CHALMERS

GÖTEBORGS UNIVERSITET

Matematisk modell av hur glutamin påverkar fosforylering av Sch9 via Target of Rapamycin i *Saccharomyces cerevisiae*

MVEX01-18-21

Kandidatarbete inom civilingenjörsutbildningen vid Chalmers

Andrea Clausen Lind
David Lund
Sebastian Olsson
Sebastian Persson

Institutionen för matematiska vetenskaper
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Göteborg, Sverige 2018

Matematisk modell av hur glutamin påverkar fosforylering av Sch9 via Target of Rapamycin i *Saccharomyces cerevisiae*

MVEX01-18-21

Kandidatarbete i matematik inom civilingenjörsprogrammet Bioteknik vid Chalmers

Andrea Clausen Lind
David Lund
Sebastian Olsson
Sebastian Persson

Handledare: Johannes Borgqvist
Marija Cvijovic
Barbara Schnitzer

Examinator: Marina Axelson-Fisk
Maria Roginskaya

Institutionen för matematiska vetenskaper
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Göteborg, Sverige 2018

Populärvetenskaplig presentation

Har du någonsin funderat över hur länge du kommer att leva? Eller hur länge du vill leva? Idag lever folk längre än någonsin tack vare de framsteg som gjorts inom forskning och sjukvård. Men även om vi lever längre betyder inte det nödvändigtvis att vi mår bättre. Därför är det viktigt att forskning bedrivs för att vi ska kunna ha ett friskare liv när vi blir äldre, vilket kallas hälsosamt åldrande. Idag har vi bättre förståelse för hur våra celler fungerar och vad som händer när de åldras. Detta kan potentiellt utnyttjas för att förhindra bildande av skada som sker i våra celler under en livstid och hade i så fall inneburit att vi kan fortsätta vara friska under en längre tid.

Ansamling av skada i celler under en livstid är en universell process som händer i alla organismer, från de mest enkla bakterier till oss människor. Hur detta sker är inte fullständigt klart, men det är känt att vissa processer relaterade till detta har samma funktion i olika organismer. Det innebär att det går att studera processer hos enklare organismer och sedan tillämpa den erhållna kunskapen på mer komplexa varelser. Den organism som studeras i denna studie är *Saccharomyces cerevisiae*, det vill säga helt vanlig bagerijäst, med målet att i slutändan kunna applicera den kunskap som erhålls på människor.

Inom åldersrelaterad forskning är jäst intressant av flera anledningar. Jäst är nämligen en välkänd, billig och lätthanterlig organism som dessutom har en kortare livslängd än människoceller. Detta är mycket fördelaktigt vid åldersrelaterad forskning där det är intressant att studera hur cellernas livslängd påverkas av olika faktorer. Målet med detta arbete är att utveckla en matematisk modell över en specifik signalväg i jästceller som är inblandad i åldrande. Det innebär att biologiska reaktioner översätts till matematiska uttryck. Denna typ av arbetssätt är kärnan inom fältet systembiologi. Arbetssättet genomsyras av ett kontinuerligt utbyte av information som erhålls från biologiska experiment och matematiska modeller. Experimenten ger information som kan användas för att bygga upp modellerna och modellerna kan i sin tur användas som verktyg för att undersöka biologiska processer. Detta gör det möjligt att studera delar av biologiska system som annars är svårstuderade i verkligheten.

De modeller som konstrueras beskriver biologiska processer *in silico*, det vill säga i en dator. Detta är fördelaktigt eftersom en sådan modell sedan kan användas för att testa hur olika förhållanden påverkar dessa processer, och i längden cellens livslängd. En fungerande modell kan spara både tid och pengar då den kan minska mängden experiment som behöver utföras. Att konstruera modellen blir lite som att lägga pussel, där man försöker hitta den kombination av parametrar som ger optimalt resultat.

Den signalväg som undersöks i denna studie heter Target of Rapamycin, förkortat TOR. Den regleras bland annat av näringsnivåerna i cellen, det vill säga hur mycket mat vi får i oss, och påverkar en mängd olika saker. Generellt går det att se det som att TOR är delaktig i cellens anpassning till olika levnadsförhållanden. Ett exempel på en process som regleras av TOR är tillverkandet av proteiner, det vill säga de byggstenar som våra celler består av. Det innebär att TOR reglerar cellens tillväxt, och således är det en intressant process att studera inom forskning relaterad till åldrande. Signalvägen existerar i en mängd olika organismer och även om det finns vissa skillnader är den på det stora hela väldigt väl bevarad genom evolutionen. Därav är det lämpligt att studera hur TOR ser ut i jäst med förhoppningen att till slut kunna applicera resultatet även på människor. Signalvägen är dock väldigt stor, med många ingående komponenter, och vid skapande av modellen uteslöts en del komponenter. Anledningen till detta var att det ej hade varit möjligt att ta hänsyn till alla komponenter på en gång på grund av modellens komplexitet. I detta arbete modelleras alltså endast en del av signalvägen.

Projektet resulterade i en modell som delvis kunde beskriva TOR, men den hade vissa brister. Bristerna medför att modellen ej kan användas för att förutsäga något om cellens livslängd om inte de inte först tas hänsyn till och åtgärdas. Den skapade modellen ger å andra sidan en indikation på att det går att skapa en fullständig modell av TOR, om mer arbete läggs på detta. Modellen skulle sedan kunna tillämpas för att erhålla ökad förståelse för hur signalvägen fungerar, och hur den påverkas av olika faktorer. I längden skulle modellen även kunna användas vid framställning av läkemedel som påverkar TOR. Allt detta skulle förhoppningsvis kunna bidra till att vi kan leva

friskare liv. För oavsett om vi lever tills vi är 80 eller 150 år gamla så vill väl alla vara friska?

Sammanfattning

Target of Rapamycin (TOR) är en signalväg i metabolismen hos eukaryota celler som reglerar tillväxt. Syftet med arbetet var att formulera en matematisk modell som kunde återskapa en specifik utsignal för TOR i jästceller, närmare bestämt aktiveringen av kinaset Sch9 som respons på aminosyran glutamin. För att åstadkomma detta översattes ingående reaktioner till matematiska hastighetsuttryck och utifrån dessa formulerades en modell bestående av ordinära differentialekvationer (ODE:er). Vissa antaganden och avgränsningar gjordes vid uppställning av ODE:erna såsom att alla sönderfall var av första ordningen. Samtidigt skrevs ett parameteruppskattningsprogram i MATLAB, med målet att bestämma de okända hastighetskonstanterna i modellerna.

Ett antal olika modeller konstruerades utifrån massverkans lag och testades med varierande resultat. Den tillgängliga mätdatan tillät konstruktion av både större och mindre modeller. Överlag lyckades de mindre modellerna fånga mätdatans utseende relativt bra, men variansen var väldigt stor hos de skattade hastighetskonstanterna. De större modellerna som testades misslyckades med att fånga utseendet hos mätdatan.

Det finns flera möjliga förklaringar till problemen med modellerna. Problemen med variansen kan bero på att de små modellerna innehåller för många okända parametrar och/eller att för lite mätdata fanns tillgänglig. Problemen hos de stora modellerna beror troligen på antagandet som gjordes vid uppställningen av ODE:erna. Även om de stora modellerna misslyckades indikerar resultatet hos de mindre modellerna att dynamiken hos signalvägen TOR är möjlig att modellera. Ytterligare arbete krävs dock innan modellen kan göra förutsägelser om TOR på ett verklighetstroget sätt.

Abstract

Target of Rapamycin (TOR) is a metabolic signaling pathway that regulates growth in eukaryotic cells. The purpose of the project was creating a mathematical model able to recreate a specific output of TOR in yeast cells, namely activation of the kinase Sch9 in response to the amino acid glutamine. To this end the included reactions were converted into mathematical reaction rates and ordinary differential equations (ODE:s). Some assumptions and delimitations were made while constructing the ODE:s, such as all dissociations being first-order reactions. Simultaneously, a parameter estimation program was created in MATLAB with the purpose of determining the unknown rate constants in the model.

A number of different models were constructed using the law of mass action and were then tested with varying results. The available data allowed construction of both large and small models. In general, the smaller models managed to capture the shape of the data relatively well, however the variance of the estimated rate constants was very large. The larger models that were tested failed to capture the shape of the data.

There are several possible explanations for why these models failed. Issues with the variance could originate from the small models containing too many unknown parameters and/or the fact that not enough data was available. The issues with the larger models can most likely be explained by the assumptions that were made when constructing the ODE:s. Even though the large models failed the results yielded from the smaller models indicate that modelling the the dynamic of the TOR pathway is possible. However, further work is needed before the model is able to make realistic predictions about the TOR pathway.

Innehåll

Förord

Arbetet har varit uppdelat på ett sådant sätt att flera uppgifter har genomförts parallellt. Efter den gemensamma litteraturstudie som utfördes initialt sammanställde Andrea, David och Sebastian O den insamlade informationen till relativt koncis överblick över signalvägen TOR. Samtidigt började Sebastian P fördjupa sig i ämnet parameteruppskattning. Därefter delades arbetet i separata arbetsområden. Andrea fördjupade sig inom teorin kring modellering av biologiska system och arbetade därefter löpande med att konstruera en modell av TOR, samt var delaktig i sökningen efter data. David arbetade med grafisk sammanställning och arbetade därefter i huvudsak med att testa modeller i MATLAB samt assisterade Andrea i uppställandet av dessa. Sebastian O var huvudsakligen ansvarig för datainsamling vilket involverade en mycket omfattande litteratursökning som pågick under större delen av arbetet. Sebastian P konstruerade parameteruppskattningsprogrammet i MATLAB. För att utföra detta krävdes även teoretisk fördjupning inom identifierbarhet och konditionstal. Sebastian P arbetade också tillsammans med David vid test av modeller. Under arbetets gång har en tidslogg förts tillsammans med individuella dagböcker där arbetsgången redovisas mer noggrant.

Mot slutet av arbetet påbörjades skrivandet av rapporten och huvudansvariga för respektive avsnitt presenteras i tabell 1. Observera att vissa avsnitt har flera ansvariga författare.

Tabell 1: *Ansvarig författare för varje avsnitt.*

Ansvarig författare	**Avsnitt**
Andrea Clausen Lind	3.2, 3.3, 5.1, 5.2, 5.4, 6, 7
David Lund	Populärvetenskaplig presentation, Sammanfattning, 2, 3.1, 3.2, 6, Bilaga B
Sebastian Olsson	Sammanfattning, 1, 2, 3.2, 4.1, Bilaga C
Sebastian Persson	3.4, 3.5, 4.2, 5.2, 5.3, 5.4, 6, Bilaga A och D

Viktigt att notera är att alla delar som berör parameteruppskattningsprogrammet och teori kring detta i delarna med fler författare ska tillskrivas Sebastian P. Sebastian O har varit ansvarig av korrekturläsning av hela rapporten. David är den som skapat alla figurer i arbetet med undantag för figur 13, då denna ska tillskrivas Sebastian P.

Vi vill rikta ett stort tack till våra handledare Johannes Borgqvist, Barbara Schnitzer och Marija Cvijovic för fantastiskt stöd genom hela arbetet. Ett extra tack även till Niek Welkenhuysen och Felix Held för att ni delade med er av er expertis och alltid fanns tillhands när det behövdes. Slutligen vill vi tacka Daniele Stracka et. al. för artikeln *Nitrogen Source Activates TOR (Target of Rapamycin) Complex 1 via Glutamine and Independently of Gtr/Rag Proteins* som tillhandahöll essentiell data för detta arbete.

1 Inledning

Target of Rapamycin, ofta förkortat TOR, är en central signalväg i metabolismen hos alla typer av eukaryota celler. TOR:s uppgift är att reglera celltillväxt utifrån den miljö som cellen befinner sig i, det vill säga utifrån näringstillgång samt olika stressfaktorer [1]. TOR är väldigt intressant ur forskningsperspektiv då det har visats att TOR i människoceller (mechanistic TOR eller mTOR) är involverad i flera medicinska tillstånd, så som olika former av cancer, fetma och åldrande [2]. TOR:s funktion och uppbyggnad är så kallat evolutionärt konserverad, vilket innebär att TOR skiljer sig väldigt lite mellan olika eukaryota organismer. Likheterna underlättar forskningen på TOR i människoceller eftersom man kan använda experimentella resultat från exempelvis *Saccharomyces cerevisiae* (*S. cerevisiae*) som utgångspunkt för hur mTOR fungerar hos människan [2].

Det är av flera aspekter enklare att genomföra forskning *in vivo* på *S. cerevisiae* än på människor men faktum kvarstår att det fortfarande är väldigt kostsamt och tidskrävande. Således uppenbaras fördelarna med att skapa en mekanistisk dynamisk modell över signalvägen. En sådan modell innebär att koncentrationerna av de ingående komponenterna beskrivs med så kallade ordinära differentialekvationer (ODE:er) som är baserade på massverkans lag [3]. Om en fungerande modell, som kan återspegla verklig data och experimentella observationer, kan formuleras så skulle denna modell i bästa fall kunna användas för att genomföra samma typ av experiment fast genom simuleringar *in silico*. På så vis kan simuleringarna användas för att skapa en vägledning i vilka experiment som verkar lovande och är mest intressanta att genomföra i verkligheten på *S. cerevisiae*. Genom att skära ned på antalet experiment som måste göras så kan både tid och pengar sparas.

En välformulerad och fungerande modell är inte enbart ett redskap för att testa experiment under olika förhållanden. Det är också något som kan användas för att förstå mekanismer som innan inte kunnat beskrivas. En modell formuleras med avsikten att spegla verklig data och om modellen lyckas med detta finns det belägg för att påstå att den mekanism som modellen beskriver, i någon mening är sann. Detta är ett väldigt bra tillvägagångsätt för att kartlägga biologiska mekanismer som är svåra att studera i detalj *in vivo* men som det existerar experimentell data för.

TOR är en stor och omfattande signalväg med många inblandade komponenter, vilket gör det problematiskt att skapa en heltäckande modell. Problematiken rotar sig i den matematiska komplexitet som uppstår när många komponenters koncentrationer ska beskrivas samtidigt. Ytterligare kräver en stor modell ansenlig mängd experimentell data som utgångspunkt för att kunna ge verklighetstrogna simuleringar. Till följd av detta har denna modell av TOR avgränsats till särskilda delar av signalvägen.

Den modell som har formulerats beskriver regleringen av bland annat protein- och ribosomsyntesen i *S. cerevisiae* [4]. Följaktligen har fokus för projektet varit den del av signalvägen TOR med funktionen i fråga. Hur TOR reglerar protein- och aminosyrasyntesen beror starkt på vilka näringskällor som finns att tillgå för jästcellen. Mer specifikt så är det extra viktigt vilka kvävekällor som finns tillgängliga. Den kvävekälla som jästcellen har enklast att tillgodogöra sig är aminosyran glutamin vilket gör den till en komponent med stor inverkan på regleringen [5]. Således är det intressant att bygga den matematiska modellen på TOR:s reglering utifrån glutamin som insignal.

2 Syfte

Syftet med projektet är att skapa en matematisk modell av hur signalvägen TOR reglerar protein- och ribosomsyntes i respons till aminosyran glutamin. Modellen ska kunna anpassas till experimentell mätdata med hjälp av ett egenkonstruerat parameteruppskattningsprogram som minimerar en kostnadsfunktion. Modellen ska också vara lämplig för att göra förutsägelser.

3 Teori

I detta avsnitt kommer all teori som ligger till grund för arbetet att beskrivas. Först presenteras den biologiska teorin om modellorganismen *S. cerevisiae* samt om signalvägen TOR. Därefter följer ett delavsnitt som presenterar grunderna bakom matematisk modellering av biologiska system, och hur detta vanligtvis tillämpas. Slutligen presenteras den ingående matematiken i arbetet, specifikt teorin bakom parameteruppskattning och identifierbarhet.

3.1 *Saccharomyces cerevisiae*

Saccharomyces cerevisiae är en encellig eukaryot organism som vardagligt går under benämningen bagerijäst. Att organismen är eukaryot innebär att dess genetiska information befinner sig i en cellkärna omgiven av ett cellmembran [6]. Under denna kategorin faller även alla flercelliga organismer, och många genetiska likheter återfinns mellan simplare eukaryoter och mer komplexa sådana. Exempelvis finns likheter mellan gener i *S. cerevisiae* och gener i *Homo Sapiens*, det vill säga människor [6].

Dessa likheter kan utnyttjas vid studier på så kallade modellorganismer. En modellorganism är en simplare organism som används i experiment i syfte att erhålla förståelse för cellfunktioner hos mer komplexa organismer för [6]. Exempel på vanliga eukaryota modellorganismer är flugan *Drosophilia melanogaster*, masken *Caenorhabditis elegans*, musen *Mus musculus* eller bagerijäst *S. cerevisiae* [7]. Det är vanligt förekommande att modellorganismer genmodifieras inför experiment, vilket genererar mutanta organismer. För att särskilja exempelvis en jäststam där gen A har blivit inaktiverad kallas denna mutanta stam ΔA medan den ursprungliga stammen kallas wild type [8].

S. cerevisiae var den fösta eukaryoten vars fullständiga genom blev kartlagt [8]. Att genomet är kartlagt innebär att allt är känt om organismens DNA, vilket kan utnyttjas för att studera cellfunktioner hos organismen, och med precision utföra genetiska mutationer. En ytterligare fördel specifikt vid åldersrelaterade studier är att livslängden hos *S. cerevisiae* är kort, vilket gör det möjligt att studera hur organismen förändras under sin livstid [9]. Det är dessutom allmänt accepterat att experimentera på jäst ur ett etiskt perspektiv.

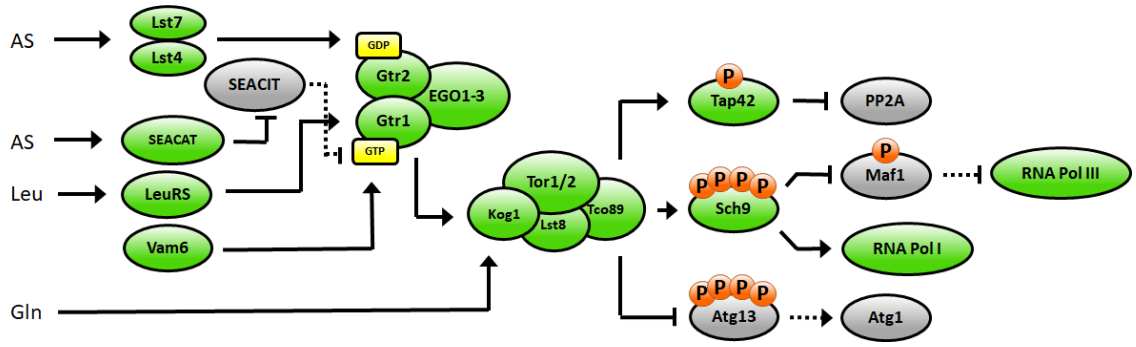
3.2 Target of rapamycin

TOR är en signalväg i *S. cerevisiae* som reglerar olika processer relaterade till tillväxt och åldrande. Själva signalvägen kretsar kring två komplex som kinaset Tor bildar tillsammans med andra proteiner, men för detta arbete är endast det ena av dessa, kallat Tor-komplex 1 (TORC1), av intresse [10]. Detta består av fyra olika komponenter: Tor1 eller Tor2, Kog1, Lst8 samt Tco89 som bildar en dimerisk struktur, vilket innebär att alla komponenter finns i dubbel uppsättning i komplexet [11]. Figur 1 visar en översikt över den del av signalvägen som involverar TORC1.

Som visas i figur 1 påverkar TORC1 flera proteiner direkt nedströms om komplexet, främst genom fosforyleringsreaktioner. Men för att TORC1 ska kunna fosforylera andra proteiner krävs först att komplexet är i ett aktivt tillstånd och denna aktivitet beror av en mängd faktorer [2]. I figur 1 illustreras endast de reaktioner som aktiverar TORC1 som respons på mängderna av olika aminosyror som finns tillgängliga i cellen. I detta arbete studeras endast aktiveringen av TORC1 via aminosyran glutamin, då denna är en av de huvudsakliga formerna i vilka kväve finns i cellen [8].

Generellt sett påverkar aminosyror TORC1 via ett annat komplex, Ego-komplexet (EGOC), som är lokaliserat vid vakuolen i cellen. Detta komplex består av proteinerna Ego1-3 som är bundna till en dimer bestående av GTPaserna Gtr1 och Gtr2 [12]. EGOC kan när det är aktiverat binda in till subenheten Kog1 på TORC1, vilket aktiverar Tor-komplexet [2]. EGOC är aktivt när GTP är bundet till Gtr1 och GDP är bundet till Gtr2 [13]. Således fyller de flesta komponenter uppströms

om EGOC en av två funktioner; antingen agerar de som GTPase-aktiverande proteiner (GAP), eller som guaninnukleotidutbytesfaktorer (GEF) [14].



Figur 1: *Översikt över hur TORC1 regleras av aminosyror, samt hur denna agerar på andra proteiner. Aktiva komponenter illustreras som gröna, inaktiva komponenter illustreras som grå. Glutamin förkortas Gln, leucin förkortas Leu och en blandning av olika aminosyror, inklusive glutamin, förkortas AS.*

Ett GAP är ett protein med förmågan att hydrolysera GTP till GDP. I figur 1 illustreras två proteiner med denna funktion uppströms om TORC1, Lst7/Lst4 och SEACIT [15]. Lst7/Lst4 påverkas av olika aminosyror, inklusive glutamin, och agerar som GAP på Gtr2 vilket gynnar den aktiva konformationen av EGOC [16]. Omvänt gäller för Gtr1, där ett GAP istället gynnar den inaktiva konformationen. Därför stimulerar aminosyrasignaler SEACAT, som då följaktligen hindrar SEACIT från att agera som GAP på Gtr1 via en okänd mekanism [14, 17]. Det är även känt att Lst7/Lst4 är involverat i en feedback-reglering av Tor-komplexet, vilket kan fosforylera Lst7/Lst4 när det är aktivt. Fosforylering får Lst7/Lst4 att dissociera från vakuolen så att det inte kan interagera med EGOC. Detta leder till minskad aktivitet hos EGOC och således minskad aktivitet hos TORC1 [18].

En GEF har istället funktionen att få GDP bundna till små GTPaser att dissociera, vilket möjliggör att GTP kan binda in istället. Eftersom Vam6 agerar som GEF på Gtr1 gynnas den aktiva konformationen av EGOC och därmed även TORC1 då Vam6 är närvarande [19, 20]. Det finns dock proteiner uppströms om EGOC som varken uppvisar GAP- eller GEF-effekt, till exempel Leucyl-tRNA-syntetas (LeuRS) som när det påverkas av aminosyran leucin direkt binder in till Gtr1 och aktiverar EGOC [2, 21]. Det har även visats att glutamin kan aktivera TORC1 direkt utan att gå via EGOC, men hur detta går till är inte känt [2, 5]. Överlag är det många mekanismer involverade i ovan nämnda processer som ännu ej är kända [2, 17, 22].

Mer känt är hur TORC1 påverkar andra proteiner och vilka effekter detta har på organismens funktion. Exempel på funktioner som regleras av TORC1 är proteinsyntes, aminosyraupptag och autofagi. Autofagi innebär nedbrytning och återvinning av dysfunktionella cellkomponenter. Proteinet Atg13 medverkar i denna process och nedregleras av TORC1 genom hyperfosforylering. Detta hindrar Atg13 från att samverka med andra Atg-proteiner för att inducera autofagi [23, 24, 25]. TORC1 är aktivt när aminosyratillgången är god, vilket innebär att Atg13 inaktiveras i detta förhållande [23, 24, 25]. Detta kan tolkas som att när det finns gott om näring behöver inte cellen bryta ned sig själv för att överleva.

TORC1 reglerar även syntes av mer aminosyror i cellen genom proteinet Tap42. Detta är en cellfunktion som blir överflödig vid god aminosyratillgång. TORC1 aktiverar Tap42 genom fosforylering. Tap42 nedreglerar sedan aktiviteten av proteinfosfatas typ 2A (PP2A) genom att binda in till denna. Detta leder bland annat till defosforylering av vissa transkriptionsfaktorer och därmed transkription av diverse proteiner [4, 26, 27].

Som tidigare nämnt är TORC1 även involverad i reglering av proteinsyntes. Regleringen sker genom Serin/Treonin-kinaset Sch9, som aktiveras när det fosforyleras av TORC1 [4, 28, 29]. När Sch9 är aktivt medverkar det i uppreglering av translation, aktivitet av RNA polymeras III (vilket sker genom deaktivering av RNA pol III-repressorn Maf1) och ribosomsyntes. Aktivt Sch9 nedreglerar

även inträde i G0-fas, cellens vilotillstånd, samt aktiviteten av RNA pol I [4, 30, 31]. Med andra ord resulterar ett aktivt Sch9 i uppreglering av proteinsyntes och celltillväxt. Ur ett åldrandeperspektiv är detta intressant att undersöka då en cells tillväxt är direkt korrelerad till dess åldrande [32].

Utöver de ovan nämnda funktionerna reglerar TORC1 ytterligare intracellulära processer, vilka ej illustreras i figur 1 [33, 34]. Aktiviteten hos TORC1 beror även av andra stimuli än aminosyror, såsom glukostillgång [35]. Med andra ord kan det konstateras att TORC1 är en mycket central komponent för många funktioner i cellen, vilka alla är intressanta att undersöka närmare. Utifrån syftet att undersöka hur TORC1 reglerar protein- och ribosomsyntes blir det dock främst intressant att studera TORC1:s reglering av Sch9. Ett verktyg för att göra detta är matematisk modellering, vilket kan användas för att fylla luckorna där rent experimentella metoder är otillräckliga.

3.3 Dynamisk modellering av biologiska nätverk

Dynamisk modellering kan användas för att beskriva biologiska nätverk *in silico*. Det gör det möjligt att beskriva hur koncentrationer av inblandade komponenter samt utsignaler från systemet ändras över tid. Vid korrekt applicering är det ett verktyg som tillåter en djupare insikt i strukturen och dynamiken hos reaktionsvägar [36, 37].

3.3.1 Dynamisk modell av ODE:er

En dynamisk modell av ett biologiskt system beskriver hur koncentrationen av olika komponenter i systemet ändras över tid vid en given insignal. Ordinära differentialekvationer, ODE:er, används ofta för att konstruera denna typ av modell. De beskriver då koncentrationsändringen av respektive ämne i reaktionsvägen, vilket kan skrivas som $\frac{\mathrm{d}x}{\mathrm{d}t}$, där x är koncentrationen av ämnet. I ett metaboliskt nätverk kan ett ämne vara inblandat i flera reaktioner, vilket innebär att ämnet både bildas och förbrukas med avseende på olika hastigheter v. Hastigheterna ställs upp via massverkans lag som beskriver villkoren för kemisk jämvikt för en reaktion [37, 38, 39]. Denna jämvikt beror av koncentrationerna av de inblandade ämnena samt en hastighetskonstant, k, för respektive reaktion. Koncentrationsberoendet kommer från att de reagerande molekylerna måste mötas för att en reaktion ska ske. Sannolikheten för att två reagerande molekyler möts är proportionell mot koncentrationen av de reagerande ämnena och därav blir reaktionshastigheten proportionell mot koncentrationerna [37, 39]. Hastighetskonstanten ger förhållandet mellan hastigheten med vilken en reaktion sker och koncentrationerna av de inblandade ämnena. Konstanten kan bestämmas experimentellt, men är mycket beroende av faktorer såsom temperatur och tryck [40]. Ett problem med experimentell bestämning av hastighetskonstanter uppstår om reaktionerna ingår i ett större system av reaktioner, såsom signalvägar som TOR. I dessa fall kan det vara svårt att undersöka hastighetskonstanten för en specifik reaktion i systemet laborativt. Ett alternativt tillvägagångssätt är då att *in silico* skatta parametrarna för ett system av uppställda reaktioner.

I modellen ställs ODE:erna upp som en summa av de hastigheter med vilka ett ämne bildas respektive förbrukas [37, 39]. Tag exempelvis den reversibla reaktionen nedan:

$$\mathrm{x_1} + \mathrm{x_2} \underset{\mathrm{k_2}}{\overset{\mathrm{k_1}}{\rightleftharpoons}} \mathrm{x_3} \tag{1}$$

Ämne x_1 och x_2 konsumeras och ämne x_3 bildas i en reaktion med hastighetskonstanten k_1. Med avseende på k_2 sker det omvända. Hastigheten, v, med vilken en reaktion sker beror på hastighetskonstantens storlek samt på koncentrationen av de ämnen som reagerar [37]. Som exempel på detta ställs reaktionshastigheterna för reaktionen i ekvation (1) upp nedan.

$$\begin{aligned} v_1 &= k_1 x_1 x_2 \\ v_2 &= k_2 x_3 \end{aligned} \tag{2}$$

I ett system av reaktioner, såsom signalvägen TOR, bildas och förbrukas en komponent med mer än en reaktion. Den totala förändringen av koncentration i tiden för en komponent x_a skrivs därmed generellt som summan av alla hastigheter med vilka ämnet bildas eller förbrukas. I summan tas hänsyn till den stökiometriska koefficienten, n_{ja}, för respektive reaktionshastighet. Index j anger vilken hastighet den stökiometriska koefficienten tillhör och a anger vilket ämne som behandlas. Den stökiometriska koefficienten beskriver om ämnet bildas eller förbrukas, samt hur mycket av ämnet som ingår i respektive reaktion. För ett ämne x_a som ingår i totalt b reaktioner ställs summan upp nedan [37, 39].

$$\frac{\mathrm{d}x_a}{\mathrm{d}t} = n_{1a}v_1 + n_{2a}v_2 + \ldots + n_{ba}v_b = \sum_{i=1}^{b} n_{ia}v_i = g_a(\mathbf{x}, \mathbf{k}, t) \tag{3}$$

Notera att ODE:n är en funktion av tillstånden, hastighetskonstanterna och tiden. Genom att sammanställa ODE:erna för de ämnen som ingår i en signalväg är det möjligt att skapa en modell av hur koncentrationen av respektive ämne med tiden. För ett generellt reaktionssystem med c ingående ämnen får modellen följande struktur

$$\begin{aligned} \frac{\mathrm{d}x_1}{\mathrm{d}t} &= g_1(\mathbf{x}, \mathbf{k}, t) \\ \frac{\mathrm{d}x_2}{\mathrm{d}t} &= g_2(\mathbf{x}, \mathbf{k}, t) \\ &\vdots \\ \frac{\mathrm{d}x_c}{\mathrm{d}t} &= g_c(\mathbf{x}, \mathbf{k}, t) \end{aligned} \tag{4}$$

Detta är hur en generell modell av en signalväg ställs upp, det är också det tillvägagångssätt som används för att ställa upp modellen av TOR.

3.3.2 Systembiologimetoden

Systembiologimetoden appliceras ofta vid modellering av biologiska nätverk, då metoden bygger på att modellering sker parallellt med laborativa experiment i en iterativ process [37, 41]. Denna process tillåter konstruktionen av integrativa modeller som beskriver biologiska nätverk, såsom signalvägar eller metaboliska processer, väl [41].

Första steget i processen är att en modell konstrueras utifrån vad som tidigare är känt om ett system av intresse. Utsignaler från modellen, i form av koncentrationer, anpassas sedan mot experimentella uppmätta utsignaler från celler via ett parameteruppskattningsprogram. Hur detta sker beskrivs i avsnitt 3.4. Utifrån resultatet från parameteruppskattningsprogrammet modifieras modellen för att bättre kunna beskriva den experimentella datan. Den modifierade modellen testas sedan på samma sätt mot experimentell data och processen upprepas [37]. De modifieringar som görs i modellen kan exempelvis vara att olika alternativa reaktioner och komponenter läggs till, eller att modellens omfattning justeras [37]. Detta fortgår till den punkt då modellen kan anses vara verifierad, det vill säga då den utan modifikation kan beskriva en ny uppsättning av experimentell data. Inom medicin kan verifierade modeller användas för att förstå mekanismen av nya läkemedel [36, 42].

Den iterativa processen, där modellen stegvis modifieras för att beskriva mätdata, är också det som möjliggör ökad förståelse för reaktionsvägens struktur och dynamik. Tag exempelvis ett fall då delar av en signalväg är okända. Då kan det faktum att modellens utsignal bättre eller sämre beskriver mätdatan när komponenter läggs till i eller tas bort från modellen användas för att bättre förstå hur reaktionsvägen är uppbyggd [37, 41]. På så sätt kan systembiologimetoden användas för att göra välriktade gissningar vid undersökning av strukturen av signalvägar, vilket minskar den mängd laborativt arbete som krävs. Jämfört med en ren experimentell metod kan systembiologimetoden

därmed sägas vara mer kostnads- och tidseffektiv. I denna studien utfördes dock inga experiment, utan all mätdata hämtades från litteratur.

3.3.3 Förenkling av en dynamisk modell av ODE:er

För att en korrekt modell ska kunna ställas upp krävs dock också att avgränsningar görs av modellen. Detta är viktigt då det är svårt för parametersuppskattningsprogrammet att identifiera det sanna värdet för respektive hastighetskonstant om antalet är för stort. Med det sanna värdet menas den enda parameteruppsättningen som genererar modellens utsignal, vilket är viktigt för att modellen ska kunna användas för korrekta prediktioner. Avgränsningar av en modell används därmed för att minska omfattningen av denna. Det kan till exempel innebära att endast effekten av en specifik insignal modelleras, att fokus endast ligger på en specifik utsignal eller att delar av reaktionsvägen utesluts beroende på vilken information som finns tillgänglig [37]. Förenklingar kan också göras i modellen genom att reaktioner läggs samman till en reaktion med gemensam hastighetskonstant under vissa antaganden [37]. Detta är ett möjligt antagande vid modellering av signalvägar, då det kan antas att koncentrationerna är låga för de ingående komponenterna i systemet [37].

3.4 Parameteruppskattning

För att anpassa en ODE-modell av TOR efter verkligheten används ett fält inom matematiken kallat parameteruppskattning [43]. Syftet inom detta fält är att med hjälp av mätdata från ett system anpassa de okända parametervärdena i systemet sådant att modellen så bra som möjligt beskriver datan. I fallet med denna studie är systemet TOR som beskrivs av en ODE-modell, som den i ekvation (4), och målet med parameteruppskattningen är att skatta de okända hastighetskonstanterna $k_1, k_2, \ldots, k_n$, vilket med vektornotation kan formuleras som att skatta den okända parametervektorn $\mathbf{k} \in \mathbb{R}^n$. Då det i denna studie endast finns mätdata för en komponent i signalvägen kommer följande punkter fokusera på att anpassa $\mathbf{k}$ efter en utsignalsvektor $\mathbf{y} = \big(y(t_1), y(t_2), \ldots, y(t_m)\big) \in \mathbb{R}^m$ som innehåller mätdata i m tidpunkter, även om metoderna som diskuteras går att generalisera till fallet med mätdata för flertalet utsignaler [43].

När det kommer till att skatta parametervektorn $\mathbf{k}$ efter mätdatan $\mathbf{y}$ finns det flertalet metoder. En möjlig, men dock ineffektiv, metod är att slumpvis generera parametervärden tills att den simulerade utsignalen för mätdatan $\hat{y}(\mathbf{k}, t)$ uppnår en bra anpassning. Ett effektivare tillvägagångsätt är att omvandla problemet till ett optimeringsproblem. Görs antagandet att skillnaden ϵ_i mellan mätdata och modellvärde för varje tidpunkt är normalfördelat enligt $\epsilon_i = y(t_i) - \hat{y}(\mathbf{k}, t_i) \sim \mathcal{N}(0, \sigma_i^2)$ och att varje observation på felet ϵ_i är oberoende går det, givet en parametervektor $\mathbf{k}$, att definiera en likelihood-funktion $L(\epsilon|\mathbf{k})$ [43] enligt:

$$L(\epsilon|\mathbf{k}) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma_i^2}} exp\bigg(- \frac{\big(y(t_i) - \hat{y}(\mathbf{k}, t_i)\big)^2}{2\sigma_i^2} \bigg) \tag{5}$$

Likelihood-funktionen är ett mått på hur sannolik en parameteruppsättning är. Att maximera den med avseende på $\mathbf{k}$ resulterar därför i den parametervektor som gör de observerade värdena $\mathbf{y}$ så sannolika som möjligt, med andra ord erhålls den parameteruppsättning som bäst beskriver datan [43]. För att underlätta maximeringen av ekvation (5) används att problemet är ekvivalent med att minimera [43] följande viktade kostnadsfunktion $f : \mathbb{R}^n \to \mathbb{R}$:

$$f(\mathbf{k}) = \sum_{i=1}^{m} \big(w_i(y(t_i) - \hat{y}(\mathbf{k}, t_i)\big)^2 \tag{6}$$

Där w_i idealt ges av $w_i = 1/\sigma_i^2$, eller om variansen inte är känd $w_i = \frac{1}{\hat{\sigma}_i}$ där $\hat{\sigma}_i$ är en skattning av variansen [43]. Vid avsaknad av tillförlitlig information kring variansen borde den antagas vara konstant och vikterna bör sättas till $w_i = 1$ för alla tidpunkter i [43].

Att minimera kostnadsfunktionen $f(\mathbf{k})$ kräver numeriska metoder. Anledningen till detta grundar sig bland annat i att $\hat{y}(\mathbf{k}, t)$ ges av att lösa det icke-linjära ODE-systemet i ekvation (4), vilket i de flesta fall är omöjligt analytiskt [44]. Minimeringen utförs därför med en lokalt verkande nedåtgående algoritm. En sådan består av följande fem steg [45]:

0. Bestäm en startpunkt $\mathbf{k_0} \in \mathbb{R}^n$. Sätt iterationsflaggan $N = 0$.
1. Välj en nedåtgående riktning $\mathbf{p}_N \in \mathbb{R}^n$.
2. Välj en steglängd $\alpha_N > 0$ sådant att $f(\mathbf{k_N} + \alpha_N \mathbf{p}) < f(\mathbf{k_N})$ gäller.
3. Låt $\mathbf{k}_{N+1} = \mathbf{k}_N + \alpha_k \mathbf{p}_N$.
4. Om ett termineringsvillkor uppfylls, stanna. Annars låt $N = N + 1$ och gå tillbaka till steg 1.

Valda tillvägagångsätt för steg ett och fyra i denna studie diskuteras i de kommande styckena. Steg två diskuteras senare i avsnitt 4.2, implementering av nedåtgående algoritm.

I en nedåtgående algoritm finns flertalet olika metoder för att välja sökriktningen $\mathbf{p}$. En möjlig metod är steepest descent i vilken $\mathbf{p}$ ges av $\mathbf{p} = -\nabla f(\mathbf{k})$, där $\nabla f(\mathbf{k}) \in \mathbb{R}^n$ är gradienten till kostnadsfunktionen [46]. Fördelen med denna metod är att den är robust, men en betydlig nackdel är att den ofta konvergerar långsamt mot ett minimum [43, 46]. Med konvergeringshastighet i åtanke är därför Newtons metod lämpligare där $\mathbf{p}$ bestäms genom att lösa följande ekvationssytem [46]:

$$H\mathbf{p} = -\nabla f(\mathbf{k}) \tag{7}$$

Där $H \in \mathbb{R}^{n \times n}$ är hessianen till kostnadsfunktionen. Det finns dock en nackdel med Newtons metod och det är att H ofta är väldigt kostnadskrävande att beräkna [46]. En tredje metod som kringgår detta problem, och som huvudsakligen används i denna studie, är quasi-Newtons metod [46]. Fördelen med denna metod är att istället för att använda H vid beräkning av $\mathbf{p}$ används en hessianapproximation $\tilde{H} \in \mathbb{R}^{n \times n}$. Ytterligare genom att ändå innehålla en approximation av H tar den fortfarande hänsyn till kostnadsfunktionens lokala utseende och konvergerar oftast snabbare än steepest descent [43, 46].

För att kunna bestämma sökriktningen $\mathbf{p}$ med quasi-Newtons metod måste $\tilde{H}$ och $\nabla f(\mathbf{k})$ beräknas. I bilaga A visas att $\tilde{H}$ och $\nabla f(\mathbf{k})$ ges av följande kompakta matrisuttryck:

$$\nabla f(\mathbf{k}) = -2A^T(\mathbf{k}) W r(\mathbf{k}) \qquad \tilde{H} = 2A^T(\mathbf{k}) W A(\mathbf{k}) \tag{8}$$

Där ett element i residualvektorn $r(\mathbf{k}) \in \mathbb{R}^m$ ges av $r_i = y(t_i) - \hat{y}(\mathbf{k}, t_i)$. Matriserna $A(\mathbf{k}) \in \mathbb{R}^{m \times n}$ och $W \in \mathbb{R}^{m \times m}$ ges av:

$$A(\mathbf{k}) = \begin{bmatrix} \frac{\partial \hat{y}(\mathbf{k},t_1)}{\partial k_1} & \frac{\partial \hat{y}(\mathbf{k},t_1)}{\partial k_2} & \cdots & \frac{\partial \hat{y}(\mathbf{k},t_1)}{\partial k_n} \\ \frac{\partial \hat{y}(\mathbf{k},t_2)}{\partial k_1} & \frac{\partial \hat{y}(\mathbf{k},t_2)}{\partial k_2} & \cdots & \frac{\partial \hat{y}(\mathbf{k},t_2)}{\partial k_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \hat{y}(\mathbf{k},t_m)}{\partial k_1} & \frac{\partial \hat{y}(\mathbf{k},t_m)}{\partial k_2} & \cdots & \frac{\partial \hat{y}(\mathbf{k},t_m)}{\partial k_n} \end{bmatrix} \quad W = \begin{bmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_m \end{bmatrix} \tag{9}$$

Utifrån uttrycket av $A(\mathbf{k})$ framgår det att de partiella derivatorna måste beräknas. En möjlig metod, vilken används i denna studie, är att använda finita differenser där den partiella derivatan $\frac{\partial \hat{y}(\mathbf{k},t_j)}{\partial k_i}$ approximeras enligt:

$$\frac{\partial \hat{y}(\mathbf{k}, t_j)}{\partial k_i} \approx \frac{\hat{y}(k_1, \ldots, k_i + \delta, \ldots, k_n, t_j) - \hat{y}(k_1, \ldots, k_i, \ldots, k_n, t_j)}{\delta} \tag{10}$$

Det finns dock ett problem med att använda ekvation (10), vilket är valet av steglängd δ. Problematiken grundar sig att parametrarna i biologiska modeller kan variera kraftigt vilket gör att ett värde på δ som approximerar derivatan bra för en parameter approximerar derivatan dåligt för än annan. Trots dessa problem går det att visa att valet $\delta = \sqrt{\text{eps}}$, där eps är maskintoleransen, resulterar i en bra approximation [47]. Med detta val av steglängd blir ekvation (10) likvärdig med en vanlig, väl fungerande, approximationsmetod som grundar sig på användandet av kedjeregeln, som i denna studie inte används över finita differenser eftersom den kräver mer arbete att implementera i MATLAB [43].

Att, med hjälp av differenskvoten i ekvation (10), enbart bestämma sökriktningen $\mathbf{p}$ via quasi-Newtons metod i ekvation (7) kan dock vara problematiskt. Två möjliga problem som kan påverka $\mathbf{p}$ är att $\tilde{H}$ är negativt definit och att $\tilde{H}$ har ett högt konditionstal $\kappa(\tilde{H})$. Det första problemet grundar sig att quasi-Newtons endast söker efter lokala extrempunkter och därav kan en uppåtgående riktning erhållas ur ekvation (7) [46]. Är $\tilde{H}$ negativt definit ($\tilde{H} \prec 0$) erhålls garanteras en uppåtgående riktning, medan om $\tilde{H} \succ 0$ erhålls en nedåtgående riktning [46]. Det andra problemet, konditionstalet hos $\tilde{H}$, $\kappa(\tilde{H})$, är ett mått på hur avrundningar i till exempel MATLAB påverkar lösningen av ekvationssystemet i ekvation (7) [45]. Ett stort $\kappa(\tilde{H})$ är en indikation på att numeriska problem kan uppstå och att lösningen blir inexakt, konsekvensen av detta blir att den erhållna sökriktningen $\mathbf{p}$ nödvändigtvis inte är nedåtgående. Sammantaget behöver båda dessa problem tas hänsyn till vid implementering, hur det görs i denna studie redovisas i avsnitt 4.2.

Förutom en metod för att bestämma sökriktningen $\mathbf{p}$ behöver en nedåtgående algoritm också lämpliga termineringsvillkor [46]. Huvudsyftet med dessa är att avgöra om algoritmen är nära ett minimum och om så är fallet avbryta sökningen. I denna studie används tre villkor, konstruerade att undvika skalningsproblem hos $\mathbf{k}$, $f(\mathbf{k})$ och $\nabla f(\mathbf{k})$ [46]. Villkoren är:

1. $\|\nabla f(\mathbf{k}_N)\| \leq \epsilon_1(1 + |f(\mathbf{k}_N)|$, $\epsilon_1 > 0$ litet
2. $f(\mathbf{k}_{N-1}) - f(\mathbf{k}_N) \leq \epsilon_2(1 + |f(\mathbf{k}_N)|)$, $\epsilon_2 > 0$ litet
3. $\|\mathbf{k}_{N-1} - \mathbf{k}_N\| \leq \epsilon_3(1 + \|\mathbf{k}_N\|)$, $\epsilon_3 > 0$ litet

Där $\|\cdot\|$ är L_2-normen. Termineringsvillkor 1 uppfylls om gradienten är nära noll, villkoret grundar sig i att om $\nabla f(\mathbf{k}) \approx 0$ så befinner sig sökningen nära en stationär punkt [46]. Villkor 2 uppfylls om kostnadsfunktionen $f(\mathbf{k})$ är plan, medan villkor 3 avbryter sökningen givet att väldigt små steg tas i parameterrummet, båda dessa är indikationer på att algoritmen stegar runt ett minimum [46]. Samtliga termineringsvillkor avbryter sökningen oavsett om minimumet är lokalt eller globalt. Konsekvensen blir att den beskrivna nedåtgående algoritmen är lokal och endast söker efter minimum nära startgissningen. Hur de tre villkoren implementeras diskuteras senare i avsnitt 4.2.

3.5 Identifierbarhet

Vid parameteruppskattning av en ODE-modell som den i ekvation (4) är en viktig frågeställning hur bra, och med vilken precision, kan parametrarna bestämmas? Denna frågeställning är relevant, eftersom om parametrarna inte kan skattas med hög precision så är det en indikation på att även om en modell beskriver formen på datan bra så lämpar sig den inte för förutsägelser [44]. Ett sätt att besvara frågeställningen är att undersöka om en modell är numeriskt identifierbar (NI) [43]. Att en modell är NI innebär att modellparametrarna kan kvantifieras utifrån en verklig, begränsad, mängd variant data [43]. I fallet med denna studie betyder det att en ODE-modell av TOR är NI om parametrarna $k_1, k_2, \ldots, k_n$ kan bestämmas utifrån den tillgängliga mätdatan (se avsnitt 4.1) med god precision.

Praktiskt undersöks NI genom att beräkna parameterkovariansmatrisen COV($\mathbf{k}$) [43]. I fallet med en icke-linjär ODE-modell som den i ekvation (4) går det dock inte att beräkna COV($\mathbf{k}$), utan den uppskattas utifrån hessianapproximationen $\tilde{H}$ i ekvation (8). Uppskattningen är baserad på att $\tilde{H} \approx F(\mathbf{k})$, där F är Fisher information matrix [43]. Cramér-Rao sats ger sedan att F^{-1} utgör en lägre gräns till COV($\mathbf{k}$) enligt; $F^{-1}(\mathbf{k}) \leq \text{COV}(\mathbf{k})$ [43]. Sammantaget resulterar allt detta i att en lägre gräns för kovariansmatrisen, $\text{COV}_{min}(\mathbf{k})$, kan approximeras enligt:

$$\tilde{H}^{-1} \approx \mathrm{COV}_{min} = \begin{bmatrix} \sigma^2_{min}(k_1) & \mathrm{cov}_{min}(k_1, k_2) & \cdots & \mathrm{cov}_{min}(k_1, k_n) \\ \mathrm{cov}_{min}(k_2, k_1) & \sigma^2_{min}(k_2) & \cdots & \mathrm{cov}_{min}(k_2, k_n) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{cov}_{min}(k_n, k_1) & \mathrm{cov}_{min}(k_n, k_2) & \cdots & \sigma^2_{min}(k_n) \end{bmatrix} \tag{11}$$

Utifrån ekvation (11) kan NI undersökas genom att studera variansen och korrelationen hos parametrarna. Variansen för de skilda parametrarna erhålls från diagonalelementen i $\tilde{H}^{-1}$. Ett högt värde på variansen hos en parameter indikerar låg precision hos det skattade värdet, med andra ord är parametern inte NI [43]. Korrelationen mellan parametrarna, $\mathrm{CORR}(k_i, k_j)$, kan erhållas genom att normera varje element $\tilde{H}^{-1}_{ij}$ med produkten $\tilde{H}^{-1}_{ii} \times \tilde{H}^{-1}_{jj}$. Ett högt korrelationsvärde nära 1 mellan parametrar är en indikation på linjärt beroende mellan dessa, följden av detta är att om en parameter ändras lite så ändras de korrelerade parametrarna också [44]. En konsekvens av detta blir att en ändring i mätdatan kan resultera i en helt annorlunda parameteruppsättning, vilket indikerar att parametrarna inte skattas med god precision och därav är modellen inte NI [44].

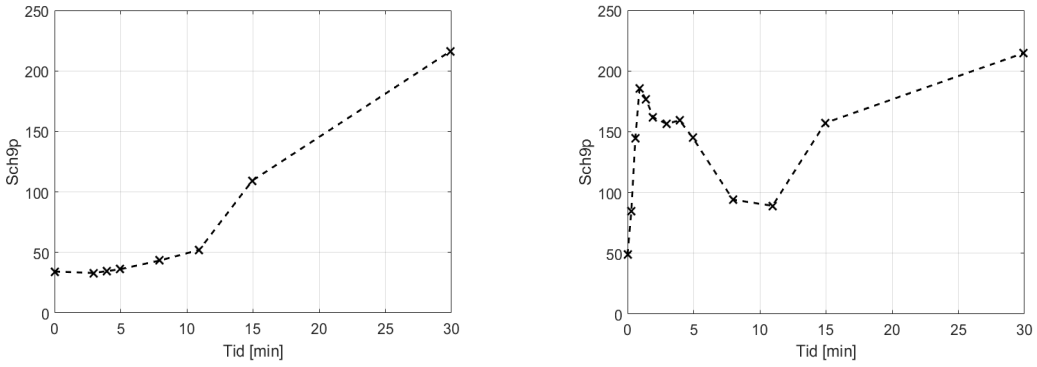
4 Metod

I detta avsnitt beskrivs hur arbetet har genomförts. Först beskrivs den data som behövdes under arbetet, samt hur denna samlades in. Därefter redovisas parameteruppskattningsprogrammet samt hur detta implementerades.

4.1 Datainhämtning

Lösning av ett ODE-system, som det i ekvation (4), förutsätter att initialvärden är kända för varje ODE. På grund av detta krävs datainhämtning för alla ingående komponenter i modellen. För att uppskatta initialkoncentrationer inhämtades data över den totala mängden av respektive komponent i cellen. Denna data hämtades från databasen *Saccharomyces* Genome Database. Uppskattningen av initialkoncentrationerna gjordes utifrån olika antaganden som varierade från komponent till komponent. Eftersom de inhämtade totalkoncentrationerna gäller för hela cellen gjordes det antaganden om hur stor andel av den totala koncentrationen som är tillgänglig för TOR och inte tar del andra reaktioner. Eftersom ett ODE-system av TOR, som det i ekvation (4), löses numeriskt krävs även startgissningar för alla parametervärden $k_1, k_2, \ldots, k_n$. Som riktmärke för vissa startgissningar användes parametervärden från en publicerad modell över mTOR av Dalle Pezze et al. [48] medan andra startgissningar gjordes godtyckligt i rimliga storleksordningar. Tillförseln av glutamin kunde bestämmas utifrån data från arbete av Stracka et al. [5]. Datan som erhölls var den intracellulära koncentrationsförändringen med avseende på koncentrationen vid tiden noll vid olika tidpunkter. Därpå gjordes en linjär anpassning för att bestämma en inflödeskonstant, se bilaga C.

För att parameteruppskattningsprogrammet ska kunna skatta alla parametervärden krävs det, som diskuterat i avsnitt 3.4, en uppmätt utsignalsvektor $\mathbf{y}$ med mätvärden i m tidpunkter. Utsignalen som används i denna studie är koncentrationen av fosforylerat Sch9, det vill säga aktivt Sch9. Eftersom modellen har ett fokus på reglering av TORC1 i respons till aminosyran glutamin så behöver tidserien över aktivt Sch9 komma från experiment med samma fokus. Efter omfattande litteraturstudier kunde lämpliga tidserier extraheras ur arbete av Stracka et al. [5], se figur 2. En tidserie erhölls för andelen fosforylerat Sch9 för wild type och bortkopplat EGO-komplex respektive. Dessa tidserier tillät en uppdelning av modellen i två moduler som diskuteras närmare i avsnitt 5.1. Tidserier extraherades ur presenterade grafer i Stracka et al. [5] med hjälp av det web-baserade grafverktyget WebPlotDigitizer. I figur 2a visas en tidserie för fosforylering av Sch9 då EGO-komplexet är bortkopplat och i figur 2b visas en tidserie för fosforylering av Sch9 i wild type jäst.



(a) *Tidserie 1* **(b)** *Tidserie 2*

Figur 2: *De experimentella mätdataserierna som modul 1 (a) och modul 2 (b) konstruerades för att återspegla. Tidserie 1 visar halten av fosforylerat Sch9 då EGO-komplexet var bortkopplat och tidserie 2 visar halten av fosforylerat Sch9 för wild type.*

4.2 Implementering av nedåtgående algoritm

För att kunna anpassa en ODE-modell av TOR efter verkligheten skrevs ett parameteruppskattningsprogram i MATLAB. Koden, tillsammans med en förklaring över funktion och samverkan hos de olika filerna och funktionerna i programmet, redovisas i bilaga D. Huvudfunktionen för programmet är att via en nedåtgående algoritm, beskriven i avsnitt 3.4, bestämma den parameteruppsättning som minimerar kostnadsfunktionen $f(\mathbf{k})$ i ekvation (6). Vid sidan av detta beräknar programmet också kovarians- och korrelationsmatrisen i syfte att avgöra identifierbarheten hos en modell. I algoritm 1 presenteras ett flödesschema över huvudfunktionsfilen *MQNR* (Modified quasi-Newton Raphson) som utför själva minimeringen av ekvation (6). I de kommande styckena kommer utvalda delar av algoritm 1 presenteras närmare.

Algoritm 1 behöver flertalet inmatningar, se rad 2, för att beräkna den parameteruppsättning $\mathbf{k} \in \mathbb{R}^n$ som minimerar $f(\mathbf{k})$. Den första inmatningen är en startpunkt i parameterrummet $\mathbf{k}_{in}$, varifrån algoritmen sedan bestämmer en sökriktning $\mathbf{p}$ och börjar stega mot ett minimum. Den andra inmatningen, I_{0c}, är en vektor innehållande alla begynnelsevärden för komponenterna i den modell som parameteruppskattas. Dessa krävs för att lösa systemet av ODE:er för den aktuella modellen vilket görs med ODE45 i MATLAB. Utifrån lösningen beräknas sedan den simulerade utsginalen $\hat{y}(\mathbf{k}, t)$. De två sista inmatningarna $\mathbf{y}_m$ och $\mathbf{t}_m$ är vektorer innehållande mätdatan respektive vilka tidpunkter mätdatan hämtades vid. Båda dessa krävs för att kunna beräkna residualerna $r_i = y_m(t_i) - \hat{y}(\mathbf{k}, t_i)$, vilka sedan krävs för att beräkna $f(\mathbf{k})$, $\nabla f(\mathbf{k})$ och $\tilde{H}$. Anledningen till att tidsvektorn $\mathbf{t}_m$ krävs vid beräkning av residualerna är att värdet på den simulerade utsignalen $\hat{y}$ behövs i tidpunkter som mätdatan finns tillgänglig vid. ODE-lösare i MATLAB löser dock inte ODE:er för specifika tidpunkter och därav används $\mathbf{t}_m$ för att interpolera fram ett värde på $\hat{y}$ i tidpunkterna mätdatan finns tillgänglig vid.

Totalt har algoritm 1 två villkor som resulterar i att sökningen efter minimum avbryts. Den första brytpunkten är via for-loopen i rad 4, där sökningen termineras och ett meddelande skrivs ut om mer än 100 iterationer utförs. Syftet med att låta programmet arbeta under for-loop, vars längd kan modifieras, grundar sig i att en parameteruppskattning inte ska ta orimligt lång tid, vilket annars skulle kunna hända vid dåligt val av startgissning. Det andra villkoret som avbryter sökningen är att minst två av de tre termineringsvillkoren presenterade i avsnitt 3.4 uppfylls. Anledningen till att minst två villkor måste uppfyllas beror på att uppfyllandet av bara ett villkor inte garanterar ett minimum, medan två villkor är en mycket starkare indikation på att sökningen är nära ett minimum [46].

Förutom två brytpunkter har programmet två metoder för att bestämma $\mathbf{p}$. Från rad 13 i algoritm 1 framgår det att om om $\tilde{H}$ är positivt definit och att konditionstalet $\kappa(\tilde{H})$ understiger en viss tolerans används quasi-Newtons metod. Uppfylls inte dessa garanteras inte att $\mathbf{p}$ är nedåtgående, och

därmed används istället steepest descent, som är långsammare men mer robust [43]. Oavsett val av sökriktning behöver $A(\mathbf{k})$ i ekvation (9) beräknas, vilket görs genom att utvärdera i differenskvoten ekvation (10) för alla parametrar och tidpunkter.

Algoritm 1 Parameteruppskattning av $\mathbf{k}$ för TOR

1: **Udata:**
   $\mathbf{k} \in \mathbb{R}^n$
2: **Indata:**
   $\mathbf{k_{in}} \in \mathbb{R}^n$, $\mathbf{y}_m \in \mathbb{R}^m$, $\mathbf{t}_m \in \mathbb{R}^m$, $I_{0c} \in \mathbb{R}^g$ och $\mathbf{t}_{int} \in \mathbb{R}^2$
3: **Initialisera:**
   $\mathbf{k} \leftarrow \mathbf{k}_{in}$, $N_{it} \leftarrow v$
4: **for** $v = 0 : 1 : 100$ **do**
5: *Modelvärden och residualer*: Beräkna modellvärdena $\hat{y}_m \in \mathbb{R}^m$ och residualvektorn $r(\mathbf{k}) \in \mathbb{R}^m$
6: *Känslighetsmatris*: Beräkna känslighetsmatrisen $A(\mathbf{k}) \in \mathbb{R}^{m \times n}$.
7: *Beräkna gradient och kostnadsfunktion*: Beräkna $f(\mathbf{k}) \in \mathbb{R}_+$ och $\nabla f(\mathbf{k}) \in \mathbb{R}^n$ enligt:

$$\nabla f(\mathbf{k}) = -2A(\mathbf{k})^T W r(\mathbf{k})$$

.
8: **if** Minst två av tre termineringsvilkor uppfylls **then**
9: Break
10: **end if**
11: *Hessianapproximation*: Beräkna hessianapproximationen $\tilde{H} \in \mathbb{R}^{n \times n}$ enligt:

$$\tilde{H}(\mathbf{k}) = 2A(\mathbf{k})^T W A(\mathbf{k})$$

12: *Sökriktning*: Beräkna sökriktningen $\mathbf{p} \in \mathbb{R}^n$ genom:
13: **if** $H(\mathbf{k}) \succ 0$ and $\kappa(\tilde{H}) < \epsilon$ **then**
14: Quasi-Newtons metod: $A(\mathbf{k})WA(\mathbf{k})^T\mathbf{p} = A(\mathbf{k})Wr(\mathbf{k})$
15: **else**
16: Steepest descent: $\mathbf{p} = -\nabla f(\mathbf{k})$
17: **end if**
18: *Steglängd*: Definierar intialt stegländen $\alpha \in \mathbb{R}_+$ som $\alpha \leftarrow 2$.
19: *Uppdatera parameter*: Sätt $\mathbf{k}_{it} \leftarrow \mathbf{k}$ och $d \leftarrow 1$ och uppdatera $\mathbf{k}$ enligt:
20: **while** $d > 0$ **do**

$$\alpha \leftarrow \frac{\alpha}{2}$$

$$\mathbf{k} \leftarrow \mathbf{k}_{it} + \alpha\mathbf{p}$$

$$d \leftarrow f(\mathbf{k}) - f(\mathbf{k}_{it})$$

21: **end while**
22: **end for**
return k

Val av stegländ α för en sökriktning $\mathbf{p}$ sker via en enkel procedur som presenteras i rad 18 till 21 i algoritm 1. Processen är sådan att först definieras $\alpha = 1$, med detta α beräknas därefter differensen $d = f(\mathbf{k_N} + \alpha_N\mathbf{p}) - f(\mathbf{k_N})$. Är d negativ, vilket indikerar att det nya kostnadsfunktionsvärdet är mindre än det gamla, väljs α som steglängd. Blir d däremot positiv väljs ett nytt α enligt $\alpha = \alpha/2$ och processen upprepas. Denna process med att successivt halvera α pågår till dess att ett negativt d erhålls. Den implementerade metoden i programmet är simpel, och det finns andra mer avancerade för val steglängd [46], men i denna studie fungerar denna metod effektivt och används därför.

5 Resultat

Utifrån den tillgängliga informationen i teorin skapades en modell av hur TORC1 reglerar protein- och ribosomsyntes i cellen. Modellen delades upp i mindre delar, kallade moduler, för vilka hastighetskonstanterna skattades via parameterskattningsprogrammet. I det kommande avsnittet redovisas den modell som konstruerats och dess moduler, samt resultatet från parameteruppskattningen.

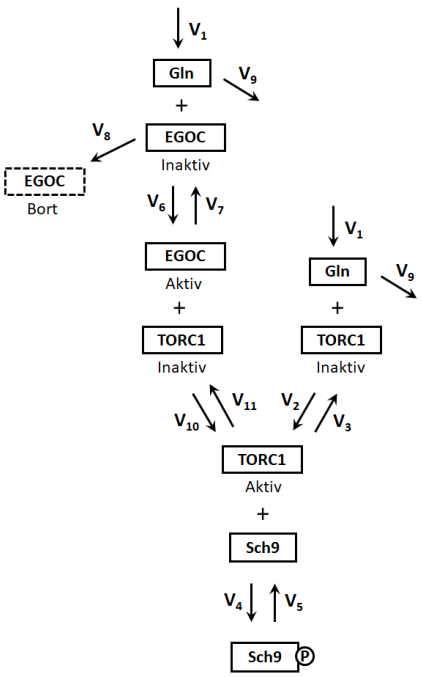
5.1 Föreslagen modell

Utifrån syftet att undersöka hur protein- och ribosomsyntes regleras av TORC1 vid en insignal av glutamin (Gln) konstrueras modellen som visas i figur 3 nedan. Modellen ställs upp utifrån vad som tidigare är känt om signalvägen, vilket presenteras i avsnitt 3.2. I den föreslagna modellen regleras aktiviteten av TORC1 av två skilda mekanismer, vilka båda aktiveras av glutamin. Den ena är direktaktivering av glutamin och den andra är aktivering via aktivt EGOC. Aktiviteten av EGOC uppregleras av glutamin och nedregleras av aktivt Sch9 (Sch9p). Detta resulterar i en feedbackinhibering av EGOC beroende på halten av Sch9p, där det inhiberade EGOC lämnar reaktionsvägen.

Glutaminhalten regleras av en tillflödes- respektive bortflödeshastighet. Bortflödestermen beror av den totala halten glutamin i cellen och beskriver det glutamin som konsumeras i andra cellulära processer. Tillflödet av glutamin beskrivs endast av en konstant. Detta baseras på den data som redovisas i bilaga C, där det kan utläsas att den intracellulära koncentrationen av glutamin ökar linjärt under de första nio minuterna. Nedströms av TORC1 ligger fokus endast på aktivering av Sch9 som sker via TORC1a. Detta då Sch9 är den komponent i signalvägen som reglerar cellfunktionerna av intresse.

$$
\begin{aligned}
x_1 &= [Gln] & v_1 &= k_1 \\
x_2 &= [TORC1]\ \text{Inaktiv} & v_2 &= k_2 x_1 x_2 \\
x_3 &= [TORC1]\ \text{Aktiv} & v_3 &= k_3 x_3 \\
x_4 &= [Sch9] & v_4 &= k_4 x_3 x_4 \\
x_5 &= [Sch9\text{p}] & v_5 &= k_5 x_5 \\
x_6 &= [EGOC]\ \text{Inaktiv} & v_6 &= k_6 x_1 x_6 \\
x_7 &= [EGOC]\ \text{Aktiv} & v_7 &= k_7 x_7 \\
x_8 &= [EGOC]\ \text{Bort} & v_8 &= k_8 x_5 \\
 & & v_9 &= k_9 x_1 \\
 & & v_{10} &= k_{10} x_7 x_2 \\
 & & v_{11} &= k_{11} x_3
\end{aligned}
$$

$$
\begin{aligned}
\frac{\mathrm{d}x_1}{\mathrm{d}t} &= v_1 - v_9 \\
\frac{\mathrm{d}x_2}{\mathrm{d}t} &= v_3 - v_2 - v_{10} + v_{11} \\
\frac{\mathrm{d}x_3}{\mathrm{d}t} &= v_2 - v_3 + v_{10} - v_{11} \\
\frac{\mathrm{d}x_4}{\mathrm{d}t} &= v_5 - v_4 \\
\frac{\mathrm{d}x_5}{\mathrm{d}t} &= v_4 - v_5 \\
\frac{\mathrm{d}x_6}{\mathrm{d}t} &= v_7 - v_6 - v_8 \\
\frac{\mathrm{d}x_7}{\mathrm{d}t} &= v_6 - v_7 \\
\frac{\mathrm{d}x_8}{\mathrm{d}t} &= v_8
\end{aligned}
$$



Figur 3: *Den föreslagna modellen med tillhörande ODE:er och hastighetsuttryck. Totalt består modellen av 8 komponenter vars ODE:er ges i nedre vänstra hörnet. Utöver detta har den föreslagna modellen* 11 *okända hastighetskonstanter som behöver skattas.*

Den tillgängliga halten av ett protein modelleras genom att alla former av proteinet inkluderas i modellen. I detta fall innebär det främst att både den aktiva och den inaktiva formen av proteinet modelleras. För alla reaktioner finns en framåt- och en bakåthastighet som motsvarar både bildning och sönderfall av komponenterna. Genom att inkludera detta erhålls en modell med 8 komponenter, 11 hastigheter och 11 hastighetskonstanter vilket illustreras i figur 3. I figuren är även hastighetsuttryck och ODE:er uppställda.

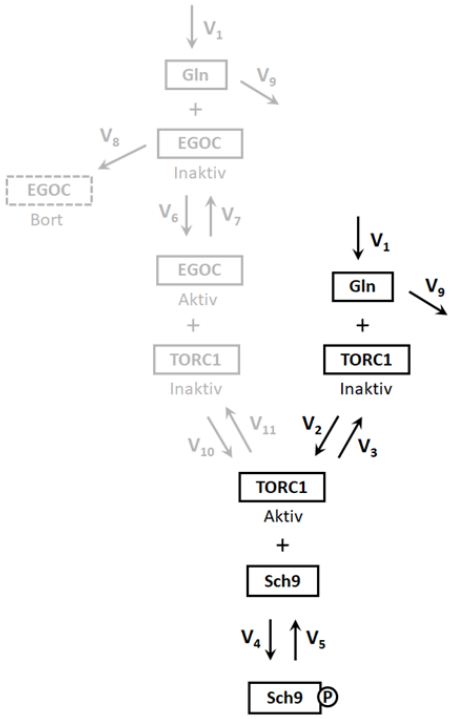
I avsnitt 3.3.3 togs det upp att antalet okända hastighetskonstanter är en begränsande faktor vid parameteruppskattning. Modellens 11 hastighetskonstanter överskrider det antal som kan skattas med god numerisk identifierbarhet. Problemet kan dock kringgås genom att modellen delas upp i separata moduler, vilka är mindre delar av en stor modell för vilka parametrarna skattas separat. Genom att dela upp en större modell i moduler blir antalet parametrar som skattas för respektive modul färre och de kan därmed bestämmas med högre säkerhet. De skattade parametrarna fixeras sedan när modulerna sammankopplas till den totala modellen. Datan i figur 2a respektive 2b tillåter uppdelning i två moduler. Den första av dessa beskriver endast den direkta aktiveringen av TORC1 via glutamin, och den andra beskriver hur TORC1 aktiveras både via EGOC och direkt av glutamin. De parametrar som skattas för den första modulen fixeras vid skattning av övriga parametervärden i den andra modulen.

5.2 Första modulen

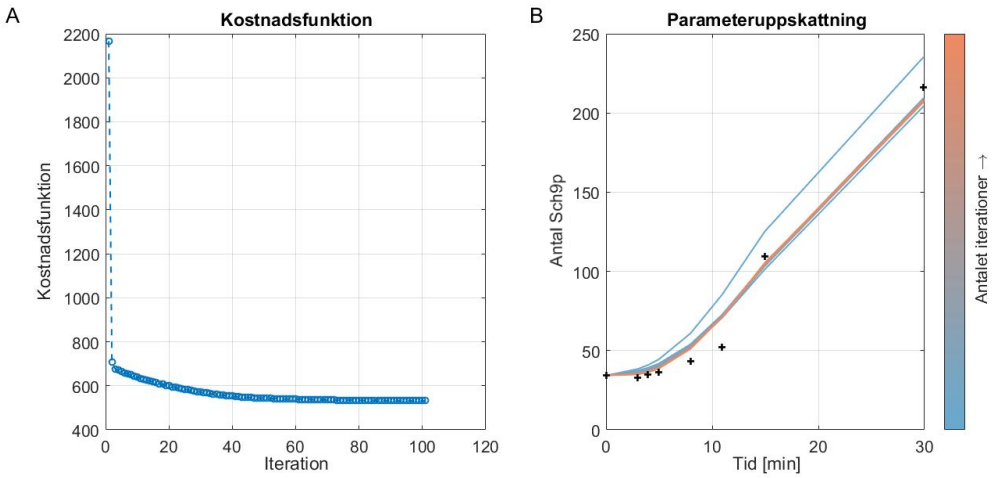
Den första modulen, illustrerad i figur 4, inkluderar endast den direkta aktiveringen av TORC1 beroende av glutamininfödet. Det innebär alltså att den korresponderande datan togs fram i *S.cerevisiae* med inaktiverat EGOC. Detta resulterar i att den modellerade modulen endast inkluderar 5 ämnen, 6 reaktioner och 6 hastighetskonstanter vilka är uppställda i bilaga B.

Via *MQNR* (parameteruppskattningsprogrammet) uppskattades parametervärden för hastighetskonstanterna i första modulen. Datan som användes vid kalibreringen kommer från mätningar i bagerijäst med inaktiverat EGOC och presenteras i avsnitt 4.1. Resultatet av uppskattningen redovisas i figur 5 i form av två grafer, där figur 5A visar kostnadsfunktionen $f(\mathbf{k})$ över antalet iterationer i *MQNR* och figur 5B visar hur den simulerade utsignalen $\hat{y}(t, \mathbf{k})$, som beskriver nivån av fosforylerat Sch9, anpassar sig efter mätdatan. Mer specifikt visar figur 5B hur $\hat{y}(t, \mathbf{k})$ över iterationerna i *MQNR*, när linjerna från blått till orange, anpassar sig relativt väl efter mätdatan. Följaktligen illustrerar detta resultat att ODE-modellen av den första modulen kan fånga det grundläggande utseendet hos mätdatan. Anpassningen är dock inte helt perfekt, utan den modellerade kurvan avviker fortfarande en del från mätdatan, vilket ytterligare illustreras av kostnadsfunktionen i figur 5A.

I syfte att undersöka identifierbarheten beräknades kovarians- och korrelationsmatrisen för de skattade parametrarna. Utifrån kovariansmatrisen beräknades därefter variationskoefficenterna ($cv(k_i) = \sigma^2/k_i$), vars värden tillsammans med parametervärdena redovisas i tabell 2. Variationskoefficienterna,



Figur 4: *Överblick av den första modulen. Pilar är utmålade för respektive reaktion och är märkta med motsvarande reaktionshastighet. De borttonade pilarna representerar reaktioner som beror av EGO-komplexet och tas därav inte med i modul 1.*



Figur 5: *A) Värdet på kostnadsfunktionen över antalet iterationer i MQNR för modul 1. B) Anpassning av simulerade utsignalen $\hat{y}(\mathbf{k}, t)$, i enhet antal foforylerade Sch9, efter mätdatan (+) för modul 1. Grafen visar hur $\hat{y}(\mathbf{k}, t)$ med ökande antal iterationer i MQNR, när färgen går från blått till orange, successivt anpassar sig bättre efter mätdatan.*

vilka är varianserna skalade med parametervärdena, analyseras istället för varianserna eftersom det lättare möjliggör analys av parametrar med skild storleksordning. För variationskoefficenter gäller att om $cv(k_i) > 1$ är parametern inte identifierbar [43]. Med detta i åtanke visar tabell 2 tydligt att variansen är väldigt hög för parametrarna k_2, k_3, k_4, k_5, k_6. Den höga variansen är en indikation på att även om parametrarna bra beskriver formen hos datan, vilket visas i figur 5B, är osäkerheten hög och därmed noggrannheten låg hos de skattade parametrarna. Denna indikation syns också i korrelationsmatrisen som har följande utseende:

$$\mathrm{COR}(\mathbf{k}) = \begin{bmatrix} 1 & 0.24 & 0.41 & -0.37 & -0.37 & 0.31 \\ 0.24 & 1 & 0.98 & -0.99 & -0.99 & 1.00 \\ 0.41 & 0.98 & 1 & -1.00 & -1.00 & 0.99 \\ -0.37 & -0.99 & -1.00 & 1 & 1.00 & -1.00 \\ -0.37 & -0.99 & -1.00 & 1.00 & 1 & -1.00 \\ 0.31 & 0.99 & 0.99 & -1.00 & -1.00 & 1 \end{bmatrix} \tag{12}$$

Korrelationsmatrisen visar på hög korrelationen mellan samtliga parametrar då nästan samtliga värden i matrisen är nära 1 eller -1. Sammantaget blir därför resultatet från identifierbarheten att parametrarna inte har kunnat bestämmas med hög noggrannhet och att modellen inte är numeriskt identifierbar.

Tabell 2: *Parametervärden och variationskoefficienter för första modulen.*

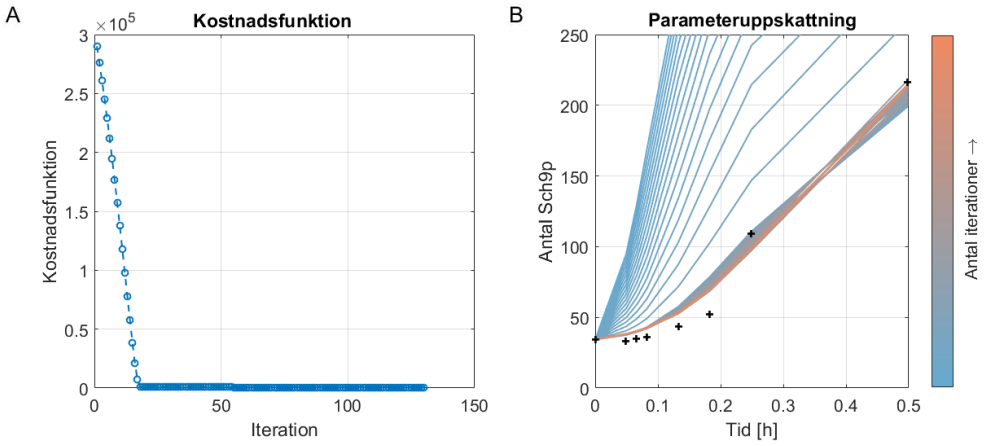
k_1	k_2	k_3	k_4	k_5	k_6
439	1.04×10^{-5}	8.63×10^{-3}	4.71×10^{-4}	10.0	1.16×10^{2}
$cv(k_1)$	$cv(k_2)$	$cv(k_3)$	$cv(k_4)$	$cv(k_5)$	$cv(k_6)$
4.8×10^{-3}	1.21×10^{6}	8.0×10^{4}	4.2×10^{4}	2.8×10^{4}	1.6×10^{2}

5.3 Alternativ modell av modul 1

Med målet att undvika problemet med identifierbarheten skapades en alternativ modell av modul 1. Ändringarna som gjordes var att tidskalan skalades om från minuter till timmar, att en kvadrat lades till på deaktiveringen av TORC1 och att antalet okända parametrar reducerades. Mer specifikt reducerades parametrarna genom att hastighetskonstanterna för v_2 och v_3 i figur 3 sattes till samma

värde och att hastighetskonstanten för v_1 (k_1) fixerades. Fixeringen av hastighetskonstanten k_1, som beskriver inflödet av glutamin, genomfördes utifrån en linjär regression av den intracellulära glutaminhalten [5], se bilaga C. Kvadrattermen lades till på v_3 sådant sätt att den nya hastigheten blev $v_3 = k_3 x_3^2$, se bilaga B för fullständig modell. Motiveringen till ändringarna av modul 1 diskuteras i avsnitt 6.

Resultatet av anpassningen efter mätdatan via *MQNR* för den alternativa modellen av modul 1 redovisas i figur 6. Figur 6B visar tydligt att den parameterreducerande modellen anpassar sig bra efter datan och att anpassningen är väldigt lik den för modul 1 i figur 4. Detta återspeglas i figur 6A där kostnadsfunktionsvärdet $f(\mathbf{k})$ för när ett slutligt värdet på cirka 630. Detta är lite högre än i figur 5A där det slutliga värdet är cirka 570, men det är ändå en relativt liten skillnad med tanke på skalningen av y-axeln. Resultatet indikerar därför att den alternativa modellen beskriver datan nästintill likvärdigt med den icke modifierade modellen.



Figur 6: *A) Värdet på kostnadsfunktionen över antalet iterationer i MQNR för alternativa modellen av modul 1. B) Anpassning av simulerade utsignalen $\hat{y}(\mathbf{k}, t)$, i enhet antal fosforylerade Sch9, efter mätdatan (+) för alternativa modellen av modul 1. Grafen visar hur $\hat{y}(\mathbf{k}, t)$ med ökande antal iterationer i MQNR, när färgen går från blått till orange, likt modul 1 anpassar sig bra efter mätdatan.*

Identifierbarheten hos den alternativa modellen undersöktes genom att beräkna korrelationsmatrisen och variationskoefficienterna. Variationskoefficienterna tillsammans med de optimerade parametervärdena redovisas i tabell 3, och det observeras att överlag är varianserna lägre och därmed bättre än de i tabell 2. Även om variansen är lägre i denna modell är den fortfarande hög för k_1, k_2 och k_4. Detta indikerar att den reducerade modellen inte heller är numeriskt identifierbar. Detta stärks ytterligare av korrelationsmatrisen som har följande utseende:

$$\mathrm{COR}(\mathbf{k}) = \begin{bmatrix} 1 & -1.00 & -1.00 & -0.98 \\ -1.00 & 1 & 1.00 & -0.98 \\ -1.00 & 1.00 & 1 & -0.99 \\ -0.98 & -0.98 & -0.99 & 1 \end{bmatrix} \tag{13}$$

Korrelationsmatrisen visar att det råder hög korrelation eftersom korrelationen är större än 0.95 för samtliga parametrar. Sammantaget blir därför resultatet att även om alternativa modellen är mer identifierbar än modul 1 så är den ändå inte numeriskt identifierbar.

Tabell 3: *Parametervärden och variationskoefficienter för alternativa modellen av modul 1.*

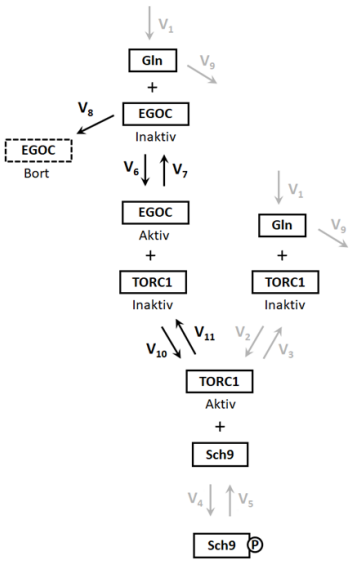
k_1	k_2	k_3	k_4
5.7×10^{-3}	2.1×10^{-2}	2.8×10^{-1}	2.2×10^{-1}
$cv(k_1)$	$cv(k_2)$	$cv(k_3)$	$cv(k_4)$
131	39.3	0.28	57.1

5.4 Andra modulen

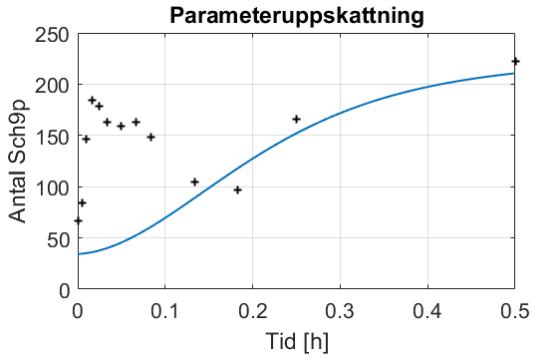
Datan som används för skattning av parametrar i modul 2, visad i figur 7, kommer från mätningar i bagerijäst med aktivt EGOC och finns tillgänglig i bilaga C. På grund av detta inkluderar den andra modulen båda aktiveringar, men med konstanta parametervärden för alla parametrar som skattats för den första modulen. De okända parametrarna i modul 2 kan alltså endast skattas om ett resultat finns för parametrarna i modul 1. Är detta resultat felaktigt påverkar det även resultatet av parameteruppskattningen för modul 2, då de skattade hastighetskonstanterna för modul 1 antas vara de sanna värdena för de ingående reaktionerna.

Med de skattade parametervärdena från modul 1 fixerade lämnar det totalt 6 hastighetskonstanter som ska skattas av parameteruppskattningsalgoritmen. Totalt i modellen ingår 7 olika ämnen och 11 hastigheter. ODE:er och hastighetsuttryck för dessa presenteras i bilaga B.

Anpassning av modul 2 till datan via *MQNR* misslyckades. Resultatet av anpassningen redovisas i figur 8, där det tydligt observeras att den simulerade utsignalen $\hat{y}(\mathbf{k}, t)$ inte fångar utseendet hos mätdatan. Problemet ligger i att modellen inte kan fånga den initiala upp- och nedgången av fosforylerat Sch9, utan enligt modellen är halten monotont ökande vilket är en indikation på modellen är felkonstruerad. På grund av allt detta genomfördes ingen identifierbarhetsanalys, eftersom det förutsätter en modell som följer mätdatan.



Figur 7: *Överblick av den andra modulen. Pilar är utmålade för respektive reaktion och är märkta med motsvarande reaktionshastighet. Endast hastighetskonstanter för helfyllda pilar skattas eftersom konstanterna för de andra reaktionerna skattades i modul 1.*



Figur 8: *Anpassning av simulerade utsignalen $\hat{y}(\mathbf{k}, t)$, i enhet antal foforylerade Sch9, efter mätdatan (+) för modul 2. Grafen visar hur den simulerade utsignalen misslyckas med att anpassa sig efter mätdatan.*

6 Diskussion

Målet med att skapa en modell som beskriver hur glutamin påverkar halten av fosforylerat Sch9 uppnåddes endast delvis. I scenariot med inaktiverat EGOC lyckades modul 1 fånga utseendet hos mätdatan, se figur 5, dock är modellen inte numeriskt identifierbar. I fallet med normal aktivitet hos EGOC misslyckades modul 2 med att beskriva mätdatan, se figur 8, vilket är en indikation på att modellen är felkonstruerad. I de kommande styckena kommer möjliga förklaringar till problemen med identifierbarheten och modul 2 diskuteras i syfte att se hur modellerna och/eller parameteruppskattningsprogrammet skulle kunna modifieras för att erhålla ett bättre resultat.

Som tidigare nämnt lyckas modul 1, illustrerad i figur 4, fånga utseendet hos mätdatan. Detta indikerar att förloppet i modul 1 är möjligt att modellera via det använda tillvägagångssättet. Dock är modellen inte numeriskt identifierbar (NI) eftersom hög varians och korrelation råder bland parametrarna. Konsekvensen av detta blir att parametrarna beskriver datan bra, men saknar fysikalisk tolkning eftersom en liten ändring i exempelvis mätdatan skulle kunna resultera i att en helt ny parameteruppsättning erhålls [44]. Orsaken till identifierbarhetsproblem kan vara flera, men ett möjligt problem är att modul 1 är överflödig och innehåller för många parametrar. Denna överparametrisering skulle kunna förklara den höga korrelationen i ekvation (12) [44]. En annan möjlig bidragande faktor är problemet med storleksordningen hos parametrarna i och med att vissa parametrar är väldigt små, 10^{-5}, medan k_1 är i storleksordningen 10^2. Storleksskillnaden resulterar i högt $\kappa(\tilde{H})$ vilket gör att numeriska problem kan påverka beräkningen av COV($\mathbf{k}$) i ekvation (10). Med målet att åtgärda dessa två möjliga orsaker skapades därför den alternativa modellen av modul 1.

Idén med den alternativa modellen av modul 1 var att reducera antalet parametar samt att skala dessa. Antalet parametrar reducerades utifrån antagandet att hastighetskonstanterna för framåt- och bakåtreaktionen mellan inaktivt TORC1 och aktivt TORC1 i figur 4 var ungefär lika stora. Ytterligare fixerades k_1 som beskriver glutamininflödet utifrån regressionen i bilaga C. Eftersom k_1 var mycket större än övriga parametrar bidrog detta också till att reducera problemet med skilda storleksordningar. De små värdena på hastighetskonstanterna åtgärdades också genom att skala om tidsaxeln från minuter till timmar. Då hastighetskonstanterna alltid är i enheten min^{-1} medförde detta en skalning av varje parameter med en faktor 60. Tidsaxeln skalades inte mer eftersom då skulle tidsintervallet för vilket ODE:erna löstes i blivit väldigt litet. Slutligen för att få en bra anpassning efter data lades en kvadrat till på sönderfallet av aktivt TORC1 sådant att hastigheten v_3 i figur 4 fick utseendet $v_3 = k_3 x_3^2$. Ingen direkt motivering till varför en kvadratterm bidrar med bättre anpassning hittades i litteraturen. En möjlig anledning till att det ändå fungerade är att nedbrytningen av aktivt TORC1 nödvändigtvis inte är en första ordningens reaktion, utan möjligtvis ett andra ordningens förlopp där ett annat protein deaktiverar TORC1. Genom att lägga till en kvadratisk term kan därmed detta andra ordningens förlopp ha lyckats fångats.

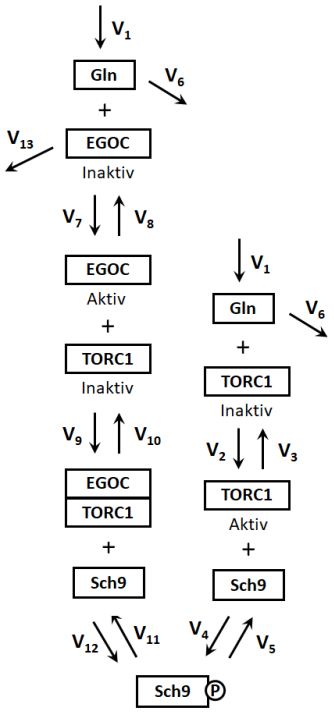
Figur 6 visar att de diskuterade begränsningarna och antaganden på parametrarna för alternativa modul 1 inte märkbart påverkade resultatet. De lägre varianserna i tabell 3 jämfört med tabell 2 indikerar också att den första modellen var överparametriserad. Trots detta är den alternativa modellen ändå inte numeriskt identifierbar och hög korrelation råder fortfarande. Utifrån detta försöktes hastighetskonstanterna för v_5 och v_6 slås samman, konsekvensen blev dock att modellen inte kunde fånga mätdatan. Allt detta pekar på att problemet inte enbart ligger i antalet parametrar. Problemet kan också ligga i mängden av tillgänglig mätdata. I nuläget finns endast mätdata tillgänglig för en modellkomponent vid 12 tidpunkter. Mätdata vid fler tidpunkter, eller flera utsignaler, skulle förmodligen kunna förbättra den skattade variansen hos parametrarna.

Gällande modul 2 illustrerad i figur 7 lyckades modellen inte beskriva mätdatan. Mer specifikt misslyckades modellen med att fånga den kraftiga upp- och nedgången för de första mätpunkterna vilket visas i figur 8. Den kraftiga uppgången kommer troligen från en hyperaktivering av TORC1 medan nedgången sannolikt beror på en snabb deaktivering av TORC1. Att modellen inte kunde fånga dessa förlopp indikerar att den är felkonstruerad.

En trolig anledning till att den stora modellen inte kunde fånga den kraftiga uppgången av fosforylerat Sch9 var de fixerade parametervärdena i modul 1. För att fånga utseendet hos mätdatan i

modul 1 krävs låga värden på parametrarna nedströms om TORC1. En konsekvens av detta blev att när båda modulerna implementerades var det omöjligt att fånga den snabba uppgång som sker initialt. Om modulerna istället var oberoende av varandra, och möttes först vid Sch9, skulle detta problem kunna åtgärdas och utifrån denna idé skapades ytterligare en alternativ modell som visas i figur 9. I den alternativa modellen är majoriteten av reaktionerna initialt oberoende av modul 1. Följaktligen kan hastighetskonstanterna i modul 2 skattas för att fånga den initiala aktiveringen och deaktivieringen av Sch9, utan att påverka den övriga modellen. Ytterligare är denna modell också biologiskt möjlig, då många detaljer om reaktionerna som sker för att aktivera TORC1 är okända.

En annan trolig anledning till att den stora modellen inte kunde fånga mätdatan var problem med modellering av feedback-regleringen. Feedback-regleringen av EGOC är sannolikt ansvarig för den kraftiga nedgången i mätdatan som sker efter den initiala aktiveringen och processen modellerades som ett bortflöde av EGOC beroende av mängden fosforylerat Sch9. Tanken med detta var att när koncentrationen av fosforylerat Sch9 når en viss nivå förblir EGOC inhiberat och fortsatt aktivitet regleras endast av modul 1. I teorin bör detta kunna beskriva mätdatan och är även möjligt att motivera ur ett biologiskt perspektiv. Resultatet från många olika parameteruppsättningar visar dock att något inte stämde med denna hypotes och således vore det intressant att undersöka om deaktiveringen styrs av andra ämnen som exempelvis TORC1.



Figur 9: *Överblick av den föreslagna modifieringen av den totala modellen. Skillnaden mot modellen i figur 3 är att de två modulerna möts vid Sch9 istället för TORC1.*

Antaganden som gjordes vid uppställning av ODE:erna kan också ha påverkat resultatet. Vid konstruktion av samtliga modeller gjordes antagandet att alla bakåtreaktioner kunde beskrivas som första ordningens sönderfall. En konsekvens av detta kan vara att de modellerade bakåtreaktionerna inte kunde ske tillräckligt fort i jämförelse med framåtreaktionerna, som är av andra ordningen, vid högre koncentrationer. En indikation på att kvadratiska termer kan hjälpa till att fånga dynamiken hittas i den alternativa modellen av modul 1 som diskuterades i avsnitt 5.3, där en kvadratterm var nödvändig. Ur ett biologiskt perspektiv skulle tillägg av kvadrattermer kunna förklaras genom att en eller flera ytterligare komponenter är involverade i deaktiveringen, ett exempel är defosforylering av Sch9. Denna typ av reaktion sker generellt sett inte spontant, utan det krävs att ett annat protein är närvarande för att reaktionen ska inträffa. Då den defosforylerande komponenten är okänd skulle detta kunna modelleras i form av en kvadratur i bakåtreaktionen. Sammantaget skulle en, eller en kombination av de föreslagna åtgärderna vara intressanta att implementera för att förbättra modellen.

Hade den stora modellen framgångsrikt modifierats indikerar resultatet från modul 1 att problem med identifierbarhet hade uppstått. Detta belyser att det vore intressant att *a priori* undersöka om modellerna är strukturellt identifierbara. Att en modell är strukturellt identifierbar (SI) innebär att parametrarna unikt kan bestämmas från oändligt mycket icke-variant data [43, 49]. Ytterligare gäller att om en modell inte är SI är den inte NI och därmed är det meningslöst att ens försöka bestämma de okända modellparametrarna [43]. Med tanke på hur mycket tid som kan sparas genom att slippa försöka parameteruppskatta en modell som inte är SI borde därav den strukturella identifierbarheten ha undersökts hos samtliga presenterade modeller. För att undersöka om en modell är SI går det bland annat att använda tillgängliga tilläggspaket i MATLAB och Mathematica

[49, 50].

Ett annat problem relaterat till parametrarna hos modellerna var de skilda parameterstorlekarna. Problemet med detta är att *MQNR:s* effektivitet påverkas negativt eftersom quasi-Newtons metod i ekvation (7) inte kan användas och att väldigt små steglängder α måste tas i parameterrummet. Anledningen till att quasi-Newtons metod inte kan användas grundar sig i att stora parameterskillnader resulterar i ett stort $\kappa(\tilde{H})$ vilket innebär att metoden nödvändigtvis inte genererar en nedåtgående riktning $\mathbf{p}$. De små steglängderna α följer från att i båda modellerna av modul 1 är vissa av parametrarna i storleksordningen 10^{-3} medan andra är större med en faktor 100 eller mer. Konsekvensen av detta blir att om $\alpha = 1$ kommer stegningen i parameterrummet $\mathbf{k} + \alpha\mathbf{p}$, där $\mathbf{p}$ är en enhetsvektor, resultera i en steglängd av ordningen 10^{-1} i samtliga parametrar. En steglängd av ordning 10^{-1} i en parameter av storlek 10^{-3} resulterar i en ny parameter ungefär 100 gånger större än den gamla, en väldigt stor skillnad som ofta resulterar i att ODE45 inte ens kan lösa systemet av ODE:er som utgör modellen. I syfte att kringgå detta problem måste därför α sättas till 10^{-3}, vilket dock resulterar i att väldigt små steglängder i parametrarna av ordning 10^{1} tas och därmed blir svårt att optimera de parametrarna. För att motverka effekten av dessa små steglängder genererades därför många olika startgissningar vid varje parameteruppskattning. Att göra detta är dock en ineffektiv metod beräkningsmässigt och det är svårt att veta om gissningarna lyckades bestämma den bästa parametern. Detta belyser behovet av en effektivare lösning som till exempel användandet av en annan parameteruppskattningsmetod som inte är lokal som den som beskrivs i avsnitt 3.4.

Oavsett åtgärder för att förbättra modellerna hade det vid vidare utveckling av projektet varit fördelaktigt att samarbeta med ett laboratorium. Det hade tillåtit att data kring komponenterna i modellen hade kunnat genereras vilket hade bidragit till en bättre parameteruppskattning. Mer specifikt hade parametrarna troligen kunnat skattas med mycket högre noggrannhet och följaktligen hade modellerna lämpat sig bättre för att göra förutsägelser. Den typ av data som hade varit intressant hade varit en tidsserie för Sch9 med fler mätpunkter och tidsserier för aktiviteten hos övriga ingående komponenter. Den sistnämnda datan hade kunnat användas för både kalibrering, men också för att erhålla en verifierad modell av TOR.

En modell av TOR som beskriver verkligheten hade kunnat användas i flera syften. Som det har diskuterats tidigare hade den kunnat användas till att simulera flera olika scenarion och därav sparat många timmar av laborativt arbete. Förutom detta hade en verifierad modell kunnat användas till att, med hjälp av känslighetsanalys, bestämma kritiska parametrar som styr dynamiken hos signalvägen [43]. Utifrån kunskapen om de kritiska parametrarna hade bland annat intressanta mål för läkemedel kunnat identifieras. Ytterligare hade till exempel modellen som beskriver hur aktiviteten av Sch9 regleras av TORC1 kunnat integreras i en större modell av TORC1, vilket eventuellt hade tillåtit en inblick i komplexets roll i åldrande och sjukdomar såsom cancer. Så även om modellen inte lyckades besvara frågan kring hur TOR reglerar protein- och ribosomsyntes så belyser potentialen hos en verifierad modell att projektet vore intressant att arbeta vidare med.

7 Slutsats

Den modell som konstruerats delades upp i modul 1 och 2, varav modul 1 kunde beskriva mätdatan medan modul 2 misslyckades med detta. Ytterligare modifieringar krävs därför ännu för att få en fungerande modell. Parametrarna som skattades för modul 1 hade låg identifierbarhet och hög kovarians och är därmed inte numeriskt identifierbara. En alternativ variant av modul 1 med fixerat inflöde av glutamin, reducerat antal parametrar och modifierad sönderfallshastighet för Sch9p påvisade liknande resultat som modul 1, men med lite förbättringar. Sammantaget var dock inte heller parametrarna för denna modell numeriskt identifierbara. Modul 2 kunde inte beskriva mätdatan, något som påvisar att modellen är felaktigt konstruerad ur ett biologiskt perspektiv. Framöver hade det varit intressant att undersöka en alternativ uppställning av modul 2, visad i figur 9. Gällande modul 1, och möjligtvis även modul 2 när denna kan fånga mätdatans kurvatur, hade det varit intressant att undersöka fler alternativa parameterreduktioner. För båda moduler kan också ytterligare modifikationer av hastighetsuttrycken i form av ändrad reaktionsordning

undersökas. Utifrån detta kan sägas att mer arbete krävs för nå den punkt då modellen kan användas för att göra förutsägelser, men att det finns indikationer på att det är möjligt att skapa en modell som uppfyller detta.

Referenser

[1] Mathieu Laplante och David M. Kuepferini. mTOR signaling in growth control and disease. *Cell*, 149(2):274, 2012.

[2] A González och M N Hall. Nutrient sensing and TOR signaling in yeast and mammals. *EMBO Journal*, 36(4):397–408, 2017.

[3] Andreas Raue, Marcel Schilling, Julie Bachmann, Andrew Matteson, Max Schelke, Daniel Kaschek, Sabine Hug, Clemens Kreutz, Brian D. Harms, Fabian J. Theis, Ursula Klingmüller och Jens Timmer. Lessons Learned from Quantitative Dynamical Modeling in Systems Biology. *PLoS ONE*, 8(9):e74335, sep 2013.

[4] Alexandre Huber, Bernd Bodenmiller, Aino Uotila, Michael Stahl, Stefanie Wanka, Bertran Gerrits, Ruedi Aebersold och Robbie Loewith. Characterization of the rapamycin-sensitive phosphoproteome reveals that Sch9 is a central coordinator of protein synthesis. *Genes & development*, 23(16):1929–43, aug 2009.

[5] D Stracka, S Jozefczuk, F Rudroff, U Sauer och M N Hall. Nitrogen source activates TOR (target of rapamycin) complex 1 via glutamine and independently of Gtr/Rag proteins. *Journal of Biological Chemistry*, sep 2014.

[6] Bruce Alberts, Alexander Johnson, Julian Lewis, David Morgan, Martin Raff, Keith Roberts och Peter Walter. *Molecular biology of the cell*. CRC Press Inc, 2014.

[7] Stanley Fields och Mark Johnston. Cell biology. Whither model organism research? *Science (New York, N.Y.)*, 307(5717):1885–6, mar 2005.

[8] Michael Madigan, John Martinko, Kelly Bender, Daniel Buckley och David Stahl. *Brock Biology of Microorganisms, Global Edition*. Pearson Education Limited, 2014.

[9] Leonard Guarente och Cynthia Kenyon. Genetic pathways that regulate ageing in model organisms. *Nature*, 408(6809):255–262, nov 2000.

[10] Robbie Loewith, Estela Jacinto, Stephan Wullschleger, Anja Lorberg, José L. Crespo, Débora Bonenfant, Wolfgang Oppliger, Paul Jenoe och Michael N. Hall. Two TOR Complexes, Only One of which Is Rapamycin Sensitive, Have Distinct Roles in Cell Growth Control. *Molecular Cell*, 10(3):457–468, sep 2002.

[11] Yoshiharu Inoue och Wataru Nomura. TOR Signaling in Budding Yeast. I: *The Yeast Role in Medical Applications*. InTech, jan 2018.

[12] Katie Powis, Tianlong Zhang, Nicolas Panchaud, Rong Wang, Claudio De Virgilio och Jianping Ding. Crystal structure of the Ego1-Ego2-Ego3 complex and its role in promoting Rag GTPase-dependent TORC1 signaling. *Cell Research*, 25(9):1043–1059, sep 2015.

[13] Svetlana Dokudovskaya och Michael P Rout. SEA you later alli-GATOR–a dynamic regulator of the TORC1 stress response pathway. *Journal of cell science*, 128(12):2219–28, jun 2015.

[14] Nicolas Panchaud, Marie-Pierre Péli-Gulli och Claudio De Virgilio. SEACing the GAP that nEGOCiates TORC1 activation. *Cell Cycle*, 12(18):2948–2952, sep 2013.

[15] M Shimobayashi, MN Hall Cell Research och undefined 2016. Multiple amino acid sensing inputs to mTORC1. *nature.com*.

[16] Marie Pierre Péli-Gulli, Alessandro Sardu, Nicolas Panchaud, Serena Raucci och Claudio De Virgilio. Amino Acids Stimulate TORC1 through Lst4-Lst7, a GTPase-Activating Protein Complex for the Rag Family GTPase Gtr2. *Cell Reports*, 13(1):1–7, oct 2015.

[17] Riko Hatakeyama och Claudio De Virgilio. Unsolved mysteries of Rag GTPase signaling in yeast. *Small GTPases*, 7(4):239–246, oct 2016.

[18] Marie-Pierre Péli-Gulli, Serena Raucci, Zehan Hu, Jörn Dengjel och Claudio De Virgilio. Feedback Inhibition of the Rag GTPase GAP Complex Lst4-Lst7 Safeguards TORC1 from Hyperactivation by Amino Acid Signals. *Cell Reports*, 20(2):281–288, jul 2017.

[19] Matteo Binda, Marie-Pierre Péli-Gulli, Grégory Bonfils, Nicolas Panchaud, Jörg Urban, Thomas W. Sturgill, Robbie Loewith och Claudio De Virgilio. The Vam6 GEF Controls TORC1 by Activating the EGO Complex. *Molecular Cell*, 35(5):563–573, sep 2009.

[20] Jacqueline Cherfils och Mahel Zeghouf. Regulation of Small GTPases by GEFs, GAPs, and GDIs. *Physiological Reviews*, 93(1):269–309, jan 2013.

[21] Grégory Bonfils, Malika Jaquenoud, Séverine Bontron, Clemens Ostrowicz, Christian Ungermann och Claudio De Virgilio. Leucyl-tRNA Synthetase Controls TORC1 via the EGO Complex. *Molecular Cell*, 46(1):105–110, apr 2012.

[22] Li Li och Kun-Liang Guan. Amino Acid Signaling to TOR Activation: Vam6 Functioning as a Gtr1 GEF. *Molecular Cell*, 35(5):543–545, sep 2009.

[23] Sebastian Alers, Antje S Löffler, Sebastian Wesselborg och Björn Stork. Role of AMPK-mTOR-Ulk1/2 in the regulation of autophagy: cross talk, shortcuts, and feedbacks. *Molecular and cellular biology*, 32(1):2–11, jan 2012.

[24] Hitoshi Nakatogawa, Kuninori Suzuki, Yoshiaki Kamada och Yoshinori Ohsumi. Dynamics and diversity in autophagy mechanisms: lessons from yeast. *Nature Reviews Molecular Cell Biology*, 10(7):458–467, jul 2009.

[25] Atsuhiro Adachi, Michiko Koizumi och Yoshinori Ohsumi. Autophagy induction under carbon starvation conditions is negatively regulated by carbon catabolite repression. *The Journal of biological chemistry*, 292(48):19905–19918, dec 2017.

[26] Gonghong Yan, Yumei Lai och Yu Jiang. The TOR Complex 1 Is a Direct Target of Rho1 GTPase. *Molecular Cell*, 45(6):743–753, mar 2012.

[27] Gonghong Yan, Xiaoyun Shen och Yu Jiang. Rapamycin activates Tap42-associated phosphatases by abrogating their association with Tor complex 1. *The EMBO Journal*, 25(15):3546–3555, aug 2006.

[28] Vitor Teixeira och Vítor Costa. Unraveling the role of the Target of Rapamycin signaling in sphingolipid metabolism. *Progress in Lipid Research*, 61:109–133, jan 2016.

[29] James E Hughes Hallett, Xiangxia Luo och Andrew P Capaldi. State transitions in the TORC1 signaling pathway and information processing in Saccharomyces cerevisiae. *Genetics*, 198(2):773–86, oct 2014.

[30] Min Wei, Paola Fabrizio, Jia Hu, Huanying Ge, Chao Cheng, Lei Li och Valter D. Longo. Life Span Extension by Calorie Restriction Depends on Rim15 and Transcription Factors Downstream of Ras/PKA, Tor, and Sch9. *PLoS Genetics*, 4(1):e13, 2008.

[31] Ian M. Willis och Robyn D. Moir. Integration of nutritional and stress signaling pathways by Maf1. *Trends in Biochemical Sciences*, 32(2):51–53, feb 2007.

[32] Simon C. Johnson, Peter S. Rabinovitch och Matt Kaeberlein. mTOR is a key modulator of ageing and age-related disease. *Nature*, 493(7432):338, 2013.

[33] Akio Nakashima, Yoko Otsubo, Akira Yamashita, Tatsuhiro Sato, Masayuki Yamamoto och Fuyuhiko Tamanoi. Psk1, an AGC kinase family member in fission yeast, is directly phosphorylated and controlled by TORC1 and functions as S6 kinase. *Journal of Cell Science*, 125:5840–5849.

[34] Yoko Otsubo, Masayuki Yamamoto, Hayao Ohno och Masayuki Yamamoto. Signaling pathways for fission yeast sexual differentiation at a glance. *Journal of cell science*, 125(Pt 12):2789–93, jun 2012.

[35] Thomas Dobrenel, Camila Caldana, Johannes Hanson, Christophe Robaglia, Michel Vincentz, Bruce Veit och Christian Meyer. TOR Signaling and Nutrient Sensing. *Annual Review of Plant Biology*, 67(1):261–285, apr 2016.

[36] Ferdi L. Hellweger, Robert J. Clegg, James R. Clark, Caroline M. Plugge och Jan Ulrich Kreft. Advancing microbial sciences by individual-based modelling, 2016.

[37] Edda Klipp och Wolfram Liebermeister. Mathematical modeling of intracellular signaling pathways, 2006.

[38] Nationalencyklopedin. *Massverkans lag - Uppslagsverk*, hämtad 2018-04-26.

[39] Wierling C Lehrach H Kowald A Klipp E, Herwig R. *Systems Biology in Practice: Concepts, Implementation and Application*. Wiley-VCH, 2005.

[40] Peter Atkins, Julio de Paula och Ronald Friedman. *Physical Chemistry - Quanta, Matter and Change*. Oxford University Press, 2014.

[41] Nurgazy Sulaimanov, Martin Klose, Hauke Busch och Melanie Boerries. Understanding the mTOR signaling pathway via mathematical modeling. *Wiley Interdisciplinary Reviews: Systems Biology and Medicine*, 9(4):e1379, jul 2017.

[42] Joshua F. Apgar, Jared E. Toettcher, Drew Endy, Forest M. White och Bruce Tidor. Stimulus design for model selection and validation in cell signaling. *PLoS Computational Biology*, 2008.

[43] Joseph J. DiStefano III. *Dynamic systems biology modeling and simulation*. Academic press, 2014.

[44] Anders Rasmuson, Bengt Andersson och Lousie Olsson. *Mathematical modeling in chemical engineering*. Cambridge University Press, 2014.

[45] Larisa Beilina, Evgenii Kerchevskii och Mikhail Karchevskii. *Numerical linear algebra: Theory and applications*. Springer International Publishing, 2017.

[46] Patriksson M. Andréasson N, Evgrafov A. *An Introduction to Optimization: Foundations and Fundamental Algorithms.2:a utgåvan*. Studentlitteratur. Clarendon Press, 2013.

[47] Ernst Hairer, Syvert P. Nørsett och Gerhard Wanner. *Solving ordinary differential equations I*. Springer series in computational mathematics. Springer International Publishing, 1993.

[48] Piero Dalle Pezze, Stefanie Ruf, Annika G Sonntag, Miriam Langelaar-Makkinje, Philip Hall, Alexander M Heberle och et al. A systems study reveals concurrent activation of AMPK and mTOR by amino acids. *Nature communications*, 7:13254, 2016.

[49] Alejandro F. Villaverde, Antonio Barreiro och Antonis Papachristodoulou. Structural Identifiability of Dynamic Systems Biology Models. *PLOS Computational Biology*, 12(10):e1005153, oct 2016.

[50] Johan Karlsson, Milena Anguelova och Mats Jirstrand. An Efficient Method for Structural Identifiability Analysis of Large Dynamic Systems. *IFAC Proceedings Volumes*, 45(16):941–946, jul 2012.

A Härledning av matrisuttryck för $\nabla f(\mathbf{k})$ och $\tilde{H}$

Härledning av matrissuttryck för $\nabla f(\mathbf{k})$

För att formulera ett matrisuttryck för $\nabla f(\mathbf{k})$ används att vikterna w_i och mätdatan $y(t_i)$ i kostnadsfunktionen inte är funktioner av parametrarna $k_1, k_2, \ldots, k_n$. Det senare resulterar i att följande gäller:

$$\frac{\partial r_i}{\partial k_j} = \frac{\partial}{\partial k_j}\Big(y(t_i) - \hat{y}(\mathbf{k}, t_i)\Big) = -\frac{\partial \hat{y}(\mathbf{k}, t_i)}{\partial k_j} = -\frac{\partial \hat{y}_i}{\partial k_j} \tag{14}$$

Givet detta kan $\nabla f(\mathbf{k})$ skrivas enligt:

$$\nabla f(\mathbf{k}) = \nabla\Bigg(\sum_{i=1}^{m} w_i\big(y_i - \hat{y}(\mathbf{k}, t_i)\big)^2\Bigg) = \sum_{i=1}^{m} w_i \nabla\big(y_i - \hat{y}(\mathbf{k}, t_i)\big)^2 = 2\sum_{i=1}^{m} w_i\big(y_i - \hat{y}(\mathbf{k}, t_i)\big)\nabla\big(y_i - \hat{y}(\mathbf{k}, t_i)\big)$$

$$= -2\sum_{i=1}^{m} w_i r_i \nabla\hat{y}(\mathbf{k}, t_i) = -2\Bigg(w_1 r_1 \begin{bmatrix} \frac{\partial \hat{y}_1}{\partial k_1} \\ \frac{\partial \hat{y}_1}{\partial k_2} \\ \vdots \\ \frac{\partial \hat{y}_1}{\partial k_n} \end{bmatrix} + w_2 r_2 \begin{bmatrix} \frac{\partial \hat{y}_2}{\partial k_1} \\ \frac{\partial \hat{y}_2}{\partial k_2} \\ \vdots \\ \frac{\partial \hat{y}_2}{\partial k_n} \end{bmatrix} + \ldots + w_m r_m \begin{bmatrix} \frac{\partial \hat{y}_m}{\partial k_1} \\ \frac{\partial \hat{y}_m}{\partial k_2} \\ \vdots \\ \frac{\partial \hat{y}_m}{\partial k_n} \end{bmatrix}\Bigg) = -2A^T(\mathbf{k})Wr(\mathbf{k})$$

Där matrisen $A(\mathbf{k}) = A \in \mathbb{R}^{m\times n}$ ges i ekvation (9). Observera att $\nabla f(\mathbf{k}) = -2A^T W r$ är exakt samma uttryck som det i ekvation (8).

Härledning av matrissuttryck för $\tilde{H}$

Approximationen av $\tilde{H}$ kan härledas genom att först betrakta ett godtyckligt element, H_{jk}, i hessianen till kostnadsfunktionen:

$$H_{jk} = \frac{\partial^2 f(\mathbf{k})}{\partial k_j \partial k_k} = \frac{\partial^2}{\partial k_j \partial k_k}\sum_{i=1}^{m} w_i r_i^2 = \sum_{i=1}^{m} w_i \frac{\partial^2 r_i^2}{\partial k_j \partial k_k} = 2\sum_{i=1}^{m} w_i\Big(\frac{\partial r_i}{\partial k_j}\frac{\partial r_i}{\partial k_k} + r_i\frac{\partial^2 r_i}{\partial k_j \partial k_k}\Big) \tag{15}$$

Problemet med att utvärdera H är att de partiella andra ordningens derivator för residualerna $\frac{\partial^2 r_i}{\partial k_j \partial k_k}$ behöver beräknas. I syfte att kringgå detta problem görs följande antagande:

$$\left|\sum_{i=1}^{m} r_i \frac{\partial^2 r_i}{\partial k_j \partial k_k}\right| << \left|\sum_{i=1}^{m} \frac{\partial r_i}{\partial k_j}\frac{\partial r_i}{\partial k_k}\right| \tag{16}$$

Antagande i ekvation (16) håller om funktionen är milt icke linjär eller om residualerna är små, det senare är rimligt kring ett minimum till kostnadsfunktionen [46]. Som tidigare motiverat gäller att $\frac{\partial r_i}{\partial k_j} = -\frac{\hat{y}(\mathbf{k}, t_i)}{\partial k_j} = \frac{\partial \hat{y}_i}{\partial k_j}$, kombineras detta med approximation i ekvation (16) erhålls en hessianapproximationen $\tilde{H} \in \mathbb{R}^{n\times n}$ med följande utseende:

$$\tilde{H} = \sum_{i=1}^{m} \begin{bmatrix} w_i \frac{\partial \hat{y}_i}{\partial k_1}\frac{\partial \hat{y}_i}{\partial k_1} & w_i \frac{\partial \hat{y}_i}{\partial k_1}\frac{\partial \hat{y}_i}{\partial k_2} & \cdots & w_i \frac{\partial \hat{y}_i}{\partial k_1}\frac{\partial \hat{y}_i}{\partial k_n} \\ w_i \frac{\partial \hat{y}_i}{\partial k_2}\frac{\partial \hat{y}_i}{\partial k_1} & w_i \frac{\partial \hat{y}_i}{\partial k_2}\frac{\partial \hat{y}_i}{\partial k_2} & \cdots & w_i \frac{\partial \hat{y}_i}{\partial k_2}\frac{\partial \hat{y}_i}{\partial k_n} \\ \vdots & \vdots & \ddots & \vdots \\ w_i \frac{\partial \hat{y}_i}{\partial k_n}\frac{\partial \hat{y}_i}{\partial k_1} & w_i \frac{\partial \hat{y}_i}{\partial k_n}\frac{\partial \hat{y}_i}{\partial k_2} & \cdots & w_i \frac{\partial \hat{y}_i}{\partial k_n}\frac{\partial \hat{y}_i}{\partial k_n} \end{bmatrix} \tag{17}$$

Observera att $\tilde{H}$, likt H, är en symmetrisk matris. Med målet att formulera uttrycket för $\tilde{H}$ i ekvation (17) på matrisform betraktas kolumn j i $\tilde{H}$:

$$\begin{aligned}\mathrm{Col}_j(\tilde{H}) &= 2\sum_{i=1}^{m}\begin{bmatrix} w_i\frac{\partial\hat{y}_i}{\partial k_1}\frac{\partial\hat{y}_i}{\partial k_j} \\ w_i\frac{\partial\hat{y}_i}{\partial k_2}\frac{\partial\hat{y}_i}{\partial k_j} \\ \vdots \\ w_i\frac{\partial\hat{y}_i}{\partial k_n}\frac{\partial\hat{y}_i}{\partial k_j}\end{bmatrix} = \begin{bmatrix} w_1\frac{\partial\hat{y}_1}{\partial k_1}\frac{\partial\hat{y}_1}{\partial k_j} + w_2\frac{\partial\hat{y}_2}{\partial k_1}\frac{\partial\hat{y}_2}{\partial k_j} + \ldots + w_m\frac{\partial\hat{y}_m}{\partial k_1}\frac{\partial\hat{y}_m}{\partial k_j} \\ w_1\frac{\partial\hat{y}_1}{\partial k_2}\frac{\partial\hat{y}_1}{\partial k_j} + w_2\frac{\partial\hat{y}_2}{\partial k_2}\frac{\partial\hat{y}_2}{\partial k_j} + \ldots + w_m\frac{\partial\hat{y}_m}{\partial k_2}\frac{\partial\hat{y}_m}{\partial k_j} \\ \vdots \\ w_1\frac{\partial\hat{y}_1}{\partial k_n}\frac{\partial\hat{y}_1}{\partial k_j} + w_2\frac{\partial\hat{y}_2}{\partial k_n}\frac{\partial\hat{y}_2}{\partial k_j} + \ldots + w_m\frac{\partial\hat{y}_i}{\partial k_n}\frac{\partial\hat{y}_m}{\partial k_j}\end{bmatrix} \\ &= 2\begin{bmatrix} w_1\frac{\partial\hat{y}_1}{\partial k_1} & w_2\frac{\partial\hat{y}_2}{\partial k_1} & \cdots & w_m\frac{\partial\hat{y}_m}{\partial k_1} \\ w_1\frac{\partial\hat{y}_1}{\partial k_2} & w_2\frac{\partial\hat{y}_2}{\partial k_2} & \cdots & w_m\frac{\partial\hat{y}_m}{\partial k_2} \\ \vdots & \vdots & \ddots & \vdots \\ w_w\frac{\partial\hat{y}_1}{\partial k_n} & w_2\frac{\partial\hat{y}_2}{\partial k_n} & \cdots & w_m\frac{\partial\hat{y}_m}{\partial k_m}\end{bmatrix}\begin{bmatrix}\frac{\partial\hat{y}_1}{\partial k_j} \\ \frac{\partial\hat{y}_2}{\partial k_j} \\ \vdots \\ \frac{\partial\hat{y}_m}{\partial k_j}\end{bmatrix} = 2D\mathrm{Col}_j(A(\mathbf{k}))\end{aligned}$$

Där matrisen $A(\mathbf{k}) = A \in \mathbb{R}^{m\times n}$ ges i ekvation (9). För att ytterligare förenkla uttrycket för $\mathrm{Col}_j(\tilde{H})$ skrivs matrisen D om enligt:

$$D = \begin{bmatrix}\frac{\partial\hat{y}_1}{\partial k_1} & \frac{\partial\hat{y}_2}{\partial k_1} & \cdots & \frac{\partial\hat{y}_m}{\partial k_1} \\ \frac{\partial\hat{y}_1}{\partial k_2} & \frac{\partial\hat{y}_2}{\partial k_2} & \cdots & \frac{\partial\hat{y}_m}{\partial k_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial\hat{y}_1}{\partial k_n} & \frac{\partial\hat{y}_2}{\partial k_n} & \cdots & \frac{\partial\hat{y}_m}{\partial k_n}\end{bmatrix}\begin{bmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_m\end{bmatrix} = A^TW \tag{18}$$

Där matrisen $W \in \mathbb{R}^{m\times m}$ ges i ekvation (9). Genom att använda definitionen av matris-matris multiplikation erhålls sedan följande uttryck för $\tilde{H}$:

$$\tilde{H} = 2\begin{bmatrix}A^TW\mathrm{Col}_1(A) & A^TW\mathrm{Col}_2(A) & \cdots & A^TW\mathrm{Col}_n(A)\end{bmatrix} = 2A^TWA \tag{19}$$

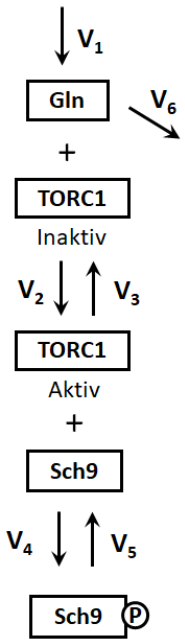
Observera att uttrycket för $\tilde{H}$ i ekvation (19) är exakt samma uttryck för $\tilde{H}$ i ekvation (8).

B Hastighetsuttryck och ODE:er för modeller

I denna bilaga redovisas hastighetsuttryck och ODE:er för de modeller som ingick i arbetet, men som ej finns fullständigt redovisade i rapporten. Modellerna som inkluderas i denna bilaga är: Modellen av endast modul 1, den alternativa modellen av båda modulerna som presenteras i avsnitt 6 samt den alternativa modellen av modul 1 som presenteras i avsnitt 5.3.

Modell av modul 1

$$
\begin{aligned}
x_1 &= [Gln] & v_1 &= k_1 \\
x_2 &= [TORC1]\ \text{Inaktiv} & v_2 &= k_2 x_1 x_2 \\
x_3 &= [TORC1]\ \text{Aktiv} & v_3 &= k_3 x_3 \\
x_4 &= [Sch9] & v_4 &= k_4 x_3 x_4 \\
x_5 &= [Sch9\text{p}] & v_5 &= k_5 x_5 \\
 & & v_6 &= k_6 x_1
\end{aligned}
$$



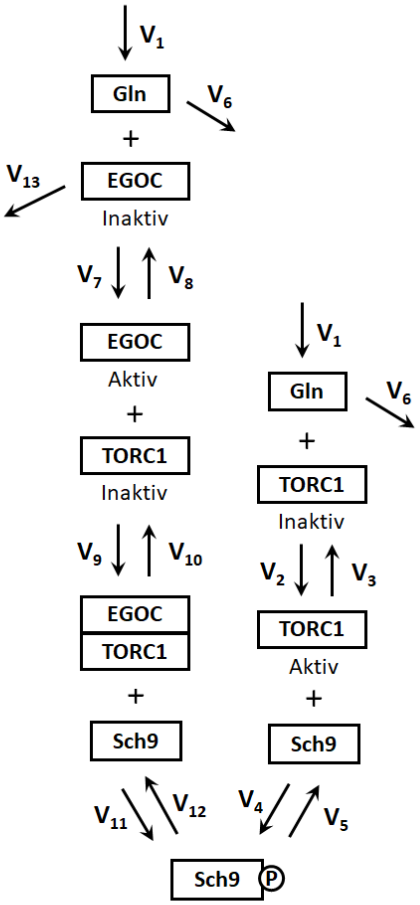
$$
\begin{aligned}
\frac{\mathrm{d}x_1}{\mathrm{d}t} &= v_1 - v_6 \\
\frac{\mathrm{d}x_2}{\mathrm{d}t} &= v_3 - v_2 \\
\frac{\mathrm{d}x_3}{\mathrm{d}t} &= v_2 - v_3 \\
\frac{\mathrm{d}x_4}{\mathrm{d}t} &= v_5 - v_4 \\
\frac{\mathrm{d}x_5}{\mathrm{d}t} &= v_4 - v_5
\end{aligned}
$$

Alternativ modell av båda modulerna

$x_1 = [Gln]$
$x_2 = [TORC1]$ Inaktiv
$x_3 = [TORC1]$ Aktiv
$x_4 = [Sch9]$
$x_5 = [Sch9\text{p}]$
$x_6 = [EGOC]$ Inaktiv
$x_7 = [EGOC]$ Aktiv
$x_8 = [EGOC - TORC1]$

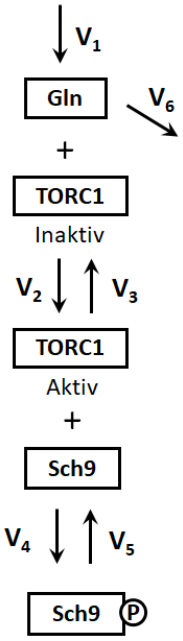
$v_1 = k_1$
$v_2 = k_2 x_1 x_2$
$v_3 = k_3 x_3$
$v_4 = k_4 x_3 x_4$
$v_5 = k_5 x_5$
$v_6 = k_6 x_1$
$v_7 = k_7 x_1 x_6$
$v_8 = k_8 x_7$
$v_9 = k_9 x_7 x_2$
$v_{10} = k_{10} x_8$
$v_{11} = k_{11} x_8 x_4$
$v_{12} = k_{12} x_5$
$v_{13} = k_{13} x_5$

$$\frac{\mathrm{d}x_1}{\mathrm{d}t} = v_1 - v_6$$
$$\frac{\mathrm{d}x_2}{\mathrm{d}t} = v_3 - v_2 + v_{10} - v_9$$
$$\frac{\mathrm{d}x_3}{\mathrm{d}t} = v_2 - v_3$$
$$\frac{\mathrm{d}x_4}{\mathrm{d}t} = v_5 - v_4 + v_{12} - v_{11}$$
$$\frac{\mathrm{d}x_5}{\mathrm{d}t} = v_4 - v_5 + v_{11} - v_{12}$$
$$\frac{\mathrm{d}x_6}{\mathrm{d}t} = v_8 - v_7 - v_{13}$$
$$\frac{\mathrm{d}x_7}{\mathrm{d}t} = v_7 - v_8$$
$$\frac{\mathrm{d}x_8}{\mathrm{d}t} = v_9 - v_{10}$$



Alternativ modell av modul 1

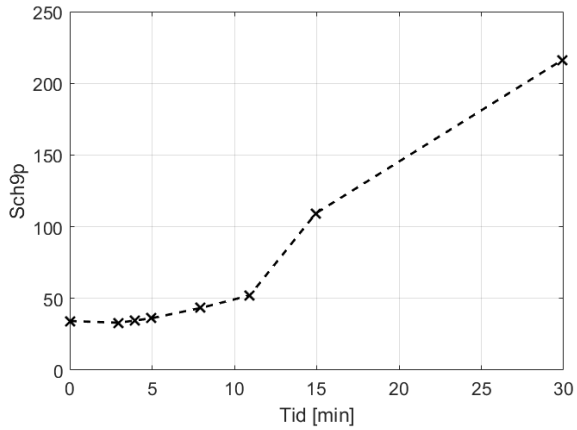
$$
\begin{aligned}
x_1 &= [Gln] \\
x_2 &= [TORC1] \text{ Inaktiv} \\
x_3 &= [TORC1] \text{ Aktiv} \\
x_4 &= [Sch9] \\
x_5 &= [Sch9\text{p}]
\end{aligned}
\qquad
\begin{aligned}
v_1 &= k_1 \\
v_2 &= k_2 x_1 x_2 \\
v_3 &= k_2 x_3^2 \\
v_4 &= k_3 x_3 x_4 \\
v_5 &= k_4 x_5 \\
v_6 &= k_5 x_1
\end{aligned}
$$



$$
\begin{aligned}
\frac{\mathrm{d}x_1}{\mathrm{d}t} &= v_1 - v_6 \\
\frac{\mathrm{d}x_2}{\mathrm{d}t} &= v_3 - v_2 \\
\frac{\mathrm{d}x_3}{\mathrm{d}t} &= v_2 - v_3 \\
\frac{\mathrm{d}x_4}{\mathrm{d}t} &= v_5 - v_4 \\
\frac{\mathrm{d}x_5}{\mathrm{d}t} &= v_4 - v_5
\end{aligned}
$$

C Data

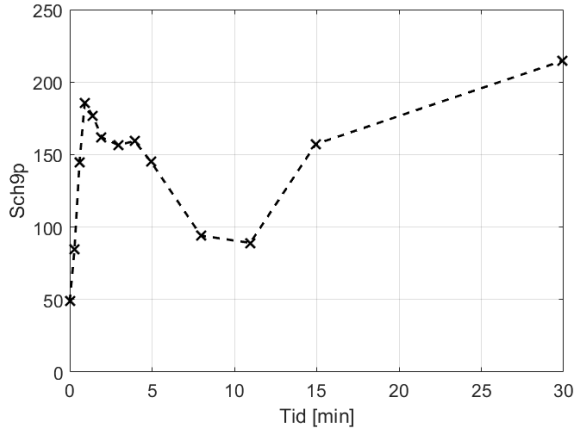
I denna bilaga redovisas den experimentella datan som har används i arbetet för att testa modellen.



Figur 10: *Mätdataserien för modul 1 med antalet fosforylerat Sch9 plottat mot tiden.*

Tabell 4: *Mätdataserien för modul 1.*

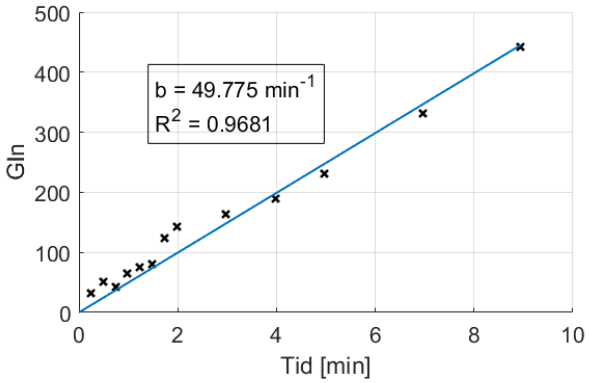
Tid (min)	Mängden Gln i cellen
0	34.3
3.0	33.0
4.0	34.6
5.0	36.2
8.0	43.3
11.0	52.1
15.0	109.4
30.0	216.2



Figur 11: *Den plottade mätdataserien för modul 2 med antalet fosforylerat Sch9 plottat mot tiden.*

Tabell 5: *Mätdataserien för modul 2.*

Tid (min)	Mängden Gln i cellen
0	48.8
0.3	84.5
0.6	144.7
0.9	185.4
1.4	176.7
1.9	161.9
2.9	156.6
3.9	159.3
4.4	144.9
8.0	94.1
11.0	89.1
14.9	157.2
29.9	215.0



Figur 12: *En linjär anpassning för glutaminförändringen relativt startkoncentrationen i jästcellen. Parametern b beskriver den relativa förändringen av glutamin per minut.*

Tabell 6: *Glutaminförändringen relativt startkoncentrationen i jästcellen.*

Tid (min)	Relativ glutaminförändring
0.2	32.7
0.5	51.8
0.7	42.6
1.0	65.1
1.2	74.6
1.5	80.9
1.7	123.2
2.0	142.3
3.0	162.9
4.0	189.0
5.0	231.3
7.0	331.6
8.9	441.5

Tabell 7: *Den initialmängd som används för respektive komponent.*

Komponent	Antal
Glutamin	1
Inaktivt EGOC	124
Aktivt EGOC	1
Inaktivt TORC1	288.0 (299.4)*
Aktivt TORC1	38.0 (26.7)*
Sch9	370.2
Sch9p	48.8

**Värde inom parantes ersätter ordinarie värde då modul 1 används.*

D MATLAB-kod

Totalt består parameteruppskattningsprogrammet av 17 m-filer vars funktion och samverkan redovisas i figur 13. Huvudfilen i programmet som åberopar alla andra filer för att utföra parameteruppskattningen och analysera resultatet är *PSI*. Totalt består *PSI* av sex sektioner, där första sektionen utför parameteruppskattningen medan övriga sektioner analyserar resultatet på olika sätt. I de kommande styckena, med utgångspunkt från *PSI*, kommer programmets funktion och filer att beskrivas.

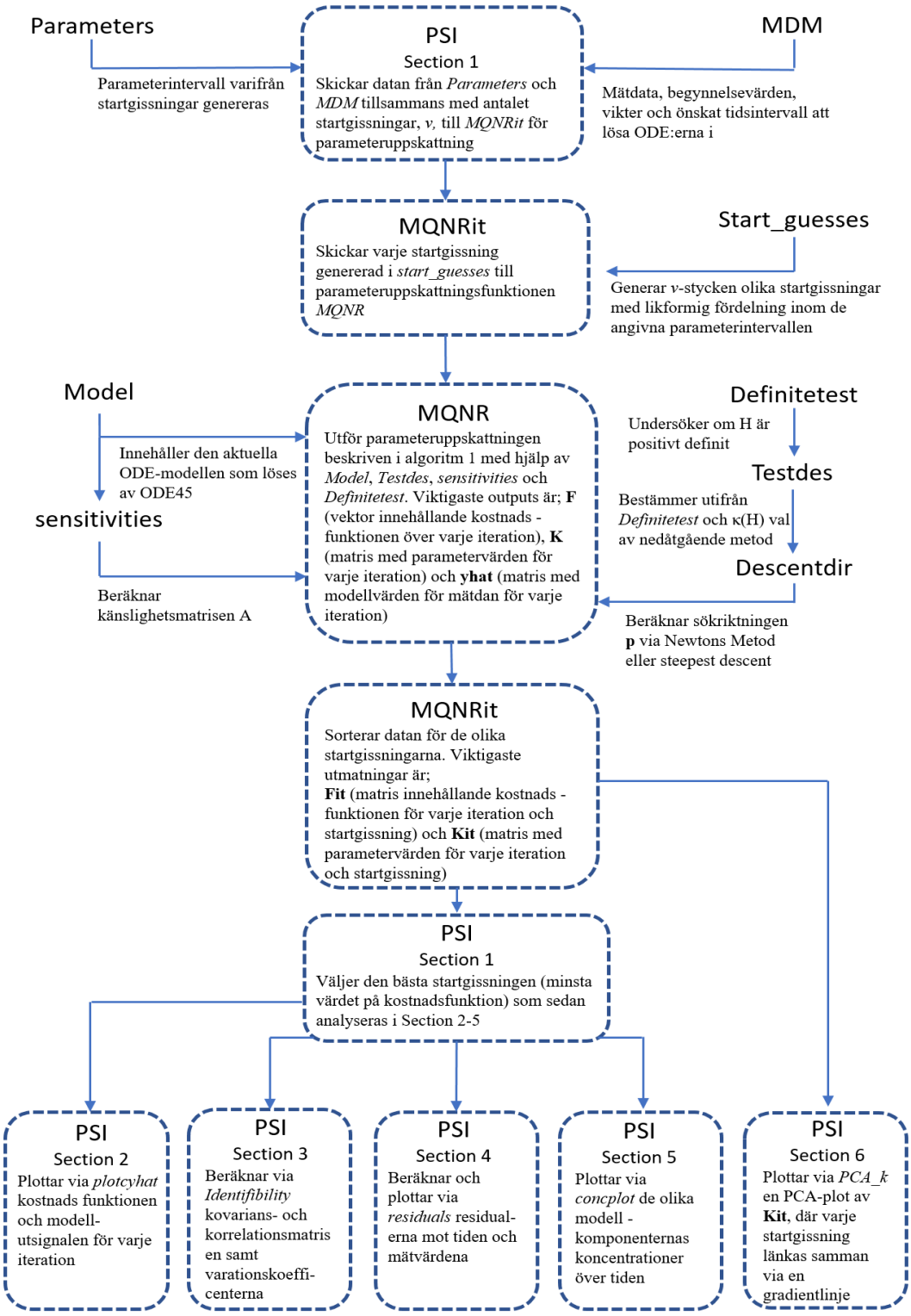
Vid en parameteruppskattning av en modell körs först sektion 1 i *PSI*. Som figur 13 visar importerar *PSI* från *MDM* mätdatan $\mathbf{y}_m$, korresponderande tidsvärdena $\mathbf{t}_m$, begynnelsevärdena $I0_c$, vikterna för viktningsmatrisen i ekvation (9) och det önskade tidsintervallet att lösa systemet av ODE:er i. Alla dessa parametrar matas in för hand i *MDM* innan körning av en modell och är nödvändiga för att *MQNR*, beskriven i algoritm 1, ska kunna köras. Vidare importerar *PSI* parameterintervall i vilka de sanna parametervärdena tros ligga från *parameters*. Slutligen definieras antalet startgissningar v, anledningen till att mer än en startgissning används är för att undvika problemet med eventuella lokala minimum.

Den importerade datan, tillsammans med antalet startgissningar, skickas sedan till *MQNRit*. Funktionen hos denna fil är skicka in de olika startgissningarna till *MQNR* och därefter sortera den erhålla utdatan. I syfte att erhålla startgissningar utifrån parameterintervallen generas i filen *startguesses* v -stycken olika startgissningar med hjälp av likformig fördelning. Dessa startgissningar skickas sedan in i parameteruppskattningsfilen *MQNR*, vars funktion beskrivs i algoritm 1. För att MATLAB-koden till *MQNR* ska vara lätt att felsöka och förstå använder den hjälpfilerna *Definitetest*, *Testdes*, *Descentdir*, *Model* och *sensitivities* till att utföra vissa av operationerna i algoritm 1. Filen *models* innehåller den aktuella ODE-modellen som löses av ODE45 och filen *sensitivities* beräknar via differenskvoten i ekvation (10) matrisen A som ges i ekvation (9). Filerna *Definitetest*, *Testdes* och *Descentdir* är alla inblandade i valet nedåtgående metod mellan Newtons-Metod eller steepest descent, där *Definitetest* undersöker om $\tilde{H}$ är positivt definit medan *Testdes* undersöker magnituden av konditionstalet $\kappa(\tilde{H})$. Är $\kappa(\tilde{H})$ stort eller är $\tilde{H} \not> 0$ beräknas sökriktningen $\mathbf{p}$ via steepest decent, annars beräknas den via quasi-Newton.

Den viktigaste utdatan från *MQNR* är **F**, **K** och **yhat**. **F** är en vektor innehållande kostnadsfunktionvärdet $f(\mathbf{k})$ för varje iteration, **K** är en matris innehållande parametervärdena för varje iteration och **yhat** är en vektor med värden på $\hat{y}(\mathbf{k}, t)$ för varje iteration. I *MQNRit* sorteras **F** och **K** in i **Fit** respektive **Kit**. Dessa matriser innehåller alla iterationsvärden för alla startgissningar för kostnadsfunktionen respektive parametrarna. Utifrån **Fit** väljer *PSI* sedan ut den bästa startgissningen, det vill säga den som erhåller minst värde på kostnadsfunktionen, och analyserar resultatet för denna.

Analys av resultatet för den bästa startgissningen sker i sektion 2 - 5 i *PSI*. Sektion 2 plottar via filen *plotcyhat* i två skilda grafer $f(\mathbf{k})$ och $\hat{y}(\mathbf{k}, t)$ över iterationerna i *MQNR*. Den senare grafen av $\hat{y}(\mathbf{k}, t)$ ger en överblick över hur modellen anpassar sig efter mätdatan, se till exempel figur 4. Sektion 3 undersöker identifierbarheten hos en modell genom att via *Identifiability* beräkna kovarians- och korrelationmatrisen samt variationskoefficienterna. Sektion 4 plottar, via *concplot*, koncentrationen, eller antalet, för de enskilda komponenterna av modellen över tidsintervallet ODE:erna löstes vid. Slutligen skapar sektion 6 en PCA-plot (primary component analysis) av **Kit**. Syftet med denna plot är att visualisera eventuella problem med lokala minimum.

Mer noggrann förklaring till de olika filernas funktion, inputs och outputs finns i MATLAB-koden som presenteras nedan. En viktig detalj i koden är att varje MATLAB-fil är uppdelad i två delar, User och Non-user. Vid byte av modell behöver innehållet i user-delarna ändras medan vid normal körning av programmet behöver non-user-delarna aldrig ändras.



Figur 13: *Flödeschema över de olika filernas funktion och samverkan i parameteruppskattnings-programmet.*

PSI.m

```
% PSI (Parameter estimation, sensitivity analysis and identifibility)

% Main program for performing the parameter estimation, sensitivity
% analysis, residual analysis and identifibility for a given model.
% Different sections in PSI  generates different plots etc. to analyze the
% result. To be capable of running section > 1, section 1 has to be run
% first.

% To run PSI for a model first fill in the user section in PSI Section 1,
% model.m, sensitivites.m, parameters.m, MDM.m, MQNR and save the files
% in a folder named ModelT (T denoting model number) as PSIT.m,
% modelT, sensitivitesT, MDMT, MQNRT and parametersT. Also in this folder
% save the non-user function Definitetest.m, Descenddir.m, Testdes
% (don't change the name of these).

% The different sections in PSI are:
% Section 1 - Parameter estimation (runs MQNR)
% Section 2 - Plotting cost function and simulated outsignal
% Section 3 - Identifiability analysis
% Section 4 - Resdiual analysis
% Section 5 - Plotting selected concentrations
% Section 6 - PCA-plot of parameter-vector

%% Section 1, Parameter estimation
% Obs, to run the other sections this has to be run first
clc, clear, clf, tic
% -------------------------- User ---------------------------------- %
% Select a given model for MDMT, MQNR and Parameters below, e.g if model 2
% is used write MDM2, Parmeters2 and MQNR2(...).

% Importing measurement data (I0c, ym and tm) from MDMT
MDM3

% Importing parameter data (intervall for each parameter) from ParametersT
Parameters3

% Select the number of random start-guesses to perform
v = 5;                                % Number of start-guesses (v)

% Calculating Kit, Nit and Fit (for explanation se MQNRit)
[Kit,Nit,Fit,Kin] = MQNRit3(I0c,ym,tm,wm,tint,v,Kint);

% Locating smallest value in Fit (best result)
[M, I] = min(Fit);
kin = Kin(I,:);                     % Selecting best start-guess

% Generating K,N,A,F and Yhat for the best start-guess

[K,N,A,F,Yhat] = MQNR_3(kin,I0c,ym,tm,wm,tint);

% ------------------------- Non User ------------------------------- %
% All the analysis in section 2-5 are for the best start-guess

H = 2*(A')*A;             % Approximated hessian
k = K(N+1,:);             % Optimized parameter vector for best start-guess

toc

clear M I t0 tf

%% Section 2 Cost function and simulated outsignal
% Plots the cost function and simulated outsignalfor each iteration in
% plotcyhat
plotcyhat

%% Section 3 Identifiability
% Calculates covariance matrix, correlation matrix and coefficients of
```

```
% variance for the parameters in Identifiability
Identifiability

%% Section 4 Residual analysis
% Plotting standarized residuals in residuals.m
sresiduals

% Normplot of residuals
figure(4)
normplot(r)

%% Section 5 Plots of selected concentrations
% Plots the selected concentrations in Concplot
Concplot

%% Section 6 PCA-plot
% Plots PCA-plot in PCA_k
PCA_k
```

MDM.m

```
% MDMT (Measurement data model T)
% File that contains the data required to parameter estimate model T
% Outputs are:
% ym - measurement data for the outsignal               (1 x m) matrix
% tm - timepoints where the outsignal is measured       (1 x m) matrix
% I0c - initial values for the components
% in model T at time 0                                  (1 x g) matrix
% wm - weight matrix with the weight for each
% observation, ideally (if possible)
% choose wi = 1/sigmai^2                                (1 x m) matrix
% tint - vector containing time intervall to
% solve the ODE:s                                       (1 x 2) matrix

% ------------------------- User -------------------------------- %
% Select time intervall tint to solve the ODE:s in.
t0 = 0;                 % Inital time value
tf = 1;                 % Final time value for solving ODE:s
tint = [t0 tf];         % Time intervall for solving ODEs


% Formulate ym, wm and tm in the following format:
% ym = [ym(1) ym(2) ... ym(m)]; where ym(i) is the i:th element in the
% time-serie. If no weights are used choose wm as wm = ones(1,m)

tm = 0:0.05:1;
ym = [0.9771 0.9354 0.9142 0.8270 0.8423 0.8468 0.8110 0.8725 0.8541 ...
    0.9085 0.8708 0.8956 0.8735 0.8927 0.8905 0.9302 0.9566 0.9267 ...
    0.9548 0.9432 0.9621];
wm = ones(1,21);

% Formulate the inital values in the following format (one for each row):
% I0c(1) = 3;                   % Which component
% I0c(2) = 4;                   % Which component

I0c(1) = 1;
I0c(2) = 1;
I0c(3) = 1;
I0c(4) = 10;
I0c(5) = 2;
I0c(6) = 1;
```

ParametersT.m

```
% ParametersT (Parameters model T)
% File that contains the interval to use as start-guesses in MQNR for the
```

```
% n parameters.
% Outputs are:
% Kint - matrix containg the interval for each parameter  (n x 2) matrix

% --------------------------- User --------------------------------- %

% Formulate the parameter intervals in the following format
% (one parameter for each row):
% Kint(i,1) = 3; Kint(i,2) = 5
% i corresponds to parameter i
% 1 denotes lower interval limit and 2 denotes upper interval limit

Kint(1,1) = 0.5; Kint(1,2) = 10;
Kint(2,1) = 0.5; Kint(2,2) = 10;
Kint(3,1) = 0.5; Kint(3,2) = 10;
Kint(4,1) = 0.5; Kint(4,2) = 10;
Kint(5,1) = 0.5; Kint(5,2) = 10;
Kint(6,1) = 0.5; Kint(6,2) = 10;
Kint(7,1) = 0.5; Kint(7,2) = 10;
Kint(8,1) = 0.5; Kint(8,2) = 10;
Kint(9,1) = 0.5; Kint(9,2) = 10;
Kint(10,1) = 0.5; Kint(10,2) = 10;
Kint(11,1) = 0.5; Kint(11,2) = 10;
Kint(12,1) = 0.5; Kint(12,2) = 10;
```

MQNRitT.m

```
function [Kit,Nit,Fit,Kin] = MQNRit3(I0c,ym,tm,wm,tint,v,Kint)
% Function that performes MQNR for each start guess generated by
% startguess.m.
% Outputs are:
% Kit - matrix containing the parameter values for
% each start guess and each iteration, (acts as
% data matrix for PCA-plot)                          (sum(Nit)+30 x n) matrix
% Nit - matrix containing the number of
% iterations for each start guess                    (1 x a) matrix
% Fit - vector containing the final
% cost function value for each iteration             (1 x a) matrix
% Kin - se function file startguess.m                (n x a) matrix
% For explanation about inputs se MQNR.m and startguess.m

% ------------------------------ Non user ----------------------------%
% Generating v random start guesses
Kin = startguess(Kint,v);

% Defining Kit
 Kit = [];

% Pre dimension Nit and Fit
 Nit = ones(1,v);  Fit = ones(1,v);

% -------------------------------  User ----------------------------%
% Select correct model for MQNR to solve
% Running MQNR for each start guess
for i = 1:1:v
    kin = Kin(i,:);                % Picking random start guess from Kin

    [K,N,A,F,Yhat] = MQNR_3(kin,I0c,ym,tm,wm,tint);

    Kit = [Kit; K];                 % Storing k-values
    Nit(i) = N;                     % Storing number of iterations
    Fit(i) = F(N+1);                % Storing cost function values
end

end
```

startguess.m

```
function Kin = startguess(Kint,v)
% Non user function that given an intervall Kint for all n parameters
% returns v start guesses genereted within the given intervalls by
% using a uniform distrubution.
% Outputs are:
% Kin - matrix containg a random start-guesses
% within a given intervall for each parameter (each
% row corresponds to a start-guess parameter vector)        (n x a) matrix
% Inputs are:
% v - number of startguesses to generate                    (scalar)
% Kint - matrix containg the intervall for each parameter   (n x 2) matrix

n = length(Kint');              % Number of parameters
Kin = ones(v,n);                % Pre dimension matrix

% Generating Kin using uniform distrubution

for i = 1:1:n
    Kin(:,i) = unifrnd(Kint(i,1),Kint(i,2),v,1);
end

end
```

MQNR.m

```
function [K,N,A,F,Yhat] = MQNR_3(kin,I0c,ym,tm,wm,tint)
 % MQNR performs a modified Quasi-Newton Rapson optimization with
 % the purpose to find the minimum with respect to k for a least square
 % cost function, where the model response in the function is given by
 % solving a coupled system of ODE:s. If Quasi-Newton method cannot be
 % performed due to bad numerical characteristics of the hessian
 % MQNR will use steepest descent.
 %
 % Outputs:
 % K - parameter matrix containg K-value
 % for each iteratoin                                    (Nit+1 x n) matrix
 % Nit - number of iterations to find a minimum          (scalar)
 % A - sensitivity matrix                                (m x n) matrix
 % F - vector with cost function value
 % for  each iteration                                   (1 x Nit+1) matrix
 % Yhat - Matrix with model value for the outsignal at
 % the time points tm for each iteration                 (m x Nit+1) matrix
 % K - matrix with parameters values for
 % each iteration                                        (Nit+1 x n) matrix
 %
 % Inputs:
 % kin - start guess parameter vector                    (1 x n) matrix
 % I0c - Inital concentrations for each parameter in
 % the model                                             (1 x g) matrix
 % ym - time series data to calibrate the model          (1 x m) matrix
 % tm - coresponding time values for ym                  (1 x m) matrix
 % wm - corresponding weight for ym                      (1 x m) matrix
 % tint - interval for solving the ODE:s                 (1 x 2) matrix
 %
 % If MQNR fails to yield a minimum within 100 iterations an error message
 % will be displayed and the search will be terminated.

    % ---------------------- Non user ------------------------------ %
    n = length(kin);        % Number of parameters
    m = length(ym);         % Number of points in time serie
    k = kin;                % Initial parameter guess
    Delta = sqrt(eps);      % Tolerance when calculating sensitivities


    % Pre dimension of matrices
    A = zeros(m,n);             % Sensitiivty matrix
    yhat = zeros(1,m);          % Model values in measurment points

```

```
44      % Generating weight matrix W (m x m)
45      W = diag(wm);
46
47      % ------------------------ User -------------------------------- %
48      % Select tolerance value for the termination criteras
49      eps1 = 1e-5; eps2 = 1e-5; eps3 = 1e-5;
50
51      % ---------------------- Non user ------------------------------ %
52      % Arbitary values set to avoid termination of  search in first the
53      % iteration
54
55      diff = 100;
56      kit = 100 * ones(1,n);
57
58      for v = 0:1:100
59          N = v;                      % Number of iterations
60          K(N+1,:) = k;               % Storing k for iteration Nit in K
61
62          % ---------------------- User ------------------------------ %
63          % Select the chosen model for ODE45 to parameter estimate, for
64          % example writing [t,x] = ODE45(@modelN,tint,I0c) means
65          % that model N will be parameter estimated
66          [t,x] = ode45(@(t,x) model3(t,x,k),tint,I0c);
67
68          % ---------------------- User ------------------------------ %
69          % Generation of yhat in line 71, select the correct outsignal
70          % e.g component one divided with component 3 (x(:,1)./x(:,3))
71          for i=1:1:m
72              yhat(i) = interp1(t,x(:,2),tm(i));
73          end
74          Yhat(:,N+1) = yhat';              % Storing yhat value in Yhat
75
76          % Generation of the A-matrix
77          A = sensitivities3(yhat,tm,I0c,Delta,tint,k);
78
79          % ---------------------  Non User ----------------------------- %
80          % Generation of residuals (r), cost function and gradient for
81          % given k
82          r = ym - yhat;                % Residual vector
83          r2 = (ym - yhat).^2;          % Square residual vector
84          Fw = sum(r2);                 % v + 1 element in F
85          F((v+1)) = Fw;                % Inserting Fw in F
86          gradf = -2*(A')*W*r';         % Gradient of f
87          H = 2*(A')*W*A;               % Approximated hessian
88
89          % Termination of search when at least 2 of 3 criterias are true
90          % (less than tol)
91          a1 = norm(gradf,2);           % L2-norm gradient of cost function
92          a2 = norm((kit-k),2);         % L2-norm of step-size
93          a3 = abs(diff);               % Difference of cost-function between
94                                        % iterations
95
96          % Tolerance level termination critera 1,2 and 3
97          tol1 = eps1*(1 + abs(Fw));
98          tol2 = eps2*(1 + abs(Fw));
99          tol3 = eps3*(1 + norm(k,2));
100
101         Term = zeros(1,3);            % Matrix to determine termination
102         if a1 < tol1
103             Term(1) = 1;
104         end
105         if a2 < tol2
106             Term(2) = 1;
107         end
108         if a3 < tol3
109             Term(3) = 1;
110         end
111
112         if Term(1) == 1 && Term(2) == 1 && Term(3) == 1
113             disp('Termineringsvilkor 1,2 och 3')
```

```
            break
        elseif Term(1) == 1 && Term(2) == 1
            disp('Termineringsvilkor 1 & 2')
            break
        elseif Term(1) == 1 && Term(3) == 1
            disp('Termineringsvilkor 1 & 3')
            break
        elseif Term(2) == 1 && Term(3) == 1
            disp('Termineringsvilkor 2 och 3')
            break
        else
        end

        % Choice of descent method between Newton (TD = 1) or steepest
        % descent (TD = 0)

        TD = Testdes(H);

        % Determination of descent direction p (depends on TD)

        p = Descentdir(TD,H,gradf);

        % Determination of serch length alpha

        diff = 1;
        alpha = 2;                  % Ensuring inital alpha to be 1
        kit = k;                    % k value before moving in k-space
        yhatstep = zeros(1,m);      % Pre dimension before the loop



        while diff > 0
            alpha = alpha/2;
            k = kit + alpha*p';

            % ---------------------- User ------------------------------ %
            % Choose the right model and outsignal, se line 56 and 71

            [t1,x] = ode45(@(t1,x) model3(t1,x,k),tint,I0c);

            for j = 1:1:m
                yhatstep(j) = interp1(t1,x(:,2),tm(j));
            end

            % ---------------------- Non user -------------------------- %
            rnew = (ym - yhatstep).^2;       % Residuals with new k-value
            Fnew = sum(rnew);                % Cost-value with new k-value
            diff = Fnew - F((v+1));
            if isnan(diff)                   % Checking if diff = NaN
                diff = 1;
            end
        end
        % Displaying a message if MQNR performs 100 iterations

        if N == 100
            disp('Terminering genom 100 iterationer')
        end

    end

end
```

model.m

```
function dx = model3(t,x,k)
% Model function which contains the ODE:s for modelT (test model). Each ODE
% corresponds to a given component in the model, for which component
% coresponds to x(i) and which parameter corresponds to k(i) se file
```

```
% measurement data modelT (MDMT.m).



% ------------------------ User -------------------------------- %
% Formulate the g differential equations in the following format
% (one for each row):
% dx1 = k(i)*x(i)+... ;
% dx2 = k(j)*x(j) + ... ;
% Then put them all in a vector with format dx = [dx1 dx2 ... dxg]';
% Don't miss the ' in the end of dx (else the code won't work).

v1 = k(1)*x(1)-k(2)*x(2);
v2 = k(3)*x(1)-k(4)*x(2);
v3 = k(5)*x(2)-k(6)*x(3);
v4 = k(7)*x(3)-k(8)*x(4);
v5 = k(9)*x(2)-k(10)*x(5);
v6 = k(11)*x(5)-k(12)*x(6);

dx1 = -v1-v2;
dx2 = v1+v2-v3-v5;
dx3 = v3-v4;
dx4 = v4;
dx5 = v5-v6;
dx6 = v6;

dx = [dx1 dx2 dx3 dx4 dx5 dx6]';

end
```

sensitivities.m

```
function A = sensitivities3(yhat,tm,I0c,Delta,tint,k)
% User function that calclulates the A-matrix containing all sensitivites
% through finite differences (one finite difference for each k-value). When
% calculating the finite differences a step length of Delta is used in the
% parameter room
% Inputs are:
% yhat - simulated outsignal in time points t1,t2,...,tm    (1 x m) matrix
% I0c - see MQNR                                            (1 x g) matrix
% tm - see MQNR                                             (1 x m) matrix
% k - parameter values for iteration Nit                    (1 x n) matrix
% tint - see MQNR                                           (1 x 2) matrix
% Delta = step length in the parameter room                 scalar
% Outputs are:
% A - sensitivity matrix                                    (m x n) matrix

% ---------------------- Non user ------------------------------ %


n = length(k);              % Number of parameters
k0 = k;                     % Inital k-value before moving in k-space
m = length(tm);             % Number of measurement points

% Pre dimension needed to calculate A

B = ones(m,n);
C = ones(m,n);

for i=1:1:n
    k = k0;                         % Resetting k for each iteration
    k(i) = k0(i) + k0(i)*Delta;     % Moving in k_i space

    % -------------------- User ---------------------------------%
    % Select the correct model to evaluate with ODE45 (se MQNR line 58
    % for further explanation)

```

```
    [t,x] = ode45(@(t,x) model3(t,x,k),tint,I0c);

    % Select the correct outsignal (se MQNR line 65 for further
    % explanation)

    for j = 1:1:m
         B(j,i) = interp1(t,x(:,2),tm(j));
    end
end

% ------------------- Non user --------------------------------%

% Generating c-matrix

k = k0;                  % Resetting k-value

for i = 1:1:n
    C(:,i) = yhat';
end

% Calculation of A

A = (1/Delta)*(B - C);

end
```

Definitest.m

```
function T = Definitetest3(H)
% Function that tests if a real n x n symmetrical matrix is positive
% definite through investigating the eigenvalues. If all eigenvalues
% are positive in a symmetrical matrix H it is positive definite.
% Inputs
% H - symmetrical approximated hessian          (n x n) matrix
% Outputs
% T = 0 (false) if H is not positve definite
% T = 1 (true) if H is positive definite

e = eig(H);                          % Column vector with eigenvalues of H

% T = 1 if there aren't any negative eigenvalues, else T = 0

T = isempty(find(e' < 0, 1));

end
```

Testdes.m

```
function [TD] = Testdes(H)
% Non user function that determines descent method, choosing between
% Newtons method or steepest descent.
% Inputs are:
% H - approximated hessian                                  (m x n) matrix
% Outputs are:
% TD = 0 --> uses steepest descent
% TD = 1 --> uses Newtons method
% p - normalized search direction                           (n x 1) matrix
% TD = 0 if H (approximated Hessian) is: non invertible, not positive
% definite or ill conditioned.
Hin = inv(H);                  % H inverse
clc

% ---------------------- Definte test ---------------------------- %
% T(1) = 1 H is positive definite, T(1) = 0 H is not positive definite

T(1) = Definitetest3(H);
```

```
T(1) = 1;

% ---------------------- Invertible test (inf) ---------------------- %
Tinf = isinf(Hin);            % Check if Hin contains inf

% T(2) = 1 H is invertible (contains no inf), T(2) = 0 Hin contains inf

T(2) = isempty(find(Tinf == 1,1));

% ----------------------- Ill condition test ------------------------- %
% This test uses the condition number, H is ill conditioned if eps <
% cond(H)

% If H isn't ill conditioned T(3) = 1 else if ill conditioned T(3) = 0

eps = 100;
K = cond(H);

if K > eps
    T(3) = 0;
else
    T(3) = 1;
end

if T(1) == 1 && T(2) == 1 && T(3) == 1
    TD = 1;
else
    TD = 0;
end

end
```

Descentdir.m

```
function p = Descentdir(TD,H,gradf)
% Non user function that calculates normalized descent direction p thorugh
% Newtons method (if TD = 1) or steepest descent (if TD = 0).
% Inputs are:
% TD - value determining method (se Testdes.m)
% gradf - approximated gradient of test function
% A - sensitivity matrix


if TD == 1
    p = H\(-gradf);
    p = p/norm(p,2);            % Normalizing p
elseif TD == 0
    p = -gradf;
    p = p/norm(p,2);            % Normalizing p
end

end
```

plotcyhat.m

```
% Non user file that plots the cost function f(k) for each iteration for
% the best start guess. Plotcyhat also plots the simulated outsignal for
% each iteration with a color gradient over the iterations (goes from blue
% to orange)

% -------------------------- Non user ------------------------------ %

Iter = 1:1:N+1;           % Vector with number of iterations
% Ploting the cost function f(k) over each iteration
figure(1)

```

```
plot(Iter,F,'--o','linewidth',1.5), grid on
title('Kostnadsfunktion'), xlabel('Iteration'), ylabel('Kostnadsfunktion')

% Plotting simulated outsignal yhat for each iteration
figure(2)

% Select color gradient (RGB) for plotting the simulated outsignal for each
% iteration (select start and end value for each color)

Rstart = 103; Rend  = 239;              % Red
Gstart = 169; Gend  = 138;              % Green
Bstart = 207; Bend  = 98;               % Blue

R = linspace(Rstart,Rend,N+1)./255;
G = linspace(Gstart,Gend,N+1)./255;
B = linspace(Bstart,Bend,N+1)./255;

% -------------------------- Non user ------------------------------ %

for i = 1:1:N+1
    p = plot(tm,Yhat(:,i),'linewidth',1.5); hold on, grid on
    xlabel('Tid'), ylabel('Koncentration'),
    title('Parameteruppskattning, Output = x2')
    c = p.Color;
    p.Color = [R(i) G(i) B(i)];
end

c = colorbar('Ticks',[]);                   % Removing markers on colormap
c.Label.String = '\fontsize{12} Number of iterations \rightarrow';

plot(tm,ym,'black+','linewidth',2.0)        % Plotting measurment data ym

clear R G B Rstart Rend Gstart Gend Bstart Bend p c i
```

Identifiability.m

```
% Non-user file that calculates the lower bound for the coavariance matrix
% for the parameters k1,k2,...,kn via the Fisher informaiton matrix (FIM)
% which is approximated as inverse(H). From Cov identifiability calculates
% then calculates correlation matrix and coefficeients of varance for each
% paramater

FIM = H;                % Calculating Fisher information matrix (FIM)
Cov = inv(FIM);         % Calculating the covariance matrix (Cov) via FIM

% Calculating correlation matrix (Cor) via Cov

Cor = corrcov(Cov);

% Calculating coefficient of variation (CCV) for each parameter

D = diag(Cov);          % Diagonal of Cor for the parameters
n = length(D);          % Number of parameters
CCV = ones(1,n);        % Pre dimension

for i = 1:1:n
    CCV(i) = sqrt(D(i))/k(i);
end

% Displaying result

disp('The covariance matrix is:')
disp(Cor)

disp('The correlation matrix is:')
disp(Cor)

disp('The coefficients of variation are:')
```

```
disp('      k1        k2')
disp(CCV)

clear i D n
```

sresiduals.m

```
% Non user File that plots standarized residuals (ym - yhat) for the
% best parameter vector.  Besides residuals the plots contain 95% and 99%
% confidence limits (red-lines) using the t-distrubution and a line. If the
% measurement data does not have a constant variance the assumptions
% for the confidence-limits doesn't hold up and the data should be weigthed
% or transformed.


yhat = Yhat(:,N+1)';        % yhat-values for optimized parameter vector
r = ym - yhat;              % Residual vector
SSE = sum(r.^2);            % Sum of squares for residual error
n = length(k);              % Number of parameters
m = length(ym);             % Number of measurements points
df = m - n;                 % Df of residual error
MSE = SSE/df;               % Estimation of residual variance
sr = r./MSE;                % Standarized residuals

% Calculating confidence limits for standarized residual plots

alpha99 = tinv(0.995,df);   % 99% confidence limit t-distribution
alpha95 = tinv(0.975,df);   % 95% confidence limit t-distribution


% Generating standarized residual plots:
figure(3)

% Residuals vs time
subplot(1,2,1)
plot(tm,sr,'blacko','MarkerFaceColor','black'), hold on
% Plotting the residuals
m = max(tm); x = 0:0.1:m;
plot(x,zeros(length(x)),'black--','linewidth',1.5), hold on
% Plotting the line r = 0
plot(x,alpha99*ones((length(x))),'red--','linewidth',1.5), hold on
% Plotting 99% upper confidence limit
plot(x,-alpha99*ones((length(x))),'red--','linewidth',1.5), hold on
% Plotting 99% lower confidence limit
plot(x,alpha95*ones(length(x)),'red--','linewidth',1.5), hold on
% Plotting 95% upper confidence limit
plot(x,-alpha95*ones(length(x)),'red--','linewidth',1.5), hold on
% Plotting 95% lower confidence limit
xlabel('Tid [min]'), ylabel('r_i'),
title('Standariserade residualer vs tid')

% Residuals vs model value
subplot(1,2,2)
plot(ym,sr,'blacko','MarkerFaceColor','black'), hold on
% Plotting the residuals
m = max(ym); x = 0:0.1:m+0.1;
plot(x,zeros(length(x)),'black--','linewidth',1.5), hold on
% Plotting the line r = 0
plot(x,alpha99*ones((length(x))),'red--','linewidth',1.5), hold on
% Plotting upper 99% confidence limit
plot(x,-alpha99*ones((length(x))),'red--','linewidth',1.5), hold on
% Plotting lower 99% confidence limit
plot(x,alpha95*ones(length(x)),'red--','linewidth',1.5), hold on
% Plotting upper 95% confidence limit
plot(x,-alpha95*ones(length(x)),'red--','linewidth',1.5), hold on
% Plotting lower 95% confidence limit
xlabel('Tid [min]'), ylabel('r_i'),
title('Standariserade residualer vs y_{hat}')
```

```
clear alpha95 alpha99 df MSE SSE sr yhat x m n
```

Concplot.m

```
% File that plots the concentration over time for selected (by the user)
% components in the model for an optimized parameter-vector k.

% -------------------------- Non user ------------------------------ %
% Generating the concentrations in tint for optimized k-vector
[t,x] = ode45(@(t,x) model3(t,x,k),tint,I0c);

% ---------------------------- User -------------------------------- %
% Select which concentrations to plot (writing plot(t,x(:,1)) plots
% component one) and the structure of the subplot. For each component
% correctly name the title

figure(5)
subplot(2,3,1)
linewidth = 2;                                  % Select line-width

% Plotting component 1
subplot(2,3,1)
plot(t,x(:,1),'b', 'linewidth',linewidth)
title('x_1')                                    % Title of plot
xlabel('Tid [min]'), ylabel('Konc'), grid on

% Plotting component 2
subplot(2,3,2)
plot(t,x(:,2),'b', 'linewidth',linewidth)
title('x_2')                                    % Title of plot
xlabel('Tid [min]'), ylabel('Konc'), grid on

% Plotting component 3
subplot(2,3,3)
plot(t,x(:,3),'b', 'linewidth',linewidth)
title('x_3')                                    % Title of plot
xlabel('Tid [min]'), ylabel('Konc'), grid on

% Plotting component 4
subplot(2,3,4)
plot(t,x(:,4),'b', 'linewidth',linewidth)
title('x_4')                                    % Title of plot
xlabel('Tid [min]'), ylabel('Konc'), grid on

% Plotting component 5
subplot(2,3,5)
plot(t,x(:,5),'b', 'linewidth',linewidth)
title('x_5')                                    % Title of plot
xlabel('Tid [min]'), ylabel('Konc'), grid on

% Plotting component 6
subplot(2,3,6)
plot(t,x(:,6),'b', 'linewidth',linewidth)
title('x_6')                                    % Title of plot
xlabel('Tid [min]'), ylabel('Konc'), grid on
```

PCA$_k$.m

```
% Non user file that creates a PCA-plot of Kit (matrix containg the
% parameter values for each iteration for all start-guesses). This file
% projects the n-dimensional data in Kit on PC1 and PC2 and connects the
% the iteraions for each start guess in the plot with a gradient line
% (so each line going from a start to an end color with increasing
% iterations corresponds to a specific start-guess).
% PCA_k also calculates Kf, a (v x n) vector containing the final k-vector
```

```
% for each start guess

% Calculating score values
[coeff,score,latent] = pca(Kit);

% Creating index matrix E, E(i+1) corresponds to the row for the final
% parameter vector (k-vector) for start guess number i in Kit
E = zeros(1,v+1);

for i = 1:1:v
    E(i+1) = sum(Nit(1:i))+i;
end

% Calculating Kf, vector containing final k-vector for each start guess
Kf = ones(v,length(k));                % Pre dimension (a x n)
for i = 1:1:v
    Kf(i,:) = Kit(E(i+1),:);
end

% Plotting each start guess as projected down on PCA1 and PCA2
figure(6)

% Select start and end color (RGB) for PCA-plot
Rstart = 103; Rend  = 239;              % Red
Gstart = 169; Gend  = 138;              % Green
Bstart = 207; Bend  = 98;               % Blue

% Marker plot ('o') on PC1 and PC2 for each start guess (each markers
% corresponds to an iteration value of k for a start guess)
for i = 1:1:v
    % Calculating score-values on PC1 (x) and PC2 (y) for start guess i
    x = score((E(i)+1):(E(i+1)),1);
    y = score((E(i)+1):(E(i+1)),2);
    L = length(x);                          % Number of colors in gradient
    R = linspace(Rstart,Rend,L)./255;
    G = linspace(Gstart,Gend,L)./255;
    B = linspace(Bstart,Bend,L)./255;
    cmap = [R' G' B'];                      % Creating colormap
    % Plotting each marker in a different color in the colormap
    hold on
    for j=1:1:L
        plot(x(j),y(j),'o','color',cmap(j,:),'MarkerSize',8, ...
            'linewidth',1.5),
    end
end

S = 5;                            % Number of plot points in each segment

% Connecting the markers with a gradient line for each start-guess
for i = 1:1:v
        % Calculating score-values on PC1 (x) and PC2 (y) for start-guess i
        x = score((E(i)+1):(E(i+1)),1);
        y = score((E(i)+1):(E(i+1)),2);
        L = length(x);                              % Number of colors
        R = linspace(Rstart,Rend,L-1)./255;
        G = linspace(Gstart,Gend,L-1)./255;
        B = linspace(Bstart,Bend,L-1)./255;
        cmap = [R' G' B'];                          % Creating colormap
        % Generating interpolated x- and y-segments for gradient line
        xi = ones(L-1,S); yi = ones(L-1,S);
        for j = 1:1:L-1
            xi(j,:) = x(j)+linspace(0,1,S)*(x(j+1) - x(j));
            yi(j,:) = y(j)+linspace(0,1,S)*(y(j+1) - y(j));
        end
        % Connecting the markers for start-guess i with line segments
        for j = 1:1:L-1
            hold on
            plot(xi(j,:),yi(j,:),'color',cmap(j,:),'linewidth',2)
        end
end
```

```
78
79 grid on, xlabel('PC_1'), ylabel('PC_2'),
80 title('PCA-plot different start guesses')
81
82 c = colorbar('Ticks',[]);            % Removing markers on colorbar
83 c.Label.String = '\fontsize{12} Number of iterations \rightarrow';
84
85 clear x y xi yi E i j S c L E
86 clear cmap R G B Rstart Rend Gstart Gend Bstart Bend
```