Projekt 2: Epidemimodellering

Numeriska metoder och Simulering

Tuva Björnberg, Nora Reneland, Matilda Stenbaek
Grupp 2

Oktober 2024

Contents

1 Metod

I denna rapport presenteras flera olika modeller som kan användas för att modellera förloppet av en epidemi. Alla dessa simulerades både deterministiskt och stokastiskt för att undersöka vilken inverkan slumpen har på resultaten.

1.1 SIR-modellen

SIR-modellen är den grundläggande modellen för smittspridning och består av tre olika tillstånd; mottaglig (S), infekterad (I) och Immun (R). Denna modell kan beskrivas med följande system av ODE:er:

$$\begin{aligned}\frac{dS(t)}{dt} &= -\beta\frac{I(t)}{N}S(t)\\ \frac{dI(t)}{dt} &= \beta\frac{I(t)}{N}S(t) - \gamma I(t)\\ \frac{dR(t)}{dt} &= \gamma I(t)\end{aligned}$$

där N är storleken på populationen, β är andelen mottagliga som blir exponerade för smitta per tidsenhet och γ är andelen sjuka som tillfrisknar per tidsenhet.

Modellen kan även göras stokastisk enligt följande:

$$\begin{aligned}S &\xrightarrow{\beta\frac{I}{N}S} I\\ I &\xrightarrow{\gamma I} R\end{aligned}$$

Modellen illustreras även grafiskt i Figur 1.

Figure 1: SIR-modell

1.2 SEIR-modellen

SEIR-modellen är en vidareutveckling av SIR-modellen som även tar hänsyn till inkubationstiden. Detta görs genom att lägga till ett nytt tillstånd, exponerad (E). Den matematiska modellen blir då:

$$\begin{aligned}\frac{dS(t)}{dt} &= -\beta\frac{I(t)}{N}S(t)\\ \frac{dE(t)}{dt} &= \beta\frac{I(t)}{N}S(t) - \alpha E(t)\\ \frac{dI(t)}{dt} &= \alpha E(t) - \gamma I(t)\\ \frac{dR(t)}{dt} &= \gamma I(t)\end{aligned}$$

där α^{-1} är inkubationstiden.

Den motsvarande stokastiska modellen är följande:

$$S \xrightarrow{\beta \frac{I}{N} S} E$$
$$E \xrightarrow{\alpha E} I$$
$$I \xrightarrow{\gamma I} R$$

Modellen illustreras även grafiskt i Figur 2.

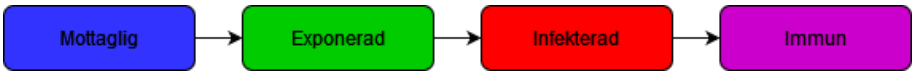

Figure 2: SEIR-modell

1.3 SEIRD-modellen

Om vi antar att sjukdomen har en dödlighet kan vi vidareutveckla de tidigare modellerna ytterligare genom att lägga till tillståndet D för antalet döda. Den matematiska modellen utvecklas då till:

$$\begin{aligned}
\frac{dS(t)}{dt} &= -\beta \frac{I(t)}{N} S(t) \\
\frac{dE(t)}{dt} &= \beta \frac{I(t)}{N} S(t) - \alpha E(t) \\
\frac{dI(t)}{dt} &= \alpha E(t) - \gamma I(t) - \mu I(t) \\
\frac{dR(t)}{dt} &= \gamma I(t) \\
\frac{dD(t)}{dt} &= \mu I(t)
\end{aligned}$$

där μ är dödligheten per tidsenhet.

Den motsvarande stokastiska modellen är följande:

$$S \xrightarrow{\beta \frac{I}{N} S} E$$
$$E \xrightarrow{\alpha E} I$$
$$I \xrightarrow{\gamma I} R$$
$$I \xrightarrow{\mu I} D$$

Modellen illustreras även grafiskt i Figur 3

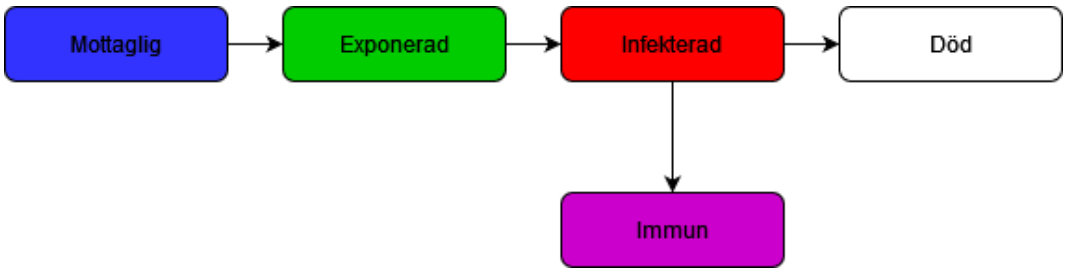

Figure 3: SEIRD-modell

1.4 Vaccinationsmodellen

Om vi även antar att vi kan tillverka ett vaccin mot sjukdomen, kan modellen utökas med tillståndet vaccinerad (V). Om vi antar att vaccinationstakten vax_rate är konstant blir modellen således:

$$\begin{aligned}
\frac{dS(t)}{dt} &= -\beta\frac{I(t)}{N}S(t) - vax_rate \\
\frac{dE(t)}{dt} &= \beta\frac{I(t)}{N}S(t) - \alpha E(t) \\
\frac{dI(t)}{dt} &= \alpha E(t) - \gamma I(t) - \mu I(t) \\
\frac{dR(t)}{dt} &= \gamma I(t) \\
\frac{dD(t)}{dt} &= \mu I(t) \\
\frac{dV(t)}{dt} &= vax_rate
\end{aligned}$$

Den motsvarande stokastiska modellen är följande:

$$\begin{aligned}
S &\xrightarrow{\beta\frac{I}{N}S} E \\
E &\xrightarrow{\alpha E} I \\
I &\xrightarrow{\gamma I} R \\
I &\xrightarrow{\mu I} D \\
S &\xrightarrow{vax_rate} V
\end{aligned}$$

Modellen illustreras även grafiskt i Figur 4.

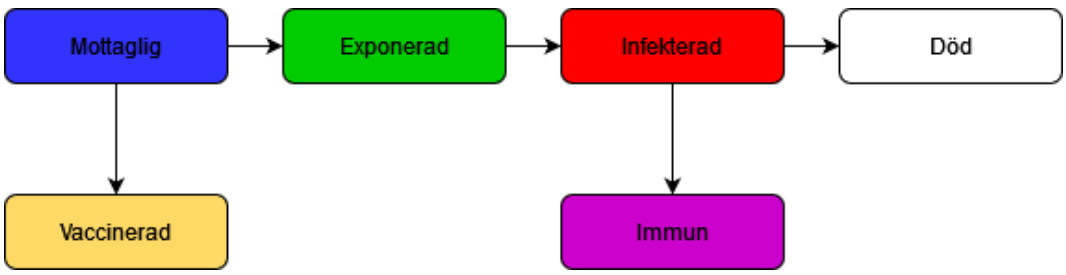

Figure 4: Vaccinationsmodellen

1.5 Egen vaccinationsmodell

I den egna vaccinationsmodellen har ett tillstånd lagts till: permanent immun (Im). En person kan bli permanent immun antingen genom att återhämta sig från en infektion och på så sätt utveckla ett eget försvar, eller genom att vaccineras med två doser vaccin. Endast en dos vaccin räcker inte eftersom vaccinet i den här modellen inte nödvändigtvis har heltäckande skydd. Om skyddet, ϵ, exempelvis är 0.94, innebär det att 6% av de som vaccineras med bara en dos tappar skyddet och blir mottagliga (S) igen. Av de som vaccineras med en andra dos antas däremot alla permanent immuna. Likt den första dosen antas vaccinationstakten för den andra dosen, $second\_dose\_rate$, vara konstant.

Andelen personer som blir permanent immuna av att återhämta sig från infektionen bestäms av parametern ζ. Förutom att bli permanent immun (Im) kan en person som återhämtat sig från en infektion (R) bli mottaglig (S) och därmed riskera att bli sjuk igen om de inte vaccinerar sig.

Sammanfattat blir den matematiska modellen:

$$\begin{aligned}
\frac{dS(t)}{dt} &= -\beta\frac{I(t)}{N} \times S(t) - vax_rate + \delta R(t) + (1-\epsilon)V(t) \\
\frac{dE(t)}{dt} &= \beta\frac{I(t)}{N} \times S(t) - \alpha E(t) \\
\frac{dI(t)}{dt} &= \alpha E(t) - \gamma I(t) - \mu I(t) \\
\frac{dR(t)}{dt} &= \gamma I(t) - \delta R(t) - \zeta R(t) \\
\frac{dD(t)}{dt} &= \mu I(t) \\
\frac{dV(t)}{dt} &= vax_rate - (1-\epsilon)V(t) - min(second_dose_rate, V(t)) \\
\frac{dIm(t)}{dt} &= min(second_dose_rate, V(t)) + \zeta R(t)
\end{aligned}$$

Den motsvarande stokastiska modellen är följande:

$$\begin{aligned}
S &\xrightarrow{\beta\frac{I}{N}S} E \\
E &\xrightarrow{\alpha E} I \\
I &\xrightarrow{\gamma I} R \\
I &\xrightarrow{\mu I} D \\
S &\xrightarrow{vax_rate} V \\
R &\xrightarrow{\delta R} S \\
V &\xrightarrow{(1-\epsilon)V} S \\
V &\xrightarrow{min(second_dose_rate, V)} Im \\
R &\xrightarrow{\zeta R} Im
\end{aligned}$$

Modellen illustreras även grafiskt i Figur 5.

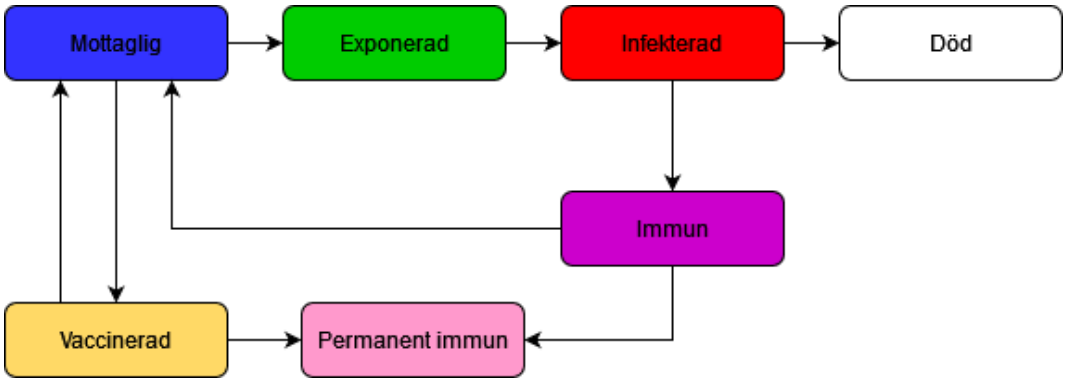

Figure 5: Egen epidemimodell

1.6 Lösningsalgoritmerna

För att simulera samtliga av ovanstående modeller användes Gillespies algoritm som stokastisk simuleringsmetod, samt ODE-lösaren solve_ivp för att genomföra en deterministisk simulering. Resultaten från dessa båda simuleringar kunde sedan jämföras och analyseras för att bl.a. dra slutsatser kring vilken påverkan slumpen har på de olika modellerna, samt i vilken utsträckning slumpen kan påverka resultatet av simuleringen.

Gillespies algoritm är en stokastisk simuleringsalgoritm som används för att simulera diskreta Markovprocesser, exempelvis modellerna som beskrivits ovan. Indatan till algoritmen består av en propensitetsvektor, P, som definierar sannolikheterna för de olika övergångarna, en stökiometrivektor, N, som beskriver de möjliga övergångarna mellan tillstånden, samt en begynnelsevektor, Y_0, som beskriver det initiala tillståndet. Utöver detta krävs även en tidsvektor som definierar tidsspannet samt parametrarna som används i modellen.

Gillespies algoritm har följande struktur:

1. Först samplas ett tidssteg, τ eller Δt, från en exponetialfördelning. Detta är tiden till att nästa reaktion sker.
2. Nästa reaktion, r, samplas sedan från en annan sannolikhetsfördelning, denna given av propensitetsvektorn, P.
3. Slutligen måste tillståndsvektorn uppdateras så att den reflekterar den senaste reaktionen, $Y = Y+N_r$, samt tiden uppdateras, $t = t + \Delta t$.

Efter detta kommer antingen ovanstående steg upprepas igen, om t < t_{end}, alternativt avslutas simuleringen, om sluttiden är nådd.

2 Resultat

Resultaten påverkas ständigt av de parametrar som införs i modellerna, både gällande initiala värden och förändringsfaktorer. Modellerna får olika konsekvenser beroende på den indata som används samt slumpen.

De värden som använts i simuleringarna är följande, om inget annat anges; β = 0.3, γ = 1/7, α = 0.5, μ = 0.01, vax_rate = 3, $second\_dose\_rate$ = 5, ϵ = 0.94, ζ = 0.3. Det initiala tillståndet för samtliga simuleringar är, om inget annat anges; N (total population) = 1000, S(0) = 995, I(0) = 5. Alla övriga tillstånd är 0.

2.1 Initialvärden

Det initiala tillståndet påverkar epidemins förlopp samt det slutgiltiga resultatet av simuleringen. Förändring av olika initiala värden kan ge olika effekt och exempelvis påverka totala antalet infekterade eller döda. Det är därför viktigt att beakta olika möjliga initiala tillstånd för att få en uppfattning om vad dessa kan ha för påverkan på simuleringen.

2.1.1 Infekterade

Antalet infekterade vid simuleringens start kommer att påverka modellens infektionsförlopp. Om det initiala värdet för infekterade ökar kommer modellen att visa högre infektionssiffor till en början, men även fler som tillfrisknar och blir immuna. För den egna modellen blir allt fler permanent immuna, medan de övriga modellerna visar att fler blir immuna, vilket illustreras i jämförelse mellan Figur 6, referensen, och efter justeringen i Figur 7 för den egna epidemimodellen respektive Figur 8 och 9 för vaccinationsmodellen.

Liknande kommer att ske för andra tillstånd, exempelvis exponerad, eftersom spridningen av sjukdomen intensifieras. Då alla tillstånd är beroende av varandra trappas förloppen upp till högre siffror under kortare tidsperioder. Det upplevs också att tiden befolkningen blir infekterad förkortas, genom att kurvan förskjuts till vänster i t-axeln.

Trots att denna parameter inte nödvändigtvis är möjlig att påverka rent praktiskt kan det vara intressant att simulera olika värden då man kan vilja ha en modell som undersöker framtiden av en situation baserat på antalet insjuknade nu. Ett exempel på sådan situation är olika typer av sjukdomsutbrott på specifika platser.

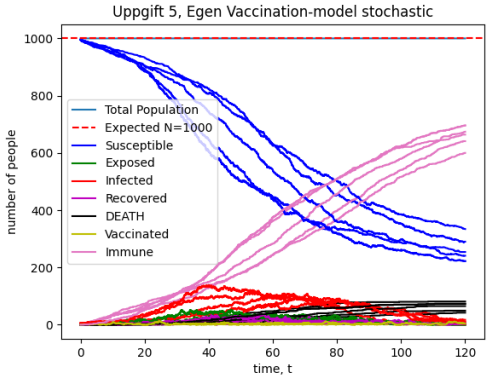

Figure 6: Egen epidemimodell, 5 initialt infekterade

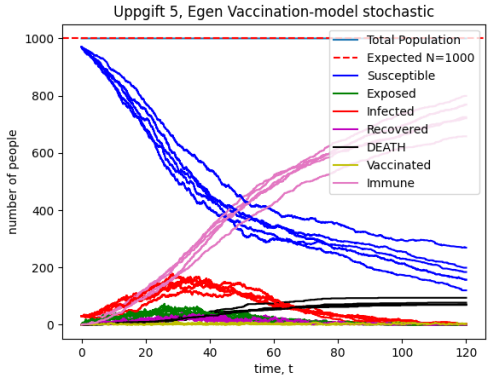

Figure 7: Egen epidemimodell, 30 initialt infekterade

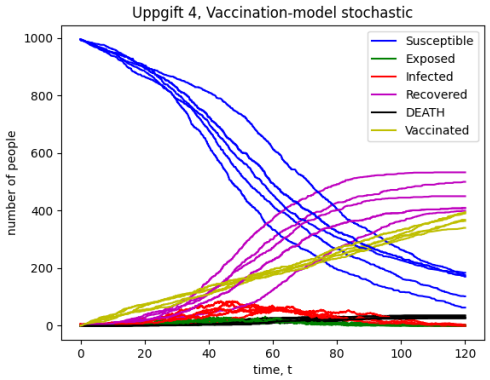

Figure 8: Vaccinationsmodellen, 5 initialt infekterade

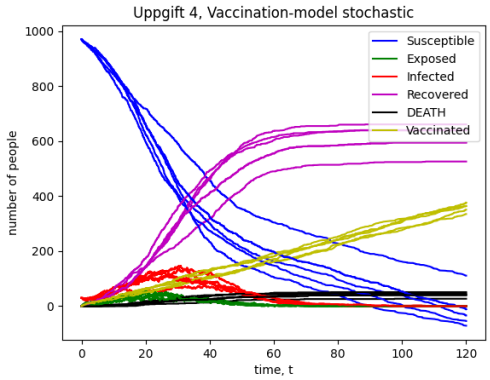

Figure 9: Vaccinationsmodellen med 30 initialt infekterade

2.1.2 Exponerade

Liknande en förändring av initialvärdet för antalet infekterade kommer en förändring av det initiala värdet för exponerade också att påverka händelseförloppet av modellen. Vid en ökning av antalet exponerade vid simuleringens start sker insjuknandet/infekteringen tidigare, men likaså tillfrisknandet. Fler är infekterade samtidigt men modellen verkar återhämta sig tidigare, vilket kan ses i Figur 11 för den egna modellen och i Figur 13 för SEIR-modellen, jämfört med 0 exponerade i Figur 10 respektive Figur 12.

Förändring av denna parameter är, precis som antalet initialt infekterade, inte riktigt möjlig att påverka i praktiken, men av samma anledning som för infekterade kan det vara intressant att simulera olika värden

för att undersöka en händelse där en grupp människor blivit exponerade men inte insjuknat än.

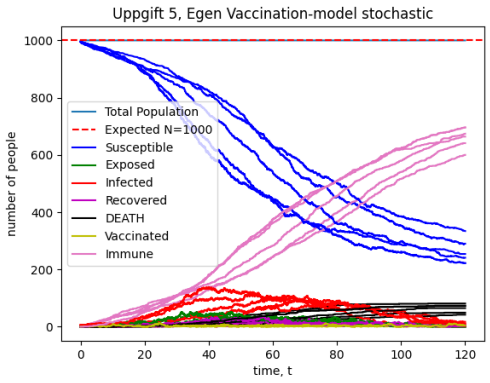

Figure 10: Egen epidemimodell, 0 initialt exponerade

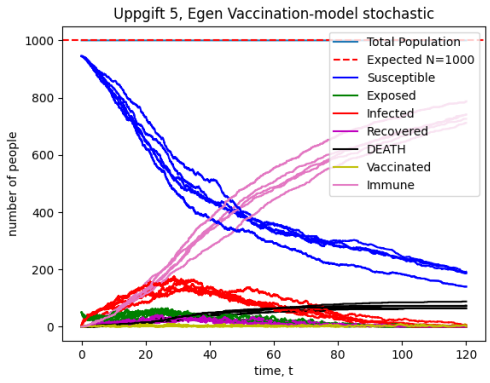

Figure 11: Egen epidemimodell, 30 initialt exponerade

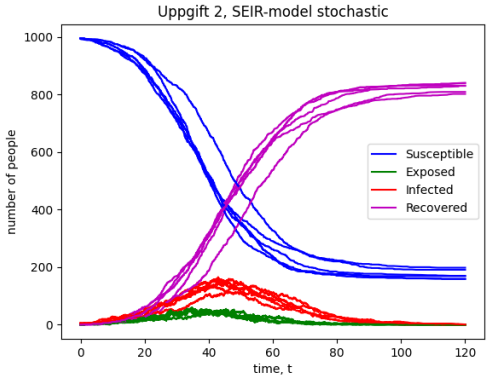

Figure 12: SEIR-modellen, 0 initialt exponerade

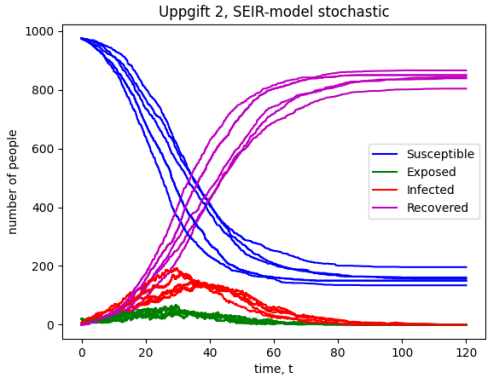

Figure 13: SEIR-modellen, 20 initialt exponerade

2.1.3 Immuna

Det är inte lika relevant för modellerna att kunna justera det initiala antalet immuna. Antalet immuna är inte ett ingångsvärde, på så sätt som antalet infekterade är, utan det är beroende av flera steg in i modellen.

För den egengjorda modellen förskjuts antalet infekterade, huvudsakligen på grund av att en del av de immuna har en viss sannolikhet att bli mottagliga igen, vilket illustreras i Figur 15 jämfört med Figur 14 för den egna modellen. I och med att inte alla immuna blir mottagliga igen fås lägre antal infekterade i totalen.

För den ursprungliga vaccinationsmodellen däremot är det inte större skillnader förutom att fler blir immuna, vilket visas i jämförelse mellan Figur 17 och Figur 16.

Förändringen i initialt immuna skulle möjligen kunna nyttjas i ett sammanhang där man vill undersöka en simulering efter ett epidemiutbrott och hur infekterade kan påverkas.

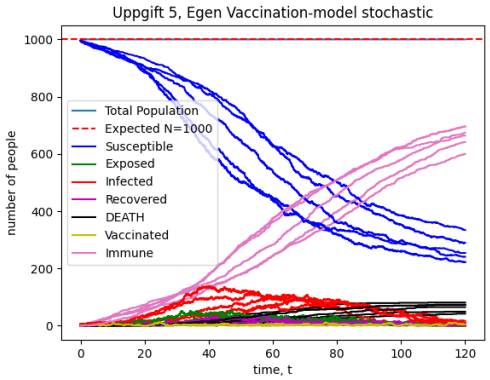

Figure 14: Egen epidemimodell, 0 initialt immuna (recovered)

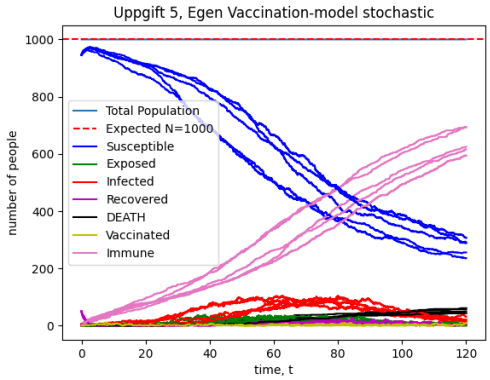

Figure 15: Egen epidemimodell, 50 initialt immuna (recovered)

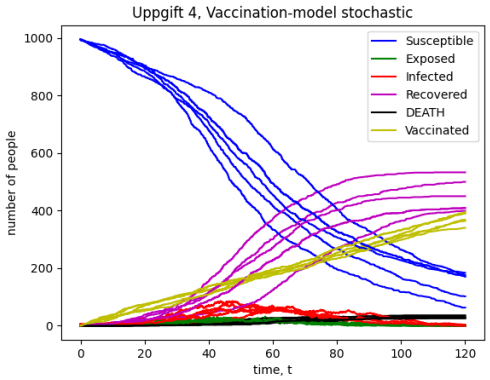

Figure 16: Vaccinationsmodellen, 0 initialt immuna (recovered)

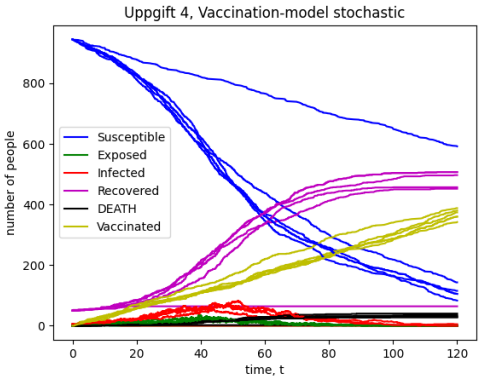

Figure 17: Vaccinationsmodellen, 50 initialt immuna (recovered)

2.1.4 Döda

Att förändra ingångsvärdet för antalet döda personer från 0 för dessa modeller uppfyller inte det syfte och resultat som förväntas av modellerna. Antalet döda personer förändrar inte infektionsförloppen mer än att färre totala personer kan räknas in för de andra tillstånden. Den totala dödssiffran kan då bli överdimensionerad.

Om de initialt döda i modellen är på grund av den epidemi man undersöker är det mer relevant att hela sjukdomsförloppet för dessa personer ingår, och inte bara dödstillståndet. Detta gäller både den egna respektive den givna vaccinationsmodellen och illustreras i jämförelse i Figur 19 och Figur 18 respektive Figur 21 och Figur 20.

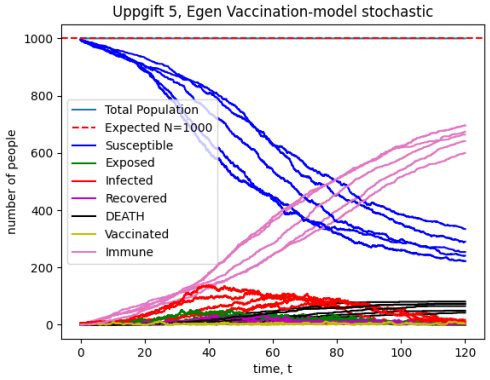

Figure 18: Egen epidemimodell, 0 initialt döda

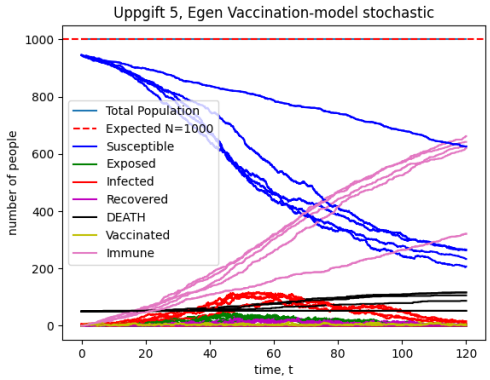

Figure 19: Egen epidemimodell, 50 initialt döda

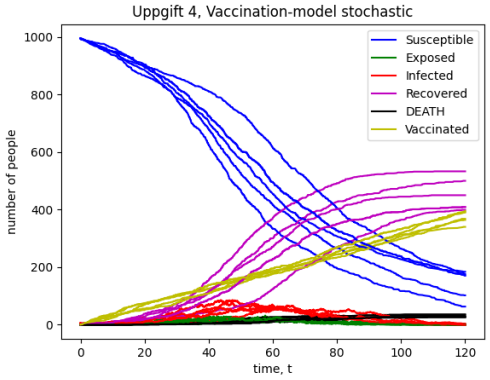

Figure 20: Vaccinationsmodellen, 0 initialt döda

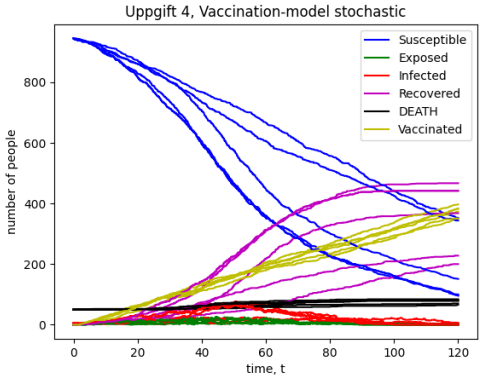

Figure 21: Vaccinationsmodellen, 50 initialt döda

2.1.5 Vaccinerade

Att justera det initiala värdet för antalet vaccinerade av sjukdomen vid simuleringens start kan också ge en missvisande bild av verkligheten, precis som med antalet döda personer. Vaccineringen är ett tillstånd som är beroende av flera andra tillstånd och deras förlopp.

Genom att öka antalet vaccinerade personer vid start fås ett resultat där framför allt infektionsförloppet kommer senare för den egna modellen, detta visas i jämförelse mellan Figur 23 och Figur 22 för den egna modellen. Detta sker på grund av att personer vaccinerade med en dos fortfarande kan bli mottagliga för infektion, istället för att bli permanent immuna.

Den givna vaccinationsmodellen förblir någorlunda lik den ursprungliga simuleringen förutom att antalet vaccinationer förskjuts med detta initialvärde, det vill säga 50 personer fler än om det hade startat med noll. Detta visas i jämförelsen mellan Figur 25 och Figur 24.

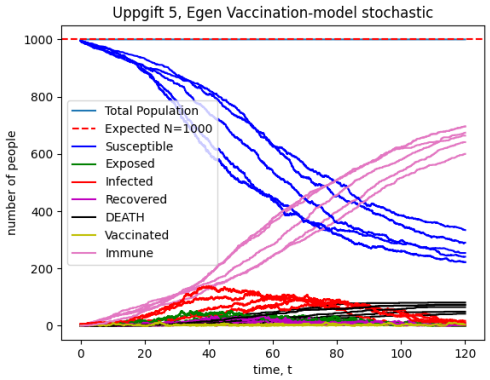

Figure 22: Egen epidemimodell, 0 initialt vaccinerade

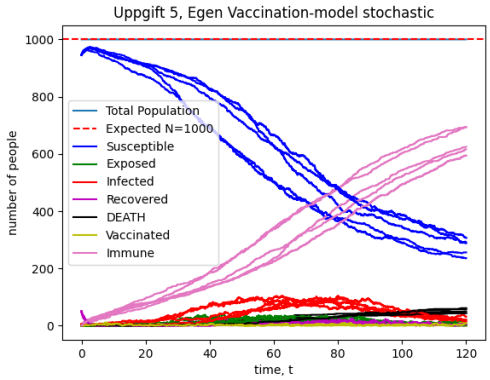

Figure 23: Egen epidemimodell, 50 initialt vaccinerade

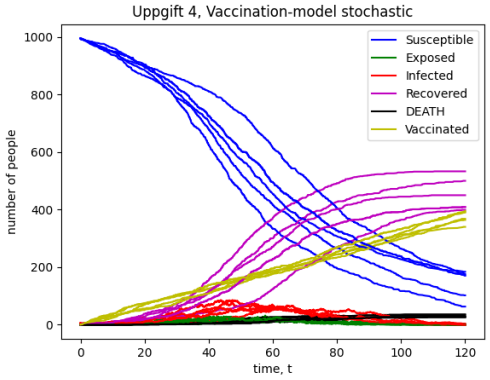

Figure 24: Vaccinationsmodellen, 0 initialt vaccinerade

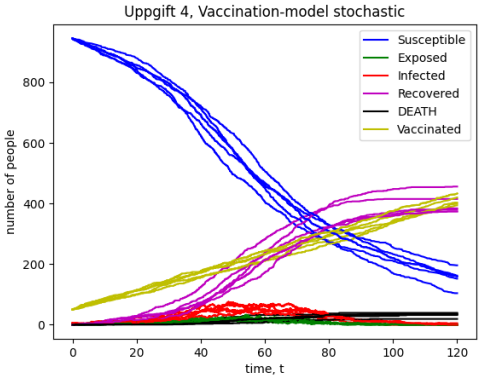

Figure 25: Vaccinationsmodellen, 50 initialt vaccinerade

2.1.6 Permanent immuna

Eftersom den egna epidemimodellen har ytterligare ett tillstånd, permanent immunitet, finns det ett sista initialvärde som kan påverkas.

Genom att förändra ingångsvärdet för antal permanenta immuna till fler än noll medförs samma problematik som med det initiala värdet för antalet döda. Det handlar om personer som inte går igenom simuleringens steg. Den permanenta immuniteten är ett sluttillstånd, precis som döden, vilket gör att kurvan huvudsakligen förskjuts och påverkar inte de andra tillstånden mer än att de 50 personerna inte går igenom modellens tillstånd. Detta kan jämföras mellan Figur 27 och Figur 26.

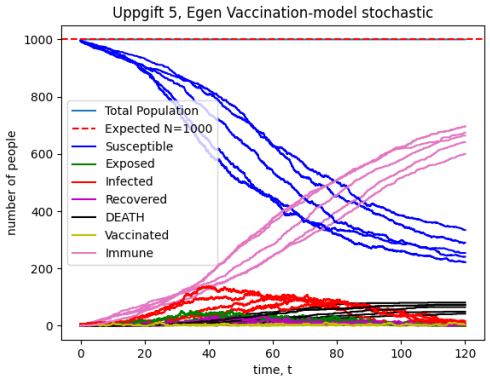

Figure 26: Egen epidemimodell, 0 initialt permanent immuna (immune)

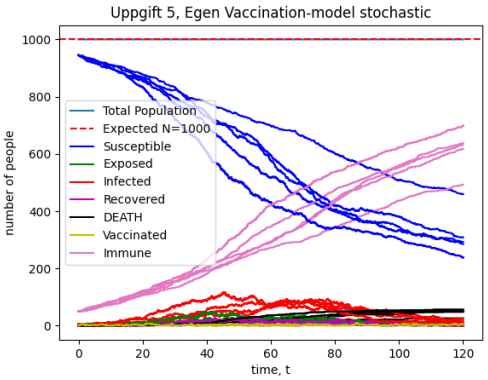

Figure 27: Egen epidemimodell, 50 initialt permanent immuna (immune)

2.2 Förändningsfaktorer

Förändingsfaktorerna; smittspridningshastigheten β, tillfriskningstiden γ, inkubationstiden α^{-1}, dödligheten μ, vaccinationstakten för första och andra dosen, vax_rate och $second\_dose\_rate$, vaccinets skyddsfaktor ϵ, samt sannolikheten för fullständig immunitet efter sjukdom ζ, påverkar simuleringarna på olika sätt där vissa är mer avgörande för modellerna än andra. I praktiken kommer endast vissa av dessa gå att påverka, exempelvis β genom social distansering och isolation, eller vax_rate genom ökade resurser till vaccinering, medan andra kommer bero helt på sjukdomen och därför inte går att påverka, som exempelvis inkubationstiden α och permanent immunitet efter sjukdom ζ.

2.2.1 β, smittspridningshastigheten

Då β är faktorn som avgör andelen som övergår från mottagliga till exponerade, smittspridningshastigheten, skulle en förändring påverka alla simuleringens tillstånd och dess resultat. Vid höga β ökar smittspridningshastigheten och fler mottagliga blir infekterade under kortare tid. Detta leder till att toppen på epidemin blir högre och i sig att antalet döda och immuna också ökar kraftigare, detta illustreras i Figur 29. Låga β å andra sidan leder till att färre insjuknar, smittan sprids långsammare, och därmed ett mildare utbrott, vilket kan ses i Figur 28, till skillnad från Figur 29 med ett högt β.

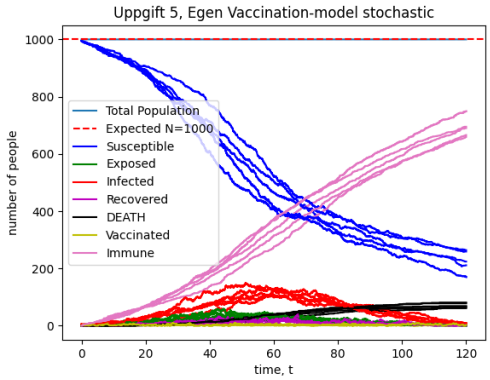

Figure 28: Egen vaccinationsmodell, $\beta = 0.3$

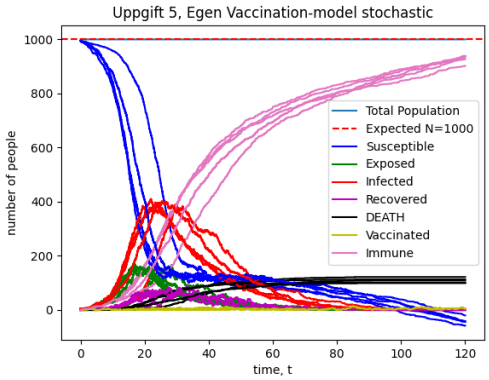

Figure 29: Egen vaccinationsmodell, $\beta = 0.7$

2.2.2 γ, tillfriskningstiden

γ är andelen av de infekterade som återhämtar sig från sjukdomen under varje tidssteg. Låga γ-värden betyder att det tar längre tid att återhämta sig från sjukdomen, färre blir friska. Detta leder till att smittan finns i populationen längre. Epidemin blir mer ihållande och fler är infekterade samtidigt, vilket kan ses i Figur 31. Höga värden å andra sidan betyder att sjukdomen är kortlivad, andelen som tillfrisknar är större, detta kan ses i Figur 30 jämfört med ett lågt γ i Figur 31.

Att tillfriskningstiden är längre innebär även att de som insjuknar utsätts för större risk att dö eftersom de är sjuka under en längre tid. Detta kommer leda till fler totalt döda då γ är lågt, vilket kan ses i skillnaden mellan Figur 30 och Figur 31.

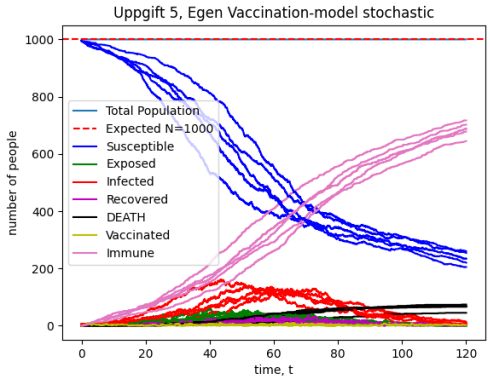

Figure 30: Egen vaccinationsmodell, γ = 1 / 7

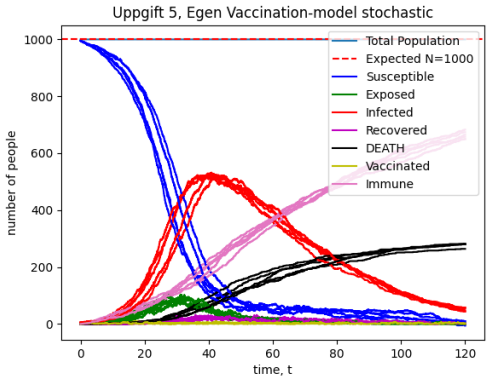

Figure 31: Egen vaccinationsmodell, γ = 1 / 30

2.2.3 α, inkubationstiden

Faktorn α kan bidra, precis som γ, till att infektionen stannar i populationen längre, då det handlar om inkubationstiden. Höga α betyder att inkubationstiden är kort och övergången från exponerad till infekterad blir kortare. Populationen får alltså fler sjuka snabbare, vilket kan ses i Figur 33, i jämförelse med Figur 32 som har ett lågt α. Låga α däremot kommer att leda till att populationen har fler individer som stannar i det exponerade tillståndet. Färre personer är infekterade samtidigt och kurvan är därmed flackare.

Ett högt α i kombination med ett lågt γ-värde kommer leda till att simuleringen får fler som är sjuka samtidigt och färre av de infekterade blir friska. Det kan alltså antas vara en svår sjukdom att utrota från populationen, vilket kan ses i extremfallet i Figur 34.

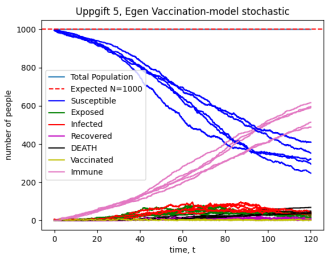

Figure 32: Egen vaccinationsmodell, α = 0.2

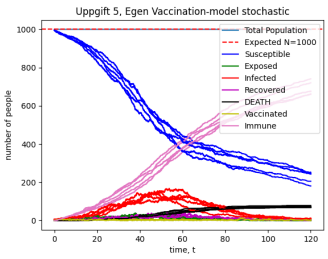

Figure 33: Egen vaccinationsmodell, α = 0.9

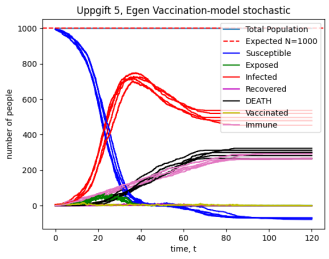

Figure 34: Egen vaccinationsmodell, α = 0.9, γ = 1 / 200

2.2.4 μ, dödligheten

μ definierar dödligheten för sjukdomen, mer specifikt andelen sjuka som dör per tidsenhet. Om μ ökar kommer en större andel sjuka att dö varje tidssteg, detta kommer i de allra flesta fall att leda till ett högre slutgiltigt dödsantal, vilket kan ses i skillnaden mellan Figur 35 och Figur 36. Detta kommer att vara fallet i ett intervall då dödligheten är hög, men inte så pass hög att alla som har sjukdomen hinner dö innan de infekterar någon, ett extremt fall som kan ses i Figur 37.

En annan effekt av hög dödlighet är att antalet immuna i större utsträckning kommer komma från vaccin, till skillnad från människor som blivit immuna efter att ha varit sjuka. Detta leder till en mer linjär utveckling av antalet immuna, Figur 36, vilket innebär att vaccinationstakten blir alltmer viktig för att uppnå flockimmunitet då dödligheten ökar.

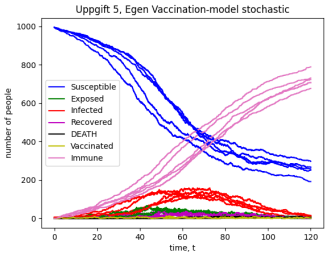

Figure 35: Egen vaccinations-modell, μ = 0.001

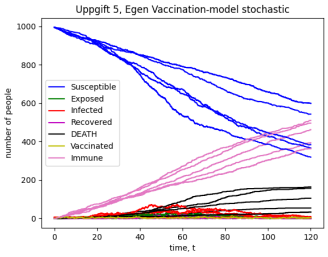

Figure 36: Egen vaccinations-modell, μ = 0.05

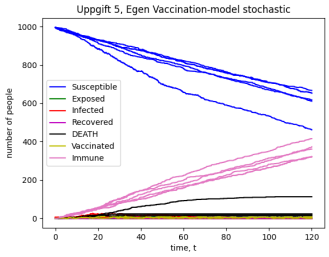

Figure 37: Egen vaccinations-modell, μ = 0.1

2.2.5 vax_rate och second_dose_rate

En förändring i vax_rate påverkar de givna modellerna genom att man utrotar sjukdomen snabbare, då en större andel av populationen går från mottaglig till vaccinerad. Då den är konstant påverkas den inte av några andra tillstånd och dess faktorer.

För den egna epidemimodellen kommer en ökning av vax_rate leda till att sjukdomen utrotas snabbare än om den vore lägre, men inte i samma takt som för de övriga modellerna. Eftersom en andel individer kan gå från att vara vaccinerad tillbaka till mottaglig, och därefter eventuellt infekterad igen. Modellen är dock begränsad till andelen som blir absolut immuna per tidssteg, $second\_dose\_rate$.

En förändring av $second\_dose\_rate$ kommer att ge liknande resultat som för vax_rate.

2.2.6 ϵ, vaccinets skyddsfaktor

ϵ definierar hur effektivt vaccinets skydd är. Ju effektivare skyddet är, desto kortare kommer epidemins förlopp att bli, detta kan ses i skillnaden mellan Figur 38 och Figur 39. Detta är fallet eftersom fler personer blir immuna reden efter en dos, och färre därför behöver vaccinera sig flera gånger för att uppnå immunitet. Detta innebär att färre blir smittade, speciellt i ett senare skede av epidemin då många har fått minst en dos vaccin.

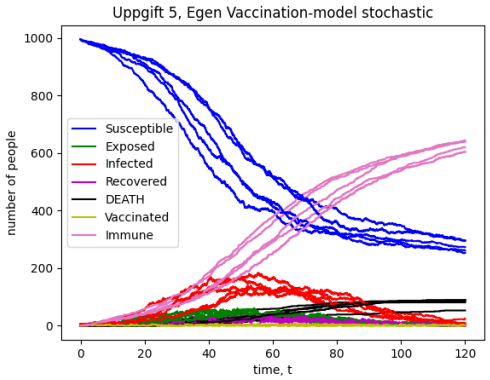

Figure 38: Egen epidemimodell, $\epsilon = 0.3$ (låg)

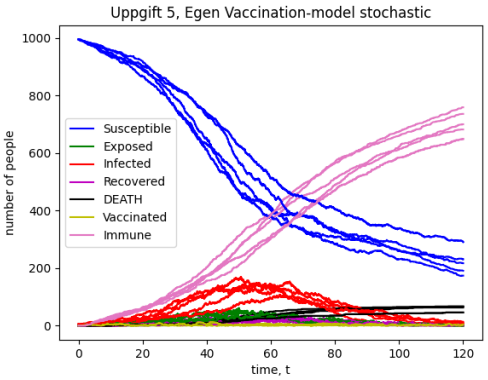

Figure 39: Egen epidemimodell, $\epsilon = 0.95$ (hög)

2.2.7 ζ, fullständig immunitet efter sjukdom

ζ är graden av människor som bara av att tillfriskna från sjukdomen även utvecklar fullständig immunitet. Ju fler som utvecklar immunitet efter sjukdom, desto färre kommer drabbas av sjukdomen flera gången, och på sätt även minska spridningen. Detta innebär att den totala mängden av sjukdomsfall minskar då ζ ökar, vilket kan ses i Figurerna 40, 41 och 42.

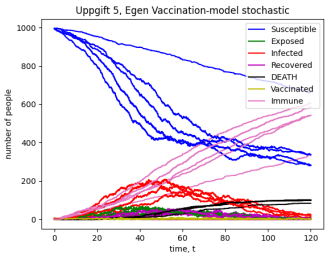

Figure 40: Egen vaccinationsmodell, $\zeta = 0.1$

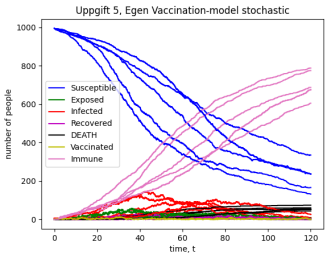

Figure 41: Egen vaccinationsmodell, $\zeta = 0.5$

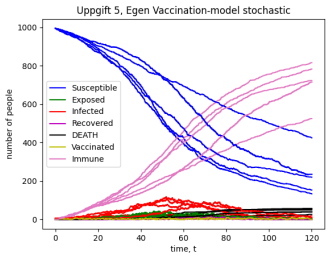

Figure 42: Egen vaccinationsmodell, $\zeta = 0.95$

2.3 Slumpen

Slumpen påverkar tidsstegen, alltså hur snabbt sjukdomen sprider sig, och vilket tillstånd som är härnäst.

Om slumpen skulle ge bästa scenariot skulle sjukdomen sprida sig långsammare och att färre individer infekteras tidigt, vilket leder till en mildare epidemi där färre personer blir smittade totalt sett. Detta betyder att vaccination och immunitet hinner ske innan sjukdomen har spridit sig för mycket i populationen.

Sämsta scenariot å andra sidan sker när slumpen gör att smittan sprider sig snabbt i början, så att många människor exponeras samtidigt. Detta kan leda till ett snabbt uppsving i antalet infekterade och ökat antal döda. Detta blir extra tydligt om vaccineringen inte hinner med spridningstakten och därmed inte kan begränsa smittan.

3 Argumentation

En matematisk modell kan inte helt återspegla vad som händer och kommer att hända i verkligheten. Den kan dock ge en fingervisning och vara ett användbart verktyg för att testa hur olika parametrar kan

påverka en situation och händelseförlopp. Vad skulle till exempel hända om ingen vaccinerar sig under en epidemi? Detta kan testas med simuleringar av modellen utan att faktiskt sätta människoliv på spel. Att testa olika strategier gör det möjligt att ta mer välgrundade beslut när en verklig situation dyker upp.

Modellerna som beskriver förhållandet mellan de olika tillstånden, som en person kan befinna sig i under en epidemi, utvecklas under detta arbete stegvis. Tillstånd och övergångar mellan tillstånd läggs till och gör modellerna mer komplexa, men också mer verklighetstrogna. I verkligheten är det många parametrar som verkar med varandra och därför är det rimligt att fler parametrar i en modell gör den mer lik verkligheten. Den mest utvecklade modellen i denna rapport, den egna vaccinationsmodellen, tar hänsyn till flera olika faktorer och går därför att anpassa för att simulera olika epidemier genom att ändra värdena på parametrarna.

I de olika modellerna finns en del faktorer som inte tas hänsyn till, eller som inte helt stämmer överens med verkligheten. I alla modeller utom den sista tas ingen hänsyn till att immuniteten som uppstår efter att man varit sjuk kanske inte är fullständig, alternativt avtar efter en viss tid. Detta kan göra att dessa modeller kan skilja sig från verkligheten på ett viktigt sätt, nämligen att de inte beaktar möjligheten att en person som har varit sjuk kan bli sjuk igen.

Ingen av modellerna tar heller i beaktning möjligheten att vara smittbärare utan att någonsin uppvisa symtom själv. Samtliga modeller antar att en person som är immun, antingen till följd av sjukdom eller vaccin, inte heller kan sprida smittan vidare. Detta kommer inte alltid att stämma överens med verkligheten. Exempelvis Covid-19 hade många asymtomatiska spridare [3], alltså infekterade personer som kunde sprida smittan, men fick inga, eller mycket milda, symtom själva.

Ingen av de undersökta metoderna beaktar heller möjligheten att det uppkommer mutationer. Vid uppkomst av nya mutationer kan människor som ansågs immuna möjligtvis bli sjuka av den nya mutationen. Olika mutationer av samma virus kan även vara olika smittsamma, samt ha olika dödlighet, inkubations- och tillfriskningstid [2], vilket kan göra det svårt eller missvisande att simulera olika mutationer med en enda modell.

I vaccinationsmodellen antas att vaccinationstakten är konstant. I verkligheten är detta mycket osannolikt. Vaccinationstakten kan bl.a. variera med avseende på sjukvårdens kapacitet att vaccinera människor, men även på människors vilja att bli vaccinerade. I tidigare presenterade modeller antogs att alla som får chansen skulle vaccinera sig, men detta stämmer inte med hur det ser ut i verkligheten [1]. Här skulle man behöva inkludera en parameter för hur troligt det är att en godtycklig person accepterar vaccinet. Viljan att vaccinera sig skulle även kunna tänkas bero på exempelvis ålder, vilket också skulle påverka resultatet. Även parametrar såsom dödlighet och smittspridning kan variera mellan olika åldersgrupper och på så sätt påverka smittspridningen i stort.

Om man vill utveckla smittspridningsmodellen ytterligare kan man även ta i beaktning vikten av social distansering. Ingen av modellerna tidigare presenterade tar hänsyn till detta utan utgår från en konstant smittspridningsfaktor. Denna faktor kan med stor sannolikhet förändras under epidemins gång till följd av social distansering. Graden av social distansering kan också tänkas påverkas av andra faktorer relaterade till epidemin, såsom dödligheten eller det totala antalet sjuka, samt förändras över tid.

4 Diskussion

En stor fördel av att genomföra en studie av en epidemi och dess spridning är att samhället kan få en bild av den nuvarande situationen och situationen framöver. Om det finns simuleringar av hur epidemin kommer utveckla sig kan dessa tas i beaktning för att planera hanteringen av den. Hur mycket resurser behöver sjukvården? Vilka försiktighetsåtgärder bör tas för att skydda de som är del av en riskgrupp? Hur effektiv är vaccineringen? Dessa frågor kan eventuellt besvaras och därmed leda till att smittspridningen och dödligheten av epidemin kan hållas ned. Dock gäller det att modellen ger en representativ och korrekt

bild av verkligheten. Som tidigare diskuterats har många av parametrarna som används i modellerna mycket stor påverkan på resultatet. Men eftersom verkligheten är mer komplex än vad som framgår i modellerna kommer parametrarnas värden till stor del vara uppskattningar eller gissningar. De parametrar som påverkar resultatet måste därför väljas med noggrannhet och vara välgrundade för att resultatet inte ska ge en missvisande bild.

Att presentera resultatet från en studie av den här karaktären kan ha också samhälleliga effekter. Dessa kommer främst från hur människor tolkar resultaten som publiceras. En epidemi kan vara något som gör många oroliga och då är det viktigt att resultaten presenteras på ett sätt som inte är missvisande. När framtiden är oviss kan människor vilja känna trygghet genom att ta tag i något som berättar vad som kommer hända. En matematisk modell kan vara ett sådant exempel och kan lätt tolkas som objektiv sanning. Trygghet i den här formen, att få en bild av vad som händer och kommer hända, kan vara något mycket positivt som gör det lättare för människor att härda ut, men tryggheten kan också vara falsk. Om resultaten av modellen visar en oproportionerligt positiv bild av verkligheten kan människor bli vårdslösa och därmed riskerar epidemin att bli svårare och längre. Om bilden som målas upp däremot är för negativ kan mer oro sprida sig istället. Problematiken finns i balansgången av att upplysa människor om ett komplext ämne med varierande och osäkra faktorer på ett sätt som varken är missvisande, alltför förenklat eller för svårt för att ta till sig.

Som exempel på detta kan man titta på dödligheten i modellen. Ett problem med modellerna i detta arbete är att de behandlar alla människor som en homogen grupp, när människor egentligen har mycket olika förutsättningar när det kommer till att bli infekterad, immun eller att dö av infektionen. Just dödligheten kan skilja sig mycket mellan olika grupper baserat på till exempel ålder eller hälsotillstånd. Om detta inte tas hänsyn till i modellen eller framgår i det presenterade resultatet kan det leda till att människor som borde vara försiktiga tar onödiga risker och därmed riskerar att smittas och eventuellt dö. Det kan också leda till att människor som egentligen inte löper någon stor risk att dö av infektionen blir uppskrämda och lever med mer rädsla och försiktighet än vad som egentligen är nödvändigt.

Att publicera resultaten av en sådan här studie kan ha andra effekter också. En önskad effekt är att fler får förståelse för den positiva effekt som vaccination ger på en stor skala. Om det framgår i resultatet att vaccination är en stor bidragande faktor till att förhindra spridningen kan fler bli motiverade till att vaccinera sig. Som diskuterats tidigare är det inte alla som väljer att vaccinera sig och dessa människor skulle eventuellt kunna påverkas av resultatet av studien.

References

[1] Folkhälsomyndigheten. *Undersökning om acceptans för vaccination mot covid-19 – Resultat april–maj 2021*. URL: https://www.folkhalsomyndigheten.se/folkhalsorapportering-statistik/statistikdatabaser-och-visualisering/vaccinationsstatistik/statistik-for-vaccination-mot-covid-19/acceptans-for-vaccination-mot-covid-19/undersokning-om-acceptans-for-vaccination-mot-covid-19--resultat-aprilmaj-2021/#:~:text=H%C3%B6g%20vaccinationsvilja,de%20erbjuds%20att%20g%C3%B6ra%20det. (visited on 05/21/2021).

[2] Folkhälsomyndigheten. *Varianter av viruset som orsakar covid-19*. URL: https://www.folkhalsomyndigheten.se/smittskydd-beredskap/smittsamma-sjukdomar/covid-19/varianter-av-viruset-som-orsakar-covid-19/ (visited on 04/29/2024).

[3] Ada's Medical Knowledge Team. *Asymptomatic COVID-19*. URL: https://ada.com/covid/asymptomatic-covid-19/ (visited on 06/10/2024).

A Koden

A.1 SIR-modell

```
import numpy as np
import scipy
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
import gillespie as gill

beta = 0.3
gamma = 1 / 7

N = 1000
infected = 5
recovered = 0

initial_sum = infected + recovered
initial = [N - initial_sum, infected, recovered]

t0 = 0
t1 = 120
t_span = [t0, t1]


# ---------------- ASSIGNEMNT 1 ------ SIR MODEL ---------------------
# ODE solver, determenistic
def ODE_SIR(t, y):
    s, i, r = y

    s_p = -(beta) * (i / N) * s
    i_p = beta * (i / N) * s - (gamma * i)
    r_p = gamma * i

    return np.array([s_p, i_p, r_p])


t_eval = np.arange(t_span[0], t_span[1], 0.1)

sol = solve_ivp(ODE_SIR, t_span, initial, t_eval=t_eval)

plt.plot(sol.t, sol.y[0], label="Susceptible")
plt.plot(sol.t, sol.y[1], label="Infected")
plt.plot(sol.t, sol.y[2], label="Recovered")
plt.xlabel("time, t")
plt.ylabel("number of people")
plt.title("Uppgift 1, SIR-model solve_ivp")
plt.legend()
plt.show()

# Gillespie, stochastic
initial = (N - initial_sum, infected, recovered)
coeff = (beta, gamma)


def stochEpedemic():
    M = np.array([[-1, 1, 0], [0, -1, 1]])
    return M


def propEpedemic(X, coeff):
    beta = coeff[0]
    gamma = coeff[1]

```

```
    s = X[0]
    i = X[1]
    r = X[2]

    w = np.array([beta * (i / N) * s, gamma * i])
    return w


for i in range(5):
    t, X = gill.SSA(propEpedemic, stochEpedemic, initial, t_span, coeff)

    plt.plot(t, X[:, 0], "b")
    plt.plot(t, X[:, 1], "g")
    plt.plot(t, X[:, 2], "r")

plt.plot(t, X[:, 0], "b", label="Susceptible")
plt.plot(t, X[:, 1], "g", label="Infected")
plt.plot(t, X[:, 2], "r", label="Recovered")

plt.xlabel("time, t")
plt.ylabel("number of people")
plt.title("Uppgift 1, SIR-model stochastic")
plt.legend()
plt.show()
```

A.2 SEIR-modell

```
import numpy as np
import scipy
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
import gillespie as gill

beta = 0.3
gamma = 1 / 7

N = 1000
infected = 5
recovered = 0
initial = [N - infected, infected, recovered]

t0 = 0
t1 = 120
t_span = [t0, t1]

# ---------------- ASSIGNMENT 2 ------ SEIR MODEL ---------------------

exposed = 0
alpha = 0.5

initial_sum = exposed + infected + recovered
initial = [N - initial_sum, exposed, infected, recovered]


# ODE solver, determenistic
def ODE_SEIR(t, y):
    s, e, i, r = y

    s_p = -(beta) * (i / N) * s
    e_p = beta * (i / N) * s - alpha * e
    i_p = alpha * e - (gamma * i)
    r_p = gamma * i

    return np.array([s_p, e_p, i_p, r_p])


t_eval = np.arange(t_span[0], t_span[1], 0.1)

sol = solve_ivp(ODE_SEIR, t_span, initial, t_eval=t_eval)

plt.plot(sol.t, sol.y[0], label="Susceptible")
plt.plot(sol.t, sol.y[1], label="Exposed")
plt.plot(sol.t, sol.y[2], label="Infected")
plt.plot(sol.t, sol.y[3], label="Recovered")
plt.xlabel("time, t")
plt.ylabel("number of people")
plt.title("Uppgift 2, SEIR-model solve_ivp")
plt.legend()
plt.show()


# Gillespie, stochastic
initial = (N - initial_sum, exposed, infected, recovered)
coeff = (beta, gamma, alpha)


def stochEpedemic():
    M = np.array([[-1, 1, 0, 0], [0, -1, 1, 0], [0, 0, -1, 1]])
    return M
```

```
63
64
65  def propEpedemic(X, coeff):
66      beta = coeff[0]
67      gamma = coeff[1]
68      alpha = coeff[2]
69
70      s = X[0]
71      e = X[1]
72      i = X[2]
73      r = X[3]
74
75      w = np.array([beta * (i / N) * s, alpha * e, gamma * i])
76      return w
77
78
79  for i in range(5):
80      t, X = gill.SSA(propEpedemic, stochEpedemic, initial, t_span, coeff)
81
82      plt.plot(t, X[:, 0], "b")
83      plt.plot(t, X[:, 1], "g")
84      plt.plot(t, X[:, 2], "r")
85      plt.plot(t, X[:, 3], "m")
86
87  plt.plot(t, X[:, 0], "b", label="Susceptible")
88  plt.plot(t, X[:, 1], "g", label="Exposed")
89  plt.plot(t, X[:, 2], "r", label="Infected")
90  plt.plot(t, X[:, 3], "m", label="Recovered")
91
92  plt.xlabel("time, t")
93  plt.ylabel("number of people")
94  plt.title("Uppgift 2, SEIR-model stochastic")
95  plt.legend()
96  plt.show()
```

A.3 SEIRD-modell

```
import numpy as np
import scipy
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
import gillespie as gill

beta = 0.3
gamma = 1 / 7

N = 1000
infected = 5
recovered = 0
initial = [N - infected, infected, recovered]

t0 = 0
t1 = 120
t_span = [t0, t1]


exposed = 0
alpha = 0.5

# ---------------- ASSIGNMENT 3 ------ SEIRD MODEL ---------------------
dead = 0
my = 0.01

initial_sum = exposed + infected + recovered + dead
initial = [N - initial_sum, exposed, infected, recovered, dead]


# ODE solver, determenistic
def ODE_SEIRD(t, y):
    s, e, i, r, d = y

    s_p = -(beta) * (i / N) * s
    e_p = beta * (i / N) * s - alpha * e
    i_p = alpha * e - (gamma * i) - my * i
    r_p = gamma * i
    d_p = my * i

    return np.array([s_p, e_p, i_p, r_p, d_p])


t_eval = np.arange(t_span[0], t_span[1], 0.1)

sol = solve_ivp(ODE_SEIRD, t_span, initial, t_eval=t_eval)

plt.plot(sol.t, sol.y[0], label="Susceptible")
plt.plot(sol.t, sol.y[1], label="Exposed")
plt.plot(sol.t, sol.y[2], label="Infected")
plt.plot(sol.t, sol.y[3], label="Recovered")
plt.plot(sol.t, sol.y[4], label="DEATH")
plt.xlabel("time, t")
plt.ylabel("number of people")
plt.title("Uppgift 3, SEIRD-model solve_ivp")
plt.legend()
plt.show()


# Gillespie, stochastic
initial = (N - initial_sum, exposed, infected, recovered, dead)
coeff = (beta, gamma, alpha, my)
```

```
63
64
65  def stochEpedemic():
66      M = np.array(
67          [[-1, 1, 0, 0, 0], [0, -1, 1, 0, 0], [0, 0, -1, 1, 0], [0, 0, -1, 0, 1]]
68      )
69      return M
70
71
72  def propEpedemic(X, coeff):
73      beta = coeff[0]
74      gamma = coeff[1]
75      alpha = coeff[2]
76      my = coeff[3]
77
78      s = X[0]
79      e = X[1]
80      i = X[2]
81      r = X[3]
82      d = X[4]
83
84      w = np.array([beta * (i / N) * s, alpha * e, gamma * i, my * i])
85      return w
86
87
88  for i in range(5):
89      t, X = gill.SSA(propEpedemic, stochEpedemic, initial, t_span, coeff)
90
91      plt.plot(t, X[:, 0], "b")
92      plt.plot(t, X[:, 1], "g")
93      plt.plot(t, X[:, 2], "r")
94      plt.plot(t, X[:, 3], "m")
95      plt.plot(t, X[:, 4], "k")
96
97  plt.plot(t, X[:, 0], "b", label="Susceptible")
98  plt.plot(t, X[:, 1], "g", label="Exposed")
99  plt.plot(t, X[:, 2], "r", label="Infected")
100 plt.plot(t, X[:, 3], "m", label="Recovered")
101 plt.plot(t, X[:, 4], "k", label="DEATH")
102
103 plt.xlabel("time, t")
104 plt.ylabel("number of people")
105 plt.title("Uppgift 3, SEIRD-model stochastic")
106 plt.legend()
107 plt.show()
```

A.4 Vaccinationsmodellen

```
import numpy as np
import scipy
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
import gillespie as gill

beta = 0.3
gamma = 1 / 7

N = 1000
infected = 5
recovered = 0
initial = [N - infected, infected, recovered]

t0 = 0
t1 = 120
t_span = [t0, t1]

exposed = 0
alpha = 0.5

dead = 0
my = 0.01

# ---------------- ASSIGNMENT 4 ------ VACCINATIONSMODELL ---------------------
vaccinated = 0
vax_rate = 3

initial_sum = exposed + infected + recovered + dead + vaccinated
initial = [N - initial_sum, exposed, infected, recovered, dead, vaccinated]


# ODE solver, determenistic
def ODE_SEIRDV(t, y):
    s, e, i, r, d, v = y

    s_p = -(beta) * (i / N) * s - vax_rate
    e_p = beta * (i / N) * s - alpha * e
    i_p = alpha * e - (gamma * i) - my * i
    r_p = gamma * i
    d_p = my * i
    v_p = vax_rate

    return np.array([s_p, e_p, i_p, r_p, d_p, v_p])


t_eval = np.arange(t_span[0], t_span[1], 0.1)

sol = solve_ivp(ODE_SEIRDV, t_span, initial, t_eval=t_eval)

plt.plot(sol.t, sol.y[0], label="Susceptible")
plt.plot(sol.t, sol.y[1], label="Exposed")
plt.plot(sol.t, sol.y[2], label="Infected")
plt.plot(sol.t, sol.y[3], label="Recovered")
plt.plot(sol.t, sol.y[4], label="DEATH")
plt.plot(sol.t, sol.y[5], label="Vaccinated")
plt.xlabel("time, t")
plt.ylabel("number of people")
plt.title("Uppgift 4, Vaccination-model solve_ivp")
plt.legend()
plt.show()

```

```
# Gillespie, stochastic
initial = (N - initial_sum, exposed, infected, recovered, dead, vaccinated)
coeff = (beta, gamma, alpha, my, vax_rate)


def stochEpedemic():
    M = np.array(
        [
            [-1, 1, 0, 0, 0, 0],
            [0, -1, 1, 0, 0, 0],
            [0, 0, -1, 1, 0, 0],
            [0, 0, -1, 0, 1, 0],
            [-1, 0, 0, 0, 0, 1],
        ]
    )
    return M


def propEpedemic(X, coeff):
    beta = coeff[0]
    gamma = coeff[1]
    alpha = coeff[2]
    my = coeff[3]
    vax_rate = coeff[4]

    s = X[0]
    e = X[1]
    i = X[2]
    r = X[3]
    d = X[4]
    v = X[5]

    w = np.array([beta * (i / N) * s, alpha * e, gamma * i, my * i, vax_rate])
    return w


for i in range(5):
    t, X = gill.SSA(propEpedemic, stochEpedemic, initial, t_span, coeff)

    plt.plot(t, X[:, 0], "b")
    plt.plot(t, X[:, 1], "g")
    plt.plot(t, X[:, 2], "r")
    plt.plot(t, X[:, 3], "m")
    plt.plot(t, X[:, 4], "k")
    plt.plot(t, X[:, 5], "y")

plt.plot(t, X[:, 0], "b", label="Susceptible")
plt.plot(t, X[:, 1], "g", label="Exposed")
plt.plot(t, X[:, 2], "r", label="Infected")
plt.plot(t, X[:, 3], "m", label="Recovered")
plt.plot(t, X[:, 4], "k", label="DEATH")
plt.plot(t, X[:, 5], "y", label="Vaccinated")

plt.xlabel("time, t")
plt.ylabel("number of people")
plt.title("Uppgift 4, Vaccination-model stochastic")
plt.legend()
plt.show()
```

A.5 Egen vaccinationsmodell

```
import numpy as np
import scipy
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
import gillespie as gill

beta = 0.3
gamma = 1 / 7

N = 1000
infected = 5  # Initial
recovered = 0  # Initial

t0 = 0
t1 = 120
t_span = [t0, t1]

# From SEIR
exposed = 0  # Inital
alpha = 0.5

# From SEIRD
dead = 0  # Initial
my = 0.01

# From SERIDV
vaccinated = 0  # Initial
vax_rate = 3


# SIERDV
def ODE_SEIRDV(t, y):
    s, e, i, r, d, v = y

    s_p = -(beta) * (i / N) * s - vax_rate
    e_p = beta * (i / N) * s - alpha * e
    i_p = alpha * e - (gamma * i) - my * i
    r_p = gamma * i
    d_p = my * i
    v_p = vax_rate

    return np.array([s_p, e_p, i_p, r_p, d_p, v_p])


initial_sum = exposed + infected + recovered + dead + vaccinated
initial_SEIRDV = [N - initial_sum, exposed, infected, recovered, dead, vaccinated]
t_eval = np.arange(t_span[0], t_span[1], 0.1)

sol_SEIRDV = solve_ivp(ODE_SEIRDV, t_span, initial_SEIRDV, t_eval=t_eval)

# ---------------- ASSIGNMENT 5 ------ EGEN SMITTSPRIDNINGSMODELL
    ---------------------

delta = 0.5  # rate av R -> S
epsilon = 0.94  # vaccine protection rate
zeta = 0.3  # rate av R -> Im
second_dose_rate = 5
immune = 0  # Initial

initial_sum = exposed + infected + recovered + dead + vaccinated + immune
initial = [N - initial_sum, exposed, infected, recovered, dead, vaccinated, immune]

```

```
62
63  # ODE solver, determenistic
64  def ODE_5(t, y):
65      s, e, i, r, d, v, im = y
66
67      s_p = -(beta) * (i / N) * s - vax_rate + delta * r + (1 - epsilon) * v
68      e_p = beta * (i / N) * s - alpha * e
69      i_p = alpha * e - (gamma * i) - my * i
70      r_p = gamma * i - delta * r - zeta * r
71      d_p = my * i
72      v_p = vax_rate - (1 - epsilon) * v - min(second_dose_rate, v)
73      im_p = min(second_dose_rate, v) + zeta * r
74
75      return np.array([s_p, e_p, i_p, r_p, d_p, v_p, im_p])
76
77
78  t_eval = np.arange(t_span[0], t_span[1], 0.1)
79
80
81  sol = solve_ivp(ODE_5, t_span, initial, t_eval=t_eval)
82
83  s, e, i, r, d, v, im = sol.y
84  total_population = s + e + i + r + d + v + im
85
86  # Plot to check if total population remains constant
87  plt.plot(sol.t, total_population, label="Total Population")
88  plt.axhline(y=N, color="r", linestyle="--", label="Expected N=1000")
89
90  plt.plot(sol.t, sol.y[0], label="Susceptible")
91  plt.plot(sol.t, sol.y[1], label="Exposed")
92  plt.plot(sol.t, sol.y[2], label="Infected")
93  plt.plot(sol.t, sol.y[3], label="Recovered")
94  plt.plot(sol.t, sol.y[4], label="DEATH")
95  plt.plot(sol.t, sol.y[5], label="Vaccinated")
96  plt.plot(sol.t, sol.y[6], label="Immune")
97
98  plt.plot(sol_SEIRDV.t, sol_SEIRDV.y[0], label="Susceptible OLD")
99  plt.plot(sol_SEIRDV.t, sol_SEIRDV.y[1], label="Exposed OLD")
100 plt.plot(sol_SEIRDV.t, sol_SEIRDV.y[2], label="Infected OLD")
101 plt.plot(sol_SEIRDV.t, sol_SEIRDV.y[3], label="Recovered OLD")
102 plt.plot(sol_SEIRDV.t, sol_SEIRDV.y[4], label="DEATH OLD")
103 plt.plot(sol_SEIRDV.t, sol_SEIRDV.y[5], label="Vaccinated OLD")
104
105
106 plt.xlabel("time, t")
107 plt.ylabel("number of people")
108 plt.title("Uppgift 5, Egen Vaccination-model solve_ivp")
109 plt.legend()
110 plt.show()
111
112
113 # Gillespie, stochastic
114 initial = (N - initial_sum, exposed, infected, recovered, dead, vaccinated, immune)
115 coeff = (beta, gamma, alpha, my, vax_rate, delta, epsilon, second_dose_rate, zeta)
116
117
118 def stochEpedemic():
119     M = np.array(
120         [
121             [-1, 1, 0, 0, 0, 0, 0],
122             [0, -1, 1, 0, 0, 0, 0],
123             [0, 0, -1, 1, 0, 0, 0],
124             [0, 0, -1, 0, 1, 0, 0],
```

```
            [-1, 0, 0, 0, 0, 1, 0],
            [1, 0, 0, -1, 0, 0, 0],
            [1, 0, 0, 0, 0, -1, 0],
            [0, 0, 0, 0, 0, -1, 1],
            [0, 0, 0, -1, 0, 0, 1],
        ]
    )
    return M


def propEpedemic(X, coeff):
    beta = coeff[0]
    gamma = coeff[1]
    alpha = coeff[2]
    my = coeff[3]
    vax_rate = coeff[4]
    delta = coeff[5]
    epsilon = coeff[6]
    second_dose_rate = coeff[7]
    zeta = coeff[8]

    s = X[0]
    e = X[1]
    i = X[2]
    r = X[3]
    d = X[4]
    v = X[5]
    im = X[6]

    w = np.array(
        [
            beta * (i / N) * s,
            alpha * e,
            gamma * i,
            my * i,
            vax_rate,
            delta * r,
            (1 - epsilon) * v,
            min(second_dose_rate, v),
            zeta * r,
        ]
    )
    return w


for i in range(5):
    t, X = gill.SSA(propEpedemic, stochEpedemic, initial, t_span, coeff)

    plt.plot(t, X[:, 0], "b")
    plt.plot(t, X[:, 1], "g")
    plt.plot(t, X[:, 2], "r")
    plt.plot(t, X[:, 3], "m")
    plt.plot(t, X[:, 4], "k")
    plt.plot(t, X[:, 5], "y")
    plt.plot(t, X[:, 6], "tab:pink")

total_population = X[:, 0] + X[:, 1] + X[:, 2] + X[:, 3] + X[:, 4] + X[:, 5] + X[:,
    6]

# Plot to check if total population remains constant
plt.plot(t, total_population, label="Total Population")
plt.axhline(y=N, color="r", linestyle="--", label="Expected N=1000")

```

```
plt.plot(t, X[:, 0], "b", label="Susceptible")
plt.plot(t, X[:, 1], "g", label="Exposed")
plt.plot(t, X[:, 2], "r", label="Infected")
plt.plot(t, X[:, 3], "m", label="Recovered")
plt.plot(t, X[:, 4], "k", label="DEATH")
plt.plot(t, X[:, 5], "y", label="Vaccinated")
plt.plot(t, X[:, 6], "tab:pink", label="Immune")

plt.xlabel("time, t")
plt.ylabel("number of people")
plt.title("Uppgift 5, Egen Vaccination-model stochastic")
plt.legend()
plt.show()
```

A.6 Gillespie

```
#==================================
# Gillespies algoritm
# prop - propensities (1D numpy-array)
# stoch - Stoichiometry matrix (2D numpy-array)
# X0 - initial state (list, tuplet or numpy-array)
# tspan - silulation time interval
# coeff - model parameters
#==================================
import numpy as np
import random

def SSA(prop, stoch, X0, tspan, coeff):
    # prop  - propensities
    # stoch - stiochiometry vector
    # Initial state vector
    tvec = np.zeros(1)
    tvec[0] = tspan[0]
    Xarr = np.zeros([1,len(X0)])
    Xarr[0,:] = X0
    t = tvec[0]
    X = X0
    sMat = stoch()
    while t<tspan[1]:
        r1, r2 = np.random.uniform(0,1, size=2)  # Find two random numbers on
            uniform distr.
        re = prop(X,coeff)
        cre = np.cumsum(re)
        a0 = cre[-1]
        if a0 < 1e-12:
            break

        tau = np.random.exponential(scale=1/a0) # Random number exponential
            distribution
        cre=cre/a0
        r=0
        while cre[r]<r2:
            r+=1

        t+= tau
        # if new time is larger than final time, skip last calculation
        if t > tspan[1]:
            break

        tvec = np.append(tvec,t)
        X = X + sMat[r,:]
        Xarr = np.vstack([Xarr,X])

     # If iterations stopped before final time, add final time and no change
    if tvec[-1] < tspan[1]:
        tvec=np.append(tvec,tspan[1])
        Xarr=np.vstack([Xarr,X])

    return tvec, Xarr
```