UPPSALA
UNIVERSITET

1650-8319, UPTEC STS 21039
Examensarbete 30 hp
Mars 2021

Flödessimulering av blodprov

Teddy Wickström

Civilingenjörsprogrammet i system i teknik och samhälle

UPPSALA
UNIVERSITET

Simulation modelling of workflow for blood cultures

Teddy Wickström

Abstract

In this thesis the workflow of blood cultures was simulated, in order to evaluate the impact of changes, in routines or alternative flows, on the total time to result. A base-case-model was created from hospital clinical records and their corresponding flow description, using classical and Bayesian inference models combined with discrete event simulation. Alternative flows, with a focus on flows related to products/concepts offered by Q-Linea, were created according to their specifications and compared to the base-case. The results indicated a clear difference between different workflows in benchmarks such as time to positivity and total time to results. The results showed that alternative flows have the potential to minimize the effects of lengthy steps. Although the results also indicated that a lot of limitations in the workflow stem from routines such as frequency of transportation and opening hours of the laboratory.

The findings in this thesis should be viewed as a starting point, or a justification, for further examination of the different steps in the workflow. And perhaps give an indication as to which steps has the greatest potential of reducing the total time to results.

Teknisk-naturvetenskapliga fakulteten

Uppsala universitet, Utgivningsort Uppsala/Visby

Handledare: Mats Gullberg Ämnesgranskare: Rolf Larsson

Examinator: Elísabet Andrésdóttir

Populärvetenskaplig sammanfattning

Sepsis och är ett livshotande tillstånd som orsakas av en kraftig reaktion på en infektion. Hos patienter med sepsis uppvisas en hög dödlighet[1]. För patienter i septisk chock, en allvarligare underkategori av sepsis ökar dödligheten med 8% för varje timme utan eller med felaktig antibiotikabehandling under de första 6 timmarna[2]. Det är alltså av största vikt att rätt antibiotikabehandling tidigt sätt in, det är därför av stort intresse att analysera tiden till resultat (TTR). För att finna rätt antibiotikabehandling behöver en rad laboratorietester utföras.
Målet med dessa tester är delvis att finna antibiotika som är effektiv, men även att identifiera bakterien. Teknikerna som används för dessa tester varierar. Likaså varierar arbetstider och tider för transport till laboratoriet. Olika rutiner kan leda till stora variationer i hur snabbt resultat från tester kan ges. En simulering av flödet har därför skapats med syfte att studera hur olika förändringar kan tänkas påverka tiden till resultat. I detta arbete skapades ett basfall med utbytbara moduler som representerar de 6 på varandra följande stegen:

- Tagning av blodprov
- Transport till laboratoriet
- Inkubation av provet
- Gramfärgning
- Identifiering av bakterien (ID)
- Identifiering av effektiv antibiotika (AST)

som alla blodprov genomgår. Efter jämförelse med klinisk data, för att se om modellen producerar rättvisa resultat, kunde modellen varieras för att modellera en jämförelse med andra alternativa system ämnade att förändra flödet och på såsätt förändra tidsåtgången kring olika steg. Två av dessa alternativ är baserade på Q-lineas produkter och ideér kring andra flödesalternativ. Varav ena alternativet undersöker effekten av att påbörja inkubationen direkt med hjälp av ett **portabelt blododlingsskåp** [3] samt ett alternativ som möjliggör **snabb AST parallellt** med ID, som det vanliga flödet annars måste vänta på.
Jämförelser kunde även göras genom att justera vissa aspekter av en befintlig simulering, för att exempelvis undersöka hur förändrade arbetstider eller rutiner kring transporter skulle kunna tänkas påverka tiden till de olika stegen och framförallt TTR.
Resultaten visade att simuleringar med det portabla blododlingsskåpet resulterade i 12% snabbare medeltid till TTR jämfört med simulering av basfallet. Den snabba och parallella AST varianten hade 20% snabbare medeltid till resultat jämfört med basfallet.Simulering av arbetstidsförändring hos basfallet visade exempelvis att en ändring av arbetstiderna hos laboratoriet från [07.00 - 20.00] till [06.00 - 22.00] resulterar i en ca 5% förkortningar av TTR.

Studiens slutsats är att olika alternativa flöden tycks påverka tiden till resultat, vidare finns mycket tid att vinna på förändring av rutiner som arbetstider. Dock menar författaren att vidare studier med ett större dataunderlag kan vara nödvändiga för att utvärdera denna typ av modell i andra flöden.

Innehåll

Kapitel 1

Inledning

Sepsis är ett livshotande tillstånd som orsakas av ett stört systematiskt svar på infektion vilket leder till organdysfunktion. **Septisk chock** är en undergrupp av sepsis där de metabola/cellulära och cirkulatoriska störningarna är tillräckligt uttalade för att markant öka dödsrisken. *Vid dessa tillstånd uppvisas en hög dödlighet men ett snabbt och adekvat omhändertagande kan ha betydelse.* [1] När patienten uppvisar misstänkt sepsis tas blodprov för att bestämma vilken typ av bakterie det är frågan om samt vilken typ av antibiotika som är effektiv. När prover är tagna påbörjas så kallad *empirisk antibiotikabehandling*; resonemanget kring val av antibiotika baseras på infektionens svårighetsgrad, underliggande sjukdomar samt andra epidemiologiska uppgifter. [1] Tidig insättning av antibiotikabehandling kan vara livsavgörande, dock är all antibiotikabehandling drivande i uppkomst och spridning av av resistenta och svårbehandlade patogener.[1] Även därför är det av stor vikt med snabba resultat då den initiala breda empiriska behandlingen kan smalnas ner så fort resultat från blodtester producerats.

Vid misstänkt sepsis är tiden till resultat av blodtester kritisk. För patienter i septisk shock ökar dödligheten med 8 % per timme med felaktig eller avsaknad av behandling under de första 6 timmarna[2]. Infektionerna medför hög mortalitet, långa vårdtider och stora kostnader för samhället. Tidig identifiering samt adekvat antibiotikabehandlig ses som den viktigaste åtgärden för att minska dessa konsekvenser[1].
Resultat från undersökningar visar att det finns stort utrymme för förbättringar i tid till resultat och även att förbättringar kan ske genom bättre användning av befintliga resurser[4].

I detta arbete studeras och jämförs alternativa arbetsflöden. Jämförelserna utförs genom modellering och simulering av dessa olika flöden för att senare utvärdera förväntade resultat av åtgärder. Arbetet utfördes på Q-linea, ett Uppsalabaserat företag som utvecklar dessa typer av lösningar. Bland de utvärderade flödena återfinns därför ett visst fokus på produkter och koncept från Q-linea.

1.1 Problembeskrivning

Tidig behandling med passande antibiotika minskar dödligheten, vårdtiden samt begränsar utvecklingen av Antibiotikaresistens hos bakterier. [5] Därför har stor fokus lagts på att utveckla snabbare tester samt andra förbättringar för att påskynda tiden till resultat av blodtester. De exakta effekterna av dessa ändringar är dock svåra att evaluera.

1.2 Syfte

Studiens syfte är att undersöka och jämföra alternativa processer för testing av blodprov genom simulering. Detta utförs i två steg.

- Undersök ifall en simulering kan skapas utifrån data och en systembeskrivning och om denna sedan kan återproducera samma typ av data
- Genom förändring av parametrar och struktur ändra simuleringen, representera andra alternativa flöden, för att sedan jämföra dessa.

Simuleringen behöver sedan kunna användas på ett lättillgängligt sätt. Därför utvecklas även ett enkelt gränssnitt som möjliggör smidig simulering av olika flödesvarianter.

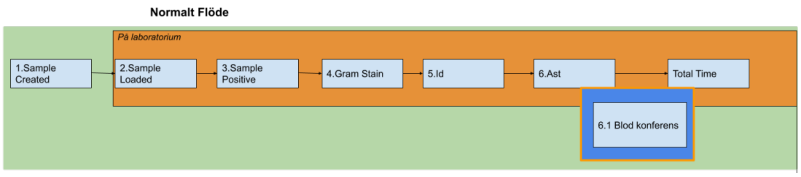
Kapitel 2

Bakgrund

För att skapa en simulering är det viktigt att få en god överblick över flödet som skall modelleras. Nedan följer generella beskrivningar, problematisering samt en rad flödesbeskrivningar. De flöden samt begränsningar kring dessa kommer senare ligga som grund för modelleringen av de olika stegen i flödet och avslutningsvis beskrivas av olika inparametrar för att smidigt kunna generera olika flödesvarianter. Beskrivningen, om det allmäna flödet, är baserad på hur data är loggad.

2.1 Beskrivning av det allmänna flödet

Denna flödesbeskrivning motsvarar basfallet för hur blodprov hanteras och därmed även basfallet för simuleringen. Den data som simuleringen försöker efterlikna beskrivs i stor utsträckning av nedanstående flödesschema figur 2.1.



Figur 2.1: Normalt Flöde

2.1.1 Sample Created

Det studerade arbetsflödet börjar då blodprovet tas på sjukhuset. Detta kan ske dygnet runt när patienten misstänks ha sepsis. Vanligen dras fyra flaskor blod för att minska risken för kontaminering.

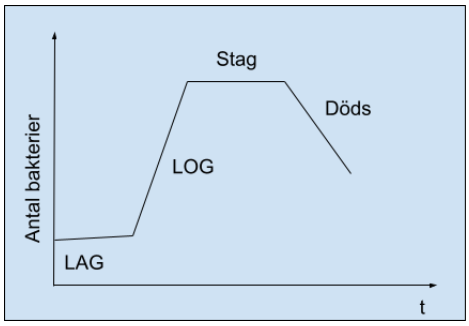
2.1.2 Sample Loaded

När blodprovet anländer till labbet sätts det in i ett blododlingsskåp, Blood culture cabinett (BCC) för inkubation. Sample Loaded syftar på tiden från att blodprovet tagits tills att det laddats in i BBC. Detta sker i vanliga fall under arbetstid 08:00 – 19:00. Transporttiden till laboratoriet, som är en underkomponent i detta steg, varierar mellan sjukhus. De som inte har eget laboratorium skickar sina prover med kurir fördelat under dagen, vanligen en transport under morgonen, kring lunch samt en under eftermiddagen. För att försäkra sig om snabba och precisa resultat bör denna tid hållas kort, tyvärr är det vanligt med förseningar pågrund av långa transporter samt laboratoriets öppettider [6] . Dessa förseningar kan även leda till felaktiga testresultat[7] , något som diskuteras nämnare i nästa steg.

2.1.3 Sample Positive

Tid positivitet (TP) beräknas som tiden tills provet blir positivt **i ett BCC**. Positivitet innebär att eventuella bakterier i provet växt, och därmed producerat en viss mängd koldioxid [6]. Då ändrar ett membran i flaskans botten färg. Censorer i odlingsskåpet känner av denna färgförändring, loggar tiden och signalerar att provet är redo för vidare analys. Tillväxthastigheten skiljer sig åt mellan bakterier[8], och påverkas även av

mängden bakterier i blodprovet samt förekomst av antibiotika i provet[9]. Prov som inte resulterar i ett positivt testresultat inom 5 dagar klassas vanligen som som negativa och slängs. *Prov som gett ett positivt resultat innan 120 minuter kasseras vanligtvis då den snabba resultat tiden sannolikt orsakats av kontaminering vid provtagningen. ?*
Bakteriell tillväxt I figur 2.2 beskrivs tillväxtfaserna för en bakterie. Under **LAG**-fasen sker lite eller ingen



Figur 2.2: Bakteriell tillväxt

tillväxt då bakterierna vänjer sig vid det nya tillväxtmediet. **LOG**-fasen kännetecknas av exponentiell tillväxt, det är även här censorn känner av en förändring. Så det är viktigt att provet laddats in innan denna tillväxtfas är passerad, för att BBC:n skall kunna känna av en förändring. Dock är detta enbart aktuellt för bakterier som har möjlighet till tillväxt utanför inkubatorer eller BCC då övriga stannar i **LAG**-fasen tills nödvändiga förhållanden infinner sig. Under **Stag**nations-fasen avstannar tillväxten då antalet bakterier ökat till taket av näring mediet tillhandahåller. I brist på näring börjar Döds-fasen och antalet bakterier minskar igen. [10]
Det finns även dokumenterade skillnader i TP för olika BCC[8]. I det studerade földet används ett *BacT/ALERT* skåp. Eftersom vissa bakterier kan växa utanför blododlingsskåpet införs även begreppet total tid till posivitivet **TTP** som räknas från provtagningstillfället.

2.1.4 Gram Stain

Gramfärgning är det första testet som görs på laboratoriet för att ta reda på vilken kategori av bakterie det är frågan om. I testet avgör om bakterien är *gram positiv* eller negativ baserat på vilken färg bakterierna får efter överstrykning med *Safranin* [11]

2.1.5 ID

Med ID menas identifiering av bakterien. *Klassisk* ID kräver isolering av bakterien innan testet kan påbörjas. Isoleringen utförs vanligen över natten. När bakterien är isolerad tar själva testet ca 1 timme. Total tid räknas vanligen till ca 18-24h. Det finns även snabbvariant där isoleringsteget inte är nödvändigt. Denna snabbvariant kallas *MALDI-ToF-MS* (MALDI) som genom mass spectrometri identifierar bakterien [12]. I det flödet som skall bli basfallet används MALDI och testet tar ca 6h .

2.1.6 Ast

Vid ett Antimicrobial susceptibilitets test (Ast) avgörs av vilken typ av antibiotika som är lämplig. Den klassiska varianten utförs vanligen över natten och tar ca 18 ± 2 timmar då testet kräver inkubation. Det finns även automatiserade system med varierande kapacitet där svarstiden kan ligga inom (3.5-16) timmar.[13]

2.1.7 Blodkonferens

Konferens när resultaten av blodtester utförda under det senaste dygnet diskuteras och ett slutgiltigt resultat sparas och presenteras för vårdhavande läkare.

2.2 Alternativa flöden

Alla sjukhus eller avdelningar ser inte likadana ut, de har olika förutsättningar och rutiner kring transporter samt olika lösningar som gör att tiden mellan stegen samt ordningen på stegen varierar. Nedan följer

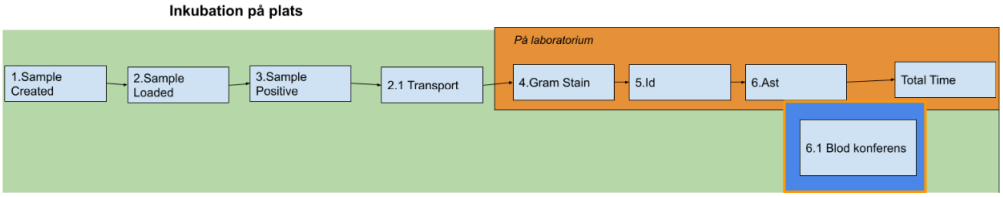
illustrationer på andra vanliga flödesbeskrivningar. Det finns studier som antyder på att tiden till positivitet ökar med längre tider i rumstemperatur, i vår beskrivning *tiden till inladdning*. Detta kan bero på att bakterier går in i en vilofas om de inte inkuberats inom ett viss tidsintervall [4].

2.2.1 Fullständig inkubation innan laboratoriet

Vissa avdelningar har tillgång till ett eget BCC, detta möjliggör att inkubationen kan påbörjas direkt (läs relativt snabbt) efter att provet har tagits. Fördelen med detta är att enbart prover som flaggats positiva skickas till laboratoriet vilket minskar trycket. Inkubationen av provet kan påbörjas snabbare och oberoende av rutiner kring transporter.

2.2.2 Portabel blododling (TCC)

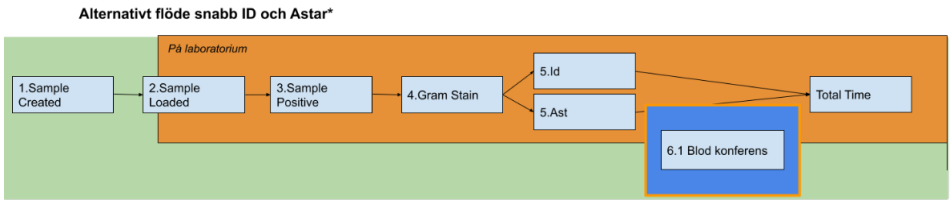
Q-linea har påbörjat utvecklingen av ett portabelt blododlingskåp,*Transportable Culture Cabinett* med syfte att bättre utnyttja tiden mellan provtagning och anländandet vid laboratoriet. Tanken är att provet kan laddas in i **TCC**:n, och påbörja odlingsporcessen direkt efter provtagning. Detta flödesalternativ innebär alltså, jämfört med de andra alternativen, framförallt att transporttiden kan tas tillvara på [3].



Figur 2.3: Inkubation innan Lab

2.2.3 Parallel Snabb AST (PSA)

Det finns även snabbare tekniker att tillgå för Ast. I figur 2.4 presenteras ett flöde med teknik från Q-lineas som möjliggör snabb Ast parallellt med ID [14]. I detta flöde behöver vi alltså inte vänta på ID för att utföra Ast.



Figur 2.4: Snabb Id och PSA

2.3 Arbetstider och Weekend Effect

Tidsåtgången för hantering av blodprov varierar mellan sjukhus. Detta beror mycket på att olika sjukhus och laboratorier har olika rutiner och arbetstider. I många fall syns även längre bearbetningstider under under helgerna då det är vanligt med kortare arbetspass och färre transporter. Detta fenomen kallas *weekend effekten* och tydliggörs i synnerhet för tidskrävande steg som *TTP* då blodprov i många fall måste komma till laboratoriet för detta steg [4].

2.3.1 Återkoppling till syfte

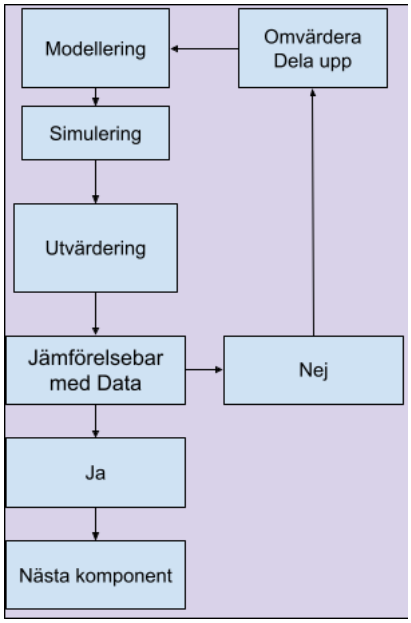
Som tidigare beskrivits i inledningen är det av stor vikt för vårdhavande läkare att resultat snabbt tillhandahålls för att möjliggöra att rätt behandling kan påbörjas. Ett av målen med simuleringen bör alltså vara att effekten av förändringar i arbetstider och därmed minimering av *weekendeffekten* kan observeras samt ge möjligheten att kvantifiera förbättringspotentialen i ett alternativt flöde.

Kapitel 3

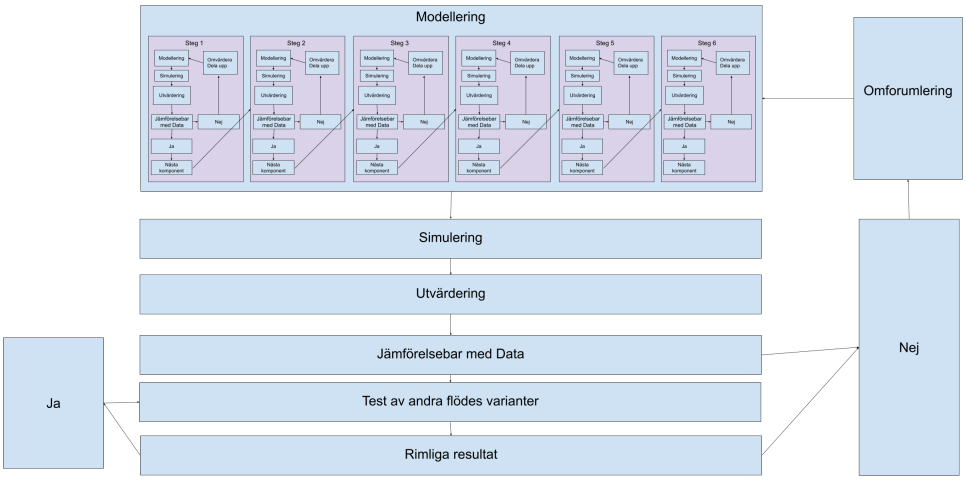
Metod

Arbetet påbörjades med en kortare förstudie för att bekanta mig med processerna kring blodprovshantering. Hanteringen delades sedan upp i olika steg i linje med tidigare studiers uppdelningar samt med beaktande för vilka steg data för tidsåtgång finns att tillgå.

Simuleringsmodeller för dessa steg togs fram genom olika inferensmetoder. Om modellen inte var jämförbar med data eller uppvisade andra brister formulerades modellen om och eller delades upp i mindre komponenter motiverade av den generella systembeskrivningen. Detta modelleringsabete utfördes alltså iterativt mellan de olika stegen med validering mot originaldata. När sedan en version av hela flödet skapats utvärderades den på samma iterativa vis. I figur 3.1 demonsterars arbetsgången för ett steg. I figur 3.2 ses arbetsgången för hela systemet



Figur 3.1: Modellering av ett steg



Figur 3.2: Modellering av hela flödet

3.1 Simulering som metod

En simulering använder sig av abstrakta modeller för att beskriva egenskaper hos ett system. Systemets processer modelleras med hjälp av sannolikhetsfördelningar för att generera slumpmässiga systemhändelser. Detta är en approach som ofta används i modellering av av komplexa system för till exempel riskanalys, där olika delar, med stor eller viss osäkerhet finns kring deras initiala värden, av flödet kan beskrivas av slumptalsgeneratorer från lämpliga fördelningar. Dessa fördelningar beskriver alltså osäkerheten hos parametrarna eller variabiliteten hos hitintill ej eller bara delvis observerade kvantiteter[15]. Fördelen med en simulering är att man snabbt kan ändra inputparametrarna för att snabbt testa alternativa metoder och resultatet av dessa, simuleringen ger även fördelen att snabbare kunna se resultat på tänkta förändringar[16]. Eftersom möjligheten till variation av simuleringen, för olika scenarion, är en så stor del av simuleringens ´syfte behöver en avvägning för de flesta stegen göras. Det kanske tydligaste exemplet på detta är steget för inladdning i blododlingskåpet, Sample loaded, där transporttiden är inkluderad. Här faller det sig naturligt att transporttiden kan variera mycket mellan olika system, därför väljs ibland inte bara de mest optimala fördelningarna utan de som lämpar sig bäst ur ett modelleringsperspektiv.

3.2 Använd mjukvara

Nedan följer beskrivningar av olika verktyg som används i detta arbete.

3.2.1 SimPy

SimPy är ett ramverk för simulering av tidsdiskreta processer. Simuleringar i SimPy består av 4 olika Klasser:

- Miljö
- Event
- Processes
- Resurser

Beteendet för aktiva komponenter, i vårt fall blodprov, modelleras som processer. Alla processer existerar i en miljö och interagerar med andra processer eller miljön genom events.
För att starta en simulering i SimPy behöver du först starta en miljö där simuleringen äger rum. Miljön håller även koll på den totala tiden som passerar då händelser äger rum. Samt hur länge simuleringen skall köras. Då en process startar ett event pausas processen. Simpy startar sedan upp processen igen då eventet (händelsen) sker. Flera processer kan vänta på samma event. En viktig typ av event, som ofta används i denna simulering, är Timeout. Denna typ av event sker då en viss tid i simuleringen har passerat. Detta event låter alltså en process sova en viss tid.

SimPy tillhandahåller tre olika resurstyper. Samtliga används för att modellera eventuella begränsningar systemet har. I vårt fall kan detta vara antalet som jobbar i labbet eller maxantalet prover som kan bearbetas samtidigt i olika steg. Den resurstyp som eftersträvas ger alltså möjligheten att beskriva kapaciteten för ett viss steg.[17]

3.2.2 SciPy ekosystemet

SciPy ekosystemet är en samling av bibliotek som används för vetenskaplig databehandling i python. Detta system innehåller en rad paket som används i detta arbete. I huvudsak Pandas för strukturering av data samt SciPy. SciPy är ett Pythonbibliotek för vetenskaplig beräkning och innehåller flertalet rutiner för numerisk beräkning, bland annat numerisk integrering, interpolation, optimering, linjär algebra samt statistik[18]. I detta arbete används huvudsakligen de olika fördelningarna som finns tillgängliga. För dessa används *fit* vilken använder ML- skattning för att finna parametrar till fördelningar. Samt *rvs* metoder för att dra stickprov från dessa fördelningar.

3.2.3 PyMC3

PyMC3 är ett probibalistisk programeringspaket i Python som låter användaren anpassa Bayesiska modeller med hjälp av Markov Chain Monte Carlo (MCMC) eller variational inference (VI)[19]. Paketet tillhandahåller en enkel syntax för att specificiera modeller samt flertalet olika samplingsalgoritmer för att anpassa modellerna till data. Samplings algoritmer inom (MCMC) som No-U-Turn Sampler (NUTS) och Hamiltonian Monte Carlo (HMC) möjliggör inferens för komplexa modeller utan specialiserad kunskap kring samplingsalgoritmer och deras begränsningar eftersom NUTS har flera inbyggda själv-reglerande strategier för de reglerande parametrarna i HMC [19].

3.2.4 Tkinter

Tkinter är ett GUI bibliotek som sedan 1994 varit en del av Pythons standardbibliotek. Gränssnitt byggs upp i lager av olika rutor (Frames) som kan innehålla exempelvis widgets med olika funktionalitet. [20]

3.3 Avgränsningar

Simuleringsflödet tar enbart hänsyn till positiva resultat, alltså där bakterier identifierats i blodprovet. Detta eftersom den tillgängliga datan enbart beskriver fallet vid positiva tester.

3.4 Beskrivnig av data

Nedan följer en kort beskrivning av de olika dataset som används i framtagandet av simuleringen. Båda dataset samt beskrivningar av dessa tillhandahölls av Q-linea.

3.4.1 Dataset 1

Data hämtat från ett danskt sjukhus. Dessa data ligger som grund för större delen av simuleringen då de bra beskriver egenskaper som simuleringen skall kunna beskriva. Innehåller 512 prover.

3.4.2 Dataset 2

Data är hämtat från ett Amerikanskt sjukhus. Detta dataset används som komplement till dataset 1 för att modellera tiden till positivitet. Alltså tiden till positivitet samt tid till inladdning för dessa prover. Då modellen ämnar beskriva olika bakteriertyper och deras egenskaper kring tid till positivitet krävs en större mängd data.

Kapitel 4

Teori

I detta avsnitt kommer teorin bakom teknikerna som används för modelleringen kortfattat beskrivas. I huvudsak handlar det om två olika tillvägagångssätt för att för att finna parametrar för de olika skedena i modellen samt försök att validera dessa resultat mot data hos det verkliga systemet.

4.1 Fördelningar

I detta avsnitt presenteras olika sannolikhetsfördelningar som används i simuleringen.

4.1.1 Gammafördelningen

En gammafördelad slumpvariabel X, har täthetsfunktion

$$f(x) = \frac{\beta^{\alpha}}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x} \tag{4.1}$$

där β och α är > 0 [15]. Och gammafunktionen $\Gamma(z)$ ges av

$$\Gamma(z) = \int_0^{\infty} x^{z-1} e^{-x}\, dx \tag{4.2}$$

4.1.2 Normalfördelning

Normalfördelning, även kallad Gauss-fördelning, är kanske den viktigaste och mest frekvent använda fördelningen. Då summering av många slumpvaraiabler så är resultatet ofta normalfördelat [21].

Normalfördelningens täthetsfunktion för en slumpvariabel X defineras enligt

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{4.3}$$

och betecknas $X \sim N(\mu, \sigma)$.

4.1.3 Trunkering

När man handskas med slumpvariabler som enbart tar värden inom ett viss intervall [a,b] kan trunkering användas för att begränsa en fördelning så att även den enbart producerar resultat inom intervallet. Detta utförs enligt formel 4.4 för täthetsfunktionen samt formel 4.5 för den kumulativa fördelningsfunktionen.

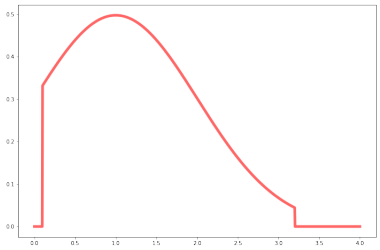
Trunkering av täthetsfunktionen

$$f(x|a < X \leq b) = \frac{g(x)}{F(b) - F(a)} = \frac{f(x) \cdot I(\{a < x \leq b\})}{F(b) - F(a)} \tag{4.4}$$

Där $g(x)$ är täthetsfunktionen för den icke trunkerade fördelnigen.
Trunkering av den kumulativa fördelningsfunktionen

$$F(x|a < X \leq b) = \frac{\int_a^x g(t)dt}{F(b) - F(a)} = \frac{F(x) - F(a)}{F(b) - F(a)} \tag{4.5}$$

Exempel trunkerad normalfördelning Den trunkerade normalfördelningen är frekvent förekommande i detta arbete och betänkas $\mathcal{TN}(\mu, \sigma, [l, u])$. I figur 4.1 presenteras ett exempel på täthetsfunktionen med μ = 1 , σ = 1 och gränser [0.1,3.2]



Figur 4.1: Exempel Trunkerad Normalfördelning

4.1.4 Halv-Normalfördelningen

Halv-Normalfördelningen betecknas $\mathcal{HN}(\sigma)$ och är ett specialfall av normalfördelingen där μ är noll och funktionen enbart är definierad för positiva värden.

4.1.5 Censurering

Censurering, jämfört med trunkering handlar istället om att observerat data har en lägre och/eller en högre gräns och all data som ligger utanför dessa ramar sparats som gränsvärdet. Detta kan uppkomma exempelvis i sammanhang då den observande censorn har begränsningar för min och max värde[22].

4.2 Wilcoxons tvåstickprovstest

Wilcoxons tvåstickprovstest kan användas för att jämföra två stickprov A: $x_1, ..., x_m$ från X samt B:$y_1, ..., y_n$ från Y. Genom att testa hypotesen H_0 :X och Y har samma fördelning, när stickprovet inte kan antas komma från någon specifik fördelning. Test bygger på att fördelningen, för stickprovs A's rangsumma R_A ges enligt ekvation 4.6

$$R_A = N(\frac{m(N+1)}{2}, \frac{m * n(N+1)}{2}) \tag{4.6}$$

där N är m + n observationer för X,Y [21]
Mann-Whitneys även kallad *Wilcoxon rank sum test* använder istället ekvation 4.7

$$U_A = R_A - \frac{m(N+1)}{2} \tag{4.7}$$

som testvariabel där $0 <= U_a <= m * n$ och fördelningen för U_a är symmetrisk kring $\frac{m(N+1)}{2}$ under H_0 [21]

4.3 Klassisk Inferens

Uppskattningen av en slumpvariabel genom klassisk inferens bygger på antagandet att slumpade experiment är oberoende upprepningsbara. Det låter oss tolka sannolikheten för en händelse A som den relativa frekvensen A är sann när antalet n repetitioner av experimentet går mot oändligheten.[15]
Denna tolkning motiveras av *De stora talens lag* (LLN) som säger att summan av alla observationer av en händelse delat med antalet försök konvergerar mot väntevärdet för händelsen.

$$\frac{1}{N}\sum_{i=1}^{N} Z_i \to E[Z], \quad N \to \infty \tag{4.8}$$

4.3.1 Maximum likelihood

Maximum likelihood används för att hitta parametrar för en sannolikhetsfördelning som beskriver data genom att maximera värdet på likelihood-funktionen beskriven nedan

$$L(\theta) = f(x_1;\theta)f(x_2;\theta)\cdots f(x_n;\theta) \tag{4.9}$$

Där x är ett slumpmässigt stickprov och $f(x;\theta)$ är fördelningens täthetsfuntion utvärderad i dessa punkter.

4.4 Bayesiansk Inferens

Inom Bayesiansk Inferens (BI) beaktas den sökta parametern, Θ, som en slumpvariabel. Så inom detta synsätt beskrivs information om Θ i termer av en fördelning[21]. BI är att genom användandet av Bayes sats 4.10 uppdatera sin tro kring hypoteser H kring dessa Θ när ny data D presenteras.

$$P(H_i|D) = \frac{P(D|H_i)P(H_i)}{P(D)} \tag{4.10}$$

Där $P(H_i|D)$ beskriver posterior sannolikheten för hypotes i, givet data, $P(D|H_i)$ är likelihood termen alltså sannolikheten för data under hypotes i, $P(H_i)$ är priori fördelningen för hypotes i och $P(D)$ är den så kallade evidenstermen eller *marignal likelihood*[23] , sannolikheten för data vilken ges av $\int_{j=i}^{n} P(D|H_j)P(H_j)$ alltså summan/integralen över alla hypoteser.[24]
Inom Bayesiansk inferens har vi som sagt en tanke om att de okända parametrarna är slumpvariabler med Prior och Posterior fördelningar. Ett värde taget från slumpvarabelns Posterior fördelning motsvarar ett *möjligt* utfall av vad den sanna parametern kan vara. [25] För att finna posteriori fördelningen $P(H_i|D)$ kan i vissa fall en analytisk lösning hittas genom väl vald priorifördelning och därmed undvika integralen i evidenstermen. Detta är dock enbart möjligt i vissa specifika fall.

4.4.1 Markov Chain Monte Carlo

När inte posteriori fördelningen kan hittas analytiskt kan den uppskattas genom Markov Chain Monte Carlo (MCMC) simulering. MCMC kan genom simulering skapa ett godtyckligt antal observationer från posteriori fördelningen [21]. Detta bygger på att integralen som beskriver evidenstermen $P(D)$ är en konstant. Därför kan Bayes sats även skrivas som $PH_i|D) \propto P(D|H_i) \cdot P(H_i)$, alltså posteriori fördelningen är proportionerlig mot produkten av likelihood och priori fördelningen. MCMC metoder använder detta för att med hjälp av slumptal och olika strategier utforska och utvärdera olika punkter i posteriorifördelningen, nya punkter accepteras genom att jämföra det aktuella värdet för $P(H_i) \cdot P(D|H_i)$. I denna process skapas en följd (ett trace) med stickprov, observationer från posteriorifördelningen samt hur de olika parametervärdena ändras. [24]. Observationerna från posteriori fördelningen är visserligen beroende men de kan ändå användas i syfte att analysera egenskaper hos posteriori fördelningen för parametrarna. Om det huvudsakliga syftet med analysen är att analysera posteriorifördelningen för en visskomponent H_j kan parametervärdet enkelt skattas som medelvärdet av observationerna [21]. Dessa observationer återfinns alltså i den ovan nämnda följden (tracet).

4.4.2 Probabilistisk programmering

Tanken med Probabilistisk programmering är att specificera probabilistiska modeller i kod för lösa dem på ett mer mindra automatiskt maner. Detta utförs genom att specificera mode Detta kan vara användbart då många modeller inte kan beskrivas på ett slutet analytiskt vis. Dessa löses därför lämpligen med numeriska metoder[23]

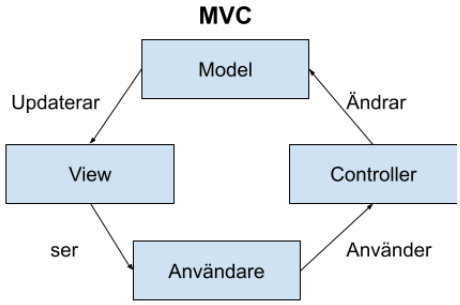
4.4.3 Hierarkiska modeller

Med Hierarariska modeller (HM) möjliggör simultan modellering av heleten och de individuella komponenterna, grupperna som den utgörs av. HM låter alltså information delas mellan grupper som skall modelleras vilket, med antagandet att grupperna inte skiljer sig allt för mycket åt, möjliggör induktion för kategorier som

har få datapunkter. Likaså kan dessa användas för att beskriva sin del i helheten. [23] HM utförs genom att sätta så kallade **hyper-priors** varifrån de kategorispecifika **priors** kommer ifrån.

4.4.4 MVC

För att hantera och organisera stora komplexa program och därmed öka möjligheterna till god skalbarhet krävs god arkitektur och koddesign. God design underlättar läsbarheten i koden och minskar risken för fel. För god design krävs att är det viktigt att följa mönster som isolerar olika funktionalitet till olika delar av programmet.
Ett klassiskt exempel på sådant mönster är **MVC** (Model-Veiw-Controller) där grundiden är att hålla data, presentation av data samt logiken i applikationen i separata komponenter.
Modeldelen hanterar data och bakomliggande struktur hos den.
View beskriver själva gränssnittet. Den visar data samt funktionalitet för användaren. Kan innehålla enklare logik för exempelvis validering inom formulär. Innehåller idealt enbart tillräcklig logik för att presentera gränssnittet till användaren samt kommunicera användarens aktioner till Controllern
Controller hanterar komandon från användaren och står för kommunikationen mellan **Modellen** och **Viewn**. Kort och gott; i princip all funktionalitet skall ligga inom Controllern [20]



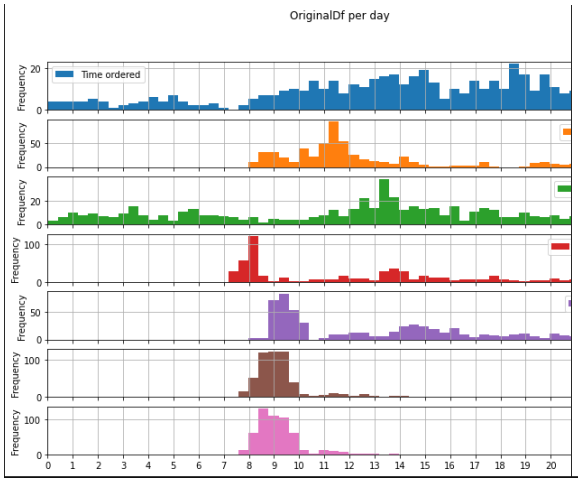
Figur 4.2: Beskrivning av MVC

Kapitel 5

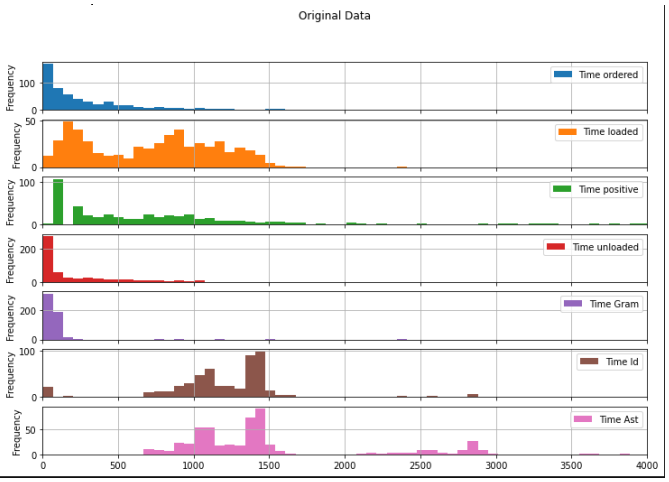
Flödesmodellering

För rättvist beskriva flödet behöver jag ta hänsyn till flera aspekter, delvis tiden mellan de olika stegen provet genomgår. Alltså tiden det tar för att genomföra exempelvis ett test, inkubationen eller transport. Men även vilka tider på dygnet stegen äger rum. I figur 5.1 ses hur frekvent olika stegen äger rum under dygnet. I figur 5.2 ges tidsåtgången i minuter mellan två steg för dataset 1.

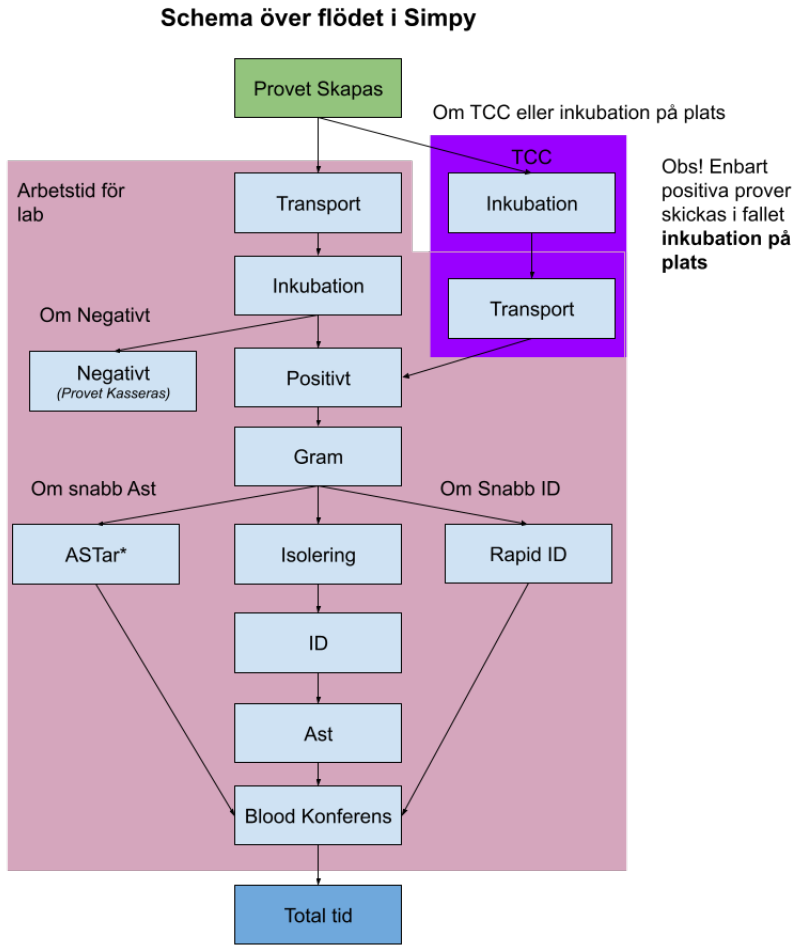
I figur 5.3 åskådliggörs strukturen för de olika flödesvarianterna i modellen. Alltså den övergripande struktur för simuleringen och hur den är utformad i SimPy. Efterföljande avsnitt går igenom huvuddragen och logiken bakom systemets individuella komponenter. Dessa avsnitt kommer även innehålla jämförelser från simuleringar för dessa avsnitt med orginaldata från dataset 1. Dessa simuleringar är gjorda med valda parametrar för att i stor utsträckning efterlikna orginaldata. I vissa fall har dock hänsyn till flexibilitet vilket leder till medveten avvikelse från data för att bättre motivera övergripande simuleringar.



Figur 5.1: Tid under dygnet (D1)



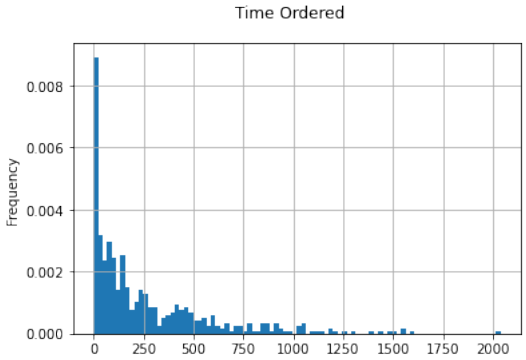
Figur 5.2: Tid för steg (D1)



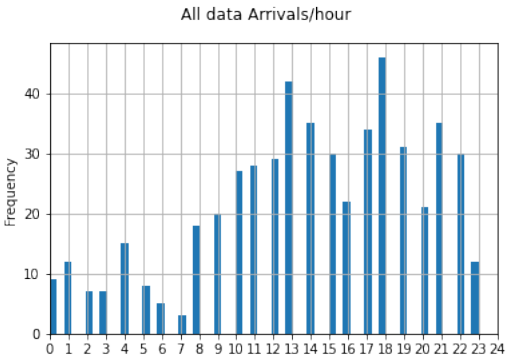
Figur 5.3: Flödesbeskrivning

5.1 Steg 1 - Provet skapas

För tidshomogena anländningsmodeller lämpar sig ofta en Poissonfördelning (process) för att beskriva antalet ankomster under en viss tidsperiod. Poissonprocessen ges av oberoende exponentialfördelade tider mellan ankomster. Detta antagande om tidsoberoende intensitet är dock ofta illa lämpat som systembeskriving.
För att med rättvisa beskriva detta behövs alltså tiden på dygnet tas i beaktande, plottad i figur 5.5. Kanske framförallt då efterföljande steg är beroende av öppettider, så om en för hög andel prover skapas utanför dessa tider blir medeltiden för hela simuleringen för långsam.



Figur 5.4: Tid mellan nya prover

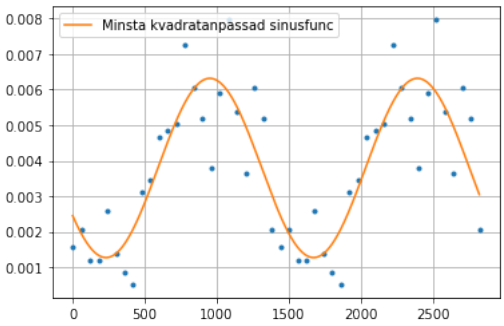


Figur 5.5: Tid på dygnet för provtagning

För att kunna beskriva denna typ av beteende, där intensiteten för blodprov som tas varierar under dygnet behövs först en intensitetsfunktion definieras. Eftersom att samma typ av beteende upprepas cykliskt dygn efter dygn faller sig en sinusfunktion lämplig. Den har även fördelen, jämfört med ett polynom eller en stegfunktion att intensiteten enkelt kan justeras då enbart amplituden, A och medelvärdet, μ för funktionen behöver ändras.

$$y(t) = A * \sin(2\pi f t + \varphi) + \mu \tag{5.1}$$

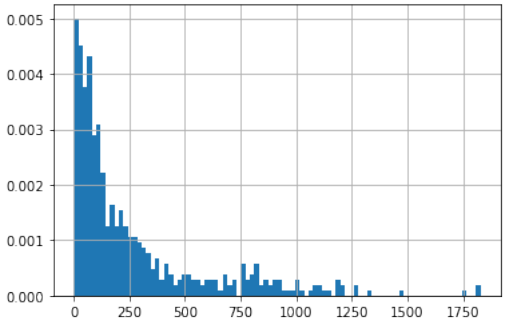
Minstakvadratanpassning från biblioteket Scipy används för att finna värdet på Amplituden $A = 0.002522$, medelvärdet, $\mu = 0.003795$ samt fasen $\varphi = 9.98744$. Frekvensen sätts till $(1/1440)$ eftersom cykeln upprepas efter ett dygn $24 * 60 = 1440$ minuter. I figur 5.6 syns denna anpassning mot datapunkterna från tid på dygnet från figur 5.5 Denna anpassningsbara sinusfunktion används sedan som input till en funktion som finner



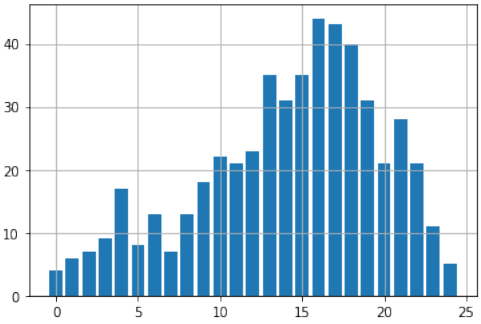
Figur 5.6: Sinuskurva för intensitet

tiden för nästa provtagning från en tidsheterogen poissonprocess. I figur 5.7 syns tidsåtgången mellan 500 simulerade prover samt i figur 5.8 tiden på dygnet dessa prover skapats.

Här kan vi konstatera att plottarna för de simulerade punkterna delvis stämmer in på orginaldata. I det färdiga programmet han användaren möjlighet att själv ge tidpunkter då prover skapats. Detta steg är alltså mer till som tilläggsalternativ då användaren inte önskar göra en direkt jämförelse med verklig data. Därför är den viktigaste egenskapen hos detta steg att intensitetsfunktionen väl beskriver skillnaden i frekvens mellan dag och natt samt att detta beteende väl beskrivs i tid på dygnet då prover skapas.



Figur 5.7: Simulerad tid mellan blodtagning



Figur 5.8: Simulerad tid på dygnet för provtagning

5.2 Steg 2 Provet Laddas in i inkubatorn

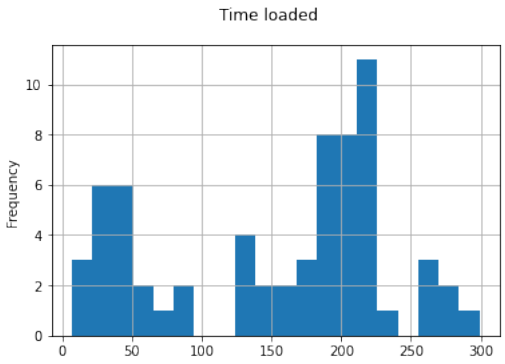
Efter att provet tagits varierar rutinerna mellan sjukhus. Vissa har laboratorium på plats på sjukhuset och har därför i regel kortare tid från dragning till att inkubationen påbörjas. Det finns även situationer där anläggningen har ett eget blododlingsskåp och inkubationen kan alltså påbörjas snabbt och transporten äger rum då provet visat sig vara positivt. Transporter mellan sjukhus varierar med ett snitt på 2-3 gånger om dagen.
I figur 5.10 syns hur tiden hur denna tidsåtgång är fördelad samt tiderna på dygnet, figur 5.11, inladdningen sker. Resterande beteende som syns i figur 5.10 antas härstamma från de tider då transport vanligen görs med den uppskattade tiden för transport adderad.
Med denna uppskattning av transporttiden återstår att härleda vilka tiderna då dessa transporter äger rum samt skapa funktionalitet för att variera tiden för transport. Från beskrivningen av detta steg fås att transport äger rum ca 8.00,10.00 samt en 14.00. Simuleringen av detta steg delas därför upp i transport och inladdning.

5.2.1 Transport

För att modellera tiden tills transport skapas en funktion som, givet en vektor med tider för transport $[TT_1, TT_2..]$, räknar ut hur lång tid det är kvar tills nästa transport äger rum.

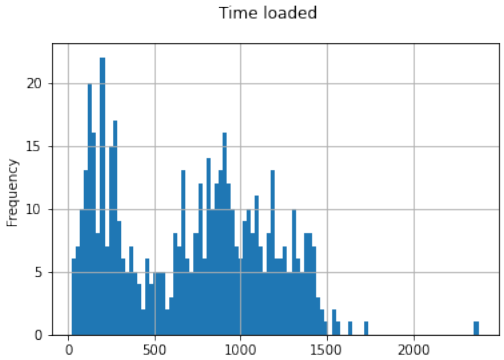
För uppskattningen tiden för den faktiska transporten isoleras de prover som tagits och transporteras under samma dygn och tagits innan klockan 8, vilket antas vara den tidigaste tiden transport är möjlig för detta dataset. Uppskattningen görs att skillnaden mellan 08.00 till tiden för insättning i skåpet ger transporttiden. Nu kvarstår 66 prover, i figur 5.9 syns fördelningen för dessa. Från figuren görs tolkningen att transporten kan ta ca 45 minuter, efterföljande peak vid 200 minuter faller bra in med $200min/60 = 03h20m \approx 11.20$ vilket antyder om att det är efterföljande transport. Det går kanske inte en transport klockan 8 alla dagar och transporterna sker kanske inte exakt på utsatt tid. För att göra denna tid enkel att variera i simulering av andra flöden väljs därför en trunkerad normalfördelning med $\mu = 45$, $\sigma = 15$ och gränser [20,60]

$$\mathrm{T} \sim \mathcal{TN}(\mu, \sigma, ([20, 60]) \tag{5.2}$$

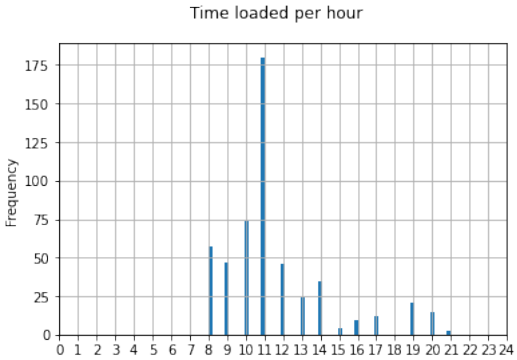


Figur 5.9: Uppskattning av transporttiden

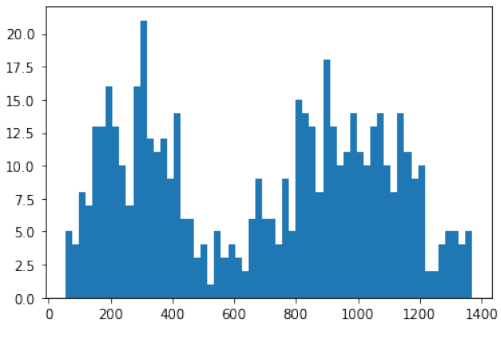
Uppskattningen av transporttiden och tiden till nästa trasport läggs sedan ihop i ett SimPy event som ser till att samtliga prover väntar på transport och leveraras samtidigt. I figur 5.12 och 5.13 ses resultatet av simulering med dessa parametrar.
Under helgerna



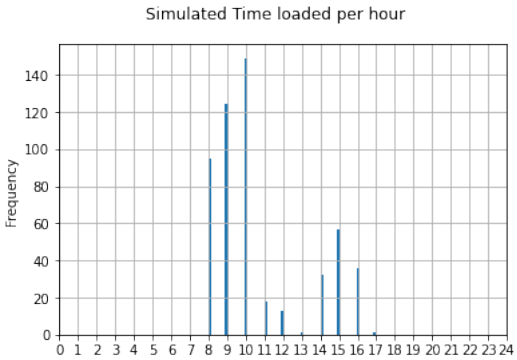
Figur 5.10: Tid till inladdning av prover



Figur 5.11: Tid på dygnet för inladdning av prover



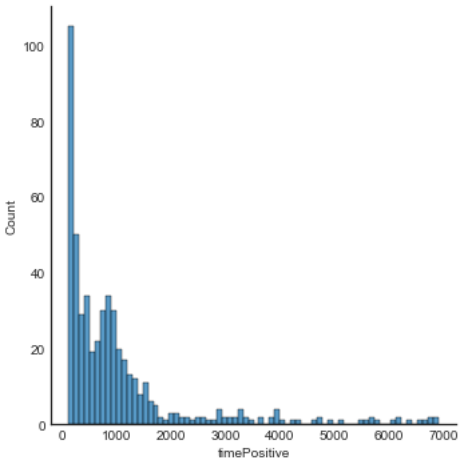
Figur 5.12: Simulerad tid för inladdning



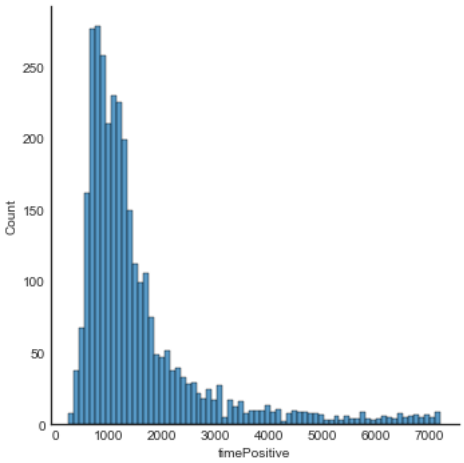
Figur 5.13: Simulerad tid för inladdning på dygnet

5.3 Steg 3 Provet är positivt

Tiden till positivitet beror som ovan nämnt på flertalet aspekter som temperatur, mängden bakterier i det tagna provet samt hur lång tid det tar innan provet sätts in i de ideala tillväxtförhållanderna blododlningsskåpet tillhandahåller. Trots vissa skillnader är detta steg det som skiljer sig minst mellan olika simulerade system, oavsett exempelvis öppettider behöver bakterierna tid att växa. Därmed läggs extra fokus på denna del av modellen. I figur 5.14 syns tiden till positivitet.



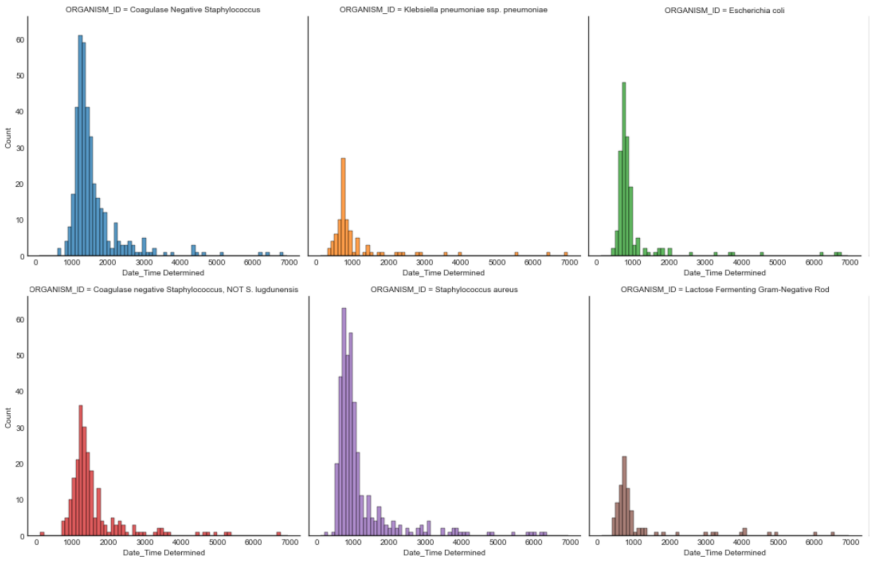
Figur 5.14: Tid till positivitet Dataset 1



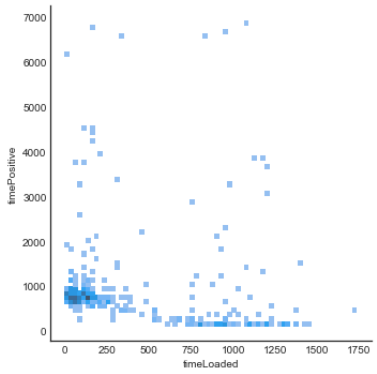
Figur 5.15: Tid till positivitet Dataset 2

Tiden tills blodprovet är klart är som tidigare nämnt begränsad av designen på blododlingsskåpet. Ett blodprov kan inte bli klart innan 120 minuter och ifall inte tillräcklig tillväxt har skett inom 5 dagar eller 7200 minuter så klassas provet som negativt. Utfallet av slumpvariabeln *tid till positiviet* är alltså begränsad inom intervallet [120,7200].

Som tidigare nämnt så växer olika bakterier olika snabbt. I figur 5.16 syns tiden till positivitet för 6 olika bakterier från dataset 2. Det blir därför vikigt för simuleringen att prevalensen för bakterier kan defineras. Så att liknande andel av snabba/medel/långsamt växande bakterier modelleras. För vissa bakterier kan tillväxt ske i rumstemperatur, dessa prover kan alltså dra nytta av tiden innan inladdning. I figur 5.17 syns tiden till positivitet plottad mot tiden innan provet satts in i skåpet för en sådan bakterie, i detta fall E.coli, liknande resultat har upptäckts i tidigare studier, vid en av dem upptäcktes en medeltid på ca 4 timmar kortare tid till positivitet i BCC efter 8.5 timmar, jämfört med 30 minuters, innan inladdning[26].
Värt att notera med dataset 2 är att tiden till inladdning är låg för detta dataset med en medeltid på 116 min och en mediantid på 94 min . I jämförelse med dataset 1 där medeltiden för inladdning är 715 min och medianen 767 min. Detta, tillsammans med prevalensen av snabbväxande bakterier antas vara den huvudsakliga förklara skillnaderna mellan tiden till positivitet för dataset 1 och 2.



Figur 5.16: Tid till positivitet top 6 Dataset 2



Figur 5.17: E.coli tid till positivitet vs load Dataset 1

5.3.1 Hierarkiska modeller

För att modellera beteendet där tiden innan provet laddats in i blododlingsskåpet påverkar totaltiden för inkubation kommer olika varianter av Hierarkiska regressionsmodeller användas. En Hierarkisk modell lämpar sig väl då vi för vissa bakterier har få datapunkter att utgå ifrån. En hierarkisk modell ger oss även möjligheten att använda oss av tidigare kunskap för slumpvariabeln *Tid till positivitet* samt variabler som kan tänkas påverka utfallet av denna. I denna bayesianska modelltyp ges även möjlighet att begränsa olika förklarande slumpvariabler och möjlighet till regression av trunkerad och censurerad data ges med funktionalitet delvis inbyggt i PyMC3's bibliotek. Denna modell använder sig av PyMC3's inbyggda sannolikhetsfunktioner för Normal, Halvnormal samt Trunkerad normalfördelning. Även *Potential*-funktionen används för att ytterligare begränsa och styra uttrycket av valda slumpvariabler.

Formatering av data För att få ett större dataset läggs dataset 1 och 2 ihop och formateras så att benämningen för de olika bakterierna är lika för dessa. Sedan väljs de 40 vanligaste bakterierna ut och numreras [0-39], resterande bakterier delas in i två kategorier. De som antas dra nytta av tiden innan inladdning får index 41 samt övriga index 40. Denna grova indelning av övriga bakterier görs genom att studera dataset 1 och de bakterier som blivit klara under 130 minuter. Något som antas vara omöjligt om inte tiden innan haft ett finger med i spelet. Datat delas sedan upp i ett *train-dataset* 75% av datan samt ett *testdataset* 25% för att senare kunna validera resultatet av modellen. kolumnen tiden till positivitet transformeras sedan för att normalisera datan. Den transformerade varianten antas nu komma från en trunkerad normalfördelning med

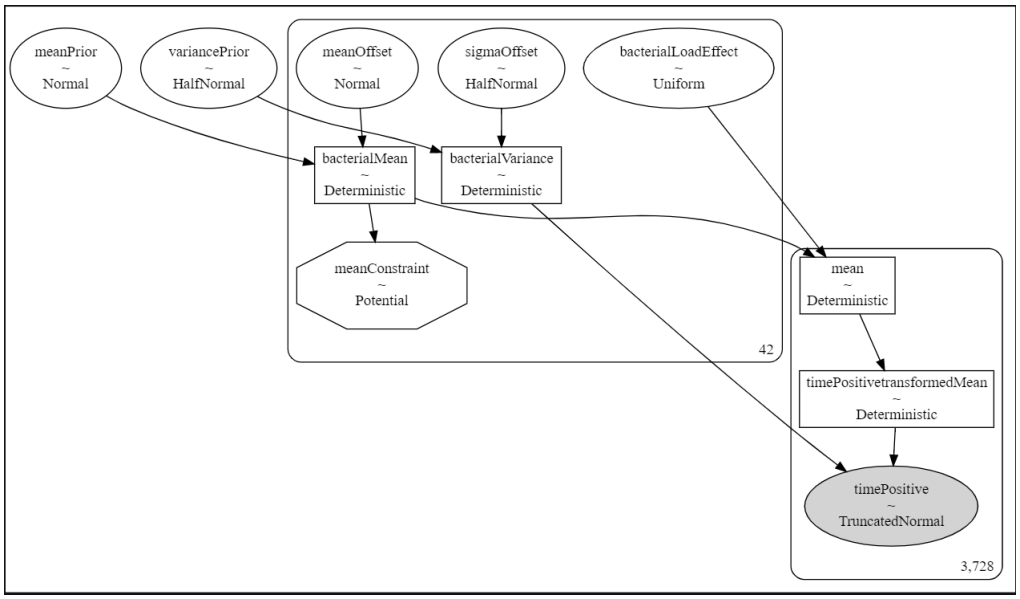
undre och övre gräns transformerad([120,7200])

Hierarkisk modell 1 Medelvärdet i denna trunkerade normalfördelning ges av de olika bakteriernas tillväxtförmåga samt deras förmåga till tillväxt innan optimala förhållanden. Medelvärdet modelleras därför enligt 42 olika regressionslinjer för de olika bakterierna där koefficienten för hur tiden innan påverkar ligger mellan [-0.8,0.2]. Tanken med den hierarkiska modellen är att samtliga provers medelvärden och varians kommer från samma typ av fördelning med lite variation emellan. Vi kan alltså använda oss av data från alla bakterier för att dra slutsatser om hur en specifik bakterie beter sig, den borde inte skilja sig allt för mycket från alla andra. I ekvation 5.3 samt figur 5.18 beskrivs denna struktur vidare.

$$\begin{aligned} \text{TP} &\sim \mathcal{TN}(\text{BoxCox}(\mu, \lambda), \epsilon, \text{BoxCox}([0.12, 7.2], \lambda)) \\ \epsilon &\sim \mathcal{HN}(0, 1) \\ \mu &= \mu_b[\text{B}] + k_b[\text{B}] * \text{TL} \\ k_b &\sim \mathcal{TN}(0, 0.2, [-0.8, 0.2]) \\ \mu_b &\sim \mathcal{N}(\mu_{Prior}, \sigma_{Prior}) \\ \mu_{Prior} &\sim \mathcal{N}(1, 0.5) \\ \sigma_{Prior} &\sim \mathcal{HN}(0.05) \end{aligned} \tag{5.3}$$

Där TP är tiden till positivitet i BBC:n, alltså den observerade variabeln som är trunkerad normalfördelad *(TN)*. Medelvärdet μ för TP ges av regressionslinjerna för de olika bakterierna. Parametrarna för regressionslinjerna ges av de olika bakteriernas medelvärden μ_b tillsammans med koefficienten,k_b för deras förmåga att nyttja tiden innan (TL).

Figur 5.18 visar strukturen av en sådan modell. Där väntevärdet för den totala tiden till positivitet bestäms av de individuella bakteriernas naturliga tillväxtförmåga samt hur väl de kan ta tillvara på tiden innan de laddats in i skåpet.

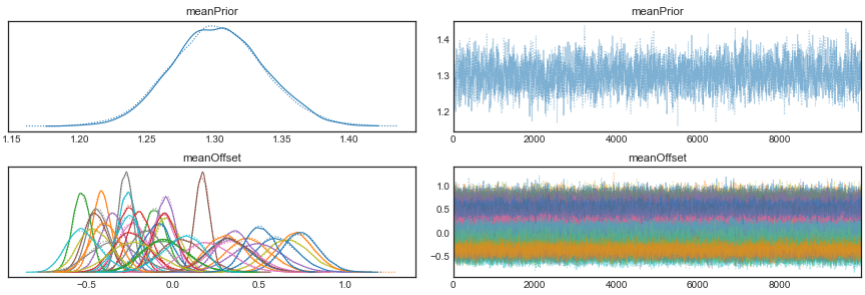


Figur 5.18: Hierarkisk modell 1 (HM:1): Enbart trunkerad

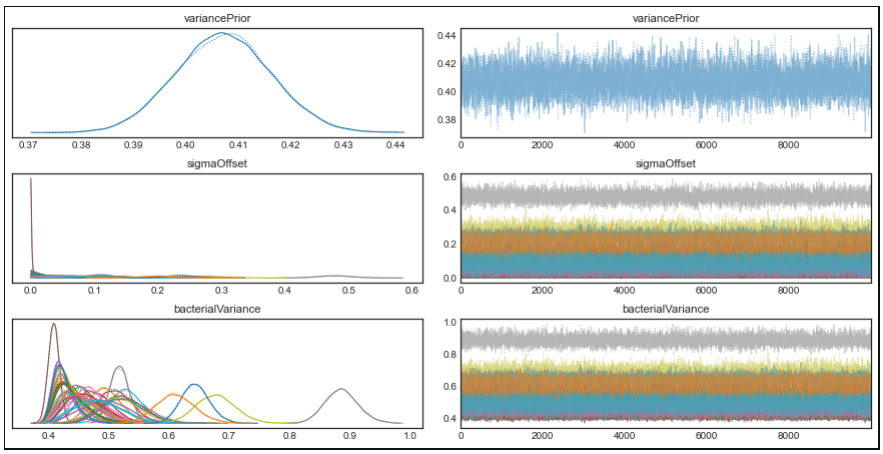
I figur 5.18 syns även en ruta kallad *meanConstraint* med fördelning *Potential* denna modul är ett sätt att försöka styra modellen till att hålla de olika medelvärdena, *bacterialMean*, för bakterierna relativt nära varandra. Tanken är att därmed låta de största skillnaderna i observerat data i större utsträckning beskrivas av vissa bakteriers förmåga till att använda tiden innan inladdnig *bacterialLoadEffect* som gestaltas av regressionslinjen kallad μ i ekvation 5.3. Figuren innehåller även termer som kallas *meanOffset* och *sigmaOffset* dessa härstammar från omparametrisering med syfte att underlätta för (MCMC) - samplern. Så att den i större utsträckning utforskar slumpvariablerna för parametrarna till μ_b. Ekvation 5.3 är alltså aningen förenklad och μ_b ges egentligen av ekvation 5.4

$$\begin{aligned} \mu_b &\sim \mathcal{N}(\mu_{Prior} + O_{\mu_b}, \sigma_{Prior} + O_{\sigma_b}) \\ O_{\mu_b} &\sim \mathcal{N}(0, 0.1) \\ O_{\sigma_b} &\sim \mathcal{HN}(0, 0.1) \end{aligned} \tag{5.4}$$

Resultat Simulering i PYMC3 med denna typ av modell resulterar i en så kallad **trace** där sannolika utfall för de olika stegen beskrivna i ekvation 5.3 samt figur 5.18 sparas. Dessa utfall väljs enligt en sampler. I detta fall används *NUTS-samplern*. När samplingen är klar kan vi plotta resultatet för samtliga delar i hierarkin för att se kring vilka värden dessa parametrar sannolikt borde ligga för att reproducera testdatat. De värden som tracet tillhandahåller kan nu användas för att, med hjälp utav en trunkerad normalfördelning, återskapa den sökta slumpvariabeln *Tiden till positivitet* för de olika bakterierna.



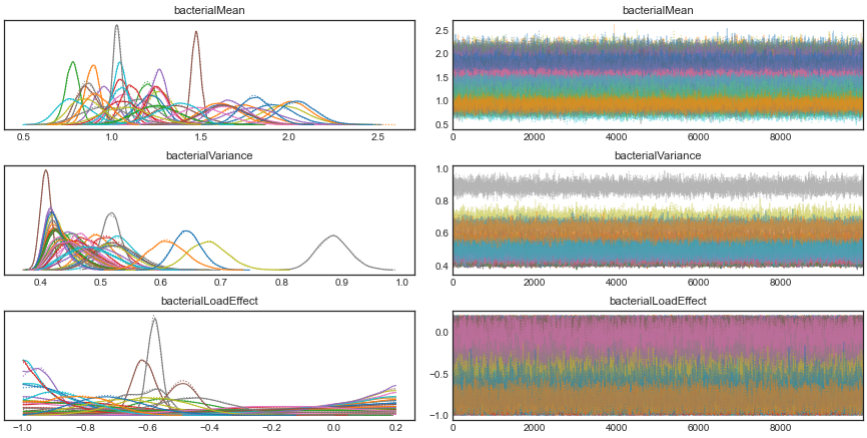
Figur 5.19: Trace plot Mean HM:1



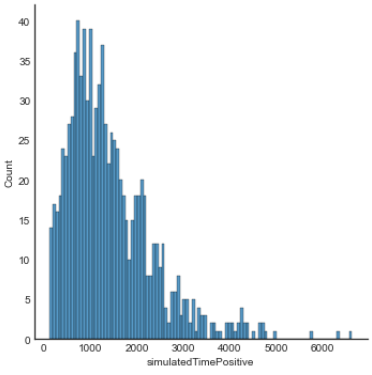
Figur 5.20: Trace plot Variance HM:1

Test/valideringsfunktionen För att testa hur bra resultatet från samplingen är skrivs en testfunktion där medelvärdet från *tracet*, figur 5.21 för *bacterialMean, bacterialVariance* och *bacterialLoadEffect* hos de olika bakterierna används tillsammans med tiden till inladdning samt bakterietyp från testdata. Dessa värden sätts som inparametrar till en trunkerad normalfördelning tillsammans med gränserna [120,7200]. Resultatet av simuleringen presenteras i figur 5.22 . För att få en snabb överblick av resultatet delas även en beskrivning, medeltal, standardavvikelse, minvärde, utvalda kvantiler (25% ,50% (Median) 75%) samt maxvärde för testdata med samma beskrivning för simulerade datapunkterna, se tabell 5.22.

I följande avsnitt presenteras andra alternativa modeller i ett kortare format.



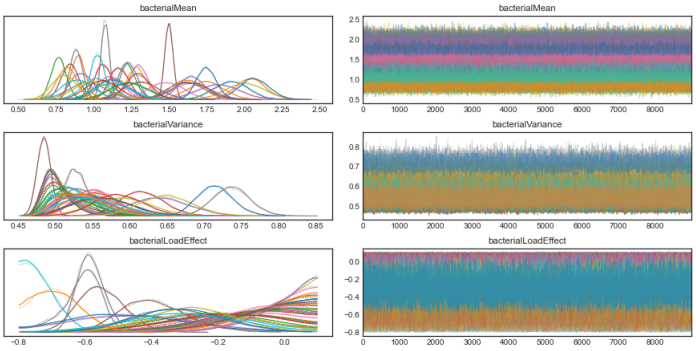
Figur 5.21: Trace plot individuella bakterier HM:1



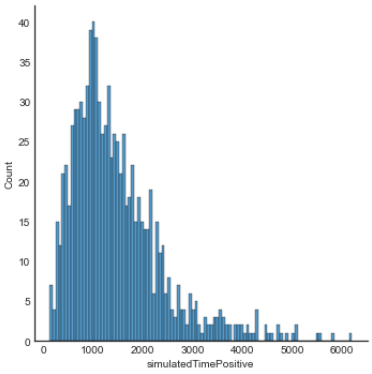
mean	0.968609
std	0.738305
min	1.046644
0.25	1.069653
0.5	1.119685
0.75	1.108925
max	0.870549

Figur 5.22: Simulerad data och jämförelsetabell med testdata HM:1

Hierarkisk modell 2: Censurering och trunkering För att lösa problematiken kring den censurerade delen av data, alltså de punkter som registreras som färdiga efter 120 minuter i blododlingsskåpet används återigen *Potential*-funktionen, med hjälp av denna funktion, som använder sig av den kumulativa fördelningsfunktionen för vår trunkerade normalfördelning kan vi, integrera ut dessa censurerade prover genom likelihood funktionen, alltså den trunkerade normalfördelningen. Detta genomförs genom att multiplicera antalet censurerade prover med den kumulativa fördelningsfunktionen utvärderade i censureringsgränsen.[22] Gränsen 120 minuter motsvarar alltså inte den hårda gränsen för tid till positivitet utan snarare minimumtiden som dessa prover kan observeras på. Så den faktiska tiden dessa prover blivit klara ligger i intervallet [0-120]. För denna modell sätts alltså gränserna för den trunkerade likelihood funktionen till [0.001,7200] med en censurering på datapunkter under 130 minuter (istället för 120 min då det kan finnas viss felmarginal).



Figur 5.23: Trace från HM:2

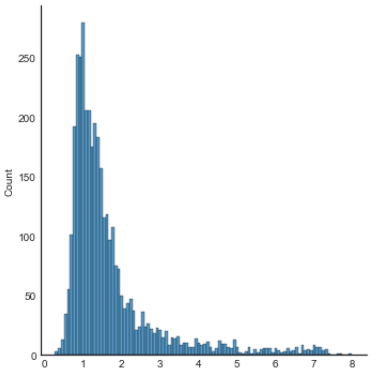


mean	0.968609
std	0.738305
min	1.046644
0.25	1.069653
0.5	1.119685
0.75	1.108925
max	0.870549

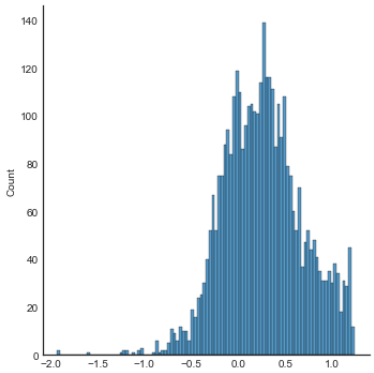
Figur 5.24: Simulerad data och jämförelsetabell med testdata HM:2

Hierarkisk modell 3: Observerad Total tid Eftersom den totala tiden till positivitet (TTP) ges av tiden till in laddning + tiden till positivitet och denna slumpvaraiabel är observerad byggs modellen ut för att ta hänsyn till detta. Den nya modellen tar vid där modell 1 slutar och mu värdet för den trunkerade fördelningen som ger denna slumpvaraiabel är $timePositive + timeLoad$. Ekvation 5.3 utökas nu med

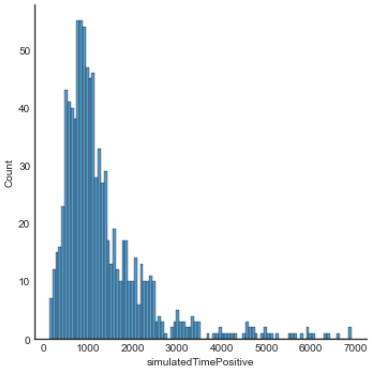
$$\begin{aligned} \mathcal{TTP} &\sim \mathcal{TN}(\text{BoxCox}(\mu_{TTP}, \lambda), \epsilon_{TTP}, \text{BoxCox}([0.12, 7.2], \lambda)) \\ \mu_{TTP} &= \mathcal{TT} + TL \end{aligned} \tag{5.5}$$



Figur 5.25: Total tid till positivitet

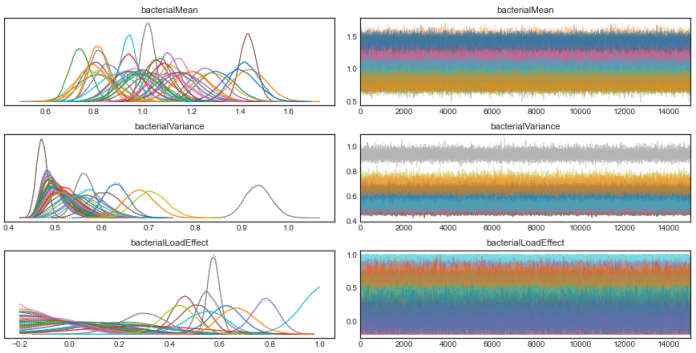


Figur 5.26: Box Cox transformerad TTP



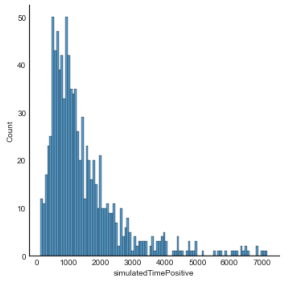
mean	0.928814
std	0.853773
min	1.085279
0.25	0.981752
0.5	0.928818
0.75	1.003112
max	0.963388

Figur 5.27: Simulerad data och jämförelsetabell med testdata HM:3



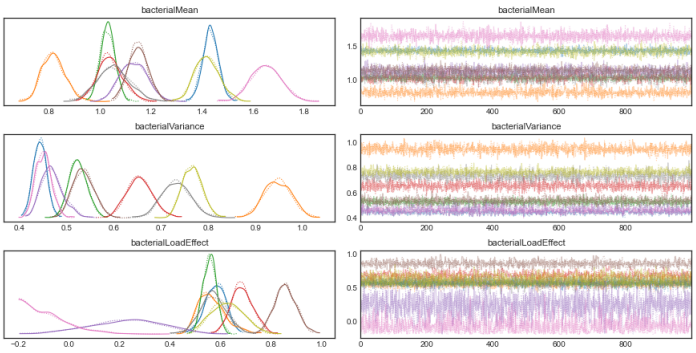
Figur 5.28: Trace från HM:3

Hierarkisk modell 4: 8 Bakteriekategorier För denna variant kategoriseras bakterierna i 8 grupper. Denna gruppering utförs på samma sätt som i en jämförelsestudie mellan två olika BCC:er [8].



mean	0.922562
std	0.868124
min	0.991736
0.25	0.882909
0.5	0.931968
0.75	1.008136
max	0.943237

Figur 5.29: Simulerad data och jämförelsetabell med testdata HM:4



Figur 5.30: Trace från HM:4

Sammanfattat resultat Efter jämförelse av dessa modeller tycks HM:3 prestera bäst, baserat på de olika jämförelsetebellerna och kommer därför att representera detta steg, HM:4 läggs åt sidan då grupperingen verkar vara för grov. Från tabell 5.2 ses att exempelvis bakterier som *Enterococcus species* inte verkar besitta samma tillväxförmåga som *Enterococcus faecalis* och i modell HM:4 antas de vara samma.
Modellen möjliggör nu simulering med olika bakterieprevalens. Eftersom bakteriens identitet ej är känd i detta skede så dras bakterier slumpmässigt utgående från flödets bakterie prevalens. Den regressionslinje som representerar bakterien, se figur 5.1 för exempel, används tillsammans med tiden som passerat innan provet laddats in i BBC,(x) för att räkna ut ett medelvärde till den trunkerade normalfördelningen. Ett slumptal dras från den trunkerade normalfördelningen och resultatet Box-Cox transformeras tillbaka med hjälp av samma λ som i den ursprungliga transformationen av data. Tiden till positivitet ges alltså av $\text{TP} \sim \text{BoxCox}(\mathcal{TN}(Y, \lambda), \sigma_b, [0.12, 7.2]), \lambda)$ där Y ges av $\mu_b + k_b * \text{x}$

Biprodukter och övriga resultat När denna typ av modell skapas ges även en inblick i de olika slumpvariabler som beskriver *tiden till positivitet* ser ut. I tabell 5.1 presenteras medelvärdet av dessa för de 10 vanligast förekommande bakterierna. 5.1

Tolkningen är att bakterier som *Enterococcus faecalis, Escherichia coli, Enterococcus faecium, Staphylococcus aureus* växer bra eller ganska bra utanför inkubatorn. Eller kanske rättare sagt, för dessa kan det finnas skäl att tro på tillväxt innan inkubering. Med bättre förkunskap, priors för de individuella bakterierna kunde sannolikt bättre uppskattningar skapas.

Modellkritik och begränsningar Eftersom den transformerade tiden till positivitet antas vara normalfördelad, vilket kanske egentligen inte är fallet, begränsas eventuellt modellens förmåga att representera extremvärden. Liknande problem uppkommer kanske tydligare i andra ändan av intervallet. Som det beskrivs tidigare i texten handlar det om en censurering vid 120 minuter. Dessutom saknar intervallet 130-180 minuter helt datapunkter, något som kanske får läsaren att höja på ögonbrynen. Detta saknar förklaring och simuleringen

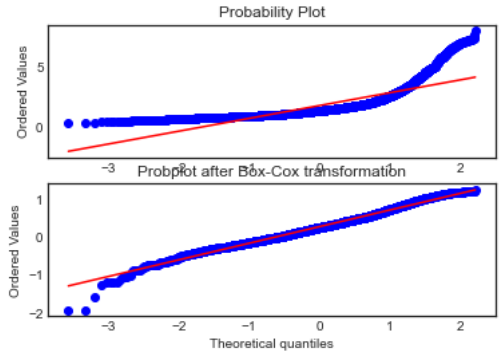
ID	Mean	Variance	LoadEffect	antal
Klebsiella pneumoniae ssp. pneumoniae	817.83	511.96	0.19	113
Pseudomonas aeruginosa	1069.20	505.27	0.20	109
Lactose Fermenting Gram-Negative Rod	945.04	524.46	0.18	110
CoNS	1431.78	471.37	0.49	780
Escherichia coli	826.65	934.21	0.56	352
Enterococcus faecalis	938.78	572.01	0.92	100
Enterococcus faecium	1104.77	498.56	0.78	122
Enterococcus species	1144.37	497.43	-0.02	131
Staphylococcus aureus	1021.73	562.77	0.58	501
other	1417.95	630.28	0.09	343

Tabell 5.1: Medeltal av slumpvariablerna hos de 10 vanligaste bakterierna för HM3

ID	Mean	Variance	LoadEffect	antal
CoNS	1413.75	441.83	0.58	795
Escherichia coli	819.42	923.50	0.53	352
Staphylococcus aureus	1006.24	549.82	0.56	501
Streptococcus	1034.30	617.54	0.66	232
Pseudomonas aeruginosa	1138.27	464.28	0.16	109
Enterococcus	1181.26	527.60	0.87	365
Candida	1626.10	452.61	-0.10	194
Enterobacteriaceae	984.36	690.66	0.62	148
other	1426.12	789.18	0.64	1032

Tabell 5.2: Medeltal av slumpvariablerna för HM4

producerar värden i detta intervall.
Rimligheten hos koefficienten, i tabell 5.1 och tabell 5.2 kallad LoadEffect, för regressionslinjen tål även att diskuteras. Som tidigare nämnts finns det skäl att tro på viss påverkan från vissa bakterier, i studier kring detta har dock denna effekt legat kring [20-40%]. Vilket jämfört med 92% för *Enterococcus faecalis* i denna studie. Nu är dock inte dessa två kanske direkt jämförbara men intuitionen är ändå att modellen kan överskatta effekten av tiden innan, framförallt då denna tid är lång. I figur 5.31 presenteras en plot där kvantilerna för orginaldata och den boxCox transformerade data plottas mot en trunkerad normalfördelning där dessa brister åskådliggörs.



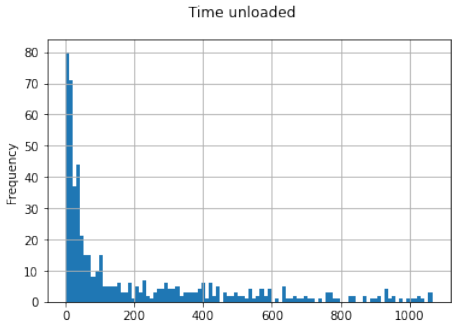
Figur 5.31: QQ-plot data vs boxCox data

I modellavsnittet nämndes även att studien, och därmed modellen, enbart tar hänsyn till positiva tester eftersom att all tillgänglig data gäller prover där bakterier infinner sig. Prover som är negativa eller kontaminerade blir *positiva/klara* först efter 7200 minuter (5 dagar). I en fullständig simulering bör detta tas i beaktande, då dessa prover nyttjar kapacitet i BCC:n eller TCC:n. För att lösa detta implementeras en enkel jämförelse med ett användarvalt procentvärde för negativa prover med en likformig slumpvariabel på intervallet [0,1]. För de prover som inte är positiva sätts bakterietyp till negativ och tidsåtgången för detta

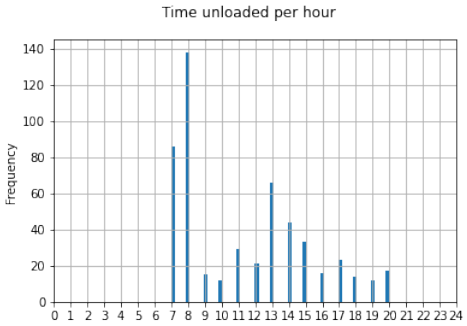
steg till 7200 minuter. Därefter avbryts flödet för dessa. Eftersom den tillgängliga datan enbart tar hänsyn till positiva prover kommer denna funktionalitet inte att användas i simuleringen.

5.4 Steg 4 Provet tas ut ur blododlingsskåpet

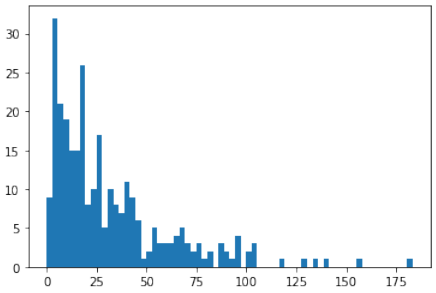
För att uppskatta tiden det det tar för att plocka ut ett prov ur blododlingsskåpet behöver vi, som i så många tidigare steg ta hänsyn till vilka tider denna händelse kan äga rum. Som figur 5.32 visar så finns en hel del extremvärden, majoriteten av dessa antas bero på att blodprovet inkuberats färdigt under natten och sedan väntat tills morgonen då personalen kan bearbeta provet.
Eftersom själva simuleringen i SimPy hanterar öppettiderna för laboratoriet kan tidsförskjutning i detta skede ignoreras. För tillfället sökes enbart ett uttryck för tidsåtgången, givet att personalen är på plats. Vi väljer därmed ut de datapunkterna då ett prov blivit klart under arbetsdagen. I figur 5.34 syns fördelningen för återstående datapunkter.



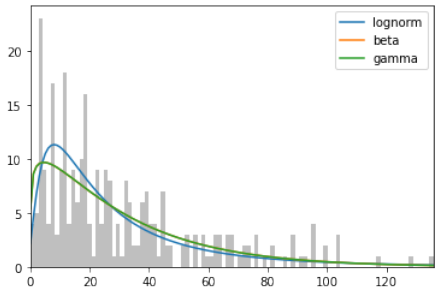
Figur 5.32: Tiden till Urladdning



Figur 5.33: Urladdning under dygnet



Figur 5.34: Tid till urladdning (samma arbetsdag)

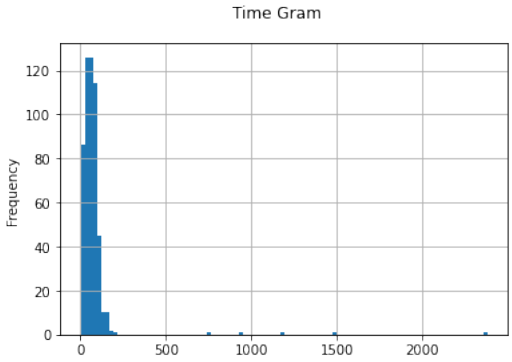


Figur 5.35: Tid till urladdning med fördelningar

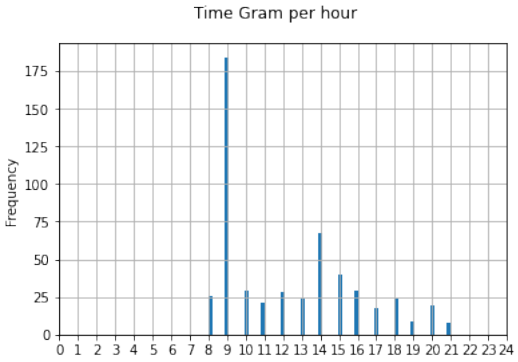
För att finna en slumpvariabel med lämpliga parametrar utförs en ML-Skattning på den formaterade datan. se figur 5.35 ML-skattningen ger gammafördelningen med $\alpha = 1.1688, location = -0.05, scale = 26.9$ eftersom vi aldrig kan ta ut en flaska innan det är möjligt att ta ut flaskan sätter vi $location = 0$ för att undvika negativa värden.

5.5 Steg 5 Provet Gram testas

För att modellera Gram testet används en liknade procedur som föregående steg för urladdning. De datapunkter som gramtestats samma dag som de tagits ut isoleras. Utöver detta vet vi att själva testet i sig kan utföras på ca 5 minuter. Gramfärgning under dygnet antas för simuleringens del till största del bero på när prover är redo att gramfärgas. Alltså när de är klara i blododlingsskåpet samt arbetstider för labbet. Förstudien antyder om att personalen på laboratoriet sannolikt följer olika rutiner för vilka tider på dygnet gramfärgning äger rum, detta är dock svårt att modellera utan mer detaljerad information kring dessa rutiner. Därför antas gramfärgningen utföras, när möjlighet ges, direkt efter urladdning. Efter jämförelse av figur 5.33 och figur 5.37 tycks detta antagande iallafall delvis stämma.



Figur 5.36: Tid till Gram

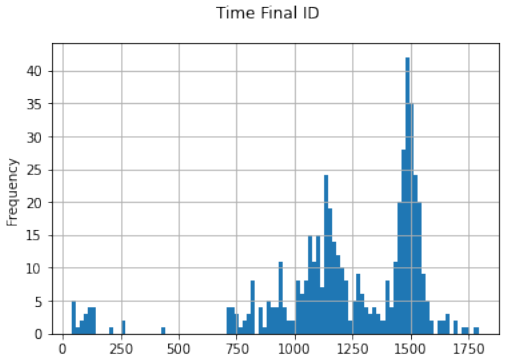


Figur 5.37: Tid till Gram under dygnet

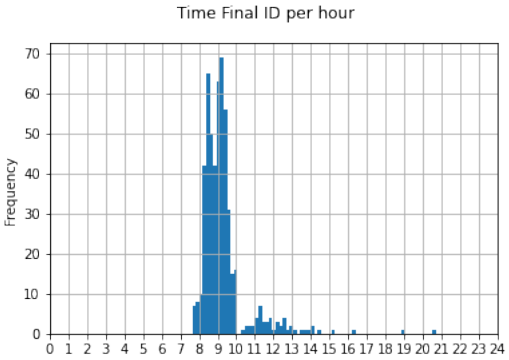
5.6 Steg 6 Id

För att modellera tiden till Id behöver vi ta två olika situationer i beaktande beroende på vilka typer av tekniker som används. Med den äldre tekniken så behöver provet prepareras (bakterierna isoleras) innan ID kan ske, detta steg görs vanligen över natten och tar ca 10 timmar. När morgonen gryr och bakterierna är isolerade tar sedan själva testet bara en par minuter.
Om snabb Id används kan identifiering ske på blodprovet som det är, vi slipper alltså isoleringstiden och testet kan påbörjas direkt efter gramfärgningen är avklarad. Tiden för id med snabbtest varierar lite med olika tekniker utifrån beskrivningen av dataset 1 tar testet 1 timme samt har en maxkapasitet på 8 simultana prover. Resultatet av testet loggas först efter blodkonferensen, detta tillsammans med arbetstiderna antas ligga bakom fördröjningen som syns i figur 5.38.

Eftersom den faktiska tiden tills detta steg är klart inte kan observeras från data så får modellen istället förlita sig på studier och utsagor. För detta används därför en trunkerad normalfördelning så användaren enkelt kan justera gränserna, minimala och maximala tiden för testet samt ge en väntad tid och en varians. För det systemet som dataset 1 kommer från används snabb ID så isoleringssteget hoppas över och hela steget modelleras därmed med en trunkerad normalfördelning med parametrarna $\mu = 60, \sigma = 3, nedre = 70$,övre= 50



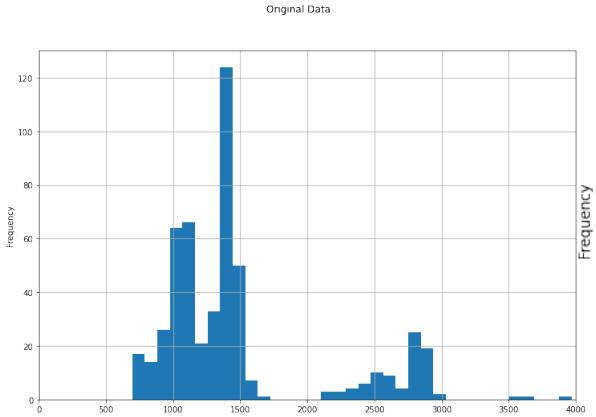
Figur 5.38: Tid till ID



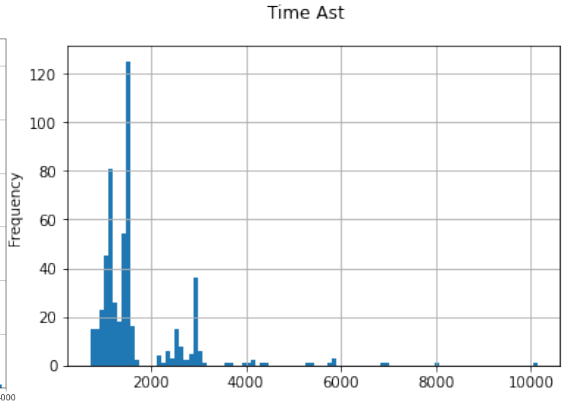
Figur 5.39: Tid till ID under dygnet

5.7 Steg 7 Ast

För att modellera tiden till Ast behöver vi än en gång ta två situationer i beaktande. En snabb variant som kan påbörjas direkt efter gram färgning samt en där vi behöver vänta på resultat från ID. Dessutom, likt ID, loggas inte resultat av detta steg innan blodkonferensen har ägt rum. Därmed finns även här ingen direkt data att tillgå för uppskattning av tidsåtgången. Modellen måste därför även här utgå från studier och utsagor. Den snabba varianten tar 6 timmar och den långsammare kan ta mellan 18-24 timmar. I fallet från dataset 1 används den långsammare varianten och resultat från ID inväntas innan AST utförs.
Liksom för ID används en trunkerad normalfördelning för att beskriva denna tidsåtgång, när modellen ska efterlikna det system dataset 1 beskriver används $\mu = 18 * 60, \sigma = 60, nedre = 17 * 60$,övre= $19 * 60$



Figur 5.40: Tid till AST



Figur 5.41: Tid till AST under dygnet

5.8 Total Tid

Den totala tiden bestäms då blodkonferensen har varit och resultatet av både ID och AST loggats. Detta steg modelleras genom två funktioner, en som genererar en vektor med alla möjliga konferenser under hela simuleringstiden med hjälp av inputparametrarna tidpunkterna på dygnet, varians och antal konferenser.
Följande funktion tar tidpunkten då ett test är slutfört och kollar när närmaste efterföljande konferens äger rum, då kommer resultatet av testerna diskuteras och loggas. När både ID och AST är loggade är flödet för det provet avslutat.
Eftersom denna del av simuleringen sker utanför SimPy simuleringsmiljön kan den därför justeras om man är osäker på vilka tider som gäller. Steget kan också bortses från i sin helhet ifall det som skall undersökas enbart är tiden tills ett test faktiskt är färdigt, dock motsvarar dessa tider enbart när själva testet är klart och inte när det potentiellt kan loggas som färdigt på grund av begränsade arbetstider. Därför presenteras resultatet av simuleringen för detta steg, samt steg ID och AST bland de övergripande resultaten i avsnitt 7.

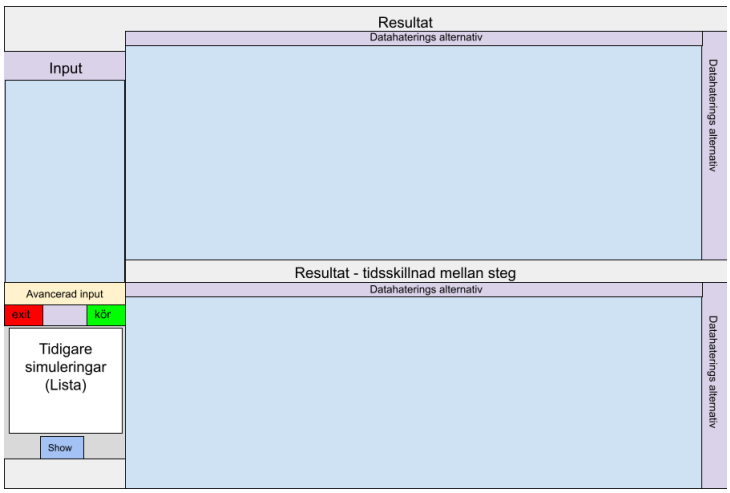
Kapitel 6

Gränssnitt och övergripande programdesign

Programmet som utvecklades delas in i 3 olika block, *Gränssnitt*, *Flödessimulering* samt *Hjälpfunktioner* för hantering av data och slumpvariabler. I detta kapitel kommer övergripande designval diskuteras och kortfattade kodexempel presenteras.

6.1 Gränssnitt

För gränssnittet valdes en simplistisk design mycket lik andra simulerings och databehandlingsprogram för att användare enkelt skulle känna igen sig. Skissen på denna design ses i figur 6.1 Tanken är att användaren



Figur 6.1: Skiss på gränssnittet

enkelt skall kunna simulera med olika varianter, vilka sedan sparas i listan *tidigare simuleringar*. Här kan sedan flera av dessa väljas och med knappen *show* presenteras i resultatfönstret. I fältet *datahanterings alternativ* kan användaren sedan spara resultatet i (*.csv*) format och fortsätta sin tolkning av resultaten i valfritt databehandlningsprogram. Gränssnittet är uppbyggt i Tkinter och koden är strukturerad enligt MVC.

6.1.1 Model

Modellen hanterar all data som programmet använder. Här sparas gamla körningar samt standardparametrar för simuleringen.

6.1.2 View

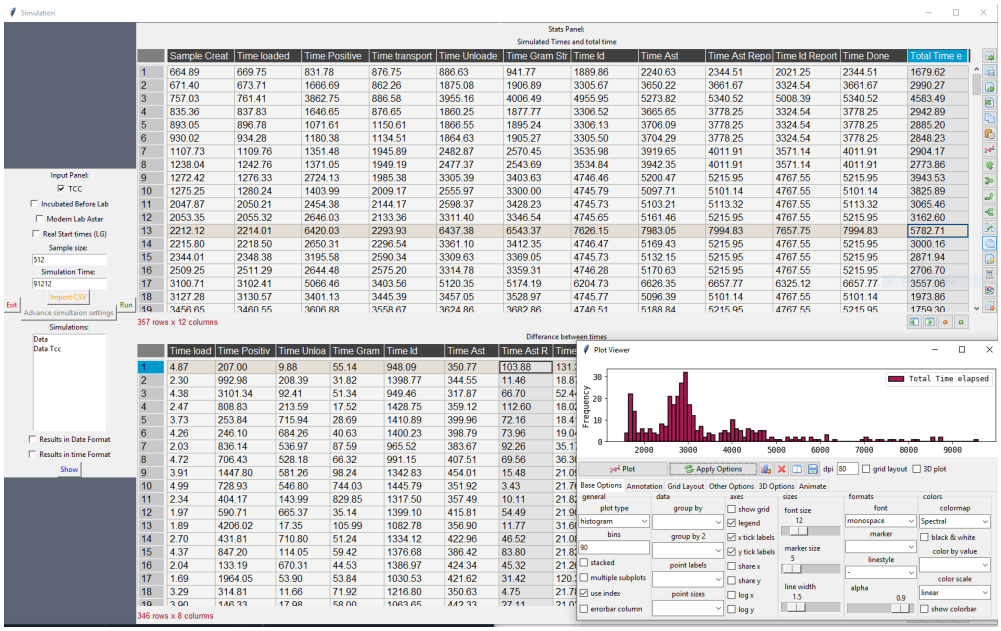
Viewn består av två paneler, *input panelen* samt *Data panelen*. I **inputpanelen** demonstreras de alternativ användaren har, alltså möjlighet att ändra på inputparametrar till simuleringen, starta simuleringen, ladda in

egen data för att simulera från egna starttider (När prov skapas) samt avsluta programmet. Inom denna panel finns även en lista över tidigare simuleringar så användaren kan jämföra dessa. **Data panelen** demonstrerar resultatet av körningen uppdelat i två paneler. Det översta med tiderna i minuter för varje steg för samtliga prover med sista spalt *total Tid*. I den undre panelen syns skillnaderna i tid mellan två efterföljande steg. Dessa paneler är skapta med pythonbiblioteket *pandastable* vilket bland annat möjliggör smidiga plottar samt exportering av resultaten. I figur 6.2 syns en särmdump av programmet med dessa paneler samt nere i högra hörnet ett exempel på *Pandastables* inbyggda plot funktionalitet.

6.1.3 Controller

Controllern sköter all logik inom gränssnittet. Den kallar på flödessimuleringen när användaren klickar på *run* med de önskade inputparametrarna. När sedan resultatet av simuleringen är klart uppdaterars Veiwn och de nya resultaten presenteras. Samtidigt sparas dessa i Modellen för att möjliggöra jämförelse mellan olika simuleringar.

I figur 6.2 demonstreras programmet efter två simuleringar körts, en med TCC aktiverad samt en utan.



Figur 6.2: Skärmdump av gränssnittet

6.2 Flödessimulering

Inom detta block körs koden som ansvarar för simuleringen. Huvudfunktionen *runSimulation* tar som input de önskade parametrarna för den aktuella simuleringen och startar en Simpy *enviroment*. Inom miljön körs sedan en *setupfunktion* som initierar en instans av flödet. Flödet sätter upp begränsningarna för denna simulering enligt inputparametrarna. *setupfunktionen* ansvarar även för generationen av prover, antingen från en lista som användaren tillhandahåller eller enligt sektion 5.1 Provet skapas. Dessa nya prover går sedan igenom flödet beskrivet i figur 5.3 baserat på den input kring tekniker simuleringen skall använda.

I kodexemplet (Listing 6.1) syns ett urklipp från flödet där två olika alternativa rutter kring logiken för inkubering och transport defineras. Om ett prov simulerats enligt ett normalt flöde så skall provet först transporteras. Här skickas ett *request* till transport *resursen* som avgör när nästa transport äger rum. Provet, tillsammans med alla andra prover som genererats innan transport, väntar sedan tills transport äger rum. När provet sedan anländer till laboratoriet väntar provet på att laboratoriet skall öppna, om så är fallet, för att sedan laddas in i BBC:n. Provet inkuberas sedan och flödet fortsätter.

Om flödet simulerats med *inkubation innan labb* sätts inladdning för provet direkt efter provtagning. Sedan väntar provet på transport, så prover skickas iväg till laboratoriet vid nästa möjliga transporttillfälle.

För varje steg den tar sig igenom loggas tiden och sparas i en Python *Dictionary*. När alla prover genomgått alla stegen eller när maxtiden för simuleringen är passerad avbryts huvudfunktionen och så skickar den tillbaka den genererade datan till controllern i gränssnittet.

```
                ...
# TCC ENABLED
elif(simulationParams['tccEnabled']):
    #LOAD SAMPLE
    with flow.sampleLoadResource.request() as request:
        yield request
        yield env.process(flow.newLoadSample(sampleId))
    #TRANSPORT SAMPLE
    with flow.sampleTransportResource.request() as request:
        yield request
        # wait if of out of hours
        yield env.timeout(timeUntilOfficeHours(
                    simulationParams['OfficeHours'],env.now))
        yield env.process(flow.transportSample(sampleId))
    #INQUBATE SAMPLE
    with flow.sampleInqubateResource.request() as request:
        yield request
        yield env.process(flow.inqubateSample(sampleId))

#NORMAL FLOW
else:
    #TRANSPORT SAMPLE
    with flow.sampleTransportResource.request() as request:
        yield request
        # wait if of out of hours
        yield env.timeout(timeUntilOfficeHours(
                    simulationParams['OfficeHours'],env.now))
        yield env.process(flow.transportSample(sampleId))
    #LOAD SAMPLE
    with flow.sampleLoadResource.request() as request:
        yield request
        # wait if of out of hours
        yield env.timeout(timeUntilOfficeHours(
                    simulationParams['OfficeHours'],env.now))
        yield env.process(flow.newLoadSample(sampleId))
    #INQUBATE SAMPLE
    with flow.sampleInqubateResource.request() as request:
        yield request
        yield env.process(flow.inqubateSample(sampleId))
            ...
```

Listing 6.1: Kodexempel från simulering i simpy, jämförelse mellan två alternativa flöden

Kapitel 7

Analys och resultat

I detta kapitel kommer resultat från en rad olika simuleringar presenteras. Simuleringarna skapas genom olika variationer på parametrarna för :

```
{
    'incubatedBeforeLab' : False,
    'tccEnabled' : False,
    'bccBeforeLab': False,
    'rapidAst': False
    }
```

Listing 7.1: Variation av dessa parametrar genererar olika flöden

Dessa leder alltså om flödet i de alternativa rutterna som presenterades i figur 5.3.

Resterande input är sedan gemensamt för samtliga flöden, och följer beskrivningen som getts i avsnitt 5. Samtliga parametrar för det så kallade basfallet, som beskrivs i större detalj i nästa avsnitt, hittas i appendix.

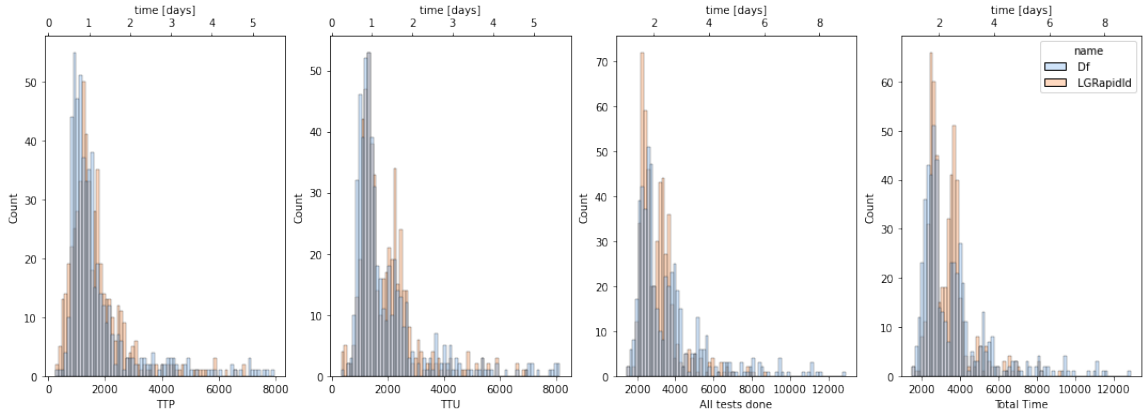
För varje simulering kommer 2 grafer och en tabell presenteras. I den första figuren kommer simuleringen undersökt i 3 händelser beskrivas. De undersökta händelserna är **Total tid till positivitet (TTP)**, **Total tid till urladdning (TTU)** som beräknas som tidsåtgången från att tagning tills positivitet respektive att provet plockats ut ur blododlingskåpet på laboratoriet. Nästa händelse är tiden tills samtliga tester är färdiga, i figurerna kallad **All tests done** och slutligen **Total Time** som representerar tiden då resultatet bokförts, alltså efter blodkonferensen ägt rum. Denna förenkling görs för att underlätta i analysen, vägen till dessa steg varierar mellan olika flöden men alla kommer hit och därför lämpar sig dessa tidpunkter väl för att jämföra prestandan för de olika flödena.

Den tabell som hör till varje flöde beräknas som en jämförelse med basfallet. För det allmänna flödet är detta originaldata *från dataset 1*. Denna jämförelse utförs, på samma sätt som tabellerna i avsnittet om *tiden till positivitet* genom att dela olika beskrivningar, som medeltid, standardavvikelse, kvantiler, av originaldata, eller basfallet med de olika flödesvarianterna. Från detta kan en känsla kring den relativa snabbhet olika flödesvarienter har till dessa steg. Utöver detta utförs även ett *Mann–Whitney U*, test för att se hur de bakomliggande slumpvariablerna förhåller sig till varandra, p-värdet och slutsatsen, i tabellen kallad *MWUs*. I basfallet, som följer närmast, jämförs om dessa bakomliggande slumpvariabler är lika stora. I övriga fall undersöks om den nya varianten är snabbare än basfallet. I tabellen presenteras dessa som $=, \neq, <, >$ om testet antyder om samma, olika och större bakomliggande storlek på bakomliggande slumpvariabler.

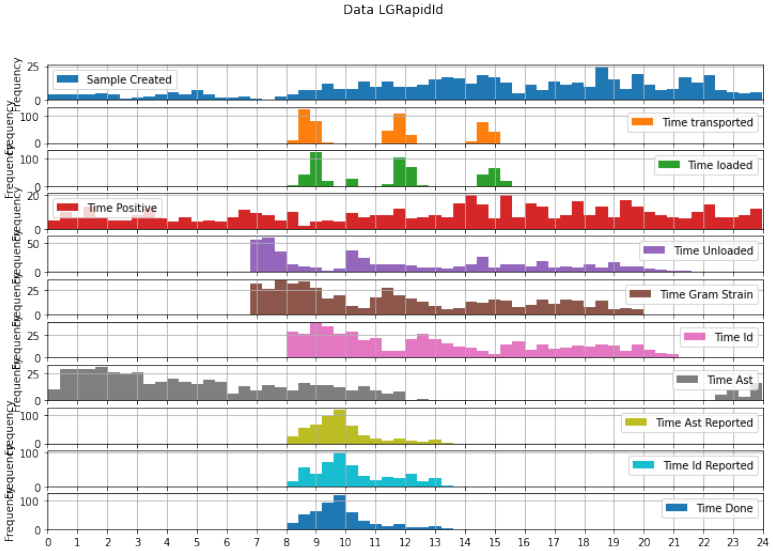
7.1 Det allmänna flödet, Basfallet

Basfallet är genererat med parametrarna som hittades i avsnitt 5

I tabell 7.1 är alltså metadata från originaldata delat på metadata från simuleringen. Optimala värden ligger alltså runt 1. Simuleringen tycks efterlikna originaldata i många aspekter men förutom för maximala värdet *max* och *75%* kvantilen. Detta tyder på att simuleringen har vissa problem att återskapa extremvärden. Medelvärdet i raden **All tests done** visar 0.86 detta kan intuitivt tolkas som tidsförlusten av att inte resultaten rapporteras in direkt då testen är klara utanför arbetstid.
Mann–Whitney U testet, med hypotesen *H0* att dessa dataset kommer från samma (*lika stora*) fördelning håller i avseendet **TTP** samt **Total time**. För **TTU** skiljer sig simuleringen och originaldata och *H0* förkastas. Tolkningen här är att modellen har sina brister i att exakt modellera samtliga steg. Dock uppvisar simuleringen *tillräckligt* bra resultat för **Totaltiden** och **TTP** och denna simulering blir nu basfall, *referensfall* för resterande simuleringar.



Figur 7.1: Normalt flöde Tid mellan steg



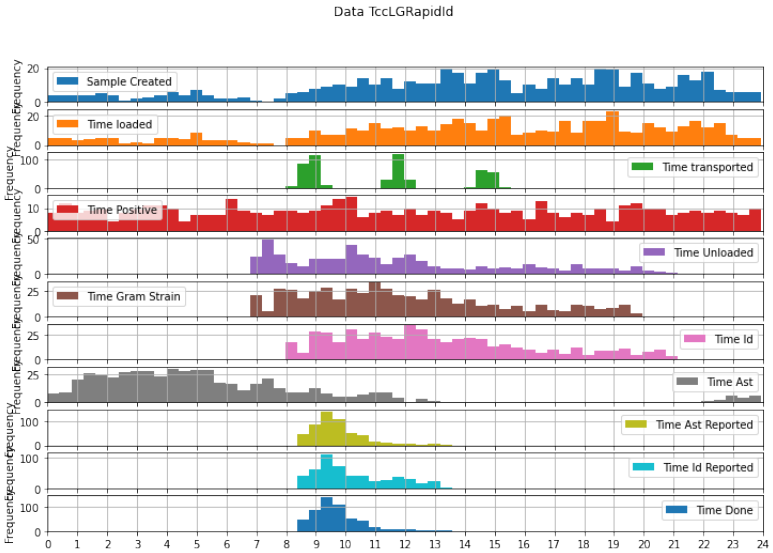
Figur 7.2: Normalt flöde per timme

	Total Time (min)	TTP	TTU	All tests done	Total Time
mean	3362	0.98	1.00	0.86	0.93
std	1065	0.80	0.78	0.56	0.58
min	1610	1.18	1.04	0.97	1.08
25%	2595	1.08	1.08	0.97	1.06
50%	3219	1.08	1.10	1.02	1.11
75%	3760	1.08	1.04	0.87	0.93
max	9168	0.87	0.85	0.67	0.71
p-värde		0.067	0.003	0.001	0.468
MWUs		=	≠	≠	=

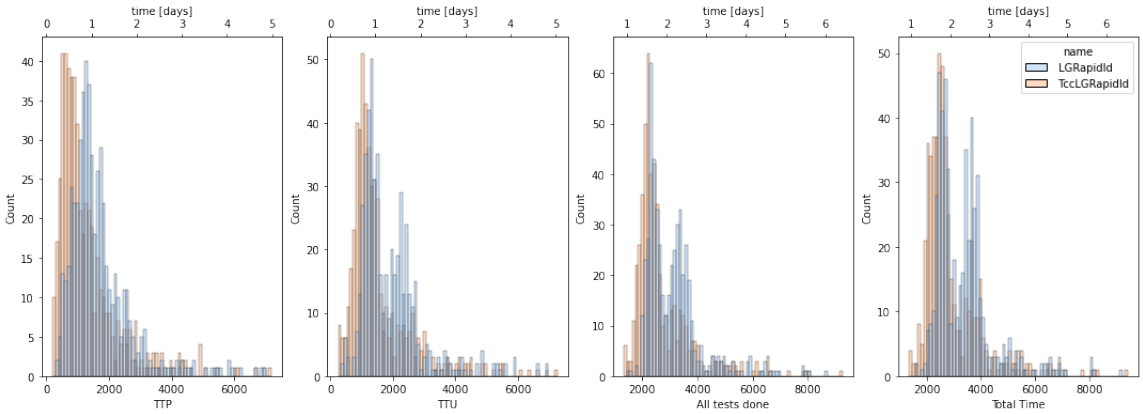
Tabell 7.1: Orginaldata jämfört med Basfallet

7.1.1 TCC

Med denna teknik kan alltså inkubation påbörjas direkt efter provtagning, och inkubationen fortsätter under transporten. I detta flöde läggs heller ingen minimum tid i BCC när blodprovet anlänt till laboratioriet. Detta kan ses i tabell 7.2 de 20% kortare tiderna för **TTC** och runt 10% kortare tider för **All tests done**. I figur 7.4 **TTP** går det även att urskilja en antydan om en högre andel prover har blivit positiva innan 1000 minuter har passerat. Denna effekt tycks förminskas i **TTU** antagligen då dessa oavsett måste invänta arbetstider för att vidare processeras.



Figur 7.3: TCC, flöde per timme



Figur 7.4: TCC, Tid mellan steg

	Total Time	TTP	TTU	All tests done	Total Time
mean	2965	0.79	0.81	0.88	0.88
std	1139	1.05	1.04	1.06	1.07
min	1347	0.64	0.70	0.91	0.84
25%	2265	0.60	0.76	0.88	0.87
50%	2584	0.69	0.78	0.81	0.80
75%	3447	0.82	0.78	0.89	0.92
max	9437	1.04	1.04	1.07	1.03
p-värde		$1.4 * 10^{-20}$	$2 * 10^{-20}$	$3 * 10^{-21}$	$1.9 * 10^{-19}$
MWUs		<	<	<	<

Tabell 7.2: Basfall jämfört med TCC

	Total Time	TTP	TTU	All tests done	Total Time
mean	3264	0.83	1.03	0.99	0.97
std	1122	1.04	1.08	1.08	1.05
min	1555	0.62	1.13	1.00	0.97
25%	2490	0.69	0.96	0.98	0.96
50%	2963	0.77	1.04	0.94	0.92
75%	3639	0.83	1.02	0.99	0.97
max	9416	1.02	1.17	1.06	1.03
p-värde		$9 * 10^{-15}$	0.55	0.08	0.001
MWUs		$<$	$\geq$	$\geq$	$<$

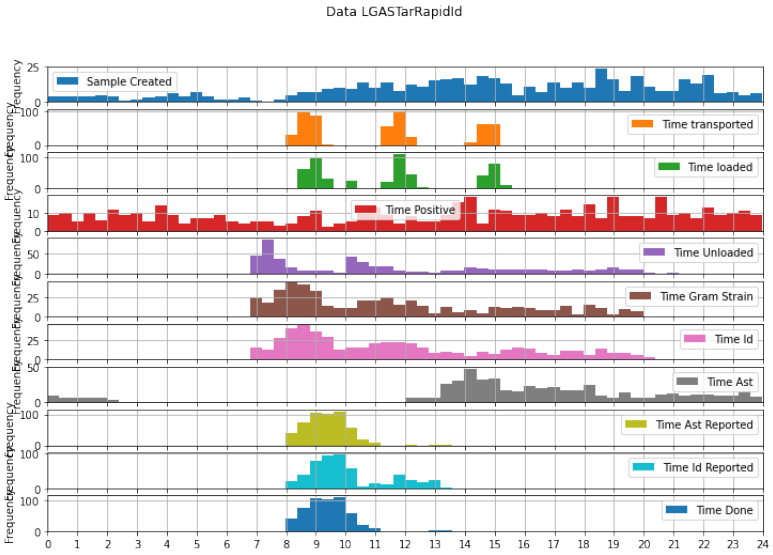
Tabell 7.3: Basfall jämfört med BCC innan laboratoriet

7.1.2 BCC på plats

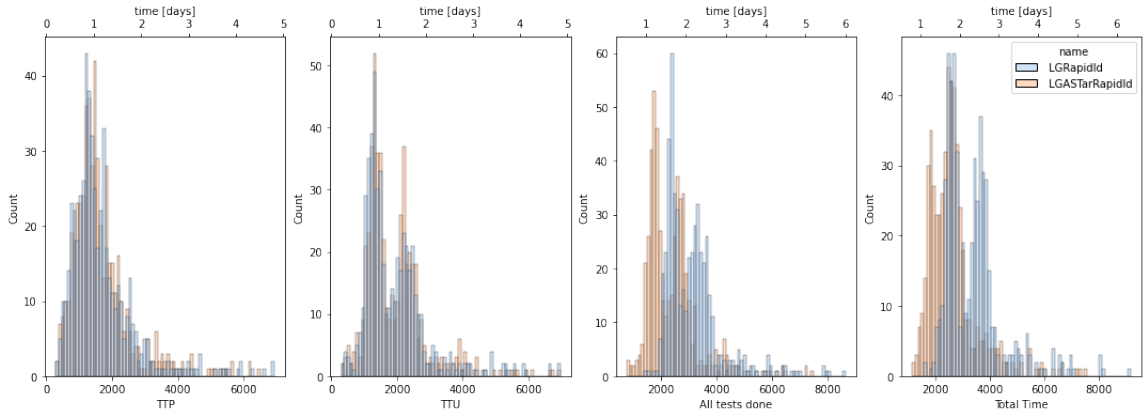
I detta flöde odlas provet innan det skickas till laboratoriet. I tabell 7.3 kan vi se att **TTP** är kortare med ca 20% och *Mann–Whitney U* testet antyder om en kortare tidsåtgång än basfallet. För resterande mätpunkter tycks tabellvärdena samt resultatet av *Mann–Whitney U* testet att detta flöde i stort producerar liknande tider som basfallet. Detta är kanske inte helt oväntat då flödet genomgår samma steg. De aningen högre värdena kan tänkas bero förlorad tid kring transporter för prover som annars vore redo för vidare tester. Som beskrivet i avsnitt 5.3 tar modellen även hänsyn till att vissa bakterier kan växa i rumstemperatur så även om det kan vara optimalt att påbörja odlingen snabbt efter provtagning så förminskas denna fördel och blir snarare en nackdel på grund av rutinerna kring transport och öppettider för laboratoriet.
Här önskas dock läsaren påminnas om beskrivningen av denna typ av flöde från avsnitt 2.1. Fördelarna med detta flöde gestaltas dåligt av simuleringen, då dessa kan innefatta en lättare last på laboratoriet och eventuellt snabbare information för vårdhavande läkare om positivt blodprov.

7.2 Parallell Snabb AST (PSA)

I detta avsnitt presenteras resultat av simuleringar med det snabbare alternativet för AST-steget, PSA. I dessa flöden kan även stegen för ID och AST ske parallellt. Likt tidigare steg jämförs dessa med basfallet. I figur 7.6 kan vi notera att fördelningarna för **TTP** samt **TTU** ser snarlika ut, detta kan även noteras i tabell 7.4. Resultatet är helt väntat, och till och med önskvärt, då simuleringen fram till denna punkt är likadan som i basfallet. Här kan vi alltså istället få en inblick i variationen i utfall för två iterationer av samma simulering. För efterföljande steg kan vi se en markant förbättring av **All tests done** och även **Total tid**. Från tabellen kan denna uppskattas till 20%. MWU testet antyder även att dessa slumpvariabler är mindre (snabbare) än basfallet.



Figur 7.5: PSA + normalt flöde, per timme



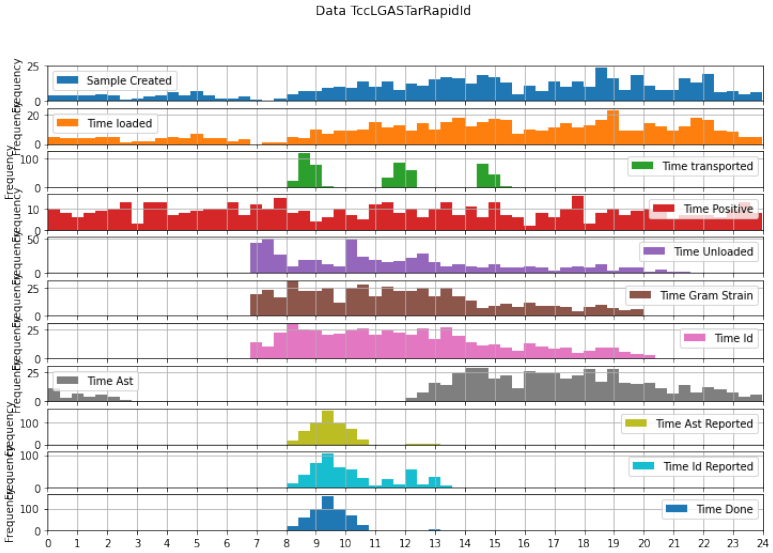
Figur 7.6: PSA + normalt flöde, tid mellan steg

7.2.1 TCC

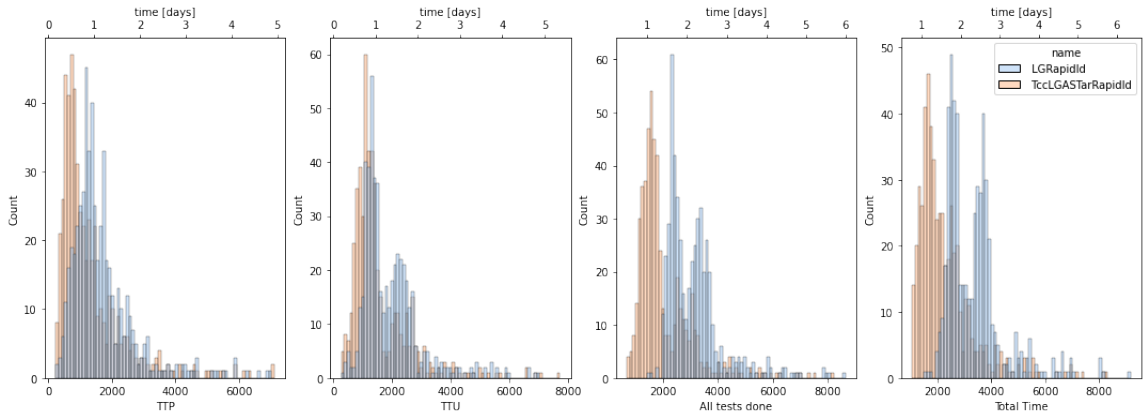
I detta flödesexempel används TCC tillsammans med PSA. Detta resulterar i märkbart kortare **Total Time** vilket kan ses i tabell 7.5 där 75% av proverna är klara inom 2625 minuter, jämfört med basfallets 3760 minuter.

	Total Time	TTP	TTU	All tests done	Total Time
mean	2645	1.00	1.00	0.77	0.79
std	950	0.93	0.94	0.93	0.89
min	1141	0.93	0.84	0.52	0.71
25%	2028	1.04	1.02	0.73	0.78
50%	2507	1.03	1.03	0.76	0.78
75%	2869	1.01	1.00	0.79	0.76
max	7505	0.93	1.00	0.85	0.82
p-värde		0.73	0.75	$7 * 10^{-41}$	$1 * 10^{-41}$
MWUs		$\geq$	$\geq$	$<$	$<$

Tabell 7.4: Basfall jämfört med PSA



Figur 7.7: PSA + TCC,per timme



Figur 7.8: PSA + TCC, tid mellan steg

	Total Time	TTP	TTU	All tests done	Total Time
mean	2313	0.80	0.83	0.66	0.69
std	1078	1.04	1.03	1.03	1.01
min	1076	0.78	0.74	0.47	0.67
25%	1624	0.63	0.76	0.60	0.63
50%	2014	0.69	0.79	0.57	0.63
75%	2625	0.82	0.86	0.71	0.70
max	8259	1.03	1.11	0.94	0.90
p-värde		$1 * 10^{-19}$	$2 * 10^{-17}$	$3 * 10^{-77}$	$1.9 * 10^{-72}$
MWUs		<	<	<	<

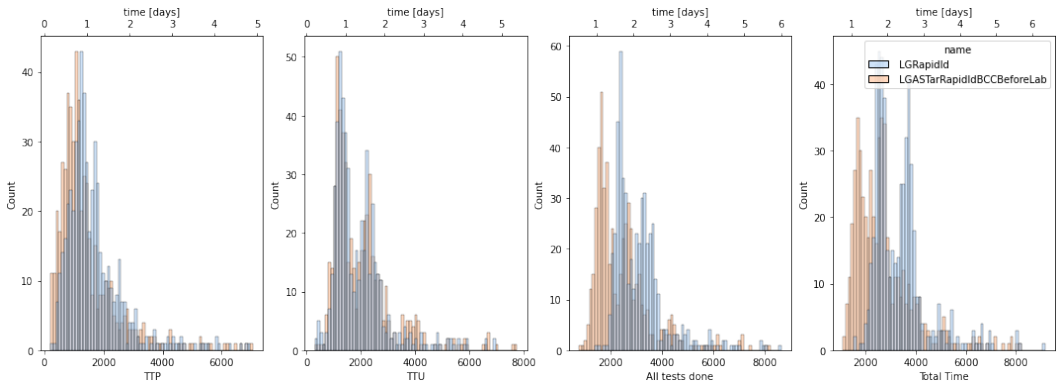
Tabell 7.5: Basfall jämfört med PSA och TCC

	Total Time	TTP	TTU	All tests done	Total Time
mean	2731	0.86	1.05	0.79	0.81
std	1210	1.06	1.12	1.12	1.14
min	1092	0.70	0.83	0.52	0.68
25%	1854	0.74	0.97	0.69	0.71
50%	2512	0.80	1.06	0.71	0.78
75%	3167	0.88	1.05	0.80	0.84
max	8196	1.02	1.12	0.95	0.89
p-värde		$7 * 10^{-38}$	0.77	$7 * 10^{-38}$	$5 * 10^{-31}$
MWUs		<	=	<	<

Tabell 7.6: Basfall jämfört med PSA och blododling innan laboratoriet

7.2.2 BCC på plats

Som fallet med BCC utan PSA producerar detta flöde en fördelning för **TTU** likt den för basfallet. Detta leder till att liknande, om inte lite långsammare resultat, jämfört de med enbart PSA. I figur 7.9 kan vi antyda att det är just tiden till urladdning, med bakgrundsfaktorer som transport och öppettider, som är ansvarig för flaskhalsen.



Figur 7.9: PSA + BCC innan labb, tid mellan steg

7.3 Övrig undersökning

I detta avsnitt kommer kortfattade resultat från olika scenarion, tester gjorda med förändringar av andra parametrar, presenteras. Detta för att få en inblick i styrkor och svagheter hos de olika flöden samt en känsla för olika rutiners påverkan på de olika milstolparna.

7.3.1 Längre transporter

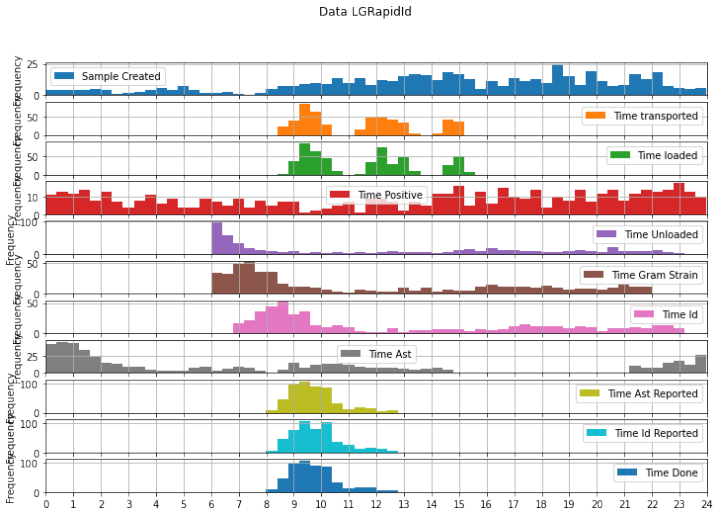
För att vidare undersöka de olika flödena testas även att förändra rutinerna kring transport. Istället för tre transporter under vardagar och en under helgen sätts istället en transport klockan 11.30 alla dagar. Dessutom förlängs medeltiden för transport från 45 till 90 minuter. För basfallet innebär detta en 7% ökning i medel för **TTP** och **TTU** och en 5% ökning i **total tid**.
I detta scenario uppvisar *TCC* en 26% kortare **TTP** och 14% kortare **Total tid** jämfört med det nya basfallet. Om både **TCC** och **PSA** används fås 3078 minuter i medel för **Total tid** Det är alltså tydligt, och kanske inte så konstigt att, förhållandena kring transport har påverkan på de olika markörerna. Redan i denna korta analys kan tendenser ses att olika flödesrutiner kan minska effekten av otillräcklig, bristande transport.

7.3.2 Weekend effect

I detta scenario försöker vi analysera hur stor påverkan blir av kortare arbetspass och färre transporter under helgerna. Nu sätts de ordinarie arbetstiderna tillbaka till basfallets [07.00 - 20.00] och för helgerna [9.00-15.00] dessutom minskas antalet transporter från 3 transporter [08.00,11.00,14.00] till 1 per dag klockan 11.00. För denna simulering ses en 8% förbättring i **TTU** och 6% i **All tests done**. Även övriga benchmarks antyder också om en förbättring dock visar *MWU* att dessa kan vara lika.

7.3.3 Utökade arbetstider

För att undersöka arbetstidernas påverkan på flödet sätts samtliga parametrar till de från basfallet förutom att arbetstiderna utökas från [07.00 - 20.00] till [06.00 - 22.00]. I jämförelse med basfallet resulterar detta i 5% kortare **TTU** och **All tests done**. **Total time** är dock oförändrad då blodkonferensen fortfarande följer samma rutiner som tidigare. I figur 7.10 kan tiden på dygnet dessa steg äger rum.



Figur 7.10: Ändrade öppettider

Kapitel 8

Diskussion och slutsats

Som basfallet i resultatdelen demonstrerade producerar modellen närliggande resultat till verklig data. Resultaten varierar dock beroende på vilket steg som undersöks.
I analysen av alternativa flöden presenterar sig fördelarna som väntat, åtgärder som i sin natur förkortar steg i flödet för med sig förkortningar totalt sätt. Dock blir det tydligt i avsnittet *Övrig undersökning* att dessa, *och för all del det ursprungliga flödet*, begränsas av rutiner och arbetstider. Resultatet av testet kan inte bearbetas utan personal som hanterar provet, att provet nått den punkten tidigare eller senare spelar i det fallet inte så står roll. Så för att på bästa sätt utnyttja snabbare tester krävs snabbare rutiner. Andra sidan visar flöden med **TCC** egenskaper som förminskar effekten var dåliga eller undermåttliga rutiner.

8.1 Framtida studier/ utveckling

För att i större utsträckning anpassa och för all del validera modellen hade data tillsammans med flödesbeskrivningar från andra flöden varit nödvändiga. Tyvärr fanns ej det tillgängligt under arbetets gång. För vidare studier och utveckling bör alltså denna modell till en början utvärderas mot andra flöden och jämföras med klinisk data från dessa.

8.1.1 Utvidgning av modellen

När mer data och flödesbeskrivnigar blir tillgängliga behövs sannolikt vissa uppdateringar och utökningar av modellen läggas till. I detta avsnitt diskuteras en par av dessa.

Transport Modellen har vissa brister då samtliga transporter antas komma från samma sjukhus. I de flesta fall betjänar ett laboratorium flertalet sjukhus och dessa har i sin tur olika rutiner kring transporter. För att analysera ett sådant system behöver alltså modellen och mer specifikt rutinerna kring transport utökas. Utvidgade rutiner kring transport kan även komma att användas för produktspecifikationer för potentiella tekniker som TCC:n. Exempelvis kan minimiantalet enheter ett system behöver samt nödvändig batteritid i enheten analyseras.

Kapacitet Med ny data där även negativa prover är inkluderade kan kapacitetsaspekter av systemet i större utsträckning analyseras. Nuvarande simulering har stöd för funktionalitet med negativa prover. Men då datan modellen utgår ifrån enbart innehåller positiva prover som egentligen är en klar minoritet av samtliga prover som analyseras blir bilden skev. Framförallt om kapaciteten i systemet skall analyseras. Detta går dock med relativ enkelhet att åtgärda. Med mer detaljerade beskrivningar av personalen på laboratoriet och deras rutiner och begränsningar kring tekniker är det enbart en fråga om mer begränsade parametrar till de befintliga SimPy resurserna i modellen.

8.1.2 Utveckling av gränssnittet

Gränssnittet som skapades under arbetet fyller sin funktion i att, på ett lättillgängligt sätt testa simuleringen och snabbt få en översikt på resultaten. Dock är inte analysen av olika de olika resultaten lika tillgängligt. För att uppnå bättre tillgänglighet för resultaten vore det därför bra att inkludera funktionalitet för mer detaljerade rapporter, likt den presenterad i analysavsnittet, där deskriptiv data från de olika simuleringarna jämförs.

Just nu ställer programmet krav på användaren att utföra denna typ av analys själv, vilket inte är idealt ur ett användarperspektiv.

8.1.3 Bayesiansk potential

Om portabla blododlingsskåp (TCC) tas i bruk kan insamling stora mängder data, tagna direkt efter provtagning lagras. Med en större datamängd där tiden till inladdning är kort kan modeller likt de som presenterades i avsnitt 5.3 utökas till att bli mer träffsäkra. Med hjälp av sådana modeller finns potential att skapa prediktioner, baserat på tiden till positivitet, kring vilken typ av bakterie det är frågan om. I bästa fall kan detta alltså ge en fingervisning redan innan någon analys på laboratoriet ägt rum. Detta bygger dock på antagandet om att bakterierna faktiskt skiljer sig tillräckligt åt. En stor problematik med modellen för tid till positivitet som används i detta arbete är dock att den enbart bygger på tiden innan inladdning samt bakterietyp. Om man zoomar in i blodprovet är antalet ursprungsbakterier tillsammans med en rad andra faktorer som exempelvis temperatur sannolikt direkt avgörande för tiden till positivitet. De parametrarna är dock svåra om inte omöjliga att på ett praktiskt sätt observera.
Sammanfattningsvis känns det som det finns stor potential kring tillväxttiden, delvis i rent praktiskt då den med bättre rutiner eller nya tekniker kan förkortas, men även i syfte att understöda detektivarbetet kring identifiering av bakterien. Här skall dock tilläggas att skribentens kunskaper i ämnet är begränsade och tanken ligger långt utanför scopet för studien.
Om vi fortsätter i den Bayesianska tankebanan ett tag till kan vi tänka oss en annan studie, där målet, istället för att försöka finna den totala tiden till resultat istället fokuserade på ett hjälpverktyg i identifiering av bakterien. Det är inte bara resultaten av de olika stegen som utförs på laboratoriet som är en indikator för vilken den sökta bakterien är. Tidsåtgången för i princip samtliga steg verkar också i viss utsträckning variera bakterier emellan. I denna studie togs enbart detta i beaktande för tiden till positivitet, en mer ingående modell med fler aspekter inkluderade hade potentiell kunnat bidra, inte bara till en bättre analys av totaltiden utan även i arbetet med identifiering av bakterien. Alltså resultatet av test/steg **n** givet tiden testet tog samt samt tidigare steg.

8.2 Felkällor

I denna typ av arbete finns många felkällor, den nyfikna läsaren har kanske tagit en titt i appendix och kikat på inparametrarna för basfallet, där finner man 66 olika parametrar, varav 63 är oberoende föränderliga. Om alla dessa bara varit binära, sant eller falskt, (som egentligen bara 6 av dem är) hade simuleringen teoretisk sätt möjliggjort 2^{63} olika flöden. Och nu är ju faktiskt större delen av parametrarna kontinuerliga, vilket göra att det faktiska antalet unika flöden som kan simulerats snarare är en filosofisk fråga. Detta innebär fantastiska möjligheter men för även med sig fantastiska problem, inte minst i modellbyggandet och valideringen. I ett så stort system är det svårt, om inte omöjligt att exakt precisera var förändringar och eventuella fel sker. Det skulle verkligen inte förvåna mig på en par ställen två små fel i slutändan blir ett rätt. Då det heller inte finns klinisk data att jämföra dessa alternativa flöden med försvåras valideringsprocessen. Därför är det viktigt att dessa resultat mer ses som en fingervisning, en förstudie för fortsatta undersökningar.

8.3 Sammanfattning

Sammanfattningsvis kan vi konstatera att det går bra att simulera för basfallet. Vid jämförelser med olika alternativa flöden tycks tiden till resultat påverkas. Resultaten visar även att finnas mycket tid att vinna på förändring av rutiner som arbetstider. Samt att dessa rutiner begränsar den potentiella effekten nya tekniker kan ha. Det finns stor potential för vidare analyser och för att rättvist jämföra prestandan av modellen skulle den behöva jämföras mot faktisk data från andra flöden. Samtidigt behöver modellen sannolikt utökas för att rättvist beskriva mer komplexa system.

Litteratur

[1] Maria Andersson m. fl. *Sepsis och septisk chock -tidig identifiering och initial handläggning*. Tekn. rapport. För Svenska Infektionsläkarföreningen, jan. 2018.

[2] Anand K m. fl. "Duration of hypotension before initiation of effective antimicrobial therapy is the critical determinant of survival in human septic shock". English. I: *Critical care medicine* 34.6 (2006), s. 1589–1596.

[3] Jarvius Jonas. *Q-linea utvecklar portabel blododlingsteknik*. 2021. URL: https://www.qlinea.com/mfn_news/q-linea-develops-portable-blood-culture-technology/. (accessed:2021-03-07).

[4] Dr Mike Weinbren m. fl. *Clinical Toolkit 8: Laboratory handling and reporting of blood cultures*. 2015. URL: https://sepsistrust.org/wp-content/uploads/2018/06/BC-toolkit-final-Sept15.pdf (hämtad 2021-03-07).

[5] B. Van den Poel m. fl. "How small modifications in laboratory workflow of blood cultures can have a significant impact on time to results". English. I: *European journal of clinical microbiology & infectious diseases* 37.9 (2018), s. 1753–1760.

[6] Mary Adamik m. fl. "Effect of delayed entry on performance of the BACT/ALERT FAN PLUS bottles in the BACT/ALERT VIRTUO blood culture system". I: *European journal of clinical microbiology & infectious diseases : official publication of the European Society of Clinical Microbiology* (okt. 2020).

[7] Julika Schwarzenbacher m. fl. "On-site blood culture incubation shortens the time to knowledge of positivity and microbiological results in septic patients". I: *PLOS ONE* 14.12 (dec. 2019), s. 1–13. URL: https://doi.org/10.1371/journal.pone.0225999.

[8] Francesco Congestrì m. fl. "Comparison of 'time to detection' values between BacT/ALERT VIRTUO and BacT/ALERT 3D instruments for clinical blood culture samples". I: *International Journal of Infectious Diseases* 62 (2017), s. 1–5. URL: https://www.sciencedirect.com/science/article/pii/S1201971217301662.

[9] Pajaree Krisanapan och Romanee Chaiwarith. "Time to blood cultures positivity of microorganisms using a continuous-monitoring automated blood cultures system". I: *Asian Biomedicine* 13.2 (1Apr. 2019), s. 61–69. URL: https://content.sciendo.com/view/journals/abm/13/2/article-p61.xml.

[10] Robert Hine. *Bacterial growth curve*. 2019. URL: https://www.oxfordreference.com/view/10.1093/acref/9780198821489.001.0001/acref-9780198821489-e-409.

[11] TJ Beveridge. "Use of the Gram stain in microbiology". I: *Biotechnic & Histochemistry* 76.3 (2001), s. 111–118. URL: https://doi.org/10.1080/bih.76.3.111.118.

[12] Samira Hosseini och Sergio O. Martinez-Chapa. "Principles and Mechanism of MALDI-ToF-MS Analysis". I: *Fundamentals of MALDI-ToF-MS Analysis: Applications in Bio-diagnosis, Tissue Engineering and Drug Delivery*. Singapore: Springer Singapore, 2017, s. 1–19. URL: https://doi.org/10.1007/978-981-10-2356-9_1.

[13] L. Barth Reller m. fl. "Antimicrobial Susceptibility Testing: A Review of General Principles and Contemporary Practices". I: *Clinical Infectious Diseases* 49.11 (dec. 2009), s. 1749–1755. URL: https://doi.org/10.1086/647952.

[14] Q-linea. *ASTar*. 2021. URL: https://www.qlinea.com/sv/vara-produkter/. (accessed:2021-03-21).

[15] Igor Rychlik och Jesper Rydén. *Probability and risk analysis: an introduction for engineers*. English. Berlin: Springer, 2006.

[16] Giuseppe Ciaburro. *Hands-On Simulation Modeling with Python*. English. 1. utg. S.l.: Packt Publishing, 2020.

[17] Team SimPy. *Simpy*. 2020. URL: `https://simpy.readthedocs.io/en/latest/index.html`. (accessed:2021-03-07).

[18] Pauli Virtanen m. fl. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". I: *Nature Methods* 17 (2020), s. 261–272.

[19] John Salvatier, Thomas V Wiecki och Christopher Fonnesbeck. "Probabilistic programming in Python using PyMC3". I: *PeerJ Computer Science* 2 (2016), e55.

[20] Alan D. Moore. *Python GUI Programming with Tkinter: Develop Responsive and Powerful GUI Applications with Tkinter*. English. Birmingham: Packt Publishing, Limited, 2018.

[21] Sven E. Alm och Tom Britton. *Stokastik: sannolikhetsteori och statistikteori med tillämpningar*. Swedish. 1. uppl. Stockholm: Liber, 2008.

[22] George Ho Luis Mario Domenzain. *Censored Data Models Censored Data Models*. URL: `https://docs.pymc.io/notebooks/censored_data.html`. accessed:2021-03-07.

[23] Osvaldo Martin. *Bayesian Analysis with Python: Introduction to Statistical Modeling and Probabilistic Programming Using PyMC3 and ArviZ, 2nd Edition*. English. 2nd. Birmingham: Packt Publishing, Limited, 2018.

[24] Therese M. Donovan och Ruth M. Mickey. *Bayesian statistics for beginners: a step-by-step approach*. English. First. Oxford, United Kingdom: Oxford University Press, 2019.

[25] Cameron Davidson-Pilon. *Bayesian methods for hackers: probabilistic programming and Bayesian inference*. English. 1. utg. New York: Addison-Wesley, 2016;2015;

[26] Luke A. Jardine m. fl. "Neonatal blood cultures: Effect of delayed entry into the blood culture machine and bacterial concentration on the time to positive growth in a simulated model". English. I: *Journal of paediatrics and child health* 45.4 (2009), s. 210–214.

Appendices

Bilaga A

SimuleringsParametrar

```
#Simulation configuration
simulationParams = {
    'timeUntilIdParams' :
                    {'rapidId': True ,
                    'meanIsolationTime':16*60,
                    'meanIsolationTimeInterval':1*60,
                    'IsolationTimestd':0.5*60,
                    'meanIdTime':1*60,
                    'meanIdTimeInterval':0.1*60,
                    'idTimeStd':0.05*60
                   },
    'timeUntilAstParams' :
                    {'rapidAst': False ,
                     'meanAstTime':16*60 ,
                     'meanAstTimeInterval':1*60,
                     'AstTimeStd':30*60},
    'timeToLoadParams' :
                    {"deliveryTimes" :
                        [8*60,11*60,14*60],
                    "weekendDeliveryTimes" :
                        [11*60],
                    "transportTime" :
                        [23,320],
                    "loadTime" :
                        [5,15]},
    'incubatedBeforeLab' : False,
    'tccEnabled' : False,
    'bccBeforeLab': False,
    'generatedFromList' : True,
    'generateSamplesIntensityParamas':
                    {"mean": 0.00378034390,
                     "amplitude": 0.002513773,
                     "phase": 10.079698332},
    'distributionParamsGramStrain' :
                    [2.1760552995989997,
                    8.693871785945708,
                    0.6517746981213182,
                    300.49210541795657],
    'sampleSize' : 512,
    'simTime' : 5*30*60*24,
    'MeanloadTime' :
                    {'meanLoadTime':16*60 ,
                      'meanLoadTimeInterval':1*60,
                      'LoadTimestd':0.5*60},
    'limitations' :
                    {'defaultResurceLimitation' : 1000,
                     'PSACapacity' : 20,
                     'tccCapacity' : 10000},
    'OfficeHours' :
                    {'endOfDay': 20*60,
                     'startOfDay' : 7*60},
    'weekendHours' :
                    {'endOfDay': 19*60,
                      'startOfDay' : 10*60},
    'distDoStuffAtLab' :
                    [4,10],
    'labPersonel': 3,
    'originalData' : originalDf,
    'prevalence' : {
        7: 0.2962962962962963,
        40: 0.2046783625730994,
        17: 0.10916179337231968,
        5: 0.09941520467836257,
        9: 0.05263157894736842,
        38: 0.05263157894736842,
        14: 0.031189083820662766,
        25: 0.021442495126705652,
        37: 0.021442495126705652,
        39: 0.01949317738791423,
        36: 0.01949317738791423,
        26: 0.01364522417153996,
        2: 0.011695906432748537,
        28: 0.009746588693957114,
        34: 0.009746588693957114,
        20: 0.007797270955165692,
        18: 0.005847953216374269,
        33: 0.003898635477582846,
        19: 0.003898635477582846,
        4: 0.001949317738791423,
        15: 0.001949317738791423,
        10: 0.001949317738791423}
}

{'conferenceParams':
            {'meanTimeOfConferences':[9.5*60,11.50*60],
            'conferenceStandardDiviation': 0.1*60,
```

```
'intervalRange':0.5*60}
}
```

Listing A.1: Imput till basfallet