CHALMERS

GÖTEBORGS UNIVERSITET

Extremvärdesanalys av nederbördstrender i Sverige

Extreme value analysis of precipitation trends in Sweden

Examensarbete för kandidatexamen i matematik vid Göteborgs universitet
Kandidatarbete inom civilingenjörsutbildningen vid Chalmers

Hanna Björklund
Elin Gustafsson
Olof Morsing
Emil Nyström

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

Extremvärdesanalys av nederbördstrender i Sverige

Kandidatarbete i matematik inom civilingenjörsprogrammet Teknisk fysik vid Chalmers
Elin Gustafsson

Examensarbete för kandidatexamen i matematisk statistik inom Matematikprogrammet vid Göteborgs universitet
Olof Morsing
Emil Nyström

Kandidatarbete i matematik inom civilingenjörsprogrammet Bioteknik vid Chalmers
Hanna Björklund

Handledare: Holger Rootzén

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

Förord

Denna rapport utgör vårt kandidatarbete vid Chalmers tekniska högskola och Göteborgs universitet. Vi vill tacka vår handledare Holger Rootzén för hans hjälp och vägledning igenom projektet.

Under projektets gång har det veckovis förts loggar över hur projektet fortskrider. Det har gjorts med ett rullande schema där varje medlem haft dagboken var fjärde vecka. Som grupp är vi eniga om att alla har gjort sin del i projektet. Följande bidragsrapport visar vem som skrivit vilken del i rapporten.

Bidragsrapport		
-	Rubrik	Författare
-	Populärvetenskaplig presentation	Hanna och Olof
-	Sammandrag	Elin
-	Abstract	Elin
1	Inledning	Elin och Hanna
1.1	Syfte och frågeställning	Elin
1.2	Avgränsningar	Elin och Olof
2	Teori	Emil
2.1	Generaliserade extremvärdesfördelningen	Emil
2.2	Block maxima	Emil
2.3	Generaliserade Pareto-fördelningen	Elin och Emil
2.4	Tröskelmetoden	Elin och Emil
2.5	Deklustring	Emil
2.6	Poissonprocesser	Emil
2.7	Trender	Emil
2.8	Maximum likelihood-skattning	Emil
2.9	Likelihoodkvot-test	Emil
2.10	Diagnostik	Elin och Emil
2.10.1	GEV-fördelningen	Emil
2.10.2	GP-fördelningen	Emil
2.10.3	Q-Q-plott med p-värden	Emil
2.11	Tolkning	Emil
3	Metod	Elin, Emil och Olof
3.1	Datainsamling	Elin, Emil och Olof
3.2	Dataanalys	Olof
3.2.1	Block maxima-metoden	Emil och Olof
3.2.2	Tröskelmetoden	Emil och Olof
3.2.3	Anpassning av inhomogen Poissonprocess	Emil och Olof
3.3	Mätfel	Emil och Hanna
4	Resultat	Elin, Emil och Olof.
4.1	Värde på ξ	Emil och Olof
4.2	Modeller med signifikanta trender	Emil, Hanna och Olof
5	Diskussion	Alla
5.1	Samhälleliga och etiska aspekter	Elin
5.2	Slutsatser	Elin och Emil
-	Användande av AI	Elin och Olof
A	Appendix 1 – Tabeller	Emil och Olof
B	Appendix 2 – Q-Q plottar	Emil och Olof
C	Appendix 3 – Kod	Olof
-	Korrekturläsning	Alla
-	Dataanalys	Olof

Populärvetenskaplig presentation

Vi lever i en tid av stigande temperatur på grund av klimatförändringar där ökande temperatur leder till kraftigare och mer frekventa extrema väder globalt. Mer extrem nederbörd är ett exempel på de typer av väderhändelser som kan bli vanligare i ett varmare klimat. Varmare luft kan hålla mer vattenånga, vilket i sin tur möjliggör kraftigare regnfall. Den ökade temperaturen kan också påskynda vattnets kretslopp och därmed påverka nederbördens intensitet. Sådana förändringar ökar inte bara risken för översvämningar, utan kan även bidra till jordskred, erosion, förorenade vattentillgångar och skador på både infrastruktur och jordbruk. För att kunna förstå, förutse och hantera dessa risker är det avgörande att kvantifiera nederbördsmängder och analysera hur de förändras över tid.

Med bakgrund i de globala förändringarna i temperatur uppstår frågan: går det även att se en ökande trend i extrem nederbörd i Sverige? Syftet med detta kandidatarbete är att undersöka just detta. För att besvara frågan analyserades dygnsnederbörden från 37 av SMHI:s väderstationer, geografiskt spridda över Sverige, med hjälp av så kallad extremvärdesanalys. Det är en statistisk metod som fokuserar på egenskaper hos extrema observationer i en datamängd. Genom att modellera dessa extremvärden med olika sannolikhetsfördelningar är det möjligt att identifiera eventuella trender i extrema nederbördshändelser. Vidare hjälper detta oss att undersöka hur intensiteten och frekvensen hos extrema nederbördshändelser har förändrats över tid.

För att fånga olika aspekter av förändringar i extrem nederbörd användes tre kompletterande metoder inom extremvärdesanalys: block maxima-metoden, tröskelmetoden och en modell baserad på en inhomogen Poissonprocess. Block maxima-metoden fokuserar på storleken på maximum i olika intervall. I detta arbete innebar det att den maximala dygnsnederbörden för varje år valdes ut, varefter trender i dessa årliga extremvärden analyserades vid varje station. Tröskelmetoden undersöker i stället om intensiteten av extrema nederbördshändelser har förändrats över tid. En tröskelnivå definieras, exempelvis en hög kvantil, och därefter studeras hur mycket extrema värden överskrider denna tröskel. För att analysera frekvenstrender anpassades en inhomogen Poissonprocess till samma data som för tröskelmetoden. Denna metod modellerar förekomsten av extrema nederbördstillfällen som en stokastisk process vars intensitet tillåts variera över tid.

Resultaten från tröskelmetoden visade få stationer med statistiskt signifikanta trender, vilket innebär att denna metod inte gav något tydligt stöd för en förändring i intensiteten hos extrem dygnsnederbörd över tid. Block maxima-metoden gav svaga tecken på trender i årliga maximum, då 16% av stationerna hade statistiskt signifikanta trender. Den inhomogena Poissonprocessen visade däremot signifikanta trender i frekvensen av extrema nederbördstillfällen vid 38% av stationerna, vilket ger tydliga stöd för att det finns trender i extrem dygnsnederbörd. Baserat på medianen av de skattade trenderna från denna modell beräknas att frekvensen av extrema dagliga nederbördshändelser har ökat med cirka 55% under en period på 60 år. Med andra ord har extrema nederbördshändelser blivit avsevärt vanligare med åren. Ökad frekvens borde i sin tur leda till ökande trender i årliga maximum i större utsträckning än vad som upptäckts här. En möjlig förklaring till detta är den begränsade längden hos flera av stationernas mätserier, vilket kan ha minskat möjligheten att upptäcka tydliga trender.

Sammandrag

Denna rapport undersöker trender i extrema dygnsnederbördshändelser vid 37 svenska mätstationer med syfte att avgöra om extrem nederbörd blivit intensivare och frekventare i takt med klimatförändringarna. Datan som användes är över dygnsnederbörd och hämtades från SMHI. Den analyserades med extremvärdesteori i programmeringsspråket R, det tillämpades huvudsakligen två metoder, tröskelmetoden och block maxima-metoden tillsammans med en GEV-fördelning och en GP-fördelning. För tröskelmetoden analyserades överskridandens avstånd till 99.5% kvantilen och för block maxima-metoden analyserades årliga maximum. Som komplettering gjordes även en modellering med en inhomogen Poissonprocess för att undersöka frekvens. För samtliga analyser användes likelihoodkvot-test varefter p-värden beräknats för att bedöma signifikans, med signifikansnivå $\alpha = 0,05$. Resultatet visade att 16% av stationerna hade signifikant trend enligt block maxima-metoden, medan 10% hade signifikans med tröskelmetoden. Den inhomogena Poissonprocessen ger däremot ett tydligare resultat, där uppvisar 38% av stationerna signifikanta trender. Baserat på medianen av trendskattningarna i frekvens har det skett en ökning med 55% de senaste 60 åren. Detta tyder på att extrem nederbörd har blivit mer frekvent, även om de årliga maximumen inte ökat i samma uträckning. Det i sin tur tyder på att den statistiska osäkerheten var för stor för att upptäcka alla trender i årliga maximum. Avslutningsvis tas samhälleliga aspekter och konsekvenser av ökad extrem nederbörd upp, såsom påverkan på infrastruktur och översvämningsrisker. Det ges också förslag på framtida projekt. Bland annat hade det varit intressant att studera hur nederbörden ökar med avseende på temperatur.

Abstract

This report studies trends in extreme daily precipitation events at 37 Swedish weather stations, aiming to determine whether these events have become more intense and frequent in response to climate change. The data is sourced from SMHI and was analyzed using extreme value theory in the R programming lanuguage. Two main methods were used, Peak over Threshold and block maxima utilizing the GEV and GP distributions. For the threshold method, the distance between exceedeances and the 99.5% quantile was analyzed, while block maxima analyzed the annual maxima. Additionally, an inhomogeneous Poisson process was applied to study the change in event frequency. All the trends were evaluated using likelihood ratio tests and p-values to assess statistical significance, with significance level of $\alpha = 0.05$. Results showed that 16% of the stations show statistically significant trends using block maxima, while 10% of the stations presented significance with the threshold method. The inhomogeneous Poisson process provided clearer outcomes, revealing significant trends at 38% of the stations. Based on the median of the trend estimates in frequency, there has been a 55% increase over the past 60 years. This suggests that extreme precipitation has become more frequent, even though the annual maxima have not increased to the same extent. This, in turn, indicates that the statistical uncertainty was too high to detect all trends in the annual maxima. The report also addresses societal consequences of increased extreme precipitation, particularly concerning infrastructure and flood risks. Future research directions are proposed, including an investigation into how precipitation scales with temperature.

Innehåll

1 Inledning

Historiskt har klimatet på jorden varierat vilket tyder på att det är ett känsligt system där atmosfärens sammansättning är en viktig del [1]. Genom utsläpp av bland annat växthusgaser påverkar människan den sammansättningen vilket leder till ett varmare klimat globalt sett. Inom klimatforskningen anses det just nu ske snabba klimatförändringar, till exempel var 2022 ungefär 1,5 grader varmare i genomsnitt jämfört med perioden 1850-1900. Klimatmodeller visar också att sannolikheten för flera extrema vädersituationer ökar i ett varmare klimat. Genom att analysera den globala medeltemperaturen kan det också ses att det skett en kraftig ökning sedan 1970-talet [2]. Det tros att en liknande ökning inte skett de senaste 2000 åren.

Idag kan det i världen konstateras att extremväder, exempelvis orkaner, bränder och översvämningar, blivit kraftigare och generellt även mer frekventa [3]. Många av dessa extremväder är kopplade till vatten, att det antingen finns för mycket eller för lite av det. Vattnet och dess kretslopp, det så kallade hydrologiska kretsloppet, påverkas i ett varmare klimat [4]. Kretsloppet snabbas på då vattnet avdunstar snabbare vilket kan orsaka torka. Detta ökar risken för mer extrem nederbörd som i sin tur kan leda till översvämningar.

Enligt SMHI kan det även i Sverige ses en tydlig uppvärmning de senaste åren [1]. I Sverige tros klimatförändringarna leda till extremväder, och där inräknat en ökad risk för skyfall [5]. Därför är ett ökat antal översvämningar något som MSB menar är bra att förbereda sig på. I ett teoretiskt händelsescenario om extrem nederbörd tog MSB upp möjliga konsekvenser av sådana översvämningar [6]. Dessa var bland annat att de översvämmade vägarna isolerade människor som inte kunde ta sig till arbete och skola. Utöver detta nämner de andra potentiella konsekvenser, såsom skador på ledningsnät, vattenskadade byggnader och förorenat dricksvatten, vilket orsakar stor skada och kostnad för såväl samhället som privatpersoner.

SMHI påpekar dock att genom att känna till riskerna för en viss mängd nederbörd kan konsekvenserna från dem förebyggas med exempelvis effektivare dräneringssystem [7]. Detta gör det intressant att studera trender i extrema nederbördshändelser i Sverige för att undersöka om det existerar ökningar redan idag. Då nederbörd kan vara väldigt lokalt poängterar SMHI att det kan vara intressant att studera ifall vissa platser är mer utsatta än andra [8]. Genom att få svar på dessa frågor kan man se var extra skyddsåtgärder bör sättas in och även vad individen, samt samhället i stort kan göra.

För att statistiskt undersöka extrema nederbördshändelser används extremvärdesteori, som är särskilt användbar för att analysera frekvens och storlek av extrema väderfenomen. Teorin bygger främst på asymptotiska resultat som ger fördelningar som kan anpassas till data. Eftersom de händelser som analyseras är ovanliga krävs ofta extrapolering av tillgänglig data för att göra meningsfulla uppskattningar av risken för framtida extrema händelser [9]. För att uppskattningarna ska bli så tillförlitliga som möjligt behövs långa och homogena mätserier.

I Sverige finns sådana mätserier för nederbörd och flöden i vattendrag. Flödesmätningar är generellt lättare att analysera, då nederbörd varierar mer och ofta har större mätfel [1]. Flödesmätningarna kan därför användas som komplement vid analys av nederbörd då vattendrag ses som stora areor där nederbörden samlas. Förra året genomfördes ett kandidatarbete där trender i dessa flödesmätningar analyserades, men utan att identifiera några tydliga trender [10]. I detta projekt kommer vi istället att fokusera på att analysera nederbördsdata för att undersöka vilket resultat det ger.

1.1 Syfte och frågeställning

Projektets syfte är att undersöka om det förekommer trender i extrema nederbördshändelser i Sverige, både vad gäller frekvens och storlek. Om trender hittas ska det avgöras hur stora de är. Följande fråga ska besvaras.

- Finns det trender i extrem dygnsnederbörd i Sverige?

1.2 Avgränsningar

Arbetet fokuserar på dygnsnederbörd och datan som används hämtas från SMHI, som har mätstationer över hela Sverige. Projektet fokuserar på stationer med data fram till 2024-11-30 och som har minst 60 års dagliga nederbördsmätningar utan några luckor. 60 år valdes som minimum för att uppnå en balans mellan mätseries längd och antalet stationer som uppfyllde kraven. Detta betyder också att längden på de analyserade stationerna kan variera, den längsta perioden som analyseras är 1836-2024 och den kortaste är 1963-2024.

Under perioden har mätmetoder ändrats, det har gått från manuella till automatiserade mätningar. SMHI diskuterar att deras automatiserade stationer mäter cirka 90-95% av de manuella värdena, detta anses ha en begränsad påverkan på resultatet och tas därför inte i beaktning [11].

Vidare studeras endast trender och inte dess orsaker eller konsekvenser då det skulle göra projektet för omfattande. Exempelvis kommer det inte diskuteras om en extrem nederbördshändelse berodde på att det var ett varmare år eller om den ledde till en översvämning.

För dataanalysen används extremvärdesteori, mer specifikt anpassas datan till en generaliserad extremvärdesfördelning, en generaliserad pareto fördelning och en inhomogen Poissonprocess. Dessa metoder möjliggör en analys av förändringar i både frekvens och intensitet.

2 Teori

Kapitlet inleder med att introducera de relevanta fördelningarna tillsammans med respektive metod varefter metoden för att analysera tidsberoende introduceras. Till sist diskuteras diagnostistik och tolkning av resultaten. Teorin är till stor del hämtad från Coles [9], men gällande Poissonprocessen och tolkning även från Ólafsdóttir [12].

2.1 Generaliserade extremvärdesfördelningen

Antag att $X_1, ..., X_n$ är oberoende och identiskt fördelade stokastiska variabler med fördelningsfunktion $F(x)$. Låt $M_n = max\{X_1, ..., X_n\}$ och låt $F_{M_n}(x)$ vara dess fördelningsfunktion. Analytiskt blir den

$$\begin{aligned} F_{M_n}(x) &= P(M_n \leq x) \\ &= P(X_1 \leq x, ..., X_n \leq x) \\ &= P(X_1 \leq x)...P(X_n \leq x) \\ &= (F(x))^n. \end{aligned}$$

Vi vet sällan den exakta fördelningsfunktionen $F(x)$ och kan därför inte beräkna $F_{M_n}(x)$. En lösning är att skatta $F(x)$ med observerade värden, men skattningen av $F_{M_n}(x)$ kan då vara instabil. Detta eftersom små förändringar i datan, och därmed skattningen av $F(x)$, leder till stora förändringar i skattningen av $F_{M_n}(x)$. Vanligare är att istället dra nytta av följande sats.

Sats 1 *Låt G vara en icke-degenererad fördelningsfunktion. Om det existerar en positiv talföljd $\{a_n\}$ och en talföljd $\{b_n\}$ sådana att*

$$P\left(\frac{M_n - b_n}{a_n} \leq z\right) \to G(z) \text{ då } n \to \infty \tag{1}$$

så ingår G i den så kallade generaliserade extremvärdes-familjen av fördelningar, härifrån kallade GEV-fördelningen [9, s. 48]. *Den har fördelningsfunktion:*

$$G(z) = exp\left\{-\left[1 + \xi\left(\frac{z-\mu}{\sigma}\right)\right]^{-1/\xi}\right\}, \tag{2}$$

med stöd i $\{z : 1 + \xi(z-\mu)/\sigma > 0\}$ och parametrar $\mu \in \mathbb{R}$, $\sigma > 0$ och $\xi \in \mathbb{R}$.

En slumpvariabel med en degenererad fördelning har ett specifikt värde med sannolikhet 1. Stöd gäller för en kontinuerlig slumpvariabel och är mängden där täthetsfunktionen är positiv.

I praktiken känner vi inte dessa talföljder, men det visar sig att vi inte behöver det. Givet (1) så är

$$P\left(\frac{M_n - b_n}{a_n} \leq z\right) \approx G(z)$$

för tillräckligt stora n. Det är ekvivalent med att

$$\begin{aligned} P(M_n \leq z) &\approx G\left(\frac{z - b_n}{a_n}\right) \\ &= G^*(z) \end{aligned}$$

där G^* är en annan GEV-fördelning med parametrar $\xi^* = \xi$, $\mu^* = a_n\mu + b_n$ och $\sigma^* = a_n\sigma$. Vi kan alltså direkt approximera fördelningen av maximum med GEV-fördelningen utan att behöva ta hänsyn till den linjära transformationen.

För $X \sim GEV(\mu, \sigma, \xi)$ så gäller att

$$E[X] = \begin{cases} \mu + \frac{\sigma}{\xi}\left(\Gamma(1-\xi) - 1\right), & \text{om } \xi < 1,\ \xi \neq 0 \\ \mu + \sigma\gamma, & \text{om } \xi = 0 \\ \infty & \text{om } \xi \geq 1 \end{cases} \tag{3}$$

där Γ är gammafunktionen och γ är Eulers konstant [13, s. 8]. Större värden på μ och σ ger då större förväntade årliga maximum.

GEV-fördelningen ger en fördelning på det maximala värdet. Den lämpar sig bra när endast de största värdena spelar roll. Exempel på detta kan vara vid konstruktionen av vallar för att skydda mot stora vågor eller vid planering av skyddsåtgärder mot översvämningar. I figur 1 ges ett exempel på hur fördelningen kan se ut.

2.2 Block maxima-metoden

Mer praktiskt kan vi tillämpa GEV-fördelningen med block maxima-metoden [9]. Den går ut på att dela in datan i ett antal block med fixerad längd, vanligtvis pragmatiskt vald som exempelvis år. De maximala värdena $m_{n,1}, m_{n,2}, ..., m_{n,i}$ för blocken betraktar vi som realiseringar av $M_{n,1}, M_{n,2}, ..., M_{n,i}$ där n är blocklängden och i antalet block. Fördelningen av dessa slumpvariablerna kan approximeras med en GEV-fördelning. Dessutom är de oberoende av varandra då $n \to \infty$, vilket gör att vi kan betrakta $m_{n,1}, m_{n,2}, ..., m_{n,i}$ som ett oberoende stickprov från den aktuella GEV-fördelningen och därmed skatta dess parametrar.

Valet av blocklängd är en avvägning mellan systematiskt fel och varians, längre block ger bättre approximation med GEV men färre mätvärden används och därmed ökar osäkerheten i de statistiska skattningarna. Om blocklängden istället är kort får vi fler mätvärden, men approximationen med GEV kanske är mindre god vilket kan leda till systematiska fel [9, s. 54].

En nackdel med block maxima-metoden är att vi bortser från potentiellt användbar data när vi endast använder det maximala värdet för varje block. Därmed skiljer vi inte på block med många stora värden och block med ett stort värde. Om den skillnaden är betydelsefull eller inte, beror på den praktiska tillämpningen.

2.3 Generaliserade Pareto-fördelningen

Det kan även vara relevant att betrakta alla värden som anses extrema, och inte bara det maximala. Vi definierar då ett extremt värde som ett värde som överstiger en bestämd tröskel u. Vad vi söker är då fördelningen för en stokastisk variabel, givet att dess värde överstiger tröskeln. Låt X beteckna en godtycklig term i serien $X_1, ..., X_n$ från början av kapitlet med fördelningsfunktion $F(x)$. Då är

$$P(X > u + y \mid X > u) = \frac{1 - F(u + y)}{1 - F(u)},\ y > 0.$$

Likt tidigare är $F(x)$ okänd, varför vi behöver approximera denna fördelningen.

Sats 2 *Definiera $X_1, ..., X_n$ och M_n som tidigare och antag att (1) är uppfyllt. Då gäller, för tillräckligt stora u, att fördelningen för $X - u \mid X > u$ approximativt har fördelningsfunktion:*

$$H(y) = 1 - \left(1 + \frac{\xi y}{\tilde{\sigma}}\right)^{-1/\xi} \tag{4}$$

med stöd i $\{y : y > 0$ och $(1 + \xi y/\tilde{\sigma}) > 0\}$, där $\tilde{\sigma} = \sigma + \xi(u - \mu)$ med parametrar som i (2) [9, s. 75].

Fördelningsfamiljen som beskrivs i (4) kallas för den generaliserade Pareto-fördelningen, härifrån kallad GP-fördelningen. När GP-fördelningen behandlas separat från GEV-fördelningen kommer parametern $\tilde{\sigma}$ kallas för σ.

Värt att notera är att fallet då $\xi = 0$, vilket i (4) motsvaras av $\xi \to 0$, som ger

$$H(y) = 1 - \exp\left(-\frac{y}{\sigma}\right),$$

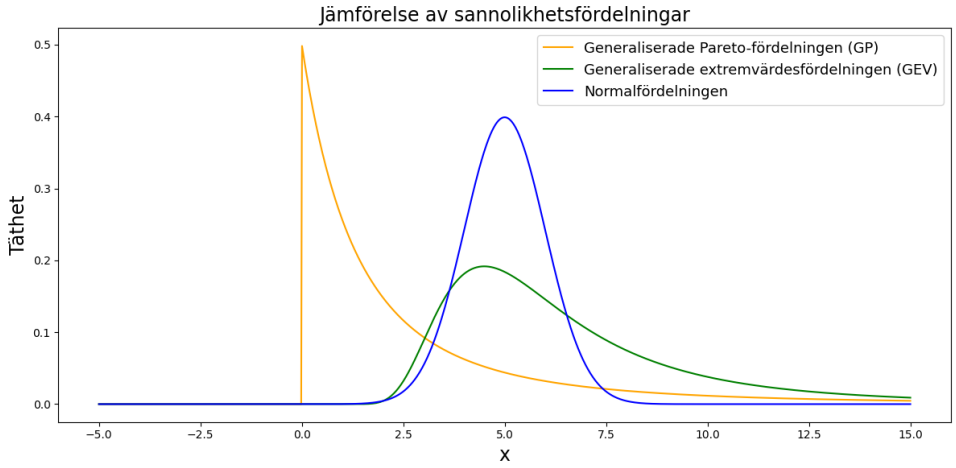
det vill säga en exponentialfördelning med parameter $1/\sigma$.

För $Y \sim GP(\sigma, \xi)$ gäller att

$$E[Y] = \frac{\sigma}{1 - \xi} \tag{5}$$

om $\xi < 1$, annars är $E[Y] = \infty$ [9, s. 78]. Större värde på ξ och σ motsvarar alltså att det är vanligare med större extremer.

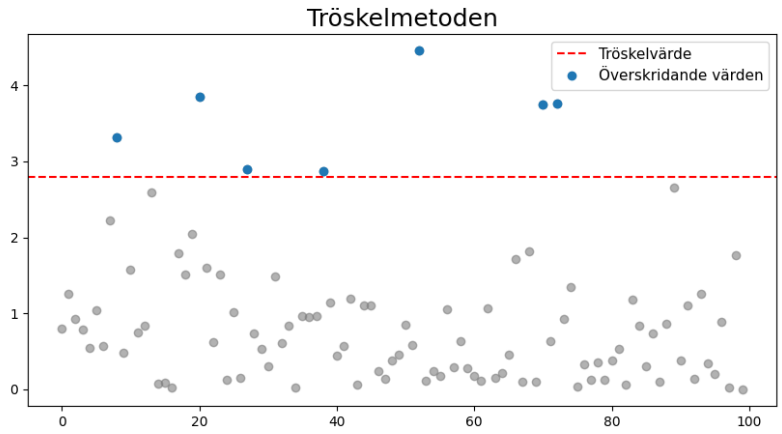
GP-fördelningen ger fördelningen på svansen för en fördelningen, det vill säga fördelningen av de största värdena. I praktiska sammanhang är detta användbart om vi bryr oss om alla förekomster av extrema händelser, till exempel alla extrema stormar och inte bara de största stormarna varje år. Figur 1 ger ett exempel på hur GP-fördelningen kan se ut.



Figur 1: Täthetsfunktionen för GEV($\mu = 5$, $\sigma = 2$, $\xi = 0,3$), GP($\sigma = 2$, $\xi = 0,5$) samt N(5,1).

2.4 Tröskelmetoden

Tröskelmetoden, på engelska kallad Peaks over Threshold, bygger på sats 2 och innebär att vi väljer en tröskel, u, som är tillräckligt hög för att sats 2 ska vara giltig. Därefter analyseras avstånden mellan tröskelvärdet och alla överskridanden. Dessa ser vi som ett oberoende stickprov från en GP-fördelning, (4), vilket låter oss skatta dess parametrar. En visualisering av tröskelmetoden visas i figur 2.



Figur 2: I tröskelmetoden analyseras enbart de blå överskridande punkterna och deras avstånd till tröskeln.

En viktig aspekt i tröskelmetoden är valet av tröskel u, vilket inte kan göras på ett objektivt sätt. Valet kan göras grafiskt, då parameterskattningarna ska bete sig på ett känt sätt om approximationen är giltig. Ett annat alternativ är att välja en lämplig övre kvantil, såsom de 1% största värdena, och använda detta som tröskelvärde.

En stor fördel med tröskelmetoden gentemot block maxima är att vi kan betrakta alla extrema värden, istället för att bara betrakta ett värde per block.

2.5 Deklustring

Ett grundläggande antagande för tröskelmetoden är att alla överskridanden är oberoende och inte kommer i kluster. I verkligheten kan detta vara ett orealistiskt antagande. Exempelvis tenderar det att vara mer nederbörd dagarna runt en extrem nederbördshändelse än i genomsnitt, vilket kan leda till kluster av överskridanden med inbördes beroende observationer.

En metod för att lösa detta kallas deklustring, vilken går ut på att filtrera bort överskridanden som uppkommer tätt intill varandra [9, s. 99]. Detta utförs genom att först definiera vad som klassas som ett kluster: hur många dagar behöver gå innan ett överskridande klassas som ett nytt kluster? Inom varje kluster identifieras det maximala värdet, vilket antas vara oberoende av andra klustermaximum. Därefter kan en GP-fördelning anpassas enligt ovan.

2.6 Poissonprocesser

En Poissonprocess är en stokastisk process, $\{N(t), t \geq 0\}$, med tidsparameter t som räknar antalet händelser i ett tidsintervall. Poissonprocesser är antingen homogena eller inhomogena.

En tidshomogen Poissonprocess med frekvens λ har följande egenskaper:

1. $N(0) = 0$
2. ökningarna är oberoende
3. antalet händelser i ett tidsintervall t är $\sim Poi(\lambda t)$.

För en tidshomogen Poissonprocess beror alltså fördelningen enbart på längden på tidsintervallet och inte var intervallet är.

En inhomogen Poissonprocess är en Poissonprocess med varierande frekvens. Det innebär att punkt 3 ovan byts ut mot att antalet händelser i delmängden A av den totala tiden är $\sim Poi(\Lambda(A))$ där

$$\Lambda(A) = \int_A \lambda(t)dt$$

är intensitetsmåttet och $\lambda(t)$ är en icke-negativ funktion, kallad frekvensfunktion [9, s. 125-126]. En konstant frekvensfunktion $\lambda(t) = \lambda$ motsvarar en homogen Poissonprocess med frekvens λ.

2.7 Trender

För att skatta trender kan parametrar göras beroende av exempelvis tid eller temperatur. För GEV-fördelningen i (2) kan det se ut på följande sätt:

$$\mu(t) = \mu_0 + \ \mu_1 t, \quad ln\, \sigma(t) = \sigma_0 + \sigma_1 t.$$

Observera den logaritmiska länkfunktionen ln för $\sigma(t)$. Den gör att $\sigma(t)$ är positiv, som den ska vara. För GP-fördelningen, (4), har vi analogt

$$ln\, \tilde{\sigma}(t) = \tilde{\sigma}_0 + \tilde{\sigma}_1 t.$$

Vi kan introducera en trend även i den gemensamma parametern ξ men den brukar antas vara konstant, och så även i denna analys.

Parametrarna μ_0, σ_0 respektive $\tilde{\sigma}_0$ motsvarar parametervärdena då $t = 0$ och μ_1, σ_1 respektive $\tilde{\sigma}_1$ är förändringen i parametervärde för varje enhetsökning av t.

För en inhomogen Poissonprocess är

$$\ln \lambda(t) = \lambda_0 + \lambda_1 t$$

ett typiskt val av frekvensfunktion för att kunna skatta trender.

2.8 Maximum likelihood-skattning

Givet en fördelning $\mathcal{F}$ med parameter θ och känd täthetsfunktion är den vanligaste formen av parameterskattning maximum likelihood metoden. Den går ut på att konstruera en så kallad likelihoodfunktion som, för ett givet ett parametervärde, ger en täthet till den observerade datan, och sedan maximera funktionen med avseende på parametervärdet.

Mer exakt så givet en realisering $\{x_1, ..., x_n\}$ av identiska oberoende slumpvariabler med täthetsfunktion $f(x;\theta)$, där θ är en parameter med okänt sant värde θ_0, så är likelihoodfunktionen definierad som

$$L(\theta) = \prod_{i=1}^{n} f(x_i;\theta).$$

Av beräkningsmässiga skäl används ofta istället log-likelihoodfunktionen

$$\ell(\theta) = ln\, L(\theta) = \sum_{i=1}^{n} ln\, f(x_i;\theta).$$

Maximum likelihood-skattningen $\hat{\theta}$ av θ_0 är det värde på θ som maximerar $L(\theta)$. Eftersom logaritmfunktionen är strängt växande, är det ekvivalent med att $\hat{\theta}$ maximerar $\ell(\theta)$.

En viktig egenskap hos maximum likelihood-skattningar är att estimatorerna är asymptotiskt normalfördelade givet vissa antaganden. Vi approximerar därför skattningarnas fördelningar med normalfördelningar för att konstruera konfidensintervall. Maximum likelihood-skattningen existerar för GEV-fördelningen och GP-fördelningen om $\xi > -1$, och om $\xi > -0,5$ är skattningen asymptotiskt normalfördelad [9, s. 55].

2.9 Likelihoodkvot-test

Ett flexibelt test för att testa skillnaden mellan olika modeller är likelihoodkvot-testet. Testet drar nytta av följande sats [9, s. 35-36].

Sats 3 *Låt M_0 och M_1 vara två modeller där M_0 är en delmodell av M_1, det vill säga om M_1 har parameter $\theta = (\theta^{(0)}, \theta^{(1)})$ så är M_0 samma som M_1 med kravet att k-dimensionella vektorn $\theta^{(1)} = \mathbf{0}$. Låt $\ell_{max}(M_0)$ och $\ell_{max}(M_1)$ vara respektive maximala värde på log-likelihoodfunktionen. För tillräckligt stora stickprov gäller att*

$$D = 2(\ell_{max}(M_1) - \ell_{max}(M_0)) \sim \chi^2_k,$$

där χ^2_k är chitvåfördelningen med k frihetsgrader, givet att stickprovet har genererats av M_0.

För att utföra likelihoodkvot-testet med signifikansnivå α beräknar vi teststatistikan D och förkastar M_0 till förmån för M_1 på signifikansnivå α, om $D > c_\alpha$, där c_α är $(1-\alpha)$-kvantilen för χ^2_k-fördelningen.

2.10 Diagnostistik

För att kunna lita på resultat från en statistisk undersökning behöver vi kontrollera att den anpassade modellen är en god representation av datan. En av de vanligaste metoderna är en så kallad Q-Q-plott.

Definition 1 *Givet ett ordnat stickprov med oberoende observationer*

$$x_1 \leq x_2 \leq ... \leq x_n$$

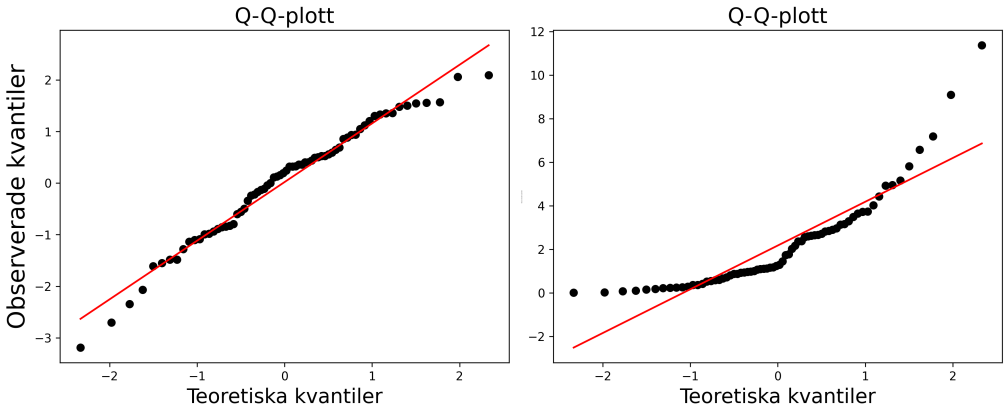
från en population med skattad fördelningsfunktion $\hat{F}$, definieras en Q-Q-plott som punkterna

$$\left\{ \left(\hat{F}^{-1}\left(\frac{i}{n+1} \right) , x_i \right) \,:\, i = 1, ..., n \right\}$$

i planet.

Namnet kommer från engelskans "quantile-quantile plot", då de två axlarna visar de teoretiska kvantilerna gentemot de observerade kvantilerna. I figur 3 visas exempel på två Q-Q-plottar.

Om punkterna ligger nära linjen $x = y$ tyder det på att den valda fördelningen passar datan, och tvärtom, om de inte gör det tyder det på att datan kommer från en annan fördelningsfamilj än den antagna.



Figur 3: Till vänster visas en Q-Q-plott där simulerade N(0,1)-värden jämförs med teoretiska och till höger jämförs Exp(2)-värden med teoretiska N(0,1).

Ett problem är att vi vill introducera trender i fördelningen, vilket innebär att observationerna antas komma från samma fördelning men med olika parametervärden. Därmed förändras de teoretiska kvantilerna och förhindrar direkt jämförelse mellan mätvärdena. En lösning är att standardisera de bakomliggande slumpvariablerna så att de får en gemensam fördelning som är oberoende av tiden.

2.10.1 GEV-fördelningen

Antag att

$$X_t \sim GEV(\hat{\mu}(t), \hat{\sigma}(t), \hat{\xi})$$

för skattade parametervärden $\hat{\mu}(t), \hat{\sigma}(t)$ och $\hat{\xi}$ [9, s. 110]. Då definieras den standardiserade slumpvariabeln $\tilde{X}_t$ som

$$\tilde{X}_t = \frac{1}{\hat{\xi}} \ln \left\{ 1 + \hat{\xi} \left(\frac{X_t - \hat{\mu}(t)}{\hat{\sigma}(t)} \right) \right\}. \tag{6}$$

Fördelningen för $\tilde{X}_t$ är känd och är $\sim GEV(\mu = 0, \sigma = 1, \xi = 0)$ med fördelningsfunktion

$$P(\tilde{X}_t \leq x) = \exp(-e^{-x}).$$

Det innebär att vi kan transformera våra mätvärden enligt (6), ordna dem efter storlek och bilda en Q-Q-plott med punkterna

$$\{(-\ln(-\ln(i/(n+1))), \tilde{x}_i), i = 1, ..., n\}.$$

2.10.2 GP-fördelningen

Likt för GEV-fördelningen vill vi, givet

$$Y_t \sim GP(\hat{\sigma}(t), \xi)$$

för skattade värden $\hat{\sigma}(t)$ och $\hat{\xi}$, standardisera till en känd fördelning [9, s. 111]. I detta fallet blir det

$$\tilde{Y}_t = \frac{1}{\hat{\xi}} \ln \left\{ 1 + \hat{\xi} \left(\frac{Y_t - u}{\hat{\sigma}(t)} \right) \right\}, \tag{7}$$

där u är tröskelvärdet som valdes. Det resulterar i

$$\tilde{Y}_t \sim GP(\sigma = 1, \xi = 0) \iff \tilde{Y}_t \sim Exp(\beta = 1)$$

med fördelningsfunktion

$$P(\tilde{Y}_t \leq y) = 1 - e^{-y}.$$

Likt för GEV-fördelningen kan vi bilda en Q-Q-plott genom att transformera alla överskridanden enligt (7), ordna dem, och visualisera med punkterna

$$\{(-\ln(1 - i/(m+1)), \tilde{y}_i) \; ; \; i = 1, ..., m\}.$$

2.10.3 Q-Q-plott med p-värden

Risken för felaktiga förkastningar av nollhypotesen vid hypotestest ökar drastiskt då mängden test ökar. Detta enligt konstruktion, då risken för varje enskilt test är bestämd med signifikansnivån α. Om vi då utför n olika hypotestest förväntar vi oss $n \cdot \alpha$ felaktiga förkastningar.

Det finns flera sätt att ta hänsyn till detta, ett är utnyttja p-värden. En viktig egenskap hos p-värden är att de är $\sim U(0, 1)$ givet nollhypotesen. Om vi därmed ordnar p-värdena efter storlek, $p_1, p_2, ..., p_n$, och bildar en Q-Q-plott med punkterna

$$\{(i/n, p_i), i = 1, ..., n\}$$

kan vi jämföra p-värdenas fördelning med den teoretiska fördelningen, givet nollhypotesen. Om punkterna ligger längs linjen $y = x$ tyder det på att nollhypotesen är sann och vi kan bortse från de eventuellt signifikanta resultaten.

2.11 Tolkning

En viktig aspekt av en statistisk undersökning är tolkning av resultat. Detta är inte alltid trivialt och kräver matematiska resonemang.

Vi är ute efter att skatta trender i parametrar, men rent konkret kan det uttrycka sig på olika sätt. Poissonprocessen i kombination med GP-fördelningen kan ge tre typer av trender i extrem nederbörd. Det kan (i) finnas en positiv trend i λ vilket ger ökad frekvens av överskridanden [12]. Det kan (ii) finnas en trend i σ för GP-fördelningen som medför större överskridanden givet att tröskelvärdet har passerats. Slutligen (iii) så kan det finnas trender i både λ och σ. Alla tre fallen ger större årliga maximum vilket motsvarar en trend i GEV-fördelningen.

Utifrån (3) kan vi göra tolkningen att positiva värden på μ_1 och σ_1 ger en positiv trend i den förväntade storleken på årliga maximum, med skillnaden att μ_1 ger en linjär trend och σ_1 en exponentiell. Värt att nämna är även att ett större värde på σ ger större varians och därmed större spridning på årliga maximum [13].

Utifrån (5) kan vi göra tolkningen att givet ett överskridande och ett värde på $\tilde{\sigma}_1$, så förväntas mängden nederbörd vara $100 \cdot (e^{\tilde{\sigma}_1} - 1)$ % större än föregående år.

På liknande sätt, givet λ_1, så ökar det förväntade antalet överskridanden med $100 \cdot (e^{\lambda_1} - 1)$ % varje år. Det förväntade antalet överskridanden under de x första åren sen mätstart blir:

$$\int_0^x e^{\lambda_0 + \lambda_1 t} dt = \frac{e^{\lambda_0}}{\lambda_1}(e^{\lambda_1 x} - 1).$$

Detta kan sedan enkelt jämföras med det uppmätta antalet för att undersöka eventuella räknefel eller diagnostistik för modellen.

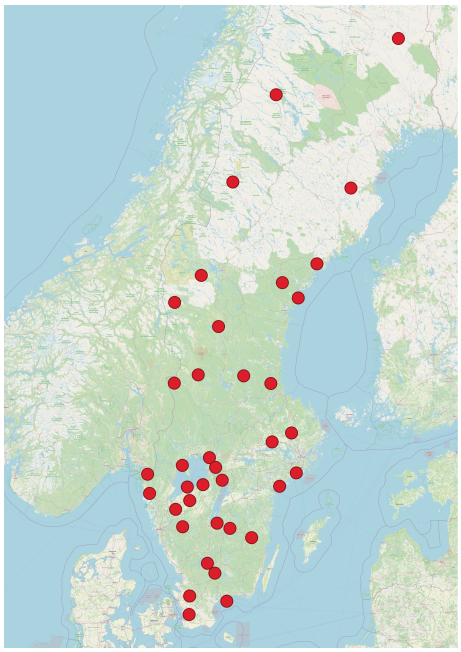
3 Metod

Projektet analyserade 37 svenska väderstationer för att undersöka om det finns trender i extrema nederbördshändelser. Till samtliga stationer anpassades en GEV-fördelning, GP-fördelning och en Possionprocess. I detta avsnitt presenteras hur dessa anpassningar och analyser genomfördes, vilket inkluderar programmering i R samt Python, datainsamling, databehandling och tester för att se vilka resultat som var signifikanta. Samtlig kod kan hittas i appendix C.

3.1 Datainsamling

Datan som användes i projektet hämtades från SMHI:s nederbördsdata och visar dygnsnederbörd, uppmätt vid SMHI:s väderstationer ([14] [11]). Nederbörd definierar SMHI som "ett meteorologiskt samlingsnamn för flytande eller fasta vattenpartiklar som faller genom atmosfären."[15]. Datan är enligt SMHI:s datapolicy fritt tillgänglig för forskning och utbildning och hämtades hem via SMHI:s API, Application Programming Interface, med hjälp av Python ([16] [17]).

Bland SMHI:s omkring 600 mätstationer för nederbörd i Sverige gjordes ett urval på 37 stationer, se figur 4. Dessa stationer var de som uppfyllde projektets avgränsningar, och hade daglig mätdata utan avbrott i minst 60 år fram till 2024-11-30.



Figur 4: Karta över de 37 utvalda stationerna med sammanhängande data minst 60 år tillbaka.

Efter att samtliga stationer hade kontrollerats, hämtades nederbördsdatan för de utvalda stationerna. Till varje mätvärde finns vilken station det tillhör och datumet när nederbörden uppmättes. Mätningarna har enheten millimeter [11].

Två datamängder skapades: en där all tillgänglig data som mätts dagligen utan avbrott fram till 2024-11-30 betraktas och en där enbart data från 1963-10-01 och framåt betraktas. Se startdatum för stationerna i appendix A. De olika datamängderna benämns fortsättningsvis datamängd 1 respektive datamängd 2.

3.2 Dataanalys

När insamling av datan var klar användes programmeringsspråket R för att genomföra extremvärdesanalysen. Följande analys genomfördes för alla utvalda stationer och signifikans testades med

signifikansnivå $\alpha = 0,05$ för samtliga parameterskattningar. Samtliga analyser gjordes på båda datamängderna.

3.2.1 Block maxima-metoden

Nederbördsdatan delades in i årliga block enligt USGS:s(United States Geological Survey) definition av vattenår, vilket startar den 1 oktober och avslutas 30 september följande år [18]. Endast fullständiga vattenår användes i analysen. Det maximala värdet i varje block identifierades och sparades. Dessa maximum är vad som därefter användes.

Utifrån maximumen skattades parametrarna för en stationär och en icke-stationär GEV-fördelning, beskrivna i avsnitt 2.1 och 2.7, med hjälp av funktionen *fevd* från R-paketet *extRemes*. Tre icke-stationära modeller med trender undersöktes: en med trend i enbart μ, en med trend i enbart σ och en med trend i båda. I samtliga modeller användes förflutna år sedan mätstart som en förklarande variabel.

Steget ovan upprepades med antagandet att $\xi = 0$ vilket ger en så kallad Gumbel-fördelning. För varje mätstation testades $H_0 : \xi = 0$ mot $H_1 : \xi \neq 0$ med ett likelihoodkvot-test varefter p-värdet beräknades med hjälp av *extRemes* funktionen *lr.test*.

En möjlighet som undersöktes var att anta att $\xi = 0$ för alla mätstationer. Detta gjordes genom att konstruera en plott med förväntade p-värden gentemot observerade, enligt avsnitt 2.10.3, vilka erhölls från det tidigare nämnda likelihoodkvot-testet. Baserat på plotten togs beslutet att antingen anta att $\xi = 0$ för samtliga stationer, eller avgöra det individuellt för stationerna. Det vill säga, det skulle antas att $\xi = 0$ för stationer där H_0 inte förkastades, och det skattade värdet på ξ skulle användas för stationer där H_0 förkastades. Beslutet gällde även för anpassning av GP-fördelning.

För alla stationer utfördes diagnostik enligt avsnitt 2.10.1.

För att testa signifikansen hos trenderna testades $H_0 : \mu_1 = 0,\ \sigma_1 = 0$ mot $H_1 : \mu_1 \neq 0,\ \sigma_1 \neq 0$, $H_0 : \mu_1 = 0$ mot $H_1 : \mu_1 \neq 0$ respektive $H_0 : \sigma_1 = 0$ mot $H_1 : \sigma_1 \neq 0$ för de tre modellerna med likelihoodkvot-test varefter p-värdena beräknades, återigen med funktionen *lr.test*. Observera att för modellen med trend i både μ och σ testades dessa gemensamt mot den stationära modellen.

3.2.2 Tröskelmetoden

För tröskelmetoden, beskriven i avsnitt 2.4, valdes 99.5% kvantilen som tröskelvärde och avstånden mellan överskridanden och tröskelvärdet betraktades. Datan deklustrades, utifrån avsnitt 2.5, med hjälp av *extRemes* deklusterfunktion *decluster*.

Funktionen *fevd* från *extRemes* användes för att skatta GP-fördelningens parametrar, beskrivna i avsnitt 2.3 och 2.7. Parametrarna skattades både för en stationär och en icke-stationär GP-fördelning med trend i σ. Även här användes förflutna år sedan första mätningen som förklarande variabel.

Steget ovan upprepades därefter med antagandet att $\xi = 0$, vilket ger en exponentialfördelning, enligt avsnitt 2.3. För varje mätstation testades $H_0 : \xi = 0$ mot $H_1 : \xi \neq 0$ med ett likelihood-kvot-test och p-värdet beräknades med hjälp av funktionen *lr.test* från *extRemes*.

Trenden i varje mätstationen testades med $H_0 : \sigma_1 = 0$ mot $H_1 : \sigma_1 \neq 0$ i den valda modellen. Detta gjordes med ett likelihood-kvot-test mellan den stationära och icke-stationära fördelningen med hjälp av *lr.test*, varefter p-värdet för testet beräknades med samma funktion. Observera att $\sigma_1 = 0$ motsvarar en stationär fördelning.

Alla stationer genomgick därefter diagnostik enligt avsnitt 2.10.2. Genom granskning av Q-Q-plotten drogs slutsatser kring hur väl lämpad den icke-stationära GP-modellen var för datan och därmed om tröskeln var lämpligt vald för att sats 2 ska gälla.

3.2.3 Anpassning av inhomogen Poissonprocess

Analogt med tröskelmetoden så initierades anpassningen av en inhomogen Poissonprocess, beskriven i avsnitt 2.6, med 99.5% kvantilen som tröskelvärde och med deklustering av värdena över tröskeln. Därefter betraktades enbart överskridningarna.

Funktionen *fitPP.fun* från R-paketet *NHPoisson* användes för att anpassa en inhomogen Poissonprocess. Som förklarande variabel i frekvensfunktionen användes antalet förflutna dagar sedan första mätningen. För jämförelse omvandlades parameterskattningarna för dagar, λ_0^* och λ_1^*, till motsvarande för år med följande transformation:

$$\lambda_0 = \lambda_0^* + \ln(365,25)$$
$$\lambda_1 = \lambda_1^* \cdot 365,25.$$

Bevis för detta kan hittas i [10].

För att testa signifikansen hos trenden användes likelihoodkvot-testet mellan den homogena och inhomogena processen och p-värdet beräknades. Detta gjordes med hjälp av funktionen *LRTpv.fun* från *NHPoisson*.

I enlighet med avsnitt 2.11 jämfördes det förväntade antalet överskridanden med det uppmätta för ett urval av stationerna. Detta för att kontrollera att modellen passade bra till datan.

3.3 Mätfel

Analysen bygger på mätdata hämtad från SMHI, som har uppmätt nederbörden vid ett antal väderstationer. När värdena uppmätts finns risk för mätfel av olika slag. Detta tas ej i beaktning av rapporten men bör ändå uppmärksammas.

Från datan har extremvärden identifierats och små ändringar i datapunkterna kan orsaka stora skillnader i uppskattningarna. Dessa små ändringar skulle kunna vara mätfel. SMHI skriver på sin hemsida om hur de utför sina mätningar av nederbörd [11]. Detta görs genom uppsamling av nederbörden i ett vattentätt kärl. Vind är den största felkällan och kan exempelvis orsaka att det samlas in mindre vatten i kärlet än omkring det. Detaljer om vindens påverkan finns att läsa på SMHI [11].

Vind kan alltså orsaka ett underskott i mätningarna. Detta påverkar framför allt årliga maximum men kan även sänka tröskeln i tröskelmetoden. Frekvensen borde dock förbli densamma.

Vidare diskuterar SMHI andra orsaker till mätfel, oftast förlust av nederbörd. Orsaker som togs upp var avdunstning och vätningsförluster. Avdunstningen har SMHI uppskattat till att vara 10-15 mm/år i södra Sverige. Vätningsförluster innebär att lite vatten är kvar i kärlet när det ska hällas upp i ett mätglas. Detta uppskattas ge ett mätfel på upp till 0,1 mm [11]. Dessa mätfel bedöms för projektets syfte vara systematiska och påverkar därmed inte resultatet nämnvärt.

SMHI tar även upp att det kan mätas för mycket nederbörd. Orsaker kan vara snö som blåst i kärlet eller dagg som bildats. SMHI skriver dock att detta oftast är ett litet fel.

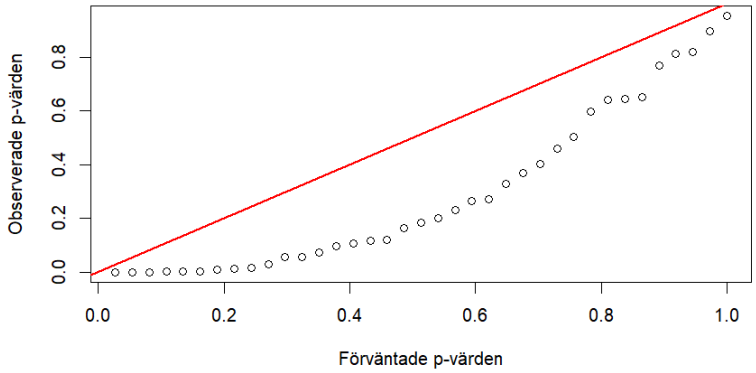
4 Resultat

Nedan presenteras och sammanfattas extremvärdesanalysens resultat med mål att besvara projektets syfte och frågeställning. Stationsspecifika resultat presenteras på projektets GitHub-sida [19]. Där återfinns bland annat de använda datamängderna samt parameterskattningarna hos de olika fördelningarna för varje station.

Den utförda diagnostistiken bedömdes vara tillfredsställande för fortsatt analys. Utvalda Q-Q-plottar kan hittas i appendix.

4.1 Värde på ξ

I figur 5 visas p-värden från likelihoodkvot-test av $H_0 : \xi = 0$ mot $H_1 : \xi \neq 0$. Utifrån figuren bedömdes att det inte kunde antas att $\xi = 0$ för samtliga stationer, varför detta avgjordes individuellt för stationerna i enlighet med avsnitt 3.2.1. Detta gäller för samtliga fortsatta resultat.



Figur 5: Q-Q-plott av observerade mot förväntade p-värden från likelihood-kvottest mellan Gumbel och GEV fördelning med trend i μ, anpassade till årliga maximum från varje station. Den röda linjen är $y = x$.

4.2 Modeller med signifikanta trender

Tabell 1 och 2 visar antalet stationer med signifikant trend i de olika modellerna för datamängd 1 respektive 2, beskrivna i avsnitt 3.1.

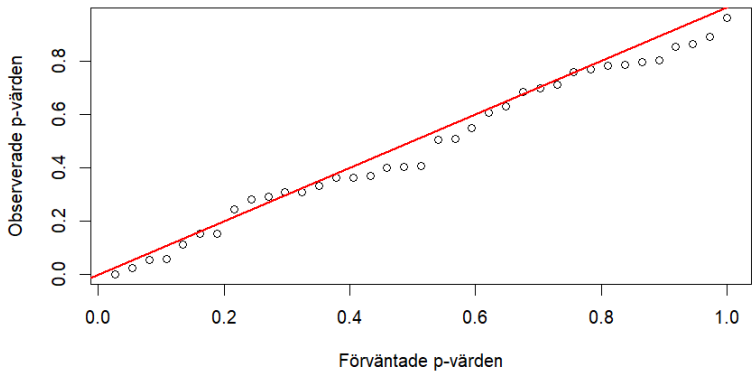
Tabell 1: Antalet stationer med signifikanta trender för datamängd 1.

Modell	Signifikant trend	Insignifikant trend
Block maxima-metoden med trend i μ	6	31
Block maxima-metoden med trend i σ	5	32
Block maxima-metoden med trend i μ och σ	5	32
Tröskelmetoden med trend i σ	4	33
Inhomogen Poissonprocess	14	23

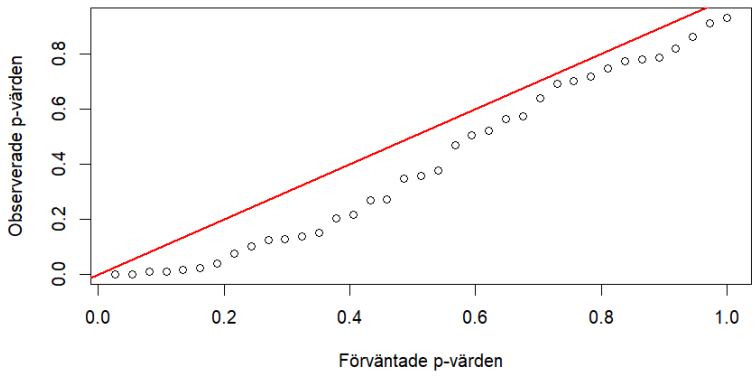
Tabell 2: Antalet stationer med signifikanta trender för datamängd 2.

Modell	Signifikant trend	Insignifikant trend
Block maxima-metoden med trend i μ	3	34
Block maxima-metoden med trend i σ	3	34
Block maxima-metoden med trend i μ och σ	5	32
Tröskelmetoden med trend i σ	4	33
Inhomogen Poissonprocess	8	29

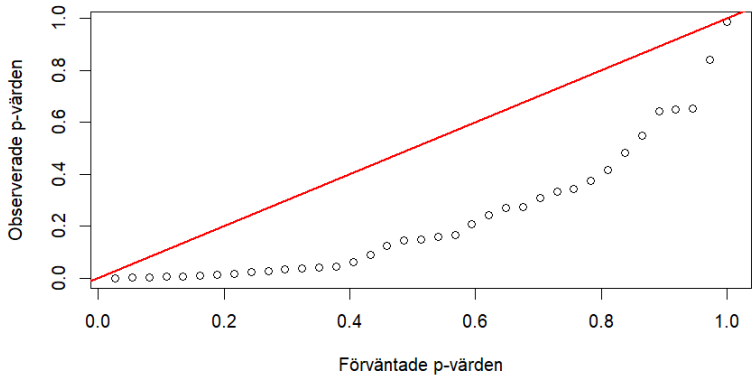
Nedan visas 3 plottar av observerade p-värden från likelihoodkvot-test, mot förväntade p-värden givet respektive nollhypotes. För samtliga har datamängd 1 använts. Figur 6 visar en Q-Q-plott med p-värden för GP-fördelningen. Figur 7 visar motsvarande plott för GEV-fördelningen med trend i μ, och figur 8 visar Q-Q-plotten för en inhomogen Poissonprocess.



Figur 6: Q-Q-plott av observerade mot förväntade p-värden från likelihood-kvottest mellan stationära och icke-stationära GP-fördelningar, anpassade till storleken på tröskelöverskridande nederbörd från varje station. Den röda linjen är $y = x$.

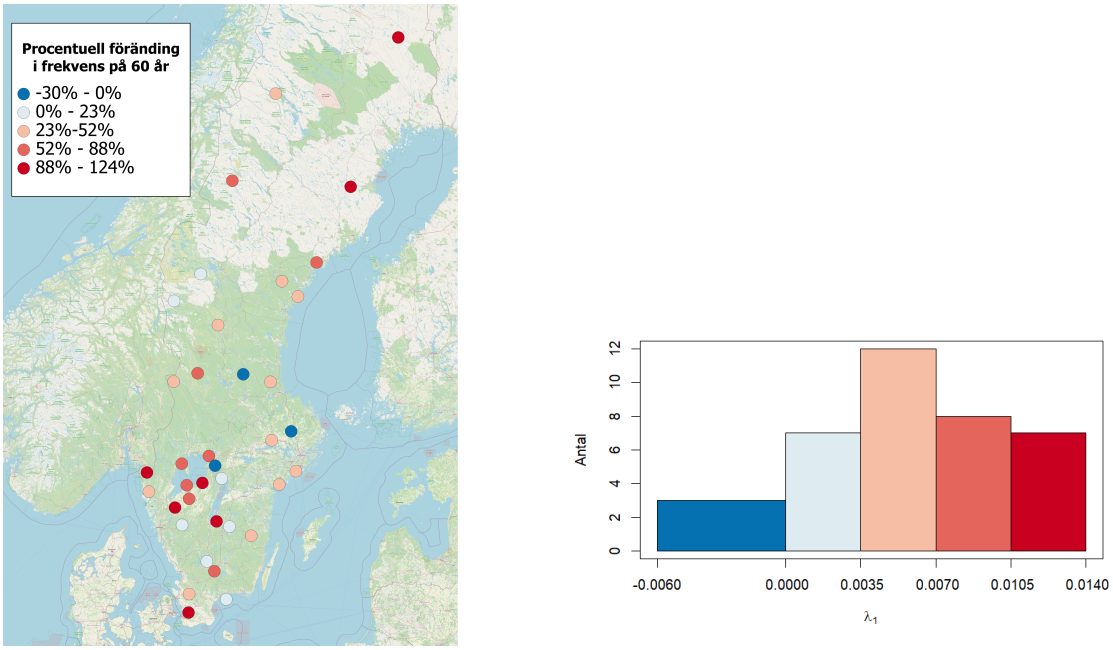


Figur 7: Q-Q-plott av observerade mot förväntade p-värden från likelihood-kvottest mellan stationära och icke-stationära GEV-fördelningar med trend i μ, anpassade till årliga maximum från varje station. Den röda linjen är $y = x$.



Figur 8: Q-Q-plott av observerade mot förväntade p-värden från likelihood-kvottest mellan en homogen och en inhomogen Poissonprocess, anpassade till frekvensen av tröskelöverskridanden från varje station. Den röda linjen är linjen $y = x$.

I figur 9b) visas fördelningen av parameterskattningarna av λ_1 från den inhomogena Poissonprocessen för samtliga stationer i datamängd 1. I figur 9a) visas den geografiska fördelningen av skattningarna av λ_1, där varje färg motsvarar en stapel i diagrammet. Observera att det är $100(e^{60\lambda_1} - 1)\%$ som visas, för att underlätta tolkning och jämförbarhet. Samtliga negativa trender var statistiskt insignifikanta. Alla skattningar för Poissonprocessen finns i appendix A.



a) b)

Figur 9: I a) visas den procentuella förändringen i årlig frekvens av extrema nederbördshändelser under en 60-årsperiod baserat på skattningar från en inhomogen Poissonprocess. I b) visas ett histogram över skattningarna av λ_1, där antalet förflutna år använts som förklarande variabel. Varje skattning motsvarar en station i datamängd 1.

5 Diskussion

Inledningsvis bedöms det, som nämnts i avsnitt 4.1, att antagandet $\xi = 0$ för GEV- och GP-fördelningen inte kan göras. Det betyder att Gumbelfördelningen och exponentialfördelningen inte bedömdes vara tillräckliga för att beskriva datan. Hela GEV- och GP-fördelningarna krävdes för detta och dessa bedömdes vara tillfredsställande utifrån diagnostistiken.

Vidare är antalet signifikanta stationer i tabell 2 färre för majoriteten av metoderna jämfört med tabell 1. Detta antyder att det var svårare att upptäcka trender hos de kortare mätserierna i datamängd 2. Anledningen till uppdelningen var att osäkerheten i de statistiska skattningarna blir mindre för datamängd 1. Samtidigt kan det försvåra upptäckandet av trender om fördelningen, från vilken datan har genererats, ändrats över tid. Exempelvis har jordens medeltemperatur ökat kraftigare sedan ca 1970, vilket kan påverka nederbörden [12]. Det kan också innebära att olika modeller passar för extrem nederbörd före och efter den tidpunkten. I datamängd 2 tas detta hänsyn till och underlättar direkt jämförelse av parameterskattningar, men på bekostnad av högre osäkerhet i skattningarna och därmed lägre statistisk styrka i likelihoodkvot-testen.

Resultaten i tabell 1 och 2 är alltså helt i linje med den lägre styrkan hos testen på datamängd 2, utan att väga upp det med stor skillnad i bakomliggande fördelning. Det färre antalet signifikanta stationer i datamängd 2 tycks därmed bero på osäkerhet i skattningarna snarare än en nödvändig avsaknad av trender, varför resultatet och vidare diskussion kommer fokusera på datamängd 1.

Som kan ses i tabell 1 fås 5-6 stationer med signifikanta trender i årliga maximum. Andelen av samtliga 37 kan tyckas liten, men sannolikheten för minst 5 respektive 6 signifikanta resultat är ungefär $0,036$ respektive $0,01$ under nollhypotesen med $\alpha = 0,05$. Detta kan jämföras med figur 7, där en Q-Q-plott över p-värden från GEV-fördelningen med trend i μ visas. P-värdena tycks följa linjen $x = y$, men ligger konsekvent under den, vilket tyder på att det observerats fler låga p-värden än förväntat under nollhypotesen. Trots detta har inga tydliga genomgående trender kunnat hittas.

För block maxima delades datan in enligt definitionen av vattenår som används i USA, vilken utgår från klimatet där. I Sverige kan det finnas en annan, bättre, indelning utefter landets klimat vilket projektet inte använt. Till exempel kan en indelning som utgår från våren vara mer lämpad då nederbörden är som minst under den perioden i Sverige. Hur detta har påverkat resultatet är svårt att säga, men kan i något fall ha delat upp ett regnoväder på två år när det mer naturligt borde ha ingått i ett år.

Figur 6 visar de observerade p-värdena från ett likelihoodkvot-test av en stationär och en icke-stationär GP-fördelning för varje station, mot de förväntade p-värdena givet nollhypotesen. Eftersom punkterna i stort sett följer linjen $y = x$, tyder detta på att tröskelmetoden resulterar i få signifikanta trender. Jämför detta med att 4 signifikanta trender har hittats, där sannolikheten för minst 4 signifikanta trender är $0,11$ under nollhypotesen. Fördelningen av p-värdena överensstämmer alltså väl med vad som förväntas under $H_0 : \sigma_1 = 0$.

När det gäller frekvensen visar tabell 1 att drygt en tredjedel av stationerna har signifikanta trender. Figur 8, som visar en Q-Q-plott av observerade p-värdena gentemot förväntade från ett likelihoodkvot-test av en homogen och icke-homogen Poissonprocess för varje station. De systematiskt lägre observerade p-värdena tyder på att nollhypotesen ofta förkastas. Vidare tyder det även på att det finns en underliggande trend i frekvensen av extrema nederbördshändelser över tid. Notera att även stationer med statistiskt insignifikant trend kan ha en trend i verkligheten, men att den statistiska osäkerheten har varit för stor för att upptäcka trenden.

I figur 9b) kan fördelningen av λ_1 ses. Samtliga negativa trender är insignifikanta och beror troligen på slumpen snarare än någon verklig negativ trend. I figur 9a) kan skattningarna lättare tolkas. Exempelvis har antalet extrema nederbördshändelser vid SMHI:s nordligaste mätstation, Saittarova, ökat med 115%, vilket motsvarar kategorin 88%-124% på kartan.

Sammantaget tyder resultaten på fall (i), diskuterad i avsnitt 2.11, att frekvensen av extrem nederbörd ökar men att fördelningen förblir konstant. Detta eftersom fler trender i λ för Poissonprocessen

har hittats medan σ i GP-fördelningen tycks vara konstant. Detta är i enlighet med vad som upptäckts i östra USA [12]. Det borde innebära att det finns fler trender i årliga maximum än vad som hittats här. Detta eftersom att fler överskridanden förväntas varje år, vilket innebär en större mängd extrema värden och ett större maximum. Att det inte har hittats bör då bero på den statistiska osäkerheten i skattningarna.

5.1 Samhälleliga och etiska aspekter

Datan som används är öppen data från SMHIs väderstationer vilket betyder att det inte är någon konfidentiell data som behandlas. Datan är därtill inte av känslig karaktär och inkräktar inte på personlig identitet.

Resultaten visar att det finns ökande trender i nederbörd, något som kan få stora konsekvenser för samhället. En extrem nederbördshändelse kan på flera sätt skada både individer och infrastrukturen, främst på grund av vattenskador vars reparation kan innebära stora kostnader. På de platser som visar på störst trender, exempelvis Saittarova, Lund eller Håvelund, kan förebyggande åtgärder sättas in. Dessa åtgärder skulle kunna inkludera bättre avrinningssystem i städerna, skyddsvallar eller höjda källarfönster i bostäder. Dock anser projektgruppen att det innan dessa beslut eller åtgärder genomförs hade varit intressant med ytterligare analys av extrem nederbörd i Sverige.

Förutom de direkta konsekvenserna finns även indirekta följder som bilolyckor som konsekvens av översvämmade vägar eller andra skador vilket ökar sjukvårdens belastning. Framkomligheten kan bli försvårad vilket påverkar utryckningsfordon och hemtjänst.

Utöver samhället kan även miljön och den biologiska mångfalden påverkas om extrem nederbörd faller i närheten av bensinmackar, industrier och deponier. Det kan leda till läckage och att föroreningar från dessa sprids och transporteras vidare ut i naturen och skadar ekosystemen där. Det finns även en risk att näringsämnen spolas bort från åkrar och betesmarker vilket kan förstöra grödor samt påverka jordens bördighet. Med detta som bakgrund kan alltså rapportens resultat även användas för att skydda vår miljö och våra ekosystem.

5.2 Slutsatser

Sammantaget visar resultaten på ökad frekvens av extrem nederbörd på många platser i Sverige. Utifrån medianen av trendskattningarna i frekvens beräknas en ökning med 55% på 60 år. Trender i frekvens bör i sin tur leda till större årliga maximum i större utsträckning än vad denna studie visar, något som kan vara värt att ha i åtanke vid samhällsplanering och klimatanpassning.

För vidare forskning skulle temperatur kunna användas som förklarande variabel istället för tid. Detta kringgår problemet att temperaturökningen går snabbare på senare år vilket bör påverka eventuella trender i extrem nederbörd. Fler stationer skulle även kunna inkluderas om större vikt läggs vid att slå ihop närliggande stationer och tillåta stationer med mindre avbrott i mätningar.

Referenser

[1] SMHI. "Klimatförändringen är tydlig redan idag". (2023), URL: https://www.smhi.se/kunskapsbanken/klimat/klimatet-forandras (hämtad 2025-02-07).

[2] Naturvårdsverket. "Klimatförändringar". (2024), URL: https://www.naturvardsverket.se/amnesomraden/klimatforandringar/ (hämtad 2025-05-01).

[3] WWF. "Extremväder". (2025), URL: https://www.wwf.se/klimat/extremvader/#vad-ar-extremvader (hämtad 2025-04-22).

[4] SMHI. "Vattnets kretslopp förändras i varmare klimat". (2024), URL: https://www.smhi.se/klimat/framtidens-klimat/sjoar-och-vattendrag-i-varmare-klimat/vattnets-kretslopp-forandras-i-varmare-klimat (hämtad 2025-04-29).

[5] MSB. "Förbered dig på översvämning". (2024), URL: https://www.msb.se/sv/rad-till-privatpersoner/extremvader-och-naturolyckor/oversvamning/ (hämtad 2025-04-22).

[6] M. för samhällsskydd och beredskap (MSB), *Händelsescenario skyfall*, Tillgänglig: https://www.msb.se/sv/publikationer/handelsescenario-skyfall/, 2020. (hämtad 2025-03-20).

[7] SMHI. "Extrem nederbörd". (2024), URL: https://www.smhi.se/klimat/klimatet-da-och-nu/klimatindikatorer/klimatindikator-extrem-nederbord-1.29819 (hämtad 2025-02-02).

[8] SMHI. "Regn". (u. å.), URL: https://www.smhi.se/kunskapsbanken/meteorologi/regn (hämtad 2025-04-22).

[9] S. Coles, *An Introduction to Statistical Modeling of Extreme Values*. Springer, 2001.

[10] E. Ahlström, N. Hammar, V. Harbander, S. Rödén och M. Savolainen, *Analys av trender i extrema vattenflöden i Sverige*, Kandidatuppsats, Institutionen för Matematiska vetenskaper, Chalmers tekniska högskola och Göteborgs universitet, Göteborg, Sverige, 2024. URL: http://hdl.handle.net/20.500.12380/309265 (hämtad 2025-04-29).

[11] SMHI. "Hur mäts nederbörd?" (u. å.), URL: https://www.smhi.se/kunskapsbanken/meteorologi/regn/hur-mats-nederbord (hämtad 2025-03-21).

[12] H. Ólafsdóttir, *Extreme rainfall modelling under climate change and proper scoring rules for extremes and inference*, doktorsavhandling, Insitutuionen för Matematiska vetenskaper, Göteborgs universitet, Göteborg, Sverige, 2024. [Online]. URL: https://gupea.ub.gu.se/handle/2077/81803 (hämtad 2025-04-16).

[13] D. K. Dey och J. Yan, *Extreme Value Modeling and Risk Analysis: Methods and Applications*. CRC Press, 2016.

[14] SMHI. "Nederbörd". (u. å.), URL: https://www.smhi.se/data/nederbord-och-fuktighet/nederbord/precipitationHourlySum/179960 (hämtad 2025-02-15).

[15] SMHI. "Nederbörd". (u. å.), URL: https://www.smhi.se/kunskapsbanken/meteorologi/nederbord (hämtad 2025-05-07).

[16] SMHI. "SMHIs datapolicy". (2024), URL: https://www.smhi.se/data/om-smhis-data/smhis-datapolicy (hämtad 2025-03-21).

[17] SMHI. "Meteorological Observations - API". (u. å.), URL: https://opendata.smhi.se/metobs/api (hämtad 2025-02-15).

[18] USGS. "Explanations for the National Water Conditions". (2016), URL: https://water.usgs.gov/nwc/explain_data.html (hämtad 2025-04-11).

[19] H. Björklund, E. Gustafsson, O. Morsing och E. Nyström. "Kandidatarbetets GitHub-sida". (2025), URL: https://github.com/morolof/KandidatarbeteVT25-MVEX11-VT25-15.

Användande av AI

AI, mer specifikt ChatGPT, har använts i syfte att effektivisera arbetet.

Den har bland annat använts för att korrekturläsa och hitta stavfel samt grammatiska fel i rapporten. Detta har gjorts genom att skicka färdigskrivna stycken och låta AI:n kommentera felen, vilka sedan manuellt korrigerats. På detta sätt kunde fel som gruppen missat, rättas till. Samtidigt gavs förslag på vad som kunde förbättras i texten men AI:n har inte själv fått generera text som används i rapporten. Det har även använts som ett språkverktyg, genom att hitta synonymer och andra sätt att uttrycka sig för att få en mer varierande text.

ChatGPT har även använts för att generera kod. Dels kod i Python för att hämta, förbehandla och visualisera data och dels kod i R för extremvärdesanalysen. Koden som genererats utgör enklare delar av koden och har genererats enligt tydlig prompt. All kod som genererats av AI har granskats och testats.

Ytterligare genererades LaTeX-tabeller med ChatGPT och det användes som stöd för LaTeXkommandon.

A Appendix 1 – Tabeller

Poissonprocess resultat

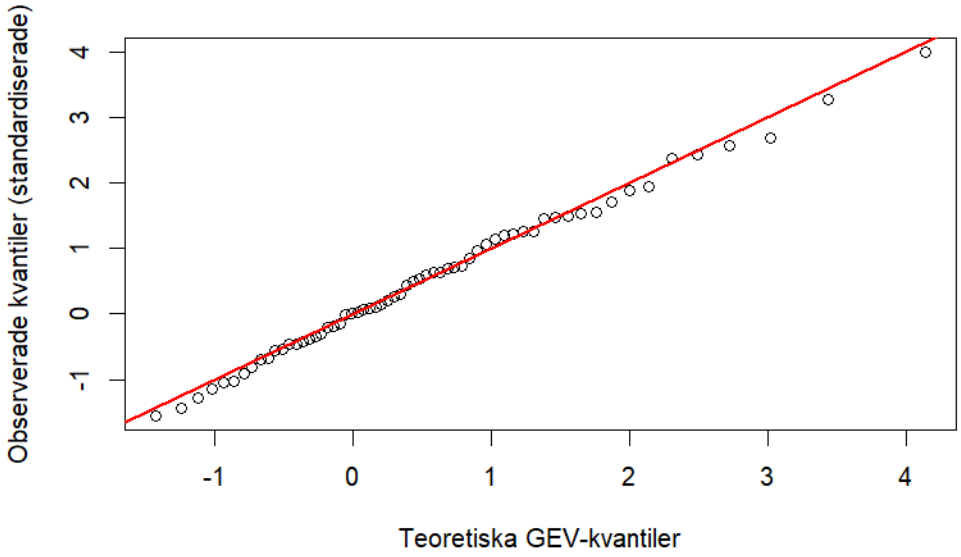
Tabellen nedan visar parameterskattningarna från den inhomogena Poissonprocessen för samtliga stationer där datamängd 1 använts. Det beräknade p-värdet från likelihoodkvot-testet samt signifikansen visas även.

Tabell A.1: Resultat från Poissonprocess med datamängd 1

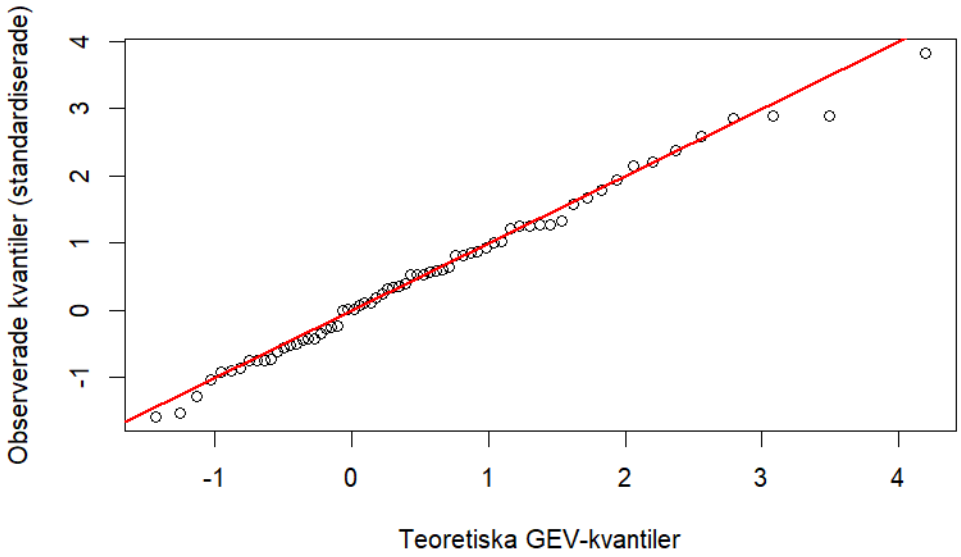
Station	λ_0^*	SE(λ_0^*)	λ_1^*	SE(λ_1^*)	p-vär.	Signif.	Första obs.
Avasjö-Borgafjäll D	-5.71	0.20	2.8e-05	1.3e-05	0.03	S	1957-07-01
Borås	-5.40	0.19	6.6e-06	1.4e-05	0.65	F	1962-07-01
Finnbacka D	-5.18	0.19	-1.6e-05	1.5e-05	0.27	F	1962-12-26
Gendalen	-5.67	0.21	2.9e-05	1.5e-05	0.04	S	1962-10-01
Grönliden	-5.82	0.21	3.5e-05	1.3e-05	0.01	S	1955-11-13
Hallstaberg	-5.45	0.19	1.1e-05	1.3e-05	0.42	F	1959-08-01
Hanö A	-5.44	0.13	4.0e-06	4.2e-06	0.34	F	1881-11-01
Heden	-5.52	0.20	1.8e-05	1.4e-05	0.21	F	1962-04-23
Hyltan	-5.67	0.19	2.5e-05	1.2e-05	0.03	S	1952-08-17
Härnösand	-5.80	0.14	1.4e-05	3.9e-06	0.00	S	1870-11-26
Håvelund	-5.81	0.19	3.1e-05	1.1e-05	0.00	S	1948-02-06
Höglekardalen	-5.40	0.20	2.6e-07	1.5e-05	0.99	F	1962-02-01
Höljes	-5.55	0.19	1.4e-05	1.2e-05	0.24	F	1953-10-04
Jäckvik	-5.56	0.18	1.4e-05	9.9e-06	0.15	F	1943-11-01
Kilagården	-5.74	0.20	2.9e-05	1.3e-05	0.02	S	1956-01-18
Klippan	-5.63	0.18	1.5e-05	1.0e-05	0.15	F	1945-01-01
Kristinehamn	-5.62	0.20	2.3e-05	1.4e-05	0.09	F	1959-10-18
Ljusnedal	-5.37	0.16	1.6e-06	7.8e-06	0.84	F	1930-01-01
Lund	-5.78	0.21	3.7e-05	1.4e-05	0.01	S	1961-01-01
Mariestad	-5.80	0.21	3.6e-05	1.3e-05	0.01	S	1958-08-01
Mångsbodarna	-5.71	0.19	2.8e-05	1.2e-05	0.02	S	1952-09-27
Ockelbo	-5.55	0.18	1.4e-05	1.0e-05	0.17	F	1945-01-01
Oxelösund	-5.47	0.20	1.1e-05	1.5e-05	0.48	F	1963-07-01
Prästkulla	-5.47	0.17	8.9e-06	1.0e-05	0.37	F	1945-01-01
Ramsjöholm	-5.76	0.20	2.9e-05	1.2e-05	0.01	S	1951-11-16
Saittarova D	-5.84	0.20	3.4e-05	1.2e-05	0.00	S	1952-02-01
Sjögärde	-5.51	0.20	1.4e-05	1.4e-05	0.31	F	1960-01-09
Säffle	-5.67	0.19	2.3e-05	1.2e-05	0.04	S	1951-10-01
Sörbytorp	-5.38	0.17	4.5e-06	9.9e-06	0.65	F	1945-01-01
Traneberg	-5.68	0.19	2.4e-05	1.1e-05	0.04	S	1951-02-11
Ungsberg	-5.57	0.19	1.8e-05	1.3e-05	0.16	F	1956-01-31
Uppsala	-5.24	0.11	-3.0e-06	2.8e-06	0.27	F	1836-01-01
Ytterberg D	-5.60	0.19	1.9e-05	1.2e-05	0.12	F	1953-10-31
Åby	-5.39	0.17	4.6e-06	1.0e-05	0.64	F	1945-02-28
Åkroken D	-5.52	0.19	1.2e-05	1.3e-05	0.33	F	1955-12-16
Åtorp	-5.23	0.18	-8.4e-06	1.4e-05	0.55	F	1961-01-01
Örnsköldsvik	-5.69	0.21	2.8e-05	1.5e-05	0.06	F	1963-10-01

B Appendix 2 – Q-Q plottar

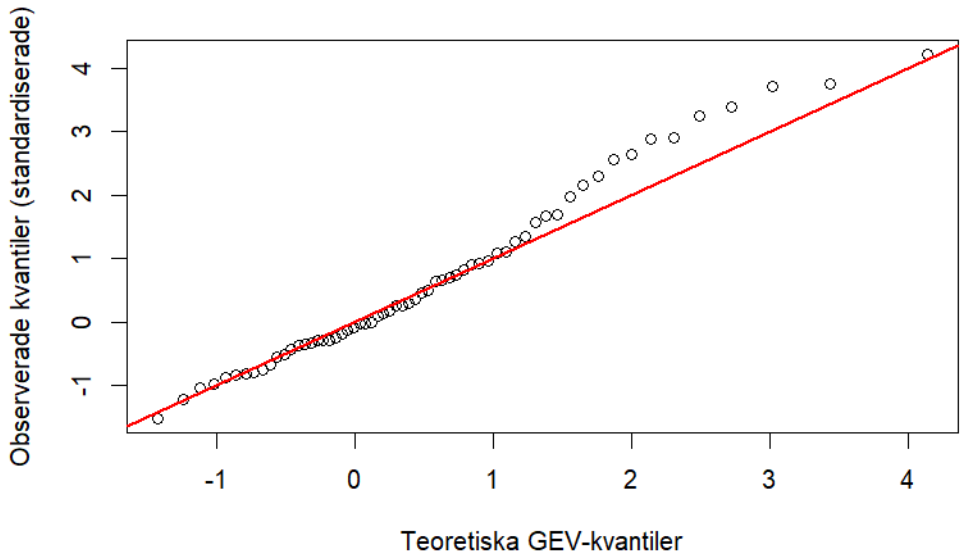
Nedan visas 6 Q-Q-plottar för skattade GEV- samt GP-fördelningar för stationerna i Borås, Mariestad och Gendalen. Därefter visas en Q-Q-plott med p-värden från likelihoodkvot-test mellan GP- och exponentialfördelningar. För samtliga har datamängd 1 används.



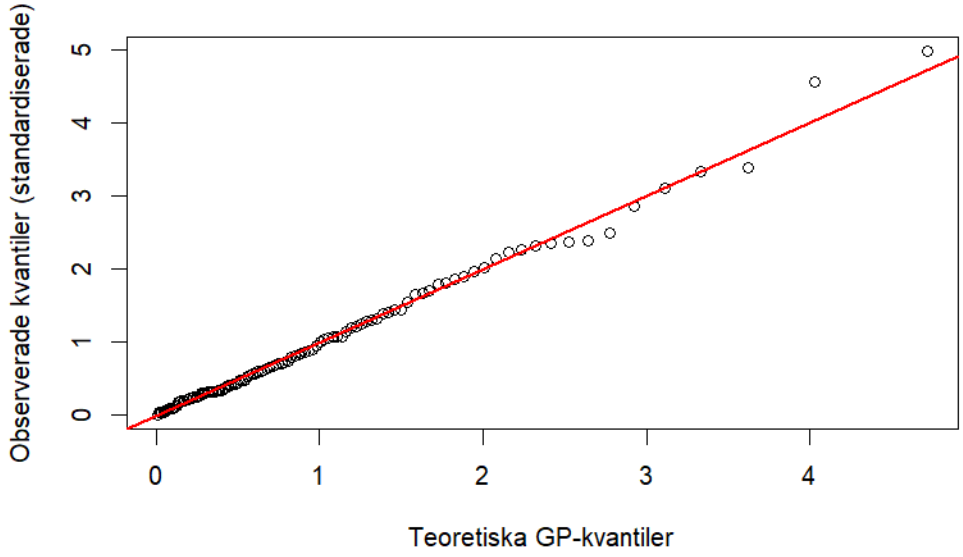
Figur 10: Q-Q-plott för GEV-fördelningen med trend i μ och σ för årliga maximum i Borås.



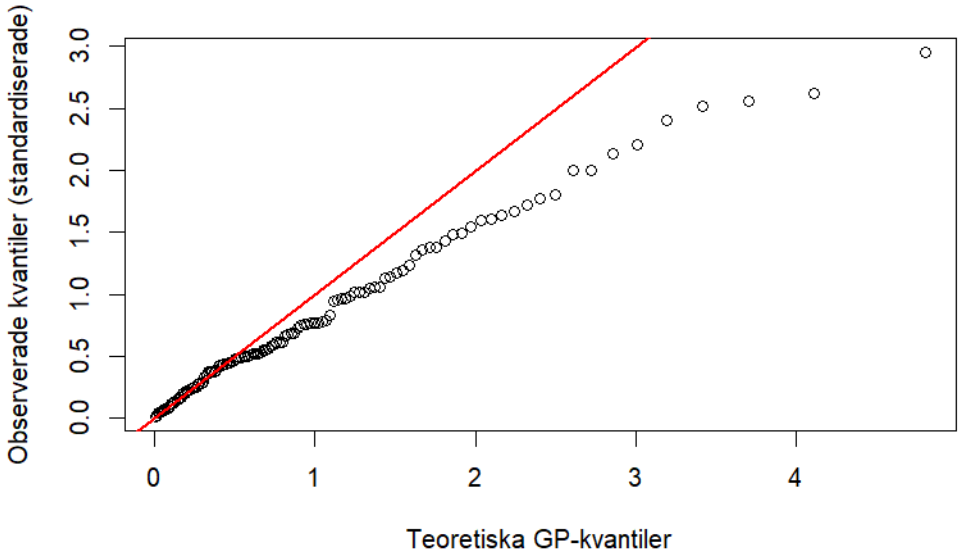
Figur 11: Q-Q-plott för GEV-fördelningen med trend i μ och σ för årliga maximum i Mariestad.



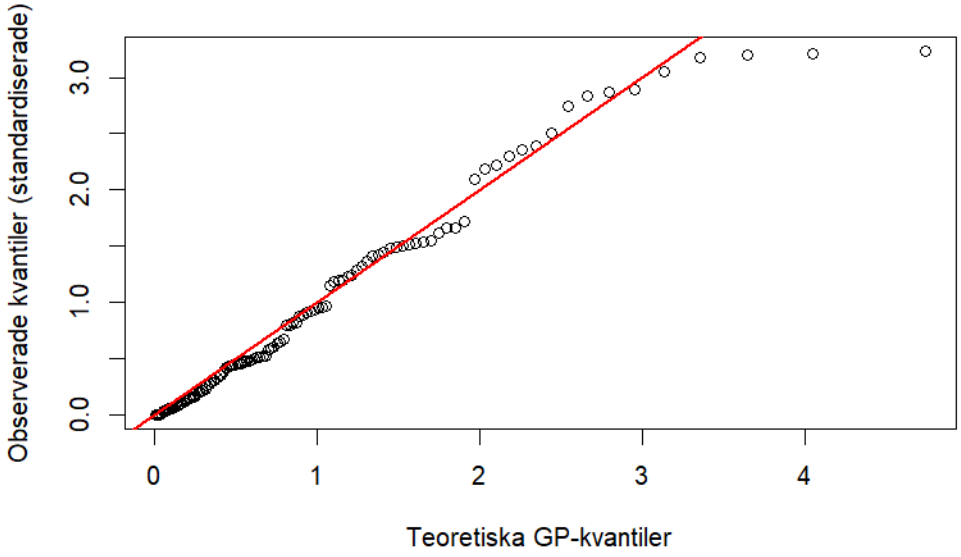
Figur 12: Q-Q-plott för GEV-fördelningen med trend i μ och σ för årliga maximum i Gendalen.



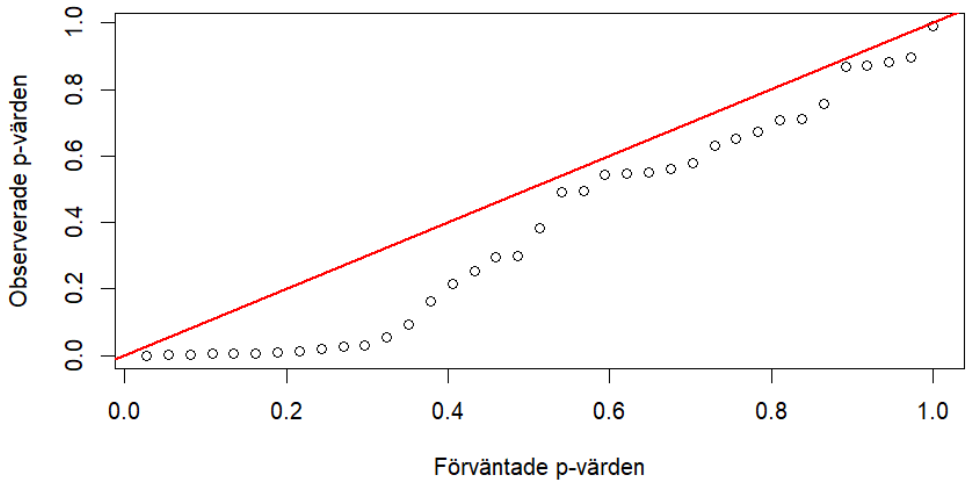
Figur 13: Q-Q-plott för Tröskelmetoden för Borås.



Figur 14: Q-Q-plott för Tröskelmetoden för Mariestad.



Figur 15: Q-Q-plott för Tröskelmetoden för Gendalen.



Figur 16: Plott av observerade mot förväntade p-värden från likelihood-kvottest mellan exponential och GP fördelningar, anpassade till nederbördsdata från varje station. Den röda linjen är linjen $y = x$.

C Appendix 3 – Kod

C.1 Python kod för insamling och behandling av data

smhi_select_stations.py

```
# Olof
# 2025

import requests
from datetime import datetime, timedelta
import math
import json

# This code looks at the metadata for all stations to create a list of
    interesting stations for our project

# Function to get a stations metadata given a specific smhi_param, see
    https://opendata.smhi.se/metobs/resources/parameter
def get_stations_metadata(smhi_param):
    url_stations = f"https://opendata-download-metobs.smhi.se/api/version
        /1.0/parameter/{smhi_param}.json"
    r_stations = requests.get(url_stations)
    stations_metadata = json.loads(r_stations.text)
    return stations_metadata['station']

# Function to convert Unix epoch (milliseconds) to a readable datetime
def convert_unix_epoch(milliseconds):
    seconds = milliseconds / 1000
    if seconds < 0:
        return (datetime(1970, 1, 1) + timedelta(seconds=seconds)).strftime
            ('%Y-%m-%d␣%H:%M:%S')
    else:
        return datetime.utcfromtimestamp(seconds).strftime('%Y-%m-%d␣%H:%M
            :%S')
```

```
# Function to calculate Euclidean distance between two points (latitude and
    longitude)
def calculate_distance(lat1, lon1, lat2, lon2):
    return math.sqrt((lat2 - lat1)**2 + (lon2 - lon1)**2)

# Function to calculate the interval between two timestamps in years
def interval_in_years(from_ts, to_ts):
    return (to_ts - from_ts) / (1000 * 60 * 60 * 24 * 365.25)  # Convert ms
        to years

# Change this parameter to either get daily rainfall, 15 min rainfall, etc
   see SMHI API parametrization for more
# Remember to change header accordingly as the csv files might look
   different
# 5 - daily rainfall, 14 - rainfall 15 min
smhi_parameter = 5
output_file_name = "csv_and_json/filtered_stations_60.json"

# Get stations metadata
stations_metadata = get_stations_metadata(smhi_parameter)

# Compute extracted data and minimum distance to other stations
extracted_data = []
for i, station in enumerate(stations_metadata):
    station_data = {
        'id': station['id'],
        'name': station['name'],
        'latitude': station['latitude'],
        'longitude': station['longitude'],
        'from_timestamp': station['from'],  # Keep original timestamp for
            calculations
        'to_timestamp': station['to'],      # Keep original timestamp for
            calculations
        'from': convert_unix_epoch(station['from']),  # Human-readable date
        'to': convert_unix_epoch(station['to']),      # Human-readable date
        'active': station['active']
    }
    # Calculate distances to all other stations
    distances = [
        (calculate_distance(
            station['latitude'], station['longitude'],
            other['latitude'], other['longitude']
        ), other['name'])
        for j, other in enumerate(stations_metadata) if i != j
    ]
    # Find the minimum distance and the corresponding station
    if distances:
        min_distance, closest_station = min(distances, key=lambda x: x[0])
        station_data['min_distance_to_other_stations'] = min_distance
        station_data['closest_station'] = closest_station
    else:
        station_data['min_distance_to_other_stations'] = None
        station_data['closest_station'] = None

    extracted_data.append(station_data)

# Filter the data to keep only active stations and those with a x+ year
   interval
year_threshold = 60
filtered_data = [
    station for station in extracted_data
```

```
    if station['active'] and interval_in_years(station['from_timestamp'],
        station['to_timestamp']) >= year_threshold
    #if interval_in_years(station['from_timestamp'], station['to_timestamp
        ']) >= year_threshold
]
print(len(filtered_data))

# Save the filtered data as a JSON file
with open(output_file_name, "w", encoding="utf-8") as json_file:
    json.dump(filtered_data, json_file, indent=4)
```

smhi_create_database.py

```
# Olof
# 2025

import requests
import json
import pandas as pd
from io import StringIO

# This code collect the data for each station from the list created in
    smhi_select_stations.py

# Change this parameter to either get daily rainfall, 15 min rainfall, etc
    see SMHI API parametrization for more
# Remember to change header accordingly as the csv files might look
    different
# 5 - daily rainfall, 14 - rainfall 15 min
smhi_parameter = 5
output_file_name = "csv_and_json/filtered_stations_60_data.csv"

# Load filtered data from the JSON file
with open("csv_and_json/filtered_stations_60.json", "r", encoding="utf-8")
    as json_file:
    loaded_data = json.load(json_file)

# Extract station names and IDs
station_names = [station['name'] for station in loaded_data]
station_ids = [station['id'] for station in loaded_data]

# Define the header line you want to find
header = "Fr n␣Datum␣Tid␣(UTC);Till␣Datum␣Tid␣(UTC);Representativt␣dygn;
    Nederb rdsm ngd;Kvalitet;;Tidsutsnitt:" # smhi_parameter = 5
#header = "Datum;Tid (UTC);Nederb rdsm ngd;Kvalitet;;Tidsutsnitt:" #
    smhi_parameter = 14

# Initialize an empty DataFrame to store all stations' data
all_stations_df = pd.DataFrame()

# Loop through all stations and collect data
for station_id, station_name in zip(station_ids, station_names):
    print(f"Fetching␣data␣for␣station:␣{station_name}␣(ID:␣{station_id})")

    # Construct the URL
    url = f"https://opendata-download-metobs.smhi.se/api/version/latest/
        parameter/{smhi_parameter}/station/{station_id}/period/corrected-
        archive/data.csv"

    # Request data from the API
    r_data = requests.get(url)
```

```
        csv_data = r_data.content.decode("utf-8")  # Convert bytes to string
        lines = csv_data.splitlines()

        # Find the number of rows to skip
        skip_rows = next((i for i, line in enumerate(lines) if line.startswith(
            header)), None)
        if skip_rows is None:
            print(f"Skipping station {station_name} due to missing header.")
            continue  # Skip this station if the header isn't found

        # Load data into pandas
        df = pd.read_csv(StringIO(csv_data), sep=";", skiprows=skip_rows,
            usecols=[0, 1, 2, 3, 4])  # smhi_parameter = 5
        #df = pd.read_csv(StringIO(csv_data), sep=";", skiprows=skip_rows,
            usecols=[0, 1, 2, 3]) # smhi_parameter = 14

        # Rename columns for clarity
        df.columns = ["From Date", "To Date", "Date", "Rainfall (mm)", "Quality
            "] # smhi_parameter = 5
        #df.columns = ["Date", "Time (UTC)", "Rainfall (mm)", "Quality"] #
            smhi_parameter = 14

        # Convert 'Date' column to datetime and remove time component
        df["Date"] = pd.to_datetime(df["Date"]).dt.date

        # Add station information
        df["Station Name"] = station_name
        df["Station ID"] = station_id

        # Append to the main DataFrame
        all_stations_df = pd.concat([all_stations_df, df], ignore_index=True)

# Save collected data to a CSV file
all_stations_df.to_csv(output_file_name, index=False, encoding="utf-8")

print(f"Data collection complete. Saved to '{output_file_name}'.")
```

remove_data_gaps_v2.py

```
# Olof
# 2025

import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import json

# This code removes all the gaps in the data and cuts the data accordingly
    to get the maximum length of the time series

# Load filtered data from the JSON file
with open("csv_and_json/filtered_stations_60.json", "r", encoding="utf-8")
    as json_file:
    loaded_data = json.load(json_file)

# Extract station names
station_names = [station['name'] for station in loaded_data]

# Import the data from CSV
df = pd.read_csv("csv_and_json/filtered_stations_60_data.csv")
```

```
# Convert 'Date' column to datetime
df['Date'] = pd.to_datetime(df['Date'])

# Dictionary to store dataframes for stations without large gaps
filtered_stations = {}

gap_threshold = pd.Timedelta(days=2)

# Loop over each station and check for gaps
for station in station_names:
    print("Checking station:", station)
    # Filter the data for the current station
    station_data = df[df['Station Name'] == station].sort_values('Date')

    # Calculate the difference between consecutive dates
    date_diffs = station_data['Date'].diff()

    # Find the rows where the date_diff is greater than the gap threshold
    large_gaps = date_diffs[date_diffs > gap_threshold]

    if not large_gaps.empty:
        latest_large_gap_idx = large_gaps.index[-1]  # last occurrence
        latest_large_gap_date = station_data.loc[latest_large_gap_idx, '
            Date']
        gap_at_latest_large_gap = date_diffs.loc[latest_large_gap_idx]

        print(f"Latest large gap for station {station} at {
            latest_large_gap_date} with gap {gap_at_latest_large_gap}.")

        # Now check the time from latest gap to end
        station_data_after_gap = station_data[station_data['Date'] >=
            latest_large_gap_date].copy()

        if station_data_after_gap.empty:
            print(f"No data after latest large gap for station {station}.
                Skipping.")
            continue

        time_span = station_data_after_gap['Date'].max() -
            station_data_after_gap['Date'].min()

        if time_span >= pd.Timedelta(days=60 * 365.25):
            # Check gaps again in the post-gap data
            date_diffs_post_gap = station_data_after_gap['Date'].diff()

            if date_diffs_post_gap.max() <= gap_threshold:
                filtered_stations[station] = station_data_after_gap
                print(f"Station {station} included from {
                    latest_large_gap_date.date()} onward.")
            else:
                print(f"Station {station} still has large gaps after latest
                    gap. Skipping.") # This should not happen but lets
                    check anyhow.
        else:
            print(f"Station {station} has less than 60 years of data after
                latest gap. Skipping.")
    else:
        # No large gaps found, keep the full station data
        filtered_stations[station] = station_data
        print(f"No large gaps found for station {station}. Station included
            .")
```

```
# Create a new combined dataframe with filtered stations
filtered_df = pd.concat(filtered_stations.values())

# Save filtered data to a new CSV
filtered_df.to_csv("csv_and_json/final_filtered_station_data.csv", index=
    False)

# Display a message summarizing the result
print(f"Number␣of␣stations␣without␣gaps␣greater␣than␣chosen␣threshold:␣{len
    (filtered_stations)}")
```

create_aligned_dataset.py

```
# Olof
# 2025

import pandas as pd

# This code creates the "aligned" dataset, mentioned as dataset 2 in the
    text

# Load the rainfall data
df = pd.read_csv("csv_and_json/final_filtered_station_data.csv")

# Convert Date column to datetime
df['Date'] = pd.to_datetime(df['Date'])

# Get start and end date for each station
station_ranges = df.groupby('Station␣Name')['Date'].agg(['min', 'max'])

# Get the latest start date (common across all stations)
common_start = station_ranges['min'].max()

# Set the custom end date to match water year (end of September)
custom_end = pd.Timestamp('2024-09-30')

# Filter for the overlapping custom range
df_aligned = df[(df['Date'] >= common_start) & (df['Date'] <= custom_end)].
    copy()

# Optional: Check that all stations have same number of entries
counts = df_aligned.groupby('Station␣Name').size()
print(counts)

# Save to CSV
df_aligned.to_csv("csv_and_json/aligned_final_filtered_station_data.csv",
    index=False)
```

C.2 R kod för extremvärdesanalysen

GEVfit_v2.R

```
# Olof
# 2025
library(extRemes)
library(dplyr)
library(lubridate)

# List of datasets to loop over
```

```
dataset_files <- list(
  "Original" = "final_filtered_station_data.csv",
  "Aligned" = "aligned_final_filtered_station_data.csv"
)

# Initialize final results
final_results_df <- data.frame()

# Loop over datasets
for (dataset_name in names(dataset_files)) {
  file_path <- dataset_files[[dataset_name]]
  data <- read.csv(file_path)
  colnames(data) <- c("From_Date", "To_Date", "Date", "Rainfall_mm", "
      Quality", "Station_name", "Station_ID")

  station_name_list <- unique(data$Station_name)
# Loop over stations
    for (station in station_name_list) {

      # Fix water-years
      station_data <- data %>%
        filter(Station_name == station) %>%
        mutate(Date = as.Date(Date),
               Water_Year = if_else(month(Date) >= 10, year(Date) + 1, year
                   (Date)),
               Dataset_Type = dataset_name) %>%
        group_by(Station_ID) %>%
        arrange(Date) %>%
        mutate(
          First_Date = first(Date),
          Last_Date = last(Date),
          First_Oct1 = if_else(
            month(First_Date) == 10 & day(First_Date) == 1,
            First_Date,
            make_date(year(First_Date) + if_else(month(First_Date) >= 10,
                1, 0), 10, 1)
          ),
          Last_Water_Year = max(Water_Year),
          Last_Sep30 = if_else(Dataset_Type == "Aligned",
                               make_date(Last_Water_Year, 9, 30),  # For
                                   aligned dataset
                               make_date(Last_Water_Year - 1, 9, 30)  # For
                                    original dataset
          )
        ) %>%
        filter(Date >= First_Oct1 & Date <= Last_Sep30) %>%
        select(-First_Date, -Last_Date, -First_Oct1, -Last_Water_Year, -
            Last_Sep30) %>%
        ungroup()


      station_data <- station_data %>%
        mutate(Days_Elapsed = as.numeric(Date - min(Date, na.rm = TRUE)),
               Year = year(Date),
               Years_Elapsed = as.numeric(difftime(Date, min(Date, na.rm =
                   TRUE), units = "days")) / 365.25)

      # These are for the results
      first_obs_date = min(station_data$Date)
      last_obs_date = max(station_data$Date)
```

```
elapsed_years <- as.numeric(difftime(last_obs_date, first_obs_date,
    units = "days")) / 365.25

# Annual maxes
rain_ann_max_with_indices <- station_data %>%
  group_by(Water_Year) %>%
  filter(Rainfall_mm == max(Rainfall_mm, na.rm = TRUE)) %>%
  slice(1) %>%
  ungroup() %>%
  select(Date, Max_Rainfall = Rainfall_mm, Years_Elapsed)

data_df <- data.frame(ann_max = rain_ann_max_with_indices$Max_
    Rainfall,
                      years = rain_ann_max_with_indices$Years_Elapsed
                          )

# Fit stationary GP models and store results
stat_gev <- fevd(x = rain_ann_max_with_indices$Max_Rainfall, type = "
    GEV", verbose = FALSE)
stat_gumbel <- fevd(x = rain_ann_max_with_indices$Max_Rainfall, type
    = "Gumbel", verbose = FALSE)

stat_summary_gev <- summary(stat_gev)
stat_summary_gumbel <- summary(stat_gumbel)

print(station)
print(dataset_name)

final_results_df <- rbind(final_results_df, data.frame(
  Station = station,
  Dataset = dataset_name,
  Type = "GEV",
  Model = "Stationary",
  Location_Intercept = as.numeric(stat_summary_gev$par["location"]),
  Location_SE_Intercept = as.numeric(stat_summary_gev$se["location"])
      ,
  Location_Slope = NA,
  Location_SE_Slope = NA,
  Scale_Intercept = as.numeric(stat_summary_gev$par["scale"]),
  Scale_SE_Intercept = as.numeric(stat_summary_gev$se["scale"]),
  Scale_Slope = NA,
  Scale_SE_Slope = NA,
  Shape = as.numeric(stat_summary_gev$par["shape"]),
  Shape_SE = as.numeric(stat_summary_gev$se["shape"]),
  Trend_lr_pval = NA,
  Shape_lr_pval = NA,
  First_Observation_Date = as.Date(first_obs_date),
  Last_Observation_Date = as.Date(last_obs_date),
  Elapsed_Years = elapsed_years
))

final_results_df <- rbind(final_results_df, data.frame(
  Station = station,
  Dataset = dataset_name,
  Type = "Gumbel",
  Model = "Stationary",
  Location_Intercept = as.numeric(stat_summary_gumbel$par["location"
      ]),
  Location_SE_Intercept = as.numeric(stat_summary_gumbel$se["location
      "]),
  Location_Slope = NA,
```

```
  Location_SE_Slope = NA,
  Scale_Intercept = as.numeric(stat_summary_gumbel$par["scale"]),
  Scale_SE_Intercept = as.numeric(stat_summary_gumbel$se["scale"]),
  Scale_Slope = NA,
  Scale_SE_Slope = NA,
  Shape = as.numeric(stat_summary_gumbel$par["shape"]),
  Shape_SE = as.numeric(stat_summary_gumbel$se["shape"]),
  Trend_lr_pval = NA,
  Shape_lr_pval = NA,
  First_Observation_Date = as.Date(first_obs_date),
  Last_Observation_Date = as.Date(last_obs_date),
  Elapsed_Years = elapsed_years
))

# Function to extract parameters from non-stat summaries
extract_pars <- function(fit_summary) {
  pars <- fit_summary$par
  ses <- fit_summary$se
  list(
    mu0 = ifelse("mu0" %in% names(pars), as.numeric(pars["mu0"]),
        ifelse("location" %in% names(pars), as.numeric(pars["location
        "]), NA)),
    mu1 = ifelse("mu1" %in% names(pars), as.numeric(pars["mu1"]), NA)
        ,
    phi0 = ifelse("phi0" %in% names(pars), as.numeric(pars["phi0"]),
        ifelse("scale" %in% names(pars), as.numeric(pars["scale"]),
        NA)),
    phi1 = ifelse("phi1" %in% names(pars), as.numeric(pars["phi1"]),
        NA),
    shape = ifelse("shape" %in% names(pars), as.numeric(pars["shape"
        ]), NA),
    se_mu0 = ifelse("mu0" %in% names(ses), as.numeric(ses["mu0"]),
        ifelse("location" %in% names(ses), as.numeric(ses["location"
        ]), NA)),
    se_mu1 = ifelse("mu1" %in% names(ses), as.numeric(ses["mu1"]), NA
        ),
    se_phi0 = ifelse("phi0" %in% names(ses), as.numeric(ses["phi0"]),
         ifelse("scale" %in% names(ses), as.numeric(ses["scale"]), NA
        )),
    se_phi1 = ifelse("phi1" %in% names(ses), as.numeric(ses["phi1"]),
         NA),
    se_shape = ifelse("shape" %in% names(ses), as.numeric(ses["shape"
        ]), NA)
  )
}

# Non-stat models
model_defs <- list(
  "Multiplicative␣(mu)" = list(location.fun = ~years, scale.fun = ~1)
      ,
  "Multiplicative␣(sigma)" = list(location.fun = ~1, scale.fun = ~
      years),
  "Multiplicative␣(mu/sigma)" = list(location.fun = ~years, scale.fun
       = ~years)
)


for (model_name in names(model_defs)) {

  model_def <- model_defs[[model_name]]
```

```
# Fit GEV nonstationary
gev_nonstat_fit <- fevd(x = rain_ann_max_with_indices$Max_Rainfall,
                        data = data_df,
                        type = "GEV",
                        location.fun = model_def$location.fun,
                        scale.fun = model_def$scale.fun,
                        use.phi = TRUE,
                        time.units = "years",
                        verbose = FALSE)

gev_summary <- summary(gev_nonstat_fit)
gev_pars <- extract_pars(gev_summary)

# Fit Gumbel nonstationary
gumbel_nonstat_fit <- fevd(x = rain_ann_max_with_indices$Max_
    Rainfall,
                           data = data_df,
                           type = "Gumbel",
                           location.fun = model_def$location.fun,
                           scale.fun = model_def$scale.fun,
                           use.phi = TRUE,
                           time.units = "years",
                           verbose = FALSE)

gumbel_summary <- summary(gumbel_nonstat_fit)
gumbel_pars <- extract_pars(gumbel_summary)

# LR test between stationary GEV and nonstationary GEV
lr_stat_nonstat_gev <- lr.test(stat_gev, gev_nonstat_fit, alpha =
    0.05)

# LR test between stationary Gumbel and nonstationary Gumbel
lr_stat_nonstat_gumbel <- lr.test(stat_gumbel, gumbel_nonstat_fit,
    alpha = 0.05)

# LR test between Gumbel nonstationary and GEV nonstationary
lr_gumbel_gev <- lr.test(gumbel_nonstat_fit, gev_nonstat_fit, alpha
     = 0.05)

# Save the GEV nonstationary fit
final_results_df <- rbind(final_results_df, data.frame(
  Station = station,
  Dataset = dataset_name,
  Type = "GEV",
  Model = model_name,
  Location_Intercept = gev_pars$mu0,
  Location_SE_Intercept = gev_pars$se_mu0,
  Location_Slope = gev_pars$mu1,
  Location_SE_Slope = gev_pars$se_mu1,
  Scale_Intercept = gev_pars$phi0,
  Scale_SE_Intercept = gev_pars$se_phi0,
  Scale_Slope = gev_pars$phi1,
  Scale_SE_Slope = gev_pars$se_phi1,
  Shape = gev_pars$shape,
  Shape_SE = gev_pars$se_shape,
  Trend_lr_pval = as.numeric(lr_stat_nonstat_gev$p.value),
  Shape_lr_pval = NA,
  First_Observation_Date = as.Date(first_obs_date),
  Last_Observation_Date = as.Date(last_obs_date),
  Elapsed_Years = elapsed_years
))
```

```
        # Save the Gumbel nonstationary fit
        final_results_df <- rbind(final_results_df, data.frame(
          Station = station,
          Dataset = dataset_name,
          Type = "Gumbel",
          Model = model_name,
          Location_Intercept = gumbel_pars$mu0,
          Location_SE_Intercept = gumbel_pars$se_mu0,
          Location_Slope = gumbel_pars$mu1,
          Location_SE_Slope = gumbel_pars$se_mu1,
          Scale_Intercept = gumbel_pars$phi0,
          Scale_SE_Intercept = gumbel_pars$se_phi0,
          Scale_Slope = gumbel_pars$phi1,
          Scale_SE_Slope = gumbel_pars$se_phi1,
          Shape = gumbel_pars$shape,
          Shape_SE = gumbel_pars$se_shape,
          Trend_lr_pval = as.numeric(lr_stat_nonstat_gumbel$p.value),
          Shape_lr_pval = as.numeric(lr_gumbel_gev$p.value),
          First_Observation_Date = as.Date(first_obs_date),
          Last_Observation_Date = as.Date(last_obs_date),
          Elapsed_Years = elapsed_years
        ))
      }
  }
}

# Check p-vals
final_results_df <- final_results_df %>%
  mutate(Significant_Trend = ifelse(Model %in% c("Multiplicative (mu)", "
      Multiplicative (sigma)", "Multiplicative (mu/sigma)") & !is.na(Trend_
      lr_pval), Trend_lr_pval < 0.05, NA))

  final_results_df <- final_results_df %>%
  mutate(Significant_Shape = ifelse(Model %in% c("Multiplicative (mu)", "
      Multiplicative (sigma)", "Multiplicative (mu/sigma)") & !is.na(Shape_
      lr_pval), Shape_lr_pval < 0.05, NA))

print("Done!")

library(readr)
write_csv(final_results_df, "GEVresults2.csv")
```

GPfit_v2.R

```
# Olof
# 2025
library(extRemes)
library(dplyr)
library(lubridate)

# List of datasets to loop over
dataset_files <- list(
  "Original" = "final_filtered_station_data.csv",
  "Aligned" = "aligned_final_filtered_station_data.csv"
)

# Initialize final results dataframe
final_results_df <- data.frame(
  Station = character(),
  Dataset = character(),
```

```
  Type = character(),
  Model = character(),
  Scale_Intercept = numeric(),
  Scale_SE_Intercept = numeric(),
  Scale_Slope = numeric(),
  Scale_SE_Slope = numeric(),
  Shape = numeric(),
  Shape_SE = numeric(),
  Trend_lr_pval = numeric(),
  Shape_lr_pval = numeric(),
  First_Observation_Date = as.Date(character()),
  Last_Observation_Date = as.Date(character()),
  Elapsed_Years = numeric(),
  stringsAsFactors = FALSE
)

# Loop over each dataset
for (dataset_name in names(dataset_files)) {
  file_path <- dataset_files[[dataset_name]]
  data <- read.csv(file_path)
  colnames(data) <- c("From_Date", "To_Date", "Date", "Rainfall_mm", "
      Quality", "Station_name", "Station_ID")

  # Get unique station names
  station_name_list <- unique(data$Station_name)

    # Loop over each station
    for (station in station_name_list) {

      # Filter for current station
      station_data <- data %>%
        filter(Station_name == station) %>%
        mutate(Date = as.Date(Date),
               Days_Elapsed = as.numeric(Date - min(Date, na.rm = TRUE)),
               Years_Elapsed = as.numeric(difftime(Date, min(Date, na.rm =
                   TRUE), units = "days")) / 365.25)

      # Calculate first and last observation dates
      first_obs_date = min(station_data$Date, na.rm = TRUE)
      last_obs_date = max(station_data$Date, na.rm = TRUE)

      # Calculate elapsed years based on these new dates
      elapsed_years <- as.numeric(difftime(last_obs_date, first_obs_date,
          units = "days")) / 365.25

      # Define threshold (e.g., 99.5th percentile)
      threshold <- quantile(station_data$Rainfall_mm, 0.995, na.rm = TRUE)

      # Declustering
      station_data_declus <- decluster(station_data$Rainfall_mm, threshold)

      # Fit stationary GP models
      stat_fit_gp <- fevd(x = station_data_declus, threshold = threshold,
          type = "GP", verbose = FALSE)
      stat_fit_gp_summary <- summary(stat_fit_gp)

      stat_fit_exp <- fevd(x = station_data_declus, threshold = threshold,
          type = "Exponential", verbose = FALSE)
      stat_fit_exp_summary <- summary(stat_fit_exp)

      # Dataframe for model fitting
```

```
GP_df_year_elaps <- data.frame(rain_data = station_data_declus, years
    = station_data$Years_Elapsed)

print(station)
print(dataset_name)

# Fit non-stationary models
nonstat_fit_gp <- fevd(x = station_data_declus,
             data = GP_df_year_elaps,
             threshold = threshold,
             scale.fun = ~years,
             type = "GP",
             use.phi = TRUE,
             time.units = "years",
             verbose = FALSE)
nonstat_fit_gp_summary <- summary(nonstat_fit_gp)

nonstat_fit_exp <- fevd(x = station_data_declus,
                        data = GP_df_year_elaps,
                        threshold = threshold,
                        scale.fun = ~years,
                        type = "Exponential",
                        use.phi = TRUE,
                        time.units = "years",
                        verbose = FALSE)

nonstat_fit_exp_summary <- summary(nonstat_fit_exp)

# Likelihood ratio tests: stationary vs non stationary models
lr_trend_gp <- lr.test(stat_fit_gp, nonstat_fit_gp, alpha = 0.05)
lr_trend_exp <- lr.test(stat_fit_exp, nonstat_fit_exp, alpha = 0.05)

# Likelihood ratio test: GP vs Exponential
lr_shape <- lr.test(nonstat_fit_gp, nonstat_fit_exp, alpha = 0.05)


# Save results for stationary models
final_results_df <- rbind(final_results_df, data.frame(
  Station = station,
  Dataset = dataset_name,
  Type = "GP",
  Model = "Stationary",
  Scale_Intercept = as.numeric(stat_fit_gp_summary$par["scale"]),
  Scale_SE_Intercept = as.numeric(stat_fit_gp_summary$se["scale"]),
  Scale_Slope = NA,
  Scale_SE_Slope = NA,
  Shape = as.numeric(stat_fit_gp_summary$par["shape"]),
  Shape_SE = as.numeric(stat_fit_gp_summary$se["shape"]),
  Trend_lr_pval = NA,
  Shape_lr_pval = NA,
  First_Observation_Date = first_obs_date,
  Last_Observation_Date = last_obs_date,
  Elapsed_Years = elapsed_years
))

final_results_df <- rbind(final_results_df, data.frame(
  Station = station,
  Dataset = dataset_name,
  Type = "Exponential",
  Model = "Stationary",
  Scale_Intercept = as.numeric(stat_fit_exp_summary$par["scale"]),
```

```
        Scale_SE_Intercept = as.numeric(stat_fit_exp_summary$se["scale"]),
        Scale_Slope = NA,
        Scale_SE_Slope = NA,
        Shape = as.numeric(stat_fit_exp_summary$par["shape"]),
        Shape_SE = as.numeric(stat_fit_exp_summary$se["shape"]),
        Trend_lr_pval = NA,
        Shape_lr_pval = NA,
        First_Observation_Date = first_obs_date,
        Last_Observation_Date = last_obs_date,
        Elapsed_Years = elapsed_years
      ))

      # Save results for non-stationary models
      final_results_df <- rbind(final_results_df, data.frame(
        Station = station,
        Dataset = dataset_name,
        Type = "GP",
        Model = "Multiplicative",
        Scale_Intercept = as.numeric(nonstat_fit_gp_summary$par["phi0"]),
        Scale_SE_Intercept = as.numeric(nonstat_fit_gp_summary$se["phi0"]),
        Scale_Slope = as.numeric(nonstat_fit_gp_summary$par["phi1"]),
        Scale_SE_Slope = as.numeric(nonstat_fit_gp_summary$se["phi1"]),
        Shape = as.numeric(nonstat_fit_gp_summary$par["shape"]),
        Shape_SE = as.numeric(nonstat_fit_gp_summary$se["shape"]),
        Trend_lr_pval = as.numeric(lr_trend_gp$p.value),
        Shape_lr_pval = NA,
        First_Observation_Date = first_obs_date,
        Last_Observation_Date = last_obs_date,
        Elapsed_Years = elapsed_years
      ))

        final_results_df <- rbind(final_results_df, data.frame(
        Station = station,
        Dataset = dataset_name,
        Type = "Exponential",
        Model = "Multiplicative",
        Scale_Intercept = as.numeric(nonstat_fit_exp_summary$par["phi0"]),
        Scale_SE_Intercept = as.numeric(nonstat_fit_exp_summary$se["phi0"])
            ,
        Scale_Slope = as.numeric(nonstat_fit_exp_summary$par["phi1"]),
        Scale_SE_Slope = as.numeric(nonstat_fit_exp_summary$se["phi1"]),
        Shape = as.numeric(nonstat_fit_exp_summary$par["shape"]),
        Shape_SE = as.numeric(nonstat_fit_exp_summary$se["shape"]),
        Trend_lr_pval = as.numeric(lr_trend_exp$p.value),
        Shape_lr_pval = as.numeric(lr_shape$p.value),
        First_Observation_Date = first_obs_date,
        Last_Observation_Date = last_obs_date,
        Elapsed_Years = elapsed_years
      ))
    }
}


# Check p-values
final_results_df <- final_results_df %>%
    mutate(Significant_Trend = ifelse(Model %in% c("Multiplicative") & !is.
        na(Trend_lr_pval), Trend_lr_pval < 0.05, NA))

final_results_df <- final_results_df %>%
  mutate(Significant_Shape = ifelse(Model %in% c("Multiplicative") & !is.na
      (Shape_lr_pval), Shape_lr_pval < 0.05, NA))
```

```
print("Done!")


# Save the results to a CSV file
library(readr)
write_csv(final_results_df, "GPresults2.csv")
```

PPfit_v2.R

```
# Olof
# 2025
library(extRemes)
library(dplyr)
library(lubridate)
library(NHPoisson)

# List of datasets to loop over
dataset_files <- list(
  "Original" = "final_filtered_station_data.csv",
  "Aligned" = "aligned_final_filtered_station_data.csv"
  #"Original" = "stations_5stations_debug.csv",
  #"Aligned" = "aligned_stations_5stations_debug.csv"
)

# Initialize results dataframe
results_df <- data.frame(
  Station = character(),
  Dataset = character(),
  b0_Estimate = numeric(),
  b0_SE = numeric(),
  b1_Estimate = numeric(),
  b1_SE = numeric(),
  lr_pval = numeric(),
  First_Observation_Date = as.Date(character()),
  Last_Observation_Date = as.Date(character()),
  Elapsed_Days = numeric(),
  stringsAsFactors = FALSE
)

# Loop over each dataset
for (dataset_name in names(dataset_files)) {
  file_path <- dataset_files[[dataset_name]]
  data <- read.csv(file_path)
  colnames(data) <- c("From_Date", "To_Date", "Date", "Rainfall_mm", "
      Quality", "Station_name", "Station_ID")

  # Get unique station names
  station_name_list <- unique(data$Station_name)

  # Loop over each station
  for (station in station_name_list) {
    # Filter for current station
    station_data <- data %>%
      filter(Station_name == station) %>%
      mutate(Date = as.Date(Date),
             Days_Elapsed = as.numeric(Date - min(Date, na.rm = TRUE)))

    # Calculate first and last observation dates
    first_obs_date = min(station_data$Date, na.rm = TRUE)
```

```
    last_obs_date = max(station_data$Date, na.rm = TRUE)

    # Calculate elapsed years based on these new dates
    elapsed_days <- as.numeric(difftime(last_obs_date, first_obs_date,
        units = "days"))

    # Define a threshold (e.g., 99.5th percentile)
    threshold <- quantile(station_data$Rainfall_mm, 0.995, na.rm = TRUE)

    # Declustering
    station_data_declus <- c(decluster(station_data$Rainfall_mm, threshold,
         replace.with = threshold - 1))

    # Fit  inhom Poisson Process model
    modelfit <- fitPP.fun(
      covariates = cbind(station_data$Days_Elapsed),
      POTob = list(T = station_data_declus, thres = threshold),
      start = list(b0 = 0, b1 = 0),
      dplot = FALSE,
      modSim = TRUE
    )

    # Extract summary
    PP_summary <- summary(modelfit)

    # Likelihood-ratio test
    lr <- LRTpv.fun(modelfit)

    # Extract estimates and standard errors
    b0_Estimate <- PP_summary@coef[1]
    b1_Estimate <- PP_summary@coef[2]
    b0_SE <- PP_summary@coef[3]
    b1_SE <- PP_summary@coef[4]


    # Save results
    results_df <- rbind(results_df, data.frame(
      Station = station,
      Dataset = dataset_name,
      b0_Estimate = b0_Estimate,
      b0_SE = b0_SE,
      b1_Estimate = b1_Estimate,
      b1_SE = b1_SE,
      lr_pval = as.numeric(lr),
      First_Observation_Date = first_obs_date,
      Last_Observation_Date = last_obs_date,
      Elapsed_Days = elapsed_days
    ))
  }
}

# Check p-vals
results_df <- results_df %>%
  mutate(Significant = lr_pval < 0.05)

print("pling")

# Save the results to a CSV file
library(readr)
write_csv(results_df, "PPresults2.csv")
```

GEVfit_QQ.R

```
    # Olof
    # 2025
library(extRemes)
library(dplyr)
library(lubridate)

# Read data
data <- read.csv("final_filtered_station_data.csv")
colnames(data) <- c("From_Date", "To_Date", "Date", "Rainfall_mm", "Quality
    ", "Station_name", "Station_ID")

# Choose station
station_data <- data %>%
  filter(Station_name == "Gendalen")

# Get the "water years"
station_data <- station_data %>%
  mutate(Date = as.Date(Date),
         Water_Year = if_else(month(Date) >= 10, year(Date) + 1, year(Date)
             )) %>%
  group_by(Station_ID) %>%
  arrange(Date) %>%
  mutate(
    First_Date = first(Date),
    Last_Date = last(Date),

    First_Oct1 = if_else(
      month(First_Date) == 10 & day(First_Date) == 1,
      First_Date,
      make_date(year(First_Date) + if_else(month(First_Date) >= 10, 1, 0),
          10, 1)
    ),
    Last_Water_Year = max(Water_Year),
    Last_Sep30 = make_date(Last_Water_Year - 1, 9, 30)
  ) %>%
  filter(Date >= First_Oct1 & Date <= Last_Sep30) %>%
  select(-First_Date, -Last_Date, -First_Oct1, -Last_Water_Year, -Last_
      Sep30) %>%
  ungroup()

# Filter for station and format Date column
station_data <- station_data %>%
  mutate(Date = as.Date(Date),
         Days_Elapsed = as.numeric(Date - min(Date, na.rm = TRUE)),
         Year = year(Date),
         Years_Elapsed = as.numeric(difftime(Date, min(Date, na.rm = TRUE),
              units = "days")) / 365.25)

# Compute the annual maximum (water years) rainfall and keep track of
    corresponding indices
rain_ann_max_with_indices <- station_data %>%
  group_by(Water_Year) %>%
  filter(Rainfall_mm == max(Rainfall_mm, na.rm = TRUE)) %>%
  slice(1) %>%
  ungroup() %>%
  select(Max_Rainfall = Rainfall_mm, Years_Elapsed)

# Create the data frame with the correct indices
data_df <- data.frame(ann_max = rain_ann_max_with_indices$Max_Rainfall,
```

```
                    years = rain_ann_max_with_indices$Years_Elapsed)

# Fit non-stat GEV
fit1 <- fevd(x = data_df$ann_max, data = data_df, type = "GEV", scale.fun =
    ~years, location.fun = ~years, use.phi = TRUE, time.units = "years")
summary_test <- summary(fit1)

# Get standardized quantiles
n <- length(data_df$ann_max)
sigma_t <- exp(as.numeric(fit1$results$par["phi0"]) + as.numeric(fit1$
    results$par["phi1"]) * (1:n))
mu_t <- as.numeric(fit1$results$par["mu0"]) + as.numeric(fit1$results$par["
    mu1"]) * (1:n)
y_t_k <- (data_df$ann_max - mu_t) / sigma_t

# Theoretical quantiles
QQ_y <- -log(-log(1:n / (n + 1)))

plot(QQ_y, sort(y_t_k),
     xlab = "Teoretiska␣GEV-kvantiler",
     ylab = "Observerade␣kvantiler␣(standardiserade)")
abline(0, 1, col = "red", lwd = 2)
```

GPfit_QQ.R

```
    # Olof
    # 2025
library(extRemes)
library(dplyr)
library(lubridate)

# Read data
data <- read.csv("final_filtered_station_data.csv")
colnames(data) <- c("From_Date", "To_Date", "Date", "Rainfall_mm", "Quality
    ", "Station_name", "Station_ID")

# Filter for station
station_data <- data %>%
  filter(Station_name == "Gendalen") %>%
  mutate(Date = as.Date(Date),
         Days_Elapsed = as.numeric(Date - min(Date, na.rm = TRUE)),
         Year = year(Date),
         Years_Elapsed = as.numeric(difftime(Date, min(Date, na.rm = TRUE),
              units = "days")) / 365.25)  # Convert days to years)

# Define a threshold (e.g., 995th percentile)
threshold <- quantile(station_data$Rainfall_mm, 0.995, na.rm = TRUE)

# Declustering
station_data_declus <-  c(decluster(station_data$Rainfall_mm, threshold,
    replace.with=threshold-1))

GP_df_year_elaps <- data.frame(rain_data = station_data_declus, years =
    station_data$Years_Elapsed)

# Fit non-stat GP
fit1 <- fevd(x = station_data_declus,
             data = GP_df_year_elaps,
             threshold = threshold,
             scale.fun = ~years,
             type = "GP",
```

```
              use.phi = TRUE,
              span = station_data$Years_Elapsed[length(station_data$Year)],
              time.units = "years")

summary(fit1)

years <- station_data$Years_Elapsed

# Get standardized quantiles
excesses <- station_data$Rainfall_mm[station_data$Rainfall_mm > threshold]
n <- length(excesses)
sigma_t <- exp(as.numeric(fit1$results$par["phi0"]) + as.numeric(fit1$
    results$par["phi1"]) * (years))
sigma_t_excesses <- sigma_t[station_data$Rainfall_mm > threshold]
y_t_k <- (excesses - threshold) / sigma_t_excesses

# Get theoretical quantiles
theoretical <- -log(1 - 1:n / (n + 1))

plot(theoretical, sort(y_t_k),
     xlab = "Teoretiska␣GP-kvantiler",
     ylab = "Observerade␣kvantiler␣(standardiserade)")
abline(0, 1, col = "red", lwd = 2)
```

pval_plots.R

```
    # Olof
    # 2025
library(dplyr)

# Read in desired results
df <- read.csv("GPresults2.csv")

df <- read.csv("GEVresults2.csv")

df <- read.csv("PPresults2.csv")

# Get the right rows
df_filtered <- df %>%
  filter(Dataset == "Original") %>%
  filter(Model == "Multiplicative␣(mu/sigma)") #%>%
  #filter(Type == "GEV")

df_filtered <- df %>%
  filter(Dataset == "Original") %>%
  filter(Model == "Multiplicative") %>%
  filter(Type == "GP")

df_filtered <- df %>%
  filter(Dataset == "Aligned")

station_name_list <- unique(df_filtered$Station)

# Get the expected and observed p-values
x <- 1:length(station_name_list)/length(station_name_list)
p_vals <- sort(df_filtered$lr_pval) # Change column name accordingly

# Plot
plot(x, p_vals,
     xlab = "F rv ntade␣p-v rden",
     ylab = "Observerade␣p-v rden"
```

```
     #main = " Q Q -plott av p-v rden fr n likelihoodkvottest"
     )
abline(0, 1, col = "red", lwd = 2)
```

histogram_lambda1.R

```
    # Olof
    # 2025
library(dplyr)

df <- read.csv("PPresults2.csv")

df_filtered <- df %>%
  filter(Dataset == "Original") #%>%
  #filter(Model == "Multiplicative") %>%
  #filter(Type == "GP")


#expected_exceedances <- function(b0, b1, x_days) {
#  (exp(b0) / b1) * (exp(b1 * x_days) - 1)
#}

station_name_list <- unique(df_filtered$Station)
lambda1s <- rep(0, length(station_name_list))
i <- 1

for (station in station_name_list) {
  station_data <- df_filtered %>% filter(Station == station)
  b0 <- station_data$b0_Estimate
  b1 <- station_data$b1_Estimate

  lambda1s[i] <- b1 * 365.25
  i <- i + 1
}

# Breaks for the bins
bin_edges_lambda1_rounded <- c(-0.006, 0.000, 0.0035, 0.007, 0.0105, 0.014)

# Color of the bars
colors <- c(
  rgb(  5, 113, 176, maxColorValue = 255 ),
  rgb(222, 235, 241, maxColorValue = 255 ),
  rgb(245, 190, 165, maxColorValue = 255 ),
  rgb(228, 101,  92, maxColorValue = 255 ),
  rgb(202,   0,  32, maxColorValue = 255 )
)

# Compute histogram
h <- hist(lambda1s, breaks = bin_edges_lambda1_rounded, plot = FALSE, right
    = FALSE)

# Compute bar midpoints
mids <- h$mids
heights <- h$counts
widths <- diff(bin_edges_lambda1_rounded)

# Start blank plot
plot(0, 0, type = "n",
     xlim = range(bin_edges_lambda1_rounded),
     ylim = c(0, max(heights)),
     xlab = expression(lambda[1]),
```

```
     ylab = "Antal",
     xaxt = "n")

# Add x-axis manually
axis(side = 1, at = bin_edges_lambda1_rounded, labels = format(bin_edges_
    lambda1_rounded, digits = 3))

# Draw bars with colored rectangles
for (i in seq_along(heights)) {
  rect(xleft  = bin_edges_lambda1_rounded[i],
       xright = bin_edges_lambda1_rounded[i + 1],
       ybottom = 0,
       ytop = heights[i],
       col = colors[i],
       border = "black")
}
```