CHALMERS

GÖTEBORGS UNIVERSITET

Stokastiska processer för evolution

En studie för att konstruera och analysera modeller för evolution

Stochastic Processes for Evolution

Kandidatarbete inom civilingenjörsutbildningen vid Chalmers

Albin Blomster
Terese Kärnell
Nils Ledin
Linda Strandberg

Institutionen för Matematiska vetenskaper
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Göteborg, Sverige 2024

Stokastiska processer för evolution

En studie för att konstruera och analysera modeller för evolution

Kandidatarbete i matematik inom civilingenjörsprogrammet Kemiteknik vid Chalmers
Linda Strandberg

Kandidatarbete i matematik inom civilingenjörsprogrammet Teknisk matematik vid Chalmers
Albin Blomster Terese Kärnell Nils Ledin

Handledare: Philip Gerlee

Institutionen för Matematiska vetenskaper
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Göteborg, Sverige 2024

Förord

Detta kandidatarbete är utarbetat vid institutionen för Matematiska vetenskaper på Chalmers tekniska högskola och Göteborgs universitet. Vi vill tacka vår handledare Philip Gerlee för handledningen som han givit genom projektets gång.

Projektet har genomförts främst genom veckovisa möten med både gruppen och handledaren där projektets utformning diskuterades. Nedan följer en bidragsrapport som ger en mer detaljerad beskrivning av huvudförfattarna för varje avsnitt.

Bidragsrapport

§	Rubrik	Författare
	Förord	Albin, Linda
	Populärvetenskaplig presentation	Linda
	Sammandrag	Linda
	Abstract	Linda
1	**Inledning**	Albin, Nils
1.1	Syfte	Albin
1.2	Problemformulering	Albin, Linda
1.3	Avgränsingar	Albin
1.4	Samhälleliga och etiska aspekter	Terese, Nils
2	**Teori**	Terese
2.1	Markovkedjor	Terese
2.2	Kolmogorovs framåtekvationer	Terese
2.3	Modell för obegränsad population	Nils, Terese
2.3.1	Selektion vid obegränsad population	Albin, Terese
2.4	Moranprocessen	Linda
2.4.1	Selektionens påverkan på Moranprocessen	Linda, Terese
2.4.2	Analytisk lösning för Moranmodellen	Nils
2.5	Den kombinerade modellen	Terese
3	**Metod**	Albin
3.1	Gillespies algoritm	Terese, Nils
3.2	Implementation av modellerna	Terese
3.3	Felberäkning	Nils
4	**Resultat**	Nils
4.1	Obegränsad tillväxt	Albin, Nils
4.1.1	Simuleringar av den obegränsade modellen	Albin, Nils
4.1.2	Felberäkning av den obegränsade modellen	Albin, Nils
4.2	Moranmodellen	Albin, Nils
4.2.1	Jämförelse av fixeringssannolikhet för Moranmodellen	Albin, Nils
4.2.2	Stora populationer för Moranmodellen	Nils
4.3	Den kombinerade modellen	Albin, Nils
4.3.1	Jämförelse av fixeringssannolikhet för den kombinerade modellen	Albin, Nils
5	**Diskussion**	Albin
5.1	Resultat	Albin
5.2	Metodval och avgränsningar	Albin
6	**Slutsats**	Albin, Linda
A	**Appendix – Notationslista**	Linda
B	**Appendix – Härledningar**	Terese
B.1	Fixeringssannolikhet av den större populationen	Terese
B.2	Väntevärden och varians av slumpvariabler	Albin, Terese
C	**Appendix – Figurer**	Nils
C.1	Obegränsade modellen	Nils
C.2	Moranmodellen	Nils
C.3	Den kombinerade modellen	Nils
D	**Appendix – Källkod**	Albin, Nils

Övriga bidrag

Utöver skrivandet av rapporten förekommer även arbete som inte innefattas av bidragsrapporten. Dessa övriga bidrag till projektet beskrivs nedan.

Planering

Planering av arbetet gjordes tidigt i processen i samband med skrivandet av planeringsrapporten. Därtill har gruppen haft regelbundna veckomöten både med och utan handledare. Under gruppmötena diskuterades arbetet som utförts den föregående veckan samt arbetet som skulle göras veckan efter och vem som förväntades göra vad. För att enklare kunna reflektera under arbetets gång fördes även mötesanteckningar av personen som var veckans dagboksansvarig, vilket roterades mellan gruppens medlemmar. Dagboken består av utdrag kring vad gruppmedlemmarna gjorde både gemensamt och individuellt för varje vecka. Därutöver fördes även individuella tidsloggar, vilka består av beskrivningar av arbetsuppgifter samt de tider som det tog för att genomföra uppgifterna. Tillsammans utgör dagboken och tidsloggen projektets loggbok.

Genomförande

Arbetet började med att gruppen läste boken *Evolutionary Dynamics* av M.A. Nowak [1] samt föreläsningsanteckningarna *Lecture Notes on Stochastic Processes with Applications in Biology* av D.F Anderson [2] för att skapa en grundläggande förståelse kring området. Sedan lades fokus på att implementera Gillespies algoritm för den obegränsade modellen för en genotyp, vilket Albin gjorde. Därefter hjälptes alla i gruppen åt att skapa Gillespies algoritm för två genotyper. Samtidigt härledde Nils och Terese uttrycken för den stokastiska obegränsade modellens medelvärde 2.5 och varians B.5. Sedermera diskuterades hur vissa delar av resultatet av den obegränsade modellen skulle presenteras. Då kom gruppen tillsammans med handledaren fram till att presentera resultatet som i avsnitt 4.1.2. Nils utvecklade sedan koden så att den blev mer objektorienterad och fick en snabbare exekveringstid. I rapporten skapade Linda figur 1 för att visualisera Moranprocessen. Därefter skrev Albin och Nils koden för Moranprocessen och den kombinerade modellen samt skapade figurerna till modellerna. I och med detta reviderades planeringsrapporten då den kombinerade modellen lades till som modell. Gruppen valde sedan efter inlämning av den reviderade planeringsrapporten att inte undersöka resultatet som modellerna ger då genotyperna har identiskt delningstakt på grund av platsbrist. Det vill säga det ansågs mer intressant och givande att undersöka den kombinerade modellen än att undersöka identisk delningstakt hos alla tillväxtmodeller. Rapporten har också genomgått ett antal revideringar som alla i gruppen har bidragit till. Rapporten har främst revideras efter fackspråkshandledning 1 och 2 samt efter de två utkast som skickades till handledaren.

Utöver de källor som handledaren gav så har även Albin och Terese hittat källor som förstärker projektet i avsnitt 1, 2, 3 och 5.

Diskussionsbidrag

Vid skrivandet av diskussionen så hade gruppen ett specifikt möte där det gemensamt diskuterades och reflekterades över de resultat som hade uppnåtts. På mötet diskuterades även hur resultaten kunde kopplas till teori, källor, avgränsningar med mera för att skapa en så allsidig diskussion som möjligt.

Populärvetenskaplig presentation

Det välkända uttrycket "survival of the fittest" kan sammanfatta evolutionsteorins fader Charles Darwins teorier som rör arters utveckling. Gregor Mendel är ett annat stort namn inom området som under samma tidsperiod forskade på exakt hur individer ärver egenskaper från sina föräldrar. Mendel och Darwin lade grunden till det vi idag kallar för evolution vilket var startskottet till Fishers, Haldanes och Wrights utveckling av populationsgenetiken, som möjliggjordes på grund av upptäckten av DNA-molekylen under 1900-talet.

Evolutionsteorin säger att den individ som är bäst anpassad till sin omgivning har störst chans att föröka sig och därmed föra vidare sina fördelaktiga egenskaper. En matematisk modell som vill beskriva hur en population förändras över tid kan då utformas på två sätt. Antingen kan man göra en modell där bättre anpassade arter *alltid* utkonkurrerar de sämre anpassade individerna, detta kallas för en deterministisk modell. Eller så kan man ta hänsyn till slumpen: även om en individ har bättre egenskaper än andra individer så kan den fortfarande till exempel dö av en spontan olycka som gör att egenskaperna inte förs vidare till nästa generation. Sådana modeller kallas för stokastiska. Eftersom stokastiska modeller ger många fler möjliga sätt för en population att förändras, medan deterministiska modeller endast ger en möjlighet, så brukar stokastiska modeller vara svårare att analysera.

Biologiska system finns i många varianter och är mycket komplexa, så för att kunna skapa begripliga modeller måste vi förenkla dem på flera sätt. Till att börja med finns det många olika sätt för djur och växter att fortplanta sig. Majoriteten av alla djur inklusive människor har dubbla uppsättningar av sina gener och behöver finna en partner för att en avkomma ska bli till. De flesta bakterier däremot har bara en kopia av sin arvsmassa och kan fortplanta sig själva genom delning. I denna rapport har den sistnämnda varianten analyserats. Även organismers beteenden är mycket varierade och komplexa. För att förenkla modellerna kommer vi att samla alla beteenden och egenskaper hos en individ i ett enda tal, tillväxthastigheten, eller delningstakten, λ. Detta är ett mått på hur snabbt en individ kan fortplanta sig i jämförelse med andra individer. En tredje aspekt som också påverkar resultatet är mutationer: förändring i cellernas genetiska material. Vi kommer i de flesta fall anta att vi har en population som består till större delen av "vanliga" individer och sedan en individ med muterad arvsmassa, som ges antingen en större eller mindre tillväxthastighet. Vi kommer sedan studera hur stor sannolikheten är att den individen lyckas föröka sig så mycket att alla framtida individer härstammar ur den ursprungliga muterade individen.

Vi kommer studera tre olika populationsmodeller. Den första kan ses modellera ohämmad bakterietillväxt. Det kommer inte finnas något tak på antalet individer och alla individer tillåts att fortplanta sig fritt utan att dö. De andra två modellerna är utformade efter ekosystem som redan är fyllda av en population och inte klarar av att bära fler individer. I den ena kommer populationsstorleken vara helt konstant över tid, så att när en individ föds så dör en annan automatiskt. Den andra modellen kommer tillåta viss variation i populationsstorleken genom att göra födsel och död oberoende av varandra och istället introducera ett tak på hur stor population systemet klarar, en så kallad bärkapacitet på systemet.

Den första modellen som undersöktes begränsades inte av maximal populationsstorlek som tidigare nämnt. Detta resulterade i en obegränsad tillväxt. I simuleringarna som utfördes varierades populationens tillväxthastigheter, sluttider samt startpopulationer där startpopulationen för de två olika populationerna var av olika storlekar. Oavsett storleken på de initiala populationsstorlekarna uppvisade de en deterministisk lösning, det vill säga att den följer "survival of the fittest". Vidare undersökning av de stokastiska och analytiska lösningarna visade även att avvikelserna ökade exponentiellt med lägre sluttider, högre tillväxthastigheter och startpopulationer.

För den andra modellen antas det att populationen förändras i stegvisa tidpunkter. Där väljs i varje steg en individ slumpmässigt som får en avkomma och dess avkomma ersätter sedan en annan

individ i populationen, alltså är populationsstorleken vid varje tidssteg konstant. Simuleringarna visade till skillnad från den obegränsade modellen att vardera realisering inte ligger nära det medelvärdet som fås av simuleringarna. Realiseringarna uppvisar istället Moranprocessens dynamik, antigen så tar den ena genotypen över eller så dör den ut och ur detta fås ett medelvärde kring andel övertagande som närmar sig den analytiskt framtagna sannolikheten.

Den tredje och sista modellen har vi benämnts som den kombinerade modellen. Detta eftersom det är en kombination av den första och andra modellen. Denna modell visade en långsammare övertagnings hastighet i jämförelse med den andra modellen. Simuleringarna av den kombinerade modellen visar att fixeringssannolikheten ökar långsammare än för Moranprocessen, detta på grund av bärkapaciteten.

Slutsatsen av denna undersökning visar på att den förstnämnda, obegränsade modellen ger en rimlig stokastisk och deterministisk lösning. Eftersom den deterministiska modellen är enklare att använda sig av kan den föredras över den stokastiska för den obegränsade modellen. För den modellen som nämndes därefter, med diskreta steg gäller det att den deterministiska modellen skiljer sig betydande från de individuella stokastiska körningarna. Men den deterministiska modellen bidrar dock med att säga hur sannolikt det är att en genotyp tar över en population, vilket ändå beskriver en viktig del av modellens dynamik. För den sista modellen, den kombinerade modellen, gäller liknande resonemang som för den andra modellen. Dock så är den kombinerade modellen mer realistisk eftersom genotyperna tillåts både konkurrera mot varandra och miljöns begränsningar. Totalpopulationen tillåts även fluktuera, vilket skapar en mer realistisk dynamik. Den kombinerade modellen är betydligt mer komplex och visar i vårt fall en dynamik som liknar den andra modellen, därför är den enklare modellen att föredra.

Forskningen om hur populationer beter sig är extra aktuellt i dagens samhälle då vi precis tagit oss ur Covid-19-pandemin, där forskare och experter över hela världen studerade smittspridningen av viruset. Genom att exempelvis studera smittspridarens beteende som rörelsemönster, sociala interaktioner och användning av skyddsåtgärder, har modeller kunnat utvecklas med mål att förutsäga och förhindra spridningen av viruset. Förutom inom virusspridning kan populationsforskningen appliceras på att studera utrotningshotade arter och inom bevarandebiologin. Genom att analysera utrotningshotade arter kan förståelse uppnås för deras behov och levnadsätt. I analysen av dess populationstrender, fortplantningsmönster och preferenser av habitat kan strategier arbetas fram för att kunna bevara de utrotningshotade arterna. Forskningen är också vital inom hållbar utveckling och samhällsplanering genom att analysera människan själv. Hur människan interagerar med sin miljö, exempelvis energianvändning, avfallshantering och konsumentmönster för att skapa en bild av hur den negativa effekten på planeten kan förminskas. Detta är bara några få exempel på varför populationers beteende behöver studeras i fortsättningen för att kunna förutse samt jobba för att förstå och förbättra dagens samhälle.

Sammandrag

Under 1800-talet i samband med Darwins studier myntades begreppet evolution genom naturligt urval. Studierna undersökte arters förmåga att anpassa sig till omgivningen och hur viktig artens ärftlighet var. Dessa idéer förenades sedan under 1900-talet med Mendels studier av ärftlighet och resulterade i Fishers, Haldanes och Wrights utveckling av populationsgenetiken. Syftet med detta projekt är att studera evolution genom olika modeller. De modeller som kommer undersökas är stokastiska eller deterministiska, har begränsade eller obegränsade populationsstorlekar samt varierande parametrar som exempelvis startpopulation och delningstakt.

Den första modellen som undersöktes var obegränsad tillväxt, simuleringar visade på en tidig populationsvariation som närmar sig den deterministiska lösningen för båda genotypernas populationer. Det gick även att se att felen i simuleringar ökade med en ökande sluttid, tillväxthastighet och startpopulation. Den andra modellen som undersöktes var Moranprocessen och resultatet visade att till skillnad från den obegränsade modellen så ligger inte vardera realisering nära det medelvärdet som fås av simuleringarna. Den tredje och sista modellen som simulerades, den kombinerade modellen, visade på att den ena genotypen tog över mindre i jämförelse med Moranprocessen, detta på grund av bärkapaciteten.

Slutsatsen som kan dras för den obegränsade modellen är att dess deterministiska och stokastiska modell ger liknande resultat och därför är den deterministiska modellen att föredra då den är enklare. För Moranmodellen skiljer sig individuella stokastiska realiseringar från den deterministiska modellen och därför är de stokastiska realiseringarna att föredra. Den kombinerade modellens resultat liknar till stor del Moranmodellen men den kombinerade modellen är mer realistisk jämfört med de andra två modellerna då den tillåter konkurrens och bärkapacitet dock på bekostnad av en högre komplexitet. Därför är Moranmodellen att föredra mot den kombinerade modellen.

Abstract

During the 19th century, the studies of Darwin defined evolution. The main purpose of the studies was to observe different species' ability to adapt to their environment and the role of heredity. In the 20th century, Darwin's and Mendel's ideas resulted in the development of population genetics by Fisher, Haldane, and Wright. The purpose of this report is to study evolution and population growth through different models, including stochastic and deterministic models, with limited or unlimited population and varying parameters such as starting population and fitness.

The first model investigated was the impact of two genotypes on each other under unlimited growth. The simulations show an early population variation that eventually converges toward the deterministic solution for both genotype populations. It is also observed that the error in the simulation grows exponentially with increasing end time, growth rate, and initial population. The second model studied was the Moran model. Unlike the unlimited growth model, the simulations do not converge towards the deterministic average of the simulations. The simulations showed that either one genotype took over or it died out, and from this an average value around the share of takeover was obtained that approached the analytically produced probability. The third and final model simulated, the combined model, showed a slower takeover rate compared to the Moran model, due to the carrying capacity.

The conclusion drawn for the unlimited growth model is that the deterministic model is simpler but provides similar results to the stochastic simulations. For the Moran model, individual stochastic runs differ from the deterministic model. Thus stochastic simulations are preferred. The Moran model and the combined model are mostly alike but the combined model is more realistic compared to the other two models as it allows for competition and a carrying capacity. But it also entails a higher complexity, thus the Moran model is preferred between the Moran model and the combined model.

Innehåll

1 Inledning

Evolution är ett begrepp som myntades i och med att evolutionsteorin uppstod under 1800-talet mycket tack vare Charles Darwin [1]. Darwin samlade stora mängder data över hur olika arter hade anpassat sig till sin miljö [1]. Senare under 1900-talet sammanfördes dessa insikter med Mendels studier kring ärftlighet till en gemensam modern teori [1]. Ur denna teori utvecklade bland andra Fisher, Haldane och Wright populationsgenetiken [1]. De lyckades på 1920- och 30-talet matematiskt beskriva termer som evolution, selektion och mutation [1]. Med verktygen de utvecklade kunde det senare förklaras icke intuitiva fenomen såsom genetisk drift [1]. Vidare har även forskning gjorts som kopplar samman ekologi och evolution så kallad, "eco-evolutionary feedbacks" [3]. Där är tanken att ekologiska effekter driver på evolutionära processer som i sin tur driver på ekologiska effekter och så vidare [3]. Ett av de första exempel som modellerade denna "feedback" var selektion som beror på en populations densitet [4]. Det vill säga beroende på en populations densitet så blir selektionstrycket olika starkt [4]. Ett annat exempel på "eco-evolutionary feedbacks" är Lotka–Volterras ekvation som beskriver relationen mellan rovdjur och bytesdjur [3]. Där är predation ett ekologisk fenomen som driver på att bytesdjur genomgår evolutionära förändringar för att överleva, vilket i sin tur driver på responsen hos rovdjurssidan [5]. På senare tid har även modeller utvecklas som tar hänsyn till att evolution, som vanligtvis är en långsam process, och ekologiska processer, som är snabbare processer, kan ske under liknande tidsrymd [3]. Även värd-parasit förhållanden har modelleras historiskt [6], där det har visat sig att hur smittsamma sjukdomar är beror på densiteten hos värdpopulationen [6]. Syftet med denna rapport är att studera hur stor påverkan slumpen har i form av genetisk drift på evolution i olika tillväxtmodeller. För att kunna studera detta behövs viss terminologi inom evolution först introduceras.

Om en individ har en kopia av en gen kallas individen för haploid medan om den har två kopior av en gen så kallas den för diploid. Exempelvis är människor och de flesta däggdjur diploida organismer samtidigt som de flesta bakterier är haploida organismer [7]. I populationsdynamik är det ofta vanligt att en gen med olika genotyper analyseras för en viss population. Det man då undersöker är hur fördelningen av genotyperna, det vill säga de olika genvarianterna av en viss gen, förändras över tid. Därtill kan selektion liknas vid begreppet naturligt urval, eller den mer allmänt kända termen "fitness", som beskriver hur framgångsrik en organism är på att överleva i relation till organismer av samma art [1]. I matematiska modeller för evolution beskrivs selektion med hjälp av en delningstakt som betecknas med λ. Det vill säga om en genotyp har större λ-värde så har genotypen större fitness. Detta leder sedan till begreppet fixeringssannolikhet, vilket betecknar sannolikheten att en genotyp som startar med en individ tar över en population det vill säga sannolikheten att genotypen blir den dominerade genotypen i en population. Sedan är genetisk drift slumpbaserat och begreppet kan förklaras som summan av alla slumpmässiga effekter på evolution. Genetisk drift blir ofta tydlig i små populationer, vilket är en anledning till att slumpbaserade eller stokastiska modeller oftast används vid modellering av sådana populationer [8]. Till skillnad från stokastiska modeller finns deterministiska modeller som skapas antingen via enkla ordinära differentialekvationer eller som medelvärdet av flera stokastiska realiseringar. I mer realistiska modeller så kan varken populationer växa obegränsat då miljön begränsar tillväxten eller för den delen hållas konstanta då resurserna och miljön konstant förändras. Därför finns det tillväxtmodeller som utnyttjar en bärkapacitet som är ett tal K som modellerar miljöns begränsningar. Det vill säga om en genotyp växer sig större i en modell med bärkapacitet kommer reproduktionstakten minska på grund av bärkapaciteten.

Inom evolutionsteori brukar evolution definieras som förändringen av frekvenserna av olika genotyper i en population över tid. Evolution kan modelleras på flera sätt men i detta arbete undersöks hur fördelningen av genotyperna förändras över tid då tillväxten tillåtas vara obegränsad, populationsstorleken sätts till att vara konstant samt tillväxten sker med en bärkapacitet. Obegränsad tillväxt kan till exempel uppstå hos bakterier på en petriskål (en cylindrisk glasskål där bakteriekulturer odlas) tills bakterierna når kanten [9]. I kontrast sker tillväxt i en konstant population eller i en population med bärkapacitet i ett naturligt stabilt ekosystem [10], [11].

1.1 Syfte

I denna uppsats är syftet att undersöka vikten av genetisk drift i olika enkla modeller av evolution. Detta undersöks genom att jämföra stokastiska och deterministiska modeller för tre evolutionsmodeller; obegränsad tillväxt, evolution för en konstant population och evolution med bärkapacitet. Studien fokuserar även också på att undersöka modellernas långsiktiga beteende.

1.2 Problemformulering

I uppsatsen undersöks olika tillväxtmodellers egenskaper. För det första kommer frågan ställas hur modellerna skiljer sig åt mellan de deterministiska och stokastiska fallen. För det andra kommer det undersökas hur delningstakten, fördelningen på startpopulationen, antal realiseringar samt sluttiden påverkar den evolutionära dynamiken. Till sist kommer rapporten att undersöka de olika tillväxtmodellernas långsiktiga beteende antingen genom fixeringssannolikhet för modellen med konstant population och modellen med bärkapacitet eller så studeras frekvensen av en viss genotyp då tiden går mot oändligheten för den obegränsade modellen.

1.3 Avgränsningar

I detta projekt kommer enkla modeller för evolution att studeras som kan tillämpas på mer generella exempel, vilket begränsar de slutsatser som kan dras. De förenklingar som görs är att en population av haploida individer med två olika genotyper studeras i en konstant miljö. Dessutom antas det att de två genotyperna har olika stor delningstakt. Till sist kan systemet antingen tillåtas ha en population som växer obegränsat, konstant population eller en population med en viss bärkapacitet. Fixeringssannolikheten för den kombinerade modellen kommer ej tas fram utan vid vid jämförelse används moranmodellens fixeringssannolikhet ty projektet skall vara av rimlig komplexitet för vad som kan förväntas av ett kandidatarbete inom matematik.

1.4 Samhälleliga och etiska aspekter

I allmänhet är evolutionsprocessen en stor del av livet på jorden och berör såväl människor som djur och natur. Det bör beaktas att kunskap om hur sjukdomar sprids kan undersökas med hjälp av evolutionsteori och detta kan leda till positiva bidrag till samhället, ifall man hittar ett sätt att bota eller utrota sjukdomar.

Även kunskapen om cancertillväxt och hur den kan förhindras är viktig då cancer är vanligt förekommande och ibland en dödlig sjukdom. Cancertumörer utvecklas och kan spridas genom mutationer av celler i vävnad samt genom selektion av cancerceller som är bäst lämpade för miljön, det vill säga cancertillväxt kan modelleras som ett evolutionärt system [12]. Kunskapen kan leda till mer effektiva och säkrare behandlingssätt samt minska risken för återfall eller bildandet av resistenta cancerceller.

En annan aspekt är att klimatförändringar kan påverka individers levnadsförhållanden som i sin tur påverkar evolutionsprocessen. Då klimatförändringarna påverkar människors och djurs tillgång till mat och levnadsutrymme bidrar detta till förändrade förhållande för selektion, vilket påverkar jordens ekosystem.

Genom hela projektet utförs stora förenklingar i modelleringen. Dessa förenklingar begränsar vilka slutsatser som kan dras då verkliga system tenderar vara mer komplexa. Det är därför viktigt att rimliga anpassningar görs i arbetet med matematiska modeller av biologiska system för det system som studeras. Ett exempel på detta är hur reproduktionen går till i en art. Exempelvis kan vissa arter uppvisa i princip slumpmässig sexuell parning såsom fiskar i en sjö där hanarna släpper ut sina spermier över stora områden där honor har lagt sina ägg [13]. Jämför detta med en organism där individerna är separerade från varandra i mindre grupper, såsom djur som bor i en grupp av öar och därför endast kan para sig med sina närmsta grannar. Dessa två system kan inte modelleras på samma sätt. En liknande slutsats kan dras kring att i projektet modelleras haploida organismer och inte diploida organismer som har annorlunda egenskaper [14].

2 Teori

I detta avsnitt kommer definitioner och teori som är relevant för projektet att introduceras. Notationslista bifogas i bilaga A. I bilaga B bifogas härledningar som används i teorin.

Låt $X(t)$ vara populationsstorleken vid tiden $t \in \mathbb{R},\ t \geq 0$ och låt $i \in S$ där S är utfallsrummet som beskriver processens tillstånd, det vill säga i är antalet individer i populationen. Tillstånden är per definition diskreta. Vi antar ytterligare att delningstakten för varje individ är konstant och oberoende av populationsstorleken. Detta kommer gälla genomgående i projektet om inget annat anges.

2.1 Markovkedjor

En Markovkedja är en stokastisk process med egenskapen att den betingade sannolikheten för framtida tillstånd bara beror på det befintliga tillståndet och inte på tidigare tillstånd. I projektet kommer Markovkedjor i kontinuerlig tid att studeras.

Definition 2.1 *En kontinuerlig stokastisk process $X(t) \in S$ för $t \geq 0$ kallas en Markovkedja i kontinuerlig tid om det för varje $i \in S$ och $s < t$ gäller att*

$$\mathbb{P}(X(t) = i | X(r), 0 \leq r \leq s) = \mathbb{P}(X(t) = i | X(s)). \tag{2.1}$$

Om det dessutom gäller att

$$\mathbb{P}(X(t) = i | X(s) = k) = \mathbb{P}(X(t-s) = i | X(0) = k) \tag{2.2}$$

för något $s \in [0, t]$ och några tillstånd $i, k \in S$, kallas kedjan tidshomogen.

Alltså ifall det framtida tillståndet $X(t > s)$ bara beror på det befintliga tillståndet $X(s)$ är X en Markovkedja. Detta motsvarar att endast den nuvarande populationsstorleken påverkar hur populationen växer i varje kontinuerligt tidsintervall, och inte hur populationen såg ut tidigare. Fortsättningsvis i denna rapport kommer Markovkedja i kontinuerlig tid förkortas som CTMC (Continuous Time Markov Chain) [15].

Intensiteten för övergångarna mellan tillstånden i en CTMC kan definieras som

$$P(X(t+h) = j | X(t) = i) = q_{i,j}h + o(h) \quad i \neq j \tag{2.3a}$$
$$P(X(t+h) = i | X(t) = i) = 1 - q_{i,i}h + o(h) \tag{2.3b}$$

där $q_{i,j}$ är *övergångsintensiteten* mellan tillstånden i och j och $q_{i,i}$ är övergångsintensiteten för att stanna kvar i tillstånd i. $o(h)$ står för lilla ordo, vilket betecknar en funktion av h som avtar mycket snabbare än h självt [16].

En egenskap hos alla CTMCs är att tiden som kedjan befinner sig i ett visst tillstånd är exponentiellt fördelad. Låt övergångsintensiteten mellan tillstånden $i, j \in S$ vara $q_{i,j}$. Ett större $q_{i,j}$ motsvarar i genomsnitt en kortare tid för processen att gå från tillstånd i till j. Alla övergångsintensiteter kan samlas i en matris enligt följande definition.

Definition 2.2 *För en CTMC $X(t)$ med övergångsintensiteter $q_{i,j}$ kallas matrisen*

$$Q = \begin{bmatrix} -q_1 & q_{1,2} & q_{1,3} & \cdot & \cdot & \cdot \\ q_{2,1} & -q_2 & q_{,23} & \cdot & \cdot & \cdot \\ q_{3,1} & q_{3,2} & -q_3 & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & & \\ \cdot & \cdot & \cdot & & \cdot & \\ \cdot & \cdot & \cdot & & & \cdot \end{bmatrix} \tag{2.4}$$

med $q_i = \sum_{j \neq i} q_{i,j}$ för X:s övergångsmatris.

2.2 Kolmogorovs framåtekvationer

Ett verktyg för att studera Markovkedjor är Kolmogorovs framåtekvationer. Dessa är ett system av differentialekvationer (ODE:er) som beskriver förändringarna i sannolikheterna för kedjan att befinna sig ett visst tillstånd över tid. Kolmogorovs framåtekvationer kan lösas både analytiskt och numeriskt.

Definition 2.3 *Låt $P_i(t)$ vara sannolikheten att en Markovkedja befinner sig i tillstånd i vid tiden t, det vill säga*

$$P_i(t) = \mathbb{P}(X(t) = i). \tag{2.5}$$

Derivatan av $P_i(t)$ ges då av

$$\frac{dP_i}{dt}(t) = -q_i P_i(t) + \sum_{j \neq i} P_j(t) q_{j,i} \tag{2.6}$$

där $q_{i,j}$ är övergångsintensiteter och $q_i = \sum_{j \neq i} q_{j,i}$ [2].

Lemma 2.4 *Ekvation (2.6) kallas för Kolmogorovs framåtekvationer. Detta kan representeras som ett system av differentialekvationer på matrisform. Om vi skriver $P(t) = (P_0(t),\ P_1(t),\ P_2(t), ...)$ så gäller det att*

$$\frac{d}{dt}P(t) = QP(t) \tag{2.7}$$

Ekvation 2.7 kan då härledas som

$$\begin{aligned}\frac{d}{dt}P_i(t) &= -q_i P_i(t) + \sum_{y \neq i} P_y(t) q_{y,i} = P_i(t) Q_{i,i} + \sum_{y \neq i} P_y(t) Q_{y,i} \\ &= \sum_y P_y(t) Q_{y,i} = (P(t)Q)_i \quad .\end{aligned}$$

Lösningen till ovanstående ODE är [17]:

$$P(t) = P_0 e^{Qt}. \tag{2.8}$$

Kolmogorovs framåtekvationer ger alltså sannolikheten att systemet befinner sig i ett visst tillstånd, vilket kan användas för att bestämma väntevärdet av populationsstorleken i varje tidpunkt. Detta kan sedan användas för att jämföra den stokastiska modellen med den deterministiska.

Med den generella teorin som stöd så kan nu de specifika modellerna beskrivas som ligger till grund för simuleringarna.

2.3 Modell för obegränsad population

Idén bakom denna modell är att X beskriver antalet individer av en haploid art, där varje individ förökar sig med takt λ och aldrig dör. Eftersom varje individ har samma tillväxthastighet kommer tiden tills en ny individ föds ur en population med n individer vara exponentialfördelad med parameter $n\lambda$.

Då fås övergångsintensiteten med $q_{i,j} = i\lambda$ för $j = i + 1$. Kolmogorovs framåtekvationer säger för detta system att

$$\begin{aligned}&\frac{d}{dt}P_i(t) = \lambda(i-1)P_{i-1}(t) - \lambda i P_i(t), \quad i \geq 2 \\ &\frac{d}{dt}P_1(t) = -\lambda P_1(t).\end{aligned} \tag{2.9}$$

Lösningen till detta system är

$$P_i(t) = e^{-\lambda t}\left(1 - e^{-\lambda t}\right)^{i-1} \tag{2.10}$$

med

$$\begin{aligned} P_i(0) &= 0, \quad i \geq 2 \\ P_1(0) &= 1. \end{aligned} \tag{2.11}$$

När vi nu vet sannolikheten att befinna sig i varje tillstånd vid tiden t så kan vi ta fram väntevärdet på processen.

Härledning 2.5

$$\begin{aligned} \mathbb{E}[X(t)] &= \sum_{i=1}^{\infty} iP_i(t) = /\ \textit{Kolmogorov}\ / = \sum_{i=1}^{\infty} ie^{-\lambda t}\left(1-e^{-\lambda t}\right)^{i-1} \\ &= \left[x = 1-e^{-\lambda t}\right] = e^{-\lambda t}\sum_{i=1}^{\infty} ix^{i-1} = e^{-\lambda t}\sum_{i=0}^{\infty}\frac{d}{dx}x^i \\ &= /\ \textit{likformig konvergens}\ / = e^{-\lambda t}\frac{d}{dx}\sum_{i=0}^{\infty}x^i = /|x|<1,\ \textit{ty}\ \lambda t > 0\ / \\ &= e^{-\lambda t}\frac{d}{dx}\left(\frac{1}{1-x}\right) = e^{-\lambda t}\frac{1}{(1-x)^2} = e^{-\lambda t}\frac{1}{(1-(1-e^{-\lambda t}))^2} \\ &= \frac{e^{-\lambda t}}{e^{-2\lambda t}} = e^{\lambda t}. \end{aligned}$$

2.3.1 Selektion vid obegränsad tillväxt

Selektion, oftast kallat naturligt urval, uppstår när populationer förökar sig med olika hastigheter [1]. Det vill säga, populationerna har olika delningstakter. I detta projekt används som tidigare nämnt två genotyper och det som skall undersökas är fallet då två genotyper har olika delningstakter. Låt genotyperna kallas A och B som har delningstakt a respektive b och låt $x(t)$ vara antalet A individer och $y(t)$ vara antalet B individer vid tiden t. Vid obegränsad population ökar populationerna för genotyp A respektive B exponentiellt eftersom de inte påverkar varandra. Alltså kan populationerna modelleras för vardera genotyp på samma sätt som beskrivs i avsnitt 2.3 Den deterministiska modellen för ändringen i populationsstorlek vid tiden t blir då modellerat med hjälp av ett system av ODE:er

$$\begin{aligned} x'(t) &= ax(t) \\ y'(t) &= by(t) \end{aligned} \tag{2.12}$$

med lösning

$$\begin{aligned} x(t) &= x_0e^{at} \\ y(t) &= y_0e^{bt}. \end{aligned}$$

Här gäller det att A förökar sig fortare om $a > b$ och B förökar sig fortare om $b > a$ och x_0 respektive y_0 är startpopulationer för genotyp A respektive B. För en obegränsad population med två genotyper innebär detta alltså att båda sorters genotyper kommer öka men i olika takt.

Vi kan även välja att studera hur proportionen av populationerna utvecklas över tid. Som tidigare så beaktas två genotyper A och B vars storlek är $x(t)$ respektive $y(t)$. Då definieras andelen av A vid tiden t som

$$\rho(t) := \frac{x(t)}{x(t)+y(t)}$$

och andelen av B vid tiden t blir således $1-\rho$. Ifall man inte har en analytisk lösning av x, y så finner man ett uttryck för ρ genom att derivera uttrycket och lösa

$$\begin{aligned} \rho' &= \frac{x'(x+y)-x(x'+y')}{(x+y)^2} = \frac{x'y-y'x}{(x+y)^2} = \frac{axy-bxy}{(x+y)^2} \\ \rho' &= \frac{x}{x+y}\cdot\frac{y}{x+y}\cdot(a-b) = \rho\cdot(1-\rho)\cdot(a-b). \end{aligned} \tag{2.13}$$

Där man sätter in x', y' från ekvation 2.12.

I fallet med obegränsad tillväxt så existerar redan analytiska lösningar för x, y och då kan proportionen direkt tas fram som en funktion av

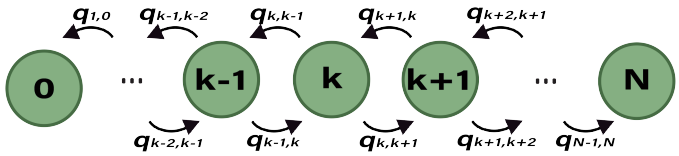
$$\rho(t) = \frac{x(t)}{x(t) + y(t)} = \frac{x_0 e^{at}}{x_0 e^{at} + y_0 e^{bt}} = \frac{1}{1 + y_0/x_0 e^{(b-a)t}}. \tag{2.14}$$

Gränsvärdet av ρ då $t \to \infty$ ger följande tre fall:

1. $a > b:\ \rho(t) \to 1$
2. $a < b:\ \rho(t) \to 0$
3. $a = b:\ \rho(t) \to \frac{x_0}{x_0 + y_0}$

2.4 Moranprocessen

Moranprocesser används för att beskriva frekvensförändringen i en konstant population med den totala storleken N. Det vill säga vi kommer undersöka hur två genotypers frekvenser i relation till varandra förändras med ökande tid. Populationen består av två olika genotyper med olika stora delningstakter [1], [18]. För att beskriva en Moranprocess används tidskontinuerliga Markovkedjor som beskrivs i Definition 2.1.



Figur 1: Moranprocessens uppbyggnad samt en visualisering av övergångsintensiteterna $q_{i,j}$ då i är starttillståndet och j är nästa tillstånd. De gröna cirklarna illustrerar ena genotypens antal, $0 \leq i, j \leq N$ där andra genotypens antal fås med $N - k$.

Moranprocessen är en av de enklaste stokastiska modellerna som används vid studier av selektion. Detta eftersom ingen mutation påverkar reproduktionen, en genotyps avkomma är av samma genotyp. På samma vis sker ingen migration från andra närliggande ekosystem. Moranprocessen utgår från att den totala populationen, N mellan två genotypers populationer förblir konstant. Detta gäller då vid varje generation väljs slumpmässigt en individ att reproducera en avkomma samtidigt väljs slumpmässigt en individ att dö ut. På så vis förblir nettoökningen för varje generation noll, i enlighet med det som illustrerats i figur 1 nämligen att för varje steg i processen finns det enbart 4 olika händelser som presenteras i tabell 1, [1].

Tabell 1: Moranprocessens fyra händelser. i är storleken på den aktuella populationen för A i samtliga fall, men populationen förändras sedan beroende på fall. Eftersom *Fall 3* och *Fall 4* resulterar i samma populationsstorlek A är de två olika händelserna lika sannolika.

	Händelse	Ny population av A	Övergångsintensitet
Fall 1	Individ ur A dör och individ ur B föds	$i - 1$	$q_{i,i-1}$
Fall 2	Individ ur B dör och individ ur A föds	$i + 1$	$q_{i,i+1}$
Fall 3	Individ ur A dör och individ ur A föds	i	$q_{i,i}$
Fall 4	Individ ur B dör och individ ur B föds	i	$q_{i,i}$

De fyra händelserna genererar tre olika utfall för genotypen A vilket beskrivs med tabell 1, A:s population kan öka eller minska med ett eller förbli den samma. Övergångsintensiteterna i tabell 1 betecknas $q_{i,j}$, där i är den aktuella populationen och j är den nya populationen. Övergångsmatrisen $Q = [Q_{i,j}]$ beskriver sannolikheten för att de olika fallen presenterade ovan kan inträffa,

det vill säga sannolikheten att röra sig från i till j. Matrisen Q är av storleken $(N+1)$x$(N+1)$. [1].

Sannolikheten för att en individ av typen A dör är i/N vilket ger att sannolikheten att en individ av typen B dör är $(N-1)/N$. Sannolikheten för reproduktion beror på delningstakterna för A och B. Vi låter B ha en delningstakt på 1 samt A låter vi ha en delningstakt som är λ. Därmed blir $\lambda i/[\lambda i + (N-i)]$ sannolikheten för reproduktion av A och $(N-i)/[\lambda i + (N-i)]$ sannolikheten för reproduktion av B [1]. Övergångsmatrisen Q för Moranprocessens fall kan därmed skrivas som

$$Q_{i,j} = \begin{cases} \frac{\lambda i}{\lambda i+(N-i)} \frac{N-i}{N}, & j = i+1, \quad 0 < i < N \\ \frac{N-i}{\lambda i+(N-i)} \frac{i}{N}, & j = i-1, \quad 0 < i < N \\ 1 - q_{i,i+1} - q_{i,i-1}, & j = i, \quad 0 < i < N. \\ 0 & \text{annars} \end{cases} \tag{2.15}$$

Processen har även två absorberande tillstånd då $X = 0$ samt $X = 1$. De absorberande tillstånden får sitt namn från att ingen förändring kan ske i dessa tillstånd då ena genotypen är helt utdöd och inte kan generera någon avkomma [1].

2.4.1 Selektions påverkan på Moranprocessen

Moranprocessen som beskrevs i avsnitt 2.4 kan också beskriva neutral evolution. Detta betyder att de två genotyperna A och B har identisk reproduktionstakt, $\lambda = 1$. Eftersom deras delningsstakter är lika stora har ingen genotyp en fördel över den andra i form av selektion. Däremot gäller det att för en begränsad population kommer en genotyp ta över och den andra kommer dö ut med en viss sannolikhet, det finns inte möjlighet för samexistens [1]. Emellertid kan en av de två genotypernas delningstakt vara större och ge en fördel till en av genotyperna. Då $\lambda > 1$ har individer av genotyp A ett övertag över population B ur ett selektionsperspektiv. Vid stora värden på λ kommer reproduktionstakten på A vara stor.

I populationer där alla individer förutom en tillhör samma genotyp är det intressant att undersöka när en av genotyperna tar över populationen och den andra dör ut. Detta beskrivs av *fixeringssannolikheten* för genotypen av den enskilda individen. Fixeringssannolikheten är sannolikheten att en ensam individ skapar avkommor som tar över hela populationen och den andra genotypen dör ut [1], alltså sannolikheten för att kedjan som beskriver Moranprocessen att fastna i tillstånd $X(t) = N$.

Låt $\gamma_i = q_{i,i-1}/q_{i,i+1}$, populationsstorleken vara N, antalet individer av genotyp A vara 1 och antalet individer av genotyp B vara $N-1$ vid start. Då beräknas fixeringssannolikheten för genotyp A för denna modell som [1]

$$\varphi_A = \frac{1}{1 + \sum_{j=1}^{N-1} \prod_{i=1}^{j} \gamma_i} \tag{2.16}$$

vilket leder till $\varphi_B = 1 - \varphi_A$. Uttrycket för fixeringssannolikheten för en population med en B individ och $N-1$ A individer vid start återfinns i appendix B. Om $\varphi_B/\varphi_A > 1$ innebär det att det mest sannolika scenariot är att genotyp B tar över hela populationen.

Vid $\lambda \neq 1$ blir $\gamma_i = 1/\lambda$ vilket leder till att fixeringssannolikheten för genotyp A ges av:

$$\varphi_A = \frac{1 - 1/\lambda}{1 - 1/\lambda^N}. \tag{2.17}$$

För stora populationer där $\varphi_A > \varphi_B$ kan approximationen $\varphi_A = 1 - 1/\lambda$ användas [1].

Då $\lambda = 1$, det vill säga att båda genotyperna har samma delningstakt, har alla genotyperna lika stor chans att väljas till att födas eller dö. Detta medför att fixeringssannolikheten för vardera individ blir den samma och fixeringssannolikheten blir då $\varphi_A = 1/N$.

2.4.2 Analytisk lösning för Moranprocessen

Med Kolmogorovs framåtekvationer kan väntevärdet för Moranprocessen tas fram. Den generella lösningen är som beskrivet i ekvation 2.8

$$\frac{d}{dt}P = QP \Rightarrow P(t) = P_0 e^{Qt}. \tag{2.18}$$

Eftersom Moranprocessen beskrivs av en ändlig övergångsmatris Q så kan vi enkelt ta fram en numerisk lösning till Kolmogorovs framåtekvationer, och därmed få fram sannolikheten att processen befinner sig i ett givet tillstånd vid en given tid.

Som exempel kan vi skapa en modell med total population $N = 5$ och $\lambda = 1$. Vi börjar processen med en individ så $P_0 = [0, 1, 0, 0, 0, 0]$. Vi beräknar numeriskt fram sannolikheten att processen befinner sig i olika tillstånd vid olika tider nedan:

$$\begin{aligned}
t = 0 \quad & P(t) = [0,\ 1,\ 0,\ 0,\ 0,\ 0] \\
t = 1 \quad & P(t) = [0.44,\ 0.29,\ 0.16,\ 0.08,\ 0.03,\ 0.01] \\
t = 2 \quad & P(t) = [0.59,\ 0.13,\ 0.10,\ 0.075,\ 0.05,\ 0.05] \\
t = 5 \quad & P(t) = [0.75,\ 0.03,\ 0.03,\ 0.03,\ 0.03,\ 0.15] \\
t = 10 \quad & P(t) = [0.79,\ 0.00,\ 0.00,\ 0.00,\ 0.00,\ 0.19] \\
t = 100 \quad & P(t) = [0.8,\ 0,\ 0,\ 0,\ 0,\ 0.2].
\end{aligned} \tag{2.19}$$

Som förväntat så ser vi att sista elementet i $P(t)$, alltså fixeringssannolikheten, går mot $\varphi_A = 1/N = 1/5$ medan resten av sannolikheten ligger i första elementet, alltså sannolikheten att individen dör ut.

Vi studerar nu kort vad som händer med Moranprocessen när populationsstorleken går mot oändligheten. Låt $x \in [0, 1]$ vara andelen av populationen med förändrat genom och N vara den totala populationsstorleken. Ifall vi låter N gå mot oändligheten så kommer fixeringssannolikheten för genomet med bättre tillväxthastighet gå mot 1 [19]. Vi kan förstå det som att de stokastiska efekterna av processen blir mindre ju större den totala populationen är. Sammantaget borde därför alla realiseringar av Moranprocessen konvergera mot lösningen till den logistiska ekvationen med mindre och mindre stokastiskt beteende då $N \to \infty$.

2.5 Den kombinerade modellen

En mer realistisk modell än de som är beskrivna i avsnitt 2.3 och 2.4 är att den totala populationsstorleken kan variera över tid och är beroende av begränsningar som ett verkligt system kan uppleva. Det är en slags kombinerad modell mellan de stokastiska modellerna för konstant och obegränsad population. Vi kommer motivera den kombinerade modellen genom att betrakta en etablerad modell från populationsdynamik.

Låt $x(t)$ vara populationsstorleken för genotyp A, $y(t)$ vara populationsstorleken för genotyp B och K vara den begränsande kapaciteten, eller *bärkapaciteten* för systemet. Låt λ_x vara genotyp A:s delningstakt och λ_y vara genotyp B:s delningstakt som är 1. Låt även x_0 och y_0 vara startpopulationen för genotyp A respektive B. Till sist låt μ_x och μ_y vara genotyp A:s respektive genotyp B:s dödstakt. Då är den deterministiska modellen för en population av två genotyper den logistiska ekvationen

$$\begin{aligned}
x'(t) &= \lambda_x x \left(1 - \frac{x+y}{K}\right) - \mu_x x \\
y'(t) &= \lambda_y y \left(1 - \frac{x+y}{K}\right) - \mu_y y.
\end{aligned} \tag{2.20}$$

Den stokastiska versionen av den kombinerade modellen kommer vara en tvådimensionell Markovkedja. De möjliga övergångarna för kedjan är att antalet A eller B individer ökar med 1 eller antalet

A eller B individer minskar med 1. Om vi låter antalet A individer vara i och antalet B individer vara j, det vill säga $x(t) = i$ och $y(t) = j$, så beskriver tabell 2 kedjans möjliga övergångar samt dess intensitet.

Tabell 2: Möjliga övergångar mellan tillstånd för den kombinerade modellen och dess intensitet

Övergångar $(i, j) \to (i', j')$	Intensitet q
$(i, j) \to (i + 1, j)$	$\lambda_x i \left(1 - \frac{i+j}{K}\right)$
$(i, j) \to (i, j + 1)$	$\lambda_y j \left(1 - \frac{i+j}{K}\right)$
$(i, j) \to (i - 1, j)$	$\mu_x i$
$(i, j) \to (i, j - 1)$	$\mu_y j$

Från tabell 2 kan vi också utläsa att intensiteten för delning av individer är avtagande då populationen växer mot bärkapaciteten, och på samma sätt är intensiteten för individ död linjärt ökande. Notera dock att dödstakten är oberoende av den totala populationsstorleken medan det finns ett beroende mellan populationsstorlek och delningstakten, nämligen $N = i + j$ [20],[21].

Då den kombinerade modellen liknar Moranprocessen i att det finns en övre begränsning för populationsstorleken påverkas systemet av selektion likande beskrivningen i avsnitt 2.4.1. Skillnaden på hur selektionen påverkar de två modellerna ligger i hur fixeringssannolikheten uttrycks. Att matematiskt formulera dessa ligger utanför projektets tidsram. I detta projekt görs istället en liten analys och jämförelse mellan de simulerade fixeringssannolikheterna för Moranprocessen och den kombinerade modellen.

3 Metod

I detta avsnitt beskrivs det hur teorin från avsnitt 2 implementeras på de modeller som beskrivs i avsnitt 1.2. Alla modeller utnyttjar Gillespies algoritm för simulering av tillväxt. Koden för Gillespies algoritm för varje modell återfinns som kodfilerna 1, 2, 3 i appendix D.

3.1 Gillespies algoritm

Algoritmen vi använder för att simulera en CTMC har många namn men vi har valt att använda namnet Gillespie algoritmen efter den amerikanska fysikern Daniel Thomas Gillespie [2]. Algoritmen använder en övergångsmatris och en startdistributionen för att simulera en CTMC.

Definition 3.1 *Vi låter X vara en CTMC för $n \geq 0$ med övergångsmatris Q och startdistribution α så att $\mathbb{P}(X(0) = k) = \alpha_k$. Vi låter också E_n för $n \geq 0$ vara en sekvens av oberoende slumpvariabler från en exponentialfördelning med parameter 1 (notera att $\frac{E_n}{\lambda}$ är exponentiellt fördelad med parameter λ). Då konstrueras en realisering av CTMCn på följande vis:*

1. *Välj $X(0)$ i enlighet med den initiala distributionen α.*
2. *Sätt $T_0 = 0$ och låt $W_0 = \frac{E_0}{\lambda_{X(0)}}$ vara tiden som kedjan befinner sig i tillståndet $X(0)$. W_0 är alltså exponentiellt fördelad med parameter $\lambda_{X(0)}$. För tiden $t \in [T_0, T_0 + W_0)$ sätter vi då $X(t) = X(0)$.*
3. *För $n \geq 1$, låt $T_n = T_{n-1} + W_{n-1}$ och välj nästa tillstånd $X(T_n)$ likformigt proportionellt med sannolikhet $\frac{\lambda_{X(T_n),X(T_{n-1})}}{\lambda_{X(T_{n-1})}}$. Sätt nu $W_n = \frac{E_n}{\lambda_{X(T_n)}}$ och för tiden mellan övergångarna, $t \in [T_n, T_n + W_n)$ sätter vi $X(t) = X(T_n)$.*
4. *Upprepa stegen ovan i processen tills en sluttid T uppnås.*

Det behövs alltså två slumpvariabler för varje steg i algoritmen, en som bestämmer hur länge processen befinner sig i ett visst tillstånd och en som bestämmer nästa tillstånd i processen [2], [22].

För en enklare beskrivning av algoritmen kan man istället förklara det på följande vis: vi väljer först ett tillstånd att börja i, sedan tar vi fram tiden tills processen går vidare till nästa tillstånd, sedan väljer vi slumpmässigt nästa tillstånd och till sist tar vi fram tiden till nästa tillstånd, och så vidare. Skillnaden mellan olika modeller kommer ligga i vilka tillstånd som processen kan vara i, hur tiderna är fördelade för varje tillstånd och sannolikheten att röra sig mellan olika tillstånd.

3.2 Implementation av modellerna

Då modellerna som ska analyseras alla bygger på CTMCs är Gillespies algoritm grunden till implementationen av alla modellerna i projektet. Vi började med att simulera exponentiell tillväxt genom att bygga CTMCn där vårt $X(t)$ är populationsstorleken vid tiden t. Då vi i detta fall har obegränsad population finns det bara en möjlighet för nästa tillstånd eftersom vi inte har någon dödstakt att ta hänsyn till. Tiden som denna process befinner sig i varje tillstånd slumpas fram med en exponentiell fördelning vars parameter är produkten av tillståndets aktuella populationsstorlek och delningstakten.

För Moranprocessen gäller det, som beskrivet i tabell 1, att för varje tillstånd finns det fyra möjligheter för nästa tillstånd. Tiden som processen befinner sig i ett tillstånd slumpas därför fram med en exponentiell fördelning vars parameter är summan av övergångssannolikheterna beskrivna i avsnitt 2.4.

Den kombinerade modellen är i grunden som den obegränsade modellen men här har vi även en dödstakt och bärkapacitet att ta hänsyn till. Detta leder till att det för varje tillstånd nu istället finns fler möjligheter för nästa tillstånd. Processens tidssteg slumpas därför fram med en exponentiell fördelning vars parameter är produkten av populationsstorleken och summan av delnings- och dödstakten.

3.3 Felberäkning

För att empiriskt avgöra ifall simuleringarna följer teorin använder vi två olika metoder. För den obegränsade modellen är det intressant att se hur nära medelvärdet av de olika realiseringarna är den analytiska lösningen och hur detta värde förändras med olika värden på parametrarna i modellen. För den begränsade och kombinerade modellen är det istället intressant att se i hur många av simuleringarna genotypen som startar med en individ tar över populationen jämfört med den analytiska fixeringssannolikheten med olika parameterval.

Som visades i härledning 2.5 så är väntevärdet av den obegränsade stokastiska processen lika med den analytiska lösningen till den deterministiska modellen. Vi kommer därför att analysera hur de olika parametrarna i den obegränsade modellen påverkar skillnaden mellan medelvärdet av ett antal simuleringar och den analytiska lösningen. Speciellt så studeras hur en populations tillväxthastighet, simuleringarnas sluttid och antalet simuleringar påverkar felet. Storleken på felet kommer beräknas på två olika sätt. Låt $\{X_n\},\ n = 1, 2, ..., S$ vara en serie av realiseringar av Markovkedjan, och medelvärdet av realiseringarna vid tiden t vara $N(t) = \frac{1}{S}\sum_{n=1}^{S} X_n(t)$, och x den analytiska lösningen $x(t) = x_0 e^{\lambda t}$. Då kan värdet på simuleringsfelet beräknas på följande vis:

$$\begin{aligned} E &= \frac{1}{T}\int_0^T (N(t) - x(t))^2 dt \\ E' &= \frac{1}{T}\int_0^T \frac{(N(t) - x(t))^2}{N(t)} dt, \end{aligned} \tag{3.1}$$

där T är sluttiden för alla simuleringar. Vi ser att skillnaden mellan E och E' är att vi normaliserar med populationsstorleken. Ifall vi ökar antalet simuleringar S så kommer enligt stora talens lag $N(t)$ bli en bättre och bättre approximation av $X(t)$ så felen borde minska. Dock om vi ökar tillväxthastigheten eller sluttiden så kommer variansen av X att öka enligt B.5, så E borde växa exponentiellt medan E' borde förbli konstant.

4 Resultat

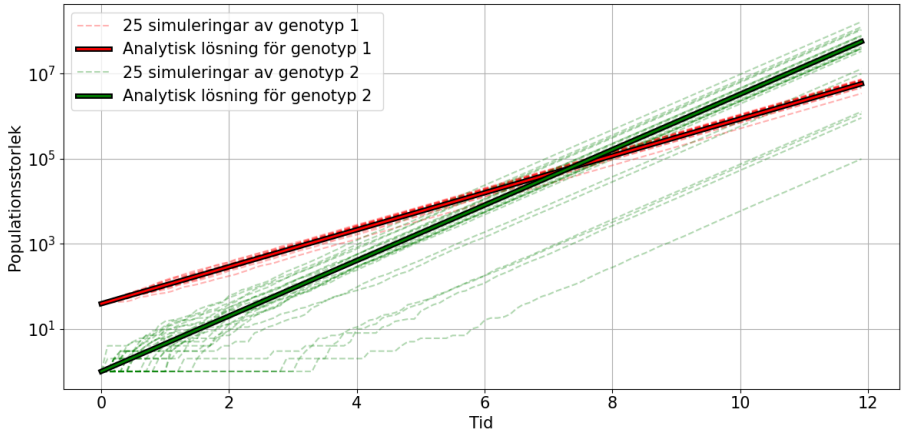
I bilaga D finns den källkod som användes för att simulera resultatens realiseringar av de stokastiska modellerna och de deterministiska modellerna. Nedan följer det resultatet från dessa simuleringar vilket bedöms som mest relevant för arbetets syfte. Fler finns i bilaga C. För alla realiseringar som jämför två olika genotyper sattes initialpopulationerna till $N_1 = 39$, $N_2 = 1$ och tillväxthastigheterna till $\lambda_1 = 1$, $\lambda_2 = 1.5$. Sluttiden varierade mellan de olika modellerna eftersom de har olika tidsskalor, men alla simuleringar kördes tills någon av genotyperna fixerades i populationen. Antalet simuleringar varierade också beroende på hur mycket detaljrikedom som ansågs nödvändigt i varje individuell graf.

4.1 Obegränsad tillväxt

Nedan följer ett urval av grafer som visar 25 simuleringar av den obegränsade modellen med två populationer som startade med $N_1 = 39$, respektive $N_2 = 1$ individer, tillväxthastighet $\lambda_1 = 1$, $\lambda_2 = 1.5$ och sluttid $T = 12$.

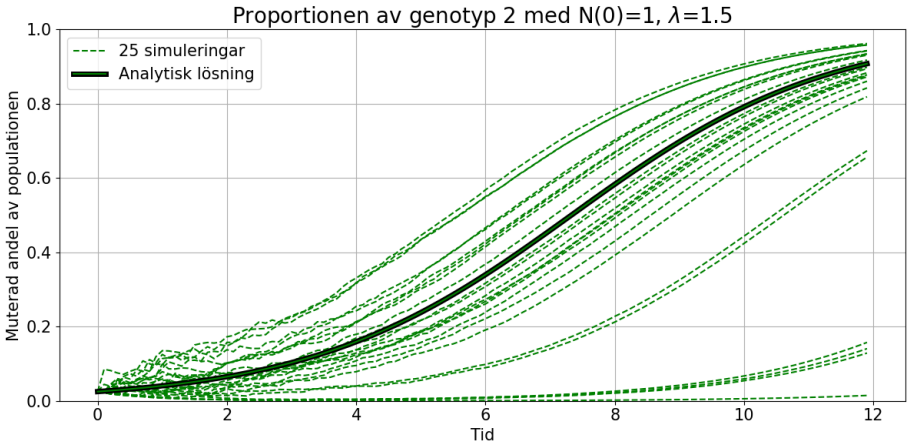
4.1.1 Simuleringar för den obegränsade modellen

Först simulerades två populationer med den obegränsade modellen. Då konstruerades figurer med en logaritmisk axel för att kontrollera så att populationerna växte exponentiellt.



Figur 2: Simulering av obegränsad tillväxt där genotyp 1 har delningstakt 1 och startpopulation 39 medan genotyp 2 har delningstakt 1.5 och startpopulation 1. De röda streckade linjerna är realiseringarna av genotyp 1, de gröna genotyp 2. Genotyp 1 och genotyp 2:s analytiska lösningar, tagna från härledning (2.5, är den röda respektive gröna heldragna linjen.

Populationerna växte linjärt i den logaritmiska skalan i figur 2 vilket betyder att populationerna som väntat växte exponentiellt. Sedan skapades figur 3 som visar andelen av den muterade genotypen.



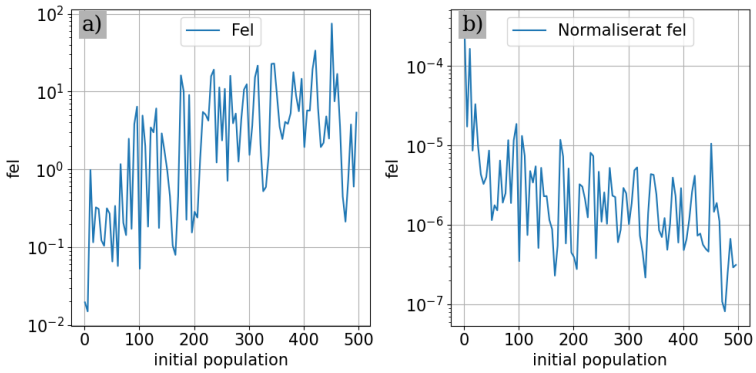
Figur 3: Proportionen av populationen med den muterade genotypen. Simuleringarna är samma som i figur 2 ovan. I figuren åskådliggörs även den analytiska lösningens frekvens, tagna från formel (2.14), som en heldragen linje.

Figur 3 illustrerar hur frekvensen, ρ mellan genotyp 2 och den totala populationen ändras över tid, tillsammans med lösningen till den deterministiska modellen. De flesta realiseringar resulterar i liknande beteende som lösningen till den deterministiska modellen, med undantag för sex realiseringar som tog betydligt längre tid att börja växa.

4.1.2 Felberäkning för den obegränsade modellen

Som beskrivet i avsnitt 3.3 så mäts felet hos den stokastiska modellen som skillnaden mellan medelvärdet av de olika realiseringarna och den analytiska lösningen 2.5. Detta kan göras på två olika sätt genom att antingen normalisera felet efter populationsstorleken, eller inte. I figurerna 4 och 5 visas hur ändring av startpopulationen och antalet simuleringar förändrar dessa värden. När felet beräknades kördes 1000 simuleringar av en population med startstorlek på 100 individer och delningstakt $\lambda = 0.5$ till sluttiden $T = 5$. Sedan testades olika värden på en parameter i taget och de andra behölls fixerade vid startvärdena. Nedan följer de mest relevanta graferna, fler återfinns i bilaga C.1.

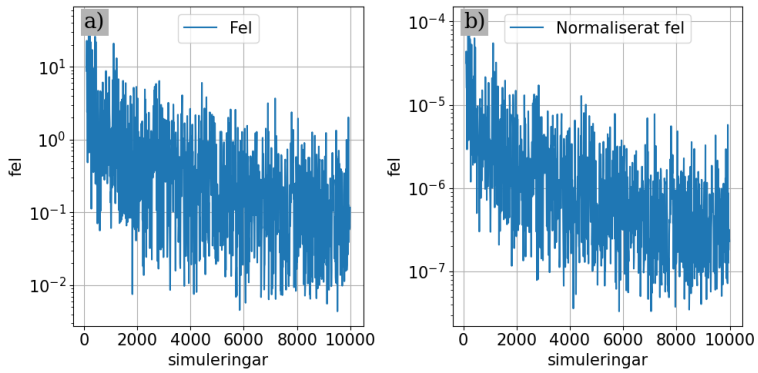
Först jämförs olika värden på startpopulationen.



Figur 4: Skillnaden mellan medelvärdet av flera realiseringar och den analytiska lösningen 2.5. Parametrarna är $\lambda = 0.5$, sluttid $T = 5$ och startpopulationen varierar längs med x-axeln. För varje värde på startpopulationen togs medelvärdet fram från 1000 körningar. Notera den logaritmiska skalan på y-axeln. a) visar det absoluta felet och b) visar det normaliserade felet.

Det absoluta felet i figur 4 ökar med större initialpopulation, medan det normaliserade minskar.

Nedan följer en jämförelse mellan olika antal simuleringar.

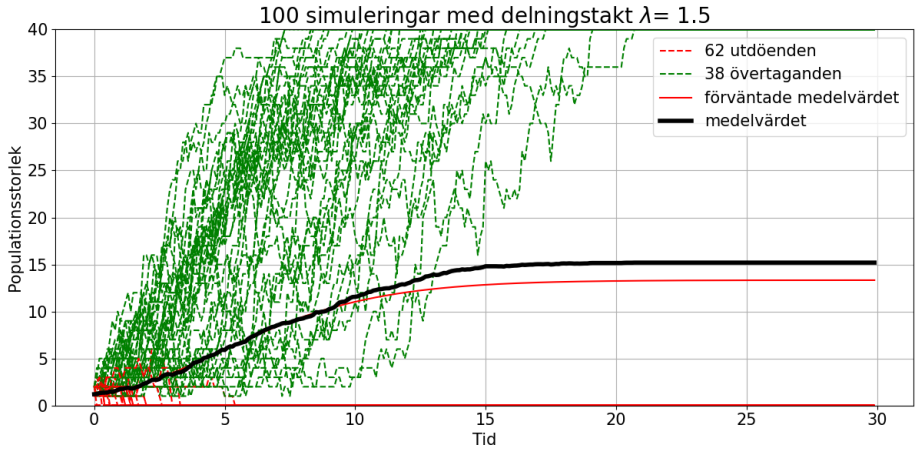


Figur 5: Skillnaden mellan medelvärdet av flera realiseringar och den analytiska lösningen 2.5. Parametrarna är $\lambda = 0.5$, sluttid $T = 5$ och startpopulation 100. Antalet körningar varierar längs med x-axeln. Notera den logaritmiska skalan på y-axeln. a) visar det absoluta felet och b) visar det normaliserade felet.

I figur 5 syns att både det absoluta felet och det normaliserade felet minskar med antalet simuleringar.

4.2 Moranmodellen

Vi valde att köra 100 simuleringar av Moranmodellen för att visualisera processen. Vi valde som tidigare att ha 39 individer av genotyp 1 med delningstakt 1 och 1 individ av genotyp 2 med delningstakt 1.5. Nedan följer simuleringarna tillsammans med sitt stickprovsmedelvärde. Vi visar även det analytiska medelvärdet av processen, framtaget numeriskt med hjälp av Kolmogorovs framåtekvationer 2.18.



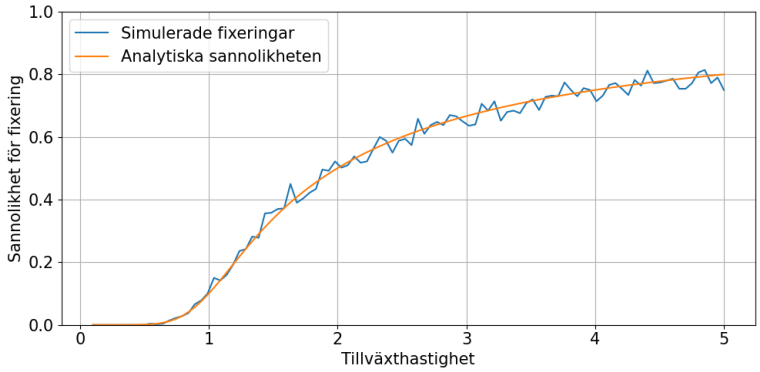
Figur 6: Resultatet från 100 simuleringar av Moranprocessen med parametrar $N_1 = 39$, $N_2 = 1$, $\lambda_1 = 1$ och $\lambda_2 = 1.5$. De streckade linjerna är genotyp 2:s realiseringar. De är färgade gröna ifall genotyp 2 fixeras i populationen och röda ifall den dör ut. Den svarta helfärgade linjen motsvarar medelvärdet av alla 100 simuleringar och den röda linjen är processens väntevärde beräknat numeriskt med hjälp av Kolmogorovs framåtekvationer 2.18.

Medelvärdet av simuleringarna i figur 6 följer väntevärdet relativt väl i början men konvergerar inte mot det då tiden ökar. Detta är på grund av att det förväntade medelvärdet går mot $N \cdot \varphi_A$ medan det realiserade medelvärdet går mot antalet fixerade körningar delat på det totala antalet

körningar. Eftersom det finns en viss variation i hur många realiseringar som fixeras så kommer detta inte nödvändigtvis vara lika med det analytiska medelvärdet. På samma sätt så skiljer sig alla individuella realiseringar mycket åt från medelvärdeslösningen eftersom de alla efter en viss tid når 0 eller 40.

4.2.1 Jämförelse av fixeringssannolikhet för Moranmodellen

För att se hur väl den analytiska fixeringssannolikheten stämmer överens med simulerad data så körde vi 500 realiseringar av Moranprocessen med populationsstorlek $N_1 = 9$, $N_2 = 1$, tillväxthastighet $\lambda_1 = 1$, $\lambda_2 = 1.5$ och sluttid $T = 50$. Vi beräknade sedan andelen fixeringar för olika värden på dessa parametrar och plottade de tillsammans med den analytiska fixeringssanolikheten. Nedan följer den mest relevanta grafen, där tillväxthastigheten varieras. Fler finns i appendix C.2.

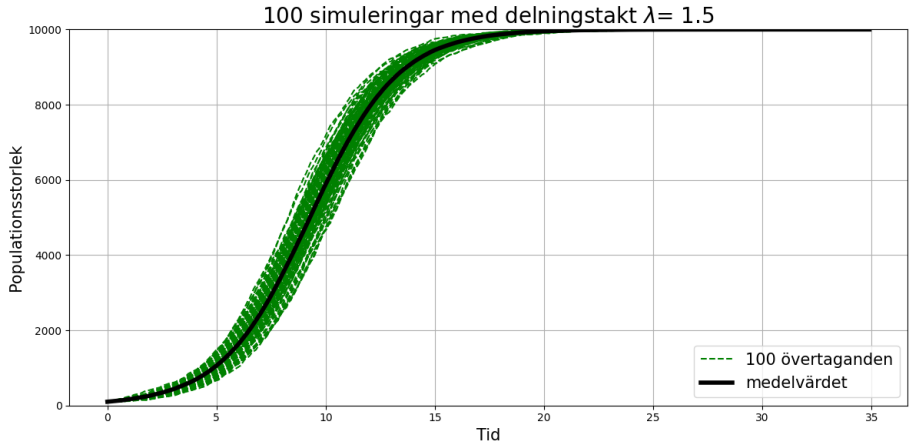


Figur 7: Den förväntade fixeringssannolikheten för olika värden på genotyp 2s tillväxthastighet med parametrar $N_1 = 9$, $N_2 = 1$, $\lambda_1 = 1$. Den blåa linjen är medelvärdet av 500 simuleringar för varje värde på λ_2 medan den orange linjen är den analytiska lösningen för fixeringssannolikhet beräknad med hjälp av ekvation 2.17

Den stokastiska fixeringssannolikheten i figur 7 följer den analytiska väl. Detsamma gäller för graferna i appendix C.2.

4.2.2 Stora populationer för Moranmodellen

Som nämnt i avsnitt 2.4.2 så väntar vi oss att Moranprocessen ska bli allt mer deterministisk ju större population som används. Nedan följer en graf med en populationsstorlek på 10 000 där 1% av populationen bär den muterade genotypen med tillväxthastighet 1.5. Två till grafer med olika populationsstorlek finns i appendix C.2. På grund av beräkningsmässiga gränser kunde vi inte ta fram det analytiska medelvärdet av simuleringarna, men stickprovsmedelvärdet visas.

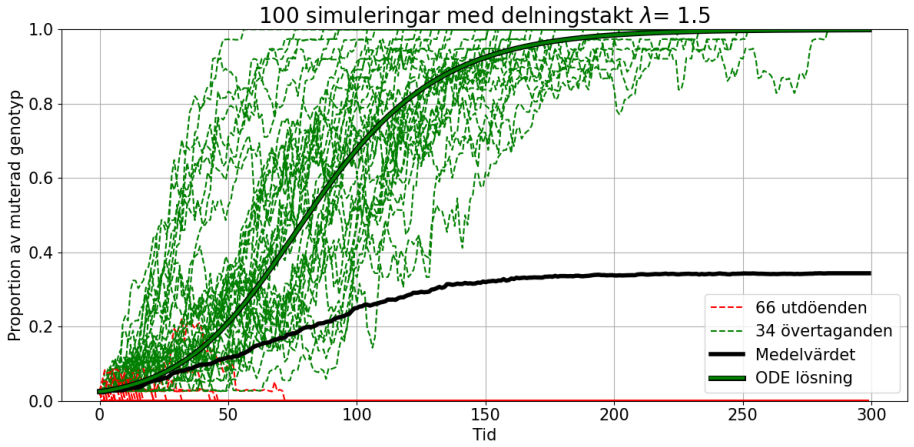


Figur 8: 100 simuleringar av Moranmodellen med en population på 10 000 individer varav 1% (=100 individer) av den muterade genotypen med tillväxthastighet 1.5, resten har tillväxthastighet 1.

Som väntat så följer alla simuleringar medelvärdet mycket närmare i figur 8 än tidigare simuleringar. Alla simuleringar resulterar i att den bättre anpassade genotypen fixeras. Figur 15 i appendix C.2 visar att om vi höjer den totala populationen till 1 000 000 så kommer alla simuleringar utvecklas nästan helt identiskt.

4.3 Den kombinerade modellen

Som tidigare så simulerade vi 100 populationer med 39 individer av genotyp 1 med tillväxthastighet 1 och 1 individ av genotyp 2 med tillväxthastighet 1.5. För att matcha denna populationsstorlek så satte vi även systemets bärandekapacitet till 40. Vi beräknade dessutom numeriskt lösningen till ODEn 2.20 och stickprovsmedelvärdet av körningarna. Sedan jämförde vi en stor mängd simuleringars fixeringsandel med den analytiska fixeringssannolikheten.



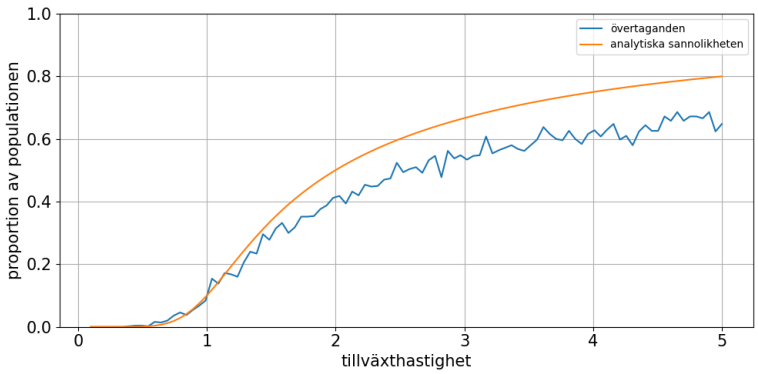
Figur 9: Resultatet av 100 simuleringar av den kombinerade modellen med en bärkapacitet på 40 och startpopulation $N_1 = 39$, $N_2 = 1$. De streckade linjerna är andelen av genotyp 2 med delningstakt 1.5 i populationen från de olika realiseringarna. De är färgade gröna ifall genotyp 2 fixeras i populationen och röda ifall den dör ut. Den svarta helfärgade linjen motsvarar medelvärdet av alla 100 simuleringar och den gröna helfärgade linjen är lösningen till differentialekvationssystemet 2.20.

Vi ser att simuleringarna i figur 9 där genotyp 2 fixeras följer ODE-lösningen relativt väl, medan

genotypen dör ut i många andra simuleringar. Återigen så är stickprovsmedelvärdet inte liknande någon av simuleringarna.

4.3.1 Jämförelse av fixeringssannolikhet för den kombinerade modellen

För att kunna jämföra den kombinerade modellen med Moranmodellen så jämför vi här hur fixeringssannolikheterna skiljer sig åt. Nedan följer den mest relevanta grafen, fler återfinns i appendix C.3. Vi jämför alltså här data från körningar av den kombinerade modellen med den analytiska fixeringssannolikheten för Moranprocessen med motsvarande parametrar som för den kombinerade modellen, alltså samma startpopulation och delningstakter.



Figur 10: Fixeringssannolikheten för den kombinerade modellen med bärkapacitet 10, $N_1 = 9$, $N_2 = 1$, $\lambda_1 = 1$. Tillväxthastigheten för genotyp 2 varierar längs med x-axeln. Den blåa linjen är den simulerade fixeringsandelen för genotyp 2 och den orange linjen är den analytiska sannolikheten som fås från 2.17. Den blåa linjen kommer från 500 realiseringar för varje värde av λ_2.

Den simulerade fixeringsandelen i figur 10 följer till stora drag den analytiska, men är generellt sett lite mindre. Även graferna i appendix C.3 följer detta mönster.

5 Diskussion

Projektet har undersökt tre olika modeller för att beskriva evolutionen i en population som består av två genotyper. I dessa modeller har startpopulationen, delningstakten och totalpopulationen varieras. Målet har också varit att avgöra modellernas långsiktiga/asymptotiska beteende och undersöka hur den genetiska driften påverkar. Väsentligt har varit jämförelsen mellan medelvärdet av ett antal realiseringar och realiseringarna i sig själva. Fixeringssannolikheten har även varit av speciellt intresse att bestämma för Moranprocessen och den kombinerade modellen för att skapa en tydligare jämförelse. I avsnitt 5.1 och avsnitt 5.2 kommer resultatet jämföras mot den teori som byggdes upp i avsnitt 2. Till sist kommer metodvalen och projektets avgränsningar diskuteras.

5.1 Resultat

Till att börja med kan den obegränsade modellen diskuteras. Enligt figur 11 så visar grafen tydligt hur både genotyp 1 och genotyp 2:s realiseringar växer exponentiellt alternativt linjärt i en semi-log graf som figur 11 visar. Därutöver går det även att urskilja hur de flesta realiseringar håller sig nära de deterministiska heldragna linjerna i figur 11. Detta stöds ytterligare av figurerna 12, 13, 4 och 5. Det vill säga figur 5 och 4 understryker hur felet mellan medelvärdet av de stokastiska realiseringar och den deterministiska modellen minskar. Däremot om sluttiden eller delningstakten varieras så går det att grafiskt att urskilja i figurerna 13, 12 att felet ökar för ökande magnitud på parametrarna. Följaktligen uppstår detta för delningstakten och sluttiden på grund av att variansen i de stokastiska realiseringarna ökar med ökande λ i enlighet med härledning B.5. Dock om normalisering utnyttjas enligt avsnitt 3.3 så håller sig felet inom ett begränsat och icke växande för både figur 13 och 12. Till sist stödjer figur 3 den obegränsade modellens asymptotiska beteende som presenteras i avsnitt 2.3.1, det vill säga då tiden går mot oändligheten går frekvensen av genotyp 2 mot 1 ty genotyp 2:s delningstakt är 1.5 medan genotyp 1:s delningstakt är 1.

För den begränsade modellen eller Moranmodellen som modellen även kallas ser vi från figur 16, 7 och 17 hur stokastiska simuleringar ligger inom ett begränsat intervall runt den analytiska lösningen oavsett hur vi ändrar en parameter. Därutöver bekräftar avsnitt 4.2.1 de värden som fås av ekvation 2.17 då populationen är stor. Vi kan även se från figur 8 och 15 hur detta beteende också framträder i simuleringar. Dock kan det vara till viss del vilseledande att enbart kolla på det stokastiska medelvärdet, ty viktig information kan gå förlorad. Detta går grafiskt att urskilja i figur 14 i bilaga C. Med andra ord så representerar den svarta linjen medelvärdet mellan två sorts ´extrema´, antingen tar den ena genotypen över eller dör ut. Detta är i kontrast mot den obegränsade modellen där populationer växer obegränsat men olika snabbt och där de flesta realiseringar ligger inom ett begränsat intervall från den analytiska lösningen. Visserligen ligger den förväntade lösningen/det analytiska värdet och medelvärdet relativt nära varandra i figur 6 i enlighet med 2.4.2 men dessa linjer beskriver bevisligen inte hela Moranmodellens dynamik.

Utifrån figurerna 9 och 6 går det att grafiskt urskilja hur den kombinerade modellen har fler realiseringar som dör ut i jämförelse med Moranmodellen även då de har samma parametrar bortsett från bärkapaciteten, 66 mot 62. Detta kan förklaras med att bärkapaciteten saktar ner övertaganden, vilket även medelvärdet för den kombinerade modellen kan bekräfta, ty den ökar mer långsamt än för Moranmodellen. Därutöver bekräftar även figur 19, 10 och 18 den tidigare tanken, det vill säga bärkapaciteten ger upphov till en lägre fixeringssannolikhet jämfört med Moranprocessen. Bortsett från denna observation är Moranmodellens och den kombinerade modellens tillväxt relativt lika i vårt fall. Dock är den kombinerade modellens betydligt mer komplex. Möjligtvis har den kombinerade modellen ett annat beteende om inte totalpopulationen är "fylld från start", det vill säga att inte totalpopulationen av de två genotyperna är lika med bärkapaciteten. Exempelvis hade det varit intressant att undersöka hur den kombinerade modellen modellerar introducerandet av en ny art i en miljö med bärkapacitet.

5.2 Metodval och avgränsningar

Till att börja med har detta projekt handlat om modeller med fundamentalt olika tillämpningsområden. Den obegränsade modellen används oftast vid undersökning av bakteriers populationsdynamik i en petriskål där resurserna är i princip obegränsade och ingen konkurrens om resurserna mellan populationer sker [9]. Medan Moranmodellen och den kombinerade modellen studerar populationer som är mer begränsade av naturen och av varandra [11] ,[10].

Flertalet avgränsningar gjordes i detta arbete, då projektet grundar sig i studier av enkla evolutionära modeller. Dock är frågan relevant huruvida dessa avgränsningar påverkar de slutsatser som kan dras. Till att börja med så begränsades projektet till att enbart behandla två genotypers samverkan med varandra. Dock påverkar inte avgränsningen resultaten för den obegränsade modellen då samverkan mellan genotyper är i princip obefintlig och liknade resultat hade fåtts med flera genotyper. Dock så hade fler genotyper gett ett mer komplext samspel för Moranmodellen och den kombinerade modellen d.v.s. vårt resultat går inte att generalisera till flera populationer i kontrast mot den obegränsade modellen. Liknade analogi kan föras för den avgränsning som görs kring konstant miljö. Detta eftersom i den obegränsade modellen blir inte individerna påverkade av bärkapaciteten då de befinner sig långt ifrån den. Samtidigt som Moranmodellen och den kombinerade modellen har ett begränsat antal individer under hela studiet och därmed blir påverkade av miljön.

En idé för att göra projektet mer enhetligt hade varit att låta den obegränsade modellen också ha en dödstakt μ. Dock om populationen börjar med en individ som de andra tillväxtmodellerna gör så riskerar populationen att dö ut direkt på grund av dödstakten. Detta är i sig inte speciellt realistiskt då vid odling av bakterier så 'smetats' flera bakterier ut på en petriskål och inte bara en. Dock är det intressant att kolla på just en startpopulation på en individ då detta simulerar mutationer som sker evolutionärt.

Vi kan även se att i projektet har valet av tidskontinuerliga Markovkedjor varit ett bra val. Tidsdiskreta Markovkedjor hade missat dynamiken kring hur snabbt den obegränsade modellen växer. Detta då övergångsintensiteten växer proportionellt mot den aktuella populationen, det vill säga i genomsnitt växer populationen snabbare vid större antal. Dock för Moranmodellen och den kombinerade modellen är inte skillnaden i hastighet mellan olika tidssteg lika stor som för den obegränsade modellen tider längre fram i processen.

6 Slutsats

Slutsatsen som kan dras för detta projekt är att för den obegränsade modellen ger både den stokastiska och den deterministiska modellen relativt lika resultat. Men den deterministiska modellen är enklare och därav smidigare att använda. För Moranmodellen gäller det att den deterministiska modellen skiljer sig betydande från de individuella stokastiska realiseringarna. Den deterministiska modellen bidrar dock med att ge fixeringssannolikheten vilket beskriver en viktig del av modellens dynamik. Därför är det att föredra de stokastiska realiseringarna i simuleringssammanhang. För den kombinerade modellen gäller liknande resonemang som för Moranmodellen. Däremot är den kombinerade modellen mer realistiskt då populationen tillåts konkurrera mot inte bara en population utan även sig själv på grund av bärkapaciteten. Totalpopulationen tillåts även fluktuera, vilket skapar en mer realistisk dynamik. Dock är den kombinerade modellen betydligt mer komplex än Moranmodellen och i vårt fall visar den kombinerade modellen en dynamik som liknar Moranmodellen. Därför är det att föredra den enklare modellen som är Moranmodellen.

Vi har jämfört den obegränsade modellen, Moranmodellen samt den kombinerade modellen var för sig genom att variera dess parametrar samt jämfört modellerna med varandra för att hitta likheter och skillnader mellan modellerna. Då modellerna är kraftigt förenklade med betydande avgränsningar kan de enbart användas för att generellt beskriva populationer med liknande förutsättningar. Därför hade vidare studier varit intressant med mera komplexa modeller. Exempelvis hade delningstakten kunnat vara frekvensberoende då detta hade kunnat simulera en sorts flockbeteende eftersom det i naturen är lönsamt att samarbeta både med sin egen populationstyp och med andra populationstyper, [23]. Det hade även varit intressant att studera diploida individer istället för haploida individer eftersom de flesta däggdjur är diploida individer. Det skulle emellertid innebära en betydligt mer komplex modell då nedärvningsprocessen är helt annorlunda.

Studier av hur populationer beter sig är extremt viktigt i nutiden. Betydande exempel i nutid kan vara exempelvis smittspridningen av Covid-19 [24]. Beteendemönstret hos smittspridarna analyseras genom att skapa modeller baserat på bland annat deras sociala interaktioner. Modeller vars syfte är att kunna skapa en prognos för framtida smittspridning av virus och därmed förhindra det. Metodiken gäller inte bara virus utan kan även användas till andra områden såsom studier av hotade djurarter och bevarandebiologi. Genom att observera hotade djurarter kan deras behov och levnadsmönster analyseras för att utveckla modeller samt strategier för att bevara deras existens. Ett tredje exempel som också kräver populationsforskning, vilket är den mänskliga populationen för att förutspå påverka miljön. Detta genom att skapa modeller på människans energiförbrukning, avfallshantering och konsumtionsvanor. Analysen av dessa modeller görs precis som i de andra exemplen att förutspå framtiden och kunna påverka den och i detta fall minska den negativa påverkan människan har på planeten. Därmed är det viktigt att fortsätta studera evolution så man kan hitta modeller som beskriver den verklighet vi lever i.

Referenser

[1] Nowak MA. Evolutionary Dynamics: Exploring the Equations of Life. Harvard University Press; 2006.

[2] Anderson DF. Lecture Notes on Stochastic Processes with Applications in Biology; 2017. Mars.

[3] Govaert L, Fronhofer EA, Lion S, Eizaguirre C, Bonte D, Egas M, et al. Eco-evolutionary feedbacks—Theoretical models and perspectives. Functional Ecology. 2019;33(1):13-30.

[4] Chitty D. In: The Natural Selection of Self-Regulatory Behavior in Animal Populations; 2017. p. 136-70.

[5] Hiltunen T, Hairston N, Hooker G, Jones L, Ellner S. A newly discovered role of evolution in previously published consumer-resource dynamics. Ecology Letters. 2014;17(8):915-23.

[6] Anderson RM, May RM. Coevolution of hosts and parasites. Parasitology. 1982;85(2):411–426.

[7] Dykhuizen D. Species Numbers in Bacteria. Proceedings California Academy of Sciences. 2005 06;56:62-71.

[8] Otto SP, Whitlock MC. The Probability of Fixation in Populations of Changing Size. Genetics. 1997 06;146(2):723-33. Available from: https://doi.org/10.1093/genetics/146.2.723.

[9] PIRT SJ. A Kinetic Study of the Mode of Growth of Surface Colonies of Bacteria and Fungi. Journal of General Microbiology. 1967 05;47(2):181–197.

[10] Law R, Murrell DJ, Dieckmann U. POPULATION GROWTH IN SPACE AND TIME: SPATIAL LOGISTIC EQUATIONS. Ecology. 2003;84(1):252-62. Available from: https://esajournals.onlinelibrary.wiley.com/doi/abs/10.1890/0012-9658%282003%29084%5B0252%3APGISAT%5D2.0.CO%3B2.

[11] Karamched BR, Ott W, Timofeyev I, Alnahhas RN, Bennett MR, Josić K. Moran model of spatial alignment in microbial colonies. Physica D: Nonlinear Phenomena. 2019;395:1-6. Available from: https://www.sciencedirect.com/science/article/pii/S0167278918304627.

[12] Casás-Selves J M & Degregori. How cancer shapes evolution, and how evolution shapes cancer. 2011.

[13] Kholodnyy V, Gadêlha H, Cosson J, Boryshpolets S. How do freshwater fish sperm find the egg? The physicochemical factors guiding the gamete encounters of externally fertilizing freshwater fish. Reviews in Aquaculture. 2019;12(2):1165-92. Available from: https://api.semanticscholar.org/CorpusID:202014791.

[14] Blythe RA, McKane AJ. Stochastic models of evolution in genetics, ecology and linguistics. Journal of Statistical Mechanics: Theory and Experiment. 2007 Jul;2007(07):P07018–P07018.

[15] Doob JL. Topics in the theory of Markoff chains. Trans Amer Math Soc. 1942;52:37-64. Available from: https://doi.org/10.2307/1990152.

[16] Norris JR. 2. In: Continuous-time Markov chains I. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press; 1997. p. 60–107.

[17] Logemann H, Ryan EP. Ordinary Differential Equations. Springer London; 2014.

[18] Moran PAP. Random processes in genetics. Mathematical Proceedings of the Cambridge Philosophical Society. 1958;54(1):60–71.

[19] Chalub FACC, Souza MO. The continuous limit of the Moran process and the diffusion of mutant genes in infinite populations; 2006.

[20] Parsons TL, Quince C. Fixation in haploid populations exhibiting density dependence II: The quasi-neutral case. Theoretical Population Biology. 2007;72(4):468-79. Available from: https://www.sciencedirect.com/science/article/pii/S0040580907000421.

[21] Parsons TL, Quince C. Fixation in haploid populations exhibiting density dependence I: The non-neutral case. Theoretical Population Biology. 2007;72(1):121-35. Available from: https://www.sciencedirect.com/science/article/pii/S0040580906001638.

[22] Gillespie DT. Exact stochastic simulation of coupled chemical reactions. The Journal of Physical Chemistry. 1977;81(25):2340-61. Available from: https://doi.org/10.1021/j100540a008.

[23] Gokhale CS, Hauert C. Eco-evolutionary dynamics of social dilemmas. Theoretical Population Biology. 2016;111:28-42. Available from: https://www.sciencedirect.com/science/article/pii/S0040580916300211.

[24] Pelinovsky E, Kurkin A, Kurkina O, Kokoulina M, Epifanova A. Logistic equation and COVID-19. Chaos, Solitons & Fractals. 2020;140:110241. Available from: https://www.sciencedirect.com/science/article/pii/S0960077920306378.

[25] Peter Olofsson MA. 2. In: Random Variables. John Wiley & Sons, Ltd; 2012. p. 76-155.

A Appendix – Notationslista

α	Startdistribution
$\hat{\mu}$	stickprovsmedelvärdet
λ	Delningstakten
$\mathbb{E}[X]$	Väntevärdet av slumpvariabeln X
$\mathbb{P}(X(t) = i)$	Sannolikheten att den stokastiska variabeln X befinner sig i tillstånd i vid tid t
μ	Väntevärdet eller medelvärdet
σ	Standardavvikelse
φ_A	Sannolikheten att en ensam A individ tar över en population som enbart består av B individer
a, b	delningstakter för A respektive B populationer
E	Felet vid simulering
E'	Det normerade felet
E_n	Sekvens av oberoende slumpvariabler för $n \geq 0$
i	Antalet individer i populationen
i	Tillstånd i som tillhör utfallsrummet S
i, k	Tillstånd
N	Antalet observerade värden, populationens totala storlek
n	Mängden individer i den obegränsadepopulationen
$N(t)$	medelvärdet av S realiseringar vid tid t
p	Frekvensfunktion
P_i	Sannolikheten att befinna sig i tillstånd i
Q	Övergångsmatris
$q_{i,j}$	Övergångsintensiteten för tillstånd i till j
$q_{i,j}$	Övergångsintensiteten mellan tillstånden i till j
$Q_{n,(n+1)}$	Övergångsintensiteten från n till $n+1$ individer
S	Antalet simuleringar
S	Utfallsrummet
s	stickprovsstandardavvikelsen
T	Sluttid för en simulering
T_0	Starttiden för en CTMC
T_n	Tid då en CTMC har genomgått n övergångar
W_0	Tiden som en CTMC befinner sig i tillstånd $X(0)$ enligt Gillespies algoritm
W_n	Tiden som en CTMC befinner sig i tillstånd $X(N)$ enligt Gillespies algoritm
X	Väntevärdet för en diskret slumpvariabel, med värden i mängden $\{x_1, x_2, ...\}$. I en obegränsad population är det antalet individer av en haploid art.

x'	Förändringen av populationen för genotyp A
$X(t)$	Populationsstorleken vid tiden $t \in \mathbb{R},\ t \geq 0$
$x(t)$	Antalet individer av genotyp A
x_0	Startpopulation för genotyp A
y'	Förändringen av populationen för genotyp B
$y(t)$	Antalet individer av genotyp B
y_0	Startpopulation för genotyp B

B Appendix – Härledningar

B.1 Fixeringssannolikhet av den större populationen

Fixeringssannolikheten för genotyp B om det finns 1 individ av genotyp B och $N-1$ individer av genotyp A är

$$\varphi_B = \frac{\prod_{k=1}^{N-1} \gamma_k}{1 + \sum_{j=1}^{N-1} \prod_{k=1}^{j} \gamma_k}. \tag{B.1}$$

För $\lambda \neq 1$ blir

$$\varphi_B = \frac{1-\lambda}{1-\lambda^N} \tag{B.2}$$

och kvoten mellan fixeringssannolikheterna blir $\varphi_B/\varphi_A = \lambda^{1-N}$ där φ_A syftar till ekvation 2.17

B.2 Väntevärden och varians av slumpvariabler

Väntevärdet av en populationsstorlek kan definieras både för diskreta och kontinuerliga slumpvariabler. Slumpvariablerna i detta projekt är populationsstorlekar och förhållanden mellan olika populationsstorlekar [25].

Definition B.1 *Väntevärdet för en diskret slumpvariabel, X med värden i mängden $\{x_1, x_2, \dots\}$ (mängden kan vara både ändlig och uppräkneligt oändlig) med frekvensfunktion p definieras som*

$$\mathbb{E}[X] = \sum_{k=1}^{\infty} x_k p(x_k).$$

Frekvensfunktionen talar om hur sannolika värdena $x_1, x_2, ..., x_k, ..$ är i förhållande till varandra.

Varians är ett mått på hur mycket populationsstorleken förväntas avvika från väntevärdet vid en viss tidpunkt.

Definition B.2 *Variansen för en slumpvariabel X med väntevärde μ definieras som*

$$Var[X] = \mathbb{E}\left[(X-\mu)^2\right] = \mathbb{E}\left[X^2\right] - \left(\mathbb{E}[X]\right)^2.$$

Ofta används begreppen medelvärde för väntevärdet och standardavvikelse för avvikelsen från medelvärdet. Medelvärdet betecknas ofta med μ och standardavvikelsen betecknas oftast med σ och definieras som $\sigma = \sqrt{Var[X]}$.

I de fall då medelvärdet eller väntevärdet eller båda är okända kan istället variansen och medelvärdet skattas genom att man analyserar ett stickprov av observerade värden från data. Då fås stickprovsmedelvärde samt stickprovsstandardavvikelse.

Definition B.3 *Stickprovsmedelvärdet μ för ett stickprov bestående av de observerade värdena $\{x_1, x_2, .., x_N\}$ definieras som*

$$\hat{\mu} = \frac{1}{N} \sum_i x_i. \tag{B.3}$$

Då antalet observerade värden ökar så går $\hat{\mu}$ mot μ. Detta kallas stora talens lag [25].

Definition B.4 *Stickprovsstandardavvikelsen s för ett stickprov bestående av de observerade värdena $\{x_1, x_2, .., x_N\}$ med stickprovsmedelvärde $\hat{\mu}$ definieras som*

$$s^2 = \frac{1}{N-1} \sum_{i=1}^{N} (x_i - \hat{\mu})^2.$$

Med standardavvikelsen kan det undersökas hur många av simuleringarna av populationsstorleken som hamnar innanför det område som den förväntade avvikelsen från väntevärdet beskriver.

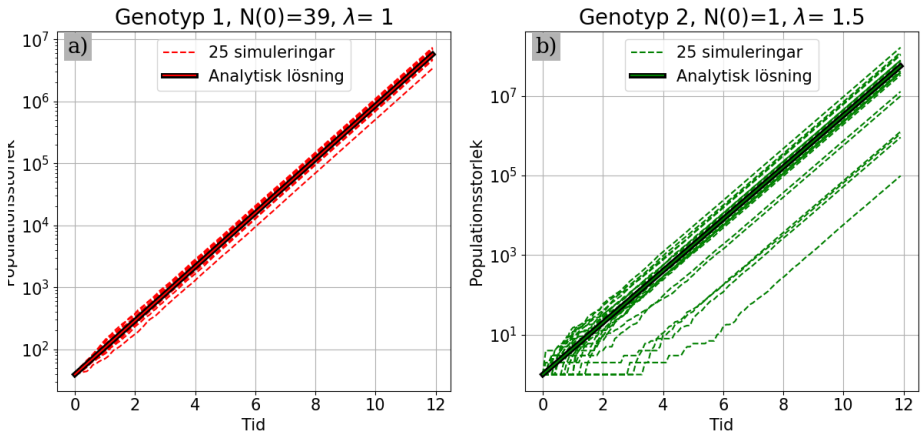
Variansen för den obegränsade modellen kan härledas på likande sätt som egenvärdet i härledning 2.5 i enlighet med definition B.2

Härledning B.5

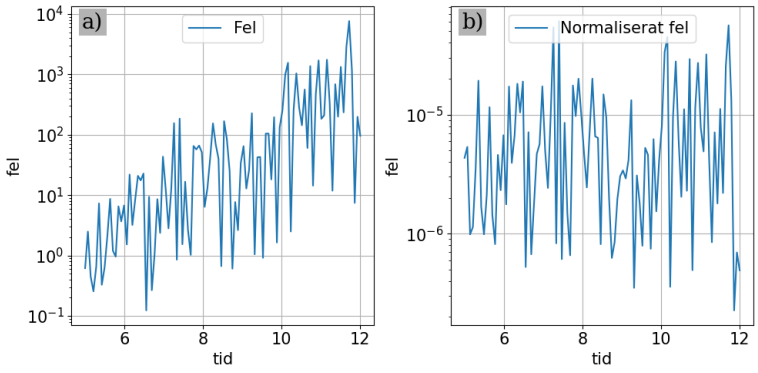
$$
\begin{aligned}
Var[X(t)] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2 &= \sum_{i=1}^{\infty} i^2 P_i(t) - \left(\sum_{i=1}^{\infty} i P_i(t) \right)^2 \\
&= e^{-\lambda t} \sum_{i=1}^{\infty} i^2 \left(1 - e^{-\lambda t}\right)^{i-1} - e^{2\lambda t} \\
&= \left[x = 1 - e^{-\lambda t}\right] = e^{-\lambda t} \sum_{i=1}^{\infty} i^2 x^{i-1} - e^{2\lambda t} \\
&= e^{-\lambda t} \sum_{i=1}^{\infty} \left(i(i-1)x^{i-1} + i x^{i-1} \right) - e^{2\lambda t} = \\
&= e^{-\lambda t} \left(x \sum_{i=2}^{\infty} i(i-1)x^{i-2} + \sum_{i=1}^{\infty} i x^{i-1} \right) - e^{2\lambda t} \\
&= e^{-\lambda t} \left(x \sum_{i=0}^{\infty} \frac{d^2}{dx^2} x^i + \sum_{i=0}^{\infty} \frac{d}{dx} x^i \right) - e^{2\lambda t} = [|x| < 1] \\
&= e^{-\lambda t} \left(\frac{2x}{(1-x)^3} + \frac{1}{(1-x)^2} \right) - e^{2\lambda t} \\
&= e^{-\lambda t} \left(\frac{2\left(1 - e^{-\lambda t}\right)}{\left(1 - (1 - e^{-\lambda t})\right)^3} + \frac{1}{\left(1 - (1 - e^{-\lambda t})\right)^2} \right) - e^{2\lambda t} \\
&= \frac{2e^{-\lambda t}\left(1 - e^{-\lambda t}\right)}{e^{-3\lambda t}} + e^{\lambda t} - e^{2\lambda t} = 2e^{2\lambda t} - 2e^{\lambda t} + e^{\lambda t} - e^{2\lambda t} \\
&= e^{2\lambda t} - e^{\lambda t} = e^{\lambda t}\left(e^{\lambda t} - 1\right)
\end{aligned}
$$

C Appendix – Figurer

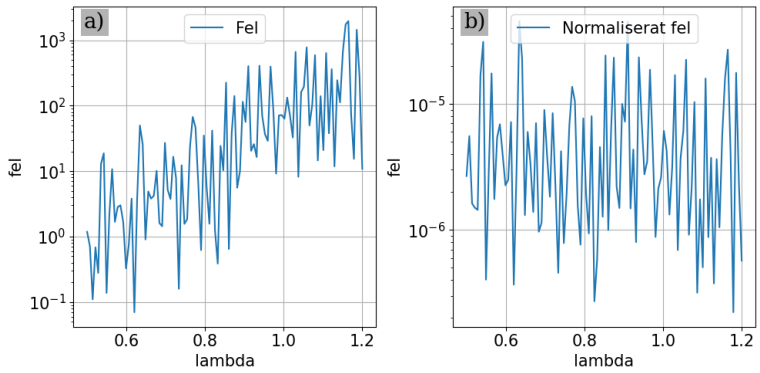
C.1 Obegränsade modellen



Figur 11: Simuleringar av den obegränsade modellen där genotyp 1 har delningstakt 1 och startpopulation 39 medan genotyp 2 har delningstakt 1.5 och startpopulation 1. Genotyp 1:s populationsrealiseringar är 25 röda streckade linjer i a) och genotyp 2:s populationsrealiseringar är 25 gröna streckade linjer i b). I både a) och b) åskådliggörs även den analytiska lösningen 2.5 som heldragna linjer.

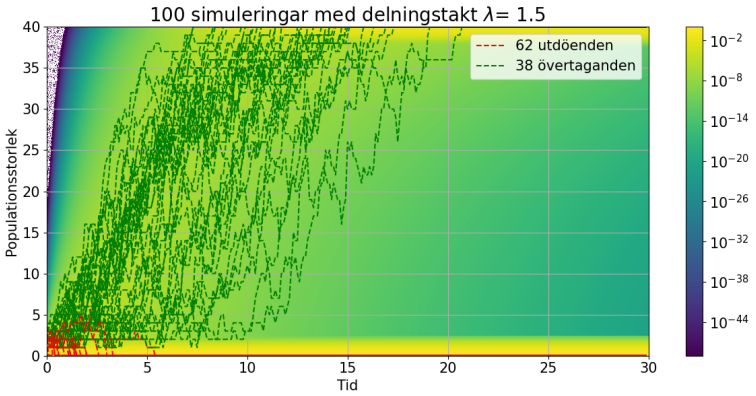


Figur 12: Skillnaden mellan medelvärdet av de olika realiseringarna och den analytiska lösningen 2.5 med delningstakt 0.5 beroende på simuleringarnas sluttid. Felet beskrivs av 1000 realiseringar med en startpopulation på 100. Notera den logaritmiska skalan på y-axeln. a) visar det absoluta felet och b) visar det normaliserade felet.

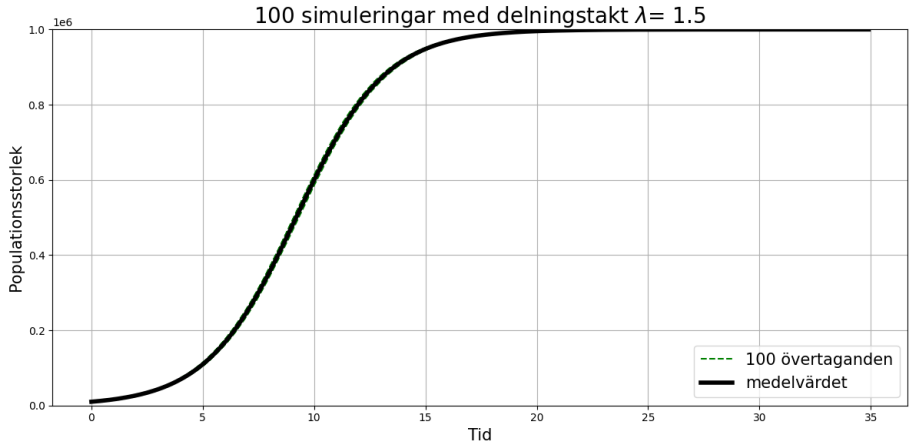


Figur 13: Skillnaden mellan medelvärdet av flera realiseringar och den analytiska lösningen 2.5. Parametrarna är $T = 5$, startpopulation 100 medan tillväxthastigheten varierar längs med x-axeln. För varje värde på tillväxthastigheten togs medelvärdet fram från 1000 körningar. a) visar det absoluta felet och b) visar det normaliserade felet.

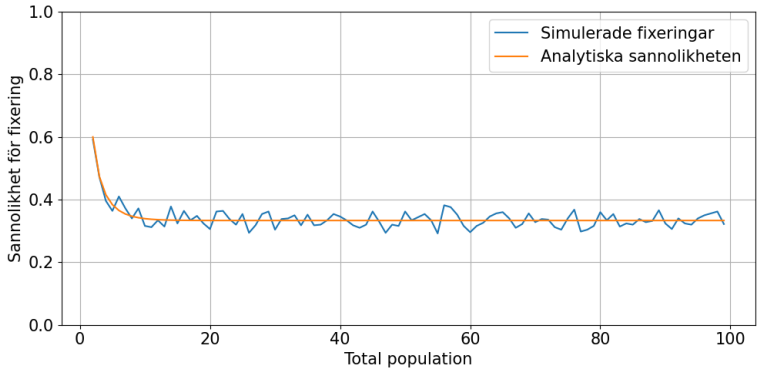
C.2 Moranmodellen



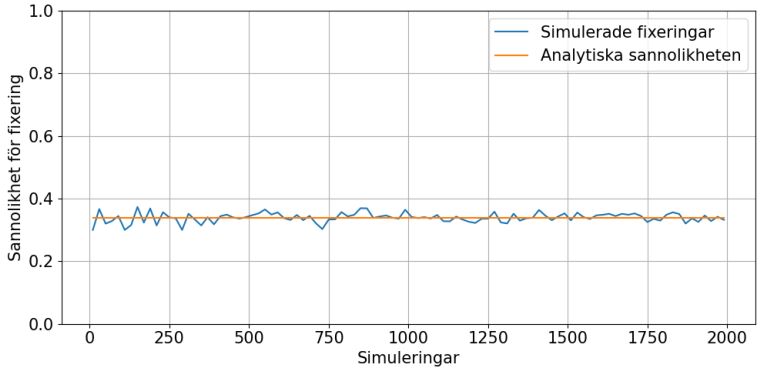
Figur 14: Resultatet av samma 100 körningar av Moranprocessen som i figur 6 med ett färgdiagram som visar sannolikheten för en enskild simulering att befinna sig i varje tillstånd vid en given tid, notera den logaritmiska skalan. Den brutna färgen i övre vänstra hörnet är ett fel som beror på att sannolikheten är så låg för en realisering att befinna sig där att datorn betraktar den som 0, vilket skapar problem på den logaritmiska skalan.



Figur 15: Simulering av Moran modellen med en population på 1 000 000 individer varav 1% (=10 000 individer) av den muterade genotypen med tillväxthastighet 1.5, resten har tillväxthastighet 1.

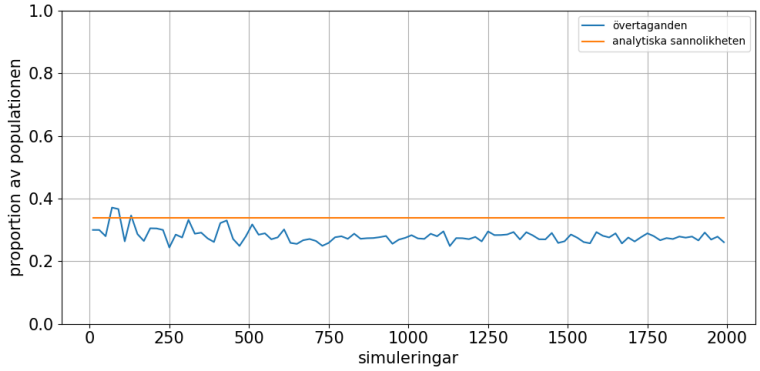


Figur 16: Den förväntade fixeringssannolikheten för olika värden på genotyp 1s startpopulation med parametrar $N_2 = 1$, $\lambda_1 = 1$, $\lambda_2 = 1.5$. Den blåa linjen är medelvärdet av 500 simuleringar för varje värde på N_1 medan den orange linjen är den analytiska lösningen för fixeringssannolikhet beräknad med hjälp av ekvation 2.17

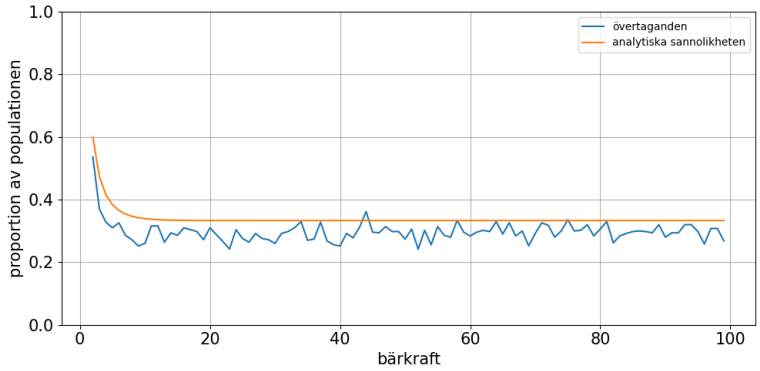


Figur 17: Den förväntade fixeringssannolikheten för olika antal simuleringar med parametrar $N_1 = 9$, $N_2 = 1$, $\lambda_1 = 1$, $\lambda_2 = 1.5$. Den blåa linjen är medelvärdet av olika antal simuleringar medan den orange linjen är den analytiska lösningen för fixeringssannolikhet beräknad med hjälp av ekvation 2.17

C.3 Den kombinerade modellen



Figur 18: Fixeringssannolikheten för den kombinerade modellen med bärkapacitet 10, $N_1 = 9$, $N_2 = 1$, $\lambda_1 = 1$ och $\lambda_2 = 1.5$. Antalet simuleringar varierar längs med x-axeln. Den blåa linjen är den simulerade fixeringsandelen för genotyp 2 och den orange linjen är den analytiska sannolikheten som fås från 2.17.



Figur 19: Fixeringssannolikheten för den kombinerade modellen med $N_2 = 1$, $\lambda_1 = 1$ och $\lambda_2 = 1.5$. Bärkraften K varierar längs med x-axeln, och därmed också $N_1 = K - 1$. Den blåa linjen är den simulerade fixeringsandelen för genotyp 2 och den orange linjen är den analytiska sannolikheten som fås från 2.17. Den blåa linjen kommer från 500 realiseringar för varje värde av λ_2.

D Appendix – Källkod

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patheffects as pe
import time
import scipy as sp
import matplotlib.transforms as mtransforms


class EXPONENTIAL_GROWTH():
    def __init__(self,
                 NUMBER_OF_SIMULATIONS = 100,
                 INITIAL_POPULATIONS = [1],
                 GROWTH_RATES = [1],
                 END_TIME = 5,
                 dt = 0.1,
                 COLORS = None):
        self.NUMBER_OF_SIMULATIONS = NUMBER_OF_SIMULATIONS
        self.INITIAL_POPULATIONS = INITIAL_POPULATIONS
        self.GROWTH_RATES = GROWTH_RATES
        self.END_TIME = END_TIME
        self.dt = dt
        self.COLORS = COLORS
        self.TIMES=np.arange(0,self.END_TIME,self.dt)
        self.LEN_TIMES = len(self.TIMES)

        self.NUMBER_OF_SPECIES = len(self.INITIAL_POPULATIONS)
        assert len(self.INITIAL_POPULATIONS) == len(self.GROWTH_RATES)
        self.SIMULATIONS=np.zeros((self.NUMBER_OF_SPECIES,self.
    NUMBER_OF_SIMULATIONS,self.LEN_TIMES))

        if self.COLORS == None:
            self.COLORS = ['r']*self.NUMBER_OF_SPECIES

    def SIMULATE(self):
        """
        Simulates a continuous time Markov chain using Gillespie's algorithm
        """
        self.SIMULATIONS=np.zeros((self.NUMBER_OF_SPECIES,self.
    NUMBER_OF_SIMULATIONS,self.LEN_TIMES))

        for SPECIES in range(self.NUMBER_OF_SPECIES):
            LAMBDA = self.GROWTH_RATES[SPECIES]
            for SIMUL in range(self.NUMBER_OF_SIMULATIONS):
                # print(f"SIMULATING species {SPECIES+1}: {100*SIMUL/self.
    NUMBER_OF_SIMULATIONS:.2f}%",end="\r", flush=True)
                TIME_TOTAL = 0
                POPULATION_SIZE = self.INITIAL_POPULATIONS[SPECIES]
                self.SIMULATIONS[SPECIES,SIMUL,0] = POPULATION_SIZE
                n = 1
                while TIME_TOTAL <= self.END_TIME:
                    TIMESTEP_PARAMETERS = [LAMBDA*pop for pop in range(
    POPULATION_SIZE,2*POPULATION_SIZE)] #LAMBDA*pop*(1-sum(pops)/K)
                    TIMESTEPS = TIME_TOTAL+np.cumsum(np.random.exponential(np.
    divide(1,TIMESTEP_PARAMETERS))) #cumulative sum of timesteps until the
    population has doubled
                    POP_INCREASE = np.searchsorted(TIMESTEPS,n*self.dt,'right') #
    find the next relevant population increase

                    while POP_INCREASE < POPULATION_SIZE and n < self.LEN_TIMES:
                        self.SIMULATIONS[SPECIES,SIMUL,n] = POPULATION_SIZE +
    POP_INCREASE
                        n +=1
                        POP_INCREASE = np.searchsorted(TIMESTEPS,n*self.dt,'right')

                    TIME_TOTAL = TIMESTEPS[-1]
                    POPULATION_SIZE *= 2


```

```
61
62          self.MEANS = np.mean(self.SIMULATIONS, axis = 1)
63          self.STDS = np.std(self.SIMULATIONS, axis = 1)
64
65          self.TIME_MATRIX = np.resize(self.TIMES, (self.NUMBER_OF_SIMULATIONS,self.
    LEN_TIMES))
66          return
67
68
69      def plot_individuals(self, ANALYTICAL = True,PLOT = False, LOG = False,
    PLOT_MEAN = False, file_name = "graphs/exponential_growth/sim_individual"):
70          """Plots all simulations with deterministic solutions highlighted"""
71
72          FIGURE, AXES = plt.subplots(1,self.NUMBER_OF_SPECIES, figsize = (12,6))
73
74          analytical = np.zeros((self.NUMBER_OF_SPECIES,self.LEN_TIMES))
75          for SPECIES in range(self.NUMBER_OF_SPECIES):
76              analytical[SPECIES] = self.INITIAL_POPULATIONS[SPECIES]*np.exp(self.
    TIMES*self.GROWTH_RATES[SPECIES])
77
78              AXES[SPECIES].plot(self.TIME_MATRIX.T,self.SIMULATIONS[SPECIES,:,:].T,
79                                 linestyle = '--',
80                                 c = self.COLORS[SPECIES],
81                                 label = f"{self.NUMBER_OF_SIMULATIONS} simuleringar")
82              if PLOT_MEAN:
83                  AXES[SPECIES].plot(self.TIMES,
84                                     self.MEANS[SPECIES],
85                                     c = 'k',
86                                     linewidth = 4,
87                                     label = "Medelvärdet av simuleringar")
88
89              if ANALYTICAL:
90                  AXES[SPECIES].plot(self.TIMES,
91                                                 analytical[SPECIES],
92                                                 c = self.COLORS[SPECIES],
93                                                 path_effects=[pe.Stroke(linewidth=5,
    foreground='k'), pe.Normal()],
94                                                 label = "Analytisk lösning")
95
96
97              AXES[SPECIES].set_title(f'Genotyp {SPECIES+1}, N(0)={self.
    INITIAL_POPULATIONS[SPECIES]}, '+r'$\lambda$'+f'= {self.GROWTH_RATES[SPECIES]}'
    , fontsize = 20)
98
99          label = ['a)','b)']
100         trans = mtransforms.ScaledTranslation(10/72, -5/72, FIGURE.dpi_scale_trans)
101
102         for lab, ax in zip(label,AXES):
103             ax.set_xlabel('Tid', fontsize = 15)
104             ax.set_ylabel('Populationsstorlek', fontsize = 15)
105             ax.grid()
106             self.legend_without_duplicate_labels(ax, loc = 'upper center')
107             ax.text(0.0, 1.0, lab, transform=ax.transAxes + trans,
108                 fontsize = 20, verticalalignment='top', fontfamily='serif',
109                 bbox=dict(facecolor='0.7', edgecolor='none', pad=3.0))
110             ax.tick_params(axis='both', which='major', labelsize=15)
111             ax.tick_params(axis='both', which='minor', labelsize=15)
112             if LOG:
113                 ax.set_yscale('log')
114
115         FIGURE.tight_layout()
116
117         if PLOT:
118             plt.show()
119         if file_name:
120             plt.savefig(file_name)
121             [ax.set_yscale('log') for ax in AXES]
122             plt.savefig(file_name+'_log')
123         return
124
```

```
125     def plot_ratios(self, PLOT = False, file_name = "graphs/exponential_growth/
    sim_ratios"):
126         """Plots the ratios of the different species with deterministic solutions
    highlighted"""
127         FIGURE, AXES = plt.subplots(1,1, figsize = (12,6))
128 
129         TOTAL_POP_DETERMINISTIC = [sum(self.INITIAL_POPULATIONS[i]*np.exp(t*self.
    GROWTH_RATES[i]) for i in range(self.NUMBER_OF_SPECIES)) for t in self.TIMES]
130         TOTAL_POP_SIMULATED = np.sum(self.SIMULATIONS,0)
131         AXES.plot(self.TIME_MATRIX.T,self.SIMULATIONS[1,:,:].T/TOTAL_POP_SIMULATED.
    T,
132                           linestyle = '--',
133                           c = self.COLORS[1],
134                           label = f"{self.NUMBER_OF_SIMULATIONS} simuleringar")
135 
136         AXES.plot(self.TIMES,
137                           self.INITIAL_POPULATIONS[1]*np.exp(self.TIMES*self.
    GROWTH_RATES[1])/TOTAL_POP_DETERMINISTIC,
138                           linestyle = '-',
139                           c = self.COLORS[1],
140                           path_effects=[pe.Stroke(linewidth=5, foreground='k'), pe.
    Normal()],
141                           label = "Analytisk lösning")
142 
143         AXES.set_title(f'Proportionen av genotyp {2} med N(0)={self.
    INITIAL_POPULATIONS[1]}, '+r'$\lambda$'+f'={self.GROWTH_RATES[1]}', fontsize =
    20)
144 
145 
146 
147         AXES.set_xlabel('Tid', fontsize = 15)
148         AXES.set_ylabel('Muterad andel av populationen', fontsize = 15)
149         AXES.set_ylim([0,1])
150         AXES.tick_params(axis='both', which='major', labelsize=15)
151         AXES.tick_params(axis='both', which='minor', labelsize=15)
152         self.legend_without_duplicate_labels(AXES)
153         AXES.grid()
154 
155         FIGURE.tight_layout()
156         if PLOT:
157             plt.show()
158         if file_name:
159             plt.savefig(file_name)
160         return
161 
162     def plot_simultaneous(self, PLOT = False, file_name = "graphs/
    exponential_growth/sim_simultaneous"):
163         """ Plots all simulations in the same figure with deterministic solutions
    highlighted. """
164 
165         FIGURE, AXES = plt.subplots(1,1, figsize = (12,6))
166 
167         for SPECIES in range(self.NUMBER_OF_SPECIES):
168             AXES.plot(self.TIME_MATRIX.T,
169                     self.SIMULATIONS[SPECIES,:,:].T,
170                     alpha = 0.3,
171                     c = self.COLORS[SPECIES],
172                     linestyle = '--',
173                     label = f"{self.NUMBER_OF_SIMULATIONS} simuleringar av genotyp
    {SPECIES+1}")
174 
175             AXES.plot(self.TIMES,
176                     self.INITIAL_POPULATIONS[SPECIES]*np.exp(self.TIMES*self.
    GROWTH_RATES[SPECIES]),
177                     c = self.COLORS[SPECIES],
178                     linewidth=2.0,
179                     zorder=10,
180                     path_effects=[pe.Stroke(linewidth=5, foreground='k'), pe.Normal
    ()],
181                     label = f"Analytisk lösning för genotyp {SPECIES+1} ")
```

```
182
183         AXES.set_xlabel('Tid', fontsize = 15)
184         AXES.set_ylabel('Populationsstorlek', fontsize = 15)
185         AXES.tick_params(axis='both', which='major', labelsize=15)
186         AXES.tick_params(axis='both', which='minor', labelsize=15)
187         AXES.grid()
188         self.legend_without_duplicate_labels(AXES)
189         FIGURE.tight_layout()
190
191         if PLOT:
192             plt.show()
193         if file_name:
194             plt.savefig(file_name)
195             AXES.set_yscale('log')
196             plt.savefig(file_name+'_log')
197         return
198
199     def legend_without_duplicate_labels(self,AXIS, loc = 'best'):
200         """ Writes all labels in AXIS without duplicates, source: https://
    stackoverflow.com/questions/19385639/duplicate-items-in-legend-in-matplotlib
    """
201         handles, labels = AXIS.get_legend_handles_labels()
202         unique = [(h, l) for i, (h, l) in enumerate(zip(handles, labels)) if l not
    in labels[:i]]
203         AXIS.legend(*zip(*unique), fontsize = 15, loc = loc)
204         return
205
206 def SIM_VARIATION(EG, attribute_function, attribute_values):
207     file_name = f"graphs/exponential_growth/{(attribute := attribute_function(EG,
    attribute_values[-1]))[0]}"
208
209     INITIALIZE_EG_ERRORS(EG)
210
211     ATTR_LEN = len(attribute_values)
212     E = np.zeros(ATTR_LEN)
213     Ep = np.zeros(ATTR_LEN)
214
215     for val in range(ATTR_LEN):
216         attribute_value = attribute_values[val]
217         attribute_function(EG, attribute_value)
218         EG.SIMULATE()
219         mean = EG.MEANS
220
221         analytical = EG.INITIAL_POPULATIONS[0]*np.exp(EG.TIMES*EG.GROWTH_RATES[0])
222
223         E[val] = 1/EG.END_TIME*sp.integrate.trapezoid((mean-analytical)**2, dx = EG
    .dt)
224         Ep[val] = 1/EG.END_TIME*sp.integrate.trapezoid(((mean-analytical)/mean)**2,
     dx = EG.dt)
225         print(f"Evaluating {attribute[1]}: {100*val/ATTR_LEN:.2f}%",end="\r", flush
    =True)
226
227     f, axs = plt.subplots(1,2, figsize = (10,5))
228
229     axs[0].plot(attribute_values, E, label = 'Fel')
230     axs[1].plot(attribute_values, Ep, label = 'Normaliserat fel')
231
232     label = ['a)','b)']
233     trans = mtransforms.ScaledTranslation(10/72, -5/72, f.dpi_scale_trans)
234     for lab, ax in zip(label,axs):
235         ax.legend(fontsize = 15, loc = 'upper center')
236         ax.grid()
237         ax.set_xlabel(attribute_function(EG,attribute_values[-1])[1], fontsize =
    15)
238         ax.set_ylabel("fel", fontsize = 15)
239         ax.tick_params(axis='both', which='major', labelsize=15)
240         ax.tick_params(axis='both', which='minor', labelsize=15)
241         ax.text(0.0, 1.0, lab, transform=ax.transAxes + trans,
242             fontsize = 20, verticalalignment='top', fontfamily='serif',
243             bbox=dict(facecolor='0.7', edgecolor='none', pad=3.0))
```

```
244
245     f.tight_layout()
246
247     plt.savefig(file_name)
248
249     [ax.set_yscale('log') for ax in axs]
250     f.tight_layout()
251     plt.savefig(file_name+'_log')
252
253 def SET_SIMULATIONS(EG, attribute_value):
254     EG.NUMBER_OF_SIMULATIONS = attribute_value
255     return "error_simulations", "simuleringar"
256
257 def SET_LAMBDA(EG, attribute_value):
258     EG.GROWTH_RATES[0] = attribute_value
259     return "error_parameter", "lambda"
260
261 def SET_END_TIME(EG, attribute_value):
262     EG.END_TIME = attribute_value
263     EG.TIMES = np.arange(0,EG.END_TIME,EG.dt)
264     EG.LEN_TIMES = len(EG.TIMES)
265     return "error_end_time", "tid"
266
267 def SET_INITIAL_POP(EG : EXPONENTIAL_GROWTH, attribute_value):
268     EG.INITIAL_POPULATIONS[0] = attribute_value
269     return "error_initial_pop", "initial population"
270
271 def timecheck(times):
272     now = time.time()
273     print(f"TIMECHECK: {now-times[-1]:.2f}")
274     times.append(now)
275     return
276
277 def ERROR_PLOTS(EG: EXPONENTIAL_GROWTH, times):
278     TIME_VALUES = np.linspace(5,12,100)
279     LAMBDA_VALUES = np.linspace(0.5,1.2, 100)
280     SIM_VALUES = np.arange(100,10000, 10)
281     POP_VALUES = np.arange(1,500,5)
282
283     INITIALIZE_EG_ERRORS(EG)
284
285     SIM_VARIATION(EG,SET_INITIAL_POP, POP_VALUES)
286     print("Finished initial population variation")
287     timecheck(times)
288     INITIALIZE_EG_ERRORS(EG)
289
290     SIM_VARIATION(EG,SET_END_TIME, TIME_VALUES)
291     print("Finished time variation")
292     timecheck(times)
293
294     SIM_VARIATION(EG,SET_LAMBDA, LAMBDA_VALUES)
295     print("Finished lambda variation")
296     timecheck(times)
297
298     SIM_VARIATION(EG,SET_SIMULATIONS, SIM_VALUES)
299     print("Finished simulation variation")
300     timecheck(times)
301
302
303 def INITIALIZE_EG_ERRORS(EG):
304     EG.__init__(
305     NUMBER_OF_SIMULATIONS = 1000,
306     INITIAL_POPULATIONS = [100],
307     GROWTH_RATES = [0.5],
308     END_TIME = 5,
309     dt = 0.1
310     )
311
312 def INITIALIZE_EG_SIMS(EG):
313     EG.__init__(
```

```
    NUMBER_OF_SIMULATIONS = 25,
    INITIAL_POPULATIONS = [39,1],
    GROWTH_RATES = [1,1.5],
    END_TIME = 12,
    dt = 0.1,
    COLORS = ['r','g']
    )

if __name__ == '__main__':
    time1 = time.time()
    times = [time1]
    EG = EXPONENTIAL_GROWTH()

    ERROR_PLOTS(EG,times)

    # INITIALIZE_EG_SIMS(EG)
    # EG.SIMULATE()
    # EG.plot_individuals()
    # EG.plot_simultaneous()
    # EG.plot_ratios()

    time2 = time.time()

    print(f"TOTAL TIME: {time2-time1:.2f}")
```

Kodfil 1: Kod för den obegränsade modellen.

```
import numpy as np
import scipy as sp
import matplotlib.pyplot as plt
from matplotlib.ticker import ScalarFormatter
import time

class MORAN_PROCESS:
    def __init__(self,
                 NUMBER_OF_SIMULATIONS = 100,
                 TOTAL_POP = 10,
                 START_POP = 1,
                 GROWTH_RATE = 2,
                 END_TIME = 10,
                 dt = 1):
        self.NUMBER_OF_SIMULATIONS = NUMBER_OF_SIMULATIONS
        self.TOTAL_POP = TOTAL_POP
        self.START_POP = START_POP
        self.GROWTH_RATE = GROWTH_RATE
        self.END_TIME = END_TIME
        self.dt = dt

        self.update_Q()
        self.EXTINCTIONS = np.zeros(NUMBER_OF_SIMULATIONS)


    def SIMULATE(self):
        """
        Simulates a continuous time Markov chain using Gillespie's algorithm
        """
        self.deaths = 0
        self.takeovers = 0

        self.TIMES=np.arange(0,self.END_TIME,self.dt)
        LEN_TIMES = len(self.TIMES)
        self.SIMULATIONS=np.zeros((self.NUMBER_OF_SIMULATIONS,LEN_TIMES))

        LAMBDA = self.GROWTH_RATE
        N = self.TOTAL_POP
        for SIMUL in range(self.NUMBER_OF_SIMULATIONS):
            # print(f"SIMULATING population of {N}: {100*SIMUL/self.
    NUMBER_OF_SIMULATIONS:.2f}%",end="\r", flush=True)
            TIME_TOTAL = 0
            i = self.START_POP
            while TIME_TOTAL <= self.END_TIME:
```

```
                P_plus = LAMBDA*i*(N-i)/N
                P_minus = (N-i)*i/N

                TIMESTEP = np.random.exponential(1/(P_plus+P_minus))

                if(np.random.uniform() > P_plus/(P_plus+P_minus)):
                    i-=1
                else:
                    i+=1

                self.SIMULATIONS[SIMUL,int(np.ceil(TIME_TOTAL/self.dt)):int(np.ceil
((TIME_TOTAL+TIMESTEP)/self.dt))] = i

                if (i == N or i == 0):
                    self.SIMULATIONS[SIMUL,int((TIME_TOTAL+TIMESTEP)/self.dt):] = i
                    self.takeovers += (takeover := int(i/N)) #0 if i=0, 1 if i=N
                    self.deaths += 1 - takeover #0 if i=N, 1 if i=0
                    self.EXTINCTIONS[SIMUL] = takeover
                    break
                TIME_TOTAL += TIMESTEP

        self.mean = np.mean(self.SIMULATIONS,0)

    def update_Q(self):
        N = self.TOTAL_POP
        pops = np.arange(1,N)
        P_plus = np.concatenate(([0],self.GROWTH_RATE*pops*(N-pops)/N,[0]))
        P_minus = np.concatenate(([0],(N-pops)*pops/N,[0]))
        self.Q = np.diag(-P_plus-P_minus)
        for i in range(1,N):
            self.Q[i,i-1] = P_minus[i]
            self.Q[i,i+1] = P_plus[i]


    def plot_populations(self, PLOT = False, WITH_DETERMINISTIC = True, WITH_MEAN =
 True, WITH_HEATMAP = False, SAVEFIG = "graphs/moran/populations", loc = "best"
):
        """Plots all simulations"""

        extinctions = sum(self.EXTINCTIONS)
        fig, ax = plt.subplots(1,1, figsize = (12,6))

        for SIM in range(self.NUMBER_OF_SIMULATIONS):
            if self.EXTINCTIONS[SIM]:
                COL = 'g'
                label = f"{int(extinctions)} övertaganden"
            else:
                COL = 'r'
                label = f"{int(self.NUMBER_OF_SIMULATIONS - extinctions)} utdöenden
"

            ax.plot(self.TIMES.T,self.SIMULATIONS[SIM,:],
                    linestyle = '--',
                    c = COL,
                    label = label)

        if WITH_HEATMAP or WITH_DETERMINISTIC:
            probs = [self.P(t) for t in self.TIMES]

        if WITH_HEATMAP:
            Z = np.zeros((self.TOTAL_POP+1,len(self.TIMES)))
            for i, t in enumerate(self.TIMES):
                Z[:,i] = np.flip(probs[i])
            plt.imshow(Z, cmap = 'viridis', extent = (0,self.END_TIME,0,self.
TOTAL_POP), aspect ='auto', norm = 'log', interpolation= 'gaussian')
            col = plt.colorbar()
            col.ax.tick_params(labelsize = 15)

        if WITH_DETERMINISTIC:
            deterministic = np.zeros(len(self.TIMES))
```

```
109             for i, t in enumerate(self.TIMES):
110                 deterministic[i] = np.sum([j*probs[i][j] for j in range(self.
    TOTAL_POP+1)])
111             ax.plot(self.TIMES,deterministic, label = 'förväntade medelvärdet', c =
     'r')
112 
113         if WITH_MEAN:
114             ax.plot(self.TIMES,self.mean, label = 'medelvärdet', c = 'k', linewidth
     = 4)
115 
116         ax.set_title(f'{self.NUMBER_OF_SIMULATIONS} simuleringar med delningstakt '
    +r'$\lambda$'+f'= {self.GROWTH_RATE}', fontsize = 20)
117         ax.set_xlabel('Tid', fontsize = 15)
118         ax.set_ylabel('Populationsstorlek', fontsize = 15)
119 
120         self.legend_without_duplicate_labels(ax, loc = loc)
121         ax.grid()
122         ax.set_ylim([0,self.TOTAL_POP])
123         fig.tight_layout()
124         ax.tick_params(axis='both', which='major', labelsize=15)
125         ax.tick_params(axis='both', which='minor', labelsize=15)
126 
127         if PLOT:
128             plt.show()
129         if SAVEFIG:
130             plt.savefig(SAVEFIG + WITH_HEATMAP*'_heatmap')
131         return
132 
133     def legend_without_duplicate_labels(self,AXIS, loc = "best"):
134         """ Writes all labels in AXIS without duplicates, source: https://
    stackoverflow.com/questions/19385639/duplicate-items-in-legend-in-matplotlib
    """
135         handles, labels = AXIS.get_legend_handles_labels()
136         unique = [(h, l) for i, (h, l) in enumerate(zip(handles, labels)) if l not
    in labels[:i]]
137         AXIS.legend(*zip(*unique), fontsize = 15, loc = loc)
138         return
139 
140     def P(self,t):
141         P_0 = np.zeros(self.TOTAL_POP+1)
142 
143         P_0[self.START_POP] = 1
144         exp = sp.linalg.expm(t*self.Q)
145 
146         return P_0 @ exp
147 
148     def get_analytical_fixation_prob(self):
149         if self.GROWTH_RATE == 1:
150             return 1/self.TOTAL_POP
151         else:
152             return (1-1/self.GROWTH_RATE)/(1-1/self.GROWTH_RATE**self.TOTAL_POP)
153 
154 
155 def SIM_VARIATION(MORAN: MORAN_PROCESS, attribute_function: callable,
    attribute_values: list[float]):
156     INITIALIZE_MORAN_PROPORTIONS(MORAN)
157     label = attribute_function(MORAN,attribute_values[-1])
158     ATTR_LEN = len(attribute_values)
159 
160     takeovers = np.zeros(ATTR_LEN, dtype = float)
161     analytical = np.zeros(ATTR_LEN, dtype = float)
162     for i, val in zip(range(ATTR_LEN), attribute_values):
163         attribute_function(MORAN, val)
164         MORAN.SIMULATE()
165         takeovers[i] = MORAN.takeovers/MORAN.NUMBER_OF_SIMULATIONS
166         analytical[i] = MORAN.get_analytical_fixation_prob()
167         print(f"Evaluating {label[1]}: {100*i/ATTR_LEN:.2f}%",end="\r", flush=True)
168 
169     f, ax = plt.subplots(1,1, figsize = (10,5))
170 
```

```
    ax.plot(attribute_values, takeovers, label = 'Simulerade fixeringar')
    ax.plot(attribute_values, analytical, label = 'Analytiska sannolikheten')

    ax.legend(fontsize = 15)
    ax.grid()
    ax.set_xlabel(attribute_function(MORAN,attribute_values[-1])[1], fontsize = 15)
    ax.set_ylabel("Sannolikhet för fixering", fontsize = 15)
    ax.set_ylim([0,1])
    ax.tick_params(axis='both', which='major', labelsize=15)
    ax.tick_params(axis='both', which='minor', labelsize=15)

    plt.tight_layout()
    plt.savefig(f"graphs/moran/{label[0]}")

def SIM_POPULATION_LIMIT(MORAN: MORAN_PROCESS):
    population_sizes = [1e2,1e4,1e6]
    mutated_proportion = 0.01

    INITIALIZE_MORAN_LIMIT(MORAN)

    for i, pop in enumerate(population_sizes):
        MORAN.TOTAL_POP = int(pop)
        MORAN.START_POP = int(pop*mutated_proportion)
        # MORAN.update_Q()

        MORAN.SIMULATE()
        MORAN.plot_populations(SAVEFIG = f'graphs/moran/inf_pop_{i}', loc = "lower
    right", WITH_HEATMAP= False, WITH_DETERMINISTIC= False)



def SET_SIMULATIONS(MORAN, attribute_value):
    MORAN.NUMBER_OF_SIMULATIONS = attribute_value
    MORAN.EXTINCTIONS = np.zeros(attribute_value)
    return "proportion_simulations", "Simuleringar"

def SET_LAMBDA(MORAN, attribute_value):
    MORAN.GROWTH_RATE = attribute_value
    return "proportion_parameter", "Tillväxthastighet"

def SET_POPULATION_VALUE(MORAN, attribute_value):
    MORAN.TOTAL_POP = attribute_value
    return "proportion_total_population", "Total population"

def timecheck(times):
    now = time.time()
    print(f"TIMECHECK: {now-times[-1]:.2f}")
    times.append(now)
    return

def PROPORTION_PLOTS(MORAN, times):
    TOTAL_POPULATION_VALUES = np.arange(2,100,1)
    LAMBDA_VALUES = np.linspace(0.1,5, 100)
    SIM_VALUES = np.arange(10,2000, 20)


    SIM_VARIATION(MORAN,SET_POPULATION_VALUE, TOTAL_POPULATION_VALUES)
    print("Finished population variation")
    timecheck(times)

    SIM_VARIATION(MORAN,SET_LAMBDA, LAMBDA_VALUES)
    print("Finished lambda variation")
    timecheck(times)

    SIM_VARIATION(MORAN,SET_SIMULATIONS, SIM_VALUES)
    print("Finished simulation variation")
    timecheck(times)

def INITIALIZE_MORAN_LIMIT(MORAN: MORAN_PROCESS):
    MORAN.__init__(NUMBER_OF_SIMULATIONS = 100,
```

```
    GROWTH_RATE = 1.5,
    END_TIME = 35,
    dt = 0.1)

def INITIALIZE_MORAN_PROPORTIONS(MORAN : MORAN_PROCESS):
    MORAN.__init__(NUMBER_OF_SIMULATIONS = 500,
    TOTAL_POP = 10,
    GROWTH_RATE = 1.5,
    END_TIME = 50,
    dt = 0.1)

def INITIALIZE_MORAN_SIMS(MORAN : MORAN_PROCESS):
    MORAN.__init__(NUMBER_OF_SIMULATIONS = 100,
    TOTAL_POP = 40,
    START_POP = 1,
    GROWTH_RATE = 1.5,
    END_TIME = 30,
    dt = 0.1)

if __name__ == '__main__':
    time1 = time.time()
    times = [time1]

    MORAN = MORAN_PROCESS()

    PROPORTION_PLOTS(MORAN,times)

    # SIM_POPULATION_LIMIT(MORAN)


    # INITIALIZE_MORAN_SIMS(MORAN)
    # MORAN.SIMULATE()
    # MORAN.plot_populations(WITH_HEATMAP= False, WITH_DETERMINISTIC= True,
    WITH_MEAN= True)
    # MORAN.plot_populations(WITH_HEATMAP= True, WITH_DETERMINISTIC= False,
    WITH_MEAN= False)


    print(f"Fixation rate: {sum(MORAN.EXTINCTIONS)/MORAN.NUMBER_OF_SIMULATIONS}")
    print(f"Expected fixation rate (Moran): {MORAN.get_analytical_fixation_prob()}"
    )

    time2 = time.time()
    print(f"TOTAL TIME: {time2-time1:.2f}")
```

Kodfil 2: Kod för Moranprocessen.

```
import exponential_growth as EG
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patheffects as pe
import time
import scipy as sp

class CARRYING_CAPACITY(EG.EXPONENTIAL_GROWTH):
    def __init__(self,
                 NUMBER_OF_SIMULATIONS = 100,
                 INITIAL_POPULATIONS = [99,1],
                 GROWTH_RATES = [1,1.5],
                 END_TIME = 600,
                 dt = 1,
                 COLORS = None,
                 DEATH_RATES = [0.1,0.1],
                 CAPACITY = 100
                 ):

        super().__init__(NUMBER_OF_SIMULATIONS = NUMBER_OF_SIMULATIONS,
                INITIAL_POPULATIONS = INITIAL_POPULATIONS,
                GROWTH_RATES = GROWTH_RATES,
                END_TIME = END_TIME,
                dt = dt,
```

```
                COLORS = COLORS)

        self.CAPACITY = CAPACITY
        self.DEATH_RATES = DEATH_RATES
        self.EXTINCTIONS = np.zeros(NUMBER_OF_SIMULATIONS)


        return

    def SIMULATE(self, TO_EXTINCTION = True):
        """
        Simulates a continuous time Markov chain using Gillespie's algorithm
        """
        self.EXTINCTIONS = np.zeros(self.NUMBER_OF_SIMULATIONS)
        self.takeovers = 0
        self.SIMULATIONS=np.zeros((self.NUMBER_OF_SPECIES,self.
    NUMBER_OF_SIMULATIONS,self.LEN_TIMES))

        LAMBDA = self.GROWTH_RATES
        MU = self.DEATH_RATES
        K = self.CAPACITY
        for SIMUL in range(self.NUMBER_OF_SIMULATIONS):
            TIME_TOTAL = 0
            POPULATION_SIZES = np.copy(self.INITIAL_POPULATIONS)

            self.SIMULATIONS[:,SIMUL,0] = POPULATION_SIZES
            n = 1
            while TIME_TOTAL <= self.END_TIME:
                D_RATES = np.multiply(MU,POPULATION_SIZES)
                G_RATES = np.multiply(LAMBDA,POPULATION_SIZES)*(1-sum(
    POPULATION_SIZES)/K)
                TIMESTEP_PARAMETER = sum(D_RATES)+sum(G_RATES)
                TIMESTEP = np.random.exponential(1/TIMESTEP_PARAMETER)
                rand = np.random.uniform()
                if (rand<D_RATES[0]/TIMESTEP_PARAMETER):
                    POPULATION_SIZES[0] -= 1
                    if POPULATION_SIZES[0] == 0:
                        self.EXTINCTIONS[SIMUL] = 1
                        if TO_EXTINCTION:
                            self.SIMULATIONS[:,SIMUL,n:] = np.tile(POPULATION_SIZES
    .T,(self.LEN_TIMES-n,1)).T
                            break
                elif (rand < (D_RATES[0] + D_RATES[1])/TIMESTEP_PARAMETER):
                    POPULATION_SIZES[1] -= 1
                    if POPULATION_SIZES[1] == 0 and TO_EXTINCTION:
                        self.SIMULATIONS[:,SIMUL,n:] = np.tile(POPULATION_SIZES.T,(
    self.LEN_TIMES-n,1)).T
                        break
                elif (rand < (D_RATES[0] + D_RATES[1] + G_RATES[0])/
    TIMESTEP_PARAMETER):
                    POPULATION_SIZES[0] += 1
                else:
                    POPULATION_SIZES[1] += 1
                TIME_TOTAL += TIMESTEP

                while TIME_TOTAL >= n*self.dt and n < self.LEN_TIMES:
                    self.SIMULATIONS[:,SIMUL,n] = POPULATION_SIZES
                    n += 1


        self.MEANS = np.mean(self.SIMULATIONS, axis = 1)
        self.STDS = np.std(self.SIMULATIONS, axis = 1)

        self.TIME_MATRIX = np.resize(self.TIMES, (self.NUMBER_OF_SIMULATIONS,self.
    LEN_TIMES))
        return

    def plot(self, ANALYTICAL = True,PLOT = False, PLOT_MEAN = True, SAVEFIG = "
    graphs/carrying_capacity/sim_individual.png"):
        FIGURE, AXES = plt.subplots(1,1, figsize = (12,6))
```

```
88
89          extinctions = sum(self.EXTINCTIONS)
90          for SIM in range(self.NUMBER_OF_SIMULATIONS):
91              if self.EXTINCTIONS[SIM] == 1:
92                  COL = 'g'
93                  label = f"{int(extinctions)} övertaganden"
94              else:
95                  COL = 'r'
96                  label = f"{int(self.NUMBER_OF_SIMULATIONS-extinctions)} utdöenden"
97
98
99              AXES.plot(self.TIMES,
100                         self.SIMULATIONS[1,SIM,:].T/sum(self.SIMULATIONS[:,SIM,:],0),
101                         linestyle = '--',
102                         c = COL,
103                         label = label)
104         if PLOT_MEAN:
105             AXES.plot(self.TIMES,
106                         self.MEANS[1]/sum(self.MEANS),
107                         c = 'k',
108                         linewidth = 4,
109                         label = "Medelvärdet")
110         if ANALYTICAL:
111             analytical_sol = self.solve_analytical(self.TIMES)
112             AXES.plot(self.TIMES,
113                             (analytical_sol[:,1]/np.sum(analytical_sol,axis = 1)).T
    ,
114                             c = 'g',
115                             linewidth = 3,
116                             path_effects=[pe.Stroke(linewidth=5, foreground='k'),
    pe.Normal()],
117                             label = "ODE lösning")
118
119
120         AXES.set_xlabel('Tid', fontsize = 15)
121         AXES.set_ylabel('Proportion av muterad genotyp', fontsize = 15)
122         AXES.grid()
123         self.legend_without_duplicate_labels(AXES)
124         AXES.tick_params(axis='both', which='major', labelsize=15)
125         AXES.tick_params(axis='both', which='minor', labelsize=15)
126         AXES.set_ylim([0,1])
127         AXES.set_title(f"{self.NUMBER_OF_SIMULATIONS} simuleringar med delningstakt
     "+r'$\lambda$'+f"= {self.GROWTH_RATES[1]}", fontsize = 20)
128         FIGURE.tight_layout()
129
130         if PLOT:
131             plt.show()
132         if SAVEFIG:
133             plt.savefig(SAVEFIG)
134         return
135
136     def get_analytical_fixation_prob(self):
137         lam = self.GROWTH_RATES[1]
138         if lam == 1:
139             return 1/self.CAPACITY
140         else:
141             return (1-1/lam)/(1-1/lam**self.CAPACITY)
142
143     def y_prime(self,y,t):
144         return y[0]*(1-(sum(y))/self.CAPACITY)-y[0]*self.DEATH_RATES[0], self.
    GROWTH_RATES[1]*y[1]*(1-(sum(y))/self.CAPACITY)-y[1]*self.DEATH_RATES[1]
145
146     def solve_analytical(self, t):
147         return sp.integrate.odeint(self.y_prime,self.INITIAL_POPULATIONS,t)
148
149 def SIM_VARIATION(CC: CARRYING_CAPACITY, attribute_function: callable,
    attribute_values: list[float]):
150     INITIALIZE_CC_PROPORTIONS(CC)
151
152     label = attribute_function(CC,attribute_values[-1])
```

```
    print(f"Evaluating {label[1]}: 0%",end="\r", flush=True)
    ATTR_LEN = len(attribute_values)

    takeovers = np.zeros(ATTR_LEN, dtype = float)
    analytical = np.zeros(ATTR_LEN, dtype = float)
    for i, val in zip(range(ATTR_LEN), attribute_values):
        attribute_function(CC, val)
        CC.SIMULATE()
        takeovers[i] = sum(CC.EXTINCTIONS)/CC.NUMBER_OF_SIMULATIONS
        analytical[i] = CC.get_analytical_fixation_prob()
        print(f"Evaluating {label[1]}: {100*i/ATTR_LEN:.2f}%",end="\r", flush=True)
    f, ax = plt.subplots(1,1, figsize = (10,5))

    ax.plot(attribute_values, takeovers, label = 'övertaganden')
    ax.plot(attribute_values, analytical, label = 'analytiska sannolikheten')

    ax.legend()
    ax.grid()
    ax.set_xlabel(attribute_function(CC,attribute_values[-1])[1], fontsize = 15)
    ax.set_ylabel("proportion av populationen", fontsize = 15)
    ax.set_ylim([0,1])
    ax.tick_params(axis='both', which='major', labelsize=15)
    ax.tick_params(axis='both', which='minor', labelsize=15)

    plt.tight_layout()
    plt.savefig(f"graphs/carrying_capacity/{label[0]}")

def SET_SIMULATIONS(CC : CARRYING_CAPACITY, attribute_value):
    CC.NUMBER_OF_SIMULATIONS = attribute_value
    CC.EXTINCTIONS = np.zeros(attribute_value)
    return "proportion_simulations", "simuleringar"

def SET_LAMBDA(CC : CARRYING_CAPACITY, attribute_value):
    CC.GROWTH_RATES = [1,attribute_value]
    return "proportion_parameter", "tillväxthastighet"

def SET_CAPACITY_VALUE(CC : CARRYING_CAPACITY, attribute_value):
    CC.INITIAL_POPULATIONS = [attribute_value-1,1]
    CC.CAPACITY = attribute_value
    return "proportion_capacity", "bärkraft"

def PROPORTION_PLOTS(CC:CARRYING_CAPACITY, times):
    CAPACITY_VALUES = np.arange(2,100,1)
    LAMBDA_VALUES = np.linspace(0.1,5, 100)
    SIM_VALUES = np.arange(10,2000, 20)


    SIM_VARIATION(CC,SET_CAPACITY_VALUE, CAPACITY_VALUES)
    print("Finished capacity variation")
    timecheck(times)

    SIM_VARIATION(CC,SET_LAMBDA, LAMBDA_VALUES)
    print("Finished lambda variation")
    timecheck(times)

    SIM_VARIATION(CC,SET_SIMULATIONS, SIM_VALUES)
    print("Finished simulation variation")
    timecheck(times)

def INITIALIZE_CC_PROPORTIONS(CC: CARRYING_CAPACITY):
    CC.__init__(
        NUMBER_OF_SIMULATIONS = 500,
        CAPACITY = 10,
        INITIAL_POPULATIONS = [9,1],
        GROWTH_RATES = [1,1.5],
        END_TIME = 1000
        )

def INITIALIZE_CC_SIMS(CC: CARRYING_CAPACITY):
    CC.__init__(
```

```
223         NUMBER_OF_SIMULATIONS = 100,
224         CAPACITY = 40,
225         INITIAL_POPULATIONS = [39,1],
226         GROWTH_RATES = [1,1.5],
227         END_TIME = 300
228         )
229
230 def SIMULATE_AND_PLOT(CC: CARRYING_CAPACITY, times : list[float]):
231     INITIALIZE_CC_SIMS(CC)
232     CC.SIMULATE()
233     CC.plot()
234
235     print(f"Fixation rate: {sum(CC.EXTINCTIONS)/CC.NUMBER_OF_SIMULATIONS}")
236     print(f"Expected fixation rate (Moran): {CC.get_analytical_fixation_prob()}")
237     timecheck(times)
238
239 def timecheck(times : list[float]):
240     now = time.time()
241     print(f"TIMECHECK: {now-times[-1]:.2f}")
242     times.append(now)
243     return
244
245 if __name__ == '__main__':
246     time1 = time.time()
247     times = [time1]
248     CC = CARRYING_CAPACITY()
249
250     # SIMULATE_AND_PLOT(CC, times)
251
252     PROPORTION_PLOTS(CC, times)
253
254     time2 = time.time()
255     print(f"TOTAL TIME:{time2-time1:.2f}")
```

Kodfil 3: Kod för bärandekapaciteten (kräver även filen med obegränsade modellen för att kunna köras).