CHALMERS | GÖTEBORGS UNIVERSITET

Dynamisk modellering av Reg1-aktiverad feedback i Snf1-signalvägen i *Saccharomyces cerevisiae*

Dynamic modeling of Reg1-activated feedback in the Snf1-signaling pathway of *Saccharomyces cerevisiae*

Kandidatarbete inom civilingenjörsutbildningen vid Chalmers

Jakob Bruchhausen
Fredrik Lorentzon
Johan Olson
Lucas von Brömsen

Institutionen för Matematiska vetenskaper
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Göteborg, Sverige 2020

Dynamisk modellering av Reg1-aktiverad feedback i Snf1-signalvägen i *Saccharomyces cerevisiae*

Fredrik Lorentzon Jakob Bruchhausen
Johan Olson Lucas von Brömsen

Kandidatarbete i matematik inom civilingenjörsprogrammet Bioteknik vid Chalmers

Fredrik Lorentzon Jakob Bruchhausen Johan Olson Lucas von Brömsen

Handledare: Marija Cvijovic
Johannes Borgqvist
Sebastian Persson

Institutionen för Matematiska vetenskaper
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Göteborg, Sverige 2020

Förord

Detta kandidatprojekt har utförts som ett grupparbete av projektets fyra medlemmar, som tätt samarbetat under projektets alla moment. Stöd och vägledning har getts av handledare, Chalmers Bibliotek och Cvijovic Lab.

Inom projektgruppen har Lucas von Brömsen varit ansvarig för förståelse av biologiska system, för utveckling av ODE-modeller samt för analys av modeller med olika metoder (såsom strukturell identifierbarhetsanalys och stabilitetsanalys). Johan Olson har varit ansvarig för metoder för lösning av ODE-system samt för analys av modeller (såsom känslighetsanalys). Jakob Bruchhausen har varit ansvarig för kontinuerlig optimering med Quasi-Newton-metoden samt för databearbetning. Fredrik Lorentzon har varit ansvarig för begränsningar inom den kontinuerliga optimeringen. Programmeringen har utförts av Johan Olson, Jakob Bruchhausen och Fredrik Lorentzon. Samtliga gruppmedlemmar har varit involverade i skrivandet och utformningen av denna rapport.

Under projektets gång har dagbok och individuella tidsloggar förts. Dagboken har bestått av beskrivningar av vad gruppen gemensamt gjort under varje dag. I de individuella tidsloggarna har varje gruppmedlem registrerat den tid personen lagt på projektet, och skrivit en kort beskrivning av vad som gjorts under denna tid.

Vi som skrivit denna rapport vill tacka våra handledare Marija Cvijovic, Johannes Borgqvist och Sebastian Persson för vägledning, tips och inspiration. Vi vill också tacka Cvijovic Lab för intressanta möten, Barbara Schnitzer för värdefulla råd, Chalmers Bibliotek för hjälp med rapportskrivande och Chalmers Matematiska Vetenskaper för administration av kandidatarbetet. Den data som detta projekt baseras på är framtagen av Niek Welkenhuysen, postdoktor vid institutionen för biologi och bioteknik samt institutionen för matematiska vetenskaper på Chalmers Tekniska Högskola.

Populärvetenskaplig presentation

Syftet var att lägga en pusselbit i pusslet om människans åldrande. Detta skulle göras genom en matematisk modell, något som visade sig vara svårare än förväntat.

Matematik är ett ämne som ofta för tankarna till beräkningar med penna och papper. Att lösa ut en okänd variabel som tämligen ofta har fått beteckningen x. Något som många finner oanvändbart och har fått dem att tappa intresset för detta egentligen användbara område. Även om matematik i vissa sammanhang kan kännas tråkig och meningslös så är det essentiellt för att vårt moderna samhälle ska fungera, samt utvecklas. Den utveckling som matematiken däremot driver handlar allt oftast inte om att bara lösa ut en okänd variabel x. Det finns en enorm uppsjö av tillämpningar på matematik som kan lösa en minst lika stor uppsjö av problem. Ett sådant problem är den stora frågan kring hur vi åldras.

Ett område som kanske kan besvara denna fråga är matematik tillämpad på de biologiska reaktioner som sker i vår kropp och dess celler. Det område som använder matematik och datorer för att beskriva dessa reaktioner kallas systembiologi. Baserat på ett förslag kring hur en reaktion sker, kan sedan en matematisk modell skapas. En matematisk modell är en beskrivning av ett system med hjälp av matematiska koncept, gjord för att förstå systemets funktion [1]. Modeller är endast ett försök att beskriva verkligheten, även om dessa modeller kan ge större insikt i hur reaktioner sker. Ett bra citat som förklarar detta är ”alla modeller är fel, men vissa kan vara användbara” ([2], s.791). Dessa modeller kan alltså ge oss användbar information kring cellerna, information som kan användas för att exempelvis bota sjukdomar [3]. Detta är det slutgiltiga resultatet som forskningen i systembiologi kan hjälpa till att lösa, men för att nå ditt krävs större förståelse för de reaktioner som är involverade i åldrande.

En viktig del i åldrandet är hur en cell hanterar förhållanden då det finns begränsad näring att tillgå, alltså när tillgången på mat är dålig. Att äta dålig mat eller för lite mat, är nog något som de flesta håller med om har stor påverkan på vårt välbefinnande och därmed vårt åldrande. Just denna studie hanterar en signalväg som är involverad då tillgången på en cells föredragna mat; glukos, ej finns tillgänglig. Med hjälp av systembiologin och dess matematiska modeller undersöks denna signalvägs uppbyggnad.

I människors celler styrs upptaget av näring från omgivningen av en så kallad signalväg, där AMPK-signalvägen (AMP-aktiverat proteinkinas) är en. Denna signalväg reglerar funktioner i cellen baserat på tillgången av glukos [4]. Vid en hög halt av glukos är denna signalväg inte speciellt aktiv och cellen kan mumsa i sig sin favoritmat utan samvetskval. När favoritmaten däremot finns i dålig tillgång så behöver cellen istället äta andra näringsämnen. Låg glukoshalt aktiverar genom signalvägen en gen kallad *SUC2*, som bildar de proteiner som tillåter intagandet och nedbrytningen av alternativa näringsämnen, såsom sukros.

Då experiment på människoceller är både kostsamt och tidskrävande är det mer praktiskt att istället använda jästceller. Jäst är en bra organism eftersom det finns många likheter mellan jäst- och människoceller, bland annat att båda är flercelliga, eukaryota organismer [5]. Detta gör att experiment kan göras med jästceller istället för människoceller och ge samma svar, men till betydligt lägre kostnad. Varför just jästceller är att föredra över andra eukaryota organismer förklaras dels i den lägre kostnaden, men även i dess fördelar när det kommer till etiska aspekter. Att utsätta djur för experiment är något många är emot då det ofta är grymt och dessutom ofta leder till en hemsk död. Endast om vinningen som dessa experiment ger är stor, kanske folk kan acceptera genomförandet. Där finns en stor fördel med jästcellerna, dessa små organismer finns i närmast oräknelig mängd. Därtill används de redan i en överväldigande mängd applikationer, exempelvis tillverkning av de många drycker som återfinns på Systembolaget. Ingen har tidigare ifrågasatt användandet av jäst på grund av något djurplågeri, liksom ingen anser att jästceller har någon uppfattning av lidande. Därför är det en etiskt mycket försvarbar organism att göra sina experiment på.

Även om de experiment som genomförs är försvarbara i den aspekten så måste frågan ställas

om det överhuvudtaget finns en vinning med experimenten. I detta fall ger större förståelse kring Snf1-signalvägen en essentiell pusselbit, som behövs för att lösa frågan kring hur åldrandet sker. Dock är större förståelse kring åldrande inte en lösning på något problem i sig, men det är behövligt för att lösa problem kopplade till åldrande. Åldrande är oundvikligt och att förstå effekterna och orsakerna bakom det kan potentiellt utnyttjas för att bota de många sjukdomar som är kopplade till åldrande i människor. Bland dessa sjukdomar finns Alzheimers och Parkinsons, som oftast uppkommer bland äldre personer. Att förstå hur åldrandet fungerar på en cellskala kan ge insikter i hur dessa sjukdomar fungerar. Med mer kunskap och förståelse kan sedan eventuella behandlingar för åldersrelaterade sjukdomar utvecklas, varvid människan skulle kunna leva längre och friskare [6].

Frågeställningen, alltså projektets syfte, kan för den oinvigde framstå något svårförstådd. Att med systembiologins hjälp ta reda på ifall det som bildas i en reaktion senare stänger av samma reaktion. Detta kallas för en feedback. Det finns anledningar att tro en sådan feedback sker i signalvägen. Den stora frågan man vill besvara är om feedbacken sker via ett protein kallat Reg1. Med hjälp av en matematisk modell som innehåller detta protein kan förhoppningsvis frågeställningen besvaras.

När det finns en modell är tanken att få denna att beskriva verkligheten så bra som möjligt. Verkligheten i detta fall är observationer hos jäst, som tagits fram av forskare i ett laboratorium. Eftersom att modellen anpassas till dessa observationer med en algoritm, blir den även användbar för andra syften. Den kan då användas till att förutspå resultaten från andra experiment.

Tyvärr hade alla modellerna i detta projekt fel, vilket gjorde det svårt att dra några tydliga slutsatser från dem. I den första modellen var problemet väldigt rättframt; modellen lyckades helt enkelt inte att återskapa de biologiska observationerna. I den andra modellen var problemet mer subtilt. Här lyckades visserligen de biologiska observationerna återskapas, men istället fanns det problem med själva modellen. Dessa problem, som var av ren matematisk karaktär, gjorde modellen instabil och opålitlig, och var därför inte lämplig för att göra förutsägningar.

Trots att studier på en signalväg i jäst i sig inte kan ses som väldigt intressant för de flesta, så är effekterna som kommer ut däremot av stor betydelse och intresse. Att bota sjukdomar är något som har stor positiv inverkan i samhället, varvid dessa studier blir intressanta. Studier som inte hade varit genomförbara utan matematik och dess tillämpning i systembiologi.

Sammanfattning

Snf1-signalvägen har en central roll i näringsupptag och användandet av alternativa kolkällor till glukos i jästen *Saccharomyces cerevisiae*. I denna signalväg har en potentiell feedback-loop observerats hos uttrycket av *SUC2*, en gen som är fundamental för utnyttjandet av alternativa kolkällor. Mekanismen bakom denna feedback-loop är okänd. En hypotes är att aktivering av Reg1-proteinet i sin tur påverkar uttrycket hos *SUC2*.

För att undersöka detta användes dynamisk modellering där matematiska modeller försökte beskriva trenden hos datan genererades. Dessa modeller bestod av ordinära differentialekvationer grundade i massverkans lag. Stabilitetsanalys och strukturell identifierbarhetsanalys genomfördes för att få information om modellernas egenskaper. Genom kontinuerlig optimering med främst *Quasi-Newton-metoden* passades modellerna till datan. Detta gjordes med minimering av en kostnadsfunktion. Känslighetsanalys utfördes på optimerade modeller för att undersöka de ingående parametrarna och deras rimlighet.

Den första modellen som presenteras i rapporten kunde inte fånga trenden hos datan och hade heller inte önskvärda egenskaper såsom stabilitet och bra prediktionsförmåga. Den alternativa modellen som introduceras kunde fånga trenden hos datan på ett tillfredsställande sätt, men saknade också de egenskaper som är väsentliga för en kraftfull modell.

Ingen av de presenterade modellerna lyckades klargöra eller förtydliga mekanismen bakom feedback-systemet. Fler modellstrukturer måste undersökas och fler experiment utvecklas och genomföras för att kunna testa nya modeller och därmed ge större insikt kring mekanismerna hos denna signalväg.

Abstract

The Snf1 pathway has a vital role in nutrient uptake and the utilization of alternative carbon sources other than glucose in the yeast *S. cerevisiae*. In this pathway a potential feed-back loop has been observed in the expression of *SUC2*, a gene essential for the utilization of alternative carbon sources. One hypothesis is that this potential feed-back loop, whose feed-back mechanism still remains unknown, is caused by the activation of the Reg1 protein which in turn affects the *SUC2* expression.

Dynamic modeling was applied to examine this and mathematical models which attempted to describe the data were created. These models were constructed with ordinary differential equations based on the mass-action law. Stability analysis and structural identifiability analysis were performed to obtain information about the properties of the models. Through continuous optimization with mainly the *Quasi-Newton method* the models were to fit the data. This was done through minimization of a cost function. Sensitivity analysis was carried out on optimized models to examine their viability.

The first model shown in the report could not characterize the general trend observed in the data or possessed the desirable characteristics of a rigorous model such as stability and a strong predicative ability. The second model introduced could be fit to the data to a strong and adequate degree, but also lacked preferable properties crucial for a robust model.

Neither of the presented models succeeded in elucidating the mechanism in the feedback system. Additional models must be studied and experiments must be developed and enacted to a greater extent to test new models and provide further understanding of this pathway.

Innehåll

1 Inledning

Cancer är den näst vanligaste dödsorsaken i världen (2018) och ett av sex dödsfall globalt kan relateras till cancer [7]. Forskning visar att allvarliga sjukdomar såsom diabetes och cancer är kopplade till ålder och blir vanligare ju äldre en människa blir [8].

Människans åldrande är en långsam process vilket gör den svår att studera. Jäst däremot har en betydligt kortare livslängd vilket gör det lättare att studera åldrande av jästceller. Jäst och människor är på en cellulär nivå väldigt lika då de båda är eukaryota organismer. Därav är många funktioner och processer konserverade mellan människor och jäst, vilket innebär att mekanismerna liknar varandra. Ett bra exempel på detta är den biologiska signalvägen Snf1 i jästen *Saccharomyces cerevisiae* (*S.cerevisiae*) som liknar AMPK-signalvägen i mänskliga celler [9].

AMPK är en viktig faktor när det kommer till att lindra högt blodsocker, höga fettvärden i blodet, bryta ned fettsyror samt förbättra insulinkänslighet. Detta innebär att dess funktion och aktivitet är starkt kopplat till diabetes typ 2 samt riskfaktorer för hjärt- och kärlsjukdomar. Forskning har även visat att AMPK är starkt involverad i att kontrollera en gen som reglerar celldelning, vilket därmed även förhindrar tumörbildning. En förändring i aktiviteten kan således leda till okontrollerad celldelning och tumörbildning. AMPK utgör till följd av detta en tydlig länk mellan hjärt- och kärlsjukdomar och cancer [10]. Studier på signalvägen har visat att förutom att vara en faktor när det kommer till många allvarliga sjukdomar så minskar AMPK-aktiviteten med åldern [11]. Det är därför intressant att studera AMPK-signalvägen närmare i samband med åldrande. Detta kan göras genom att studera Snf1, dess ortolog i jäst, närmare.

I jäst så reglerar Snf1-signalvägen många olika gener, en av dem är *SUC2*. Under glukossvält, alltså i glukosfattiga miljöer så ökar uttrycket av *SUC2* [12]. Nyligen insamlad data visar att detta ökade uttryck vid en drastisk sänkning av glukoskoncentrationen inte är permanent, utan följs av en nedgång av *SUC2*-uttryck. Detta ger indikation på att det finns en feedback-mekanism i Snf1-signalvägen som, under långa perioder av glukossvält, sänker uttrycket av *SUC2*. *SUC2* verkar således regleras av både tillgången på glukos men även av en annan faktor.

Ett sätt att studera signalvägar i jäst är genom ett systembiologiskt tillvägagångsätt [13]. För att studera Snf1 kan en dynamisk modell skapas som beskriver signalvägen och uttrycket av *SUC2*. Denna modell kan sedan passas till tillgänglig data, som den som nämnts tidigare. Optimeringsprocessen kan således generera en modell som beskriver verkliga experiment och kan förutspå andra experiment.

En teori till den observerade feedback-loopen är att minskningen av *SUC2* beror på att *SUC2* inhiberar den aktiva formen av Snf1. En möjlighet kan istället vara att koncentrationen av Reg1 ökar, vilket därmed kan leda till att Snf1 inaktiveras och *SUC2*-expressionen minskar. Detta förklarar den observerade feedback-loopen. För att förstå denna dynamik ska en dynamisk modell tas fram som på ett bra sätt beskriver denna del av Snf1-signalvägen.

2 Syfte

Syftet med detta projekt är att utveckla en kinetisk, dynamisk ODE-modell som beskriver uttrycket av genen *SUC2* i Snf1-signalvägen i *Saccharomyces cerevisiae*. Snf1-signalvägen aktiveras då cellen utsätts för glukossvält, och då ökar koncentrationen av *SUC2*. Efter en tid av glukossvält minskar dock *SUC2*-koncentrationen. Detta projekt ska undersöka huruvida denna minskning beror på ett feedback-system som regleras av proteinet Reg1.

ODE-modellens parametrar ska skattas för att optimalt efterlikna biologisk mätdata i form av en tidsserie. Denna mätdata beskriver hur aktiviteten av genen *SUC2* förändras då celler hamnar i en glukosfattiga miljö. Parameterskattningen ska ske genom kontinuerlig optimering med *Quasi-Newton-metoden*, genom en algoritm som byggs i projektet. Modellens pålitlighet ska undersökas genom att genomföra känslighetsanalys, stabilitetsanalys, samt identifierbarhetsanalys.

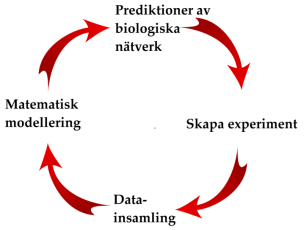
3 Teori

Följande kapitel innefattar de delar som är essentiella för att förstå processen som dynamisk modellering utgör inom systembiologin. Vidare förklaras de delar som är specifika för detta projekt mer ingående, exempelvis Snf1-signalvägen och verktyg som kan appliceras för den framtagna modellen. Ämnen som kommer att behandlas i detta avsnitt är grunderna inom systembiologi, Snf1-signalvägen, dynamisk modellering, parameteruppskattning samt känslighets-, stabilitets- och identifierbarhetsanalys.

I denna rapport betecknas vektorer och matriser med respektive små och stora bokstäver i fetstil. Gradienten betecknas med ∇. Approximerade variabler betecknas med $\sim$ och skattade värden med $\hat{}$.

3.1 Grunderna inom systembiologin

Dynamiken hos biologiska signalvägar är ofta för komplex för att kunna förstås enbart via ett experimentellt tillvägagångsätt. Bland annat hur ingående komponenter samverkar med andra system samt hur mekanismerna fungerar, gör det komplicerat att beskriva systemet. För att förstå detta komplexa system behöver modellering och experimentell data därför kombineras.

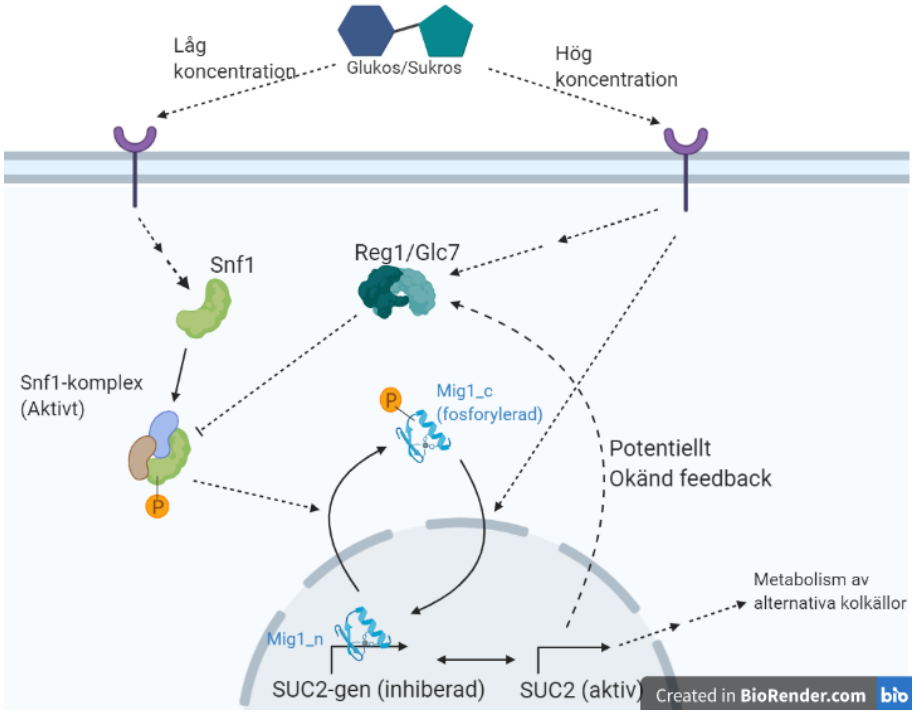


Figur 1: *Den konventionella systembiologiska cykeln [14].*

Systembiologins tillvägagångssätt är en cyklisk process som börjar med att experiment utförs på det biologiska systemet. Detta genererar data och utifrån denna kan man ställa upp matematiska modeller av systemet. Ifall den matematiska modellen beskriver datans beteende kan denna modell användas till prediktion av det biologiska systemet. Utifrån denna prediktion kan ytterligare experiment utföras varvid ny data inhämtas på vilken ny modellering kan göras. Därmed når cykeln (fig. 1) sitt slut och kan påbörjas igen. Denna process kan itereras för att förbättra och förfina modellen tills man har en modell kraftfull nog att beskriva det betraktade systemet [15].

3.2 Snf1-signalvägen

Snf1-signalvägen är en signalväg hos *S. cerevisae* som är involverat i näringsupptag hos cellen (fig. 2). Det centrala Snf1-proteinet ingår i ett proteinkomplex [12]. Komplexet har visat sig vara en del i regleringen av många olika gener, men den gen som är av intresse i detta projekt är *SUC2* som regleras av glukoskoncentrationen i cellen. I glukosrika miljöer har Snf1 observerats vara ofosorylerad och därmed inaktiv. I dessa miljöer är proteinet Reg1 essentiellt för defosforylering och inaktivering av Snf1 [16]. När det istället finns lite glukos i cellen fosforyleras Snf1 och aktiveras. Konsekvensen av detta är att Mig1-proteinet som inhiberar *SUC2*-genen lämnar cellkärnan i en fosforylerad form. Detta gör att *SUC2*-genen blir aktiv vilket medför att cellen kan utnyttja andra kolkällor än glukos, framför allt sackaros [17].



Figur 2: *De delar av Snf1-signalvägen som har studerats. Heldragna pilar beskriver en förändring/reaktion av ett ämne till ett annat medan streckade linjer beskriver en påverkan på en reaktion.*

3.3 Dynamiska ODE-modeller inom systembiologin

Inom systembiologi används ofta system av ordinära differentialekvationer (ODE:er) för att beskriva reaktions- och signalvägar i organismer. Detta för att reaktionerna som utgör uppbyggnad och sönderfall av komponenter i det studerade systemet ofta är kända. Differentialekvationerna kan byggas upp genom massverkans lag, det vill säga antagandet att ett ämnes bildnings- och förbrukningshastighet är direkt proportionellt mot ämneskoncentrationenerna av de andra reaktanterna [18]. Proportionaliteten beskrivs genom en hastighetskonstant k_i. Ekvation (1) beskriver en simpel kemisk reaktion med reaktanterna x_1 och x_2 vars reaktionshastigheter beskrivs med hjälp av hastighetskonstanterna k_1 och k_2.

$$x_1 \underset{k_2}{\overset{k_1}{\rightleftharpoons}} x_2 \tag{1}$$

Bakåt- och framåtreaktionen i (1) kan beskrivas som ett system av två linjära ODE:er där de två ekvationerna definieras med hjälp av massverkans lag , ekv. (2). Samma teori kan appliceras på ett mer komplicerat system med fler än en reaktion varpå ett större system ekvationer erhålls.

$$\begin{cases} \frac{\mathrm{d}x_1}{\mathrm{d}t} = - & k_1 x_1 + k_2 x_2 \\ \frac{\mathrm{d}x_2}{\mathrm{d}t} = & k_1 x_1 - k_2 x_2 \end{cases} \tag{2}$$

Ju fler reaktanter och reaktionsvägar som inkluderas i modellen, desto fler ekvationer kommer ODE-systemet att innehålla. Modellen kan även kompletteras med termer för enzymatisk inhibering, vilket kan resultera i icke-linjära termer.

3.4 Parameteruppskattning

Vid konstruktion av en dynamisk ODE-modell erhålls ett antal okända hastighetskonstanter, $k_i \in \mathbf{k}$, som viktar funktionerna i modellen mot varandra. För att modellen ska kunna användas krävs att dessa parametrar uppskattas till värden som väl beskriver signalvägen i verkligheten.

Detta kan göras genom att passa modellen till experimentell data med en kostnadsfunktion. Kostnadsfunktionen är en funktion som beskriver hur bra en modell passar den givna datan. Kostnadsfunktionen definieras ofta som en *minsta kvadrat-funktion* [19], $MK(\hat{\mathbf{k}})$, vars ekvation ges av ekv. (3). $MK(\hat{\mathbf{k}})$ är en summa som beskriver det kvadratiska felet mellan modell och data, även kallat RSS (residual sum of squares) [20].

$$\mathcal{L}(\hat{\mathbf{k}}) \propto MK(\hat{\mathbf{k}}) = \sum_{j=1}^{n} \sum_{k=1}^{l} \left(y_j(t_k) - \hat{y}_j(t_k, \hat{\mathbf{k}}) \right)^2 \tag{3}$$

I ekvation (3) är y funktionsvärden, j anger index av n funktioner och k anger index för tidsserie med l datapunkter. Genom att minimera ekv. (3) så kommer modellen att passas till datan på ett sådant sätt att residualfelet blir så litet som möjligt. $MK(\hat{\mathbf{k}})$ har sitt ursprung i att felet enligt centrala gränsvärdessatsen kan antas vara normalfördelat [21]. Då kan funktionen härledas ur den negativa logaritmen av likelihoodfunktionen $\mathcal{L}(\hat{\mathbf{k}})$. I avsnitt 3.4.5 beskrivs hur $MK(\hat{\mathbf{k}})$ minimeras genom en nedåtstegande algoritm.

3.4.1 Lösning av modell

De ODE:er som utgör en modell behöver lösas för att beräkna $MK(\hat{\mathbf{k}})$. System av ODE:er kan inneha olika egenskaper varpå olika sätt att lösa dem blir mer eller mindre effektiva. Dynamiska system kan kategoriseras som antingen styva eller icke-styva. För ett styvt system är det fördelaktigt att använda en flerstegsmetod (*multistep method*). För lösning av ett icke-styvt problem fungerar det bättre med en *Runge-Kutta-metod*. Båda lösarna är så kallade adaptiva lösare, vilket innebär att steglängden anpassas med hjälp av det trunkerade felet på den tidigare lösningen i algoritmen [22].

3.4.2 Kontinuerlig optimering och val av nedåtstegande riktning

Modellen kan passas till datan, det vill säga kostnadsfunktionen $MK(\hat{\mathbf{k}})$, ekv. (3), minimeras genom kontinuerlig optimering. Kontinuerlig optimering grundar sig i att minimera eller maximera en funktion genom en iterativ sökalgoritm där varje iteration genererar en nedåtstegande riktning i parameterrummet $\mathbb{R}^p$, där p är antalet parametrar [20]. Här kan *Quasi-Newton-metoden* (QNM) användas för att beräkna den nedåtgående riktningen för algoritmen. Ett steg i denna riktning kommer att generera en ny uppsättning parametrar som i sin tur ger ett lägre funktionsvärde.

Två olika metoder för att från punkten $\hat{\mathbf{k}}$ beräkna en nedåtgående stegriktning $\mathbf{p}$ är ovan nämnda QNM, ekv. (4), och *steepest descent-metoden*, ekv. (5). QNM är mer komplex då den tar hänsyn till funktionens kurvatur runt punkten $\hat{\mathbf{k}}$, vilket möjliggör mer effektiva steg. Efter beräkning av $\mathbf{p}$ normeras den till en enhetsvektor $\frac{\mathbf{p}}{\|\mathbf{p}\|}$.

$$\widetilde{\mathbf{H}}(\hat{\mathbf{k}}) \cdot \mathbf{p} = -\nabla MK(\hat{\mathbf{k}}) \tag{4}$$

$$\mathbf{p} = -\nabla MK(\hat{\mathbf{k}}) \tag{5}$$

3.4.3 Approximation av gradient och hessianmatris för att hitta stegriktning

En nödvändig del i QNM är att hessianmatrisen tas fram, vilket effektivast görs genom en approximation. $\widetilde{\mathbf{H}}(\hat{\mathbf{k}})$ är en sådan approximation av hessianmatrisen, som beräknas enligt ekv. (6)

$$\widetilde{\mathbf{H}}(\hat{\mathbf{k}}) = \frac{1}{s^2} \left(\nabla MK(\hat{\mathbf{k}}) \cdot \nabla MK(\hat{\mathbf{k}})^T \right) \qquad . \tag{6}$$

Där s är variansen av den biologiska datan. Gradienten $\nabla MK(\hat{\mathbf{k}})$ beräknas med finita differenser enligt ekv. (7)

$$\nabla MK(\hat{\mathbf{k}}) = \frac{MK(\hat{\mathbf{k}} + \delta) - MK(\hat{\mathbf{k}} - \delta)}{2\delta} \quad . \tag{7}$$

δ är steglängden. I detta projekt användes $\delta = \sqrt{eps}$ där *eps* är maskintoleransen, eftersom denna steglängd ger en god gradientapproximation i alla parameterriktningar [23].

Från punkten $\hat{\mathbf{k}}$ med den nedåtgående stegriktningen $\mathbf{p}$ beräknas nästa punkt enligt ekv. 8

$$\hat{\mathbf{k}}_{i+1} = \hat{\mathbf{k}}_i + \alpha \frac{\mathbf{p}_i}{\|\mathbf{p}_i\|} \quad . \tag{8}$$

Ett krav för att QNM med säkerhet ska generera en nedåtgående stegriktning är att $\widetilde{\mathbf{H}}(\hat{\mathbf{k}})$ är positivt definit i punkten $\hat{\mathbf{k}}$, alltså att funktionen $MK(\hat{\mathbf{k}})$ är strikt konvex i $\hat{\mathbf{k}}$ [24]. Detta testas genom att egenvärdena $\boldsymbol{\lambda}$ till $\widetilde{\mathbf{H}}(\hat{\mathbf{k}})$ undersöks. Då $\widetilde{\mathbf{H}}(\hat{\mathbf{k}})$ är symmetriskt innebär ett positivt egenvärde $\lambda_i > 0$ att funktionen är konvex i den aktuella parameterriktningen. Om alla egenvärden i $\boldsymbol{\lambda}$ är positiva är $\widetilde{\mathbf{H}}(\hat{\mathbf{k}})$ positivt definit.

3.4.4 Fördelar med QNM

QNM använder sig av en approximation av en funktions andraderivator till skillnad från steepest descent som endast använder sig av gradienten $\nabla MK(\hat{\mathbf{k}})$. QNM kan utläsa information om en extrempunkts egenskaper och är således mindre benägen att ”fastna” i ett lokalt minimum eller en sadelpunkt. Vid användning av steepest descent-metoden ökar då risken att en optimeringsalgoritm avstannar i ett lokalt extremvärde och processen således termineras.

När QNM inte kan generera en säker riktning, det vill säga när $\lambda_i < 0$ (avsnitt 3.4.3), eller när stegriktningen inte kan beräknas (när $\widetilde{\mathbf{H}}(\hat{\mathbf{k}})$ ej är inverterbar) måste en annan metod användas. Istället för QNM , ekv. (4), kan stegriktningen då det är nödvändigt beräknas med steepest descent-metoden, ekv. (5).

3.4.5 Beräkning av steglängd

En metod för att beräkna steglängden α är *Armijos regel*, även kallad *backtracking line search* [24]. Det är en iterativ algoritm som går ut på att först testa ett relativt stort värde på α och sedan successivt minska det. En variant av Armijos regel beskrivs i ekv. (9)

$$\alpha_k = \left(\frac{1}{2}\right)^k, \qquad k = 0, 1, 2, \ldots, k_{max} \quad . \tag{9}$$

I den första itereringen av *Armijos regel* sätts $k = 0$ och $MK(\hat{\mathbf{k}})$ beräknas med denna steglängd, alltså $MK\left(\hat{\mathbf{k}}_i + \alpha_0 \frac{\mathbf{p}_i}{\|\mathbf{p}_i\|}\right)$. Om villkoret i ekv. (10) uppfylls avslutas itereringen

$$MK\left(\hat{\mathbf{k}}_i + \alpha_k \frac{\mathbf{p}_i}{\|\mathbf{p}_i\|}\right) < MK\left(\hat{\mathbf{k}}_i\right) \quad . \tag{10}$$

Om villkoret i ekv. (10) inte uppfylls ökar k med 1 och villkoret testas igen. Detta upprepas till villkoret uppfylls. Om villkoret inte är uppfyllt när $k = k_{max}$ sätts $\alpha = \alpha_{k_{max}}$.

3.4.6 Latin hypercube sampling (LHS)

Eftersom modellen inte är konvex finns det flera lokala minimum. Om endast en stargissning skapas är risken stor att optimeringen terminerar i ett lokalt minimum, så för att hitta det globala minimumet behöver flera startgissningar prövas. LHS är en statistisk metod för att generera startgissningar jämt distribuerade inom ett n-dimensionellt parameterrum. I metoden definieras en n-dimensionell kub (en hyperkub) och N antal startgissningar genereras inuti denna hyperkub. n och N är positiva heltal. LHS genererar startgissningarna så att ett visst värde på en parameter endast uppträder en gång i hyperkuben [25]. I två dimensioner är detta ekvivalent med att rita ett rutnät i en kvadrat, och låta varje rad och kolumn av rutnätet innehålla exakt en startgissning. En

fördel med LHS jämfört med helt slumpmässig generering av startgissningar är att LHS minskar risken att två startgissningar genereras nära varandra. En annan fördel är att LHS ökar sannolikheten för att startgissningarna fördelas mer jämt inom hyperkuben [25]. Ekvation (11) visar att halveringstiden $t_{1/2}$ för en första ordningens parameter är inversproportionerlig mot ln(2), varför hyperkubens sidlängd bör vara proportionell mot ln(2).

$$\frac{\mathrm{d}x}{\mathrm{d}t} = -k_1 x \quad \Rightarrow \quad \int_{2x}^{x} \mathrm{d}x = -\int_{0}^{t_{1/2}} k_1 \mathrm{d}t \quad \Rightarrow \quad k_1 = \frac{1}{t_{1/2}} \ln(2) \tag{11}$$

3.5 Känslighetsanalys

Syftet med en känslighetsanalys är att undersöka hur en förändring av en parameter i en modell påverkar dess lösning. Detta görs genom användning av finita differenser. Detta har stor likhet i hur gradienten beräknas, då sensitiviteten är gradienten av lösningen för en av variablerna i en given tidpunkt. Genom att beräkna sensitiviterna för alla variabler i ett antal tidpunkter kan känslighetsmatrisen definieras. Utifrån denna känslighetsmatris kan sedan en summering av varje parameters inverkan på lösningen av varje variabel göras. Detta värde normeras med antalet tidpunkter och sensitiviteter till ett RMS-värde (*Root-Mean-Square*). Ett lågt parametervärde relativt de andra parametrarna kan då avslöja att en parameter har liten betydelse för lösningen. Denna parameter kan därefter elimineras och således förenkla modellen [26].

3.6 Stabilitetsanalys

För att testa rimlighet och prediktionsförmåga hos biologiska modeller kan stabilitetsanalays användas som verktyg. Biologiska system uppvisar generellt en homeostatisk egenskap och drivkraft att nå ett steady-state. Därmed kan man utvärdera stabiliteten hos biologiska modeller för att testa trovärdigheten hos modellen, men även använda stabilitetsanalys som ett verktyg för att undersöka modellen och eventuella begränsningar i modellen [27]. Således kan man sätta avgränsningar, ändra modellen och förbättra den för att ge modellen ännu bättre prediktionsförmåga, vilket gör stabilitetsanalys till ett kraftfullt verktyg inom matematisk modellering [28].

Matematiskt görs detta genom att identifiera alla steady-state punkter i modellen. Därefter tas Jacobianen till modellen fram, och Jacobianen utvärderas i de framtagna steady-state-punkterna. Slutligen tas den karakteristiska ekvationen fram, varpå observation av egenvärdenas tecken avgör om modellen är stabil. Är realdelen av alla egenvärden negativa är modellen stabil, är realdelen av minst ett egenvärde positivt kommer modellen att vara instabil [28].

Vidare information om detta och ett exempel på genomförd stabilitetsanalys kan ses i appendix A.

3.7 Identifierbarhetsanalys

Identifierbarhet är ett viktigt begrepp inom matematisk modellering eftersom det ger information om de ingående parametrarna i systemet av ODE:er. Parametrar kan vara oidentifierbara vilket innebär att parametrarna inte kan skattas med god precision. Identifierbarhetsanalys ämnar att bestämma huruvida parametrarna i modellen är identifierbara [29].

Identifierbarhetsanalys delas vanligtvis in i två olika områden; strukturell identifierbarhet och numerisk identifierbarhet. Strukturell identifierbarhet är en *a priori*-metod och indikerar om parametrarna kan bestämmas unikt enbart från själva modellstrukturen [29]. Analys för strukturell identifierbarhet genomfördes med programmeringsspråket Wolfram Mathematica version 12.0 och paketet `IdentifiabilityAnalysis` [30]. Exempel på strukturell identifierbarhetsanalys kan ses i Appendix B.

Numerisk identifierbarhet beror istället på de numeriska erhållna parametervärdena från modellen och är följaktligen en *a posteriori*-metod. Är en modell numeriskt identifierbar är den även strukturellt identifierbar, men vice versa gäller ej. Den numeriska identifierbarheten hänger

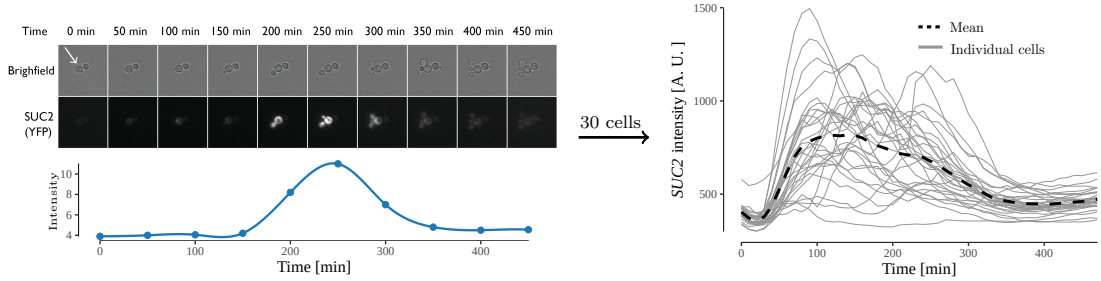
även ihop mycket med känslighetsanalysen då hur känslig en parameter är för en liten ändring också kan tolkas som vilken precision den kan uppskattas med [29]. Ingen numerisk identifierbarhetsanalys genomförs i detta arbete, men en känslighetsanalys genomförs vilket kan beskriva känsligheten hos parametrarna och därmed säga något om den numeriska identifierbarheten hos parametrarna.

4 Metod

Arbetet i detta projekt har varit uppdelat i tre olika delar där varje del utfördes under olika tidpunkter.

4.1 Biologisk data

Datan som detta projekt bygger på kan ses i fig. 3. Datan i denna figur genererades ifrån bildanalys av mikroskopibilder på jästceller, där bilder togs var tionde minut. Från bilderna mättes ljusstyrka från fluorescenta proteiner i jästen, där ljusstyrkan är proportionell mot *SUC2*-aktivitet i cellen. Även jästcellernas storlek mättes. Datan innehåller brus på grund av att bilderna togs med autofokus. Den aktivitetssänkning som sker initialt beror på en effekt kallad *bleaching*. Denna effekt uppkommer på grund av experimentets utformning.



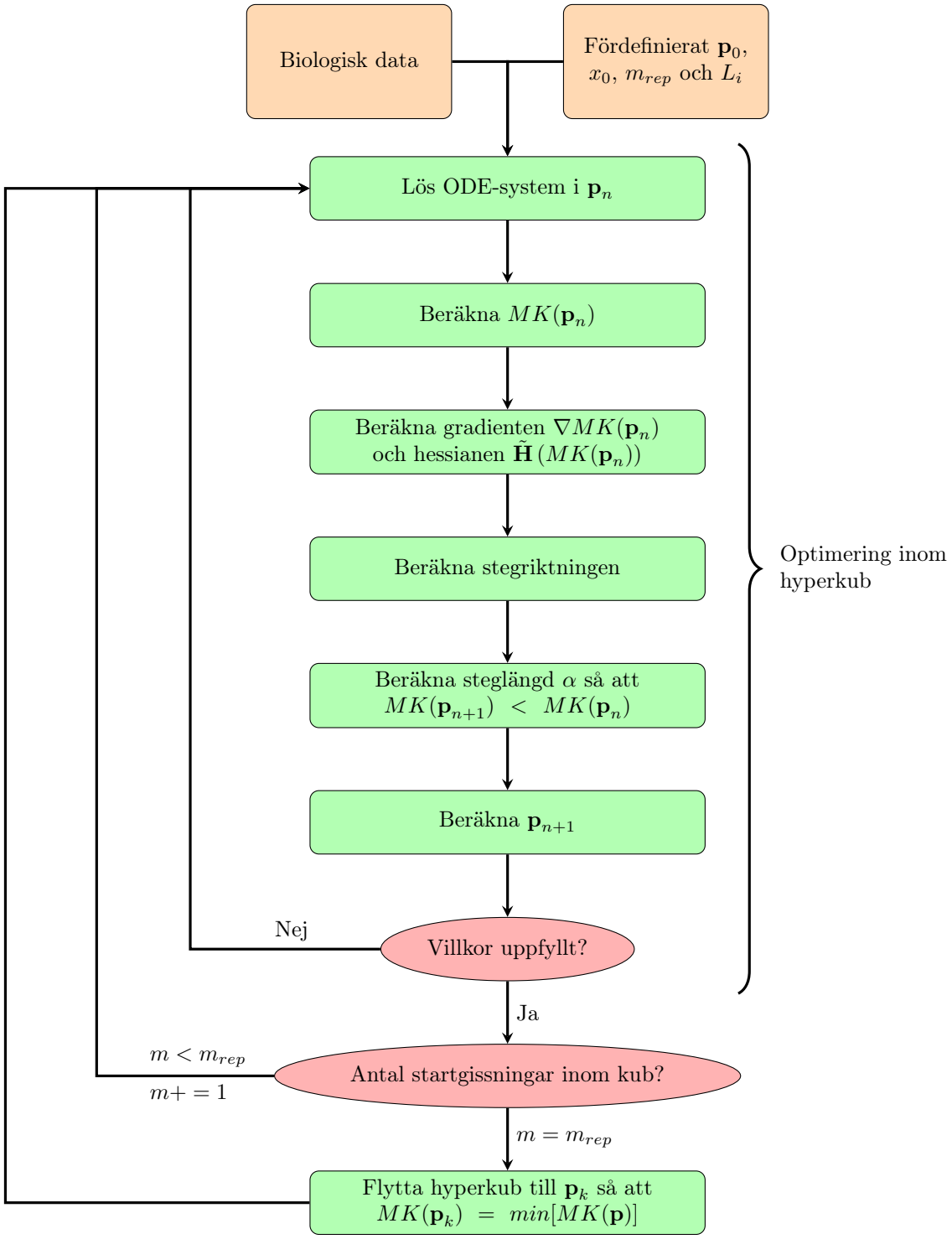
Figur 3: *Vänster: förändring av aktiviteten av SUC2 hos en enskild cell. Höger: förändring av aktiviteten av SUC2 hos samtliga celler samt aktiviteternas medelvärde. Bilden är adapterad med tillåtelse av författarna, från [31]*

4.2 Val av programmeringsspråk

Ett beslut initialt var att bestämma vilket programmeringsspråk som skulle användas för implementering av en optimeringsalgoritm. Programmeringsspråket Python valdes, dels på grund av dess användarvänlighet, dels på grund av att det är *open-source*. Den version som användes var Python 3.7. I Python användes paketen Scipy (version 1.4.1) samt Numpy (version 1.18.1) [32].

4.3 Optimering av modellen

För att anpassa modellen till data byggdes en optimeringsalgoritm. I huvudsak bestod optimeringsalgoritmen av tre loopar. Den yttersta loopen genererade en 8-dimensionell hyperkub med m_{rep} antal startgissningar i sig. Mittpunkten av hyperkuben var den uppsättning parametrar som i den föregående iterationen genererade det lägsta värdet på $MK(\hat{\mathbf{k}})$. Den mittersta loopen itererade $m_{rep} = 40$ gånger och gav varje gång en ny startgissning till den innersta loopen. Den innersta loopen optimerade varje startgissning genom att med QNM minimera $MK(\hat{\mathbf{k}})$. Parametrarna tilläts inte att lämna hyperkuben och optimeringen fortsatte tills något av termineringsvillkoren uppfylldes. Nedan följer en mer detaljerad beskrivning av de olika stegen i algoritmen, fig. 4.



Figur 4: *Schematiskt flödesschema över optimeringsalgoritmen.*

4.3.1 Hyperkub och generering av startgissningar med LHS

All kontinuerlig optimering i detta projekt skedde i 8-dimensionella hyperkuber, en dimension per parameter. Längden på kubens sidor var $L_i = \ln(2) \approx 0.693$ längdenheter och mittpunken för den första hyperkuben låg i $2 \cdot \ln(2)$ för alla parametrar. Under den kontinuerliga optimeringen tilläts parametrarna aldrig lämna hyperkuben. Om en parameter efter ett steg hamnade utanför hyperkuben gjordes en ortogonal projicering tillbaka till hyperkubens kant.

Inuti hyperkuben genererades $m_{rep} = 40$ startgissningar med LHS-metoden (avsnitt 3.4.6). Varje startgissning kördes tills något av termineringsvillkoren (se avsnitt 4.3.3) uppfylldes. Då alla 40 startgissningar körts flyttades hyperkuben så att dess nya mittpunkt blev den parameteruppsättning $\hat{\mathbf{k}}$ som i den förevarande hyperkuben genererat det lägsta värdet på $MK(\hat{\mathbf{k}})$. För att undvika att hyperkuben innefattade negativa parametervärden justerades dess position i dimensioner vars parameter var $k_i < \frac{\ln(2)}{2}$. I dessa dimensioner sattes mittpunkten till $\frac{\ln(2)}{2}$.

Målet med att definiera en hyperkub var att leta efter ett lokalt minimum inom en begränsad del av parameterrummet $\mathbb{R}^p$. När ett tillräckligt stort antal startgissningar inom hyperkuben hade testats kunde det lägsta värdet på $MK(\hat{\mathbf{k}})$ som hittats anses vara det globala minimumet inom hyperkuben. Storleken på hyperkuben har betydelse. En liten hyperkub fångar med större sannolikhet hyperkubens sanna globala minimum medan en större hyperkub söker igenom en större del av $\mathbb{R}^p$. Hyperkubens sidlängder sattes till $L_i = \ln(2)$ längdenheter, ett beslut som baserades på att halveringstiden för första ordningens parametrar bör vara proportionerliga mot $\ln(2)$ (se avsnitt 3.4.6). Dessutom ansågs $L_i = \ln(2) \approx 0.693$ utgöra en bra avvägning av fördelarna mellan en stor och en liten hyperkub. Även antalet startgissningar inom varje hyperkub, m_{rep}, har betydelse. Ett stort värde på m_{rep} ökar sannolikheten för att hitta hyperkubens sanna globala minimum medan ett litet värde sparar beräkningstid. $m_{rep} = 40$ ansågs vara en god avvägning mellan dessa faktorer.

Det kan diskuteras om L_i och m_{rep} ska vara konstanta eller variabla värden. L_i och m_{rep} sammanhänger på ett sådant sätt att en mindre hyperkub kräver färre startgissningar för att genomsökas, och tvärt om. Därför är det mest rimligt att variera L_i och m_{rep} gemensamt, eller att inte variera någon av dem. I detta projekt valdes det senare alternativet. Anledningen till detta är dels att fördelarna med att variera L_i och m_{rep} inte är uppenbara, dels att det skulle göra koden och analysen mer komplex. Dessutom skulle andra delar av koden rimligtvis behöva uppdateras. Exempelvis skulle det inte vara meningsfullt att testa steglängden $\alpha = 1$ i en hyperkub där den längsta möjliga steglängden är $\alpha < 1$.

4.3.2 Stegriktning och steglängd

En nedåtgående stegriktning $\mathbf{p}$ beräknades med hjälp av gradienten $\nabla MK(\hat{\mathbf{k}})$ och hessianapproximationen $\widetilde{\mathbf{H}}(\hat{\mathbf{k}})$, se ekv. (4), (6) och (7) i avsnitt 3.4.5. Steglängden α beräknades med Armijos regel, se ekv. (9) och (10) i avsnitt 3.4.5. Armijos regel-algoritmen gjorde $k_{max} + 1$ försök att uppfylla villkoret i ekv. (10), i detta projekt användes $k_{max} = 40$. Om villkoret inte var uppfyllt efter $k_{max} + 1$ försök ansågs Quasi-Newton-algoritmen inte ha beräknat någon nedåtstigande riktning och $\mathbf{p}$ bestämdes istället med steepest descent-metoden, ekv. (5). Steglängden förblev $\alpha = \alpha_{k_{max}}$.

För att undersöka definitheten av $\tilde{\mathbf{H}}(\hat{\mathbf{k}})$ bestämdes dess egenvärden $\boldsymbol{\lambda}$. Detta gjordes med funktionen `numpy.linalg.eig`. Varje element λ_i av $\boldsymbol{\lambda}$ beskrev definiteten av $MK(\hat{\mathbf{k}})$ i den parameterriktningen, $\lambda_i > 0$ innebar konvexitet vilket var det önskvärda. Om något egenvärde istället uppfyllde $\lambda_i < 0$ sattes stegriktningen för denna parameter till 0, alltså $p_i = 0$. På detta sätt togs steg endast i de konvexa parameterriktningarna.

4.3.3 Termineringsvillkor

Den kontinuerliga optimeringen terminerades då ett av följande villkor uppfylldes.

1. $\|\nabla MK(\hat{\mathbf{k}})\| < 0.1$: gradientnormen understeg ett toleransvärde på 0.1.

2. $|MK(\hat{\mathbf{k}}_{i-15}) - MK(\hat{\mathbf{k}}_i)| < 0.1$: absolutvärdet av skillnaden mellan det senaste och det femtonde senaste värdet på $MK(\hat{\mathbf{k}})$ understeg ett toleransvärde på 0.1.

3. $\lambda_i < 0,\ \forall i$: alla egenvärden till $\tilde{\mathbf{H}}(\hat{\mathbf{k}})$ är negativa.

4. Antalet iterationer nådde 1000.

5. Koden kunde av någon oväntad anledning inte fullföljas.

5 Resultat

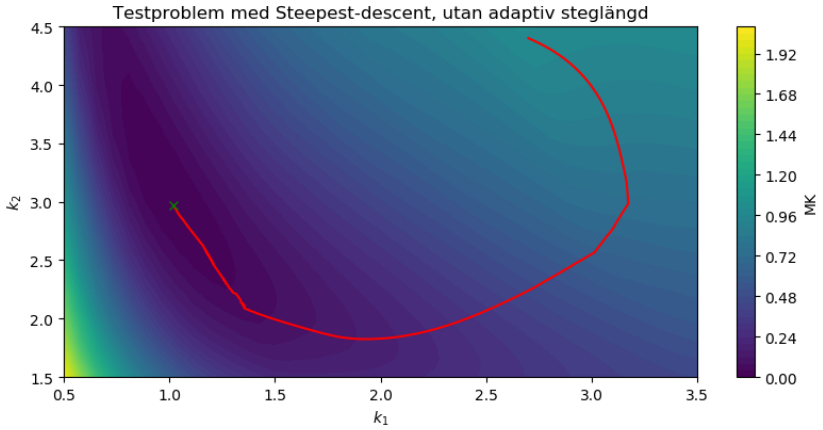
Följande kapitel innefattar de resultat som genererades under projektet. Totalt presenteras resultaten från tre modeller: det tvådimensionella testproblemen samt de två modeller som undersöktes under projektet. Dessutom visas hur valet av metod för ODE-lösning gjordes. Projektets två modeller analyserades för att se hur väl de kunde anpassas till datan samt huruvida deras matematiska egenskaper, som stabilitet och identifierbarhet, var tillfredsställande. Målet var att kunna bekräfta eller dementera projektets hypotes: att Reg1 är ansvarig för en feedback-inhibering av *SUC2*.

5.1 Algoritmen fungerade i enkla förhållanden

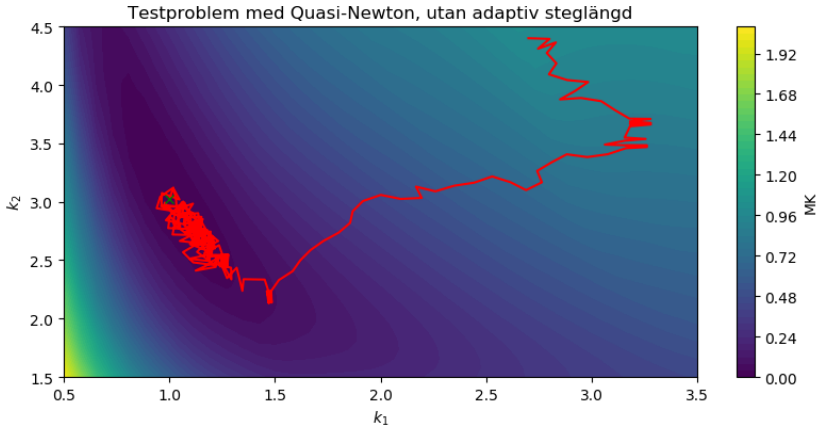
I början av projektet behövde kod testas i en enklare miljö och för det ändamålet togs ett tvådimensionellt testproblem fram. Kraven på testproblemet var att det skulle ha ett globalt minimum, att dess gradient skulle gå att beräkna analytiskt samt att det tydligt skulle kunna visualiseras grafiskt. ODE-systemet i ekv. (12) ansågs lämpligt för uppgiften.

$$\begin{cases} \frac{\mathrm{d}x_1}{\mathrm{d}t} = 1.5x_1 - x_1x_2 \\ \frac{\mathrm{d}x_2}{\mathrm{d}t} = -3x_1x_2 + x_1x_2 \end{cases} \tag{12}$$

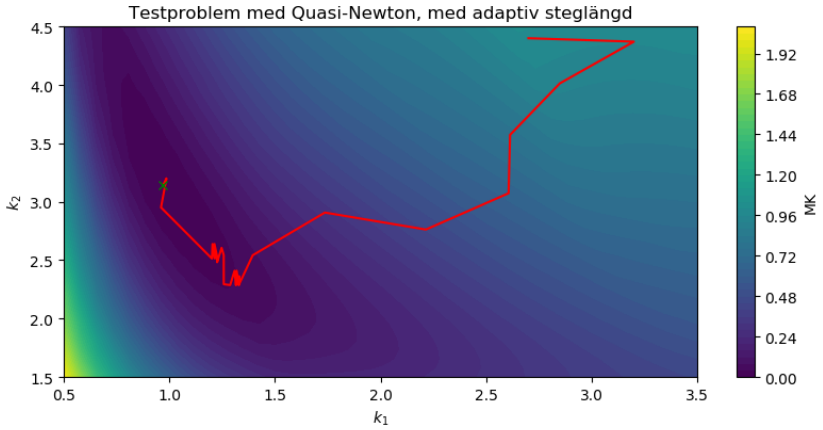
För optimeringen implementerades tre olika optimeringsmetoder: steepest descent-metoden (ekv. (5) med $\alpha = 0.1$), Quasi-Newton-metoden utan adaptiv steglängd (ekv. (4) med $\alpha = 0.1$ (se avsnitt 3.4.5)) och Quasi-Newton-metoden med adaptiv steglängd (ekv. (4) där α beräknades med ekv. (9)). För försöken med testproblemet sattes inga begränsningar för parametrarna och det fanns heller ingen kod för att automatiskt testa olika startgissningar. Resultatet från de olika metoderna kan ses i fig. 5a, 5b och 5c. Närmare diskussion om kodens uppbyggnad och innehåll ges i avsnitt 4.3.



(a) *Steepest descent-metoden utan adaptiv steglängd.*



(b) *Quasi-Newton-metoden utan adaptiv steglängd,* $\alpha = 0.1$.



(c) *Quasi-Newton-metoden med adaptiv steglängd.*

Figur 5: *Optimering av testproblem med olika metoder i rummet för minsta kostnadsfunktionen* $MK(\hat{\boldsymbol{k}})$*, ekv.* (12)*. Startposition: (2.7, 4.4).*

5.2 Stora skillnader i lösningsmetodernas prestanda

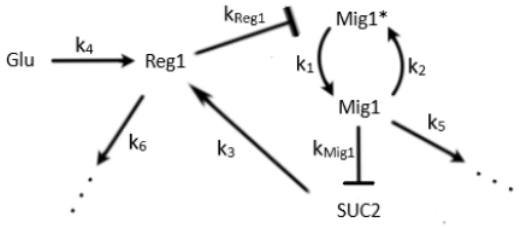
Att lösa ODE:er är en beräkningsintensiv process och ett bra val av metod för ODE-lösning kan därför spara tid. För att lösa ODE-systemet som utgör en modell användes Scipy-paketet `scipy.integrate.solve_ivp` [33]. Paketet har flera inbyggda metoder för numerisk ODE-lösning och den viktigaste skillnaden mellan dessa metoder är huruvida de är anpassade för styva eller icke-styva problem. För att ta reda på vilken metod som skulle passa bäst i detta projekt gjordes en undersökning där hastigheten hos respektive metod mättes. Tiderna från samtliga metoder presenteras i tabell 1. Utifrån resultatet ansågs metoden *backward differentiation formula* (BDF), en lösningsmetod för styva problem, vara mest lämplig. Därav är BDF den metod som användes genom hela projektet.

Tabell 1: *Tid i sekunder för att lösa systemet av ODE:er med olika ODE-lösare i scipy.integrate.solve_ivp.*

Lösningsmetod	**Tid (s)**
RK45	3.81
RK23	3.00
DOP853	3.54
Radau (styv)	0.0315
BDF (styv)	0.0156
LSODA (styv)	0.0312

5.3 Modell med Reg1 som inhibitor av Mig1 fångade inte datan

Modellen som togs fram bygger på fig. 2. Genom att använda sig av massverkans lag och inhiberingstermer formulerades modellen. I modellen görs antagandet att Reg1 inhiberar fosforylerat Mig1. Detta eftersom fosforylerat Snf1 (som inhiberas av Reg1) har en direkt korrelation med mängd fosforylerad Mig1. För att göra modellen mindre komplex och minska antalet parametrar finns detta mellansteg inte med i modellen. Även feedback-loopen, som enligt hypotesen antas vara orsakad av Reg1, kan observeras i modellen. En schematisk bild över detta kan ses i fig. 6.

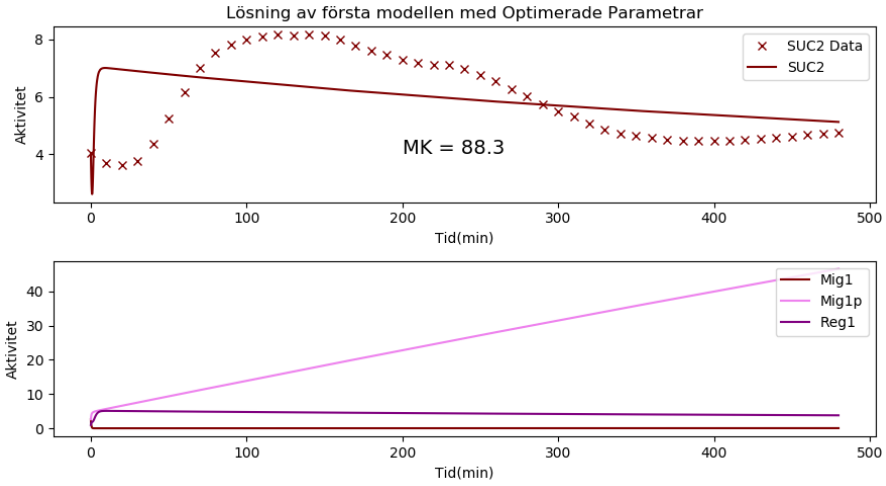


Figur 6: *Schematisk bild över reaktionsschemat.*

Med detta reaktionsschema kunde ODE-systemet i ekv. (13) formuleras. Det var denna modell som huvudsakligen testades i projektet. $Mig1^*$ betecknar den fosforylerade formen av Mig1-proteinet.

$$\begin{cases} \frac{\mathrm{d}Mig1}{\mathrm{d}t} &= k_1 Mig1^* - k_2 Mig1 - k_5 Mig1 & Mig1_0 = 2.6 \\ \frac{\mathrm{d}Mig1^*}{\mathrm{d}t} &= \frac{k_{Reg1}}{(0.1+Reg1)} + k_2 Mig1 - k_1 Mig1^* & Mig1_0^* = 2.6 \\ \frac{\mathrm{d}SUC2}{\mathrm{d}t} &= \frac{k_{Mig1}}{(0.1+Mig1)} - k_3 SUC2 & SUC2_0 = 4.0 \\ \frac{\mathrm{d}Reg1}{\mathrm{d}t} &= k_3 SUC2 + k_4[Glu] - k_6 Reg1 & Reg1_0 = 1.0 \end{cases} \tag{13}$$

Den optimerade lösningen för denna modell där Reg1 inhiberar fosforylerat Mig1 kan ses i fig. 7.



Figur 7: *Lösning för modell med Reg1-inhiberat fosforylerat Mig1 med optimerade parametrar. Heldragen linje för det modellerade värdet och kryss för den data som optimeringen skede mot samt modellens MK-värde.*

5.3.1 Optimerade parametervärden gav en otillfredsställande lösning

Optimeringen gav att de lägsta värdet på $MK(\hat{\mathbf{k}})$ för modellen gavs av parameteruppsättningen i ekv. (14).

$$\hat{\mathbf{k}}_{opt,forsta} = \begin{bmatrix} k_{1_{opt}} \\ k_{2_{opt}} \\ k_{3_{opt}} \\ k_{4_{opt}} \\ k_{5_{opt}} \\ k_{6_{opt}} \\ k_{Reg1_{opt}} \\ k_{Mig1_{opt}} \end{bmatrix} = \begin{bmatrix} 0.00306275 \\ 2.12175094 \\ 0.85285423 \\ 1.51931328 \\ 1.12286672 \\ 1.23356132 \\ 0.51101053 \\ 0.62893558 \end{bmatrix} \tag{14}$$

$\hat{\mathbf{k}}_{opt}$ genererade $MK(\hat{\mathbf{k}}_{opt}) = 88.25$. Fig. 7 visar detta resultat grafiskt.

5.3.2 Modellen med Reg1 som inhibitor uppvisade instabilitet och oidentifierbarhet

Modellen som antar inhiberin av fosforylerat Mig1 kunde konstateras vara strukturellt identifierbar, givet att någon av parametrarna k_1, k_2, k_5 eller k_{Reg1} hölls konstant (Appendix B). Detta krav är sannolikt uppfyllt då några av dessa parametrar visade sig ha liten inverkan på lösningen som ses nedan i resultatet av känslighetsanalysen.

Utifrån de optimerade parametrarna som tagits fram studerades lösningen med känslighetsanalys. Detta gjordes genom att beräkna sensitiviten för *SUC2*:s lösning i de tidpunkter som det fanns data för. En känslighetsmatris skapades och sedan kvadrerades och summerades sensitiviterna för varje parameter till ett värde. För att få ett RMS-värde normerades värdet genom att dividera med antalet mätpunkter. Dessa åtta värden återfinns i ekv. (15)

$$\mathbf{RMS}_{\mathbf{k},forsta} = \begin{bmatrix} RMS_{k_1} \\ RMS_{k_2} \\ RMS_{k_3} \\ RMS_{k_4} \\ RMS_{k_5} \\ RMS_{k_6} \\ RMS_{k_{Reg1}} \\ RMS_{k_{Mig1}} \end{bmatrix} = \begin{bmatrix} 52.09881663 \\ 30.4418167 \\ 910.47934951 \\ 1.02747983 \\ 24.21669792 \\ 15.15338539 \\ 18.3581606 \\ 925.4494055 \end{bmatrix} \tag{15}$$

Utav dessa värden sticker ett värde ut, nämligen det för parameter k_4, som är av en helt annan storleksordning. När denna parameter eliminerades från modellen och en lösning plottades med annars samma parametervärden erhölls en närmast identisk lösning för *SUC2*. Därmed konkluderades att denna parameter eventuellt skulle kunna elimineras från modellen. Antagandet om strukturell identifierbarhet som nämndes tidigare går att motivera, dock uttrycker modellen problem angående numerisk identifierbarhet.

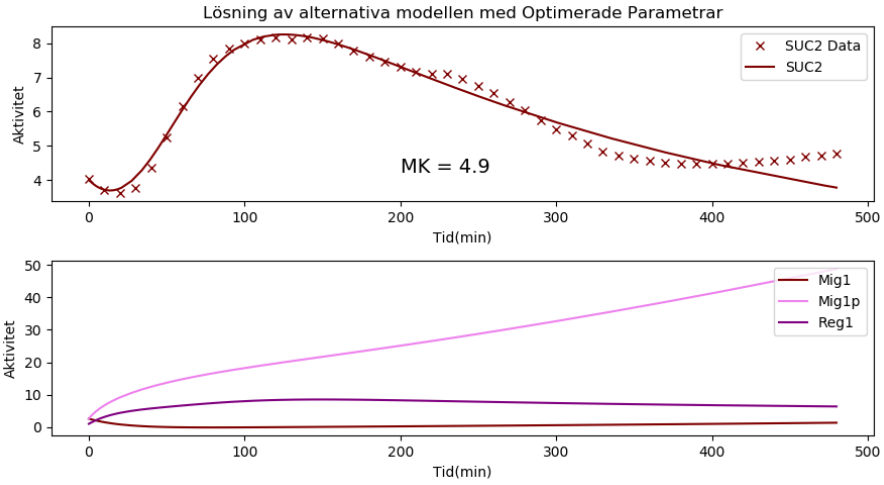
Modellen kunde även konstateras vara instabil efter linjär stabilitetsanalys (Appendix A). Detta kan konkret observeras i fig. 7 där fosforylerat Mig1 ses öka konstant och inte når ett steady-state.

5.4 En alternativ modell med konstant nedbrytning av Mig1 fångade datan bättre

Förutom projektets primära modell, ekv. (13), gjordes även försök på ODE-systemet i ekv. (16). Skillnaden mellan dessa två modeller var att termen $-k_5 Mig1$ i ekv. (13) ersattes av $-k_5$ i ekv. (16) samt att konstantvärdet i inhiberingstermen ändrades för att ge bättre anpassning. Resultatet av ekv. (16) efter optimering ges i fig. 8. Denna modell ger ett $MK(\hat{\mathbf{k}}_{opt}) \approx 4.9$, betydligt lägre än motsvarande värde för ekv. (13).

$$\begin{cases} \frac{\mathrm{d}Mig1}{\mathrm{d}t} &= k_1 Mig1^* - k_2 Mig1 - k_5 & Mig1_0 = 2.6 \\ \frac{\mathrm{d}Mig1^*}{\mathrm{d}t} &= \frac{k_{Reg1}}{(1+Reg1)} + k_2 Mig1 - k_1 Mig1^* & Mig1^*_0 = 2.6 \\ \frac{\mathrm{d}SUC2}{\mathrm{d}t} &= \frac{k_{Mig1}}{(1+Mig1)} - k_3 SUC2 & SUC2_0 = 4.0 \\ \frac{\mathrm{d}Reg1}{\mathrm{d}t} &= k_3 SUC2 + k_4[Glu] - k_6 * Reg1 & Reg1_0 = 1.0 \end{cases} \tag{16}$$

Den optimerade lösningen för denna alternativa modell med en konstant nedbrytningsterm av Mig1 kan ses i fig. 8.



Figur 8: *Lösning av modellen med konstant nedbrytning av Mig1 med optimerade parametrar samt modellens MK-värde. Heldragen linje för det modellerade värdet och kryss för den data som optimeringen skedde mot.*

5.4.1 Optimerade parametervärden gav en delvis tillfredsställande lösning

Optimerade värden för modell med konstant nedbrytning av Mig1, ekv. (17).

$$\hat{\mathbf{k}}_{opt,alternativ} = \begin{bmatrix} k_{1_{opt}} \\ k_{2_{opt}} \\ k_{3_{opt}} \\ k_{4_{opt}} \\ k_{5_{opt}} \\ k_{6_{opt}} \\ k_{Reg1_{opt}} \\ k_{Mig1_{opt}} \end{bmatrix} = \begin{bmatrix} 0.00198706 \\ 0.04083219 \\ 0.02653471 \\ 1.11447347 \\ 0.03711069 \\ 0.05146058 \\ 1.02456348 \\ 0.21713617 \end{bmatrix} \tag{17}$$

Parametrarna i $\hat{\mathbf{k}}_{opt}$, ekv. (17), genererade $MK(\hat{\mathbf{k}}_{opt}) = 4.92$.

5.4.2 En konstant nedbrytning av Mig1 genererade ett förbättrat resultat men modellen är fortfarande bristfällig

Den modell där Mig1 bröts ned med en konstant term kunde konstateras vara strukturellt identifierbar utan några antaganden (Appendix B). Därmed uppvisar själva modellstrukturen inga uppenbara brister.

Känslighetsanalys utfördes på denna modell precis som för den första modellen (se 5.3.2). De normerade värden för detta fall kan ses i ekv. (18)

$$\mathbf{RMS}_{\mathbf{k},alternativ} = \begin{bmatrix} RMS_{k_1} \\ RMS_{k_2} \\ RMS_{k_3} \\ RMS_{k_4} \\ RMS_{k_5} \\ RMS_{k_6} \\ RMS_{k_{Reg1}} \\ RMS_{k_{Mig1}} \end{bmatrix} = \begin{bmatrix} 6.96743043 \\ 25.9949222 \\ 99.1979073 \\ 0.00869513 \\ 17.7446562 \\ 5.09237234 \\ 4.18024578 \\ 92.1153168 \end{bmatrix} \tag{18}$$

Även i detta fall kunde parameter k_4 konstateras ha liten påverkan av *SUC2*:s lösning och därmed borde den även i detta fall kunna elimineras från modellen. Modellen uppvisar likt den ursprungliga modellen brister i den numeriska identifierbarheten. Instabilitet observerades i modellen (Appendix A) då fosforylerat Mig1 också i detta fall ökade konstant (fig. 8).

6 Diskussion

I helhet lyckades projektet endast delvis uppnå syftet. Det råder fortfarande stor oklarhet i hur feedback-mekanismen i Snf1-signalvägen fungerar. Optimalt hade någon av de testade modellerna lyckats fånga trenden hos datan ännu bättre samt haft önskvärda egenskaper hos en bra modell såsom stabilitet, rimliga värden på koncentrationer och parametrar, hög identifierbarhet och en god prediktionsförmåga. Ingen modell lyckades uppfylla alla dessa och därmed går det inte att bekräfta projektets hypotes. Dock går det heller inte att med säkerhet dementera hypotesen, eftersom projektet endast har testat ett begränsat antal modeller. I Appendix C diskuteras algoritmens prestanda, hur väl algoritmen kunde appliceras på ett testproblem av lägre komplexitet än modellerna samt vilka slutsatser som kan dras utifrån detta.

Den första modellens resultat i fig. 7 beskrev dessvärre ej datans trend särskilt bra. Dock kunde den ändå fånga en ökning av *SUC2*:s aktivitet i början samt en viss minskning med tiden, något som skulle stödja att en feedback-mekanism existerar. En del i denna lösning som dock var mindre bra var att aktiviteten sjönk i början. Detta händer förvisso även för den data som insamlats men anledningen där är så kallad bleaching. Den modell som ställdes upp undersökte inte detta och att då modellen trots detta lyckades fånga det väcker ytterligare frågor kring modellens tillförlitlighet. Det som förväntades av modellen var att först kunna fånga *SUC2*-aktivitetens ökning, och sedan dess feedback reglerad av Reg1. Aktiviteten för Reg1 ökar i modellen kraftigt inledningvis, i samband med att *SUC2*:s aktivitet når sitt högsta värde. Detta stärker hypotesen att *SUC2* feedbackinhiberas av Reg1. Dock förblir Reg1:s aktivitet nästan konstant, vilket inte har någon fysikalisk förklaring. Vad som hade förväntats skulle vara att när *SUC2*:s aktivitet återvänder till det normala, så sjunker även Reg1:s aktivitet. Ett problem som minskar denna modells tillförlitlighet är aktiviteterna för Mig1, där den sammanlagda halten av Mig1 och Mig1p i cellen ska vara mer eller mindre konstant. Dessvärre skjuter den fosforylerade Mig1:s aktivitet i höjden men detta kan bero på att inte några restriktioner på aktiviteterna togs med i modellen. Dessutom förväntas Reg1:s feedbackinhibering via Mig1 sänka *SUC2*:s aktivitet. I en korrekt modell borde därför Mig1:s aktivitet sjunka efter att först inledningsvis stigit med *SUC2*:s aktivitet. En annan problematisk del var att känslighetsanalysen visade att den term som bestämmer inverkan av glukoskoncentration i modellen ej påverkade lösingen. Detta är något som förefaller felaktigt, eftersom det är via just denna som *SUC2* annars aktiveras.

I den alternativa modellen där Mig1 bröts ned med en konstant term erhölls däremot en betydligt bättre anpassning till datan, då denna modell lyckades fånga trenden i datan vid rätt tidpunkter, se fig. 8. Dock, precis som för den första modellen där Reg1 inhiberar Mig1, så var det ej egentligen eftertraktat att modellen fångade minskningen i *SUC2*:s aktivitet inledningsvis i förloppet. Det finns stora likheter med lösningen för den första modellen, eftersom den generella trenden är densamma för nästan alla reaktanter. Endast Reg1 har en något ny trend, en trend som stämmer bättre överens med vad som skulle förväntas vid en feedback. När *SUC2*:s aktivitet ökar så ger detta en effekt på Reg1 varvid dess aktivitet ökar, och via en feedback sänker *SUC2*:s aktivitet. En sänkt aktivitet för *SUC2* ger då i sin tur att också Reg1:s aktivitet sjunker, även om det inte sker väsentligt. En förklaring till detta är att Reg1-aktiviteten förblir hög under ett tag innan den återgår till sin ursprungliga aktivitet. Denna modellens trovärdighet undermineras dock av att aktiviteten av fosforylerat Mig1 ökar konstant, något som visas i stabilitetsanalysen då även denna modell var instabil. Vidare utveckling av denna modell skulle behöva hantera denna avvikelse, sannolikt genom någon form av Mig1-inhibering. Dock, bortsett från Mig1:s ofysikaliska beteende är modellen lovande, eftersom den lyckas anpassa sig väl till datan och beter sig någorlunda som förväntat.

Dessutom erhålls precis samma problem i känslighetsanalysen där den termen som har med modellens glukospåverkan att göra är insignifikant för *SUC2*:s lösning. Det kan därför konstateras att ingen av modellerna kan ge ett direkt svar på hypotesen. Däremot finns det potential att utifrån denna information ta fram en ny modell som eventuellt skulle kunna lösa problemet med Mig1:s aktiviteter. Följaktligen hade ett faktiskt svar på huruvida en potentiell feedback via Reg1 faktiskt sker på ett konkret sätt skulle kunna fastställas eller klarläggas tydligare.

Vidare arbete med projektet hade fokuserat mer på modellstrukturen och att få den att bete sig rimligt ur ett matematiskt och biologiskt perspektiv. Den första modellen presenterad i resultatet ritades vid framtagningen upp grafiskt för att undersöka om den var stabil och skulle fortsätta användas. Resultatet då var att den ansågs vara stabil och alla koncentrationskurvor ansågs vara stabila och förhöll sig rimligt. Efter inkorporering av villkor, ändrade startgissningar och begränsningar i programmen visade det sig dock att modellen för den nya parameteruppsättningen var instabil och fosforylerat Mig1 ökade konstant, (fig. 7), vilket initialt inte var fallet med denna modell. Detta illustrerar att stabiliteten hos modellen är beroende av de ingående värdena i modellen samt begränsningarna som tas hänsyn till. Detta kan även observeras i stabilitetsanalysen i Appendix A. Analysen visar att olikheten $k_5 k_6 < G_1 G_2$, som är beroende av parametervärden, är en faktor som påverkar den observerade instabiliteten hos modellen. Villkor hade kunnat implementeras i den skrivna koden för att få modellen att uppvisa ett mer stabilt beteende. Detta skulle dock troligen medföra mycket sämre anpassning till datan och inte vara värt besväret då modellen fortfarande skulle ha identifierbarhetsproblem. Förslagsvis är det därför en bra idé att skapa nya modellstrukturer för att vidare testa hypotesen som projektet utgår ifrån. Även i framtagandet av nya modeller kontrollera kontinuerligt om modellen beter sig stabilt, och bibehållandet av denna egenskap skulle vara av stor vikt. Detta hade lett till en mer önskvärd dynamik och rimlighet i modellens beteende.

Den alternativa modellen visade sig vara mest lovande då den anpassade sig väl efter datan. Den tydligaste bristen med denna modell var att koncentrationen av Mig1 divergerade mot oändligheten, vilket är biologiskt orimligt. Om denna brist skulle åtgärdas skulle modellen ha potential att anses tillräckligt pålitlig för kunna bekräfta hypotesen. För att med säkerhet kunna bekräfta hypotesen skulle dock även nya biologiska experiment behöva göras. Detta för att ytterligare testa modellens prediktionsförmåga. Hade detta genomförts skulle en större förståelse om den studerade mekanismen uppnås, vilket skulle leda till en större insikt i åldrande hos biologiska organismer som jäst och i slutändan människan.

Referenser

[1] Ningthoujam SS, Talukdar AD, Sarker SD, Nahar L, Choudhury MD. Chapter 2 - Prediction of Medicinal Properties Using Mathematical Models and Computation, and Selection of Plant Materials. In: Sarker SD, Nahar L, editors. Computational Phytochemistry. Elsevier; 2018. p. 43 – 73. Available from: http://www.sciencedirect.com/science/article/pii/B978012812364500002X.

[2] Box GE. Science and statistics. Journal of the American Statistical Association. 1976;71(356):791–799.

[3] Goldstein S. The cellular basis of aging. Canadian family physician Medecin de famille canadien. 1984 Mar;30:585–589. 21279075[pmid]. Available from: https://www.ncbi.nlm.nih.gov/pubmed/21279075.

[4] Shashkova S. Dynamic regulation of the Mig1 transcriptional repressor under glucose de/repression; 2016.

[5] Hardie DG. AMP-activated/SNF1 protein kinases: conserved guardians of cellular energy. Nature reviews Molecular cell biology. 2007;8(10):774–785.

[6] Fulop T, Larbi A, Witkowski JM, McElhaney J, Loeb M, Mitnitski A, et al. Aging, frailty and age-related diseases. Biogerontology. 2010;11(5):547–563.

[7] World Health Organisation. Cancer; The Global Cancer Observatory; 2018. Available from: https://www.who.int/news-room/fact-sheets/detail/cancer.

[8] Giovannuci E, Harlan DM, Archer MC, Bergenstal RM, Gapstur SM, Habel LA, et al. Diabetes and cancer: a consensus report. Diabetes care. 2010;33(7):1674–1685.

[9] Hedbacker K, Carlson M. SNF1/AMPK pathways in yeast. Frontiers in bioscience: a journal and virtual library. 2008;13.

[10] Luo Z, Zang M, Guo W. AMPK as a metabolic tumor suppressor: control of metabolism and cell growth. Future oncology. 2010;6(3):457–470.

[11] Salminen A, Kaarniranta K, Kauppinen A. Age-related changes in AMPK activation: Role for AMPK phosphatases and inhibitory phosphorylation by upstream signaling pathways. Ageing Research Reviews. 2016;28:15 – 26. Available from: http://www.sciencedirect.com/science/article/pii/S1568163716300368.

[12] Shashkova S. Dynamic regulation of the Mig1 trancsriptional repressor under glucose de/repression [doktorsavhandling på internet]. Göteborg: Göteborgs universitet; 2016.

[13] Klipp E, Liebermeister W. Mathematical modeling of intracellular signaling pathways. BMC neuroscience. 2006 11;7 Suppl 1:S10.

[14] Kitano H. Computational systems biology. Nature. 2002;(420):206–210.

[15] Katze MG. Systems Biology. No. v. 363 in Current Topics in Microbiology and Immunology. Springer; 2013.

[16] Rubenstein EM, McCartney MM, Zhang C, Shokat KM, Shirra MK, Arndt KM, et al. Snf1 Activation Loop Phosphorylation Is Controlled by Availability of the Phosphorylated Threonine 210 to the PP1 Phosphatase. J Biol Chem. 2008;283(1):222–230. Available from: https://dx.doi.org/10.1074/jbc.M707957200.

[17] Carlson M. Glucose repression in yeast. Current opinion in microbiology. 1999;2(2):202–207. Available from: https://doi.org/10.1016/S1369-5274(99)80035-6.

[18] Chellaboina V, Bhat SP, Haddad WM, Bernstein DS. Modeling and analysis of mass-action kinetics. IEEE Control Systems Magazine. 2009;29(4):60–78. Available from: https://ieeexplore.ieee.org/abstract/document/5184956.

[19] Klipp E, Herwig R, Kowald A. Systems Biology in Practice. 1st ed. John Wiley Sons; 2005.

[20] Distefano-III J. Dynamic Systems Biology Modeling and Simulation. Elsevier; 2013.

[21] Devore JL. Probability and Statistics for Engineering and the Sciences. 8th ed. Brooks/Cole; 2012.

[22] Atkinson K, Han W, Stewart DE. Numerical solution of ordinary differential equations. vol. 108. John Wiley & Sons; 2011.

[23] Hairer E, Nørsett SP, Wanner G. Solving Ordinary Differential Equations. 1st ed. Springer International Publishing; 1987.

[24] Andreasson N, Evgrafov A, Patriksson M, Gustavsson E, Önnheim M. Introduction to Continuous Optimization. 2013;.

[25] Raue A, Schilling M, Bachmann J, Matteson A, Schelke M, Kaschek D, et al. Lessons Learned from Quantitative Dynamical Modeling in Systems Biology. PLoS ONE. 2013;8(9). Available from: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0074335.

[26] Bjorck A. Numerical methods for least squares problems. vol. 51. Siam; 1996.

[27] Cook B, Fischer J, Bachmann J, Matteson A, A Hall B, Ishtiaq S, et al. Finding instability in biological models. Springer; 2014.

[28] Jasmine A Dirody, Padmini Rangamini. An introduction to linear stability analysis for deciphering spatial patterns in signaling networks. bioRxiv. 2016;Available from: https://doi.org/10.1101/065474.

[29] Attila Gábor, Alejandro F Villaverde, Julio R Banga. Parameter identifiability analysis and visualization in large-scale kinetic models of biosystems;.

[30] Inc WR. Mathematica, Version 12.0;. Champaign, IL, 2019.

[31] Persson S, Shashkova N, Cvijovic M. SUC2 via a SNF1-dependent feedback loop;. Manuskript under granskning.

[32] Oliphant T. NumPy: A guide to NumPy; 2006. [Online; hämtad 2020-05-05]. USA: Trelgol Publishing. Available from: http://www.numpy.org/.

[33] Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods. 2020;.

[34] Stewart A Levin. Descartes' rule of sign - How hard can it be? Stanford University. 2002;.

Appendix

A Stabilitetsanalys

För att utvärdera stabiliteten hos den framtagna modellen görs nedanstående procedur:

1. Identifiera steady-state punkter
2. Ta fram Jacobianen till modellen
3. Utvärdera Jacobianen i steady-state punkterna
4. Lös ut egenvärdena från den karakteristiska ekvationen och observera tecknena på egenvärdena.

Nedan följer denna procedur för den första modellen som presenteras i resultatet.

$$\frac{\partial \mathbf{x}}{\partial t} = F(\mathbf{x}) = \begin{cases} \frac{\partial x_1}{\partial t} = F_1(\mathbf{x}) = k_1 x_2 - k_2 x_1 - k_5 x_1 \\ \frac{\partial x_2}{\partial t} = F_2(\mathbf{x}) = \frac{k_{Reg1}}{0.1+x_4} + k_2 x_1 - k_1 x_2 \\ \frac{\partial x_3}{\partial t} = F_3(\mathbf{x}) = \frac{k_{Mig1}}{0.1+x_1} - k_3 x_3 \\ \frac{\partial x_4}{\partial t} = F_4(\mathbf{x}) = k_3 x_3 + k_4 Glu - k_6 x_4 \end{cases}$$

Steady-states då $\mathrm{F}(\mathbf{x}) = 0 \rightarrow F_1(\mathbf{x}) = \mathrm{F}_2(\mathbf{x}) = \mathrm{F}_3(\mathbf{x}) = \mathrm{F}_4(\mathbf{x}) = 0$

$$F_4(\mathbf{x}) = 0 \rightarrow x_3 = \frac{k_6 x_4 - k_4 Glu}{k_3}$$

Vi får då

$$F_3(\mathbf{x}) = \frac{k_{Mig1}}{0.1+x_1} - k_3 x_3 = 0 \rightarrow k_{Mig1} = (0.1 + x_1) * k_3 * x_3 \rightarrow \left[x_3 = \frac{k_6 x_4 - k_4 Glu}{k_3} \right]$$

$$\rightarrow k_{Mig1} = (0.1 + x_1) * k_3 * \left(\frac{k_6 x_4 - k_4 Glu}{k_3} \right) = (0.1 + x_1) * (k_6 x_4 - k_4 Glu) \rightarrow x_1 = \frac{k_{Mig1}}{k_6 x_4 - k4 Glu} - 0.1$$

Vi har även $\mathrm{F}_1(\mathbf{x}) = 0$ vilket ger

$$F_1(\mathbf{x}) = k_1 x_2 - k_2 x_1 - k_5 x_1 = 0 \rightarrow x_2 = \frac{x_1(k_2+k_5)}{k_1} = \left(\frac{k_{Mig1}}{k_6 x_4 - k4 Glu} - 0.1 \right) * \frac{k_2+k_5}{k_1} =$$

$$= \frac{Z}{k_6 x_4 - k_4 Glu} - 0.1 * \frac{k_2+k_5}{k_1}$$

där $Z = \frac{k_{Mig1}(k_2+k_5)}{k_1}$

Slutligen har vi även $\mathrm{F}_2(\mathbf{x}) = 0$ som ger oss

$$F_2(\mathbf{x}) = \frac{k_{Reg1}}{0.1+x_4} + k_2 x_1 - k_1 x_2 = 0 \rightarrow \frac{k_{Reg1}}{0.1+x_4} + k_2 x_1 = k_1 x_2 \rightarrow \frac{k_{Reg1}}{0.1+x_4} + k_2 \left(\frac{k_{Mig1}}{k_6 x_4 - k4 Glu} - 0.1 \right) =$$

$$= k_1 \left(\frac{Z}{k_6 x_4 - k_4 Glu} - 0.1 * \frac{k_2+k_5}{k_1} \right) \rightarrow \frac{k_{Reg1}}{0.1+x_4} + \frac{k_{Mig1} k_2}{k_6 x4 - k_4 Glu} - 0.1 * k_2 = \frac{k_1 Z}{k_6 x_4 - k_4 Glu} - 0.1A$$

där $A = k_2 + k_5$

Vi multiplicerar sedan ekvationen med $(0.1 + x_4)(k_6 x_4 - k_4 Glu)$ och flyttar alla termer till ena sidan vilket ger oss

$k_{Reg1}k_6x_4 - k_{Reg1}k_4Glu + 0.1k_{Mig1}k_2 + k_{Mig1}k_2x_4 - 0.01k_2k_6x_4 + 0.01k_2k_4Glu - 0.1k_2k_6x_4^2 +$
$+ 0.1k_2k_4Glu * x_4 - 0.1k_1Z - k_1Zx_4 + 0.01Ak_6x_4 - 0.01Ak_4Glu + 0.01Ak_6x_4^2 - 0.1Ak_4Glu * x_4 = 0$

Detta ger oss till sist ekvationen $\beta x_4^2 + \alpha x_4 + \epsilon = 0$

där

$$\begin{cases} \beta = 0.1(Ak_6 - k_2) \\ \alpha = k_{Reg1}k_6 + k_{Mig1}k_2 - 0.01k_2k_6 + 0.1k_2k_4Glu - k_1Z + 0.01Ak_6 - 0.1Ak_4Glu \\ \epsilon = -k_{Reg1}k_4Glu + 0.1k_{Mig1}k_2 + 0.01k_2k_4Glu - 0.1k_1Z - 0.01Ak_4Glu \end{cases}$$

Alltså får vi lösningen och därmed steady-state-punkten till x_4 som $x_4 = -\frac{(\alpha/\beta)}{2} \pm \sqrt{(\frac{(-\alpha/\beta)}{2})^2 - \epsilon}$

Eftersom lösningen är en koncentration och måste vara positiv får vi steady-state punkten som $x_4 = -\frac{(\alpha/\beta)}{2} + \sqrt{(\frac{(-\alpha/\beta)}{2})^2 - \epsilon}$

Uttrycket för denna variabel kan sedan stoppas in i de andra uttrycken för att få samtliga steady-state punkter.

Vi beräknar sedan Jacobianen som

$$\mathbb{J}(\mathbf{x}) = \begin{bmatrix} \frac{\partial F_1}{\partial x_1} & \frac{\partial F_1}{\partial x_2} & \frac{\partial F_1}{\partial x_3} & \frac{\partial F_1}{\partial x_4} \\ \frac{\partial F_2}{\partial x_1} & \frac{\partial F_2}{\partial x_2} & \frac{\partial F_2}{\partial x_3} & \frac{\partial F_2}{\partial x_4} \\ \frac{\partial F_3}{\partial x_1} & \frac{\partial F_3}{\partial x_2} & \frac{\partial F_3}{\partial x_3} & \frac{\partial F_3}{\partial x_4} \\ \frac{\partial F_4}{\partial x_1} & \frac{\partial F_4}{\partial x_2} & \frac{\partial F_4}{\partial x_3} & \frac{\partial F_4}{\partial x_4} \end{bmatrix}$$

Vår Jacobian blir då

$$\mathbb{J}(\mathbf{x}) = \begin{bmatrix} -(k_2 + k_5) & k_1 & 0 & 0 \\ k_2 & -k_1 & 0 & \frac{-k_{Reg1}}{(0.1+x_4)^2} \\ \frac{-k_{Mig1}}{(0.1+x_1)^2} & 0 & -k_3 & 0 \\ 0 & 0 & k_3 & -k_6 \end{bmatrix}$$

och vår Jacobian utvärderat i Steady-state blir

$$\mathbb{J}(\mathbf{x}^*) = \begin{bmatrix} -(k_2 + k_5) & k_1 & 0 & 0 \\ k_2 & -k_1 & 0 & G_1 \\ G_2 & 0 & -k_3 & 0 \\ 0 & 0 & k_3 & -k_6 \end{bmatrix}$$

där G_1 respektive G_2 är uttrycken som fås av att stoppa in respektive steady-state-uttryck för x_4 och x_1 i Jacobianen.

Vi får då determinanten som

$det|\mathbb{J}(\mathbf{x}^*) - \lambda\mathbf{I}| = 0 \rightarrow (-\lambda - k_3) * (-\lambda - k_6) * (\lambda^2 + \lambda(k_1 + k_2 + k_5) + k_1k_5) - G_1G_2k_1k_3 = 0 \rightarrow \ldots \rightarrow \lambda^4 + \lambda^3((k_1 + k_2 + k_5) + (k_3 + k_6)) + \lambda^2(k_1k_3k_5k_6(k_3 + k_6)(k_1 + k_2 + k_5)) + \lambda(k_1k_3k_5k_6(k_3 + k_6)(k_1 + k_2 + k_5)) + k_1k_3k_5k_6 - G_1G_2k_1k_3 = 0$

eller på en lättare form

$$\lambda^4 + \theta_1\lambda^3 + \theta_2\lambda^2 + \theta_2\lambda + Y = 0$$

där

$$\begin{cases} \theta_1 = k_1 + k_2 + k_3 + k_5 + k_6 \\ \theta_2 = k_1k_3k_5k_6(k_3 + k_6)(k_1 + k_2 + k_5) \\ Y = k_1k_3k_5k_6 - G_1G_2k_1k_3 \end{cases}$$

och λ är egenvärdet i den karakterisktiska ekvationen.

Eftersom alla parametrar är positiva kommer θ_1 och θ_2 vara positiva termer medan Y endast kommer vara positiv om $k_5k_6 > G_1G_2$. Genom att sätta in värden på de optimerade parametrarna från resultatet fås att $k_5k_6 < G_1G_2$, och därmed kommer Y i detta fall vara negativt.

Genom att applicera Descartes teckenregel på ekvationen $\lambda^4 + \theta_1\lambda^3 + \theta_2\lambda^2 + \theta_2\lambda + Y = 0$, där Y är ett negativt tal, erhålls resultatet att en av lösningarna till ekvationen alltid kommer vara positiv [34]. Eftersom minst ett av egenvärdena är positivt, kommer alltså denna modell vara instabil.

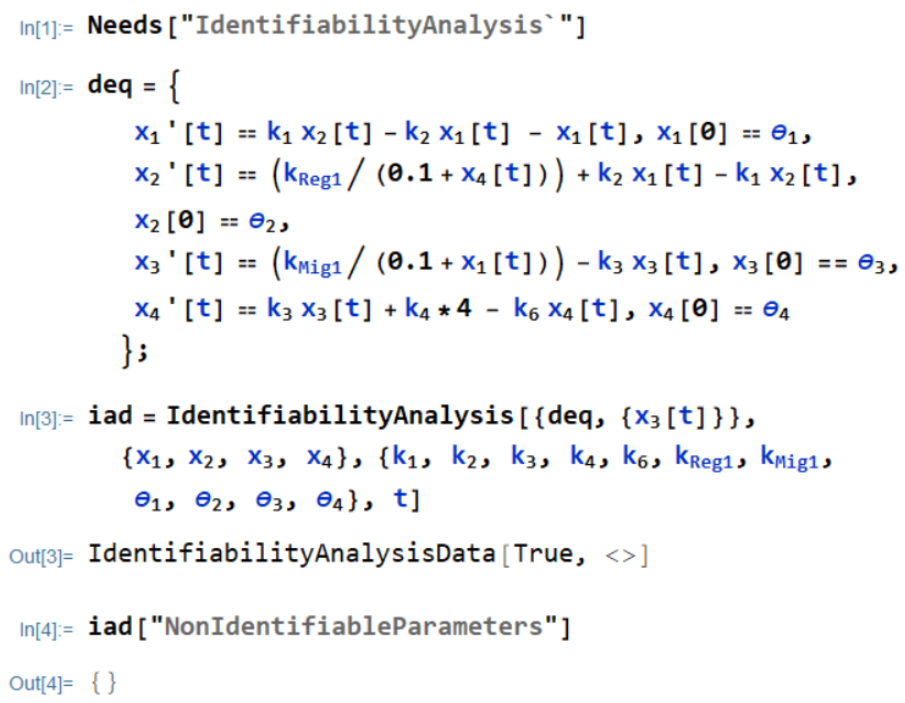
Även den alternativa modellen som testades kunde också konstateras vara instabil enligt samma procedur.

B Strukturell identifierbarhetsanalys

Nedan syns strukturell identifierbarhetsanalys för den första modellen presenterad i resultatet (fig. 9) och den alternativa modellen (fig. 10). Oidentifierbara parametrar ses längst ned i *out-putet* hos figurerna. Strukturell identifierbarhetsanalys genomfördes med hjälp av Wolfram Mathematica version 12.0 och paketet `IdentifiabilityAnalysis`.

```
In[•]:= Needs["IdentifiabilityAnalysis`"]

In[•]:= deq = {
          x1'[t] == k1 x2[t] - k2 x1[t] - k5 x1[t], x1[0] == θ1,
          x2'[t] == (kReg1 / (0.1 + x4[t])) + k2 x1[t] - k1 x2[t], x2[0] == θ2,
          x3'[t] == (kMig1 / (0.1 + x1[t])) - k3 x3[t], x3[0] == θ3,
          x4'[t] == k3 x3[t] + k4 * 4 - k6 x4[t], x4[0] == θ4
         };

In[•]:= iad = IdentifiabilityAnalysis[{deq, {x3[t]}}, {x1, x2, x3, x4},
        {k1, k2, k3, k4, k5, k6, kReg1, kMig1, θ1, θ2, θ3, θ4}, t]

Out[•]= IdentifiabilityAnalysisData[False, <>]

In[•]:= iad["NonIdentifiableParameters"]

Out[•]= {k1, k2, k5, kReg1, θ2}
```

Figur 9: *Strukturell identifierbarhetsanalys i Wolfram Mathematica för den första modellen.*



Figur 10: *Strukturell identifierbarhetsanalys i Wolfram Mathematica för den alternativa modellen.*

Den ursprungliga modellen visar sig vara strukturellt identifierbar om någon av parametrarna k_1, k_2, k_5 eller k_{Reg1} hålls konstant, vilket är ett rimligt antagande om det visar sig att någon av dem har liten påverkan på lösningen.

Den alternativa modellen utan k_5 är som en konsekvens av föregående resonemang strukturellt identifierbar som den ser ut, utan några antaganden eller liknande.

C Analys av algoritmens prestanda

Testproblemet var ett ODE-system som endast var beroende av två parametrar. Detta gjorde det möjligt att visualisera problemet och lösningsgångarna med en nivåkurva. Illustrationer av detta finns i fig. 5a, 5b och 5c. För startgissningar långt ifrån det globala minimumet terminerade algoritmen vid lokala minimum, långt ifrån det globala. Detta poängterade hur stor betydelse startgissningar kunde ha, en betydelse som blev än viktigare i modellerna med fler parametrar. Testproblemet gav även ett tillfälle att visualisera hur olika lösningmetoder såg ut. En metod att använda vid kontinuerlig optimering är steepest descent-metoden, fig. 5a, där ett steg tas i den riktning där lutningen är störst. Detta kan dock kräva många steg innan minimum till slut nås och det är också svårt att veta hur stor steglängd som är nödvändig. Genom att använda Quasi-Newton metoden i fig. 5b och 5c, så togs en mer direkt väg till minimum varvid problemet kunde lösas snabbare, särskilt när steglängden var adaptiv.

Det kan diskuteras varför BDF-metoden för att lösa ODE:er gick snabbare än RK45. Båda metoderna använder sig av en algoritm av fjärde ordningen och följaktligen ett trunkerat fel av femte ordningen. Således borde inte noggrannheten i beräkningen vara skälet till den mer tidskrävande beräkningen. Metoderna skiljer sig genom att de är olika väl lämpade för styva och icke-styva problem. En RK-metod är bättre lämpad för icke-styva problem och en Adams-metod för styva problem. Vid beräkning av ett styvt problem med RK uppstår ofta problem så som divergerande beräkningar samt ovanligt många iterationer i algoritmen. Då beräkning av den framtagna modellen med RK45 var tidskrävande och gav felmeddelanden kan argumentet ges att modellen är en styv modell som är lämpad för lösning med en Adams-metod [22]. I tabell (1)

syns det att modellen mest troligen är styv och för att spara tid så borde den metod med kortast lösningstid väljas. Detta visade sig då vara BDF varvid denna metod användes.

D Kod

I koden användes tilläggen `numpy` [32] samt `scipy` [33].

```
import numpy as np
import matplotlib.pyplot as plt
import scipy.integrate as integrate
from scipy.interpolate import interp1d
import pandas as pd
import sys
import time
import random
from datetime import datetime
import os

# ------------------------------ DATA READING ----------------------------------
plt.ion()
# Read the data file into a pd data-frame and convert to numpy array (for speed)
data = pd.read_csv("C:/Users/fredr/OneDrive/Dokument/-- Skola --/Kandidaten/Data/Data_set_mean_noq
# data = pd.read_csv("C:/Users/Jakob/Documents/Chalmers/Skola/Kandidat/Data/Data_set_mean.csv")

x_data = data.values  # Turn the data into np-array (matrix)
global t_data
global t_eval
global SUC2_data
global SUC2_var
global break_code
break_code = 0
t_data = x_data[:, 0]
t_eval = np.linspace(0, 500, 1000)
SUC2_data = x_data[:, 3]
SUC2_var = np.var(SUC2_data)

#---------------------Define functions-------------------------------


def ODE_model(t, z, k1, k2, k3, k4, k5, k6, kReg1, kMig1):
    Mig1, Mig1p, SUC2, Reg1 = z

    Glu = 0.2
    dMig1  = k1 * Mig1p - k2 * Mig1 - k5 * Mig1
    dMig1p = kReg1 / (0.1 + Reg1) + k2 * Mig1 - k1 * Mig1p
    dSUC2  = kMig1 / (0.1 + Mig1) - k3 * SUC2
    dReg1  = k3 * SUC2 + k4 * Glu - k6 * Reg1
    return [dMig1, dMig1p, dSUC2, dReg1]


def LHS_generator(K_mid, K_range, num_samples):
    num_param = len(K_mid)
    K = []
    K_min = K_mid - K_range
    K_max = K_mid + K_range
    for i in range(num_param):
        K.append(np.linspace(K_min[i], K_max[i], num_samples))
        # K är en lista av np.arrays.

    all_start_guess = []
```

```
    for j in range(num_samples):
        start_guess_list = []
        for i, _ in enumerate(K):
            rand_int = random.randint(0, num_samples - 1 - j)
            start_guess = K[i][rand_int]
            start_guess_list.append(start_guess)
            K[i] = np.delete(K[i], rand_int)
        all_start_guess.append(start_guess_list)
    return all_start_guess, K_min, K_max


def y_int(x_sol):
    f_interp_x = interp1d(x_sol.t, x_sol.y[2], fill_value='extrapolate')
    x_sol_interpolated = f_interp_x(t_data)
    return x_sol_interpolated


def MK(x_data, x_sol):
    sum_error_x = 0
    x_sol_interpolated = y_int(x_sol)
    # Totalt error mellan modellen och experimentella datan

    for e1, e2 in zip(x_data[:, 3], x_sol_interpolated):
        sum_error_x += (e1-e2) ** 2

    return sum_error_x


def gradient(x_data,K_0,t_span,x0):
    gradient = np.zeros(np.size(K_0))
    h = np.sqrt(np.finfo(np.float).eps)      # Maskintoleransen, vår steglängd för finita differens
    K_0 = np.array(K_0)
    for param in range(np.size(K_0)):
        K_low          = K_0 + 0
        K_high         = K_0 + 0
        K_low[param]  -= h
        K_high[param] += h

        sol_k_low  = integrate.solve_ivp(ODE_model, t_span, x0, method='BDF', args=K_low)
        sol_k_high = integrate.solve_ivp(ODE_model, t_span, x0, method='BDF', args=K_high)

        gradient[param] = (MK(x_data, sol_k_high)-MK(x_data, sol_k_low))/(2 * h)

    return gradient

def H(K):
    hess_grad = gradient(x_data, K, t_span, x0)
    grad_nx1  = hess_grad[:, np.newaxis]
    hessian   = np.dot(grad_nx1, np.transpose(grad_nx1)) / (SUC2_var ** 2)
    return hessian

def step_length_iter(init_K, direction):
    alpha_max = False
    alpha_start   = 1
    max_iter      = 40
    init_func_val = integrate.solve_ivp(ODE_model, t_span, x0, method='BDF', args=(init_K))
```

```
    init_MK_val   = MK(x_data, init_func_val)
    for i in range(max_iter):
        alpha  = alpha_start * (2 ** - i)
        next_K = init_K - alpha * direction
        try:
            next_func_val = integrate.solve_ivp(ODE_model, t_span, x0, method='BDF', args=(next_K)
            next_MK_val   = MK(x_data, next_func_val)
        except:
            next_MK_val = init_MK_val
        if next_MK_val < init_MK_val:
            print(f'Step length iter: {i} ')
            break
        if i == max_iter - 1:
            alpha_max = True
            print(f'Maximum iteration in step_length_iter.')

    return alpha, alpha_max


def optimize(x_data, K_0, t_span, x0, K_min, K_max):
    K = K_0

    # Initial lösning med vår ingångsgissning på parametrarna; K_0
    x_sol = integrate.solve_ivp(ODE_model, t_span, x0, method='BDF', args=(K_0))
    opt_count = 1



    # Skapa tomma listor där uppdaterade värden på K och uppdaterade funktionsvärden läggs till ko
    K_list, X_list, MK_list = [K], [x_sol], [MK(x_data,x_sol)]

    # En while-loop som ändrar parametrarna K:s värden så att modellen passar datan bättre, till e
    while True:
        print(f'Optimize loop: {opt_count}')
        start_time = time.time()
        #Beräkning av gradient, hessian och medföljande stegriktning
        try:
            grad = gradient(x_data, K, t_span, x0)
        except:
            print(f'Break 5: Interpolation not possible: breaking loop')
            break_code = 5
            break
        print(f'Grad: \n {grad}')
        norm = np.linalg.norm(grad)
        print(f'Gradientnormen: {norm}')
        hess = H(K)
        eigval, eigvect = np.linalg.eig(hess)
        try:
            # inv_hess = np.linalg.inv(H(K))
            # direction_non_norm = np.matmul(grad, inv_hess)
            direction_non_norm = np.linalg.solve(hess, grad)
            direction = direction_non_norm / np.linalg.norm(direction_non_norm)
        except:
            print(f'H not invertible: use gradient descent')
            direction = grad / norm
        for i in range(len(eigval)):
```

```
    if eigval[i].real < 0:
        direction[i] = 0
print(f'Direction: \n {direction}')

#Beräkning av nya parametervärden samt funktionsvärden
step_length, alpha_max = step_length_iter(K, direction)
if alpha_max:
    K = K - step_length * grad / norm
else:
    K = K - step_length * direction

for i in range(len(K)):
    if K[i] < K_min[i]:
        K[i] = K_min[i]
    if K[i] > K_max[i]:
        K[i] = K_max[i]
K_list.append(K)
print(f'K: \n {K}')


x_sol = integrate.solve_ivp(ODE_model, t_span, x0, method='BDF', args=(K))
X_list.append(x_sol.y)
# print('New function values: \n', x_sol)

MK_new = MK(x_data, x_sol)
MK_list.append(MK_new)
print(f'MK: \n {MK_new}')

plt.close()
fig = plt.figure()
mngr = plt.get_current_fig_manager()
mngr.window.setGeometry(0, 0, 1000, 600)
ax1 = fig.add_subplot(111)
#ax2 = fig.add_subplot(222)
ax1.plot(t_data, SUC2_data, 'x')
ax1.plot(x_sol.t, x_sol.y[2], label='SUC2')
ax1.plot(x_sol.t, x_sol.y[0], label='Mig1')
ax1.plot(x_sol.t, x_sol.y[1], label='Mig1p')
ax1.plot(x_sol.t, x_sol.y[3], label='Reg1')
ax1.legend()
plt.pause(0.2)
fig.canvas.draw()

opt_count += 1
print(f'Looptid: {time.time() - start_time}')

# Termineringsvillkor
tol = 0.1
if norm < tol:
    print(f'Break 1: Gradient norm reached tolerance')
    break_code = 1
    break
MK_max_iter = 15
MK_tol = 0.1
if len(MK_list) > MK_max_iter:
    if abs(MK_list[-1] - MK_list[-MK_max_iter]) < MK_tol:
```

```
                print(f'Break 2: Change in MK too small')
                break_code = 2
                break
        if np.linalg.norm(direction) == 0:
            print(f'Break 4: All eigenvalues negative')
            break_code = 4
            break
        if opt_count > 300:
            print(f'Break 5: Too many optimization loops')
            break_code = 5
            break

    best_MK = min(MK_list)
    best_MK_index = MK_list.index(best_MK)
    best_K = K_list[best_MK_index]
    print(f'\n\n ------------- SUMMARY OF OPTIMIZATION -------------')
    print(f'Number of optimizations: {opt_count - 1}')
    print(f'Smallest MK value: \n {best_MK}')
    print(f'Optimized K: \n {best_K}')
    try:
        print(f'Break code: {break_code}')
    except:
        print(f'Error: No break code available')
    print(f'{datetime.now()}')
    print(f'------------------------------------------------------------ \n\n')

    return best_K, best_MK


# Känslighetsmatrisen för modellen, indata i form av parametervärden K
def sensitivity2(K,t_span,x0):
    h     = np.sqrt(np.finfo(np.float).eps)  # Maskintoleransen, vår steglängd för finita differer
    n_par = len(K)
    t_len = len(t_data)
    S     = np.zeros([len(x0), n_par, t_len])
    n     = 0

    for e in K:
        K1     = K + 0
        K2     = K + 0
        K1[n] += h
        K2[n] -= h

        Sol_high = integrate.solve_ivp(ODE_model, t_span, x0, method='BDF', args=(K1))
        Sol_low = integrate.solve_ivp(ODE_model, t_span, x0, method='BDF', args=(K2))

        Sol_1 = y_int(Sol_high)
        Sol_2 = y_int(Sol_low)

        m = 0
        for e2 in x0:
            S[m, n, 0:t_len] = (Sol_1[m]-Sol_2[m]) / (2 * h)
            m += 1
        n += 1
    return S
# Utdatan är en matris där varje egen matris är samlingen av sensitiviter för varje y-variabel och
```

```python
# Känsligheten av varje y-variabel map varje parameter, viktat över alla tidpunkter
def RMS_sense(S, K, x_sol):
    SumS = np.zeros([len(S), len(S[0])])
    i = 0
    for e1 in S:
        k=0
        for e2 in S[0]:
            j=0
            Sum = 0
            for e3 in S[0,0]:
                if x_sol.y[i, j] == 0:
                    Sum = Sum + 0
                else:
                    Sum = Sum + ((S[i, k, j] ** 2) * ((K[k] / x_sol.y[i, j]) ** 2))
                j += 1
            SumS[i, k] = np.sqrt(Sum / len(S[0, 0]))
            k += 1
        i += 1

    return SumS

# ---------------------- Calculations --------------------------

t_span  = [0, 481]
x0      = [2.6, 2.6, 4.0, 1.0]
box_num = 100
first_run = True

K_mid_val = np.array([0.73829924, 2.02092358, 1.02791562, 1.57228213, 1.09499564, 1.32033847, 0.91
K_range = .5 * np.log(2)
num_samples = 20

while True:

    if first_run:
        LHS_start_guess, K_min, K_max = LHS_generator(K_mid_val, K_range, num_samples)
    else:
        K_mid_val = best_K_list[best_run_index]
        for i in range(len(K_mid_val)):
            if K_mid_val[i] < K_range:
                K_mid_val[i] = K_range
        LHS_start_guess, K_min, K_max = LHS_generator(K_mid_val, K_range, num_samples)

    best_K_list = []
    best_MK_list = []
    failed_opt = []
    LHS_start_guess, K_min, K_max = LHS_generator(K_mid_val, K_range, num_samples)
    LHS_start_guess = np.array(LHS_start_guess)
    LHS_rows = np.ma.size(LHS_start_guess, 0)

    for n in range(LHS_rows):
        now     = str(datetime.now()).replace(':', '.')
        dir_title = str(f'C:/Users/fredr/PycharmProjects/Kandidattest_19_02/Resultat/Resultat box
        title   = str(f'C:/Users/fredr/PycharmProjects/Kandidattest_19_02/Resultat/Resultat box {b
        try:
```

```
            sys.stdout = open(title, 'w')
        except:
            try:
                os.makedirs(dir_title)
                sys.stdout = open(title, 'w')
            except OSError:
                print(f'Creation of the directory {dir_title} failed.')
            else:
                print(f'Successfully created the directory {dir_title}')
        print(f'Box {box_num}, optimization {str(n)} Fredrik Initiated: {now}')
        print(f'Initial parameter values: \n {LHS_start_guess[n, :]}')
        try:
            best_K, best_MK = optimize(x_data, LHS_start_guess[n, :], t_span, x0, K_min, K_max)
            best_K_list.append(best_K)
            best_MK_list.append(best_MK)
        except:
            print(f'\n\n%%% OPTIMIZATION ERROR %%% \n Could not optimize start guess \n {LHS_start
            failed_opt.append(n)

    summary_title = str(f'C:/Users/fredr/PycharmProjects/Kandidattest_19_02/Resultat/Resultat box
    sys.stdout = open(summary_title, 'w')
    best_run_index = best_MK_list.index(min(best_MK_list))
    print(f'\n\n ------------- SUMMARY OF OPTIMIZATION BOX {box_num} ------------- \n')
    print(f'Number of start guesses: {LHS_rows} \n')
    print(f'Number of successful runs: {LHS_rows - len(failed_opt)}')
    print(f'Failed attempts indexes: {failed_opt} \n')
    print(f'Mid point: \n {K_mid_val} \n = ln(2) * \n {K_mid_val / np.log(2)} \n')
    print(f'K range (mid point +-): {K_range} \n = ln(2) * \n {K_range / np.log(2)} \n')
    print(f'Best run index: {best_run_index} \n')
    print(f'Best MK: \n {best_MK_list[best_run_index]} \n')
    print(f'Best parameters: \n {best_K_list[best_run_index]} \n')
    print(f'Best MK list: \n {best_MK_list} \n')
    print(f'Best K list: \n {best_K_list} \n\n')
    print(datetime.now())
    print(f'------------------------------------------------------------ \n\n')

    box_num += 1
    first_run = False
```