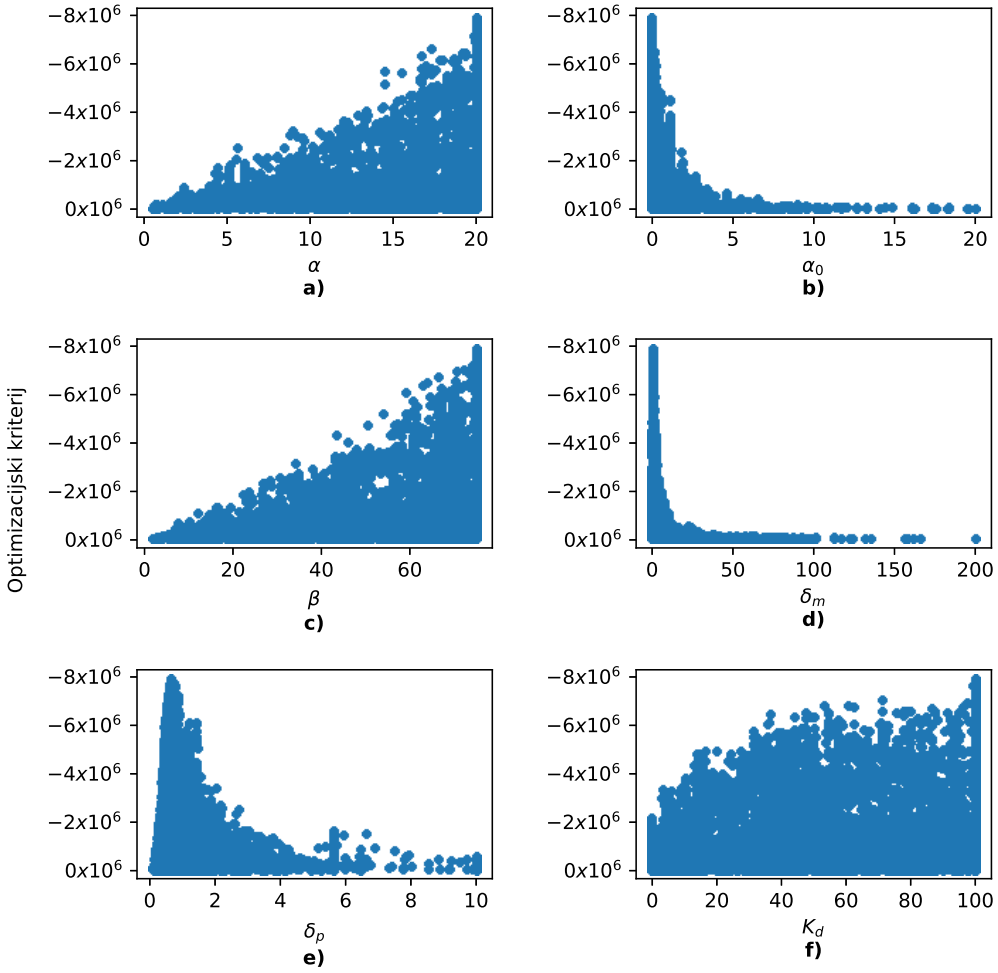
Kot zanimivost si poglejmo, kakšne oblike tvorijo podatki v preseku posameznih dimenzij. Vseh presekov v nalogi ne prikazujemo in izpostavimo le nekaj najbolj zanimivih. Slika 5.5 prikazuje vzorce modela represilatorja v preseku posameznih dimenzij.



Slika 5.5 Prikaz vzorcev modela represilatorja v preseku posameznih dimenzij. Različne barve predstavlja osebke posameznih razredov. Rdeča predstavlja razred 1, modra razred 2, zelena razred 3 in rumena razred 4. Slika a) prikazuje presek za kinetične parametre δ_p, δ_m in n. Podatki v tem preseku tvorijo vzpetino, ki je sprva položna, nato pa se ta začne vzpenjati. Slika b) prikazuje presek za kinetične parametre n, δ_p in K_d. Ti tvorijo nekakšen kvadrat. Slika c) prikazuje presek za kinetične parametre K_d, δ_p in δ_m. V tem preseku je moč opaziti del plašča valja. Zaradi lepše vizualizacije prikažemo le 10% najboljših osebkov znotraj posameznega razreda.

Pri analizi bioloških modelov nas zanima tudi, kako vpliva posamičen kinetičen parameter na delovanje sistema. Podatke lahko vizualiziramo tako, da predstavimo vsak osebek kot točko na grafu dveh dimenzij. Os x predstavlja vrednost kinetičnega parametra, os y pa vrednost cenovne funkcije. Slika 5.6 prikazuje vrednosti cenovne funkcije pri različnih vrednostih posameznih kinetičnih parametrov. Omeniti je potrebno, da je na sliki os vrednosti cenovne funkcije zaradi lepšega prikaza invertirana. Vrednost cenovne funkcije pada pri višanju vrednosti kinetičnih parametrov α, β in K_d. Pri parametru K_d padec cenovne funkcije ni tako izrazit. Pri višanju vrednosti kinetičnih parametrov α_0 in δ_m pa vrednost cenovne funkcije narašča. Pri parametru δ_p je vrednost cenovne funkcije na začetku 0, ta nato hitro pada in doseže minimum pri vrednosti okrog 0,8. Hillovega koeficienta tukaj ne prikazujemo, saj ta nima takšnega vpliva na vrednost cenovne funkcije. Iz slike 5.6 lahko razberemo tudi, kako občutljiv je model na perturbacijo

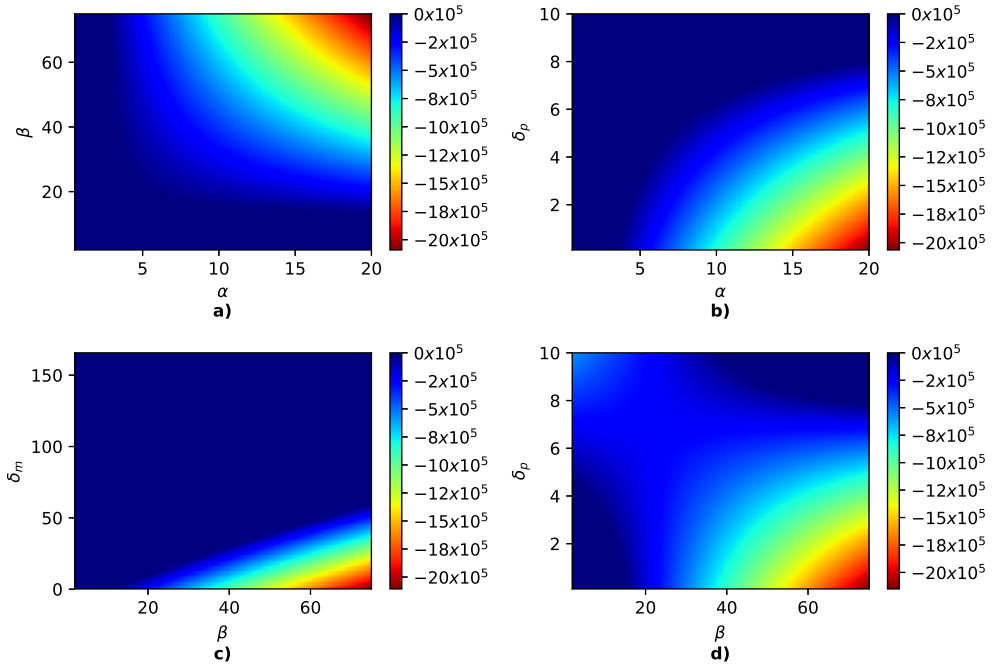
posameznega kinetičnega parametra. Parameter pri katerem bo cenovna funkcija močno padala ali naraščala ima večji vpliv na odziv modela od parametra, ki ne vpliva na vrednost cenovne funkcije. Paramteri α, α_0, β, δ_m in δ_p imajo velik vpliv na odziv sistema, medtem ko parametra K_d in n takšnega vpliva nimata.



Slika 5.6 Graf raztrosa za vrednost cenovne funkcije pri različnih parametrih modela represilatorja. Modre točke na grafih predstavljajo posamezne osebke. Zaradi lepšega prikaza je os, ki predstavlja vrednost cenovne funkcije, invertirana.

Vpliv dveh kinetičnih parametrov na vrednost cenovne funkcije lahko ponazorimo s toplotno karto. Osi predstavljata različna kinetična parametra, medtem ko je vrednost cenovne funkcije prikazana z različno barvo. Slika 5.7 prikazuje toplotne karte za različne

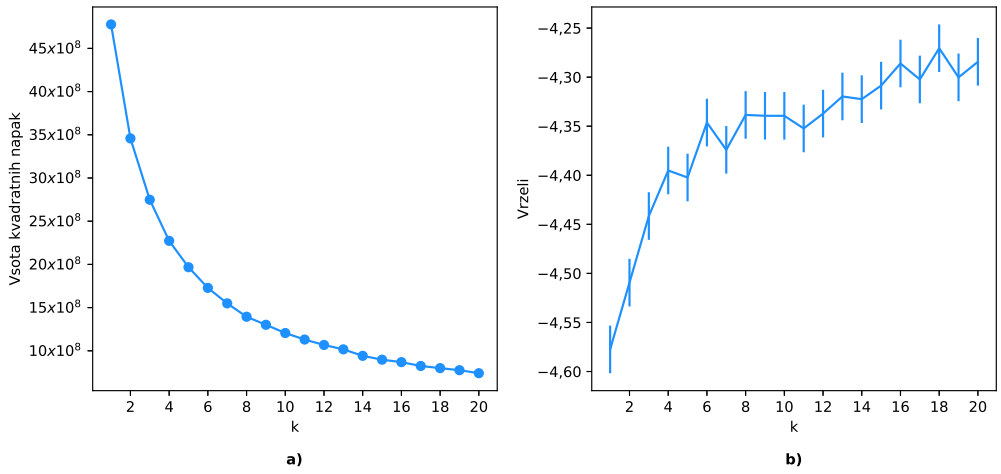
pare kinetičnih parametrov. Za potrebe vizualizacije podatke z linearno interpolacijo predhodno interpoliramo na mrežo velikosti $150 * 150$ točk [14]. Tudi ta prikaz potrjuje, da so vzorci modela represilatorja razporejeni po robu prostora dopustnih rešitev. Z večanjem vrednosti parametrov α, α_0 in β se vrednost cenovne funkcije niža, medtem ko se z višanjem parametrov δ_p in δ_m vrednost cenovne funkcije viša. Toplotne karte ostalih parov kinetičnih parametrov niso reprezentative, zato jih v nalogi ne prikazujemo.



Slika 5.7 Toplotna karta modela biološkega represilatorja za različne pare kinetičnih parametrov. Vrednost cenovne funkcije je prikazana z različno barvo. Rdeča predstavlja nizko vrednost cenovne funkcije, medtem ko predstavlja modra barva visoko vrednost cenovne funkcije.

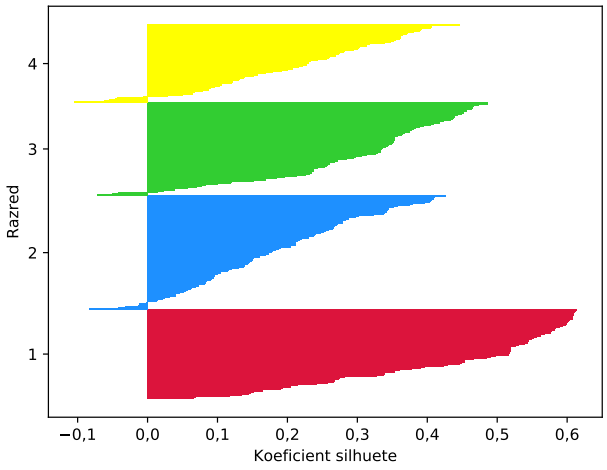
5.2 Rezultati modela biološke pomnilne celice D s predpomnenjem

Pri modelu biološke pomnilne celice D s predpomnenjem smo z vzorčenjem z genetskim algoritmom pridobili 15641 vzorcev. Slika 5.8 prikazuje krivulji komolčne metode in statistike vrzeli na podatkih modela biološke pomnilne celice. Pri modelu biološke pomnilne celice D optimalno število razredov ni tako očitno kot je pri modelu represilatorja. Na podlagi komolčne metode tudi ne moremo določiti optimalnega števila razredov, saj komolec ni tako izrazit, zato se za izbiro optimalnega števila razredov opremo na metodo statistika vrzeli. Ta nakaže na optimalno število razredov pri $k = 4$, saj je vrzel štirih razredov višja od razlike vrzeli petih razredov in njene standardne napake. Krivulja vrzeli strmo narašča in se pri štirih razredih počasi ustali, zato za število razredov pri razvrščanju z voditelji tudi tukaj izberemo štiri razrede, v katere bomo razvrščali podatke.



Slika 5.8 Graf komolčne metode in statistike vrzeli na podatkih modela biološke pomnilne celice D s predpomnenjem. Slika a) prikazuje krivuljo komolčne metode, slika b) pa krivuljo statistike vrzeli. Pri obeh metodah smo predpostavili maksimalno število razredov $k = 20$.

Slika 5.9 prikazuje silhueto razredov pridobljenimi z razvrščanjem z voditelji. Prevladuje prvi razred, ki ima največjo silhueto, sledi mu razred 3, razreda 2 in 4 pa imata najnižjo vrednost silhuete. Vrhovi silhuete razredov 2, 3 in 4 so tudi ožji od vrha silhuete razreda 4. Silhuete razredov modela pomnilne celice D so nižje od silhuet razredov modela represilatorja in vsebujejo nekoliko večji delež osebkov z negativno silhueto. Kljub temu je ta delež dokaj majhen, zato lahko predpostavimo, da je razbitje kakovostno. Nekoliko slabša silhueta sovpada tudi s krivuljama komolčne metode in statistike vrzeli, kar nakazuje, da podatki modela pomnilne celice D v prostoru dopustnih rešitev ne tvorijo tako jasno definiranih skupin.



Slika 5.9 Silhueta štirih razredov razvrščanja z voditelji na vzorcih modela biološke pomnilne celice D s predpomnenjem.

Poglejmo si kako je posamezen razred zastopan, kako so osebki znotraj razreda razpršeni ter kakšne so povprečne vrednosti cenovne funkcije, amplitude ter periode in kakšni so njihovi standardni odkloni. Zanima nas tudi, kakšno delovanje izkazujejo predstavniki razredov. Lastnosti razredov modela biološke pomnilne celice D s predpomnenjem prikazuje tabela 5.3. Najbolj zastopan je razred 2, sledijo mu razredi 3, 4 in 1. Osebki razreda 2 so tudi najbolj razpršeni. Povprečna amplituda razredov znaša približno 97 nM, povprečna perioda razredov pa je približno 25 h. Standardna odklona sta pri amplitudi in periodi nizka. To je pričakovano, saj cenovna funkcija, ki smo jo uporabili pri preiskovanju prostora modela biološke pomnilne celice, favorizira oscilacije z amplitudo 100 nM in periodo 24 h. Predstavniki razredov imajo bistveno boljšo vrednost cenovne funkcije od povprečja, kar pomeni, da so se osebki razporedili po lokalnih ekstremih. Tudi tukaj je razred 2 najrobustnejši, saj je močno zastopan, podatki so po razredu razpršeni, njegovi osebki pa izkazujejo optimalno delovanje s pravilno periodo in amplitudo oscilacij z nizkim standardnim odklonom.

Tabela 5.3 Lastnosti posameznih razredov modela biološke pomnilne celice D s predpomnenjem.

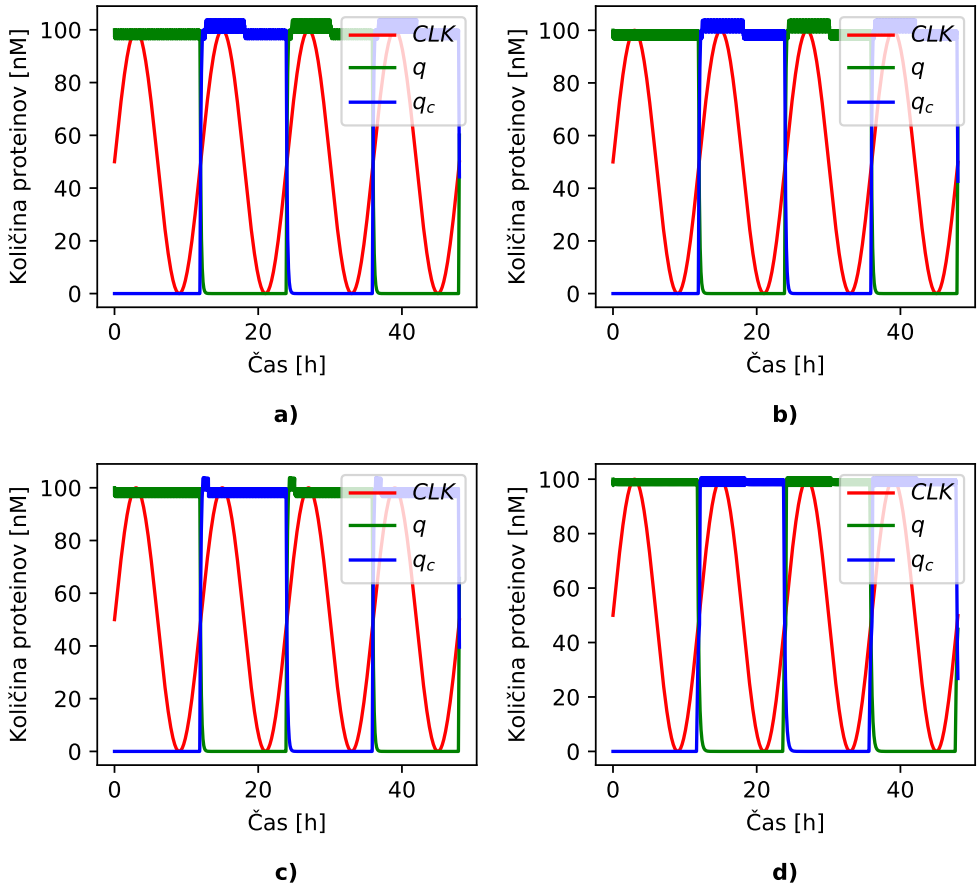
	Razred 1	Razred 2	Razred 3	Razred 4
Število osebkov	3431	4600	3938	3672
Varianca razreda	231,8	355,48	322,72	332,82
Povprečna vrednost cenovne funkcije	88073	104335	90447	137567
Deviacija cenovne funkcije	125780	122052	116628	129325
Povprečje amplitude [nM]	96,94	97,67	97,2	96,05
Deviacija amplitude [nM]	8,5	7,21	9,29	9,82
Povprečje periode [h]	25,03	25,31	24,92	25,17
Deviacija periode [h]	2,82	2,84	2,59	2,51
Vrednost cenovne funkcije predstavnika razreda	21231	28366	18540	34769

Tabela 5.4 prikazuje kinetične parametre predstavnikov razredov modela biološke pomnilne celice. Opazimo lahko, da so tako hitrosti produkcije kot tudi degradacije proteinov visoke. To je potrebno, če želimo hitre prehode iz nizkega v visoko stanje. Disociacijske konstante so po večini nizke, razen Kd_7, ki je visoka pri vseh razredih.

Tabela 5.4 Kinetični parametri predstavnikov razredov za model biološke pomnilne celice D s predpomnenjem.

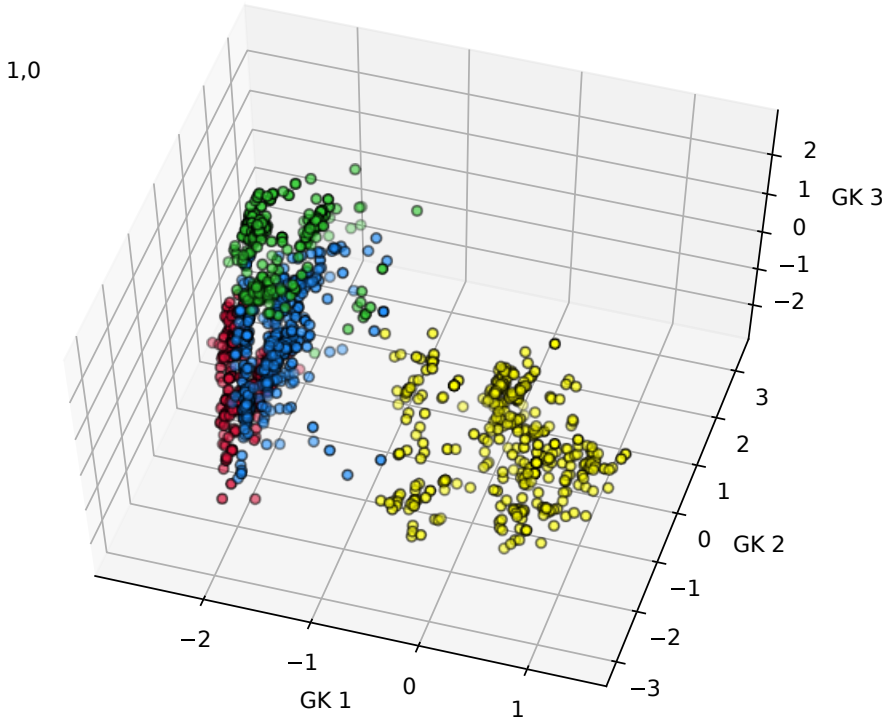
Parameter	Razred 1	Razred 2	Razred 3	Razred 4
α_1	622.71	156,75	484,59	610,2
α_2	663	529,62	61,84	557,61
α_3	684,86	474,43	559,73	269,79
α_4	678,09	577,66	653,21	457,21
δ_1	4,84	3,79	2,1	4,68
δ_2	5,26	3,96	4,89	2,79
Kd_1	26,27	26,08	31,66	33,05
Kd_2	24,19	23,36	27,71	23,12
Kd_3	15,58	13,54	21,78	18,97
Kd_4	66,28	58,67	47,79	56,17
Kd_5	47,46	42,85	44,9	40,96
Kd_6	10,06	20,47	20,98	23,83
Kd_7	97,64	97,77	98,07	96,67

Slika 5.10 prikazuje simulacijske rezultate predstavnikov razredov modela pomnilne celice D s predpomnenjem. Vsi predstavniki izkazujejo optimalno delovanje z amplitudo 100 nM in periodo 24 h.



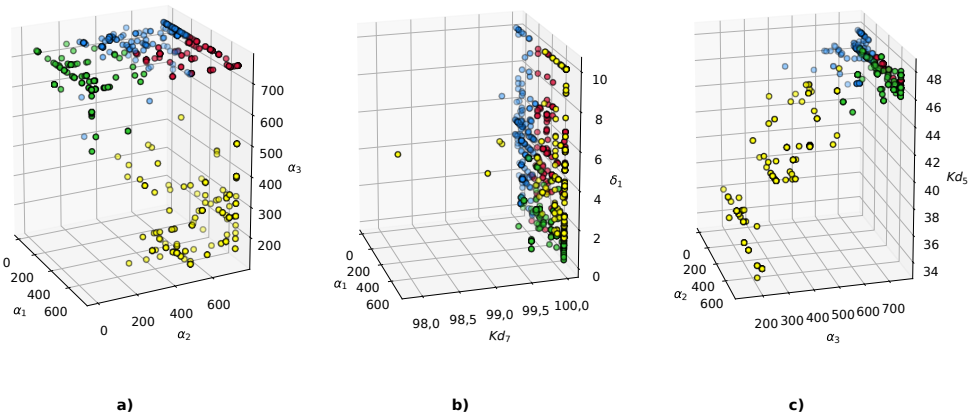
Slika 5.10 Simulacije modela biološke pomnilne celice D s predpomnenjem za predstavnike razredov. Slika a) predstavlja razred 1, slika b) predstavlja razred 2, slika c) predstavlja razred 3, slika d) predstavlja razreda 4.

Kako pa so vzorci modela pomnilne celice D s predpomnenjem razporejeni po prostoru dopustnih rešitev? Slika 5.11 prikazuje projekcijo standardiziranih vzorcev modela pomnilne celice D s predpomnenjem na prve tri glavne komponente. Opazimo lahko, da so prvi trije razredi močno povezani, medtem ko je razred 4 ločen. Razred 1 se razredu 2 močno prilega. Iz projekcije je razvidno, da bi lahko osebke razvrstili tudi v dve ali tri skupine.



Slika 5.11 Projekcija prvih treh glavnih komponent na podatkih modela biološke pomnilne celice D s predpomnenjem, kjer predstavlja rdeča barva osebke razreda 1, modra osebke razreda 2, zelena osebke razreda 3 in rumena osebke razred 4. Zaradi lepše vizualizacije prikažemo le 10% najboljših osebkov znotraj posameznega razreda.

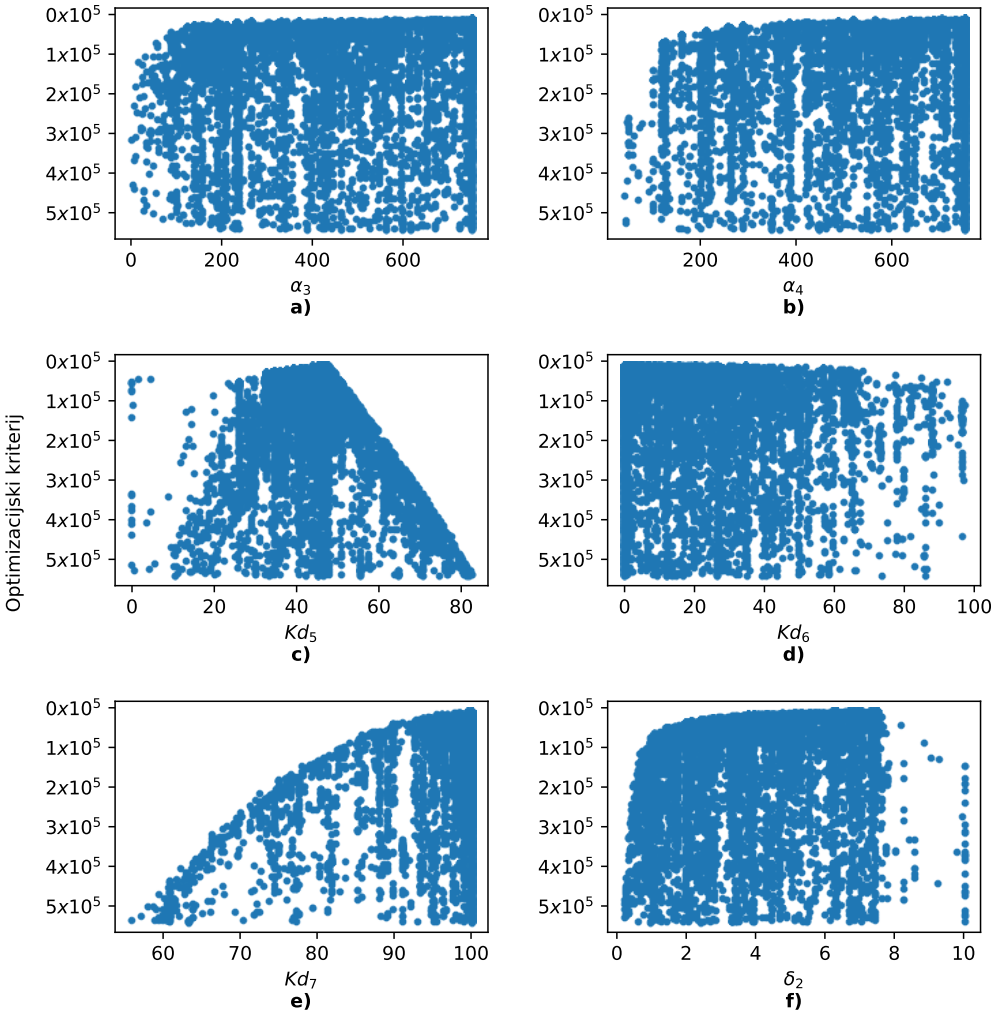
Zanimivi so tudi preseki posameznih dimenzij. Slika 5.12 prikazuje preseke različnih dimenzij za vzorce modela pomnilne celice. Vseh presekov v nalogi ne prikazujemo in izpostavimo le nekaj najbolj zanimivih.



Slika 5.12 Prikaz vzorcev modela biološke pomnilne celice D s predpomnenjem v preseku posameznih dimenzij. Različne barve predstavljajo osebke posameznih razredov. Rdeča predstavlja razred 1, modra razred 2, zelena razred 3 in rumena razred 4. Slika a) prikazuje presek za kinetične parametre α_1, α_2 in α_3. Vzorci razredov 1, 2 in 3 na robu parametra α_3 in tvorijo ravnino, medtem ko so vzorci razreda 4 ločeni. Slika b) prikazuje presek za kinetične parametre α_1, Kd_7 in δ_1. Tudi tukaj se vzorci razporedijo po robu parametra Kd_7. Slika c) prikazuje presek za kinetične parametre α_2, α_3 in Kd_5. Vzorci razredov 1, 2 in 3 se razporedijo na robu kinetičnega parametra α_3. Vrednosti parametra Kd_5, ki jo ti vzorci zavzamejo so med 44 nM in 48 nM. Osebki razreda 4 tvorijo nekakšen lok. Zaradi lepše vizualizacije prikažemo le 10% najboljših osebkov znotraj posameznega razreda.

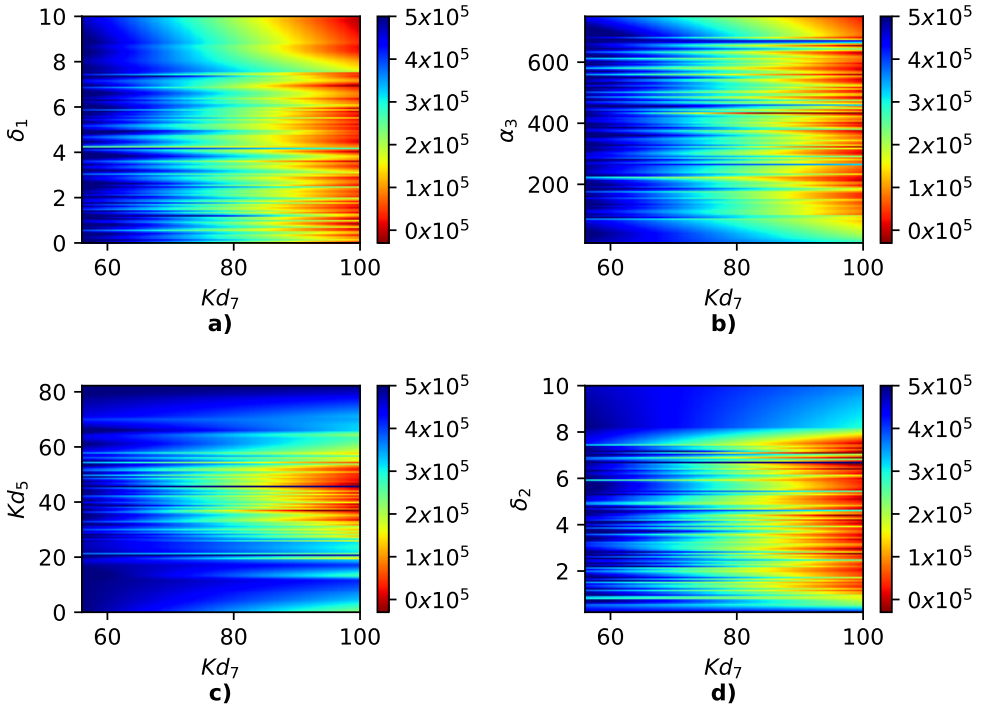
Poglejmo še, kako vpliva posamezen kinetičen parameter na odziv modela. Slika 5.13 prikazuje vrednost cenovne funkcije za različne kinetične parametre. Omeniti je potrebno, da je zaradi lepšega prikaza tudi tukaj os vrednosti cenovne funkcije invertirana. Z večanjem vrednosti parametrov α_3, α_4 in Kd_7 se vrednost cenovne funkcije niža. Pri parametru Kd_5 in δ_2 se z večanjem vrednosti parametra cenovna funkcija niža in doseže minimum približno pri 50 nM. Prav tako se z večanjem vrednosti parametra δ_2 cenovna funkcija niža in doseže minimum približno pri 7,5 h^{-1} Pri Kd_5 z večanjem vrednosti parametra vrednost cenovne funkcije počasi narašča. Parametri Kd_5, Kd_7 in δ_2 imajo močan vpliv na odziv modela. Rečemo lahko tudi, da je model občutljiv na te parametre. Ostalih parametrov ne prikazujemo, saj ti ne vplivajo toliko na vrednost cenovne funkcije. Vsi kinetični parametri prikazani na sliki 5.13 nastopajo pri kemijskih rekcijah produkcije

in degradacije izhodnih proteinov q in q_c, zato imajo od ostalih kinetičnih parametrov, ki pri teh reakcijah ne nastopajo, tudi večji vpliv na dinamiko modela.



Slika 5.13 Graf raztrosa za vrednost cenovne funkcije pri različnih parametrih modela biološke pomnilne celice D s predpomnenjem. Modre točke na grafih predstavljajo posamezne osebke. Zaradi lepšega prikaza je os, ki prikazuje vrednost cenovne funkcije, invertirana.

Vplive kombinacij dveh kinetičnih parametrov modela biološke pomnilne celice D s predpomnenjem ponazorimo s toplotno karto. Slika 5.14 prikazuje toplotne karte za različne pare kinetičnih parametrov. Podatke predhodno z linearno interpolacijo še interpoliramo na mrežo velikosti $150 * 150$. V vseh primerih lahko opazimo, da se z večanjem parametra Kd_7 niža tudi vrednost cenovne funkcije. Parametra δ_1 in α_3 na vrednost cenovne funkcije nimata takšnega vpliva. Model izkazuje najoptimalnejše delovanje, če zavzema kinetični paremeter Kd_5 vrednosti med 30 in 60 nM. Prav tako mora za optimalno delovanje pomnilne celice D s predpomnenjem kinetični parameter δ_2 zavzemati vrednosti med 1 in 7,5 nM. Ozke vodoravne črte na toplotnih kartah so posledica napak pri interpolaciji podatkov in ne predstavljajo dejanskega stanja sistema. Toplotne karte ostalih parov kinetičnih parametrov niso tako reprezentativne, zato jih v nalogi ne prikazujemo.



Slika 5.14 Temperaturni prikaz modela biološke pomnilne celice D s predpomnenjem. Vrednost cenovne funkcije je prikazana z različno barvo. Rdeča predstavlja nizko vrednost cenovne funkcije, medtem ko predstavlja modra barva visoko vrednost cenovne funkcije.

6 Diskusija

Analiza vpliva posameznih kinetičnih parametrov na odziv modela nam pove, za katere vrednosti parametrov bo biološki model izkazoval želeno delovanje. V kolikor nas zanima velikost in oblika prostora dopustnih rešitev, moramo ubrati drugačne pristope. Metodologija za analizo prostora dopustnih rešitev dinamičnih bioloških modelov, ki smo jo v delu predlagali je primerna, če prostor vsebuje večje število lokalnih ekstremov, kar je tudi pogoj za nepovezanost prostora dopustnih rešitev. S predlagano metodologijo lahko najdemo in analiziramo različne regije, ki se v prostoru dopustnih rešitev pojavljajo, identificiramo najrobustnejše območje ter dobimo občutek o velikosti in povezanosti prostora dopustnih rešitev kinetičnih parametrov.

Prednost predlagane metodologije je, da je intuitivna in preprosta za implementacijo. V našem primeru smo za implementacijo uporabili programski jezik *Python*, kjer je večina algoritmov, ki v metodologiji nastopajo, že implementiranih. Slabost metodologije je, da je počasna. Prostor rešitev moramo v celoti preiskati, kar je časovno potratno. Predlagana metodoligija je zato primerna, če optimalnih rešitev kinetičnih parametrov ne poznamo in moramo prostor rešitev v vsakem primeru preiskati.

Koncept predlagane metodologije smo za analizo občutljivosti pomnilne celice D s predpomnenjem uporabili že v [13], kjer smo ugotovili, da imajo kinetični parametri reakcij produkcije in degradacije izhodnih proteinov večji vpliv od ostalih kinetičnih parametrov. Ker pa smo za oceno vpliva posameznega parametra uporabili Morrisovo analizo občutljivosti [7, 8, 10, 13] le v lokalni okolici optimalnih rešitev, lahko ti rezultati od naših nekoliko odstopajo.

Omeniti moramo še, da obstajajo tudi drugi možni pristopi za analizo prostora dopustnih rešitev kinetičnih parametrov. V [35] avtorji najprej prostor rešitev vzorčijo in ga nato aproksimirajo s polinomi višjih stopenj več spremenljivk. Prav tako v [36] avtorji prostor rešitev vzorčijo, ki ga nato omejijo z elipsoidami. Te se lahko prekrivajo, zato lahko dobro okarektiziramo tudi nekonveksne regije. Vsem tem metodologijam je skupno vzorčenje prostora rešitev. Glavna njihova slabost je v tem, da se v primeru pomanklji-vega vzorčenja podatkom preveč prilagodijo, kar lahko pripelje do rešitev, ki ne odražajo dejanskega stanja sistema. To nakazuje, da je nepoznavanje prostora dopustnih rešitev kinetičnih parametrov še vedno pereč problem v sintezni in sistemski biologiji, ki ga je bilo potrebno nasloviti.

V našem primeru se je izkazalo, da sta prostora dopustnih rešitev modela represila-torja in modela biološke pomnilne celice D s predpomnenjem kompleksna, na kar nakazuje tudi hitrost konvergence preiskovanja s simuliranim ohlajanjem (glej sliki 4.4 in 4.5). Pri obeh modelih je z višanjem začetne temperature in velikosti izbire soseščine konvergenca preiskovanja naraščala. To nakazuje na prisotnost večjega števila lokalnih ekstremov, saj dobimo slabšo konvergenco, če izbiramo za naslednike vedno samo boljša stanja in se zato ujamemo v lokalni ekstrem. Višja konvergenca pri večjem koraku generiranja soseščine nakazuje na to, da je prostor dopustnih rešitev relativno majhen v primerjavi s celotnim prostorom. Z večjimi koraki tako hitreje dosežemo tudi druge regije dopustnih rešitev. To potrjuje, da sta prostora dopustnih rešitev kinetičnih parametrov modela represilatorja in biološke pomnilne celice D s predpomnenjem kompleksna z velikim številom lokalnih ekstremov.

Pri obeh modelih smo prostor rešitev razdelili v štiri razrede. V primeru represila-torja so ti razredi jasno definirani, se ne prekrivajo, so ohlapno povezani in ležijo na robu prostora rešitev. Pri modelu biološke pomnilne celice se prvi trije razredi drug dru-gemu nekoliko bolj prilegajo, četrti razred pa je od ostalih treh ločen. V tem primeru bi lahko izbrali tudi nižje število razredov za razvrščanje, ker pa smo se opirali na komolčno

metodo in statistiko vrzeli, smo izbrali štiri razrede. Da je takšno razbitje kvalitetno, je potrdila tudi silhueta posameznih razredov. Pri modelu biološke pomnilne celice D s predpomnenjem je prostor dopustnih rešitev nepovezan, kar dokazuje obstoj lokalnih ekstremov. To potrjuje tudi dejstvo, da imajo predstavniki razredov modela pomnilne celice D s predpomnenjem bistveno boljšo vrednost cenovne funkcije od povprečja razreda. To je moč razložiti tako, da se osebki pri preiskovanju z genetskim algoritmom porazdeljujejo v okolici lokalnih ekstremov.

7 Zaključek

V nalogi smo predlagali novo metodologijo za analizo prostora dopustnih rešitev v visokodimenzionalnih dinamičnih modelih bioloških sistemov in jo ovrednotili na modelih biološkega represilatorja ter biološke pomnilne celice D s predpomnenjem, ki smo ju z različnimi predpostavkami še dodatno poenostavili. Ta dva modela smo izbrali, saj izkazujeta oscilatorno dinamiko, ki je s stališča kompleksnosti analiz za nas zanimiva, hkrati pa so modeli s stališča računske zahtevnosti in števila parametrov še vedno obvladljivi. Primerjali smo enakomerno vzorčenje, vzorčenje s simuliranim ohlajanjem in vzorčenje z genetskimi algoritmi. Genetski algoritmi so se v našem primeru izkazali za najprimernejše, saj ti izkazujejo najhitrejšo konvergenco. Prostor dopustnih rešitev kinetičnih parametrov, ki smo ga omejili tako, da so najdene rešitve biološko relevantnejše, smo zato vzorčili z genetskim algoritmom. Pri obeh modelih smo za oceno kvalitete rešitev uporabili drugačno cenovno funkcijo. Pri modelu biološkega represilatorja smo v frekvenčnem prostoru sestavili takšno cenovno funkcijo, ki optimizira amplitudo in periodo oscilacij. Odzivi najdenih rešitev so tako izkazovali različno amplitudo in periodo oscilacij. Pri modelu biološke pomnilne celice D s predpomnenjem smo izbrali takšno cenovno

funkcijo, ki favorizira signale s točno določeno amplitudo in periodo oscilacij. Primerjali smo tudi različne algoritme za razvrščanje rešitev v posamezne razrede. Tako smo primerjali hierarhično razvrščanje in razvrščanje z voditelji. Zaradi prevelike časovne zahtevnosti hierarhičnega razvrščanja smo posamezne rešitve v različne razrede razvrstili z algoritmom razvrščanje z voditelji. Kvaliteto razbitja smo ovrednotili z metodo silhuete in posamezne razrede tudi analizirali ter identificirali najrobustnejši razred. Za oceno optimalnega števila razredov smo se poslužili komolčne metode in metode statistika vrzeli, slednja se je izkazala za primernejšo v primeru, da skupine v podatkih niso jasno definirane. Pri obeh modelih smo prostor dopustnih rešitev razdelili v štiri razrede. Najdene rešitve smo še projecirali na prve tri glavne komponente in analizirali različne preseke posameznih dimenzij modelov. Tako smo dobili občutek o velikosti, obliki in povezanosti regij, ki jih tvorijo rešitve kinetičnih parametrov. Izkazalo se je, da je prostor dopustnih rešitev pri obeh modelih kompleksen, pri modelu biološke pomnilne celice D s predpomnenjem pa je tudi nepovezan. Medtem ko so se rešitve modela represilatorja porazdelile po robu prostora, so se rešitve modela pomnilne celice D s predpomnenjem porazdelile okoli lokalnih ekstremov. Pri obeh modelih smo analizirali še vpliv posameznih kinetičnih parametrov na odziv sistema in to tudi vizualizirali z grafom raztrosa in toplotno karto. Za implementacijo smo se posluževali programskega jezika *Python* in ustreznih knjižnjic kot so $DEAP$, $NumPy$ ter $scikit-learn$.

Algoritmov za razvrščanje na podlagi gostote v nalogi nismo obravnavali, vseeno pa bi bilo zanimivo videti kakšne regije bi dobili z njihovo uporabo. Tipičen predstavnik algoritmov za razvrščanje na podlagi gostote je DBSCAN (ang. *Density Based Spatial Clustering of Applications with Noise*) [30]. V delu smo predlagano metodologijo ovrednotili na dveh modelih umetnih sistemov GRO. Zanimivo bi bilo videti tudi, kakšen prostor rešitev izkazujejo modeli, ki temeljijo na drugih motivih, kateri v biološkem organizmu izkazujejo oscilatorno dinamiko. Primer takšnih motivov so cirkadiani ritmi [17]. Dodatno bi lahko preverili tudi kako vpliva izbor različnih cenovnih funkcij na velikost in obliko prostora dopustnih rešitev.

Literatura

[1] D. Floreano, C. Mattiussi, Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies, MIT Press, 2008.

[2] U. Alon, An introduction to systems biology: design principles of biological circuits, CRC press, 2006.

[3] N. Le Novere, Quantitative and logic modelling of molecular and gene networks, Nature Reviews Genetics 16 (3) (2015) 146–158.

[4] M. Moškon, Modeli in metrike dinamike preklopa v enostavnih bioloških sistemih za potrebe računalniških struktur prihodnosti, doktorska disertacija, Univerza v Ljubljani (2012).

[5] H. El Samad, M. Khammash, L. Petzold, D. Gillespie, Stochastic modelling of gene regulatory networks, International Journal of Robust and Nonlinear Control 15 (2005) 691–711.

[6] M. Stražar, M. Mraz, N. Zimic, M. Moškon, An adaptive genetic algorithm for parameter estimation of biological oscillator models to achieve target quantitative system response, Natural Computing 13 (1) (2014) 119–127.

[7] Z. Zi, Sensitivity analysis approaches applied to systems biology models, IET systems biology 5 (6) (2011) 336–346.

[8] B. Taylor, T. J. Lee, J. S. Weitz, A guide to sensitivity analysis of quantitative models of gene expression dynamics, Methods 62 (1) (2013) 109–120.

[9] Ž. Pušnik, L. Magdevska, M. Mraz, N. Zimic, M. Moškon, Computational framework for global sensitivity analysis of high-dimensional and poorly connected parameter spaces, in: The IET/SynbiCITE Engineering Biology Conference, 2016.

[10] E. Borgonovo, E. Plischke, Sensitivity analysis: a review of recent advances, European Journal of Operational Research 248 (3) (2016) 869–887.

[11] J. W. Haefner, Modeling Biological Systems: Principles and Applications, Springer Science & Business Media, 2005.

[12] F. Crick, Central dogma of molecular biology, Nature 227 (5258) (1970) 561.

[13] L. Magdevska, Ž. Pušnik, M. Mraz, N. Zimic, M. Moškon, Computational design of synchronous sequential structures in biological systems, Journal of Computational Science, 18 (2017) 24–31.

[14] B. Orel, Osnove numerične matematike, Fakulteta za računalništvo in informatiko, 1997.

[15] L. Petzold, A. Hindmarsh, LSODA (Livermore solver of ordinary differential equations), Computing and Mathematics Research Division, Lawrence Livermore National Laboratory, Livermore, CA 24.

[16] B. Lemmer, The importance of circadian rhythms on drug response in hypertension and coronary heart disease—from mice and man, Pharmacology & therapeutics 111 (3) (2006) 629–651.

[17] O. Froy, The relationship between nutrition and circadian rhythms in mammals, Frontiers in neuroendocrinology 28 (2) (2007) 61–71.

[18] D. E. Cameron, C. J. Bashor, J. J. Collins, A brief history of synthetic biology, Nature Reviews Microbiology 12 (5) (2014) 381.

[19] Bionumbers, http://homepages.ulb.ac.be/~dgonze/BIONUMBERS/bionumbers.html, dostop: 7-7-2018.

[20] E. O. Brigham, R. Morrow, The fast Fourier transform, IEEE spectrum 4 (12) (1967) 63–70.

[21] C. E. Shannon, Communication in the presence of noise, Proceedings of the IRE 37 (1) (1949) 10–21.

[22] P. A. Ade, N. Aghanim, M. Arnaud, M. Ashdown, J. Aumont, C. Baccigalupi, A. Banday, R. Barreiro, J. Bartlett, N. Bartolo, Planck 2015 results: XIII. Cosmological parameters, Astronomy & Astrophysics 594 (2016) A13.

[23] D. Pham, D. Karaboga, Intelligent optimisation techniques: genetic algorithms, tabu search, simulated annealing and neural networks, Springer Science & Business Media, 2012.

[24] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Optimization by simulated annealing, science 220 (4598) (1983) 671–680.

[25] A. Shapiro, Monte Carlo sampling methods, Handbooks in operations research and management science 10 (2003) 353–425.

[26] P. Kosobutskyy, A. Kovalchuk, M. Kuzmynykh, M. Shvarts, Geometric calculation of Pi using the Monte Carlo method, in: Perspective Technologies and Methods in MEMS Design (MEMSTECH), 2016 XII International Conference on, IEEE, 2016, 167–169.

[27] W. K. Hastings, Monte Carlo sampling methods using Markov chains and their applications, Biometrika 57 (1) (1970) 97–109.

[28] F.-A. Fortin, F.-M. De Rainville, M.-A. Gardner, M. Parizeau, C. Gagné, DEAP: Evolutionary algorithms made easy, Journal of Machine Learning Research 13 (2012) 2171–2175.

[29] J. Han, J. Pei, M. Kamber, Data mining: concepts and techniques, Elsevier, 2011.

[30] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise., in: Int'l Conference on Knowledge Discovery in Databases and Data Mining (KDD), Vol. 96, 1996, 226–231.

[31] S. Kotsiantis, P. Pintelas, Recent advances in clustering: A brief survey, WSEAS Transactions on Information Science and Applications 1 (1) (2004) 73–81.

[32] R. Tibshirani, G. Walther, T. Hastie, Estimating the number of clusters in a data set via the gap statistic, Journal of the Royal Statistical Society: Series B (Statistical Methodology) 63 (2) (2001) 411–423.

[33] R. H. Riffenburgh, Linear discriminant analysis, doktorska disertacija, Virginia Polytechnic Institute (1957).

[34] H. Abdi, L. J. Williams, Principal component analysis, Wiley interdisciplinary reviews: computational statistics 2 (4) (2010) 433–459.

[35] C. Schillings, M. Sunnåker, J. Stelling, C. Schwab, Efficient characterization of parametric uncertainty of complex (bio) chemical networks, PLoS Comput Biol 11 (8) (2015) e1004457.

[36] E. Zamora-Sillero, M. Hafner, A. Ibig, J. Stelling, A. Wagner, Efficient characterization of high-dimensional parameter spaces for systems biology, BMC systems biology 5 (1) (2011) 142.

A Priloga

A.1 Izvorna koda

Izvorna koda z ustreznimi datotekami in navodili je dosegljiva na javnem GitHub repozitoriju: https://github.com/zigapusnik/analiza-dopustnih-resitev-bioloskih-modelov.