CHALMERS | GÖTEBORGS UNIVERSITET

# Exceptionella klustringsprocesser för urskog

## Simulering och analys av exceptionell klustring i trädpopulationer

# Exceptional Cluster Processes for Old-Growth Forest

*Examensarbete för kandidatexamen i matematik vid Göteborgs universitet*
*Kandidatarbete inom Civilingenjörsutbildningen vid Chalmers*


Mats Vallin Crnoja
Amelia Jonsson
Alexander Lenngren
Lisa Magnusson



Institutionen för Matematiska vetenskaper
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Göteborg, Sverige 2025

# Exceptionella klustringsprocesser för urskog

Simulering och analys av exceptionell klustring i trädpopulationer

*Examensarbete för kandidatexamen i matematisk statistik vid Göteborgs universitet*
Amelia Jonsson Lisa Magnusson

*Examensarbete för kandidatexamen i matematisk statistik inom Matematikprogrammet vid Göteborgs universitet*
Mats Vallin Crnoja

*Kandidatarbete i matematik inom Civilingenjörsprogrammet Teknisk fysik vid Chalmers*
Alexander Lenngren

Handledare: Aila Särkkä

Institutionen för Matematiska vetenskaper
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Göteborg, Sverige 2025

# Förord

I detta kandidatarbete inom matematisk statistik utforskar vi olika modeller för exceptionella kluster av trädväxt i skog. Arbetet har skrivits under handledning av Aila Särkkä vid institutionen för matematiska vetenskaper vid Chalmers och Göteborgs universitet.

Vi vill först och främst uttrycka vår tacksamhet till vår handledare Aila Särkkä som har gjort det lilla extra i hennes stöttande och vägledning av oss under arbetets gång. Vi vill även tacka Naturresursinstitutet i Finland för tillgång till punkterna i deras ERIKA-projekt.

Det har under projektet förts en veckovis dagbok innehållande individuella och gemensamma bidrag samt timmarna vi spenderat på arbetet. Ansvaret för denna har skötts genom ett rullande schema. Nedan finns även en tabell där huvudförfattare för respektive del står listad, samt en tabell på vem som har kodat vilka modeller. Samtliga medlemmar har även reviderat hela rapporten.

Under arbetet har skrivandet och kodandet delats upp på följade sätt:

| | |
|---|---|
| Populärvetenskaplig | Amelia |
| Sammandrag & abstract | Amelia, Lisa |
| Inledning | Alexander, Amelia, Lisa |
| Teori för spatiala punktprocesser | Lisa, Alexander |
| | 2.1 Lisa, Alexander, Mats |
| | 2.2 Lisa, Amelia, Alexander |
| | 2.3 Alexander |
| | 2.4 Lisa |
| | 2.5 Lisa, Alexander |
| Klusterprocesser | Alexander |
| | 3.1 Alexander |
| | 3.2 Lisa, Amelia |
| | 3.3 Lisa, Alexander |
| | 3.4 Mats |
| | 3.5 Mats |
| Sammanfattande Statistiker | Lisa, Alexander |
| | 4.1 Lisa, Alexander |
| | 4.2 Lisa, Alexander |
| | 4.3 Lisa, Alexander |
| | 4.4 Lisa, Alexander |
| Sammanfattning | Lisa |
| Framtida forskning | Samtliga |

| | |
|---|---|
| Thomasprocessen: Standard, Omodifierad | Amelia, Lisa |
| Thomasprocessen: Halvvägs | Lisa |
| Iterativ Thomasprocessen | Alexander, Lisa |
| Log Guassian Cox Process | Mats |
| Intensitetsstyrd Thomasprocess | Mats |
| Plots för sammanfattande statistikor | Alexander |

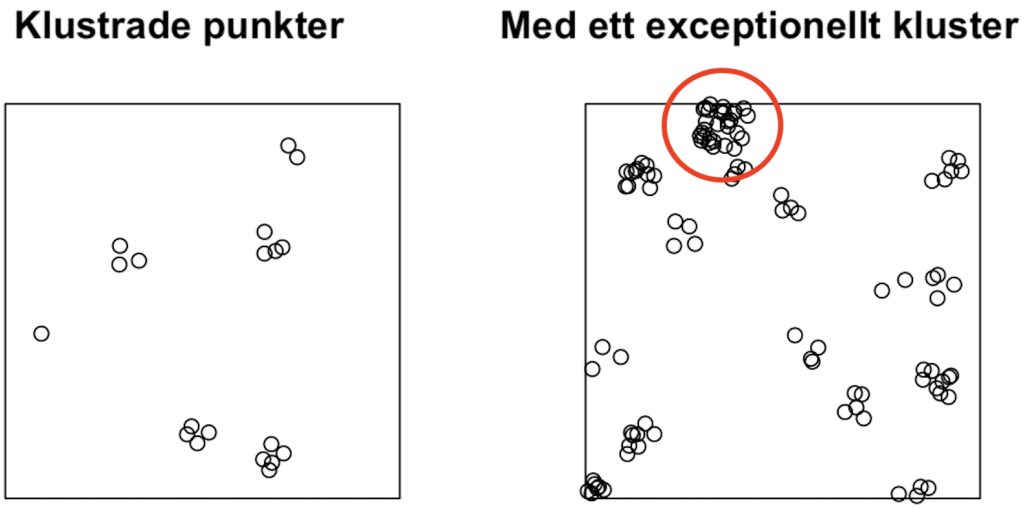
# Användandet av AI

Under detta arbete har AI i vissa fall använts för förslag av omformuleringar på färdigskrivna texter, samt för att öka förståelse för vissa paket inom de olika programmeringsspråken. Den AI som har använts är GPT-4o.

# Populärvetenskaplig presentation

För att förstå världen runtomkring oss delar vi ofta in objekt i mönster. Har du någonsin tittat upp på stjärnhimlen, lagt märke till hur vissa områden har fler stjärnor än andra och hur några nästan verkar klumpa ihop sig? Detta är vad vi kan kalla för kluster av stjärnor. På samma sätt som vi kan observera sådana mönster på stjärnhimlen, kan vi upptäcka liknande kluster bland trädens placering i skogen.

Föreställ dig att vi har en karta som representerar ett skogsområde, där varje träd markeras med en motsvarande punkt. På detta sätt kan vi tolka den fysiska skogen som ett punktmönster, där varje punkt representerar ett träd. Ett sådant punktmönster anses vara klustrat om det visar tydliga grupperingar av punkter. Ibland kan dessa kluster vara mer påtagliga, där vi ser både mindre och större samlingar av punkter, sådana mönster kallas för exceptionellt klustrade punktmönster. I figuren nedan ser vi en visualisering av skillnaderna mellan dessa.




För att analysera hur punkterna i mönstret förhåller sig till varandra använder vi oss av matematiska modeller som kallas för spatiala punktprocesser. Det betyder att fokuset inte ligger på att återskapa de enskilda trädens exakta positioner, utan att fånga upp det övergripande mönstret som punkterna skapar och analysera det. Detta blir särskilt utmanande när mönstret är exceptionellt klustrat, eftersom de vanligaste punktprocessmodellerna för klustrade punkter inte är tillräckliga för att beskriva sådana mönster på ett bra sätt. Syftet är därför att bygga modeller som kan efterlikna exceptionella klustermönster, som till exempel det vi ser i bilden till höger ovan.

Den data som används i detta arbete är en del av ERIKA-projektet som genomfördes på Naturresursinstitutet i Finland. Projektet var en omfattande insamling av många olika typer av skogsdata. Inkluderat i detta var även trädens spatiella placering i olika områden av urskog, som presenterades i form av ett punktmönster. I en tidigare studie kom man fram till att ett av dessa skogsområden, VES13, är exceptionellt klustrat. Det är detta område som vi fokuserar på, och målet är att bygga nya modeller som kan återskapa det exceptionellt klustrade mönstret på ett sätt som stämmer överens med det som observerats.

**Sammandrag**

I detta arbete modelleras ett exceptionellt klustrat punktmönster med hjälp av spatiala punktprocesser. Punktmönstret representerar den spatiella placeringen av träd i skogsområdet VES13, givet av ERIKA-projektet som genomförts i Finland. Syftet är att återskapa det observerade mönstret genom att utveckla och analysera modeller anpassade till denna typ av data. Simuleringar genomförs i programmeringsspråket R med hjälp av paketet `spatstat`.

För att efterlikna det trädmönstret givet av datamängden testades först några standardmodeller för klustrade punktmönster såsom Thomasprocessen, den omodifierade Thomasprocessen samt en kombination av dessa kallad halvvägs Thomasprocess. Modellerna lyckades skapa kluster men inte exceptionella kluster, så för att bättre fånga dessa utvecklades tre nya modeller som även tar hänsyn till biologiska faktorer. den iterativa Thomasprocessen skapar kluster genom en iterativ generering av dotterpunkter, medan den intensitetsstyrda Thomasprocessen tillåter parameteranpassningar för större kontroll över storleken och tätheten på kluster. Slutligen anpassas en hierarkisk Poissonprocess där intensiteten styrs av ett latent fält, vilket modellerar klustring genom variation av intensiteten.

Avslutningsvis jämfördes modellerna med ett test och olika sammanfattande statistiskor. Resultaten visar att de modeller med hänsyn till biologiska faktorer lyckas efterlikna det exceptionella klustermönstret från datamängden bäst.

**Abstract**

In this study, an exceptionally clustered point pattern is modelled using spatial point processes. The point pattern in question represents the spatial distribution of trees in the forest area VES13, provided by the ERIKA project conducted in Finland. The aim is to replicate the observed pattern by developing and analyzing models suitable for this type of data. Simulations are performed in the R programming language using the `spatstat` package.

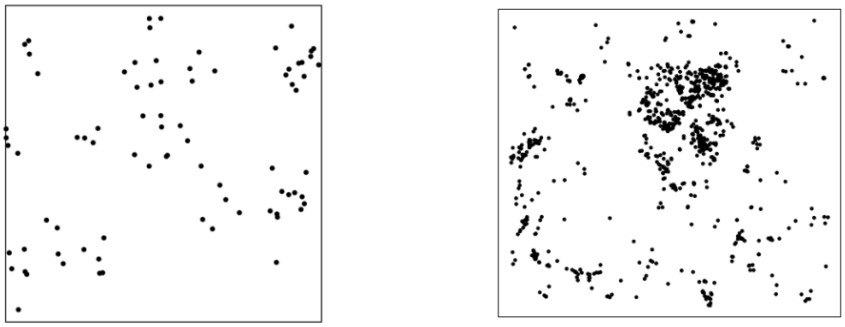
To replicate a point pattern like the one observed in the given dataset, some standard models for clustered point patterns were tested, namely the Thomas process, the unmodified Thomas process, and a hybrid version referred to as the halfway Thomas process. While these models were able to produce clustering, they failed to capture the exceptional clustering seen in the data. To better capture this, three new models incorporating biological factors were developed. The iterative Thomas process creates clusters through generations of offspring, while the intensity-controlled Thomas process allows parameter adjustments to control the size and density of clusters. Finally, a hierarchical Poisson process was fitted, where the intensity is dependent on a latent field, modeling clustering through spatial variation in intensity.

Lastly, the models were compared using one test and various summary statistics. The results indicate that the models incorporating biological factors were better at replicating the exceptional clustering observed in the data.

# Innehåll

# 1 Inledning

Data i form av positioner av punkter, såsom positioner av träd i en skog, celler i kroppen eller stjärnor i himlen, kallas punktmönster. I andra ord är ett punktmönster en datamängd som ger de observerade spatiala lägen av observationer, till exempel träd, celler eller stjärnor. Ett punktmönster kan ses som resultatet från en punktprocess, vilket är en matematisk modell som slumpmässigt genererar punkter inom ett givet område. Ett "vanligt" klustrat punktmönster har ungefär lika stora kluster över hela punktmönstret som kan ses i vänstra bilden i figur 1. Ett exceptionellt punktkluster har däremot ett, eller flera, kluster som är mycket större eller tätare än de andra, vilket syns i den högra bilden i figur 1. Den högra bilden är just det punktmönstret som vi är intresserade av, och det visar hur träd har växt i en skog.



Figur 1: Vänstra bilden visar ett "vanligt"klustrat punktmönster med ungefär lika stora kluster. Högra bilden visar ett punktmönster med flera exceptionella kluster.

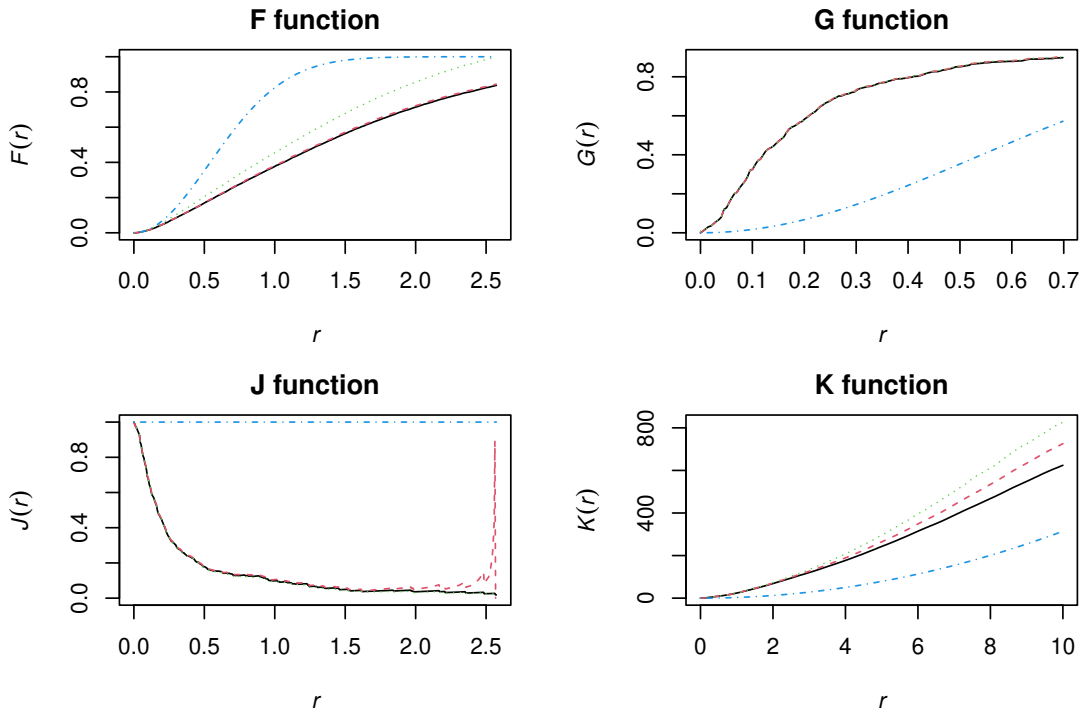
Målet med detta projekt är att konstruera en punktprocessmodell som genererar exceptionellt klustrade punktmönster som efterliknar punktmönstret till höger i figur 1, som kallas VES13. Visuellt syns det att punktmönstret i den högra bilden är klustrat med exceptionella kluster, men det finns även spatiala analys metoder, som till exempel Ripley's K-funktion, som kommer användas för att avgöra hur väl modellerna passar till vårt givna punktmönster.

Till en början kommer den så kallade Thomasprocessen användas för att modellera klustrade punktmönster där alla kluster, i genomsnitt, har lika många punkter. Det är en väletablerad metod för att skapa kluster, och därmed en naturlig startpunkt. Thomasprocessen utgår från ett antal Poissonfördelade föräldrarpunkter och placerar sedan ut ett Poissonfördelat antal dotterpunkter i kluster med en viss intensitet runt föräldrarpunkterna.

Till börjar att med så definieras grundläggande egenskaper för modellerna, såsom antalet föräldrarpunkter och deras spatiala fördelning, samt antalet dotterpunkter och deras fördelning runt föräldrarna. Genom att justera olika parametrar försöker vi återskapa ett punktmönster med exceptionella kluster som liknar datamängden, VES13, för att förstå strukturen på våra kluster. I detta sammanhanget är det troligt att Thomasprocessen inte är tillräckligt bra för att återskapa det punktmönster som observerats. Detta eftersom en Thomasprocess kommer att skapa kluster, men inte exceptionella kluster. För att korrigera detta testas en omodifierad Thomasprocess, där tomma mängder av dotterpunkter inte är tillåtna. Efter detta går vi vidare till mer avancerade metoder, där det till exempel tillåts inhomogenitet i processen som genererar punkterna.

Med en visuell koll kan det delvis avgöras om punkter är klustrade eller inte, och möjligtvis om de är exceptionellt klustrade, men det krävs sammanfattande statistikor för att avgöra vart gränsen mellan klustrad och icke-klustrad data går. Med hjälp av dessa kan klustringen i det genererade punktmönstret analyseras och jämföras mot datamängden. För att göra detta används fyra vanliga statistikor för spatiala punktprocesser: Ripley's K-funktion, F-funktion, G-funktion och J-funktion. Figur 2 visar K-, G-, F- och J-funktionerna tillämpade på vår datamängd. L-funktionen, en om-

formulering av K-funktionen, ges av $L(r) = \sqrt{K(r)}$ och används för att jämföra modeller då det blir enklare att jämföra modellerna än i K-funktionen.




Figur 2: F-, G-, J-, K-funktionerna för datamängden.

För statistisk analys av punktmönster används programmeringsspråket R, som främst används inom statistisk analys, speciellt paketet Spatstat som lämpar sig för analys av punktmönster, samt Python. Kod skrivs från grunden, främst med hjälp av boken Spatial Point Patterns, Methodology and Applications with R [2]. Om ingen referens angetts, så kan läsaren anta att teorin är tagen från denna bok.

# 2 Teori för spatiala punktprocesser

Spatiala punktprocesser, som beskriver hur punkter ligger i förhållande till varandra, används som matematiska modeller för punktmönster. Ofta delas punktmönster in i tre huvudgrupper, helt spatialt slumpmässiga (ingen uppenbar struktur), klustrade och reguljära. Vi är specifikt intresserade av klustrade mönster, som kan förekomma i många olika sammanhang och skalor, såsom positioner av små träd i en skog eller positioner av sjukdomsfalls. Deras breda tillämpbarhet är vad som gör dem så intressanta att studera vidare och modellera. Målet är inte att modellera generella punktmönster, utan att hitta en matematisk modell som är klustrad på liknande sätt som en datamängd. I detta kapitel definieras inledningsvis punktprocesser, med särskilt fokus på Poissonprocessen. Därefter introduceras ett test samt några sammanfattande statistikor som kan användas för att utvärdera anpassningen i de olika modellerna. Avslutningsvis behandlas även kanteffekter, som är viktiga att ta hänsyn till när olika statistikor skattas för en punktmängd.

## 2.1 Definition av Punktprocesser

En spatial punktprocess, $X$, är en stokastisk samling av punkter som i detta arbete är planet $\mathbb{R}^2$. Begreppet process syftar till fördelningen som använts för att få punkterna på ett specifikt sätt. I ett område $S \subseteq \mathbb{R}^2$ kan en punktprocess ses som ett slumpmässigt räknemått, sådan att för varje begränsad mängd $W \subseteq S$, antalet punkter i $X$ i $W$ betecknas som $N(W) = \#(X \cap W)$. Den

observerade datamängden är då $X_1, ..., X_n$, $X_i = (x_i, y_i)$ och studiefönstret av vår datamängd är $W = [0, 40] \times [0, 40] \subset \mathbb{R}^2$, som även är det fönster som används för modellerna. Avståndet $r$ mellan en vald punkt $X \in W$ och en annan punkt i området $X_i \in W$ skrivs

$$r = d\big(X, X_i\big) = \sqrt{(x - x_i)^2 + (y - y_i)^2}.$$

Att en punktprocess $X$ är stationär innebär att den har samma statistiska egenskaper om det förflyttas med en translationsvektor $\vec{v}$. Detta innebär att $X$ har samma intensitet, det vill säga, det förväntade antalet punkter per enhetsyta. En punktprocess är isotrop om $X$ har samma egenskaper som $X$ roterat runt origo. Det innebär att statistikor kan användas utan inskränkning.

När man analyserar punktmönster är det vanligt att testa om mönstret kan antas vara helt spatialt slumpmässigt (completely spatially random, CSR). Det vill säga att punkterna skulle vara oberoende av varandra och likformigt fördelade på området $W$, och om inte, är punktmönstret reguljärt eller klustrat. Det blir något punktprocesserna förhåller sig till i bland annat Clark-Evans testet och olika sammanfattande statistikor.

## 2.2 Poissonprocessen

I en homogen (stationär) Poissonprocess har vi ett Poissonfördelat antal punkter som är oberoende av varandra och likformigt fördelade i området. Sannolikheten att ha exakt k punkter är då

$$\mathbb{P}(N(W) = k) = \frac{(\lambda|W|)^k}{k!} e^{-\lambda|W|}, \quad k = 0, 1, 2, \ldots,$$

för något område $W \subseteq S$. Det genomsnittliga antalet punkter per enhetsyta, $\lambda$, är konstant. En Poissonprocess kan ses som CSR (completely spacially random) för punktprocesser [8].

Inhomogen Poissonpunktprocess definineras av en intensitetsfunktion, $\lambda(s) \geq 0$, där $s \in W$ och $ds$ integreras över hela området. Funktionen ger antalet punkter i ett mindre område, vi antar att punktmönstren mellan icke-överlappande områden är oberoende, och att antalet punkterna i givet område är Poissonfördelade. Då för någon region $W \subseteq S$ och antal punkter $N(W) \in W$ är intensitetsfunktionen

$$\mathbb{E}[N(W)] = \int_W \lambda(s)\, ds.$$

Då blir

$$P(N(W) = k \mid \lambda) = \frac{\left(\int_W \lambda(s)\, ds\right)^k e^{-\int_W \lambda(s)\, ds}}{k!}.$$

Ett specialfall som kommer användas för LGCP (log-Gaussian Cox-process) är

$$\log \lambda(s) = m(s) + Z(s), \quad Z(s) \sim \mathrm{GP}(0, C), \tag{1}$$

och $Z$ är ett Gaussiskt fält. Så, $Z = \{Z(s) : s \in S\}$ är ett Gaussiskt slumpfält om $\{Z(s_1), \ldots, Z(s_n)\} \sim \mathcal{N}_n(m, C)$, där $\mathcal{N}_n$ är en n-dimensionell normalfördelnig, för varje ändlig mängd $\{s_1, \ldots, s_n\} \subset S$.

## 2.3 Clark-Evans test

Clark-Evans testet är ett enkelt sätt att avgöra om ett punktmönster är CSR. Testet går ut på att jämföra de observerade närmaste avståndet mellan punkterna med det förväntade avståndet i en Poissonprocess med samma intensitet. Om vi har en Poissonprocess med $n$ punkter och intensitet $\lambda$ definieras R-värdet för Clark-Evans testet

$$R = \frac{\bar{d}}{\mathbb{E}[D]} = \frac{2\sqrt{\lambda}}{m} \sum_{i=1}^{m} d_i,$$

där $\mathbb{E}[D] = \frac{1}{2\sqrt{\lambda}}$ är det väntade avståndet från en godtycklig punkt i mönstret till närmsta grannpunkt. Avståndet till närmsta punkten från en punkt, $\bar{d} = \frac{1}{m} \sum_{i=1}^{n} d_i$, med $m$ antalet slumpade

punkter ur mönstret, och skattade intensiteten $\bar{\lambda}$. Med ett Clark-Evans test är det viktigt att korrigera för kanteffekter, vilket kommer bli ännu viktigare för statistikorna. R-värdet är inte begränsat men $R > 0$, och talar om

$$R \begin{cases} < 1, & \text{punkterna är klustrade} \\ \approx 1, & \text{punkterna är CSR} \\ > 1, & \text{punkterna är reguljärt fördelade.} \end{cases}$$

Detta test är ett bra sätt att göra en första koll på både vår datamängd och våra modeller, men det är ett test som ger väldigt lite information. För att få mer information, tittar vi vidare till spatiala sammanfattningsstatistikor.

## 2.4 Spatiala sammanfattningsstatistikor

Ett annat sätt att testa om det observerade stationära punktmönstret kan antas komma från en Poissonprocess är att använda sammafattandestatistikor. K-funktionen är ett mått på klustring där man jämför ett punktmönster mot en Poissonfördelning som är CSR. G-funktionen tittar på avståndet till närmaste grannen för varje punkt, och även här jämförs datamängden mot en Poissonprocess av punkter. F-funktionen är en "empty space" funktion, det vill säga $F(r)$ uppskattar tomrummet. Sist är J-funktionen som är ett kompositmått av G- och F-funktionen, $J(r) = \frac{1-G(r)}{1-F(r)}$. Dessa jämförs mot det teoretiska referensvärdet för Poissonprocessen.

Ripley's K-funktion är ett vanligt sätt att analysera punktmönster, och är ett kumulativt snitt av mängden av datapunkter som ligger inom ett avstånd $r$ av en godtycklig datapunkt u, och har form,

$$K(r) = \frac{1}{\lambda} \mathbb{E}\left[\text{antal av } r\text{-grannar till } u \mid X \text{ har en punkt i } u\right], \tag{2}$$

för alla $r \geq 0$. Funktionen jämförs mot referensvärdet $K(r) = \pi r^2$, som visar Poissonprocessen. Skattningen av K-funktionen för våra punktmönster fås sedan av

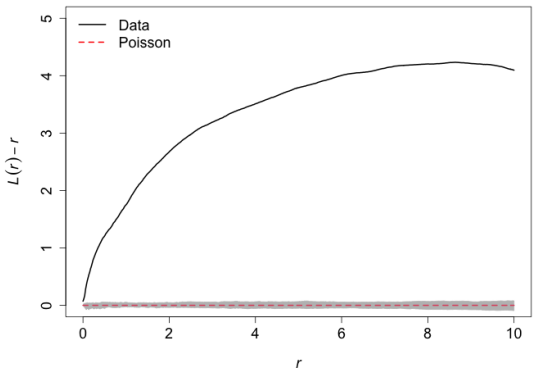
$$\hat{K}(r) = \frac{|W|}{n(n-1)} \sum_{i}^{n} \sum_{j \neq i}^{n-1} \mathbf{1}(d_{ij} \leq r) e_{ij}, \tag{3}$$

där $W$ är arean av det studerade området, $n$ är det totala antalet observerade punkter, $d_{ij}$ är avståndet mellan punkt $i$ och punkt $j$, $\mathbf{1}(d_{ij} \leq r)$ är en indikatorfunktion som är 1 om $d_{ij} \leq r$ och 0 annars, och $e_{ij}$ är en kantkorrigeringsfaktor. Fallet $K(r) = \pi r^2$ ger då en CSR Poissonprocess, $K(r) > \pi r^2$ indikerar klustring, och $K(r) < \pi r^2$ indikerar reguljäritet. Med andra ord, om datamängden ligger över referenskurvan är den klustrad, om den ligger under är den reguljärt fördelad, och om den ungefär ligger på referenskurvan är den CSR.

Ofta används en version av K-funktionen som kallas L-funktionen, definierad $L(r) = \sqrt{\frac{K(r)}{\pi}}$. Den hjälpsamma omformuleringen $L(r) - r$ normaliserar värdena genom att stabiliserar variansen för att enklare kunna tyda resultaten, och då gäller att

$$L(r) - r \begin{cases} < 0, & \text{punkterna är reguljära} \\ \approx 0, & \text{punkterna är CSR} \\ > 0, & \text{punkterna är klustrade} \end{cases}$$

Som ses i figur 3 verkar datamängden vara klustrad, då den tydligt ligger över $L(r) - r$ för Poissonprocessen och konfidenshöljet.




Figur 3: Omformuleringen av L-funktionen för datamängden och ett hölje runt teoretiska värdet av Poissonprocessen.

Närmsta granne fördelningar är andra sammanfattande statistikor som ofta används. En av dem är G, som istället för snittet av alla närmsta grannar använder en kumulativ fördelningsfunktion av avståndet från en godtycklig punkt av processen till den närmsta grannen. För att bygga denna definieras först närmaste grannen till en punkt $x_i$ som

$$d_i = \min_{j \neq i} \|x_j - x_i\| \quad \text{(alternativt skrivet som } d(x_i, X \setminus x_i)\text{)}. \tag{4}$$

Detta leder till kumulativa fördelningsfunktionen G(r) av de närmaste grannavståndet $d(x_0, X \backslash x_0)$ som definieras enligt

$$G(r) = \mathbb{P}\{d(x_0, X \setminus x_0) \leq r \mid X \text{ har punkt i } x_0\} \tag{5}$$

för alla $r \geq 0$ där $x_0$ är en godtycklig punkt av $X$. För en helt slumpmässigt Poissonprocess är $G(r) = 1 - e^{-\lambda \pi r^2}$, där $\lambda$ är intensiteten. När G-funktionen är större än de G-värden som fås från en CSR Poissonprocess tyder det på att datamängden är klustrad, och om G-funktionen är mindre tyder det på att datamängden är ordnad. Skattningen av G-funktionen får vi genom

$$\hat{G}(r) = \frac{1}{n(X \cap W)} \sum_i \mathbf{1}(d_i \leq r), \tag{6}$$

över varje punkt $X_i$ i snittet $X \cap W$.

Istället för att uppskatta avståndet mellan punkter som G(r) gör, uppskattar F(r) tomrummet i ett punktmönster $X$. Avståndet från en godtycklig punkt $y$ i $R^2$ till den närmsta punkten i processen ges av

$$d(y, X) = \min\{||y - x_i|| : x_i \in X\}, \tag{7}$$

och den kumulativa fördelningsfunktionen F(r) av detta avstånd ges av

$$F(r) = \mathbb{P}\{d(y, X) \leq r\} \tag{8}$$

för alla $r \leq 0$. Därigenom fås en bild av hur mycket tomrum som finns i punktmönstret. Den skattade F-funktionen ges av

$$\hat{F}(r) = \frac{1}{m} \sum_{j=1}^{m} \mathbf{1}(d(y_j, X) \leq r), \tag{9}$$

som är en funktion av avståndet $r \geq 0$.

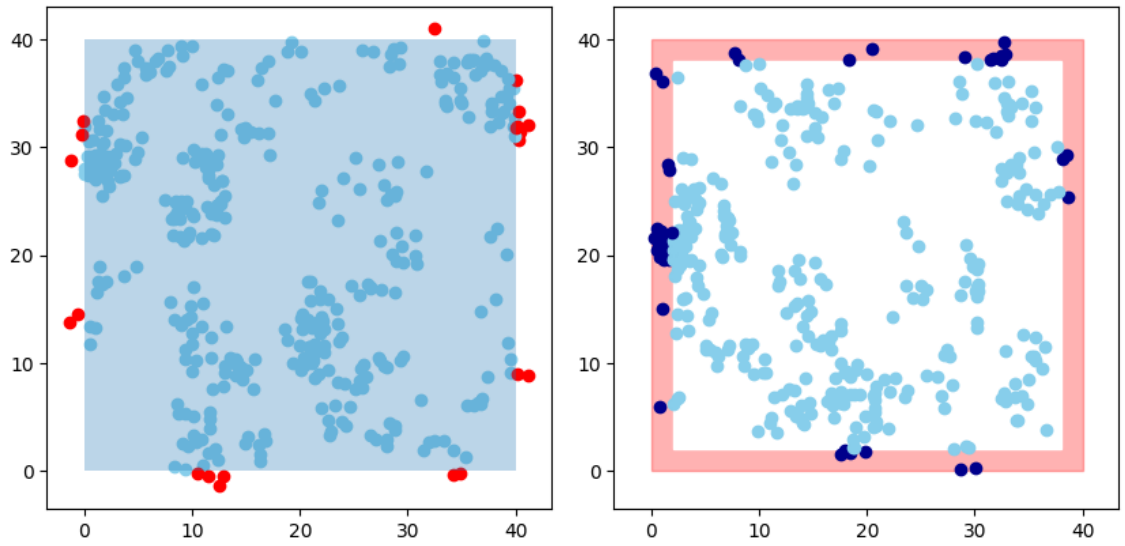
Eftersom G- och F-funktionen är lika när vi har ett CSR fall, den ena minskar när den andra växer, kan det vara intressant att mäta hur mycket ett punktmönster avviker från att vara CSR genom att jämföra dem. Detta kan göras med hjälp av J-funktionen som definieras av

$$J(r) = \frac{1 - G(r)}{1 - F(r)} \tag{10}$$

och är definierad för alla $r \geq 0$ och $F(r) < 1$. För en CSR process har vi att $J(r) = 1$, och datamängden är klustrad när $J(r) < 1$. Det som är bra med J-funktionen är att man inte behöver oroa sig för kanteffekter då de tar ut varandra mellan F- och G-funktionerna.

### 2.5 Kanteffekter

Punktmönster observeras inom ett begränsat område, men mönstret sträcker sig utanför detta område. Detta måste korrigeras för att undvika förvrängningar då punkter utanför det observerade området kan påverka vår analys utan att de syns i punktmönstret. För att avgöra hur punktmönstret ska bestämmas i gränsen används kantkorrigeringar. Denna korrigering kan ske på många olika sätt beroende på vilken funktion som ska tillämpas. F-, G- och J-funktionerna korrigeras med en reducerad delmängd, som genererar punkter från föräldrarpunkterna och sedan endast behåller de punkter som faller i fönstret. När en analys på det begränsade fönstret genomförs, (de blå fönstret i 4), kan man ta med punkterna utanför fönstret i analysen (de röda punkterna i 4. På K- och L-funktionerna används däremot ett starkare kantkorrigeringsverktyg; isotropisk kantkorrigering. Denna korrigering kan inte tillämpas på F-, G-, och J-funktionerna, och därför används reducerad delmängd i de fallen.




Figur 4: Till vänster syns alla punkter $X \in W$, och de punkter i rött som kan orsaka kanteffekter. Till höger illustreras kantkorrigeringen reducerad delmängd, $RS \subset W$.

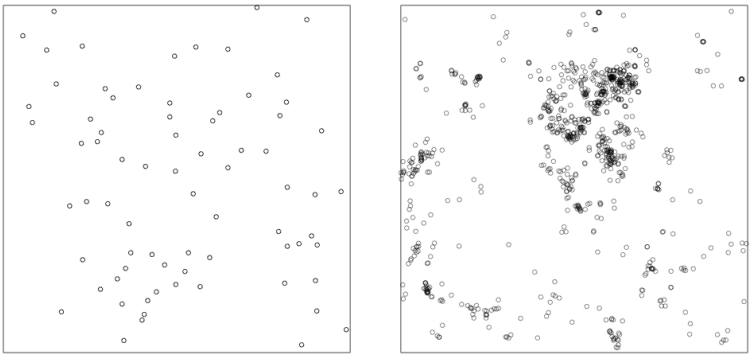
## 3 Klusterprocesser

Med teorin för klusterprocesser som utgångspunkt, utvecklas egna punktprocesser som är konstruerade med både diskreta och kontinuerliga metoder. Vissa processer använder all tillgänglig information, medan andra utgår från så lite information som möjligt. I vissa fall skattas parametrar med hjälp av olika typer av fält. Det som gör projektet särskilt intressant är möjligheten att justera skattningar utifrån olika perspektiv för att efterlikna det punktmönster som observeras i datamängden.

### 3.1 Preliminär analys av data

Arbetet inleds med att skapa en bättre förståelse för hur VES13 är strukturerad. Sauli Valkonen på Naturresursinstitutet (Luke) i Finland har gett oss tillgång till datamängden, som är en del av ERIKA-projektet, vilket var en stor insamling av data ur skogen. Datamängden innehåller

trädens spatiella positioner i form av punktmönster, med både större träd (föräldrarpunkter) och mindre träd (dotterpunkter). Denna datamängd användes tidigare av Kuronen et al. [7] för att anpassa en LGCP för alla skogsområden. De fann dock att datamängden VES13 var exceptionellt klustrad, och kom fram till att en vidare metod behövs. Därmed ligger fokuset i detta arbetet på att modellera dotterpunkterna genom att använda föräldrarpunkterna från datamängden, vars punktmönster syns till höger respektive vänster i figur 5.



Figur 5: Det spatiala mönstret av de stora (vänster) respektive små (höger) träden.

## 3.2 Thomasprocessen: Standard-, Halvvägs, Omodifierad

Thomasprocessen är en klusterprocess där punktmönster skapas genom att först generera föräldrarpunkter som följer en Poissonprocess med intensitet $\kappa$. Därefter genereras dotterpunkter runt varje förälder, vars antal följer en poissionfördelning med väntevärde $\mu$. Dotterpunkternas placering, i förhållande till föräldrarpunkterna, bestäms av en normalfördelning, med väntevärde 0 och varians $\sigma^2$, centrerad vid respektive förälder.

För att simulera Thomasprocessen, även kallad modifierad Thomasprocess i litteraturen, i R används parametrar skattade från datamängden: intensiteten för föräldrarpunkter, $\kappa = 0.046$, den genomsnittliga mängden dotterpunkter per förälder, $\mu = 11.92$, samt standardavvikelsen för avståndet mellan förälder- och dotterpunkter, $\sigma = 8.12$. Sedan slumpas nya föräldrarpunkter fram enligt en Poissonprocess med intensiteten $\kappa$. För varje föräldrarpunkt genereras ett Poissonfördelat antal dotterpunkter med intensitet $\mu$ som har ett normalfördelat avstånd till föräldrarna.

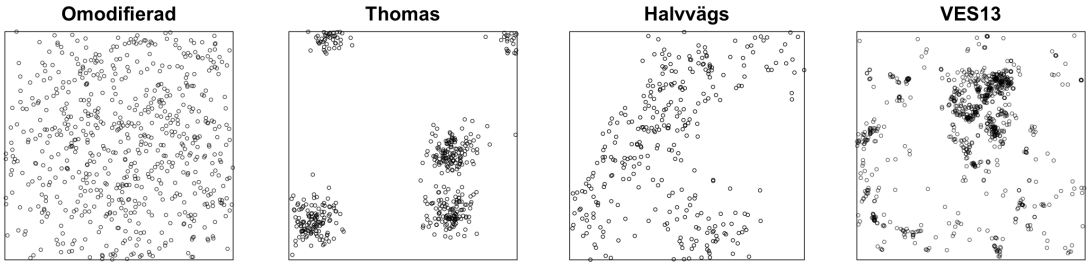
För att genomföra detta användes spatstat-funktionen KPPM för simuleringen en Thomasprocess baserad på vår datamängd. I andra grafen från vänster i figur 6 visualiseras ett utfall av Thomasprocessen. Detta liknar datamängden då det finns både täta kluster och tomma ytor, dock har alla kluster liknande storlek och är därmed inte exceptionella samt så saknas punkter mellan dessa kluster. Eftersom Thomasprocessen tillåter tomma kluster, vilket kan vara orsaken bakom mängden tomrum i punktmönstret, testas en version version av Thomasprocessen som inte tillåter detta i hopp om att minska mängden tomrum. Denna modellen kallar vi omodifierad Thomasprocess, och ett utfall av denna kan ses längst till vänster i figur 6.

Vid jämförelse mellan den omodifierade Thomasprocessen och datamängden observeras tydliga skillnader. Vi ser att den omodifierade modellen helt saknar tomrum och liknande klustersamlingar som är synliga i datamängden. Detta betyder att den omodifierade Thomasprocessen inte är tillräcklig för att modellera klustringen i vår data. För att undersöka en bättre anpassning på datamängden så utvecklas Thomasprocessen ytterligare till en hybridmodell som vi kallar halvvägs Thomasprocess. Detta gjordes genom att ändra på $\mu$, $\kappa$, och $\sigma$ så att:

- $\mu$ är ett harmoniskt medelvärde av Thomasprocessen och den omodifierade Thomasprocessen,
- $\sigma$ är ett harmoniskt medelvärde av Thomasprocessen och den omodifierade Thomasprocessen,

- $\kappa$ är ett geometriskt medelvärde av Thomasprocessen och den omodifierade Thomasprocessen.

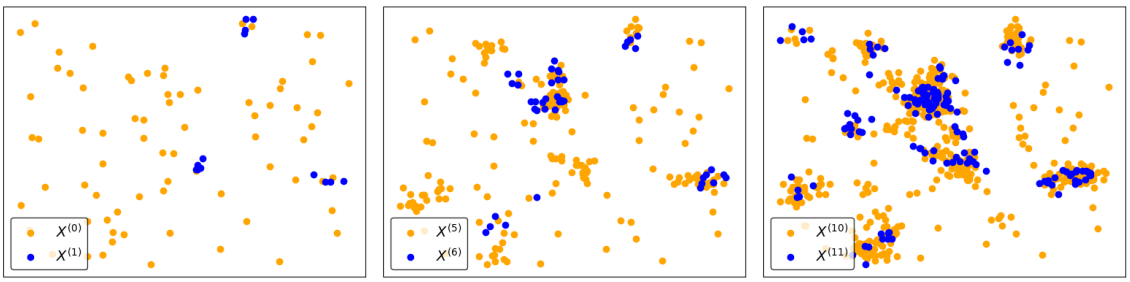
Med andra ord så bygger modellen på utfall från både Thomasprocessen och den omodifierade Thomasprocessen. Parametrarna $\mu$, $\kappa$, och $\sigma$ skattades enligt de ovannämnda medelvärdesmetoderna. Flera olika typer av medelvärden testades och den kombinationen som gav det bästa utfallet valdes för halvvägs Thomasprocessen. Som syns i figur 6 visar punktmönstret från processen både kluster och punkter mellan dem, vilket liknar strukturen i datamängden. Däremot saknas de exceptionella klustren som finns i datamängden, samtidigt som processen skapar för mycket tomrum. Därmed avslutas undersökningen av Thomasprocesser med slutsatsen att halvvägsmodellen, baserat på visuell jämförelse, ger det bästa resultatet hittills. Dock lyckas den fortfarande inte helt fånga de mönster som är synliga i datamängden. Thomasprocesserna är standardmodeller för klustrade punktmönster och eftersom dessa inte lyckas fånga exceptionella kluster, utvecklas nu mer flexibla modeller som även tar hänsyn till biologiska faktorer.




Figur 6: Utfall från våra tre versioner av Thomasprocessen, till höger ser vi datamängden för jämförelse.

## 3.3 Iterativ Thomasprocess

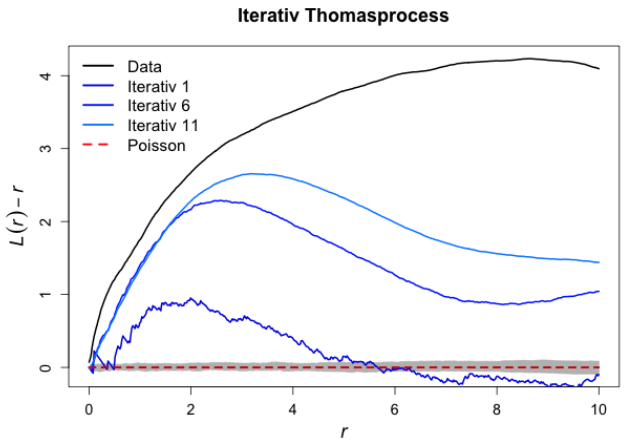
För att bättre modellera naturliga fenomen tillämpades en iterativ punktprocess. Denna typ av modell kan användas inom biologi, exempelvis för att beskriva celltillväxt, men den kan även tillämpas på trädpopulationer där fröspridning sker nära föräldrarträdet. Det initiala antalet föräldrarpunkter, $\kappa$, slumpas från en Poissonfördelning där $\lambda$ är antalet föräldrarpunkter från datamängden, som sedan placeras ut uniformt inom fönstret $W$. Därefter väljs slumpmässigt fem procent av dessa punkter till att förbli föräldrarpunkter medan resterande blir dotterpunkter som inte genererar nya punkter. För varje förälder genereras ett antal dotterpunkter enligt en Poissonfördelning med intensitet $\lambda = 5$, dessa placeras sedan normalt fördelade runt respektive föräldrarpunkt. Därefter väljs återigen slumpmässigt fem procent av punkterna som nya föräldrarpunkter. Genom att upprepa denna processen flera gånger skapas tätare och tätare kluster. Som visas i figur 7, så resulterar denna metoden i både tätare och glesare kluster, vilket gör att de större klustren syns som exceptionella som önskas.




Figur 7: Ett utfall av den iterativa processen efter ett, sex och elva steg.

För att avgöra hur många punkter som skulle genereras sattes ett maximalt antal, som motsvarade det totala antalet punkter i datamängden. Iterationerna avslutades när detta antal uppnåddes.

Sedan analyserades $L(r) - r$ funktionen för olika antal iterationer och resultaten jämfördes mot datamängden i figur 8 för att se det optimala antalet iterationer. Det visade sig att en iteration inte var tillräckligt, medan sex och elva iterationer gav lika resultat fram till $r = 2$. För större radier observeras dock avvikelser från datamängden, vilket motiverade vidare utveckling av modellerna.




Figur 8: L(r)-r funktionen för Iterativ Thomasprocessen efter ett, sex och elva iterationer tillsammans med motsvarande funktion för datamängden och ett intervall runt teoretiska värdet av Poissonprocessen.

## 3.4 Intensitetsstyrd Thomasprocess

Eftersom de tidigare Thomasprocesserna inte gav möjlighet att variara klusterstoleken mellan kluster eller intensiteten av punkmönstret, introduceras nu en modell som möjliggör detta. Den intensitetstyda Thomasprocessen skapar ett stokastiskt intensitetsfält som påverkar antalet dotterpunkter kring varje föräldrarpunkt. Detta möjliggör en manipulation av intensitetsvariationer i fältet som sedan direkt översätts till storlek på kluster.

Startpunkten i modellen är att definiera ett kontinuerligt intensitetsfält $\lambda(\mathbf{x})$ på vårt fönster $W$ som anger varje punkts benägenhet att alstra kluster. Höga värden innebär många potentiella punkter, medan låga värden innebär få. Detta görs med ett Gaussiska slumpfält, $Z = \{Z(\mathbf{x}) : \mathbf{x} \in W\}$ där alla ändliga vektorer $(Z(\mathbf{x_1}), \ldots, Z(\mathbf{x_n}))$ är multivariat normalfördelade, som är en samling av reella slumpvektorer [1]. Täthetsfunktionen för $(Z(\mathbf{x_1}), \ldots, Z(\mathbf{x_n}))$ är:

$$p(z_1, \ldots, z_n) = \frac{1}{(2\pi)^{n/2}\,|\Sigma|^{1/2}} \exp\Big(-\tfrac{1}{2}(\mathbf{z}-\boldsymbol{\mu})^\top \Sigma^{-1}(\mathbf{z}-\boldsymbol{\mu})\Big), \tag{11}$$

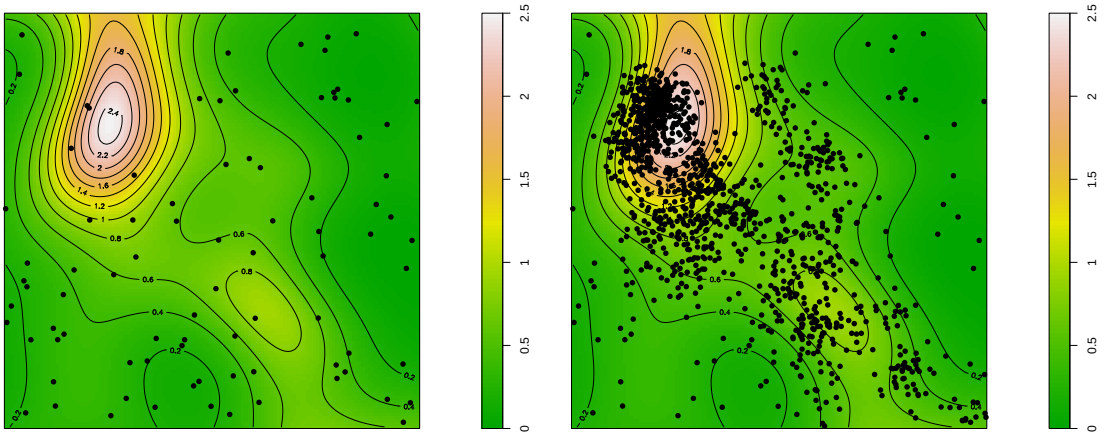
där $\mathbf{z} = (z_1, \ldots, z_n)$ är en vektor av observationer av det stokastiska fältet i punkterna $\mathbf{x}_1, \ldots, \mathbf{x}_n$, $\mu_i = \mu(\mathbf{x}_i)$ är fältets medelvärdefunktion och $\Sigma_{ij} = C(\mathbf{x}_i, \mathbf{x}_j)$ är kovariansfunktionen för fältet. Medelvärdesfunktionen för modellen blev

$$\mu(\mathbf{x}) = \mathbb{E}[Z(\mathbf{x})] = 4 - 1.5\left(\frac{x_1}{40} - 0.5\right)^2 + 2\left(\frac{x_2}{40} - 0.5\right)^2 \qquad \mathbf{x} = (x_1, x_2) \in W, \tag{12}$$

där valet av polynom syftar att få maximum i mitten av fönstret vid $(20, 20)$ och koefficienterna fastställdes genom prövning. Kovarians funktionen blev

$$C(\mathbf{u}, \mathbf{v}) = Cov(Z(\mathbf{u}), Z(\mathbf{v})) = \sigma^2 \exp\left(-\frac{||\mathbf{u}-\mathbf{v}||^2}{2\alpha^2}\right) \tag{13}$$

där $u, v \in W$ och parametrarna igen fasställdes genom prövning. Det fält som används i modellen är en realisation av denna fördelning, och ett utfall av fältet syns i figur 9.



Figur 9: Till vänster visas ett utfall av ett fält och på detta likformigt fördelade föräldrarpunkter, till höger visas resultatet av Thomasprocessen där fältets intensitet styr antalet dotterpunkter.

Efter fältet är skapat placeras ett bestämt antal föräldrarpunkter oberoende och likformigt på området $W$, vilket till exempel kan se ut som den vänstra bilden i 9. Runt varje förälder simuleras sedan ett kluster av dotterpunkter, där antalet dotterpunkter i ett kluster styrs av fältets intensitet i föräldrarpunkterna enligt

$$N_i = \max\{0, \lfloor A[\lambda(p_i) - t]\rfloor\}, \tag{14}$$

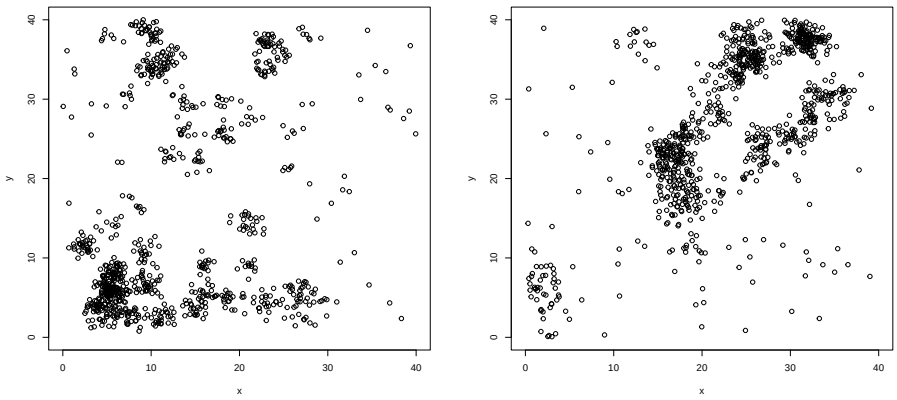
där $t \geq 0$ är ett tröskelvärde för klusterbildning och $A > 0$ är en skalningsparameter. Densitet av klustret styrs av det lokala värdet $\lambda(p_i)$, där $p_i$ är den tillhörande föräldrarpunkten. Placeringen av dotterpunkter sker enligt en Gaussisk fördelning, $x_{ij} \sim \mathcal{N}(p_i, \sigma^2 I)$, med $j = 1, \ldots, N_i$, där $\sigma$ reglerar klusterspridningen. Klustrens egenskaper blir därmed en avbildning av variationen i intensitetsfältet. Parametrarna $A$, $t$, och $\sigma$ kan justeras för att få olika klusterstorlekar och intensiteter.

För att anpassa modellen bättre till datamängden är det möjligt att optimera parametrarna med avseende på en förlustfunktion. Denna baseras på en kombination av de sammanfattande statistikorna (L-, G-, F-, och J-funktionerna) jämförda mellan simulerade punktmönster och datamängden VES13. Förlustfunktionen blir

$$\mathcal{L}(A, t, \sigma) = \sum_{f \in \{L,G,F,J\}} \int_0^{r_{\max}} [f_{\text{sim}}(r) - f_{\text{obs}}(r)]^2 dr$$

som sedan approximeras med trapetsmetoden [5]. Funktionen är byggd så att L-funktionen bidrar mest. Genom en rutnätsökning över intervallen $A \in [10, 100]$, $t \in [0, 1]$, och $\sigma \in [0.2, 3]$ optimeras parametrarna. Möjliga utfall från modellen ses i figur 10, som tydligt visar exceptionell klustring.

Sammanfattningsvis möjliggör denna metod kontroll över klustrens intensitet och storlek samtidigt som den anpassar sig till datamängden som vill efterliknas.




Figur 10: Två utfall av en intensitetsstyrd Thomasprocess.

## 3.5 Hierarkisk Poisson

Trädfördelning i det exceptionellt klustrade urskogsområdet som ses i datamängden härrör både från mätbara processer (t.ex. hur stor trädstammar hämmar plantetablering) och från omätbara eller stokastiska faktorer (mikrotopografi, djurspridning, småskalig mortalitet ect). En Poissonregression kan beskriva de förstnämnda, men har problem med att fånga residual klustring som inte kan förklaras med kovariater.

För att fånga residual klustring tilläggs en slät additiv term $\xi(X)$ som estimeras med Poisson-GAM enligt Dovers et al. [4]; se appendix A.2 och A.3 för kvadratur och implementeringsdetaljer. Intensiteten modelleras då som

$$\log \hat{\lambda}(X) = \beta_0 + \beta_1 C_1(X) + \xi(X).$$

Vi vill konstruera $\lambda(X)$ så att funktionen fångar de egenskaper som vuxna träd har på träplantor samtidigt som funktionen kan användas för att generera nya mönster liknande Figur 1. Följaktligen specificeras en enda biologiskt motiverad kovariat som mäter hur stora träd hämmar nyetablering av plantor i sin närhet. Inkluderingen av kovariaten motiveras av en minskning på AIC (Akaike informationskriteriet) med 4.4 då den inkluderas. Den definieras som

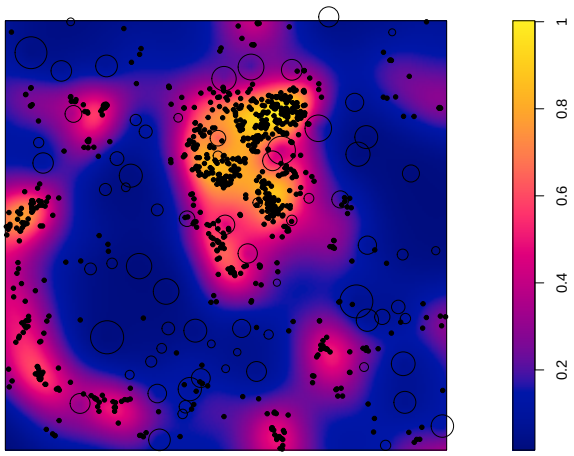
$$C_1(X) = \sum_k \mathrm{dbh}_k \, \exp\left(-\frac{d(X, X_k)}{t}\right),$$

där $X_k$ är koordinaten för träd $k$, $\mathrm{dbh}_k$ dess stamdiameter och $t \approx 10\,\mathrm{m}$ reglerar räckvidden för konkurrenseffekten, värdet för $t$ bestäms genom en rutnätssökning som testas i regressions modellen varpå det värdet som som minimerar AIC väljs,

$$\mathrm{AIC} = 2k - 2\log \hat{L},$$

där $k$ är antalet fria parametrar och $\hat{L}$ är maximala likelihood som ges av regressions modellen.

Trots att $\hat{\lambda}(X)$ fångar de dominerande trenderna i datamängden, är modellen fortfarande deterministisk. För att kunna simulera punktmönster som inte bara följer medelstrukturen utan också återger den variation och slumpmässighet som observeras i fält, krävs en stokastisk utvidgning. Detta uppnås genom att modellera intensiteten som en stokastisk process via en hierarkisk Poissonprocess där variationen kring medelintensiteten omformas genom att använda $\hat{\lambda}(X)$ som medelvärde i ett Gaussiskt fält, där kovariansen hos fältet är vad som ger varierande fält. Fältet som alstras av metoden ges av figur 11




Figur 11: Skattade intensitets fältet som ges av regressionen på datamängden.

Den hierarkiska Poissonprocessen, som i detta är en LGCP (se appendix A.1) består av två stokastiska steg. I det första steget perturberas $\hat{\lambda}(X)$ genom att addera ett Gaussiskt latentfält $Z(X)$ med Matérn-kovarians, vilken Kuronen et al. [7] visat vara lämpad för att fånga biologiskt kluster i skogsmiljöer. Vilket ger intensitetsfält där intensitetens toppar och dalar förskjuts slumpmässigt runt det ursprungliga mönstret. På så vis kan punktmönster med samma medelstruktur genereras, men med slumpmässigt förskjutna kluster. Matérn-kovariansen ges av:

$$\mathrm{Cov}(Z(u), Z(v)) = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left( \frac{\|u - v\|}{\rho} \right)^{\nu} K_\nu \left( \frac{\|u - v\|}{\rho} \right),$$

så att den stokastiska intensiteten blir

$$\Lambda(X) = \exp\Big( \log \hat{\lambda}(X) + Z(X) \Big) = \hat{\lambda}(X)\, e^{Z(X)}$$

dock så ändras medel intensiteten efter tillägget av $Z(X)$ enligt:

$$\mathbb{E}[\Lambda(X)] = \hat{\lambda}(X)\, \mathbb{E}[e^{Z(X)}] = \hat{\lambda}(X)\, e^{\sigma^2/2} \; > \; \hat{\lambda}(X),$$

alltså ökar fältets medelvärde. För att bevara den ursprungliga medelintensiteten delar vi därför med $e^{\sigma^2/2}$, så vi får

$$\tilde{\Lambda}(X) = \frac{\Lambda(X)}{e^{\sigma^2/2}},$$

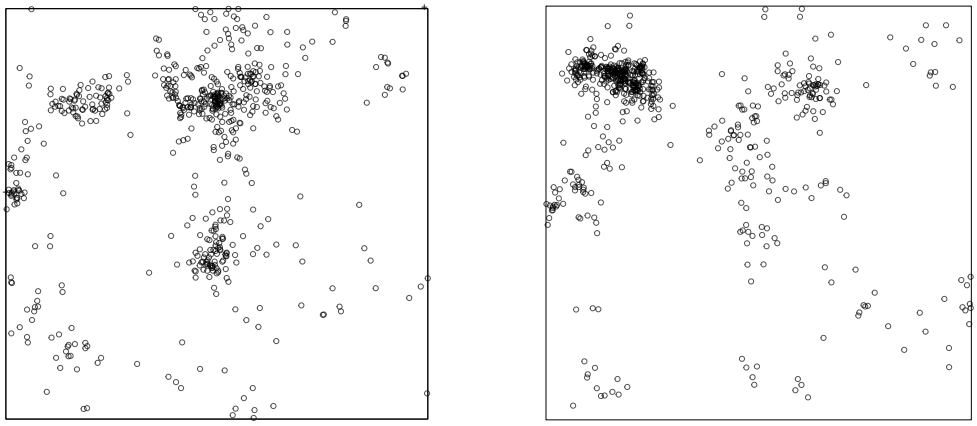
vilket ger $\mathbb{E}[\tilde{\Lambda}(X)] = \hat{\lambda}(X)$. I det andra steget genereras punkterna, villkorat på $\tilde{\Lambda}(X)$, som en Poissonprocess:

$$N(A) \,|\, \tilde{\Lambda}(X) \sim \mathrm{Poisson}\left( \int_A \tilde{\Lambda}(X)\, \mathrm{d}X \right).$$

Parametrarna $\sigma^2$, $\rho$, $\nu$ valdes genom grid-sökning som minimerar den kvadrerade avvikelsen mellan simulerad och observerad $L$-funktion; se appendix A.4 för full skattningsprocedur. Den bästa passformen ficks för

$$\sigma^2 = 2.356, \quad \rho = 9.1, \quad \nu = 0.6,$$

För illustration väljs simuleringar vars L-kurvor ligger nära genomsnittet av alla simuleringar, se figur 18. I figur 12 ses resultatet av metoden.



Figur 12: Två utfall av en Hierarkisk Poissonprocess.

Den biologiska tolkbarheten hos kovariaten, kombinerat med flexibiliteten från det latenta Guassiska fätet, gör metoden lämplig att modellera spatiala punktprocesser där både observerbar och latenta faktorer bidrar till klustring.

# 4 Anpassning av modeller

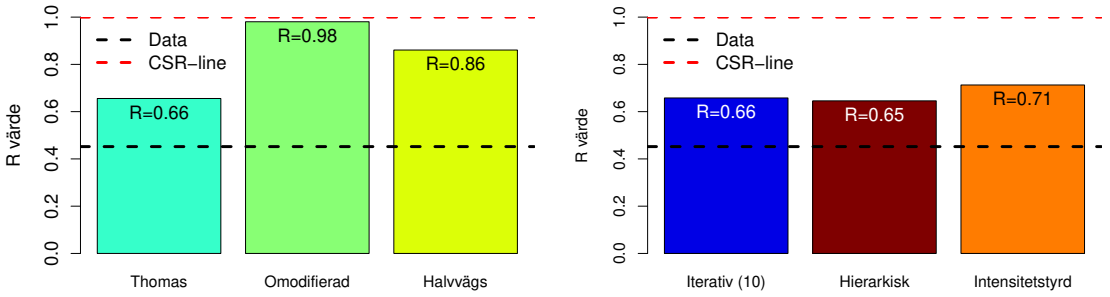
I ett försök att efterlikna klusteringen av trädväxt i datamängden, har flera olika modeller presenterats. För att undersöka hur väl dessa modeller passar vår datamängd används olika spatiala sammanfattade statistikor. Den huvudsakliga metoden är L-funktionen som, tillsammans med ett konfidenshölje, ger ett sätt att jämföra klustrandet i datamängden med utfallen från modellerna. Utöver L-funktionen används även Clark-Evans testet och några andra spatiala sammanfattningstatistikor för att jämföra de olika modellerna.

## 4.1 Clark-Evans

Clark-Evans testet avgör om modellerna är lika klustrade som VES13. För att genomföra testet används den inbyggda funktionen i spatstat paketet, vilket gav att $R = 0.45624$ för datamängden. Ett $R < 1$, indikerar att datamängden är klustrad och det låga p-värdet ($p < 2.2e - 16$) visar att detta resultatet är signifikant. Enligt Clark-Evans testet är datamängden därmed klustrad.

Detta test säger inget om hur klustringen ser ut, bara hur mycket punktmönstret är klustrat. I figur 13 syns det att alla modeller är mindre klustrade än datamängden som visas som en referenslinje. Utav Thomasprocesserna är den omodifierade nästintill helt CSR, och därmed nästan inte klustrad alls. Eftersom resultaten från detta testet visar att den omodifierade Thomasprocessen inte fångar exceptionella kluster alls, kommer denna inte vara en del av den resterande analysen. Vidare visar Clark-Evans testet att halvvägs Thomasprocessen är något mer klustrad än den omodifierade medan den vanliga Thomasprocessen är mest klustrad. Detta är ingen överraskning, då exakt detta är synligt i figur 6. Den intensitetstyrda modellen har ett R-värde som är mellan Thomasprocessen och halvvägs Thomasprocessen, så den är relativt klustrade men inte lika mycket som VES13, Thomasprocessen, den iterativa modellen eller den hierarkiska Poissonprocessen. Den iterativa modellen har samma R värde som Thomasprocessen, medan hierarkisk Poisson har ett något lägre värde. Detta betyder att det är dessa tre modeller som ligger närmst värdet för VES13. Dock betyder inte liknande R-värden att punktmönstret från modellerna liknar datamängden i sin helhet. Eftersom R-värdet enbart baseras på avståndet till närmaste granne, säger det därför främst något om graden av klustring eller regelbundenhet, och inte klustrens storlek eller

antal. Anledningen till att modellerna får andra R-värden än datamängdens beror sannolikt inte på avsaknaden av kluster eftersom Thomasprocessen är väldigt klustrad och har ett högre R-värde än VES13 utan istället på att våra modeller inte har tillräckligt många exceptionella kluster som minskar medelavståndet mellan punkterna. Däremot kanske andra statistikor kan visa att det är just detta som är skillnaden mellan våra modeller och datamängden.

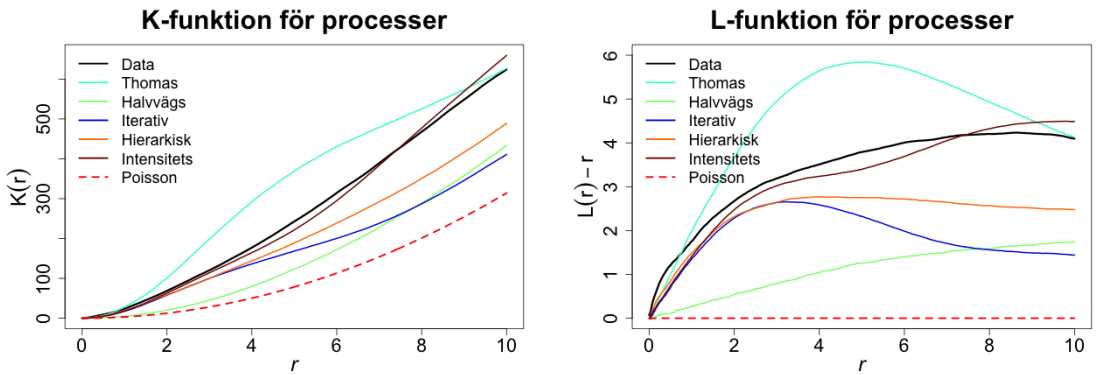



Figur 13: Figuren visar R-värden från Clark-Evans testet för datamängden, en simulering av de olika processerna samt en CSR-process.

## 4.2 K- och L- funktionerna

Både K- och L-funktionen beskriver mängden datapunkter inom ett visst avstånd från en godtycklig punkt, och är ett mått på hur klustrat ett punktmönster är. I figur 14 visas resultaten för både K- och L-funktionen, där grafen för L-funktionen är lättare att tolka, eftersom funktionen är lika med noll för Poissonprocessen. För att skatta K-funktionen användes kest funktionen i spatstat, vilket genererade figur 14.

Figur 6 visar att Thomasprocessen genererar flera kluster av liknande storlek, vilket medför att inga exceptionella kluster bildas. Modellen visar även en avsaknad av punkter mellan klustren, vilket indikerar att modellen är för klustrad för att efterlikna datamängden. Detta bekräftas av resultaten som visualiseras i figur 14, då L-funktionen visar att Thomasprocessen är mer klustrad än VES13 för nästan alla $r$. Det här kan vara på grund av definitionen av L-funktionen samt att alla punkter i mönstret befinner sig i kluster. Klustringen är som tydligast när $r = 5$, vilket sannolikt beror på att klusterradien för processen är 5. I kontrast så är halvvägs Thomasprocessen den modellen som är minst klustrad. Modellen växer i takt med $r$-värdet, vilket betyder att den är mer reguljär och saknar exceptionella kluster. Dessa två modeller är de som avviker mest från datamängden enligt denna statistika, dock betyder det inte att modellerna inte kan representera datamängden väl enligt andra mått.

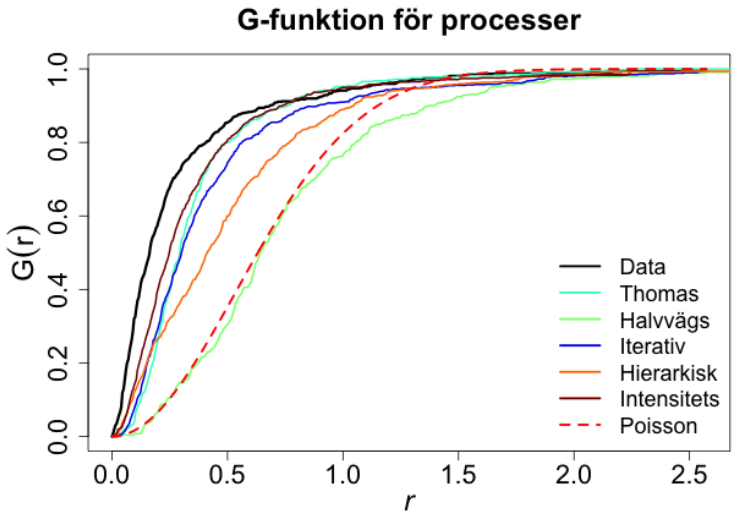



Figur 14: K- och L-funktionen för de olika punktprocesserna jämfört med en Poissonprocess.

Den hierarkiska modellen visar ett tillväxtmönster som liknar halvvägs Thomasprocessen, men den är mer klustrad vid $r \geq 4$. Som syns i figur 14, uppnår modellen maximal klustring vid $r = 3$, därefter avtar den svagt. Utöver halvvägs Thomasprocessen, är den iterativa processen den modellen som är minst klustrad. För låga r-värden liknar klustermönstret det som observerats i datamängden, men modellen avviker tydligt vid högre värden på r. Enligt L-funktionen så uppnår datamängden maximal klustring när $r = 8$, medan den iterativa modellen har mest klustring vid $r = 3$, vilket tyder på att den iterativa modellen har mindre radie på klustren än datamängden VES13. Detta leder till den mest välanpassade modellen; den intensitetesstyrda Thomasprocessen. I figur 14 syns det att modellen följer kurvan för datamängden nästan exakt för alla r. Detta är ingen överraskning eftersom modellen är optimerad just på K- och L-funktionerna.

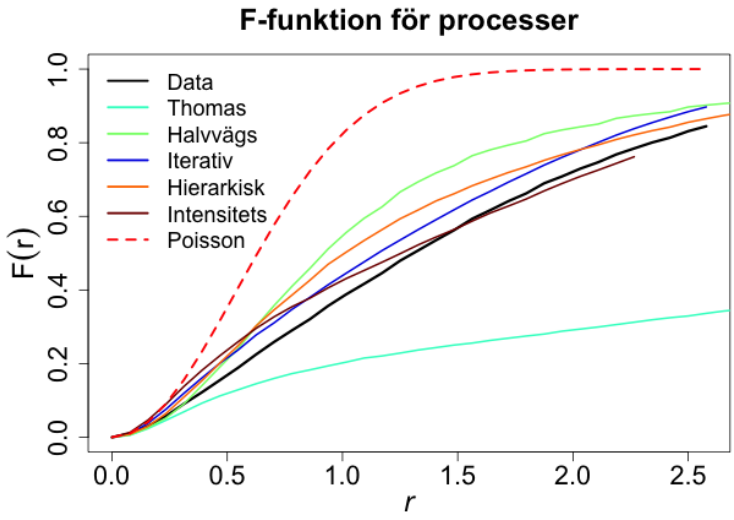
## 4.3 F-, G-, och J-funktionerna

Ett annat sätt att mäta avståndet till närmaste granne är med G-funktionen. Till skillnad från K- och L-funktionerna tar den inte hänsyn till fler punkter än de allra närmaste, därmed ger metoden mindre information om punkterna under analysen, vilket kan leda till andra resultat. Figur 15 visar att den intensitetsstyda Thomasprocessen och Thomasprocessen är de modeller som mest liknar datamängden enligt just denna statistika, medan halvvägs Thomasprocessen nästan exakt följer kurvan från Poissonprocessen och därmed är sämst anpassad till datamängden. Detta tyder på att alla modeller är klustrade, då de tydligt ligger över kurvan för Poissonprocessen, med undantag för halvvägs Thomasprocessen, som enligt denna analys är CSR. Resultatet är överraskande, eftersom punktmönstret för halvvägs Thomasprocessen i figur 6 inte ger någon indikation av att denna modellen skulle vara CSR. I figur 15 framgår det även att den iterativa Thomasprocessen har en liknande klustring till datamängden, dock är denna något mindre klustrad än både den intensitetsstyrda Thomasprocessen och Thomasprocessen, medan hierarkisk Poissonprocess är lite mer utspridd men fortfarande klustrad.




Figur 15: G-funktionen för de olika punktprocesserna jämfört med en Poissonprocess.

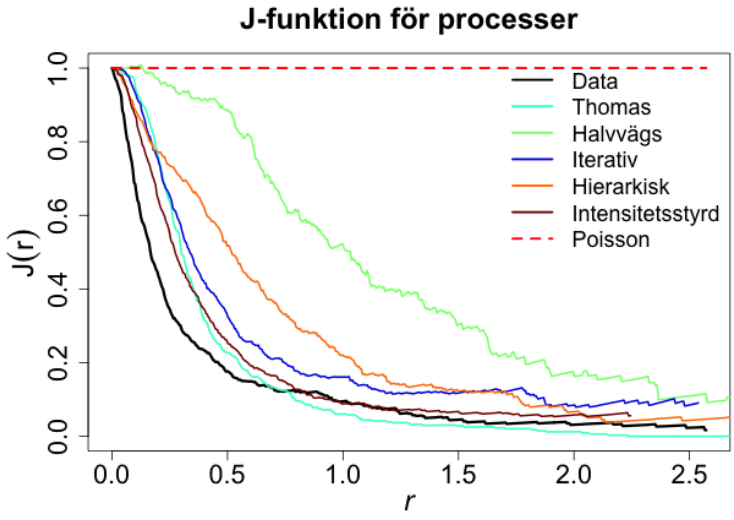
Tomrums-statistika F mäter hur mycket av punktmönstret som inte tas upp av punkter, med andra ord hur mycket tomrum det finns i mönstret. Figur 16 visar att $F(r)$-värdena för alla modeller ligger under motsvarande värde från Poissonprocessen för alla $r$. Detta tyder på att det finns mer tomrum, givet en godtyckligt punkt, i både punktprocesserna och i datamängden. Thomasprocessen avviker från de andra modellerna, sannolikt på grund av dess täta kluster och avsaknaden av punkter mellan klustren, vilket resulterar i mer tomrum.




Figur 16: F-funktionen för de olika punktprocesserna jämfört med en Poissonprocess.

Enligt denna statistika är den iterativa Thomasprocessen och intensitetsstyrda Thomasprocessen bäst, då de fångar tomrummet i datamängdens punktmönster mycket väl för alla värden på $r$. Därefter är hierarkisk Poissonprocess och halvvägs Thomasprocessen, på god väg att efterlikna datamängden. Detta är speciellt intressant eftersom resultaten från G-funktionen visade att halvvägs Thomasprocessen liknande en CSR Poissonprocess. Thomasprocessen är återigen för klustrad, vilket även framgick av resultatet från L-funktionen.

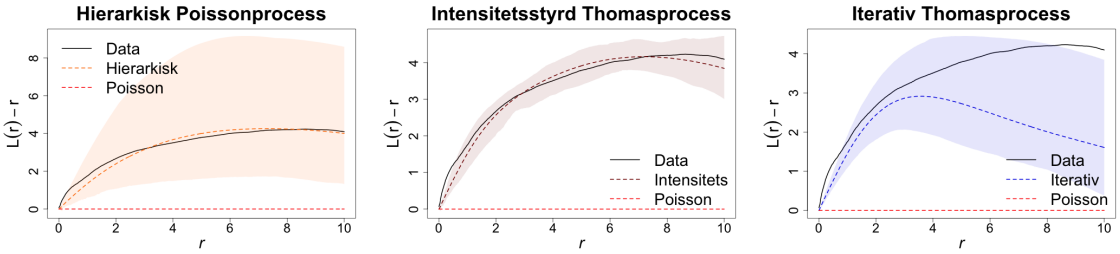
För att kombinera information från G- och F-funktionen, skattades även J-funktionen. Enligt resultaten från denna statistika, se figur 17, så är Thomasprocessen och den intensitetsstyrda Thomasprocessen bäst anpassade till datamängden för alla r. Även de andra punktprocesserna visar likheter med datamängden, med undantag för halvvägs Thomasprocessen som avviker från datamängden. Den iterativa Thomasprocessen verkar speciellt välanpassad för $r = 0.75$ och framåt. Det är värt att poängtera att ingen av modellerna följer linjen för den teoretiska Poissonprocessen, och eftersom både modellerna och datamängden ligger under linjen anses de vara klustrade.




Figur 17: J-funktionen för de olika punktprocesserna jämfört med en Poissonprocess.

## 4.4 Konfidenshölje för bästa punktprocess

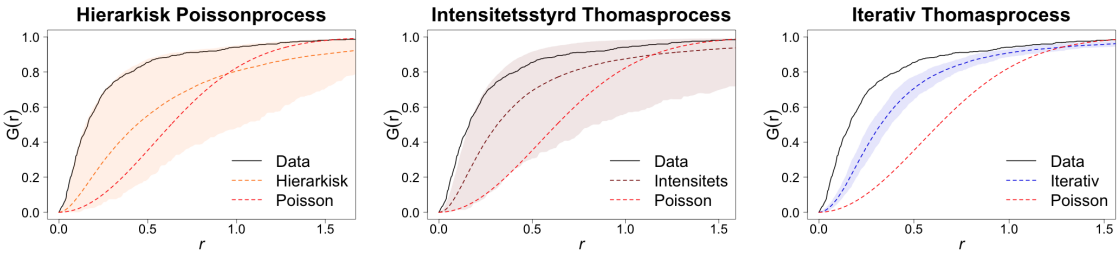
En ytligare metod för att visualisera hur modellerna fångar klustringen i datamängden är att lägga ett konfidenshölje runt punktprocesserna för de olika statistikorna. Ett 95% konfidenshölje skapas genom att simulera 100 utfall av respektive punktprocess, på detta tillämpas L-, G-, F-, J-funktionerna, och sist trimmas höljet till de värden som faller i intervallet $[2.5\%, 95\%]$.




Figur 18: Jämförelse av L-funktionen för datamängden, en Poissonprocess och de tre bästa anpassade punktprocesserna med tillhörande konfidenshöljen.

L-funktionen för de tre punktprocesserna som visade sig vara mest intressanta i tidigare analys illustreras i figur 18. Figuren visar att processernas konfidenshöljen mestadels täcker datamängden. Enligt denna statistika följer den hierarkiska Poissonprocessen datamängdens klustring väl, men eftersom konfidenshöljet är väldigt brett så leder det till en lägre grad av tillit i modellen. Den iterativa Thomasprocessen följer datamängden väl och har ett snävt konfidenshölje för $r \leq 3$, sedan avviker modellen och höljet blir mycket bredare. Klusterradien för den iterativa modellen har tidigare angetts som $r = 3$ och radien för datamängden är $r = 8$, vilket förmodligen är det som minskar modellens anpassning för större $r$. Processen med det minsta konfidenshöljet är den intensitetsstyrda Thomasprocessen. Detta beror på samma anledningen som gör att modellen fångar klustringen i datamängden väl; den är optimerad på L-funktionen. Höljet ger en indikation på modellens varians, och variansen kan minskas om en mer anpassad process önskas. Om variansen dras ned för mycket tappar modellen möjligheten att skapa unika punktmönster och återskapar istället den inmatade datamängden.

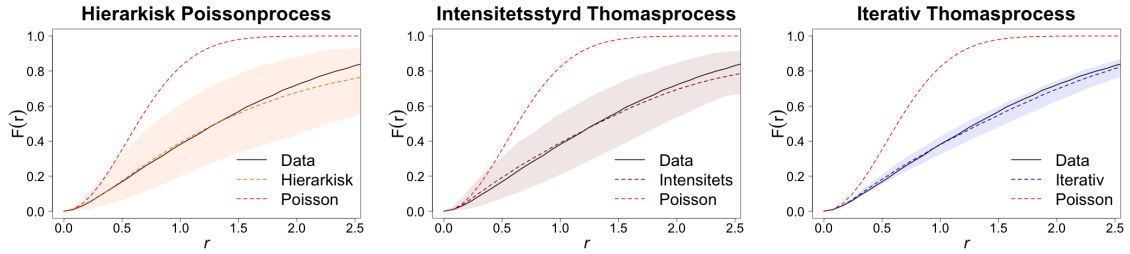
G-funktionen för de tre olika modellerna och deras konfidenshölje presenteras i figur 19. Höljet för den hierarkiska Poissonprocessen och den intensitetsstyrda Thomasprocessen har en övre gräns i linje med datamängden, medans den lägre gränsen sträcker sig förbi Poissonprocessens G-funktion. Att Poissonprocessen ligger inom intervallet säger oss att vi inte kan vara helt säkra på att punktmönstren som modellerna genererar inte är CSR. Konfidenshöljet för den iterativa Thomasprocessen är snävare och skiljer sig ifrån den teoretiska Poissonlinjen samtidigt som den skiljer sig från datamängdens. Detta betyder att modellen sannolikt inte fångar beteendet av ett CSR mönster eller datamängden enligt G-funktionen.




Figur 19: Jämförelse av G-funktionen för datamängden, en Poissonprocess och de tre bästa anpassade punktprocesserna med tillhörande konfidenshöljen.

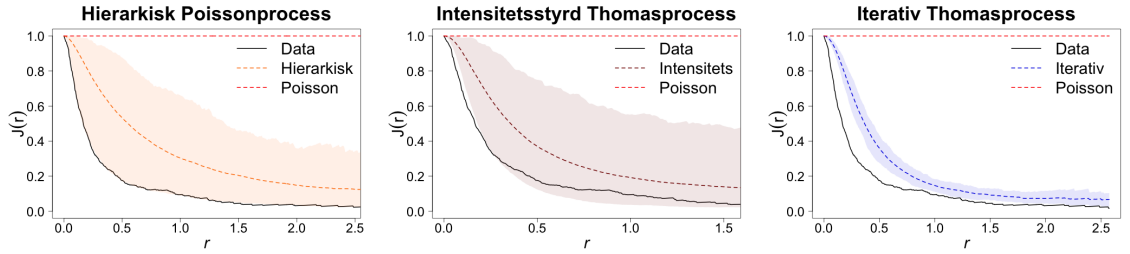
Tomrums statistika $F(r)$ för alla tre modeller i figur 20 följer datamängdens F-funktion väldigt bra för alla $r$. Detta tyder på att processerna och datamängden har en liknande mängd av tomrum som

datamängden. Alla tre modeller har i detta fall ett konfidenshölje som inte sträcker sig över den teoretiska Poissonlinjen, så alla tre modeller är klustrade. Det extra snäva höljet för den iterativa Thomasprocessen som även innehåller datamängden ger denna modellen styrka enligt detta test.




Figur 20: Jämförelse av F-funktionen för datamängden, en Poissonprocess och de tre bästa anpassade punktprocesserna med tillhörande konfidenshöljen.

Inom samtliga tre konfidenshöljen för den hierarkiska Poissonprocessen, den intensitetsstyrda Thomasprocessen och den iterativa Thomasprocessen i figur 21 gäller $J(r) < 1$, vilket innebär att punktprocesserna för dessa höljen visar klustring. Höljet för hierarkisk Poissonprocess och intensitetsstyrd Thomasprocess är båda relativt lika, men de skiljer sig åt i respektive medelkurva. Det är även intressant att notera hur datamängden omger höljena, då en liknande trend observeras i $G(r)$-funktionen som ingår i J-funktionen (ekv. 10). Utifrån denna figur har den iterativa Thomasprocessen det snävaste konfidenshöljet, men den täcker inte J-funktionen för datamängden för alla $r$.




Figur 21: Jämförelse av J-funktionen för datamängden, en Poissonprocess och de tre bästa anpassade punktprocesserna med tillhörande konfidenshöljen.

Den största fördelen och nackdelen med den intensitetsstyrda Thomasprocessen är att den är väldigt anpassad på den givna datamängden. Eftersom intensiteten skattas från datamängden kan modellen väldigt väl replikera observerade punktmönster. Detta gör dock att generaliserbarheten begränsas då modellen endast återskapar kluster liknande de man tränar den på. Den iterativa Thomasprocessen och den hierarkiska Poissonprocessen har däremot omvända styrkor, då de kan generera punktmönster med kluster som varierar i både placering och struktur, vilket gör dem mindre beroende av träningsdata. Med den iterativa modellen är det osannolikt att lyckas replikera datamängdens punktmönster, eftersom möjligheten till variation mellan utfall är hög. Detta betyder att modellerna är passande för olika användningsområden, både den itererade och hierarkiska modellen är flexibla och användbara med begränsad information om området, medan den intensitetsstyrda Thomasprocessen kan generera ett väldigt exakt punktmönster för att möjliggöra analys och förståelse när man har denna informationen.

# 5 Sammanfattning

Sammanfattningsvis har detta arbete med hjälp av spatiala punktprocesser undersökt olika modelleringsstrategier för att återskapa ett exceptionellt klustrat punktmönster, som är en datamängd från ERIKA-projektet kallat VES13. En rad modeller prövades, inklusive standard-, omodifierad-,

och en egenbyggd halvvägs Thomasprocess, en iterativ Thomasprocess, en intensitetsstyrd Thomasprocess samt en hierarkisk Poissonprocess. Syftet var att bygga dessa modeller och identifiera vilken modell som bäst återger de observerade klusteregenskaperna i datamängden. De första tre Thomasmodellerna visade sig inte kunna fånga mönstrets exceptionella karaktär, medan de andra modellerna fångade smådetaljerna bättre, särskilt i den iterativa Thomasprocessen och i den intensitetsstyrda Thomasprocessen, som tillät större flexibilitet i klusterbildning. Med hjälp av Clark-Evans testet och K-, L-, F-, G-, och J-funktionerna kunde modellerna analyseras, och resultaten visar att beroende på vilken statistikor som användes presterar olika punktprocesser bäst. Det vill säga de fångar olika aspekter av datamängdens punktmönster. Detta arbete ger därmed en god grund för fortsatt modellutveckling inom området exceptionella punktkluster.

Även om detta arbetet inte behandlar personuppgifter eller känslig information, så finns det samhälleliga aspekter att ta hänsyn till. Modeller skapade för klusteranalys kan tillämpas inom områden såsom ekologi och tillverkningsindustrin. Även om vi inte kan förutse exakt hur dessa modeller kommer att användas i framtiden, har de utvecklats med goda avsikter och medvetenhet om deras potentiella påverkan.

# 6 Framtida forskning

Under projektets gång har en del idèer om hur man skulle kunna utvidga modellerna och ta fram nya som kanske skulle fånga den exceptionella klustringen ännu bätte. Den itererade modellen skulle kunna utvecklas till att använda överlevnadsanalys så att ett visst antal punkter inte överlever i varje iteration. Reader föreslog att det är möjligt att tillämpa överlevnadsanalys på alla kontinuerliga variabler som endast har positiva värden och tillämpade detta på ett mått av avstånd mellan händelser [9]. Med detta i åtanke borde det vara möjligt att tillämpa denna metod även på vår iterativa modell. Som förbättring till vår hierarkiska Poissonprocess föreslår vi att ändra så att antalet dotterpunkter blir deterministiskt medan antalet föräldrarpunkter förblir stokastiska. Då blir optimeringen betydligt enklare att endast använda sig av derivatan för att optimera.

# Referenser


[1] R. J. Adler. The Geometry of Random Fields. *The Geometry of Random Fields*, 1 2007.

[2] A. Baddeley, E. Rubak, and R. Turner. *Spatial point patterns: methodology and applications with R.* CRC Press: Taylor & Francis Group, 2016.

[3] M. Berman and T. R. Turner. Approximating Point Process Likelihoods with GLIM. Technical Report 1, 1992.

[4] E. Dovers, J. Stoklosa, and D. I. Warton. Fitting Log-Gaussian Cox Processes Using Generalized Additive Model Software. *American Statistician*, 2024.

[5] A. Gezerlis. Numerical Methods in Physics with Python. *Numerical Methods in Physics with Python*, 8 2020.

[6] J. Møller and R. Plenge Waagepetersen. *Statistical Inference and Simulation for Spatial Point Processes*, volume 21. Chapman and Hall, 2003.

[7] M. Kuronen, A. Särkkä, M. Vihola, and M. Myllymäki. Hierarchical log Gaussian Cox process for regeneration in uneven-aged forests. *Environmental and Ecological Statistics*, 29(1):185–205, 3 2022.

[8] G. Last and M. Penrose. Poisson Processes. *Lectures on the Poisson Process*, pages 19–25, 10 2017.

[9] S. Reader. Using survival analysis to study spatial point patterns in geographical epidemiology. *Social Science & Medicine*, 50(7-8):985–1000, 4 2000.

# A Teori

## A.1 Log Guassian Cox process, LGCP

Log Guassian Cox processen (LGCP) är en Poisson styrd punktprocess där intensiteten är stokastisk och ges av en Guassisk process. Mer formellt låter vi $\{Y(\xi) : \xi \in \mathbb{R}^d\}$ vara ett latent Gaussiskt fält med medelfunktion $m(\xi)$ och kovariansfunktion $c(\xi, \eta)$. Intensitetsfunktionen definieras då som:

$$\rho(\xi) = \exp(Y(\xi))$$

Vilket innebär att intensiteten varierar över rummet enligt den latenta processen [6] LGCP tillåter modellering av klusterbildning genom att låta intensiteten vara rumsligt korrelerad. Intensiteten och parkorrelations funktionen ges av:

$$\rho(\xi) = \exp(m(\xi) + c(\xi, \xi)/2), \quad g(\xi, \eta) = \exp(c(\xi, \eta))$$

Där $g(\xi, \eta)$ beskriver den rumsliga korrelation mellan intensitet [6] (75). LGCP anpassar sig till lokala variationer i fröspridningen. Det latenta fältet i LGCP kan anta flexibla former, vilket gör att modellen finner finare variationer i klustermönstret.

## A.2 Berman och Turner

Berman och Turner [3] metoden är följande: Fönstret $W \subset \mathbb{R}^2$ delas in i ett regelbundet rutnät av pixlar $A_{i,j}$ med gemensam area $\Delta A$. Varje ruta representeras av sin pixelpunkt $(\mathbf{X}_i)$, och kvadraturvikten sätts till

$$w_{i,j} \;=\; \iint_{A_{i,j}} dx\, dy \;=\; \Delta A.$$

Då gäller approximationen

$$\iint_W \lambda(x, y)\, dx\, dy \;\approx\; \sum_{i,j} w_{i,j}\, \lambda(x_{i,j}, y_{i,j}),$$

vilket är just Berman–Turner-kvadraturen. Genom att registrera

$$N_{i,j} \;=\; \begin{cases} 1, & \text{om en punkt observeras i pixel } (i,j), \\ 0, & \text{annars,} \end{cases}$$

erhålls den viktade Poisson-likelihooden

$$\sum_{i,j} w_{i,j} \left( \frac{N_{i,j}}{w_{i,j}} \log \lambda(x_{i,j}, y_{i,j}) - \lambda(x_{i,j}, y_{i,j}) \right),$$

som kan skattas med verktyg för generaliserade linjära modeller.

Vi modellerar intensiteten log-linjärt,

$$\lambda(x_{i,j}, y_{i,j}) \;=\; \exp\big(\eta_{i,j}\big), \qquad \eta_{i,j} = \beta_0 + \sum_k \beta_k\, C_k(x_{i,j}, y_{i,j}),$$

där $C_k$ är valfria kovariater och $\beta = (\beta_0, \beta_1, \dots)$ är okända parametrar. Satt i uttrycket ovan fås den upp till en konstant Poisson-log-likelihood

$$\ell(\beta) = \sum_{i,j} \Big( N_{i,j}\, \eta_{i,j} - w_{i,j} \exp(\eta_{i,j}) \Big).$$

Maximering av $\ell(\beta)$ görs via en Poisson-GLM med respons $N_{i,j}/w_{i,j}$ och vikter $w_{i,j}$ – ger $\hat{\beta}$. Det skattade intensitetsfältet erhålls därefter som

$$\hat{\lambda}(x, y) = \exp\Big( \hat{\beta}_0 + \sum_k \hat{\beta}_k\, C_k(x, y) \Big),$$

vilket kan utvärderas i varje pixel för visualisering, simulering eller vidare inferens.

## A.3 Generalized Additive Model i Generaliserad Poisson Regression

En GAM introducerar en anpassningsbar komponent i den linjära prediktorn som kan användas i till exempel Berman och Turner kvadratur och låter intensitetsfältet $\lambda(s)$ svara mot mönstret i data som kvarstår efter att observerade kovariater beaktats. Detta ger en mer menings full skattning av $\lambda$ även när kovariater misslyckas att förklara variationen[4].

Antag en punkt process på $W \subset \mathbb{R}^2$ med

$$\lambda(\mathbf{s}) = \exp\{\eta(\mathbf{s})\}, \quad \eta(\mathbf{s}) = \beta_0 + C(\mathbf{s})\beta + \xi(\mathbf{s})$$

där $C(\mathbf{x})$ är en rad kovariater och $\xi(\mathbf{s})$ ett latent fält som modellerar strukturer som kovariaterna inte fångar. För att approximera det latenta fältet $\xi$ enligt [4] skrivs det som

$$\xi(\mathbf{s}) \;\approx\; \sum_{j=1}^{k} B_j(\mathbf{s})\, b_j, \qquad b_j \overset{\text{iid}}{\sim} N(0, \sigma^2).$$

här introduceras en fast uppsättning punkter $s_1^*, \ldots, s_k^* \in W$ kallade knots vilka fungerar som stöd för bas funktionen $B_j(\mathbf{s})$. Knotpunkterna bestämmer en kovariansmatris

$$C^* = [C(s_r^*, s_\ell^*)]_{r,\ell=1}^{k}, \quad C(s,t) = \left(1 + \frac{\|s-t\|^2}{\rho}\right) \exp\left(-\frac{\|s-t\|^2}{\rho}\right),$$

Enligt [4] definieras då basisfunktionen enligt

$$B_j(s) = \sum_{r=1}^{k} C(s, s_r^*) \left[C^{*\,-1/2}\right]_{rj}, \quad j = 1, \ldots, k.$$

För varje datapunkt $s_i$ räknas alla $B_j(s_i)$; värdena bildar rad $i$ i designmatrisen $\mathcal{Z}$.

Konstruktionen ger

$$\mathrm{Cov}\{\xi(s), \xi(t)\} = \sigma^2 C(s,t)$$

samtidigt som koefficienterna $b_j$ blir oberoende $b_j \sim \mathcal{N}(0, \sigma^2)$. Den fullständiga prediktorn blir

$$\eta(s_i) = X_i\beta + \sum_{j=1}^{k} B_j(s_i) b_j$$

utvärderas vid varje datapunkt $s_i$, där $X_i$ är rad $i$ av kovariatmatrisen.

## A.4 Parameter skattning av Matérn-kovarians via förlust funktion

För att skatta parametrarna i Matérn-kovarians $(\sigma^2, \rho, \nu)$ genomfördes en grid sökning inom ett grovt intervall. För varje kombination simuleras 100 realiseringar av den hierarkiska Poissonprocess och för varje simulering beräknas även L funktion. Sedan, för varje kombination av $(\sigma^2, \rho, \nu)$, beräknas den genomsnittliga L kurvan $(\overline{L}(r))$ av dom 100 simuleringarna:

$$\overline{L_{\text{sim}}}(r) = \frac{1}{N_{\text{sim}}} \sum_{i=1}^{N_{\text{sim}}} [L_{\text{sim},i}(r) - r]$$

varpå $\overline{L}(r)$ jämfördes med L funktionen från det observerade klustret i datamängden genom följande förlust funktion:

$$\text{loss}(\sigma^2, \rho, \nu) = \sum_k \left(\overline{L}(r_k) - L_{\text{obs}}(r_k)\right)^2,$$

# B Kod för punktprocesser

## B.1 Thomasprocess (R)

```r
fitT <- kppm(X ~ 1, "Thomas") # datamängd X
X_thomas <- simulate(fitT,nsim=1)
X_thomas <- X_thomas[[1]] # x,y points
```

## B.2 Halvvägs Thomasprocess (R)

```r
lambda_avg <- 2 / (1/lambda_p_hat + 1/0.003943241)
mu_avg <- sqrt(mu_hat * 139.7962)
sigma_avg <- sqrt(sigma_hat * 2.097576813)
num_parents_avg <- rpois(1, lambda_avg * area.owin(window)) # skattad
parents_avg <- runifpoint(num_parents_avg, win = window) # Simulate intensity

offspring_x <- c()
offspring_y <- c()

for (i in 1:num_parents_avg) {
  num_offspring <- rpois(1, mu_avg)
  if (num_offspring > 0) {
    x_offset <- rnorm(num_offspring,mean= 0, sd = sigma_avg)
    y_offset <- rnorm(num_offspring,mean= 0, sd = sigma_avg)

    new_x <- parents_avg$x[i] + x_offset
    new_y <- parents_avg$y[i] + y_offset

    valid <- inside.owin(new_x, new_y, window) # Points inside window
    offspring_x <- c(offspring_x, new_x[valid])
    offspring_y <- c(offspring_y, new_y[valid])
  }
}

halfway_thomas <- ppp(offspring_x, offspring_y, window=window)
```

## B.3 Iterativ Thomasprocess (Python)

Iterativ Thomas växer fram i iterationer. Nedan kod har skalats ner för plats. Främst plot av punkter och form av data tagits bort. Full kod vid förfrågan.

```python
def iter_thomas_unif(outx, outy, lam=5, iter=0):
    all_points = np.vstack([outx, outy]).T
    x_history=[outx]
    y_history=[outy]

    count_all = count_all - 1
    size_unif = int(np.array(outx).shape[0]*0.05)
    index = np.random.randint(0,all_points.shape[0], size=size_unif)
    x_new_10_parent = [all_points[index][:,0]]
    y_new_10_parent = [all_points[index][:,1]]
```

```
    outx = np.concatenate(x_new_10_parent).ravel().tolist()
    outy = np.concatenate(y_new_10_parent).ravel().tolist()
    new_x = np.array(outx)
    new_y = np.array(outy)
    n_children = rng.poisson(lam=lam, size=new_y.shape[0])

    for i in range(new_y.shape[0]):
        x_child = rng.normal(new_x[i], size=n_children[i])
        y_child = rng.normal(new_y[i], size=n_children[i])
        x_history.append(x_child)
        y_history.append(y_child)

    outx = np.concatenate(x_history).ravel().tolist()
    outy = np.concatenate(y_history).ravel().tolist()

    return outx, outy

for i in range(maxiter):
    outx, outy = iter_thomas_unif_pic(outx, outy, iter=i)
    save_points[i] = [outx,outy]
```

## B.4 Intensitetsstyrd Thomasprocess (R)

```
library(spatstat); library(pracma)

p <- "C:/.../VES13_small.csv"
data <- read.csv(p)
win <- owin(c(0, 40), c(0, 40))
obs_ppp <- ppp(data$x, data$y, window = win)

# summary stats ------------------------------------------
L_obs <- Lest(obs_ppp, correction = "Ripley")
r_vals <- L_obs$r
Delta_obs <- L_obs$iso - L_obs$r

G_obs <- Gest(obs_ppp, correction = "rs")
g_vals <- G_obs$r
G_vec <- G_obs$rs

F_obs <- Fest(obs_ppp, correction = "rs")
f_vals <- F_obs$r
F_vec <- F_obs$rs

G_onF <- approx(g_vals, G_vec, xout = f_vals, rule = 2)$y
J_vec <- (1 - G_onF) / (1 - F_vec)

# LGCP field ---------------------------------------------
field_fun <- function(x, y) 4 - 1.5 * ((x / 40) - .5)^2 + 2 * ((y / 40) - .5)^2
m <- as.im(field_fun, W = win)
X <- rLGCP("gauss", m, var = 0.6, scale = 10, win = win, saveLambda = TRUE)
Lambda <- attr(X, "Lambda")
lam_nrm <- (Lambda - min(Lambda)) / (max(Lambda) - min(Lambda)) * 2.5
lam_fun <- as.function(lam_nrm)
contour(lam_nrm, nlevels = 10, main = "", xlab = "x", ylab = "y")

# generate children --------------------------------------
make_children <- function(parents, A, t, sigma) {
```

```r
  n_child <- pmax(0, floor(A * (parents$lambda - t)))
  kids <- do.call(rbind, lapply(seq_len(nrow(parents)), function(i) {
    if (n_child[i] > 0)
      data.frame(x = rnorm(n_child[i], parents$x[i], sigma),
                 y = rnorm(n_child[i], parents$y[i], sigma))
  }))
  if (!is.null(kids))
    kids <- kids[inside.owin(kids$x, kids$y, win), , drop = FALSE]
  kids
}

# loss function ------------------------------------------
loss_factory <- function(parents) {
  function(A, t, sigma) {
    kids <- make_children(parents, A, t, sigma)
    pts <- rbind(parents[, c("x", "y")], kids)
    if (nrow(pts) < 5) return(Inf)
    sim_ppp <- ppp(pts$x, pts$y, window = win)

    L_sim <- Lest(sim_ppp, correction = "Ripley")
    D_sim <- L_sim$iso - L_sim$r
    loss_L <- trapz(r_vals,
      (approx(L_sim$r, D_sim, xout = r_vals, rule = 2)$y - Delta_obs)^2)

    G_sim <- Gest(sim_ppp, correction = "rs")
    loss_G <- trapz(g_vals,
      (approx(G_sim$r, G_sim$rs, xout = g_vals, rule = 2)$y - G_vec)^2)

    F_sim <- Fest(sim_ppp, correction = "rs")
    F_int <- approx(F_sim$r, F_sim$rs, xout = f_vals, rule = 2)$y
    loss_F <- trapz(f_vals, (F_int - F_vec)^2)

    G_int <- approx(G_sim$r, G_sim$rs, xout = f_vals, rule = 2)$y
    J_int <- (1 - G_int) / (1 - F_int)
    loss_J <- trapz(f_vals, (J_int - J_vec)^2)

    loss_L + loss_G + loss_F + loss_J
  }
}

# grid search --------------------------------------------
A_seq <- seq(10, 100, length.out = 15)
t_seq <- seq(0, 1, length.out = 15)
s_seq <- seq(0.2, 3, length.out = 15)
n_par <- c(86, 86, 86)

# main loop ----------------------------------------------
for (n_f in n_par) {
  parents <- data.frame(x = runif(n_f, 0, 40), y = runif(n_f, 0, 40))
  parents$lambda <- lam_fun(parents$x, parents$y)
  loss_fn <- loss_factory(parents)

  best_loss <- Inf
  best_par <- c(NA, NA, NA)
  for (A in A_seq) for (t in t_seq) for (s in s_seq) {
    Lval <- loss_fn(A, t, s)
    if (Lval < best_loss) { best_loss <- Lval; best_par <- c(A, t, s) }
  }

  kids <- make_children(parents, best_par[1], best_par[2], best_par[3])
```

```r
  plot(lam_nrm, main = sprintf(
    "n_f=%d: A=%.1f, t=%.2f, =%.2f, loss=%.3g",
    n_f, best_par[1], best_par[2], best_par[3], best_loss))
  points(parents$x, parents$y, pch = 16)
  if (!is.null(kids)) points(kids$x, kids$y, pch = 16)

  pts_all <- rbind(parents[, c("x", "y")], kids)
  all_ppp <- ppp(pts_all$x, pts_all$y, window = win)

  # point pattern only
  plot(NA, xlim = c(0, 40), ylim = c(0, 40), xlab = "x", ylab = "y")
  points(parents$x, parents$y, pch = 1, col = "black")
  if (!is.null(kids)) points(kids$x, kids$y, pch = 1, col = "black")

  # L ------------------------------------------------------
  env_L <- envelope(all_ppp, Lest, nsim = 99, correction = "Ripley", verbose = FALSE)
  plot(env_L$r, env_L$obs - env_L$r, type = "l", col = "blue", lwd = 2,
       xlab = "r", ylab = expression(L(r) - r),
       main = sprintf("L, n_f = %d", n_f))
  lines(env_L$r, env_L$hi - env_L$r, lty = 2)
  lines(env_L$r, env_L$lo - env_L$r, lty = 2)
  lines(r_vals, Delta_obs, col = "black", lwd = 2)
  abline(h = 0, lty = 3)
  legend("topleft", legend = c("sim", "obs"),
         col = c("blue", "black"), lty = 1, bty = "n")

  # G ------------------------------------------------------
  G_full <- Gest(all_ppp, correction = "rs")
  plot(g_vals, G_vec, type = "l", lwd = 2,
       xlab = "r", ylab = "G(r)", main = sprintf("G, n_f = %d", n_f))
  lines(G_full$r, G_full$rs, col = "blue", lwd = 2)
  legend("bottomright", legend = c("obs", "sim"),
         col = c("black", "blue"), lty = 1, bty = "n")

  # F ------------------------------------------------------
  F_full <- Fest(all_ppp, correction = "rs")
  plot(f_vals, F_vec, type = "l", lwd = 2,
       xlab = "r", ylab = "F(r)", main = sprintf("F, n_f = %d", n_f))
  lines(F_full$r, F_full$rs, col = "blue", lwd = 2)
  legend("bottomright", legend = c("obs", "sim"),
         col = c("black", "blue"), lty = 1, bty = "n")

  # J ------------------------------------------------------
  G_i <- approx(G_full$r, G_full$rs, xout = f_vals, rule = 2)$y
  F_i <- approx(F_full$r, F_full$rs, xout = f_vals, rule = 2)$y
  J_sim <- (1 - G_i) / (1 - F_i)
  plot(f_vals, J_vec, type = "l", lwd = 2,
       xlab = "r", ylab = "J(r)", main = sprintf("J, n_f = %d", n_f))
  lines(f_vals, J_sim, col = "blue", lwd = 2)
  legend("bottomright", legend = c("obs", "sim"),
         col = c("black", "blue"), lty = 1, bty = "n")
}
```

## B.5 Hierarkisk Poissonprocess (R)

```r
library(spatstat)
library(mgcv)
library(parallel)
```

```r
events   <- read.csv("C:/.../VES13_small.csv")  # x,y
bigTrees <- read.csv("C:/.../VES13_large.csv")  # x,y,dbh

xrange <- range(events$x)
yrange <- range(events$y)
area   <- diff(xrange) * diff(yrange)
W      <- owin(xrange, yrange)
t      <- 4
r0     <- 1
mu     <- 1.8
sigma  <- 0.8

# kovariat funktionerna -----------------------------
computeC  <- function(sx, sy) {
  d <- sqrt((sx - bigTrees$x)^2 + (sy - bigTrees$y)^2)
  sum(bigTrees$dbh * exp(-d / t))
}

lognorm_k <- function(r) {
  ifelse(r > r0,
         dlnorm(r - r0, meanlog = mu, sdlog = sigma),
         0)
}

computeC2 <- function(sx, sy) {
  d <- sqrt((sx - bigTrees$x)^2 + (sy - bigTrees$y)^2)
  sum(bigTrees$dbh * lognorm_k(d))
}

# för att förstå nedan se Berman Turner och Dovers --
q        <- 100000
quad     <- data.frame(x = runif(q, xrange[1], xrange[2]),
                       y = runif(q, yrange[1], yrange[2]),
                       pt = 0,
                       wt = area / q)
events$pt <- 1
events$wt <- 1
dat       <- rbind(events, quad)

dat$C  <- mapply(computeC,  dat$x, dat$y)
dat$C2 <- mapply(computeC2, dat$x, dat$y)

fit <- gam(pt ~C + s(x, y, bs = "gp", k = 200) +
             offset(log(wt)),
           data   = dat,
           family = poisson,
           method = "REML")


print(summary(fit))
print(AIC(fit))
W      <- owin(xrange = range(events$x), yrange = range(events$y))

L_emp  <- Lest(ppp(events$x, events$y, window = W),
               lambda     = im_intensity,
```

```
                correction = "trans")
r_vals <- L_emp$r
L_obs  <- L_emp$trans
matern_corr <- function(r, rho, nu) {
  z   <- sqrt(2*nu) * r / rho
  out <- (2^(1-nu)/gamma(nu)) * z^nu * besselK(z, nu)
  out[r == 0] <- 1
  out
}

# --- närsökning, ändra 30 till 10 för kortare sökning--
grid <- expand.grid(
  sigma2 = seq(2.756, 3.1, length = 30),
  rho    = seq(7,12,   length = 30),
  nu     = seq(0.4,1,length = 30)
)

n_iter <- nrow(grid)
loss   <- numeric(n_iter)
n_sim  <- 100    # antal simuleringar per parameterkombination

for(i in seq_len(n_iter)) {
  p <- grid[i, ]
  L_sim_mat <- replicate(n_sim, {
    lgcp_ppp <- rLGCP("matern",
                      mu    = log(im_intensity),
                      var   = p$sigma2,
                      scale = p$rho,
                      nu    = p$nu,
                      win   = W)
    Lest(lgcp_ppp, lambda = im_intensity, correction = "trans")$trans
  })

  L_pred <- rowMeans(L_sim_mat)
  loss[i] <- sum((L_obs - L_pred)^2, na.rm = TRUE)
  cat(sprintf("Iter %3d/%3d  ²=%.3f  =%.3f  =%.3f  loss=%.4f\n",
              i, n_iter, p$sigma2, p$rho, p$nu, loss[i]))
}
best        <- grid[which.min(loss), ]
field.var   <- best$sigma2
field.range <- best$rho
nu          <- best$nu

print(best)
cat("Minsta loss =", min(loss), "\n")
```

# C BGMM: Den bortglömda modellen

En alternativ metod för att modellera de exceptionella kluster som observeras i figur 1 är att använda sig av en Gaussian mixture model. Denna modell lär sig av data för att sedan generera liknande mönster. Metoden är lämplig när man saknar en känd priori distribution och vill använda klustrad data för att generera motsvarande kluster. Fördelningen möjliggör en anpassning av antalet kluster, och låter varje kluster representeras av en normalfördelning. För att få tätare kluster

kan ett kluster absorbera punkter från närliggande kluster.

Låt $\{x_i\}_{i=1}^N$ beteckna data för dotterpunkterna där varje punkt representeras av en femdimension vektor

$$x_i = \left(s_{i,1},\, s_{i,2},\, r_i,\, d_i^{(1)},\, w_i\right) \in \mathbb{R}^5.$$

Eller andra mer relevanta kvoariater.

Här utgör $(s_{i,1},\, s_{i,2})$ dotterpunkternas spatiala position, medan de övriga är kovariater som härleds från träddatan enligt

- $r_i = \min_{1 \le j \le M} \|s_i - t_j\|$, där $t_j$ är positionen för träd $j$,
- $d_i^{(1)}$ motsvarar diametern (DBH) hos den närmsta föräldrarpunkten,
- $w_i$ definieras som en viktad summa av trädens DBH-värden med avseende på avstånd.

För att generera modellen antar vi att dotterpunkterna genereras från en blandning av $K$ guassiska fördelningar, där varje kluster antas ha en specifik medelvärdes vektor $\mu_k$ och kovariansmatris $\Sigma_k$. Sannolikheten för att en datapunkt $x_i$ observeras ges av

$$p(x_i) = \sum_{k=1}^{K} \pi_k \, \mathcal{N}(x_i \mid \mu_k, \Sigma_k),$$

där

$$\mathcal{N}(x \mid \mu_k, \Sigma_k) = \frac{1}{(2\pi)^{5/2} |\Sigma_k|^{1/2}} \exp\!\left( -\frac{1}{2} (x - \mu_k)^\top \Sigma_k^{-1} (x - \mu_k) \right),$$

och $\pi_k$ är mixningsvektorn. Parametrarna $\{\pi_k, \mu_k, \Sigma_k\}$ uppskattas med Bayesiask inferens (variational Bayes). Efter detta tilldelas varje datapunkt $x_i$ det kluster som maximerar sannolikheten

$$z_i = \arg \max_{1 \le k \le K} \pi_k \, \mathcal{N}(x_i \mid \mu_k, \Sigma_k).$$

För att skapa täta kluster används en sammanslagnings strategi av klustren, där medelvärde av klustren definieras som

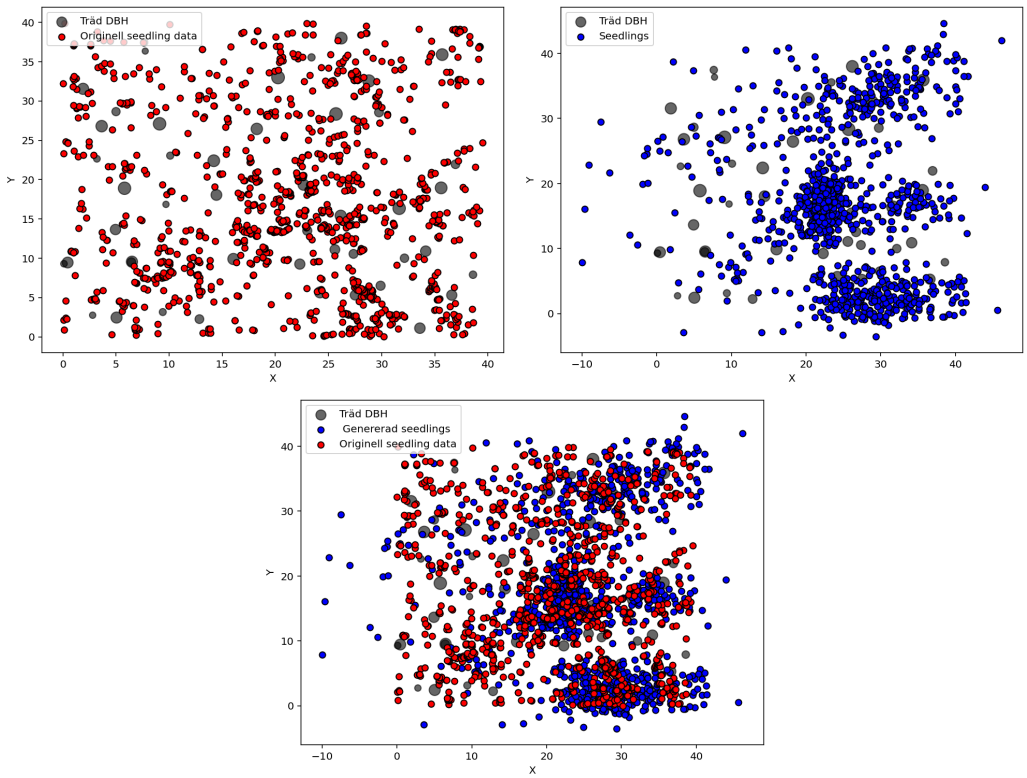
$$\mu_k^{xy} = (\mu_{k,1},\, \mu_{k,2}).$$

Om avståndet mellan två kluster, $\|\mu_i^{xy} - \mu_j^{xy}\|$, är under ett tröskel värde $\tau$ så slås de samman. Istället för flera kluster får vi nu ett.

För att generera nya dotterpunkter definieras andelen punkter för varje kluster som

$$p_k = \frac{n_k}{N}.$$

där $n_k$ är antalet dotterpunkter som kluster $k$ har. En ny dotterpunkt genereras genom att först välja ett kluster med sannolikheten $p_k$ och därefter dra ett prov från den guassiska fördelningen $\mathcal{N}(\mu_k, \Sigma_k)$.




Figur 22: BGMM resultat, röda punkter visar observerad data, blå punkter genererad data