KU LEUVEN

FACULTEIT INGENIEURSWETENSCHAPPEN

Multilevel Monte Carlo voor de Analyse van de Betrouwbaarheid van Systemen

Michiel Vromans

Thesis voorgedragen tot het behalen van de graad van Master of Science in de ingenieurswetenschappen: wiskundige ingenieurstechnieken

Promotoren:

Prof. dr. ir. Stefan Vandewalle
Prof. dr. ir. David Moens

Assessoren:

Prof. Nigel Smart
Prof. dr. Ir. Dirk Nuyens

Begeleider:

Ir. Pieterjan Robbe

Academiejaar 2018 – 2019

Voorwoord

Hier wil ik graag iedereen danken die een aandeel heeft in het afronden van de masterproef. Ten eerste wil ik mijn promotor, prof. dr. ir. S. Vandewalle, bedanken voor het aanreiken van dit interessante onderwerp en de mogelijkheid om mij hierin te verdiepen. Verder wil ik mijn begeleider, ir. Pieterjan Robbe, bedanken voor de steun en wekelijkse feedback. Dit zorgde voor de nodige inzichten en kennis om de thesis tot een succesvol einde te brengen.
Verder wil ik mijn familie en vrienden vermelden die gezorgd hebben voor de nodige rust, het vertrouwen en mij gesteund hebben doorheen dit onderzoek.

Michiel Vromans

Inhoudsopgave

Samenvatting

Een efficiënte simulatie van de gemiddelde tijd tot aan falen van systemen wordt steeds belangrijker als gevolg van de vraag naar meer nauwkeurige risicoanalyses, een groeiende complexiteit van systemen, etc. De meest algemeen toepasbare techniek is Monte Carlo waarbij er gebruik gemaakt wordt van de snedes van het systeem. De rekenkost van deze techniek heeft een complexiteit $\mathcal{O}(\varepsilon^{-2})$, wat voor kleine nauwkeurigheden zeer hoog is. Mogelijke manieren om de rekenkost te verlagen zijn bijvoorbeeld het toepassen van de methode van de controlevariabelen of door gebruik te maken van Quasi-Monte Carlo.
Deze masterproef bestudeert een andere methode die mogelijk gecombineerd kan worden met deze technieken: Multilevel Monte Carlo. Het toepassen van Multilevel Monte Carlo voor de bepaling van de gemiddelde tijd tot aan falen is gebaseerd op het werk in [2]. Het algoritme bouwt een hiërarchie op van deelverzamelingen van de verzameling van snedes met een groeiende kost en een dalende variantie in ieder level. Door de levensduur op de verschillende levels te combineren wordt een berekening bekomen met dezelfde nauwkeurigheid maar met een lagere rekenkost. De resultaten in deze masterproef tonen een verschil van enkele grootteordes tussen de rekenkost van Multilevel Monte Carlo en standaard Monte Carlo. De complexiteit van beide methoden is $\mathcal{O}(\varepsilon^{-2})$.
In dit werk wordt het algoritme uitgebreid tot Multilevel Quasi-Monte Carlo. Het Multilevel Quasi-Monte Carlo algoritme wordt uitgevoerd met deterministische, goed gekozen punten zodat deze meer uniform verdeeld zijn. Door deze keuze wordt er in de resultaten een lagere rekenkost waargenomen voor het bepalen van de gemiddelde tijd tot aan falen van systemen dan in Multilevel Monte Carlo. Afhankelijk van de grootte van de dimensies en de variantie van de ingangsvariabelen wordt er een complexiteit $\mathcal{O}(\varepsilon^{-p})$ met $p < 2$ waargenomen. Bij zeer hoge dimensies (50 componenten) is $p \approx 2$, bij lagere dimensies (20 componenten) wordt er $p \approx 1.4$ waargenomen.
Het resultaat van deze masterproef is dat Multilevel Monte Carlo een lagere rekenkost heeft dan Monte Carlo. De asymptotische complexiteit van beide methoden is dezelfde, er is enkel een verlaging van de rekenkost met een constante factor. Door gebruik te maken van de combinatie Multilevel Monte Carlo en Quasi-Monte Carlo is er bijkomend een verlaging van de complexiteit naar $\mathcal{O}(\varepsilon^{-p})$ met $p < 2$.

Lijst van figuren

Lijst van afkortingen en symbolen

α	parameter van de afname van de correctieterm per level in de ML(Q)MC berekening
β	parameter van de afname van de variantie van de correctieterm per level in de ML(Q)MC berekening
γ	parameter van de toename van de kost per level in de ML(Q)MC berekening
ε	vooropgestelde nauwkeurigheid
$\#C$	het aantal snedes in de verzameling
bias	afwijking
$\mathcal{C}$	Verzameling van minimale snedes
$\mathbb{E}$	Verwachte waarde
FFT	Fast Fourier Transform
i.i.d.	onafhankelijke en gelijke verdelingen (independent and identical distributions)
k,λ	vorm- en schaalparameter voor de kansverdelingen
MC	Monte Carlo
MLMC	Multilevel Monte Carlo
MLQMC	Multilevel Quasi-Monte Carlo
MOCUS	Method for Obtaining Cut Sets
MSE	Mean Square Error of gemiddelde kwadratische kost
P(X)	De kans dat X voorkomt
QMC	Quasi-Monte Carlo
RMSE	Root Mean Square Error of kwadraat van de gemiddelde kwadratische fout
s	seconden
t	tijd
T,$\hat{T}$	Levensduur, berekende levensduur
T_S	Levensduur van systeem S
$\mathbb{V}$	Variantie
V	Berekende variantie

Hoofdstuk 1

Inleiding

Het uitvoeren van numerieke berekeningen, simulaties van processen en risicoanalyses wordt steeds belangrijker bij het ontwerpen van producten, plannen van onderhoud en het onderzoeken van het falen van systemen. In al deze berekeningen moet er rekening gehouden worden met onzekerheden in de werking en met stochastische factoren. Enkele directe toepassingen zijn het blijven werken van een smartphone, een batterij die blijft leven, een vliegtuig dat nooit crasht, een kerncentrale die verschillende natuurrampen moet doorstaan.
De wetenschap die zich toespitst op de betrouwbaarheid van systemen bestaande uit componenten met tussenliggende verbindingen is bekend als de betrouwbaarheidsanalyse. De doelstelling van deze analyse is het vermijden van het falen van een systeem. De analyse van de betrouwbaarheid van een systeem is een steeds belangrijker wordend onderdeel van het ontwerpproces. Het is mogelijk dat een analyse van de betrouwbaarheid moet gebeuren in iedere stap van de ontwerpfase. Bijkomend wil men verschillende constructies of opties gaan vergelijken en beoordelen, men gaat dus meerdere analyses moeten doen van het systeem [2]. Deze berekeningen zijn hoog-dimensionale problemen waardoor de rekenkost van de klassieke methoden zeer hoog kan zijn. Dit betekent dat er vraag is naar efficiënte methodes voor de berekening van de betrouwbaarheid.
Het onderdeel van de betrouwbaarheidsanalyse waar deze thesis zich op focust, is de berekening van de gemiddelde tijd tot aan falen van een systeem waarvoor de verdeling van de levensduur van de componenten gekend is. Hierbij wordt er gebruik gemaakt van de verzameling van de snedes van een systeem. Zo een snede is een verzameling van componenten waarbij het systeem faalt als deze verzameling faalt. Een systeem wordt verondersteld werkende te zijn als er een pad van ingang naar uitgang bestaat met enkel werkende componenten, met andere woorden er is geen snede die de ingang en uitgang afscheidt.
Een van de meest toepasbare methoden voor het bepalen van de gemiddelde tijd tot aan falen is de Monte Carlo methode. De rekenkost van de Monte Carlo methode wordt bepaald door de variantie van de levensduur en de opgegeven nauwkeurigheid maar ook door de grootte van de systemen. Systemen met meer componenten hebben snedes met meer componenten of meer snedes in de verzameling waardoor de

evaluatie van de levensduur van het systeem in een simulatie duurder wordt. Door de groei van systemen, wordt het numeriek bepalen van de betrouwbaarheid steeds moeilijker en stijgt de rekenkost in het slechtste geval exponentieel. Hierdoor wordt een standaard Monte Carlo methode te duur. De verlaging van de rekenkost bij Monte Carlo kan gebeuren door de variantie te reduceren door gebruik te maken van bijvoorbeeld de methode van controlevariabelen, of door gebruik te maken van andere punten (Quasi-Monte Carlo) [7].
Recent zijn er ontwikkelingen om de Monte Carlo methode te versnellen door gebruik te maken van Multilevel Monte Carlo (MLMC). In de context van partiële differentiaalvergelijkingen met willekeurige coëfficienten is er reeds aangetoond dat door Multilevel Monte Carlo te gebruiken er een verhoging van de efficiëntie is [10]. Ook is er een eerste introductie in de betrouwbaarheidsanalyse van systemen gebeurd door L.J.M. Aslett [2]. Multilevel Monte Carlo maakt hierbij gebruik van een hiërarchie van verzamelingen van snedes. In deze thesis wordt er verder gebouwd op deze introductie waarbij Multilevel Monte Carlo wordt toegepast op arbitraire systemen en enkele reële toepassingen. Verder wordt de extensie van Multilevel Monte Carlo naar Multilevel Quasi-Monte Carlo (MLQMC) bekeken waarbij er geen willekeurige punten genomen worden maar deterministische goed gekozen punten om de rekenkost verder te drukken.
In hoofdstuk 2 wordt de betrouwbaarheidsanalyse geïntroduceerd. Hierin worden de systemen besproken en de constructie van de verzameling van de snedes. Hoofdstuk 3 bespreekt de Monte Carlo methode en Multilevel Monte Carlo. In dit hoofdstuk wordt ook Multilevel Quasi-Monte Carlo geïntroduceerd. De opbouw van de hiërarchie in de verzameling van snedes, de implementatie van de Multilevel Monte Carlo en de aanpassing naar Quasi-Monte Carlo wordt gegeven in Hoofdstuk 4. De numerieke resultaten van arbitraire systemen en toepassingen worden gegeven in hoofdstuk 5. Hoofdstuk 6 is het besluit van de masterproef.

Hoofdstuk 2

Betrouwbaarheidsanalyse

Met betrouwbaarheid van een component of systeem bedoelt men het vermogen van een component of systeem om een vereiste functie uit te voeren in een bepaalde omgeving en onder bepaalde omstandigheden over een bepaalde tijd. In de context van deze thesis bekijken we enkel de levensduur van een systeem, de gemiddelde tijd tot aan falen. De gemiddelde tijd tot aan falen is de verwachte tijd totdat het systeem een bepaalde functie niet meer kan uitvoeren (BS4778 of IEC50(191))[6]. Dit kan echter uitgebreid worden naar een functionaal van de levensduur [21].
De betrouwbaarheidsanalyse van systemen koppelt het falen van een systeem aan het falen van de componenten. Het doel van deze wetenschap is de verhoging van de betrouwbaarheid van systemen door bijvoorbeeld componenten met een korte levensduur te vervangen door componenten met een langere levensduur, componenten met meer vermogen over een langere tijd. Een andere manier is tijdig onderhoud uit te voeren op de zwakke punten om zo het systeem langer te kunnen gebruiken.
Tegenwoordig bestaat deze wetenschapstak uit het onderzoeken van de grens tussen een operationeel systeem en een systeem dat faalt. Hierbij zijn de volgende zaken de kern van het onderzoek:

- gebruik maken van betrouwbaarheidsanalyse om oorzaken van systeemfalen te ontdekken,
- het meest betrouwbare systeem ontwerpen voor de vereisten,
- het testen en meten van de betrouwbaarheid via simulaties,
- het onderhouden van een systeem via foutdetectie en prognose.

De bepaling van de gemiddelde tijd tot aan falen is gebaseerd op een kwantitatieve definitie van betrouwbaarheid in de waarschijnlijkheidstheorie: beschouw de continue variabele *levensduur* T, de betrouwbaarheid op tijdstip t is de waarschijnlijkheid dat het systeem niet faalt, de waarschijnlijkheid dat T is groter dan t

$$\text{betrouwbaarheid} = P(T > t) \tag{2.1}$$

Een andere grootheid waarin men geïnteresseerd is, is de beschikbaarheid van een systeem. De beschikbaarheid van een systeem wordt gegeven door het vermogen

van het systeem om een taak uit te voeren. Hierbij richt men zich op herstelbare systemen waarbij de componenten hersteld kunnen worden na een defect zonder dat het systeem volledig uitvalt. Onder herstelling verstaat men het vervangen of repareren van defecte componenten, preventief onderhoud (gebaseerd op inspectie), conditioneel onderhoud (gebaseerd op verminderd vermogen) en predictief onderhoud (gebaseerd op de evolutie van het vermogen) [29].
De thesis behandelt twee soorten systemen, systemen met herstelbare en systemen met niet-herstelbare componenten. Bij een niet-herstelbaar component wordt er enkel gekeken naar de levensduur totdat het component voor de eerste keer faalt. Het doel hierbij is een efficiënte berekening van de levensduur waarbij er rekening gehouden wordt met een vooropgestelde nauwkeurigheid.
Herstelbare systemen zijn interessanter doordat er de mogelijkheid is om de levensduur te verlengen via herstellingen. Als de herstelling gebeurt voordat het systeem in zijn geheel faalt, dan stijgt de betrouwbaarheid van het totale systeem en wordt de levensduur verlengd [21].
De context van dit onderzoek is systemen die bestaan uit componenten waarvan de verdeling van de levensduur bekend is. De gemiddelde tijd tot aan falen van zo een systeem is afhankelijk van de levensduur van deze componenten (Sectie 2.1) en van de manier hoe deze componenten met elkaar verbonden zijn (Sectie 2.2).

2.1 Componenten

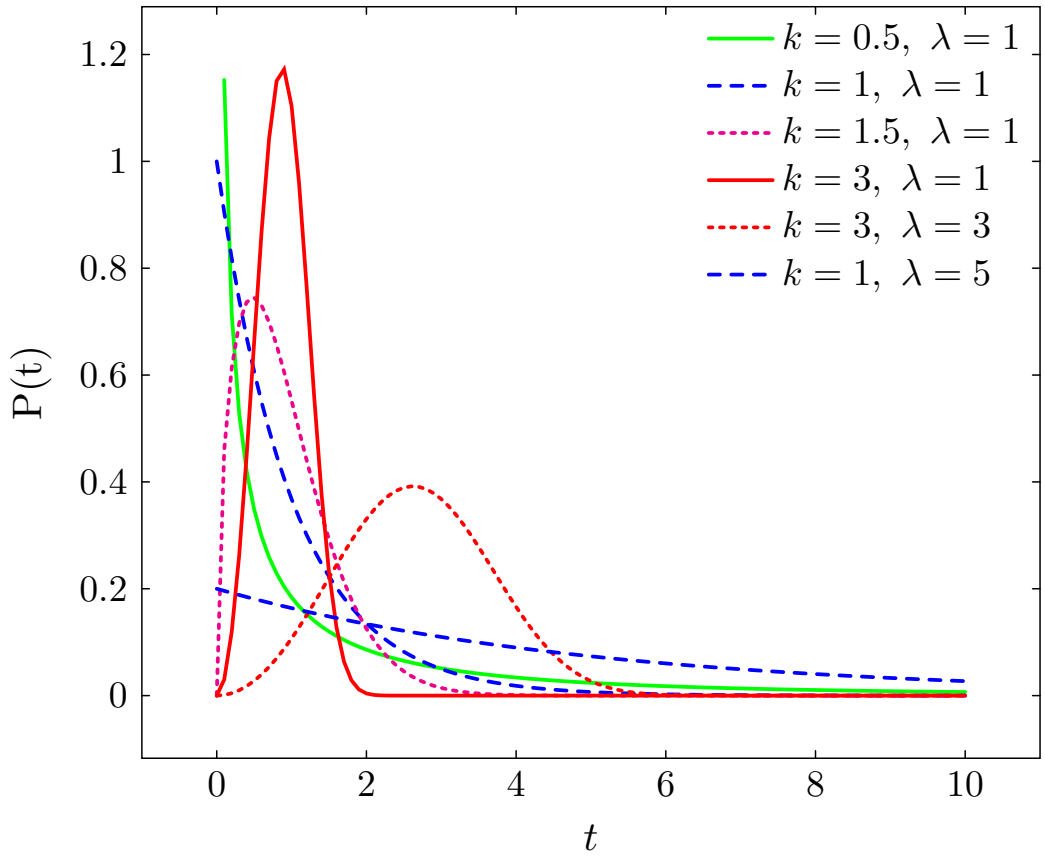
De bouwblokken van een systeem zijn de componenten. Deze componenten worden in deze thesis als onafhankelijk beschouwd en hebben een levensduur die bepaald wordt door een gekende onderliggende verdeling. De kansverdeling van processen als levensduur, fabricagetijd en levertijden worden meestal voorgesteld door middel van een Weibull-verdeling (zie figuur 2.1). Deze verdeling wordt bepaald door 2 parameters: de vormparameter $k > 0$ en de schaalparameter $\lambda > 0$. De kansdichtheid voor $x \geq 0$ wordt gegeven door

$$f(x; \lambda, k) = \frac{k}{\lambda} \left(\frac{x}{\lambda} \right)^{k-1} e^{-(x/\lambda)^k} \tag{2.2}$$

De vormparameter geeft een beeld van de uitvalsnelheid van het component en kan als volgt worden geïnterpreteerd:

- $k < 1$: de uitvalsnelheid daalt met de werkingsduur. Een voorbeeld hiervan zijn systemen waarbij kinderziektes aanwezig zijn die snel worden opgelost. Dit hangt samen met het Lindy effect, het fenomeen dat de verwachte levensduur proportioneel is met de leeftijd van het component. Hierbij impliceert elke periode van overleving een langere levensverwachting.

- k=1 : uitvalsnelheid is constant over de tijd. De voornaamste redenen van een defect zijn storingen door willekeurige externe gebeurtenissen.



Figuur 2.1: Weibull-verdeling voor verschillende vorm- en schaalparameters

- k>1 : uitvalsnelheid wordt groter over de tijd. De componenten ondervinden storingen veroorzaakt door veroudering. Naargelang een component ouder wordt of langer in werking is, is de kans op falen groter.

In deze masterproef wordt er in de arbitraire systemen gebruik gemaakt van de Weibull-verdeling, in een uitbreiding wordt er gekeken naar het invoeren van andere verdelingen die voorkomen in de realiteit, o.a. exponentiële verdeling, uniforme verdeling.

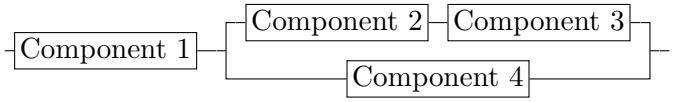
2.2 Structuur

De systemen die onderzocht worden zijn vaak uniek en hierdoor is het niet mogelijk of niet praktisch om een statistische foutenanalyse uit te voeren. De methode om dit aan te pakken is de werking van het systeem relateren aan werking van de componenten via logische relaties (EN-, OF-poorten). Deze logische structuur is dan representatief voor de functie en kan gebruikt worden voor de analyse van de betrouwbaarheid. Een mogelijke manier om de logische structuur weer te geven, is door gebruik te

maken van een betrouwbaarheidsblokdiagram (reliability block diagram). Dit is een grafische voorstelling van het systeem die het netwerk van componenten weergeeft tussen de ingang en de uitgang. De structuur kan ook weergegeven worden met een structuurfunctie; dit is een meer wiskundige voorstelling. Een andere manier is de foutenboom; deze techniek laat toe om de oorzaken voor een ongewenste gebeurtenis te achterhalen. De duale analyse van de foutenboom is de gebeurtenissenboom (event tree). In de volgende secties worden deze methoden verder in detail besproken waarbij de definitie, het verband tussen de methoden en de manier waarop deze gebruikt worden in de masterproef duidelijk worden.

2.2.1 Betrouwbaarheidsblokdiagram

De manier om de systemen in deze masterproef voor te stellen is door gebruik te maken van een betrouwbaarheidsblokdiagram. Dit is een grafische voorstelling van het systeem die de componenten en de verbindingen tussen de componenten weergeeft. Hieruit kan de structuurfunctie worden afgeleid, net als het verband tussen de ingang en de uitgang. Een voorbeeld van een systeem in zo een diagram is weergegeven in figuur 2.2.



Figuur 2.2: Voorbeeld van een betrouwbaarheidsblokdiagram

2.2.2 Structuurfunctie

Een systeem kan wiskundig worden voorgesteld door een structuurfunctie. Hiervoor wordt de toestandsvector $\mathbf{x} = (x_1, x_2, ..., x_n)$ van de componenten gedefinieerd waarbij de toestandsvariabele x_i de toestand is van component i (1 betekent werkend, 0 betekent niet-werkend). De structuurfunctie van het systeem is dan gegeven door

$$\phi(\mathbf{x}) = f(x_1, x_2, ..., x_n) = \begin{cases} 1 & \text{systeem werkt} \\ 0 & \text{systeem is gefaald} \end{cases} \tag{2.3}$$

Als een systeem werkt enkel en alleen als alle componenten werken, dan is de logische structuur serieel. De structuurfunctie voor een serieel systeem wordt gegeven door

$$\phi(\mathbf{x}) = x_1 \cdot x_2 \cdots x_n = \prod_{i=1}^{n} x_i \tag{2.4}$$

Als een systeem werkt als minstens 1 van de componenten werkt dan is de logische structuur parallel. De structuuurformule voor parallele systemen is

$$\phi(\mathbf{x}) = 1 - (1 - x_1) \cdot (1 - x_2) \cdots (1 - x_n) = 1 - \prod_{i=1}^{n} (1 - x_i) \tag{2.5}$$

Meer complexe systemen kunnen gecreëerd worden door de seriële en parallelle structuren te combineren. Voor kleine systemen kan de structuurfunctie uitgeschreven worden door visueel het systeem te bestuderen als een combinatie van serie- en parallelschakelingen. Voor grotere systemen wordt dit echter zeer moeilijk tot onmogelijk. Voor deze systemen doen we beroep op methoden voor het bepalen van paden of van snedes van systemen.
Een pad P is een verzameling componenten waarvoor het volgende geldt: als deze componenten werken, dan werkt het systeem. Een pad is minimaal als er geen componenten kunnen weggelaten worden zonder de betekenis van pad te verliezen. Voor ieder minimaal pad P_j is er een structuurfunctie zoals een serieel systeem:

$$\rho_j(\mathbf{x}) = \prod_{i \in P_j}^{n} x_i \tag{2.6}$$

De structuurformule van het systeem kan vervolgens geschreven worden als

$$\phi(\mathbf{x}) = \prod_{i=j}^{n} (1 - \rho_j) \tag{2.7}$$

Dit is gelijk aan een parallelschakeling van de paden. Een snede K is een verzameling van componenten waarvoor het systeem faalt als deze componenten falen. Een snede is minimaal als er geen componenten kunnen weggelaten worden zonder de betekenis van snede te verliezen. Voor iedere minimale snede K_j is er een structuurfunctie zoals een parallel systeem:

$$\kappa_j(\mathbf{x}) = 1 - \prod_{i \in K_j}^{n} (1 - x_i) \tag{2.8}$$

De structuurformule van het systeem kan vervolgens geschreven worden als

$$\phi(\mathbf{x}) = \prod_{i=j}^{n} \kappa_j(\mathbf{x}) \tag{2.9}$$

Het systeem kan bijgevolg geschreven worden als een serieschakeling van de snedes. De definities worden geïllustreerd aan de hand van het voorbeeld gegeven in figuur 2.2. Stel een systeem met 4 componenten: Component 1 (C1), Component 2 (C2), Component 3 (C3) en Component 4 (C4), met een structuur zoals in de figuur. Dit systeem heeft de volgende verzamelingen:

- Paden: {C1,C2,C3}, {C1, C4}, {C1,C2,C4}, {C1,C2,C4}, {C1, C2, C3,C4}
- Minimale paden: {C1,C2,C3}, {C1, C4}
- Snedes: {C1}, {C2, C4}, {C3,C4}, {C1, C2, C4}, {C1, C3, C4} , {C1,C2,C3}, {C1, C2, C3, C4}
- Minimale snedes: {C1}, {C2, C4},{C3, C4}

2.2.3 Foutenboom

De foutenboom is een logische voorstelling van de causale relaties voor een systeemfalen door het falen te analyseren in zijn mogelijke oorzaken. Om zo een foutenboom op te stellen, is het belangrijk om de structuur van het systeem te begrijpen. Dit kan gebeuren via een betrouwbaarheidsblokdiagram. Het opstellen van de foutenboom begint bij het selecteren van het te onderzoeken defect (top-event). De volgende stap is het bepalen van de gebeurtenissen die het top-event als direct gevolg kunnen hebben. Er zijn 4 mogelijkheden [21]:

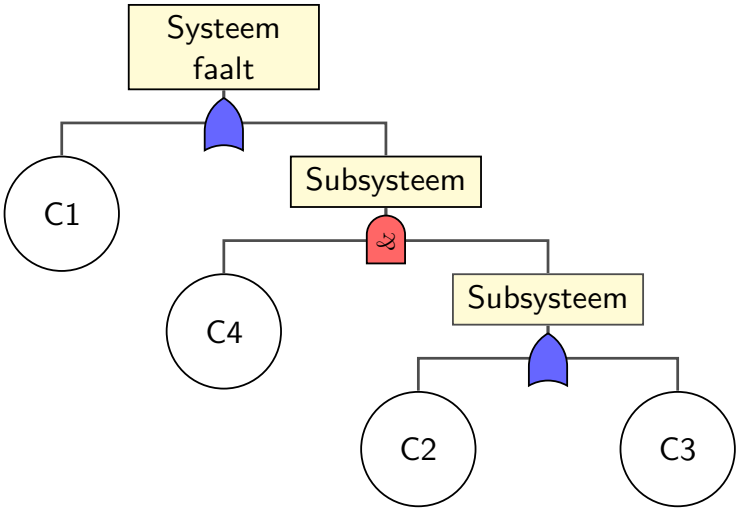
1. Het systeem heeft geen input.
2. De gebeurtenis is een primair defect van het systeem (willekeurig, ouderdom, designfout).
3. Menselijke fout door een foute actie of een verkeerde instelling.
4. De gebeurtenis is een secundair defect (overbelasting door andere componenten, omgeving, etc.).

De verschillende gebeurtenissen die een oorzaak zijn van het falen worden gecombineerd door een logische poort (EN, OF,...). Als alle oorzaken van het top-event zijn geïdentificeerd, wordt elke oorzaak verder onderzocht. Als de oorzaak primair is wordt de tak beëindigd. Dit betekent dat deze oorzaak niet door andere gebeurtenissen veroorzaakt kan worden en dat er een waarschijnlijkheid van voorkomen kan worden toegekend aan deze gebeurtenis. Als de gebeurtenis geen primair event is, wordt deze verder ontbonden in meer elementaire oorzaken. Alle gebeurtenissen worden recursief geanalyseerd tot alle takken uitgewerkt zijn tot primaire gebeurtenissen, de bladeren, waarvan de waarschijnlijkheid van voorkomen gekend is.
Het systeem kan vervolgens worden geanalyseerd door de foutenboom te herschrijven als een booleaanse algebraïsche vergelijking. Via Booleaanse algebra is het mogelijk om het falen van het systeem te schrijven in functie van het falen van de componenten. Deze relatie kan dus gebruikt worden om de waarschijnlijkheid van voorkomen van het top-event te bepalen als de waarschijnlijkheid van voorkomen van primaire gebeurtenissen gekend is. De foutenboom van het eenvoudige voorbeeld (figuur 2.2) is gegeven in figuur 2.3. De algebraïsche vergelijking die hiermee overeenkomt is

$$C1|((C2|C3)\&C4).$$

2.2.4 Gebeurtenissenboom

De tegenhanger van de foutenboom is de gebeurtenissenboom. De gebeurtenissenboom is een inductieve manier voor het identificeren van het mogelijke falen van het systeem veroorzaakt door een gebeurtenis. Inductief wordt er dan een sequentie opgesteld van mogelijke gevolgen van een bepaalde gebeurtenis.



Figuur 2.3: Voorbeeld van een foutenboom

De constructie van de gebeurtenissenboom start met de primaire gebeurtenis, het falen van een component. Voor de gebeurtenis worden de directe gevolgen bepaald, voor elk direct gevolg ontstaat er een tak, die vervolgens uitgewerkt wordt tot de te onderzoeken gebeurtenis plaatsvindt. Elke gebeurtenis in de gebeurtenissenboom kan geïnterpreteerd worden als het top-event van een foutenboom, de waarschijnlijkheid van voorkomen van dit event is dan bepaald door de conditionele waarschijnlijkheid van het voorkomen van de gebeurtenissen voorafgaand in de sequentie. Voor iedere primaire gebeurtenis, falen van een component wordt er een gebeurtenissenboom uitgewerkt tot het falen van het systeem. Door deze bomen vervolgens te analyseren kan er een structuurformule van het systeem bepaald worden.

2.3 Methode voor het bepalen van de snedes

De berekening van de levensduur van een systeem wordt hier gebaseerd op de structuurfunctie van het systeem die gebaseerd is op de minimale snedes. Voor kleine of eenvoudige systemen kunnen de minimale snedes via visuele inspectie worden opgesteld. Voor grotere systemen wordt er een algoritme voorgesteld, het MOCUS algoritme (Method for Obtaining CUt Sets)[21].
Het algoritme start met het opstellen van de foutenboom van het systeem waarbij het top-event het falen van het systeem is en de boom uitgewerkt wordt tot de bladeren overeen komen met de componenten. Een serieschakeling in een systeem komt overeen met een OF-poort en een parallelschakeling met een EN-poort.
Vervolgens wordt de verzameling opgebouwd. Deze constructie start met een vector met één element of gebeurtenis, het top-event (“het systeem faalt”). De elementen worden nu telkens vervangen tot de componenten allemaal toegevoegd zijn aan de vector en er geen opdeling meer mogelijk is. De opbouw is gebaseerd op de poorten.

- Bij een OF-poort wordt elke ingang geschreven in een verschillende rij,
- bij een EN-poort wordt elke ingang geschreven in een verschillende kolom.

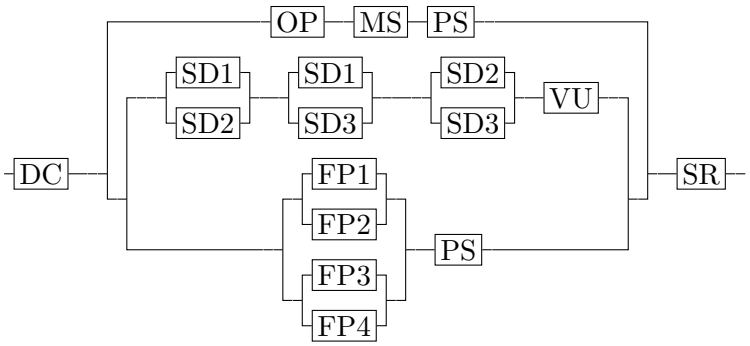
Als elke tak is doorlopen dan is elke rij van de verzameling een snede. De laatste stap van het algoritme is het minimaliseren van de verzameling van snedes zodat de verzameling enkel bestaat uit minimale snedes.

2.3.1 Toepassing: bepalen van de verzameling van minimale snedes van een brandalarm

Als voorbeeld passen we het algoritme toe op een model van een brandalarm uit [21]. Dit systeem wordt gevisualiseerd in figuur 2.4. Het detectiesysteem van het brandalarm is aangesloten op een voeding (DC) en gaat af wanneer het startrelais (SR) geactiveerd wordt. De activatie kan veroorzaakt worden door 3 deelsystemen, een rookmelder die een signaal stuurt als er rook aanwezig is, een pneumatisch systeem dat warmte detecteert of een actie door een persoon. De eigenschappen van deze deelsystemen zijn als volgt:

- De rookmelder bestaat uit 3 sensoren (SD1, SD2 en SD3). Deze meten de rook in de ruimte op verschillende plaatsen en sturen een signaal naar een 2-van-de-3 schakelaar (VU). De 2-van-de-3 schakelaar geeft een signaal wanneer 2 van de 3 sensoren een signaal sturen naar de schakelaar.
- De warmtedetectie gebeurt met een pneumatische systeem dat geïnstalleerd is in de ruimte. Dit systeem bestaat uit een pneumatische pijp met 4 identieke thermische zekeringen (FP). In het pneumatisch systeem is er een druk van 3 bar. Dit systeem is verbonden met een druksensor (PS). De zekeringen worden luchtdoorlatend bij een temperatuur hoger dan 72 ℃. Als een of meer zekeringen worden geactiveerd dan zal de druk verlagen. Deze drukval wordt gedetecteerd door de druksensor en die zal een signaal sturen naar het startrelais.
- De handmatige activatie gebeurt door de operator (OP), een persoon. deze detecteert de brand en activeert de manuele schakelaar (MS). Deze schakelaar is verbonden met het pneumatisch systeem van de warmtedetectie en zorgt voor hetzelfde effect als een thermische zekering. De drukval in de pijp zorgt dat de druksensor wordt geactiveerd en deze stuurt dan een signaal naar het startrelais.

De functie die hier beschouwd wordt is het afgaan van het alarm. Het MOCUS algoritme start met het opstellen van de foutenboom, zie figuur 2.4. Het top-event is dus het alarm gaat niet af. Er zijn 3 mogelijke oorzaken die ervoor zorgen dat het alarm niet kan werken. Dit is een defecte voeding (DC), het detectiesysteem stuurt geen signaal of het startrelais (SR) is defect. De voeding en het startrelais zijn elementaire componenten, bladeren van de foutenboom, de opbouw van deze takken stopt dus hier. Het detectiesysteem bestaat uit meerdere deelsystemen. Het falen van het detectiesysteem wordt veroorzaakt door het falen van de 3 volgende functies (EN-poort omdat de 3 functies moeten falen om het event te laten gebeuren).



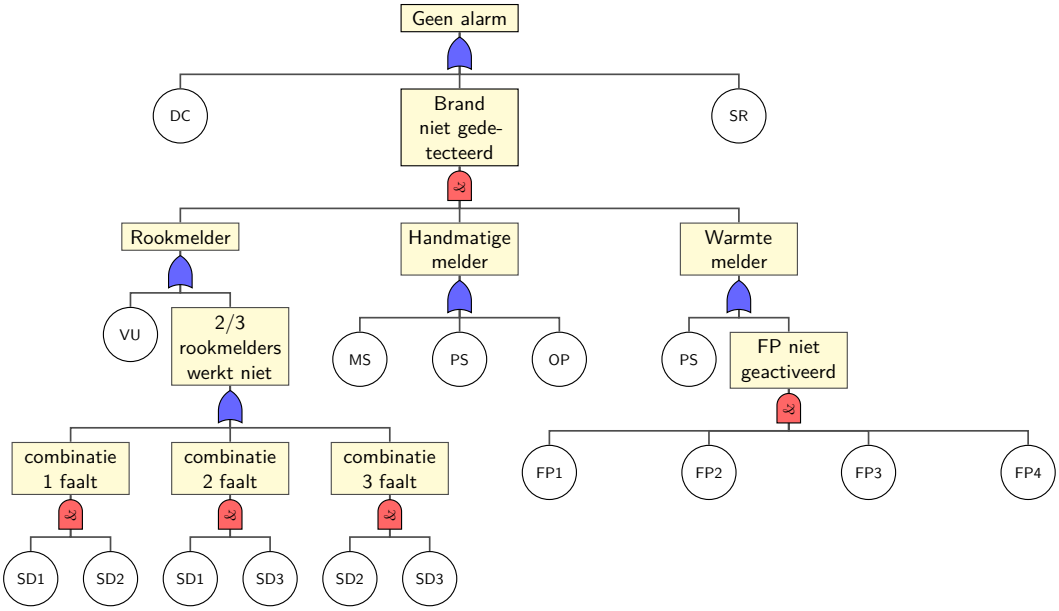
Figuur 2.4: Betrouwbaarheidsblokdiagram van het brandalarm

- Het alarm wordt handmatig geactiveerd.
- De rookmelder geeft een signaal.
- De warmtedetectie geeft een signaal.

Voor ieder van deze redenen is er een diepere oorzaak. Het niet lukken van de handmatige activering kan te wijten zijn aan het niet opengaan van de manuele schakelaar, een defect in de druksensor of de operator kan de schakelaar niet bedienen. Hier hebben we een OF-poort. De warmtedetectie geeft geen signaal omdat de druksensor defect is of de thermische zekeringen defect zijn (OF-poort). Het defect van de thermische zekeringen is een EN-poort omdat alle zekeringen moeten falen voordat het pneumatische systeem faalt. Het laatste detectiesysteem, de rookmelder, faalt als de 2-van-de-3 schakelaar defect is of 2 van de 3 rooksensoren defect zijn (OF-poort). Het defect van 2 van de 3 rooksensoren is een OF-poort van de mogelijke combinaties van rooksensoren. De combinaties zelf zijn EN-poorten omdat beide sensoren defect moeten zijn voordat de combinatie defect is.
Vanuit de foutenboom kunnen de snedes bepaald worden. Men start van het top-event en werkt door de foutenboom naar beneden. Telkens als er een OF-poort tegengekomen wordt, worden de ingangen van deze poort in verschillende rijen geschreven. Bij een EN-poort worden de ingangen in verschillende kolommen geschreven. In het voorbeeld van de rookmelder is het top-event: “geen alarm”. Er is een OF-poort dus de ingangen worden in verschillende rijen geschreven:

{DC},
{“geen detectie”},
{SR}.

Het algoritme wordt verder uitgevoerd tot enkel componenten in de lijst geschreven staan en geen systeem verder opgesplitst kan worden:



Figuur 2.5: Foutenboom van het brandalarm

- "geen detectie" is een EN-poort van de 3 detectiesystemen, dus:

  {DC},
  {"rookmelder", "handmatige activering", "warmtemelder"},
  {SR}.

- de detectiesystemen zijn allen OF-poorten. De rookmelder wordt opgesplitst in twee delen, VU en "2/3 werkt niet":

  {DC},
  {VU, "handmatige activering", "warmtemelder"},
  {"2/3 werkt niet", "handmatige activering", "warmtemelder"},
  {SR}.

- Vervolgens wordt de handmatige schakelaar uitgewerkt:

  {DC},
  {VU, MS, "warmtemelder"},
  {"2/3 werkt niet", MS, "warmtemelder"},
  {VU, PS, "warmtemelder"},
  {"2/3 werkt niet", PS, "warmtemelder"},
  {VU, OP, "warmtemelder"},
  {"2/3 werkt niet", OP, "warmtemelder"},
  {SR}.

- De warmtemelder bestaat uit de “FP inactief” en PS, de opsplitsing geeft:

  {DC},
  {VU, MS, PS},
  {“2/3 werkt niet”, MS, PS},
  {VU, PS, PS},
  {“2/3 werkt niet”, PS, PS},
  {VU, OP, PS},
  {“2/3 werkt niet”, OP, PS},
  {VU, MS, “FP inactief”},
  {“2/3 werkt niet”, MS, “FP inactief”},
  {VU, PS, “FP inactief”},
  {“2/3 werkt niet”, PS, “FP inactief”},
  {VU, OP, “FP inactief”},
  {“2/3 werkt niet”, OP, “FP inactief”},
  {SR}.

- De uitwerking van de elementen “2/3 werkt niet” en “FP inactief” zijn gelijkaardig en worden hier niet vermeld. Enkel de uiteindelijke verzameling van snedes wordt gegeven.

  {DC},
  {VU, MS, PS},
  {SD1, SD2, MS, PS},
  {SD1, SD3, MS, PS},
  {SD2, SD3, MS, PS},
  {VU, PS, PS},
  {SD1, SD2, PS, PS},
  {SD1, SD3, PS, PS},
  {SD2, SD3, PS, PS},
  {VU, OP, PS},
  {SD1, SD2, OP, PS},
  {SD1, SD3, OP, PS},
  {SD2, SD3, OP, PS},
  {VU, MS, FP1, FP2, FP3, FP4},
  {SD1, SD2, MS, FP1, FP2, FP3, FP4},
  {SD1, SD3, MS, FP1, FP2, FP3, FP4},
  {SD2, SD3, MS, FP1, FP2, FP3, FP4},
  {VU, PS, FP1, FP2, FP3, FP4},
  {SD1, SD2, PS, FP1, FP2, FP3, FP4},
  {SD1, SD3, PS, FP1, FP2, FP3, FP4},
  {SD2, SD3, PS, FP1, FP2, FP3, FP4},
  {VU, OP, FP1, FP2, FP3, FP4},
  {SD1, SD2, OP, FP1, FP2, FP3, FP4},
  {SD1, SD3, OP, FP1, FP2, FP3, FP4},
  {SD2, SD3, OP, FP1, FP2, FP3, FP4},
  {SR}.

Het resultaat is een vector van snedes. De laatste stap is het minimaliseren van de verzameling van de snedes. Dit geeft

{DC},
{VU, PS},
{SD1, SD2, PS},
{SD1, SD3, PS},
{SD2, SD3, PS},
{VU, OP, FP1, FP2, FP3, FP4},
{VU, MS, FP1, FP2, FP3, FP4},
{SD1, SD2, MS, FP1, FP2, FP3, FP4},
{SD1, SD3, MS, FP1, FP2, FP3, FP4},
{SD2, SD3, MS, FP1, FP2, FP3, FP4},
{SD1, SD2, OP, FP1, FP2, FP3, FP4},
{SD1, SD3, OP, FP1, FP2, FP3, FP4},
{SD2, SD3, OP, FP1, FP2, FP3, FP4},
{SR}.

Deze verzameling van minimale snede wordt gebruikt voor de berekening van de levensduur.

2.4 Levensduur van een systeem

Gebruikmakend van de verdeling van de levensduur van de componenten en de verzameling van minimale snedes kan de levensduur van het systeem gesimuleerd worden. In dit onderzoek ligt de focus op niet-gerichte netwerken bestaande uit faalbare componenten en verbindingen die perfect betrouwbaar zijn. Niet-perfect betrouwbare verbindingen worden weergegeven door een serieschakeling van een perfect betrouwbare verbinding en een faalbare component[3]. Een systeem is niet gefaald als er een pad is van de ingang naar de uitgang met enkel werkende componenten. Als alle componenten in een snede C falen is er geen pad meer van de ingang naar de uitgang van het systeem. De levensduur is dus bepaald door de verzameling van minimale snedes $\mathcal{C} = \{C\}$ en kan geschreven worden door volgende formule [5]:

$$T_s = \min_{C \in \mathcal{C}} \{ \max_{c \in C} \{t_c\} \}. \tag{2.10}$$

Hier is t_c de levensduur van component c en T_s is de levensduur van het systeem voor deze simulatie . Met de Monte Carlo methodes die geïntroduceerd worden in de volgende hoofdstukken wordt er een berekening gedaan voor de verwachte waarde van T_s, $\mathbb{E}[T_s]$, gebaseerd op een reeks simulaties.

Hoofdstuk 3

Multilevel Monte Carlo

De betrouwbaarheidsanalyse van de systemen in deze thesis is een probleem met hoge dimensies. De bepaling van de levensduur van een systeem is een s-dimensionaal probleem met s het aantal componenten. Monte Carlo (MC) is een techniek om de verwachte waarde van een grootheid te bepalen die geen last heeft van de "vloek van de dimensie" ("*curse of dimensionality*") [16]. Multilevel Monte Carlo is een uitbreiding op de standaard Monte Carlo geïntroduceerd door M. Giles in [4]. Hierin is Multilevel Monte Carlo toegepast op stochastische differentiaalvergelijkingen. Dit hoofdstuk bespreekt Monte Carlo en de evolutie naar Multilevel (Quasi-)Monte Carlo. In het eerste deel wordt uitleg gegeven over standaard Monte Carlo (sectie 3.1). Vervolgens wordt er in sectie 3.2 dieper ingegaan op de Multilevel Monte Carlo en het laatste deel, sectie 3.3, bespreekt het gebruik van Multilevel Quasi-Monte Carlo.

3.1 Monte Carlo methode

Stel de integraal over een s-dimensionale eenheidskubus $[0,1]^s$,

$$I(f) = \int_0^1 \ \dots \ \int_0^1 f(x_1, x_2, \dots, x_s) dx_1 dx_2 \ \dots \ dx_s = \int_{[0,1]^s} f(\mathbf{x}) \mathrm{d}\mathbf{x}. \tag{3.1}$$

De Monte Carlo methode wordt gebruikt om een benadering te bepalen van de integraal door een kwadratuurformule te gebruiken die het gemiddelde neemt van willekeurige simulaties van de functie gebaseerd op N onafhankelijke gelijk verdeelde punten $\mathbf{t}_1, \dots, \mathbf{t}_N$ [16]:

$$Q_N(f) = \frac{1}{N} \sum_{j=1}^{N} f(\mathbf{t}_j). \tag{3.2}$$

Dit levert een schatting op die geen afwijking vertoont, $\mathbb{E}[Q_N(f)] = I(f)$ met een gemiddelde kwadratische fout die gegeven wordt door

$$\mathbb{E}[(I(f) - Q_N(f))^2] = \frac{\mathbb{V}(f)}{N} \tag{3.3}$$

waarbij $\mathbb{V}(f)$ de variantie van de functie $f(x)$ is.

3.1.1 De verwachte waarde via Monte Carlo

De verwachte waarde van een bepaald proces of functie $\mathbb{E}[g(x)]$ kan geschreven worden als een integraal. Dit is gegeven door het LOTUS theorema. Als x afkomstig is van een continue verdeling $f(x)$ dan is de verwachte waarde gegeven door

$$\mathbb{E}[g(x)] = \int_D g(x)f(x)dx. \tag{3.4}$$

met $D \in \mathbb{R}$ en $f(x)$ volgens de waarschijnlijkheidsdichtheid:

$$f(x) > 0 \quad \forall x \in D, \quad \int_D f(x)dx = 1.$$

Statistisch gezien kan x aanschouwd worden als een willekeurige sample van een toevalsvariabele met een waarschijnlijkheidsdichtheid $f(x)$. $g(x)$ kan dan gezien worden als een toevalsvariabele. De verwachte waarde van g(x) is dan

$$G \coloneqq \mathbb{E}[g(x)] = \int_D g(x)f(x)dx.$$

De variantie van $g(x)$ wordt gegeven door

$$\mathbb{V}[g(x)] = \int_D [g(x) - G]^2 f(x)dx = \mathbb{E}[g(x)^2] - G^2.$$

De Monte Carlo methode schat de waarde van G door een reeks van N waarden voor x te samplen: $x_1, x_2, x_3, \ldots, x_N$ volgens de waarschijnlijkheidsdichtheid $f(x)$ en vervolgens de waarde $g(x)$ te bepalen voor al de waarden in de reeks: $g(x_1), g(x_2), g(x_3,), \ldots, g(x_N)$. Het gemiddelde, het resultaat van de MC schatter, wordt gegeven door

$$G_N = \frac{1}{N}\sum_{i=1}^{N} g(x_i).$$

G_N is een toevalsvariabele die de verwachte waarde van de functie $g(x)$ benadert. De verwachte waarde en variantie van de MC schatter kunnen geschreven worden als

$$\mathbb{E}[G_N] = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[g(x_i)], \quad \mathbb{V}[G_N] = \frac{1}{N^2}\sum_{i=1}^{N}\mathbb{V}[g(x_i)],$$

waarbij de verwachte waarde $\mathbb{E}[g(x_i)]$ en variantie $\mathbb{V}[g(x_i)]$ de verwachte waarde en variantie zijn van $g(x)$. Omdat de waarschijnlijkheidsdichtheid $f(x)$ onafhankelijk is van de rangorde van de waarde, volgt hieruit dat

$$\mathbb{E}[g(x_i)] = \mathbb{E}[g(x)] = G, \quad \mathbb{V}[g(x_i)] = \mathbb{V}[g(x)].$$

De waarde voor G kan dus benaderd worden door een schatting met $N \gg 1$:

$$\mathbb{E}[g(x)] \approx \frac{1}{N}\sum_{i=1}^{N} g(x_i) = G_N, \tag{3.5}$$

$$\mathbb{V}[G_N] = \frac{1}{N}\left(\mathbb{E}[g(x)^2] - \mathbb{E}[g(x)]^2\right). \tag{3.6}$$

Waarbij de veronderstelling gemaakt is dat de waarschijnlijkheid onafhankelijk is van de rangorde en dat de reeks samples steeds representatief is voor het fysiek proces [29].

3.1.2 Simulatie van de levensduur

De Monte Carlo berekening van de bepaling van de levensduur van een systeem gebaseerd op de s componenten en de verzameling van de minimale snedes $\mathcal{C} = \{C\}$ wordt gegeven door:

$$\mathbb{E}[T_s] \approx \hat{T}_s = \frac{1}{N}\sum_{i=1}^{N} T_s^{(i)} = \frac{1}{N}\sum_{i=1}^{N} \min_{C\in\mathcal{C}}\{\max_{c\in C}\{t_c^{(i)}\}\}. \tag{3.7}$$

$T_s^{(i)}$ is hier de i-de simulatie van de levensduur van het systeem met $t_c^{(i)}$ de levensduur van de componenten $c = 0, 1, \ldots, s$ voor deze simulatie. De rekenkost van de Monte Carlo methode is de rekenkost voor het berekenen van één simulatie vermenigvuldigd met het aantal nodige simulaties om de gewenste nauwkeurigheid ε te bekomen. De rekenkost van de Monte Carlo berekening wordt bijgevolg bepaald door drie factoren. Ten eerste, de grootte van de verzameling van de snedes en de grootte van de snedes. In het minst gunstige geval groeit de verzameling van de snedes exponentieel met het aantal componenten, dit resulteert in een exponentiële groei van de rekenkost van de berekening van een simulatie. Hetzelfde geldt bij de groei van de snedes, grotere snedes betekent een grotere rekenkost per snede en dus een grotere rekenkost per simulatie. Een tweede factor die de rekenkost beïnvloed is de variantie van de levensduur van het systeem, dus de verdeling van de levensduur van de componenten en de derde factor is de opgegeven nauwkeurigheid.
De gemiddelde kwadratische fout (MSE) bij Monte Carlo berekeningen is bepaald door de variantie van de Monte Carlo berekening en een afwijking die ontstaat als de simulatie niet exact gebeurt (bias):

$$\text{MSE} = \mathbb{E}[(\hat{T}_s - \mathbb{E}T_s)^2] = \mathbb{E}[(\hat{T}_s - \mathbb{E}\hat{T}_s)^2] + (\hat{T}_s - \mathbb{E}T_s)^2 = \frac{1}{N}\mathbb{V}[T_s^{(i)}] + \text{bias}^2. \tag{3.8}$$

Het resultaat van de Monte Carlo berekening in de context van de betrouwbaarheidsanalyse is een zuivere schatting. De gemiddelde kwadratische fout bestaat dus enkel uit de variantie van de Monte Carlo berekening. Deze variantie is evenredig met N^{-1} en is bepaald door de variantie van de levensduur. Als de variantie van de levensduur van het systeem groot is, zal het aantal nodige simulaties (N) ook groot zijn [2]. De derde factor, de opgegeven nauwkeurigheid ε is vanzelfsprekend. Men wilt een gemiddelde kwadratische fout MSE $< \varepsilon^2$, als men ε verlaagd moet men een groter aantal simulaties uitvoeren.

3.2 Multilevel Monte Carlo

Multilevel Monte Carlo heeft als doel de variantie te reduceren door simulaties op verschillende levels met stijgende nauwkeurigheid te gebruiken zodat in de berekening

dezelfde nauwkeurigheid als in de standaard Monte Carlo berekening wordt bekomen maar met een lagere kost. Een eerste introductie van het idee werd gedaan in [11] door Stefan Heinrich via parametrische integratie. In parametrische integratie wordt er gezocht naar de verwachte waarde $\mathbb{E}[f(x,\lambda)]$ van een bepaalde functie $f(x,\lambda)$ met x een eindig-dimensionale toevalsvariabele en λ een parameter.
Stel λ een getal tussen 0 en 1 en de verwachte waarde $\mathbb{E}[f(x,0)]$ en $\mathbb{E}[f(x,1]$ zijn gekend dan geldt voor $\lambda = \frac{1}{2}$:

$$\mathbb{E}\left[f(x,\frac{1}{2})\right] = \frac{1}{2}(\mathbb{E}[f(x,0)] + \mathbb{E}[f(x,1)]) + \mathbb{E}[f(x,\frac{1}{2}) - \frac{1}{2}(f(x,0) + f(x,1))]. \quad (3.9)$$

Dit kan vervolgens recursief uitgevoerd worden voor het bepalen van de verwachte waarden op een fijn rooster [11].
Een andere introductie naar Multilevel Monte Carlo begint bij de methode van de controlevariabelen, dit wordt besproken in een eerste subsectie (sectie 3.2.1). Een tweede deel bespreekt de Multilevel Monte Carlo berekening geïntroduceerd door M. Giles in 2008 (sectie 3.2.2) en wordt het theorema van Multilevel Monte Carlo besproken. In het laatste deel, sectie 3.3, wordt de theoretische rekenkost van Quasi-Monte Carlo geïntroduceerd.

3.2.1 Monte Carlo met twee levels

Multilevel Monte Carlo kan gezien worden als het recursief toepassen van de methode van de controlevariabelen. Het gebruik van de methode van de controlevariabelen is een manier om de variantie te reduceren. Hierbij wordt een variabele g gebruikt die gecorreleerd is met f en waarvan de verwachte waarde $\mathbb{E}[g]$ gekend is . De methode van de controlevariabelen schatter wordt gegeven door

$$\mathbb{E}[f] = \lambda\mathbb{E}[g] + \mathbb{E}[f - \lambda g] \quad (3.10)$$

$$\mathbb{E}[f] = \frac{1}{N}\sum_{n=1}^{N}\left(f^{(n)} - \lambda(g^{(n)} - \mathbb{E}[g])\right). \quad (3.11)$$

λ is hier een variabele waarvan de optimale waarde geschat kan worden gebaseerd op een aantal samples [25].
Bij Multilevel Monte Carlo met 2 levels wordt dit idee gebruikt. Stel dat men een benadering $\mathbb{E}[T_1]$ wilt bepalen aan de hand van een ruwere benadering $\mathbb{E}[T_0]$ met een lagere kost. Omdat de verwachte waarde een lineaire operator is, kan men $\mathbb{E}[T_1]$ bepalen via

$$\mathbb{E}[T_1] = \mathbb{E}[T_0] + \mathbb{E}[T_1 - T_0] \quad (3.12)$$

$$= N_0^{-1}\sum_{n=1}^{N_0} T_0^{(n)} + N_1^{-1}\sum_{n=1}^{N_1}(T_1^{(n)} - T_0^{(n)}). \quad (3.13)$$

Hier is $(T_1^{(n)} - T_0^{(n)})$ het verschil tussen T_1 en T_0 voor dezelfde simulatie.
Stel dat C_0 en C_1 de kost zijn van de berekening van respectievelijk T_0 en $[T_1 - T_0]$,

dan is de totale kost van de schatter met twee levels: $N_0C_0 + N_1C_1$. Als V_0 en V_1 de varianties zijn van T_0 en $[T_1 - T_0]$ dan is de totale variantie $N_0^{-1}V_0 + N_1^{-1}V_1$. In de benadering zijn N_0 en N_1 variabelen, de variantie wordt geminimaliseerd bij een vaste kost via

$$N_1/N_0 = \frac{\sqrt{V_1/C_1}}{\sqrt{V_0/C_0}}. \tag{3.14}$$

3.2.2 Multilevel Simulatie

Multilevel Monte Carlo is in 2008 geïntroduceerd door M. Giles voor de integratie van stochastische differentiaalvergelijkingen waarbij er een brownse diffusie plaatsvindt [4]. Veronderstel $L + 1$ levels waarbij de gezochte waarde T geschat wordt met benaderingen $\hat{T}_\ell$ met $\ell = 0, 1, ..., L$ waarbij de benadering $\hat{T}_\ell$ nauwkeuriger is dan $\hat{T}_k$ als $\ell > k$ is. Door enkel gebruik te maken van de benadering op het grootste level, $E[\hat{T}_L]$, wordt er een standaard Monte Carlo simulatie uitgevoerd. De verwachte waarde is een lineaire operator dus deze benadering kan, zoals in de methode met twee levels, herschreven worden naar een telescopische som die de benaderingen op elk level bevat

$$\mathbb{E}[\hat{T}_L] = \mathbb{E}[\hat{T}_0] + \sum_{\ell=1}^{L} \mathbb{E}[\hat{T}_\ell - \hat{T}_{\ell-1}]. \tag{3.15}$$

De verwachtte waarde $\mathbb{E}[\hat{T}_\ell - \hat{T}_{\ell-1}]$ wordt bepaald met N_ℓ simulaties en $\hat{T}_\ell$ en $\hat{T}_{\ell-1}$ zijn de benadering op level ℓ en $\ell - 1$ van dezelfde simulatie.
Stel dat C_0 en $V_0 = \mathbb{V}[\hat{T}_0]$ de kost en variantie van de benadering Y_0 voor $\mathbb{E}[\hat{T}_0]$ en C_ℓ en $V_\ell = \mathbb{V}[\hat{T}_\ell - \hat{T}_{\ell-1}]$ de kost en variantie van de benadering Y_ℓ voor $\mathbb{E}[\hat{T}_\ell - \hat{T}_{\ell-1}]$. De variantie V_ℓ wordt kleiner naarmate ℓ groter wordt omdat $\hat{T}_{\ell-1}$ en $\hat{T}_\ell$ een benadering zijn van dezelfde grootheid T met stijgende nauwkeurigheid. De benaderingen voor de verwachte waarden die hier gebruikt worden zijn het gemiddelde van N_ℓ onafhankelijke simulaties (de MC methode):

$$Y_0 = N_0^{-1} \sum_{i=1}^{N_0} (\hat{T}_0^{(i)}), \tag{3.16}$$

$$Y_\ell = N_l^{-1} \sum_{i=1}^{N_l} (\hat{T}_\ell^{(i)} - \hat{T}_{\ell-1}^{(i)}), \tag{3.17}$$

$$\mathbb{E}[\hat{T}_L] \approx Y = \sum_{\ell=0}^{L} Y_\ell. \tag{3.18}$$

Het superscript $^{(i)}$ is toegevoegd om aan te geven dat dezelfde simulatie is gebruikt voor $\hat{T}_\ell$ en $\hat{T}_{\ell-1}$ te bepalen. De totale variantie van de schatter is gegeven door

$$\mathbb{V}\left[\hat{T}_L\right] = \sum_{\ell=0}^{L} N_\ell^{-1} V_\ell, \quad V_\ell = \mathbb{V}[\hat{T}_\ell - \hat{T}_{\ell-1}]. \tag{3.19}$$

De kost van de berekening is evenredig met $\sum_{\ell=0}^{L} N_\ell C_\ell$. De variantie kan men

minimaliseren bij een vaste kost door N_ℓ evenredig te nemen met $\sqrt{V_\ell C_\ell}$ [10].
Om er voor te zorgen dat de standaardfout van de simulatie lager is dan de opgegeven tolerantie ε zijn er twee parameters die gekozen moeten worden. Ten eerste moet level L zo gekozen worden dat $(\mathbb{E}[\hat{T}_L] - \mathbb{E}[T])^2 < \frac{1}{2}\varepsilon^2$ en ten tweede moet N_ℓ voor $\ell = 0, 1, 2, ..., L$ evenredig gekozen worden met $\sqrt{V_\ell / C_\ell}$ zodat de totale variantie lager is dan $\frac{1}{2}\varepsilon^2$. M. Giles geeft in [10] een theorema voor de efficiëntie van Multilevel Monte Carlo (Theorema 1).

Theorema 1 *Stel dat T een functie is van de oplossing van een stochastisch proces en $\hat{T}_\ell$ een benadering op level ℓ. Dan geldt dat: als er onafhankelijke benaderingen Y_ℓ, gebaseerd op N_ℓ Monte Carlo samples, bestaan en er zijn positieve constanten α, β, γ, c_1, c_2, c_3 zodat:*

1. $\alpha > \frac{1}{2}\min(\beta, \gamma)$
2. $\left| \mathbb{E}[\hat{T}_\ell - T] \right| \leq c_1 2^{-\alpha\ell}$
3. $\mathbb{E}[Y_0] = \mathbb{E}[\hat{T}_0]$
4. $\mathbb{E}[Y_\ell] = \mathbb{E}[\hat{T}_\ell - \hat{T}_{\ell-1}]$
5. $\mathbb{V}[Y_\ell] \leq c_2 N_\ell^{-1} 2^{-\beta\ell}$
6. $\mathbb{E}[C_\ell] \leq c_3 2^{\gamma\ell}$

Dan bestaat er een positieve constante c_4 zodat er voor een $\varepsilon < 1$ een L en een N_ℓ voor $\ell = 0, 1, \ldots, L$ bestaat zodat de multilevel schatter $Y = \sum_{l=0}^{L} Y_\ell$ een gemiddelde kwadratische fout heeft met grens $\mathbb{E}[(Y - \mathbb{E}[T])^2] < \varepsilon^2$ met een verwachte rekenkost C gebonden door:

$$C \leq \begin{cases} c_4 \varepsilon^{-2} & , \quad \beta > \gamma \\ c_4 \varepsilon^{-2} (log\ \epsilon)^2 & , \quad \beta = \gamma \\ c_4 \varepsilon^{-2-(\gamma-\beta)/\alpha} & , \quad 0 < \beta < \gamma \end{cases} \tag{3.20}$$

Standaard Monte Carlo simulaties hebben $O(\varepsilon^{-2})$ samples nodig voor een MSE $< \varepsilon^2$. Uit het theorema is af te leiden dat de kost afhankelijk is van β, de parameter van de afname van de variantie en γ, de parameter gekoppeld met de toename van de kost. Asymptotisch zijn er 3 situaties:

- $\beta < \gamma$: het aantal nodige simulaties op ieder level voor de MLMC neemt trager af dan de toename van de kost voor een simulatie op ieder level. De rekenkost van de simulatie is vooral bij de hoogste levels. De kost is in dat geval $\approx V_L C_L$.
- $\beta = \gamma$: de afname van de variantie wordt opgeheven door de toename in de kost. De bijdrage aan de totale kost is gelijk op ieder level.
- $\beta > \gamma$: De bijdrage aan de totale kost van ieder level daalt omdat de variantie sneller daalt dan de kost toeneemt. Dit is de meest wenselijke situatie want de kost is in dat geval ongeveer gelijk aan $V_0 C_0$.

De rekenkost voor Monte Carlo is ongeveer $V_0 C_L$. De performantie van Multilevel Monte Carlo is het grootste als $\beta > \gamma$, hierbij is de rekenkost ongeveer $V_0 C_0$. Door de lagere kost op het laagste level is Multilevel Monte Carlo efficiënter dan Monte Carlo. In het minst gunstige geval, $\beta < \gamma$, is de kost van MLMC $\approx V_L C_L$ door de levels op te bouwen zodat $V_L < V_0$ blijft Multilevel Monte Carlo efficiënter als Monte Carlo. Multilevel Monte Carlo is een methode dat gecombineerd kan worden met andere methodes die zorgen voor een reductie in de rekenkost. Een mogelijke manier om de kost verder drukken is door gebruik te maken van Quasi Monte Carlo. Dit wordt besproken in de volgende sectie.

3.3 Quasi-Monte Carlo

De uitbreiding van deze masterproef op de bestaande implementatie is de introductie van Quasi-Monte Carlo in het Multilevel Monte Carlo algoritme met als doel de rekenkost verder te verlagen. Bij de standaard Monte Carlo methodes worden de waarden voor van de ingangsvariabelen willekeurig gekozen. Bij Quasi-Monte Carlo worden deze waarden op een deterministische manier gekozen. Dit zorgt voor meer uniforme verdeling. De mogelijke methodes die gebruikt worden voor het bepalen van de punten zijn Sobol' sequenties, digitale netten en rang-1 roosters.
Volgens M. Giles, zou het gebruik van Quasi-Monte Carlo het meest effectief zijn in het geval dat $\beta > \gamma$ omdat in dit geval het optimale aantal simulaties in ieder level zorgt dat de grootste rekenkost bij de lage levels ligt. Dit is een ideale situatie voor Quasi-Monte Carlo simulaties waarbij een rekenkost mogelijk zakt tot $O(\varepsilon^{-1})$ [4][17][18].
Quasi-Monte Carlo simulaties werden gebruikt door Giles en Waterhouse in [27]. De punten worden gegenereerd door gebruik te maken van rang-1 roosterregels. Resultaten tonen dat het gebruik van Quasi-Monte Carlo het meest effectief is in problemen met een lage effectieve dimensie. Enkele theoretische ontwikkelingen tonen goede resultaten voor het gebruik van Quasi-Monte Carlo, resulterend in methodes met een rekenkost van $O(\epsilon^{-p})$ met $p < 2$ in gunstige omstandigheden [10].

3.3.1 Rang-1 roosterregels

In dit onderzoek wordt er gebruik gemaakt van de rang-1 roosterregels. Voor een s-dimensionaal probleem over de eenheidskubus $[0,1]^s$ worden de punten bepaald volgens een genererende vector $\mathbf{z} \in \mathbb{Z}^s$. De punten worden gegeven op basis van volgende structuur:

$$\mathbf{x}_k = \frac{\mathbf{z}k}{N} \coloneqq \frac{\mathbf{z}k}{N} \bmod 1 = \frac{\mathbf{z}k \ mod \ N}{N} \quad k = 0, 1, \ldots, N. \tag{3.21}$$

Een visuele weergave van willekeurige punten, zoals gebruikt in de MC methode, en Quasi-Monte Carlo punten,deterministische punten, in twee dimensies is gegeven in figuur 3.1. Een goede vector is belangrijk voor de correctheid en effectiviteit van de methode. Een minder goede keuze voor de genererende vector zou zijn: $\mathbf{z} = (1, ..., 1)$.

Hierbij liggen alle punten op de diagonaal in $[0,1]^s$ en deze vector geeft dus punten die niet representatief zijn voor het fysieke proces.
De genererende vector wordt opgesteld via het component-per-component constructie algoritme van Dick et al. [8]. Dit algoritme heeft een efficiënte implementatie door de snelle FFT implementatietechniek van Nuyens en Cools [20].
De introductie van Quasi-Monte Carlo zorgt voor een afwijking in de gevraagde variabele. Bijkomend is het niet meer mogelijk een betrouwbaarheidsinterval te berekenen voor de verwachte waarde. De oplossing hiervoor is het gebruik maken van onafhankelijk gelijk verdeelde willekeurige verschuivingen (i.i.d. random shifts) van het rooster veroorzaakt door een vector $\Delta \in [0,1)^s$:

$$\mathbf{x}_k^{(i)} = (\mathbf{x}_k + \Delta^{(i)}) \bmod 1 = \frac{(\mathbf{z}k + N\Delta^{(i)}) \bmod N}{N} \quad i = 0,\ldots,M,\ k = 1,\ldots,N. \tag{3.22}$$

Vector Δ wordt verondersteld als een willekeurige variabele in s-dimensies. Figuur 3.1c toont de verschoven Quasi-Monte Carlo punten uit figuur 3.1b met willekeurige verschuiving $\Delta = [0.205, 0.078]^T$.
De Quasi-Monte Carlo methode bestaat dus uit een reeks berekeningen die wordt herhaald met M verschillende willekeurige vectoren Δ. Dit levert M onafhankelijke schattingen voor de gezochte waarde: $Q_N^{(1)}, Q_N^{(2)}, ..., Q_N^{(M)}$. Waarbij een berekening gegeven is door

$$Q_N^{(i)} = \frac{1}{N}\sum_{k=1}^{N} f(\mathbf{x}_k^{(i)}). \tag{3.23}$$

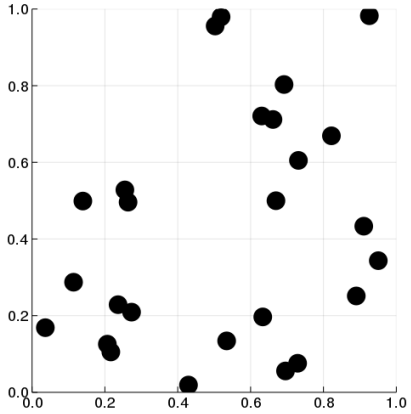
De punten $\mathbf{x}_k^{(i)}$ zijn de punten gegeven door formule 3.22. Hieruit volgt een zuivere schatter waarbij het mogelijk is om een betrouwbaarheidsinterval op te stellen:

$$\bar{Q}_{MxN} = \frac{1}{M}\sum_{i=1}^{M} Q_N^{(i)} \tag{3.24}$$

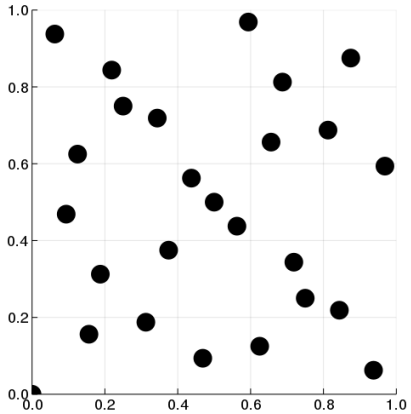
$$\bar{Q}_{MxN} = \frac{1}{M}\sum_{i=1}^{M}(\frac{1}{N}\sum_{k=1}^{N} f(\mathbf{x}_k)). \tag{3.25}$$

3.3.2 Sobol' sequentie

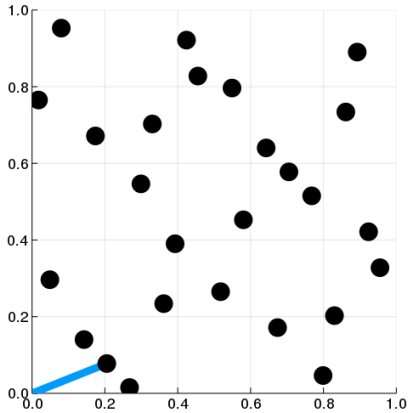
Sobol' sequenties zijn een andere mogelijkheid om uniform verdeelde, Quasi-Monte Carlo getallen te genereren. Ze behoren tot de digitale netten met basis 2. De digitale netten worden opgebouwd met een digitaal constructie schema waarbij er vertrokken wordt van een vector van genererende matrices $C_1, C_2, \ldots, C_s$. De constructie van de Sobol' punten is gebaseerd op een primitieve veelterm en een initiële richting voor elke dimensie. Deze veelterm en richtingen leiden tot de genererende matrices C_j voor iedere dimensie $j = 0, 1, 2, ..., s$. Voor een gedetailleerde beschrijving van de Sobol' sequenties wordt er verwezen naar [1] en [14].
Voor de generatie van de sobol' getallen wordt er gebruik gemaakt van de Sobol' generator beschikbaar gesteld door F. Y. Kuo en D. Nuyens [16][13].



(a) Monte Carlo punten



(b) Quasi-Monte Carlo punten



(c) Verschoven Quasi-Monte Carlo punten ($\Delta = [0.205, 0.078]^T$)

Figuur 3.1: MC punten,QMC punten in 2 dimensies en de toepassing van een willekeurige verschuiving

Hoofdstuk 4

Implementatie

Voor de implementatie van Multilevel Monte Carlo in de betrouwbaarheidsanalyse is er gebruik gemaakt van de programmeertaal Julia. De reden voor deze keuze is de mogelijkheid om de reeds aanwezige code van het departement Computerwetenschappen te gebruiken. Het doel is om de gemiddelde tijd tot aan falen in de betrouwbaarheidsanalyse die tot op heden gebeurt met Monte Carlo te berekenen met een Multilevel Monte Carlo methode. Er is gebruik gemaakt van de methode voorgesteld door Aslett in [2], deze methode combineert de theorie van M. Giles met benaderingen van de levensduur afkomstig van deelverzamelingen van de verzameling van minimale snedes van een systeem. Deze benaderingen zijn ruwere benaderingen van de levensduur, de gemiddelde tijd tot aan falen, die minder nauwkeurig zijn maar bepaald kunnen worden met een lagere rekenkost.
Gebaseerd op de voorgaande hoofdstukken is het mogelijk om $L+1$ geneste deelverzamelingen van de verzameling van minimale snedes $\mathcal{C}_0 \subset \mathcal{C}_1 \subset \ldots \subset \mathcal{C}_{\ell-1} \subset \mathcal{C}_\ell \subset \ldots \subset \mathcal{C}_L = \mathcal{C}$ te construeren. De benaderingen van de gemiddelde tijd tot aan falen $\hat{T}_0, \ldots, \hat{T}_L = \hat{T}_S$ gebaseerd op deze deelverzamelingen worden bepaald met formule 3.7 waarbij de simulaties voor level ℓ gebeuren over de deelverzameling $\mathcal{C}_\ell$ in plaats van over de verzameling van minimale snedes $\mathcal{C}$:

$$T_\ell^{(i)} = \min_{C \in \mathcal{C}_\ell} \{\max_{c \in C} \{t_c^{(i)}\}\}. \tag{4.1}$$

Deze benaderingen hebben een stijgende nauwkeurigheid maar ook een grotere kost als $\ell \to L$. De simulatie op level L geeft de standaard Monte Carlo berekening. Volgens de theorie van Multilevel Monte Carlo (sectie 3.2) kan er een telescopische som gevormd worden:

$$\mathbb{E}[\hat{T}_S] = \mathbb{E}[\hat{T}_0] + \sum_{\ell=1}^{L} \mathbb{E}[\hat{T}_\ell - \hat{T}_{\ell-1}]. \tag{4.2}$$

Dit is een schatter die een verwachte waarde heeft die gelijk is aan de verwachte waarde van de standaard Monte Carlo schatter maar voor een opgegeven nauwkeurigheid een lagere rekenkost heeft.
In dit hoofdstuk wordt er dieper ingegaan op Multilevel Monte Carlo specifiek voor

de betrouwbaarheidsanalyse. Sectie 4.1 richt zich op de constructie van de deelverzamelingen, de selectie van de levels. Sectie 4.2 bespreekt het numerieke algoritme voor de Multilevel Monte Carlo berekening. De overstap naar Multilevel Quasi-Monte Carlo is gedaan in sectie 4.3. Tenslotte, geeft sectie 4.4 mogelijke uitbreidingen die toegepast kunnen worden.

4.1 Selecteren van de levels

Voor het opstellen van de levels is er vertrokken van het algoritme vernoemd in het artikel van Aslett [2]. Dit geeft een algoritme dat binnen acceptabele tijd een levelstructuur op een inductieve manier opbouwt. Het algoritme selecteert een groep minimale snedes $\mathcal{C}_\ell$ voor level ℓ zodat ieder level ℓ $\lceil \#C/2^{L-\ell} \rceil$ snedes bevat en waarbij $\mathcal{C}_{\ell-1} \subset \mathcal{C}_\ell$, met $\#C$ het aantal minimale snedes van het systeem. De verdubbeling van het aantal snedes zorgt voor een optimale groei van de rekenkost per level [4].

4.1.1 level $\ell = 0$

Het is gewenst dat dit level de beste benadering is voor de levensduur van het systeem aangezien het grootste aantal simulaties gedaan wordt op level 0. De beste benadering betekent in dit geval de verzameling van $\lceil \#C/2^L \rceil$ minimale snedes die de kortste levensduur heeft. Om deze te bepalen start het algoritme met een ruwe Monte Carlo simulatie waarbij N' simulaties (vb. $N' = 100$) worden uitgevoerd. In elke simulatie wordt de levensduur van elk component genomen uit de verdeling en wordt de levensduur van elke minimale snede bepaald volgens

$$\eta_i := \frac{1}{N'} \sum_{j=1}^{N'} \max_{C \in C_i} t_c^{(j)} \quad \forall C_i \in \mathcal{C}. \tag{4.3}$$

De snedes worden gesorteerd volgens stijgende waarden van η_i. Level 0 is de verzameling van de eerste $\lceil \#C/2^L \rceil$ minimale snedes.

4.1.2 level $\ell > 0$

Stel dat de levels $0, 1, ..., \ell - 1$ reeds gevormd zijn. De snedes die toegevoegd worden aan de verzameling voor level ℓ worden geselecteerd zodat de efficiëntie van de Multilevel Monte Carlo berekening wordt gemaximaliseerd. Volgens theorema 1 is dit als $\beta > \gamma$, het doel is dus de snelle daling van de variantie van de correctieterm. Dit gebeurt door de snedes uit $\mathcal{C}_{trial} = \mathcal{C} \backslash \mathcal{C}_{\ell-1}$ te selecteren waarvoor $\mathbb{E}[T_{\ell-1} - T_\ell]$ maximaal wordt. Dit zorgt dat de variantie vooral aanwezig is bij de laagste levels. De deelverzameling $\mathcal{C}_\ell$ van level ℓ wordt gegeven door

$$\mathcal{C}_\ell = \arg \max_{C_\ell \subseteq C} \mathbb{E}[T_{\ell-1} - T_\ell] \qquad \text{s.t. } \mathcal{C}_{\ell-1} \subset \mathcal{C}_\ell,\ \#\mathcal{C}_\ell = [\#C/2^{L-\ell}]. \tag{4.4}$$

De uitleg hiervoor is als volgt. De verzamelingen worden bepaald zodat $\mathcal{C}_0 \subset \mathcal{C}_1 \subset ... \subset \mathcal{C}_{\ell-1} \subset \mathcal{C}_\ell \subset ... \subset \mathcal{C}_L$. De levensduur op level ℓ is dezelfde als de levensduur in level $\ell - 1$ tenzij er in $\mathcal{C}_\ell \backslash \mathcal{C}_{\ell-1}$ een snede C_k is waarvoor $\max_{c \in C_k}\{t_c\} < T_{\ell-1}$. Omdat er gewenst is dat de kleinere levels de beste benadering zijn voor de levensduur, moet men de snedes voor een verzameling op level ℓ kiezen zodat $T_\ell < T_{\ell-1}$. De beste keuze is de verzameling snedes zodat T_ℓ veel kleiner is dan $T_{\ell-1}$. Door de levels op deze manier te selecteren, wordt de bijdrage van elk level gemaximaliseerd.
Er is een bovengrens voor $\mathbb{E}[T_{\ell-1} - T_\ell]$ aangezien de levensduur van een verzameling bepaald wordt door de minimale levensduur van de snedes. Omdat $\mathcal{C}_{\ell-1} \subset \mathcal{C}_\ell$ is de bovengrens gegeven door:

$$\mathbb{E}[T_{\ell-1} - T_\ell] \leq \mathbb{E}[T_{\ell-1} - \min\{T_{\ell-1}, \max_{c \in C}\{t_c\}\}], \quad \forall C \in \mathcal{C}_{trial}. \tag{4.5}$$

De groep van de snedes die toegevoegd worden aan de verzameling voor level ℓ minimaliseert deze bovengrenzen, de selectie gaat als volgt.

1. Gebruik de N' simulaties van de ruwe Monte Carlo simulatie op level $\ell = 0$.
2. Bereken volgende benadering voor de bovengrenzen:

$$\delta_k = \frac{1}{N'} \sum_{j=1}^{N'} [T_{\ell-1} - \min\{T_{\ell-1}, \max_{c \in C_k}\{t_c^{(j)}\}\}], \quad \forall C_k \in \mathcal{C}_{trial}.$$

3. Sorteer δ_k volgens dalende waarde.
4. De verzameling van snedes in level ℓ bestaat uit de verzameling van snedes in level $\ell - 1$, $\mathcal{C}_{\ell-1}$, met de M_ℓ snedes met de grootste δ_k waarbij $M_\ell = \lceil \#C/2^{L-\ell} \rceil - \lceil \#C/2^{L-(\ell-1)} \rceil$.

Door de verzamelingen voor elk level op deze manier te bepalen wordt er een levelstructuur bekomen waarbij (i) de bijdrage van de levels vermindert als $\ell \to L$ waardoor het algoritme vroegtijdig gestopt kan worden omdat vanaf een bepaald level k de bijdrage lager is dan de gewenste tolerantie ε: $\sum_{\ell=k}^{L} \mathbb{E}[T_\ell - T_{\ell-1}] \ll \varepsilon$ (ii) de variantie neemt af per level waardoor de kost voor Multilevel Monte Carlo lager is dan MC en (iii) level 0 heeft de deelverzameling met $\lceil \#C/2^L \rceil$ snedes die het best de gemiddelde tijd tot aan falen benadert van het systeem.

4.2 Algoritme

Het algoritme dat gebruikt wordt voor het bepalen van de gemiddelde tijd tot aan falen in systemen is de gegeven in Algoritme 1. Hierbij is een stopcriterium toegevoegd om de simulatie te optimaliseren in het geval het aantal te simuleren levels niet noodzakelijk het aantal levels is dat gegenereerd is door het selectie-algoritme uit sectie 4.1. Stel dat het selectie-algoritme voor een systeem L levels genereert en er bestaat een level k waarvoor geldt dat $\sum_{\ell=k}^{L} Y_\ell < \varepsilon/2$, dan kan de simulatie stoppen

bij level k omdat de afwijking door het weglaten van de levels $k, ..., L$ voldoende klein is. Dit stopcriterium is de test voor zwakke convergentie en is de afwijking van de berekening als er niet gesimuleerd wordt op alle levels (bias).

Algoritme 1: Multilevel Monte Carlo

Data: nauwkeurigheid ε , levelstructuur, maximum aantal levels L
Result: $\hat{T}$
$\hat{L} = 2$;
$N_\ell :=$ 100 samples op level $\ell = 0, 1, 2$;
bepaal Y_ℓ met vergelijking 3.16 op level $\ell = 0, 1, 2$;
schat varianties V_ℓ en kost C_ℓ voor elk level $0, 1, 2$;
bepaal $\hat{N}_\ell$ voor $\ell = 0, 1, 2$ volgens vergelijking 4.6;
while *bias* $> \varepsilon/2$ *&* $\hat{L} < L$ **do**
  while *geen convergentie* ($N_\ell < 0.01 \cdot \hat{N}_\ell$ *voor* $\ell = 0, 1, \ldots, \hat{L}$) **do**
    $N_\ell = \max\{N_\ell, \hat{N}_\ell\}$ voor elke level $\ell = 0, 1, \ldots, \hat{L}$;
    bereken bijkomende simulaties op ieder level $\ell = 0, 1, \ldots, \hat{L}$;
    bepaal Y_ℓ, V_ℓ, C_ℓ voor elk level $\ell = 0, 1, \ldots, \hat{L}$;
    bereken het optimale aantal simulaties $\hat{N}_\ell$ (vergelijking 4.6) voor elk level $\ell = 0, 1, \ldots, \hat{L}$;
  $\hat{L} = \hat{L} + 1$;
  $V_{\hat{L}} = V_{\hat{L}-1}/2$;
return $\hat{T} = \sum_{\ell=0}^{\hat{L}} Y_\ell$;

Het optimale aantal simulaties $\hat{N}_\ell$ per level ℓ wordt berekend met

$$\hat{N}_\ell = \left\lceil 2\varepsilon^{-2}\sqrt{V_\ell C_\ell}\left(\sum_{\ell=0}^{\hat{L}} \sqrt{V_\ell/C_\ell}\right)\right\rceil. \tag{4.6}$$

dit samen met de keuze van het maximum level $\hat{L}$ zorgt ervoor dat de gemiddelde kwadratische fout lager is dan ε^2 met ε de opgegeven nauwkeurigheid. Het aantal extra simulaties dat berekend wordt, is het verschil tussen de optimale N_ℓ en de het reeds berekende aantal simulaties. Als dit verschil positief is, worden er simulaties uitgevoerd en wordt de schatting voor V_ℓ aangepast.
De test voor convergentie, $N_\ell < 0.01 \cdot \hat{N}_\ell$, zorgt ervoor dat de gemiddelde kwadratische fout door de variantie geconvergeerd is naar $\varepsilon^2/2$ [4]. De test voor zwakke convergentie zorgt ervoor dat $\left|\, \mathbb{E}[T - T_L] \,\right| < \varepsilon/\sqrt{2}$, wat resulteert in een gemiddelde kwadratische fout kleiner dan ε^2 met ε de opgegeven nauwkeurigheid van de simulatie.
Het theorema van Multilevel Monte Carlo (theorema 1) veronderstelt $\mathbb{E}[T_\ell - T_{\ell-1}] \propto 2^{-\alpha\ell}$, hierdoor kan de afwijking van de berekening als het aantal gebruikte levels kleiner is dan het aantal levels uit het selectie-algoritme bepaald worden. De afwijking van de berekening is gelijk aan

$$\mathbb{E}[T - T_L] = \sum_{\ell=L+1}^{\infty} \mathbb{E}[T_\ell - T_{\ell-1}] = \mathbb{E}[T_L - T_{L-1}]/(2^\alpha - 1). \tag{4.7}$$

Op basis van α kan de afwijking geschat worden, de test voor zwakke convergentie wordt dus gegeven door $\left| \mathbb{E}[T_{\hat{L}} - T_{\hat{L}-1}] \right| / (2^{\alpha} - 1) < \varepsilon/\sqrt{2}$ [10].

4.3 Quasi-Monte Carlo

Zoals besproken in hoofdstuk 3 wordt in het geval van Multilevel Quasi-Monte Carlo de benadering Y_ℓ op ieder level bepaald door M reeksen van N_ℓ simulaties waarbij iedere reeks een verschillende verschuiving is ondergaan door een willekeurige vector Δ. De M reeksen op ieder level zorgen voor M gemiddelde waarden voor de gezochte waarde Y_ℓ en door de willekeurige verschuiving kan er een zuivere schatting gedaan worden van de variantie op basis van deze gemiddelde waarden.
De gemiddelde kwadratische fout wordt bepaald door de som van de varianties op ieder level $\sum_{\ell=0}^{L} V_\ell$ en de afwijking $\mathbb{E}[T - \hat{T}_{\hat{L}}]$. De afwijking wordt bepaald door het grootste level $\hat{L}$ en is enkel aanwezig als men niet simuleert over alle levels uit de levelstructuur [27]. Het algoritme voor Multilevel Quasi-Monte Carlo is gegeven in algoritme 2.

Algoritme 2: Multilevel Quasi-Monte Carlo

Data: nauwkeurigheid ε , levelstructuur, maximum aantal levels L
Result: $\hat{T}$
$\hat{L} = 2$;
$N_\ell :=$ 10 reeksen van 100 samples op level $\ell = 0, 1, 2$;
bepaal Y_ℓ met vergelijking 3.16 op level $\ell = 0, 1, 2$;
schat varianties V_ℓ en kost C_ℓ voor elk level $0, 1, 2$;
while *bias* $> \varepsilon/2$ *&* $\hat{L} < L$ **do**
  while *geen convergentie* $(\sum_{\ell=0}^{L} V_\ell > \frac{\varepsilon^2}{2})$ **do**
    verhoog N_k op het level $k = \arg\max_\ell \frac{V_\ell}{C_\ell N_\ell}$;
    bereken bijkomende simulaties op ieder level $\ell = 0, ..., \hat{L}$;
    bepaal Y_ℓ en schat V_ℓ en C_ℓ voor elk level $\ell = 0, ..., \hat{L}$;
  $\hat{L} = \hat{L} + 1$;
  $N_{\hat{L}} = 100$;
return $\hat{T} = \sum_{\ell=0}^{\hat{L}} Y_\ell$;

In dit algoritme wordt het aantal simulaties verhoogd in de levels waar de variantie groot is in vergelijking met de rekenkost die nodig is om deze simulaties te berekenen. De verhoging van het aantal simulaties zorgt voor een verlaging van de variantie en dus een verlaging van de gemiddelde kwadratische fout [10].
De Quasi-Monte Carlo punten van de ingangsvariabelen worden bepaald door een rang-1 roosterregel te gebruiken. In dit onderzoek is een 3600-dimensionale uitbreidbare roosterregel gebruikt die wordt opgebouwd met het component-per-component algoritme uit [20]. De genererende vector is afkomstig van [15].
De Quasi-Monte Carlo punten in hoge dimensies worden meer uniform verdeeld in de lagere dimensies. Voor de meest efficiënte implementatie wordt er een herordening gedaan van de componenten voor de berekening van de levensduur [19]. Deze her-

ordening gebeurt op basis van sensitiviteitscoëfficiënten en dit wordt in sectie 4.3.1 besproken.

4.3.1 Rangorde

De herordening van de ingangsvariabelen bij een Quasi-Monte Carlo berekening kan er voor zorgen dat de verdeling van de levensduur van de belangrijkste componenten meer uniform is. In de context van de betrouwbaarheid van systemen is de uitgang te schrijven als: $T = f(t_1, t_2, ..., t_n)$. Hier is t_c de levensduur van component c voor $c = 1, 2, \ldots, n$ en T de levensduur van het systeem. De herordening heeft als doel om de variantie zo ver modelijk te reduceren, hiervoor kunnen 2 sensitiviteitsociëfficiënten gebruikt worden, nl. de eerste orde sensitiviteit S of de totale orde sensitiviteit TOC. De eerste orde sensitiviteit wordt gegeven door

$$S_j = \frac{\mathbb{V}_{t_j}[\mathbb{E}_{T_{-j}}[T|t_j]]}{\mathbb{V}[T]}.$$

Waarbij $\mathbb{E}_{T_{-j}}[T|t_j]$ de verwachte waarde is genomen over alle ingangsvariabelen buiten t_j. S_j kan geschat worden door een curve op te stellen voor $\mathbb{E}_{T_{-j}}(T|t_j)$ (T versus t_j) en de variantie te bepalen van deze curve over t_j. S_j kan beschreven worden als de fractie van reductie in de variantie van T als t_j constant gehouden wordt.
De totale orde sensitiviteitcoëfficiënt, TOC_j, is te schrijven als

$$TOC_j = \frac{\mathbb{E}_{T_{-j}}[\mathbb{V}_{T_j}[T|T_{-j}]]}{\mathbb{V}[T]}.$$

In woorden betekent dit dat TOC_j de fractie is van de variantie veroorzaakt door t_j als al de andere factoren constant gehouden wordt.
De herordening is gebaseerd op deze coëfficiënt, de totale orde sensitiviteit, deze wordt geschat op basis van willekeurig gegenereerde getallen met het algoritme in [23]. De ordening gaat als volgt:

1. Construeer een $N \times 2k$ QMC sequentie met N het aantal samples en k de dimensie van de input (het aantal componenten). Hier wordt een Sobol' sequentie gegenereerd door gebruik te maken van de Sobol sequentie generator van S. Joe en F. Kuo [13] [14].

2. De matrix wordt opgesplitst in 2 matrices: $N \times k$ matrix A en $N \times k$ matrix B.

3. De waarden komen van een uniforme verdeling. Deze wordt omgevormd van een Uniform(0,1) naar een Weibull(k,λ) verdeling specifiek voor het component die overeenkomt met dimensie i. De omvorming is volgens

$$y = -\lambda^k \ln(1 - x)^{\frac{1}{k}}. \tag{4.8}$$

4. Maak k extra matrices $A_B^{(j)}$ voor de benadering van TOC_j (kolom j is van matrix B en de rest van matrix A). Het verschil tussen matrix A en matrix $A_B^{(j)}$ is dan enkel een verandering van de levensduur t_j.

5. Bereken de levensduur van het systeem: $f(a_i)$ en $f(a_b^{(j)})$.

6. Bepaal de benadering voor

$$TOC_j = \frac{\mathbb{E}_{T_{-j}}[\mathbb{V}_{T_j}[T|T_{-j}]]}{\mathbb{V}[T]} \approx \frac{1}{2N\mathbb{V}[T]} \sum_{i=1}^{N} (f(a_i) - f(a_b^{(j)}))^2.$$

7. Rangschik de componenten volgens dalende waarden van TOC_j.

Het algoritme maakt gebruik van Sobol' sequenties die gegenereerd worden in een andere programmeertaal en omgevormd moeten worden van een eenheidshyperkubus naar de verdeling van de levensduur van de componenten. Hierdoor is er gekeken naar een vereenvoudiging van het algoritme die geen gebruik maakt van Sobol' reeksen. In de vereenvoudiging worden de matrices A en B geconstrueerd als (k x N) matrices het k het aantal componenten en N het aantal samples. Deze waarden worden willekeurig gegenereerd uit de verdelingen van de levensduur van de componenten. Iedere kolom bevat dus een simulatie van de levensduur van elke component. Door deze vereenvoudiging te gebruiken kunnen de simulaties die gebruikt zijn in de constructie van de levelstructuur hergebruikt worden voor de rangorde te bepalen.

4.4 Uitbreiding

4.4.1 gebruik van minder levels

M. Giles vermeldt in [10] dat het niet nodig is om een geometrische levelstructuur te hebben. Bij Multilevel Monte Carlo is het enkel nodig dat de nauwkeurigheid en de kost stijgt in elk level en de variantie van het verschil afneemt. In bepaalde systemen is het mogelijk dat de selectie van de levels deze afname niet vertoont. Hierbij kan het voordelig zijn om de levelstructuur aan te passen. Is het aantal levels niet te groot dan kan dit gebeuren via exhaustief zoeken naar een optimale levelstructuur. Een methode die gebruikt kan worden als er een levelstructuur aanwezig is, is het bepalen of een bepaald level ℓ geen significante bijdrage heeft aan de multilevel Monte Carlo simulatie. De efficiëntie kan dan verhoogd worden door dit level te verwijderen. De bijdrage van level ℓ aan de totale Multilevel Monte Carlo simulatie wordt bepaald door de correctieterm in level ℓ en level $\ell + 1$:

$$\frac{1}{N_\ell} \sum_{n=1}^{N_\ell} (T_\ell^{(n)} - T_{\ell-1}^{(n)}) + \frac{1}{N_{\ell+1}} \sum_{n=1}^{N_{\ell+1}} (T_{\ell+1}^{(n)} - T_\ell^{(n)}). \tag{4.9}$$

In het optimale geval is N_ℓ proportioneel met $\sqrt{V_\ell/C_\ell}$ met $V_\ell = \mathbb{V}[T_\ell - T_{\ell-1}]$ en C_ℓ de kost voor het berekenen van een correctie $[T_\ell - T_{\ell-1}]$. Door de verhouding van het aantal simulaties op level ℓ en level $\ell + 1$ te nemen kan geschreven worden dat

$$N_\ell = N_{\ell+1} \sqrt{\frac{V_\ell}{V_{\ell+1}} \frac{C_{\ell+1}}{C_\ell}}.$$

De variantie en de kost die beïnvloed wordt door level ℓ is

$$\frac{1}{N_\ell}V_\ell + \frac{1}{N_{\ell+1}}V_{\ell+1} = \frac{V_{\ell+1}}{N_{\ell+1}}\left(1+\sqrt{\frac{V_\ell C_\ell}{\mathbb{V}_{\ell+1}C_{\ell+1}}}\right), \tag{4.10}$$

$$N_\ell C_\ell + N_{\ell+1}C_{\ell+1} = N_{\ell+1}C_{\ell+1}\left(1+\sqrt{\frac{V_\ell C_\ell}{V_{\ell+1}C_{\ell+1}}}\right). \tag{4.11}$$

Het product van de variantie en de kost van de 2 termen is

$$\mathbb{V}_{\ell,\ell+1}C_{\ell,\ell+1} = V_{\ell+1}C_{\ell+1}\left(1+\sqrt{\frac{V_\ell C_\ell}{V_{\ell+1}C_{\ell+1}}}\right)^2 \tag{4.12}$$

Als level ℓ uit de levelstructuur gehaald wordt dan worden de twee termen vervangen door 1 term:

$$\frac{1}{\hat{N}_{\ell+1}}\sum_{n=1}^{\hat{N}_{\ell+1}}\left(T_{\ell+1}^{(n)} - T_{\ell-1}^{(n)}\right).$$

Het product van de variantie en de kost wordt gegeven door $\hat{V}_{\ell+1}\hat{C}_{\ell+1}$. Als dit product kleiner is dan het product van de variantie en kost van de twee termen dan is het voordelig om level ℓ uit de levelstructuur te halen. De kostterm in beide gevallen wordt voornamelijk bepaald door de simulatie op level $\ell+1$. We kunnen dus aannemen dat $\hat{C}_{\ell+1} \approx C_{\ell+1}$, de ratio's $C_\ell/C_{\ell+1}$ en $V_\ell/V_{\ell+1}$ zijn gekend of kunnen empirisch bepaald worden. De hoofdvraag hier is hoe $\hat{V}_{\ell+1}$ zich verhoudt tot $V_{\ell+1}$. Dit kan geschat worden door een ruwe Monte Carlo simulatie uit te voeren. De implementatie gebeurt zoals in algoritme 3.

Algoritme 3: Verwijderen van niet-significante levels

```
Data: nauwkeurigheid ε, levelstructuur, maximum aantal levels L
Doe een ruwe MC simulatie op ieder level ℓ = 0, ..., L;
for ℓ = 1 : L do
    bepaal V_ℓ, C_ℓ, V_{ℓ+1}, C_{ℓ+1} ;
    bepaal variantie V̂_{ℓ+1} ;
    if V̂_{ℓ+1} < V_{ℓ+1}(1 + sqrt(V_ℓ C_ℓ / (V_{ℓ+1} C_{ℓ+1})))^2 then
        verwijder level ℓ;
```

4.4.2 Andere verdelingen

Het onderzoek is uitgevoerd op systemen met componenten waarvan de levensduur bepaald wordt door een Weibull-verdeling. In realiteit is er echter geen beperking op welke verdeling de levensduur heeft. Een speciaal geval waar aandacht aan gegeven moet worden is wanneer alle componenten dezelfde verdeling hebben. In deze situatie is het niet duidelijk welke snedes tot welke levels behoren. Door het selectie-algoritme dat hier gebruikt wordt, is er een verdubbeling van het aantal snedes per level waardoor de kost in ieder level verdubbeld wordt. Dit zorgt ook dat de verwachte waarde van de correctieterm en variantie van de correctieterm daalt voor $\ell \to L$ [2]. De multilevel Monte Carlo implementatie is dus robuust voor dit geval.

Exponentiële verdeling

De kansverdeling van de levensduur die bepaald wordt door een exponentiële verdeling is gegeven door

$$f(x;\lambda) = \lambda e^{-\lambda x}. \tag{4.13}$$

$\lambda > 0$ is hier de intensiteitsparameter. Als de parameter $\mu = \frac{1}{\lambda}$ wordt geïntroduceerd dan verandert de exponentiële verdeling in

$$f(x;\lambda) = \frac{1}{\mu} e^{-(x/\mu)} = \frac{1}{\mu}(\frac{x}{\mu})^0 e^{-(x/\mu)}. \tag{4.14}$$

Dit komt overeen met een Weibull verdeling met schaalparameter $\mu = \frac{1}{\lambda}$ en vormparameter $k = 1$. Het gebruik van de exponentiële verdeling voor de levensduur van de componenten geeft daarom geen moeilijkheden.

Uniforme verdeling

Door de kansverdeling van de levensduur van de componenten te wijzigen naar een uniforme verdeling worden er geen moeilijkheden verwacht. Bij de theorie van Multilevel Monte Carlo, de bespreking van het selectie-algoritme en het algoritme voor de Multilevel Monte Carlo wordt er geen voorwaarden opgelegd voor de verdeling van de ingangsvariabelen. Er wordt dus verwacht dat de berekening robuust is voor een verandering van de verdelingen bij de ingangsvariabelen.

4.4.3 Onderzoeken van de kansverdeling van de levensduur van het systeem

In bepaalde gevallen is de kansverdeling van de levensduur gewenst in plaats van een schatting van de levensduur. De manier die hier voorgesteld wordt, is een experimentele methode voor het opstellen van de waarschijnlijkheidsdichtheid. Er wordt op ieder level een willekeurige simulatie genomen, op level $\ell = 0$ is dit $T_0^{(i)}$ en op level $\ell > 0$ is dit $[T_\ell^{(i)} - T_{\ell-1}^{(i)}]$. Deze willekeurig genomen simulaties worden opgeteld tot de levensduur van het systeem $T^{(i)} = T_0^{(i)} + \sum_{\ell=1}^{L}[T_\ell^{(i)} - T_{\ell-1}^{(i)}]$. Dit is één simulatie van de totale levensduur. Deze wordt opgeslagen in een vector $\mathbf{T}$. Dit wordt N keer herhaald en vervolgens wordt de histogram van deze vector $\mathbf{T}$ gegenereerd. Negatieve waarden voor de levensduur hebben geen betekenis en worden weggelaten. De histogram geeft een weergave van de kansverdeling van de levensduur van het systeem [2].

4.5 Schatter

Voor de Multilevel (Quasi-)Monte Carlo berekening is er gebruik gemaakt van het pakket `MultilevelEstimators` van ir. Pieterjan Robbe [22]. Dit pakket bevat de implementatie van de Multilevel Monte Carlo methode en de Multilevel Quasi-Monte Carlo methode zoals hierboven vermeld. Het pakket `MultilevelEstimators` bezit

een type `Estimator` met twee parameters, dit is de schatter voor de simulatie. De parameters zijn de index set en de sample methode. Voor de implementatie in de context van de masterproef wordt er gebruik gemaakt van de index set: `ML()`. Er worden twee sample methodes gebruikt: `MC()` voor Monte-Carlo simulaties en `QMC()` voor Quasi-Monte Carlo simulaties.
Om de schatter te gebruiken moet er een simulatiefunctie worden opgesteld met als inputwaarden een level en een vector met de waarden voor de ingangsvariabelen, de componenten van het systeem, en als uitgangswaarden de waarde van de simulatie en de correctieterm op het level (`sample_levensduur(Level,`ω`)`). Bijkomend vraagt de schatter een vector van de verdelingen van de ingangsvariabelen, de componenten (`distributions`). De schatter wordt gedefinieerd met de code

```
estimator= Estimator(ML(),MC(),sample_levensduur,distributions)
```

Er zijn verschillende opties en instellingen mogelijk die bijgevoegd kunnen worden aan deze toekenning, voor een gedetailleerde beschrijving wordt er verwezen naar [22]. De berekening wordt uitgevoerd door het commando `run(estimator,`ε`)` met ε de nauwkeurigheid. Een algemene simulatiefunctie wordt gegeven door

```
function sample_levensduur(level,w,systeem,levels)
        # levensduur op het level
        lijst = systeem.lijst #lijst met de componenten
        sampleLevensduur(lijst,w);
        Qf = levensduur(levels[level]);
        # correctieterm
        dQ = Qf
        if level != 1
                Qc = levensduur(levels[level-1]);
                dQ -= Qc
        end
        return dQ, Qf
end
systeem = System(naam,Structuur,nodes)
levels = selectLevel(systeem)
sample_levensduur(level,w) = sample_levensduur(level,w, ...
                             systeem,levels)
```

De functies `sampleLevensduur(lijst,w)` kent de levensduur uit `w` toe aan de componenten. De functie `levensduur(levels[level])` berekent de levensduur van dit level.

Hoofdstuk 5

Resultaten

In dit hoofdstuk worden de resultaten besproken die bekomen zijn met de algoritmen en de schatter uit hoofdstuk 4. De berekening is eerst uitgevoerd op willekeurig gegenereerde systemen waarvan de componenten niet herstelbaar zijn (sectie 5.1). De analyse is gericht op de rekenkost van de Multilevel Monte Carlo methode en de Multilevel Quasi-Monte Carlo methode. Er wordt verder aandacht besteed aan de uitbreidingen. Er worden resultaten gegeven van de kansverdeling van het systeem en andere kansverdelingen worden onderzocht. Vervolgens wordt er in sectie 5.2 overgegaan naar systemen met herstelbare componenten. De focus ligt hier weer op de verlaagde rekenkost van Multilevel Monte Carlo. In sectie 5.3 worden enkele toepassingen besproken. De eerste toepassing is het brandalarm vermeld in hoofdstuk 2. De tweede toepassing is de waterkrachtcentrale uit [9], een artikel van George-Williams et al.

5.1 Niet-herstelbare systemen

De systemen die gebruikt zijn voor het testen van de methode zijn willekeurig gegenereerd. De systemen worden opgebouwd door te starten met een systeem met één component en vervolgens een willekeurig component te vervangen door twee componenten in serie, twee componenten in parallel of een verbinding toe te voegen tussen twee componenten. Dit gebeurt met een vaste kans per systeem.
Het selectie-algoritme uit sectie 4.1 wordt toegepast. Dit geeft de levelstructuur die gebruikt wordt in de Multilevel (Quasi-)Monte Carlo schatter. Vervolgens wordt de MLMC schatter gedefinieerd en de berekening wordt uitgevoerd met een opgegeven nauwkeurigheid ε.
Secties 5.1.1 tot 5.1.3 geven de resultaten van een systeem met respectievelijk 20 componenten, 30 componenten en 50 componenten. Voor ieder systeem wordt gekeken naar de berekende levensduur, de RMSE (de Root Mean Square Error of de standaardfout) en de ratio RMSE/ε wordt berekend. Het algoritme werkt correct als deze ratio lager is dan 1 [4]. Telkens is er een diagnose van de berekening waarbij de voorwaarden van het theorema van Multilevel Monte Carlo (theorema 1) worden bekeken. In deze diagnose wordt er gekeken naar (i) de verwachte waarde van de

benadering en de correctieterm in ieder level, (ii) de variantie van de benadering en de correctieterm in ieder level, (iii) de kost van één simulatie in de levels en (iv) het aantal simulaties per level. De diagnose geeft een schatting voor de parameters $\alpha,\ \beta,\ \gamma$ in het theorema 1.
Voor elk systeem is er een vergelijking gemaakt tussen de rekenkost van Monte Carlo, Multilevel Monte Carlo en Multilevel Quasi-Monte Carlo. De rekenkost van de Monte Carlo simulatie wordt geschat door het aantal simulaties op level $\ell = 0$ vermenigvuldigd met de kost van een simulatie op het hoogste level $\ell = L$, dit level berekent de levensduur van het systeem,

$$\text{cost}_{MC} = N_0 C_L. \tag{5.1}$$

De MSE van de Monte Carlo methode wordt bepaald door de variantie van de schatter. Formule 3.8 geeft het verband tussen het aantal simulaties en deze fout. Hieruit volgt dat er verwacht wordt dat de rekenkost bij de Monte Carlo methode verloopt volgens $\mathcal{O}(\varepsilon^{-2})$. Waarbij ε de opgegeven nauwkeurigheid is voor de standaardfout. De rekenkost van de Multilevel Monte Carlo simulatie wordt gegeven door

$$\text{cost}_{MLMC} = \sum_{\ell=0}^{L} C_\ell N_\ell. \tag{5.2}$$

Volgens theorema 1 verloopt de rekenkost van Multilevel Monte Carlo in het geval dat $\beta > \gamma$ volgens $\mathcal{O}(\varepsilon^{-2})$. In de vergelijking van de rekenkost tussen Monte Carlo en Multilevel Monte Carlo wordt dezelfde orde verwacht in relatie tot de opgegeven nauwkeurigheid waarbij de rekenkost van MLMC lager is dan de rekenkost can MC. De rekenkost van de Multilevel Quasi-Monte Carlo simulatie is gegeven door

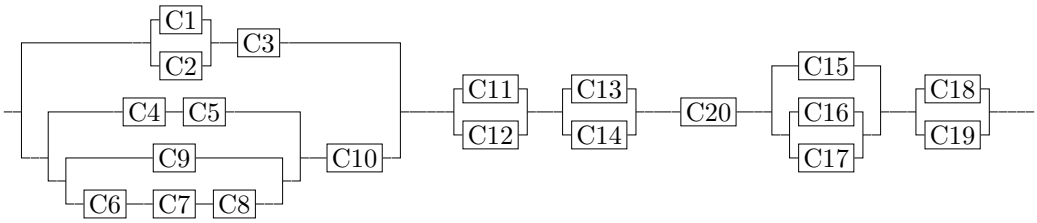
$$\text{cost}_{MLQMC} = \sum_{\ell=0}^{L} MC_\ell N_\ell \tag{5.3}$$

waarbij M het aantal willekeurige verschuivingen zijn van de Quasi-Monte Carlo punten in de berekening. In hoofdstuk 3 werd aangehaald dat de complexiteit van de rekenkost hier $\mathcal{O}(\varepsilon^{-p})$ met $p < 2$ is. We verwachten dit ook te zien in de resultaten.

5.1.1 Systeem met 20 componenten

Het eerste systeem is een systeem met 20 verschillende componenten die een levensduur hebben volgens een Weibull-verdeling met een vormparameter k en een schaalparameter λ. Het systeem wordt bekeken in 3 verschillende situaties. De situaties verschillen in de vormparameter ($k = 0.5, 1, 3$). In deze sectie wordt de berekening besproken waarbij verdeling van de levensduur van de componenten een vormparameter $k = 1$ hebben. Voor de bespreking van de berekeningen met de andere vormparameter wordt er verwezen naar de bijlagen. De schaalparameter is een willekeurig getal uit een uniforme verdeling tussen 2 en 10.
Het systeem is gegenereerd door een systeem met één component te definiëren en dit systemen uit te breiden. De uitbreiding gebeurt door een willekeurig component

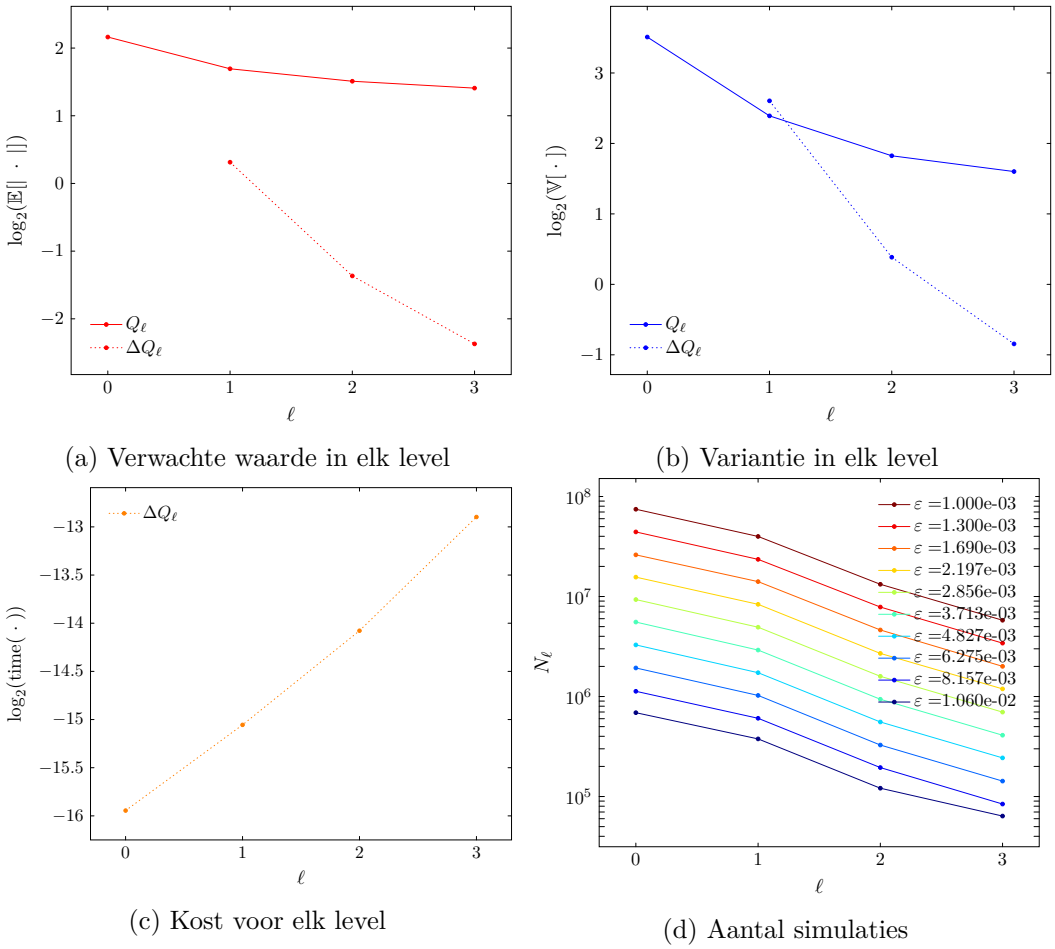
te vervangen door een serieschakeling of parallelschakeling van 2 componenten. Het betrouwbaarheidsblokdiagram van dit systeem is weergegeven in figuur 5.1.
Dit systeem heeft een verzameling van minimale snedes met 19 snedes. Het selectie-algoritme levels geeft 4 levels: level 0 bevat 3 snedes, level 1 heeft 5 snedes, level 2 heeft 10 snedes, level 3 heeft 19 snedes. De grootte van de snede wordt in ieder level verdubbeld, dit geeft de meest efficiënte implementatie [2].



Figuur 5.1: Systeem met 20 componenten

Simulatie

De berekening is uitgevoerd met de Multilevel Monte Carlo schatter gedefinieerd in sectie 4.5 met een opgegeven nauwkeurigheid $\varepsilon = 10^{-3}$. De schatter geeft een volledige diagnose van de berekening. Figuur 5.2 geeft de verwachte waarden, de varianties, de kost en het aantal simulaties dat is uitgevoerd op ieder level. De berekende levensduur van dit systeem is $\hat{T} = 2.6543$ met een variantie $\mathbb{V} = 4.99 \cdot 10^{-7}$. De fout van de simulatie is bepaald door de variantie en de afwijking van de schatter maar omdat er gesimuleerd is op alle levels is de schatter hier zuiver. De fout wordt dus enkel bepaald door de variantie. De gemiddelde kwadratische fout is $4.99 \cdot 10^{-7}$, de ratio RMSE/ε is $7.07 \cdot 10^{-4}/10^{-3} = 0.707$. De schatter geeft dus een voldoende nauwkeurig resultaat.
Figuur 5.2a geeft de verwachte waarde op ieder level $\mathbb{E}[\hat{T}_\ell]$ en de verwachte waarde van de correctieterm $\mathbb{E}[\hat{T}_\ell - \hat{T}_{\ell-1}]$. De correctieterm daalt zoals verwacht en deze daling heeft een geschatte parameter $\alpha = 1.34262$. Figuur 5.2b geeft de variantie van de levensduur $\mathbb{V}[\hat{T}_\ell]$ en de correctieterm op ieder level $\mathbb{V}[\hat{T}_\ell - \hat{T}_{\ell-1}]$. De parameter β van de daling van de variantie van de correctieterm wordt geschat op $\beta = 1.72561$. De kost van de simulatie wordt gegeven in figuur 5.2c. De kost per level stijgt, de parameter γ heeft een geschatte waarde $\gamma = 1.07959$. Dit is zoals verwacht wordt aangezien het aantal snedes verdubbelt in ieder level, wat de kost zal verdubbelen. De parameter van de afname van de variantie is groter dan de parameter van de stijging van de kost. Dit zorgt voor een ideale situatie voor Multilevel Monte Carlo: de variantie daalt sneller dan de kost stijgt, de rekenkost wordt dus het meest bepaald door de lage levels. Het aantal simulaties uitgevoerd op ieder level is gegeven in figuur 5.2d. Het aantal uitgevoerde simulaties is niet enkel weergegeven voor de opgegeven nauwkeurigheid maar voor 10 toleranties die een factor 1.3 verschillen. Dit geeft de mogelijkheid om het gedrag van de rekenkost te bestuderen in relatie met de nauwkeurigheid (figuur 5.4). Op de figuur is duidelijk te zien dat de meeste



(a) Verwachte waarde in elk level

(b) Variantie in elk level

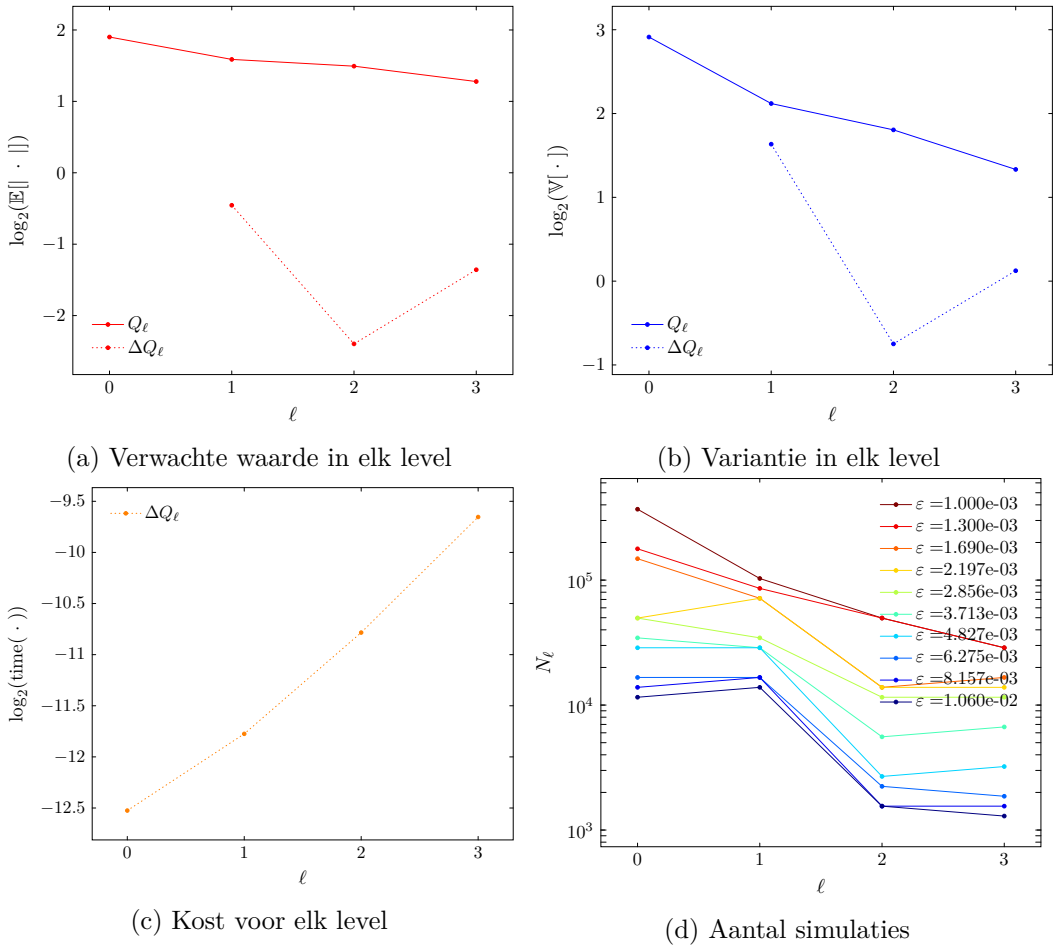
(c) Kost voor elk level

(d) Aantal simulaties

Figuur 5.2: 20 componenten, diagnose van de Multilevel Monte Carlo berekening

simulaties gebeuren op het laagste level en het aantal simulaties lager is bij de hogere levels.

De berekening is herhaald met een Multilevel Quasi-Monte Carlo schatter. Deze schatter heeft nog een extra parameter, het aantal shifts. Voor dit systeem zijn er 10 shifts gebruikt. De figuren van de diagnose zijn gegeven in figuur 5.3 waarop hetzelfde gedrag gezien wordt. De daling van de variantie ($\beta = 0.752889$) is bij deze berekening niet groter dan de stijging van de kost ($\gamma = 1.06423$). In figuur 5.3b wordt de variantie weergegeven in ieder level, hier is een stijging te zien tussen level $\ell = 2$ en level $\ell = 3$. Dit zorgt er voor dat meer simulaties nodig zijn op level $\ell = 3$. Figuur 5.3d toont het aantal simulaties in ieder level. De daling van het aantal simulaties per level is minder sterk als bij Multilevel Monte Carlo. Bij enkele nauwkeurigheden is er zelfs een stijging is bij het laatste level. Bij een vergelijking met figuur 5.2d wordt er wel opgemerkt dat er veel minder simulaties nodig zijn om de opgegeven nauwkeurigheid te bereiken.



(a) Verwachte waarde in elk level

(b) Variantie in elk level

(c) Kost voor elk level

(d) Aantal simulaties

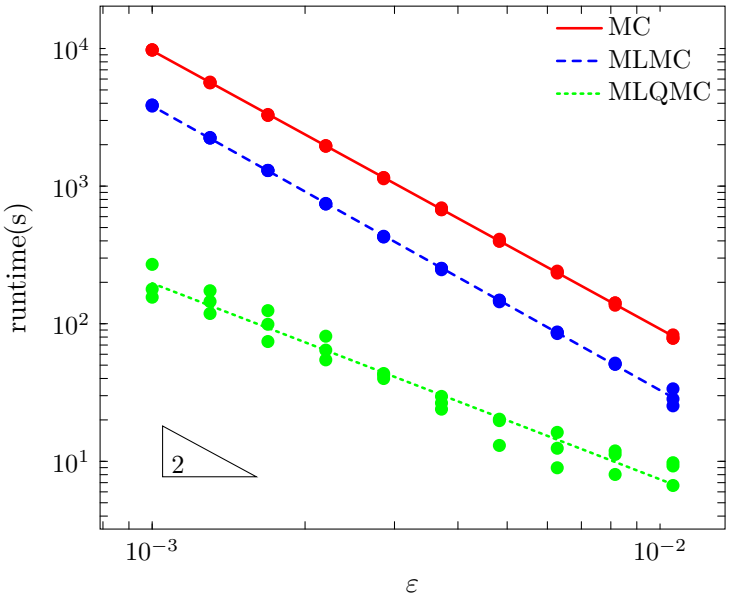
Figuur 5.3: 20 componenten, diagnose van de Multilevel Quasi-Monte Carlo berekening

De rekenkost is bepaald voor 10 toleranties met als kleinste nauwkeurigheid $\varepsilon = 10^{-3}$. Het verschil van twee opeenvolgende toleranties is een factor 1.3. De vergelijking is weergegeven in figuur 5.4. De punten zijn de gemeten rekenkost, de lijnen geven de regressie over 3 berekeningen. De rekenkost van de Monte Carlo simulatie wordt geschat door het aantal simulaties op level $\ell = 0$ vermenigvuldigd met de kost van een simulatie op het hoogste level $\ell = 3$. De rekenkost van de Multilevel Monte Carlo simulatie wordt gegeven door

$$\sum_{\ell=0}^{3} C_\ell N_\ell. \tag{5.4}$$

Op deze figuur is te zien dat de rekenkost voor Multilevel Monte Carlo lager ligt dan de rekenkost voor Monte Carlo. De ratio van de rekenkost van Multilevel Monte Carlo over de rekenkost van Monte Carlo ligt binnen het bereik 0.32-0.41. Dit betekent dat het gebruik van Multilevel Monte Carlo de berekening 2.5 tot 3 keer versnelt. Door gebruik te maken van Quasi-Monte Carlo is er nog een bijkomende reductie van

de rekenkost. Hier is een versnelling zichtbaar die oploopt tot 50 keer de rekenkost bij een tolerantie $\varepsilon = 10^{-3}$. Door de regressie kan de complexiteit geschat worden, voor van de Monte Carlo simulatie en de Multilevel Monte Carlo simulatie is deze $\mathcal{O}(\varepsilon^{-2})$. De gemeten complexiteit van de Multilevel Quasi-Monte Carlo is hier $\mathcal{O}(\varepsilon^{-1.42})$.



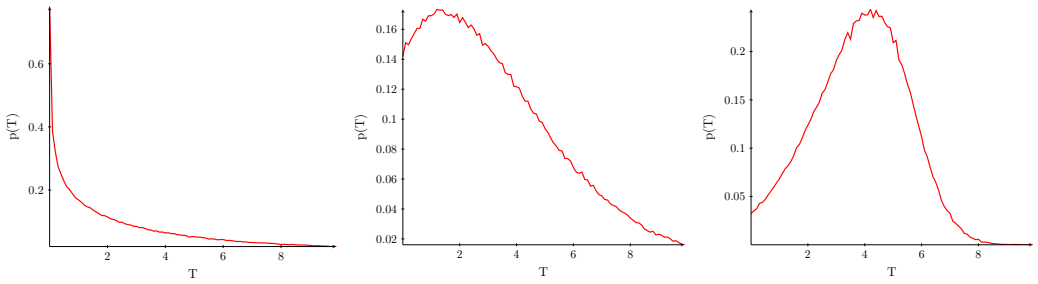
Figuur 5.4: 20 componenten, rekenkost van de Monte Carlo (MC), Multilevel Monte Carlo (MLMC) en Multilevel Quasi-Monte Carlo (MLQMC)

Invloed componenten

De simulatie is herhaald voor hetzelfde systeem waarbij de componenten een andere vormparameter hebben (Bijlage A.1). Dit geeft de mogelijkheid om de invloed van de verdeling van de levensduur van de componenten te analyseren. De rekenkost voor hetzelfde systeem met een hogere vormparameter is lager terwijl de rekenkost voor hetzelfde systeem met een lagere vormparameter een hogere rekenkost heeft. Dit kunnen we verklaren door naar de variantie van de levensduur van het systeem te kijken in de diagnose. De variantie van de levensduur van dit systeem met een vormparameter $k = 1$ is volgens figuur 5.2b (of figuur 5.3b) $\mathbb{V}[\hat{T}_s] = 19.338$. De variantie van dit systeem met een vormparameter $k = 3$ is $\mathbb{V}[\hat{T}_s] = 4.449$. Dit komt overeen met de theorie. Volgens formule 3.8 is de MSE van de berekening afhankelijk van de variantie en het aantal simulaties uitgevoerd. Als de variantie op de levensduur groter is moeten er dus meer simulaties uitgevoerd worden om dezelfde MSE te bekomen.

Kansverdeling

In sectie 4.4.3 wordt de uitbreiding vermeld om de kansverdeling van de levensduur van het systeem weer te geven. Figuur 5.5 geeft een waarschijnlijkheidsdichtheid van de verdeling van de levensduur van het systeem met 20 componenten. Deze figuur geeft de kansverdeling voor het systeem met vormparameter $k = 0.5$ (figuur 5.5a), $k = 1$ (figuur 5.5b) en $k = 3$ (figuur 5.5c). Er is een duidelijk verschil tussen de drie situaties waarbij er de kans op een langere levensduur hoger is bij een hogere vormparameter. Door de individuele componenten te verbeteren kan dus de betrouwbaarheid van het systeem verhoogd worden.



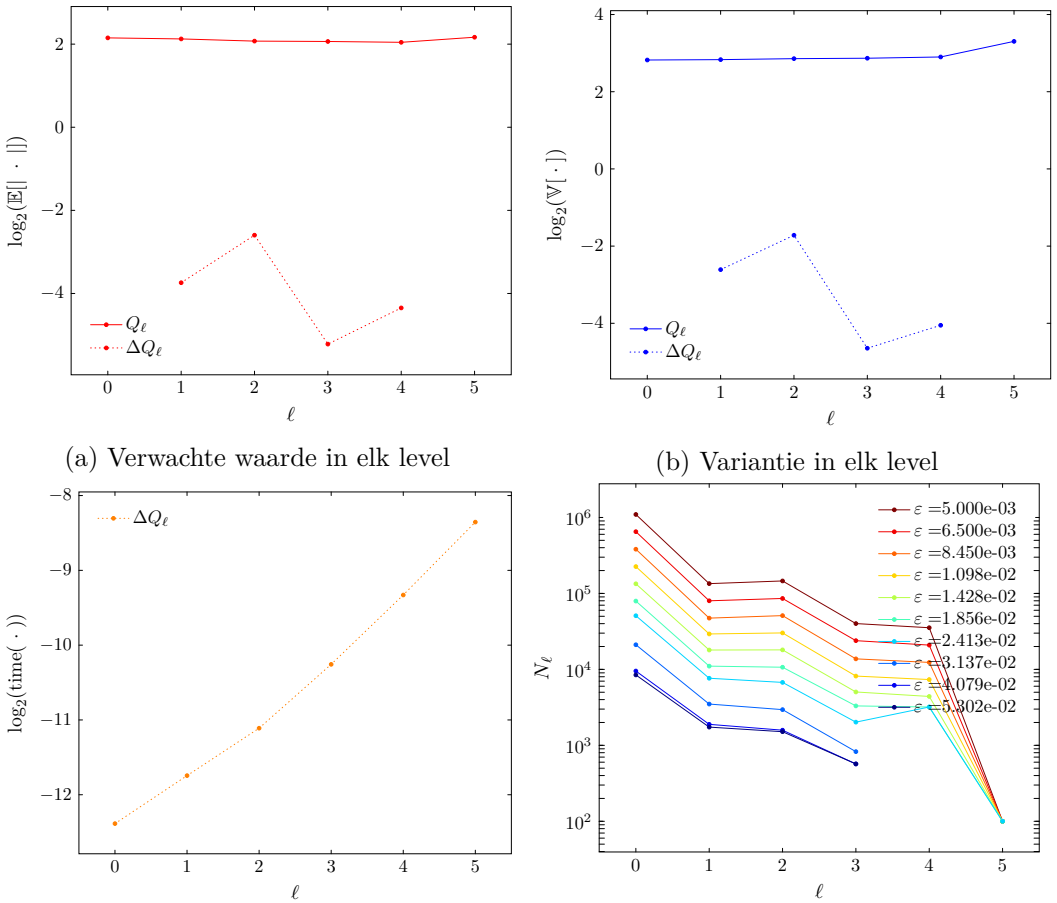
(a) $k = 0.5$ (b) $k = 1$ (c) $k = 3$

Figuur 5.5: 20 componenten, benadering van de verdeling de levensduur

5.1.2 Systeem met 30 componenten

Een systeem wordt gegenereerd zoals eerder uitgelegd tot dit systeem bestaat uit 30 componenten. De componenten hebben een levensduur volgens de Weibull-verdeling met vormparameter $k = 1$ en een schaalparameter uit een uniforme verdeling tussen 2 en 10. De schatter bepaalt de levensduur van dit systeem met een opgegeven nauwkeurigheid $\varepsilon = 5 \cdot 10^{-3}$.
De levensduur bepaald in de berekening is 4.12582 met een variantie $\mathbb{V} = 5.0006 \cdot 10^{-7}$. Er zijn simulaties op elk level gebruikt dus de ratio is $\text{RMSE}/\varepsilon = 0.70$. De diagnose van de schatter is gegeven in figuur 5.6. Figuur 5.6a toont de berekende levensduur op ieder level en de afname van de verwachte waarde van de correctieterm, de afname is naar schatting bepaald door $\alpha = 0.815067$. De afname van de variantie van de correctieterm is te zien in figuur 5.6b, de parameter van de afname is $\beta = 1.07812$. Er is geen monotone daling van de variantie en de waarde van de correctieterm (hogere waarde bij level $\ell = 2$). Dit fenomeen werd ook in de MLQMC berekening van het vorige systeem opgemerkt en wordt in meerdere berekeningen van de betrouwbaarheid van systemen gezien. De hogere variantie en correctieterm kan het resultaat zijn van een te ruwe Monte Carlo berekening in het selectie-algoritme (sectie 4.1) waarbij de waarde onvoldoende nauwkeurig geschat wordt en minder optimale deelverzamelingen geselecteerd worden voor een bepaald level.
De variantie van de correctieterm bij de hogere levels is altijd lager dan de variantie van de levensduur, dit zorgt dat er een lagere rekenkost is bij MLMC. De kost per

level stijgt, zoals te zien is in figuur 5.6c ($\gamma = 0.883929$). In figuur 5.6d worden het aantal simulaties per level weergegeven. Hierbij is er duidelijk een afname van het aantal simulaties bij de hogere levels. Bij hoge waarden voor ε kan de berekening zelfs eerder beëindigd worden omdat de afwijking veroorzaakt door het niet simuleren van de laatste levels, lager is dan $\varepsilon/\sqrt{2}$.



(a) Verwachte waarde in elk level

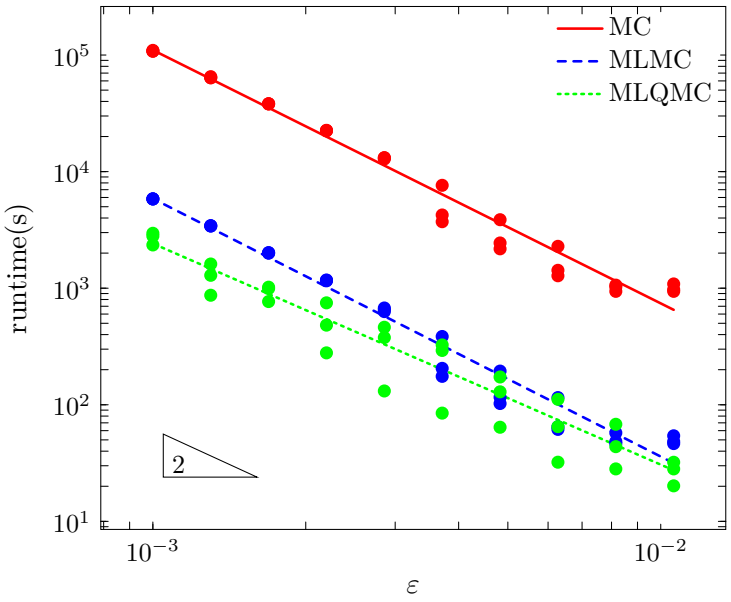
(b) Variantie in elk level

(c) Kost voor elk level

(d) Aantal simulaties

Figuur 5.6: 30 componenten, diagnose van de Multilevel Monte Carlo berekening

De vergelijking van de rekenkost van de Monte Carlo berekening, de Multilevel Monte Carlo berekening en de Multilevel Quasi-Monte Carlo berekening wordt gegeven in figuur 5.7. De rekenkost wordt op dezelfde manier berekend als in het begin van dit hoofdstuk is aangegeven. De figuur laat duidelijk zien dat Multilevel Monte Carlo een lagere rekenkost heeft dan Monte Carlo voor dit systeem. De ratio $cost_{MLMC}/cost_{MC}$ ligt tussen 0.04 en 0.06. De Multilevel Monte Carlo berekening is dus 17 tot 23 keer sneller dan de Monte Carlo berekening. Een tweede conclusie uit de figuur is dat de rekenkost verder gedrukt wordt door Quasi-Monte Carlo te gebruiken. Dit werd ook gezien bij het vorige systeem. De asymptotische complexiteit van de rekenkost van MLQMC is hier $\mathcal{O}(\varepsilon^{-p})$ met $p \approx 1.82$.



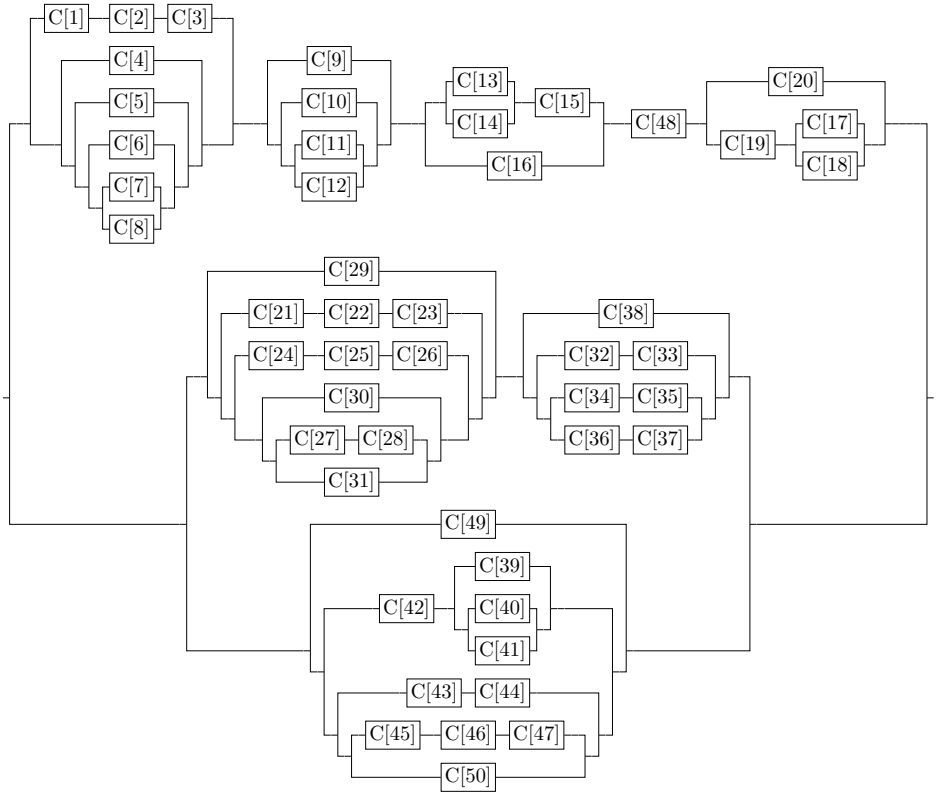
Figuur 5.7: 30 componenten, rekenkost van Monte Carlo, Multilevel Monte Carlo en Multilevel Quasi-Monte Carlo

Invloed componenten

De simulatie is herhaald voor hetzelfde systeem waarbij de vormparameter voor de componenten op $k = 3$ is gezet. De bespreking van deze berekening is terug te vinden in de bijlage. De rekenkost voor dezelfde nauwkeurigheid ligt lager, dit werd reeds opgemerkt in het vorige systeem. De verdeling van de levensduur van de componenten zorgen voor een grotere variantie van de levensduur van het systeem. Dit heeft een invloed op de rekenkost van de berekening. De hogere variantie van de levensduur zorgt voor een hogere MSE. Voor een vaste nauwkeurigheid moet er dus meer simulaties uitgevoerd worden als de variantie van de levensduur hoger is. Deze hogere variantie ontstaat blijkbaar door de hogere variantie in de levensduur van de componenten.

5.1.3 systeem met 50 componenten

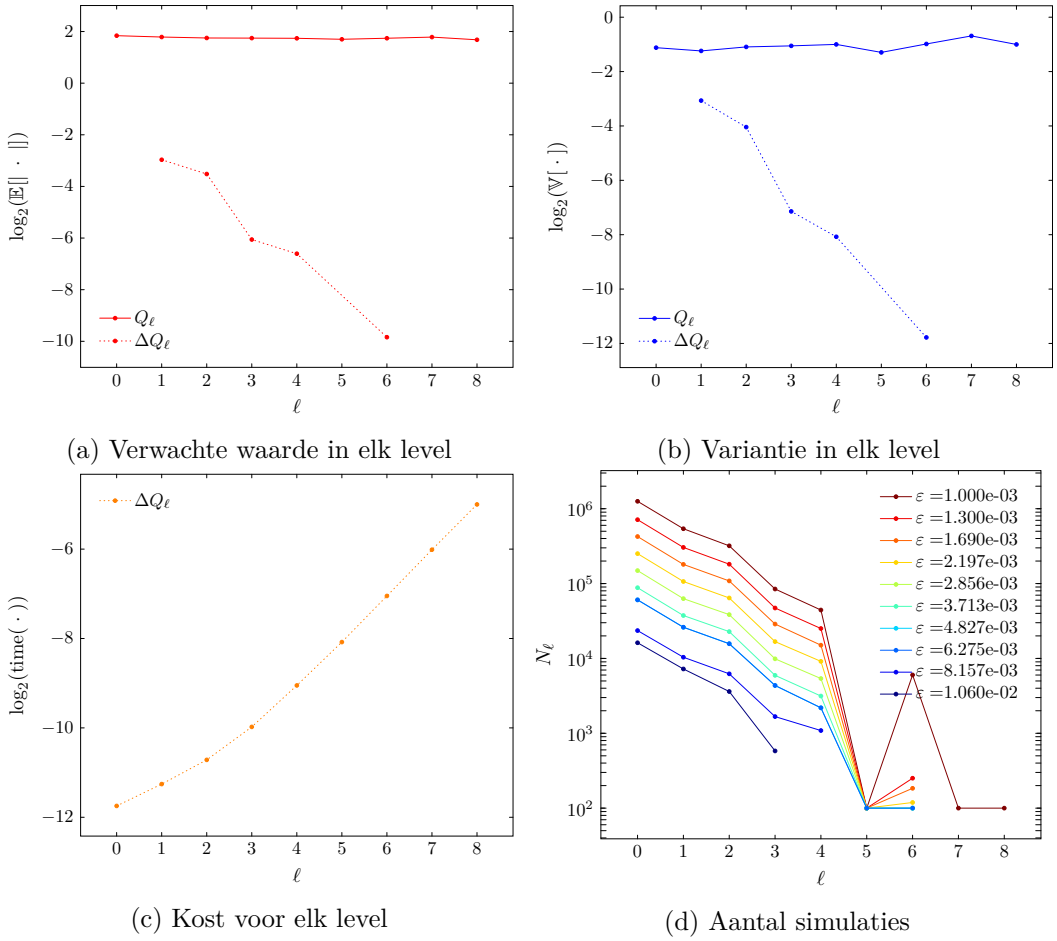
Het laatste systeem dat hier besproken wordt is een systeem met 50 componenten met een vormparameter $k = 3$ en de schaalparameter is een waarde uit een uniforme verdeling tussen 2 en 10. Het is op dezelfde manier gegenereerd als het systeem met 20 componenten. Het betrouwbaarheidsblokdiagram van dit systeem is weergegeven in figuur 5.8. Het systeem heeft 2808 minimale snedes. De levelstructuur bestaat uit 10 levels waarbij er ongeveer een verdubbeling is van het aantal snedes per level.
De diagnose van de berekening wordt gegeven in figuur 5.9. De opgegeven nauwkeurigheid voor de berekening is $\varepsilon = 10^{-3}$. De verwachte waarde op ieder level en de



Figuur 5.8: Systeem met 50 componenten

verwachte waarde van de correctieterm wordt linksboven gegeven. De variantie in ieder level en op de correctieterm worden gegeven in de figuur 5.9b. De kost van de simulatie in ieder level wordt weergegeven in de figuur linksonder. Deze figuren tonen het nodige gedrag zoals in de voorwaarden van het theorema van Multilevel Monte Carlo. De schatter geeft een benadering voor de parameters: $\alpha = 1.2726$, $\beta = 1.71243$ en $\gamma = 0.882981$. Deze waarden geven de indruk dat voor dit systeem de MLMC berekening optimaal efficiënt is omdat de variantie van de correctieterm sneller daalt dan de kost stijgt ($\beta > \gamma$). Rechtsonder wordt het aantal simulaties op ieder level weergegeven. Hierop is een daling te zien van het aantal simulaties bij de hogere levels. Bij level $\ell = 5$ is er een lager aantal simulaties uitgevoerd dan bij level $\ell = 6$. De oorzaak van dit verschijnsel is dat op level $\ell = 5$ de correctieterm en de variantie van de correctieterm nul zijn ($\mathbb{E}[\hat{T}_5 - \hat{T}_4] = 0$ en $\mathbb{V}[\hat{T}_5 - \hat{T}_4] = 0$). Dit betekent dat level 5 geen bijdrage heeft aan de MLMC berekening en de berekening uitgevoerd kan worden zonder dit level (sectie 4.4.1). Algemeen is de verwachte trend zichtbaar: er zijn minder simulaties nodig bij de hogere levels dan bij de lagere levels. De berekende waarde voor de levensduur is $\hat{T}_S = 3.337$ met een variantie $\mathbb{V} = 9.9124 \cdot 10^{-7}$. De levelstructuur heeft 10 levels. Er is enkel gebruik gemaakt van 9 levels. Er is dus een afwijking omdat het systeem geen simulatie uitvoert op he fijnste level. Deze afwijking is geschat op: bias $= 3.95016 \cdot 10^{-5}$. De MSE wordt gegeven door: MSE $= \mathbb{V} + \text{bias}^2$. Voor deze berekening is deze : MSE $= 9.9124 \cdot 10^{-7} + (3.95016 \cdot 10^{-5})^2 = 9.928 \cdot 10{-}7$.

De ratio RMSE/$\varepsilon = 9.9639 \cdot 10^{-4}/10^{-3} = 0.996$ bewijst de correctheid van de berekening. Figuur 5.10 toont de rekenkost van Multilevel Monte Carlo vergeleken met standaard Monte Carlo.



(a) Verwachte waarde in elk level

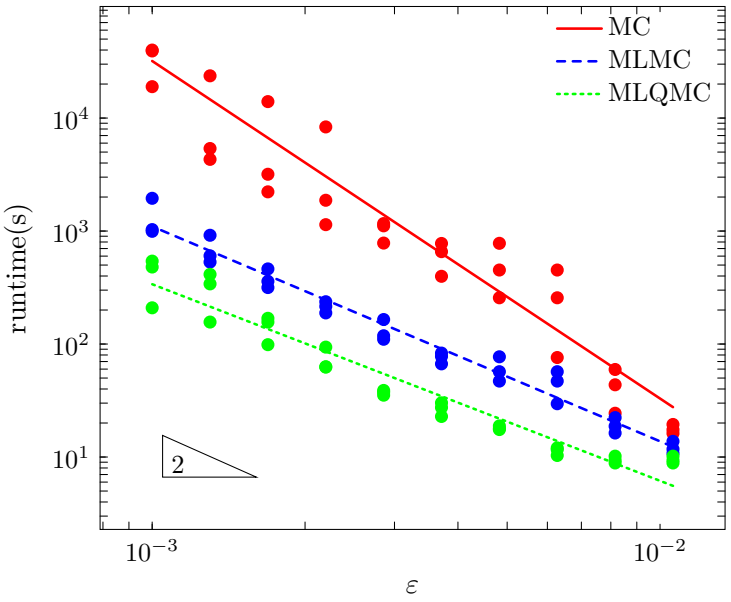
(b) Variantie in elk level

(c) Kost voor elk level

(d) Aantal simulaties

Figuur 5.9: 50 componenten, diagnose van de Multilevel Monte Carlo berekening

De gemeten rekenkost wordt weergegeven door de scatter punten, de lijnen zijn de regressie door 3 simulaties. De figuur toont dat Multilevel Monte Carlo een lagere rekenkost heeft dan de standaard Monte Carlo methode. De figuur geeft ook de rekenkost van Multilevel Quasi-Monte Carlo. Door gebruik te maken van Quasi-Monte Carlo wordt de rekenkost nog verder gereduceerd. Figuur 5.11 geeft de diagnose van de Multilevel Quasi-Monte Carlo waarbij opnieuw de afname van de correctieterm, de afname van de variantie van deze term en de toename van de kost in elk level te zien is. Door Quasi-Monte Carlo te gebruiken wordt de simulaties van de levensduur van de componenten meer uniform. De simulatie van de levensduur van het systeem wordt beëindigd bij level $\ell = 6$. De bias die geïntroduceerd wordt is $3.62 \cdot 10^{-4}$. De geschatte levensduur van dit systeem is $\hat{T} = 3.365$ de RMSE $= 9.65 \cdot 10^{-4}$ is lager dan de opgegeven nauwkeurigheid. De rekenkost van Multilevel Quasi-Monte



Figuur 5.10: 50 componenten, rekenkost van Monte Carlo, Multilevel Monte Carlo en Multilevel Quasi-Monte Carlo

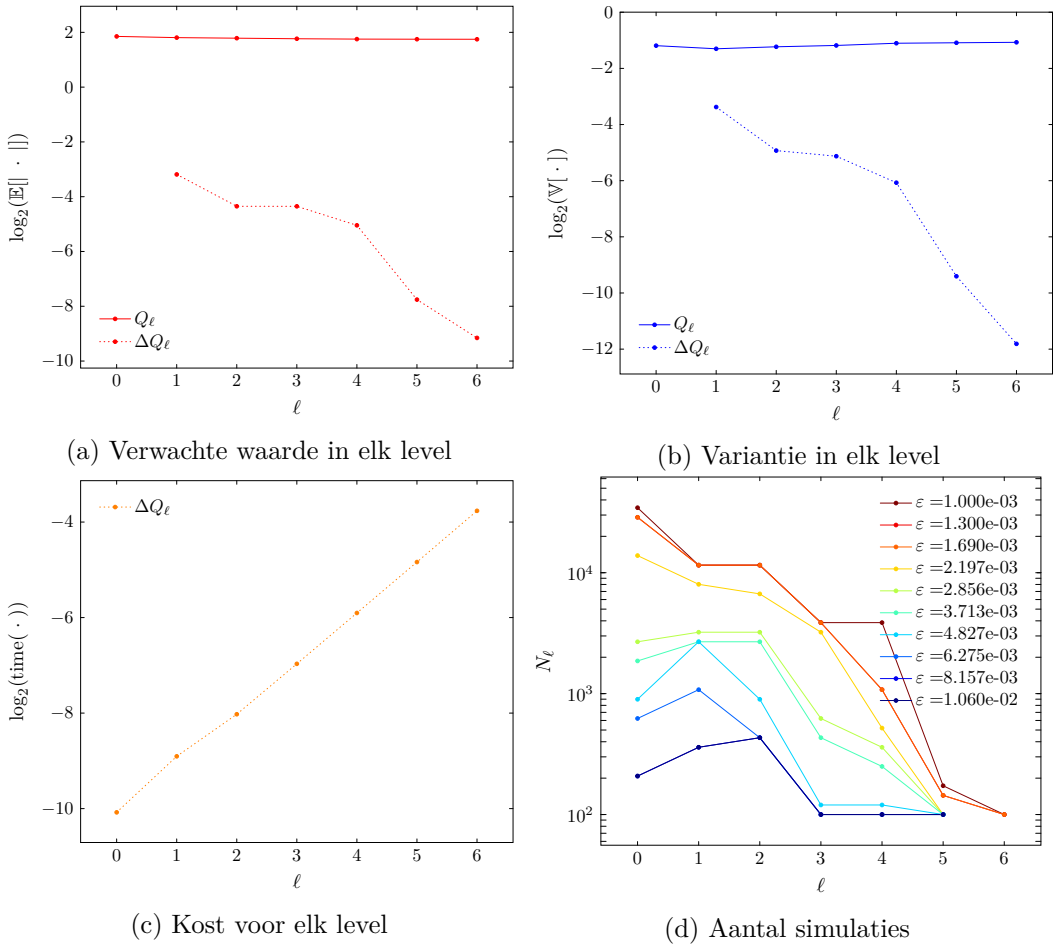
Carlo ligt lager dan de rekenkost van de Multilevel Monte Carlo, er is echter geen verlaging zichtbaar van de complexiteit zoals bij het systeem van 20 componenten en 30 componenten. De asymptotische complexiteit is hier dus $\mathcal{O}(\varepsilon^{-2})$.

5.1.4 Andere verdeling van de levensduur

De berekeningen die tot hier zijn uitgevoerd, zijn toegepast op systemen waarbij de levensduur van de componenten een Weibull-verdeling vertonen met een bepaalde vormparameter en een verschillende schaalparameter. Omdat we een methode willen die algemeen toepasbaar is, is er in hoofdstuk 4 gesproken over het gebruik van andere verdelingen, exponentiële verdelingen en uniforme verdelingen. Hierbij is gezien dat de exponentiële verdeling een speciaal geval is van de Weibull-verdeling en de methode is dus vanzelfsprekend toepasbaar in situaties met exponentiële verdelingen. De uitbreiding naar uniforme verdelingen is minder vanzelfsprekend. In dit deel wordt een systeem onderzocht waarbij de levensduur van de componenten door een uniforme verdeling bepaald wordt. Bijkomend wordt er gekeken naar een systeem dat bestaat uit componenten die een levensduur hebben volgens onafhankelijke en gelijke verdelingen.

Uniforme verdeling

Een berekening van de levensduur van het systeem in sectie 5.1.1 wordt uitgevoerd waarbij de kansverdeling van de levensduur van de componenten een uniforme verdeling U(0,b) is met b willekeurig tussen 2 en 10. De opgegeven nauwkeurigheid



(a) Verwachte waarde in elk level

(b) Variantie in elk level

(c) Kost voor elk level

(d) Aantal simulaties

Figuur 5.11: 50 componenten, diagnose van de Multilevel Quasi-Monte Carlo berekening

is $\varepsilon = 5 \cdot 10^{-4}$. De figuren met de diagnose van de berekening en de rekenkost zijn gegeven in figuur 5.12. De weergegeven diagnose is de diagnose van de MLMC schatter. De diagnose van de MLQMC schatter is terug te vinden in bijlage A.6.
De diagnose van de berekening toont geen vreemde resultaten. De correctieterm en de variantie van de correctieterm daalt bij de hogere levels en de kost van een simulatie is hoger bij deze levels. De voorwaarden van het theorema zijn dus voldaan.
De rekenkost van Monte Carlo, Multilevel Monte Carlo en Multilevel Quasi-Monte Carlo tonen, in tegenstelling tot het systeem met de Weibull-verdeling, enkel een verlaging met een constante factor. Er is geen verlaging van de complexiteit van de rekenkost. De berekening met Multilevel Monte Carlo is 1.5 tot 2.5 keer sneller dan Monte Carlo. Multilevel Quasi-Monte Carlo is nog enkele malen efficiënter waarbij de berekening 10 tot 50 keer sneller is dan Monte Carlo.
De methode is dus robust voor andere verdelingen en kan dus toegepast worden op reële toepassingen waarvan de componenten een levensduur hebben die bepaald is

door een andere verdeling. Figuur 5.12f toont de kansverdeling van de levensduur van het systeem die opgesteld is met de manier besproken in sectie 4.4.3. Hier is geen duidelijke gelijkenis te zien met de uniforme verdeling van de levensduur van de componenten.

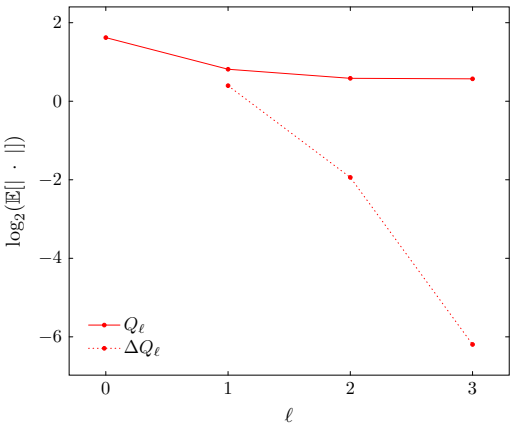
Onafhankelijke en gelijke verdeling (i.i.d.)

Een speciaal geval van systemen zijn systemen die bestaan uit componenten met een onafhankelijke en gelijk verdeelde levensduur. Dit zijn systemen die bestaan uit 1 soort component. Een voorbeeld van zo een systeem is een netwerk van identieke servers waarbij de toegang tot dataopslag gebeurt via deze cluster. De toegankelijkheid van de dataopslag hangt dan samen met de levensduur van de servers [24].
Bij dit soort systemen is het niet duidelijk welke snedes tot welke levels behoren. Het selectie-algoritme kan in dit geval geen ideale levelstructuur bepalen, maar desondanks bouwt het algoritme een levelstructuur op waarbij het aantal snedes in de levels verdubbelt ten opzichte van het vorige level. Dit zorgt voor een stijgende kost in de levels en een dalende variantie [2].
Het systeem dat gebruikt wordt voor de berekening is een systeem met 25 componenten. Dit systeem is gevisualiseerd in figuur 5.13. De levensduur van de componenten hebben een Weibull-verdeling met vormparameter $k = 3$ en schaalparameter $\lambda = 5$. De berekening van de levensduur is enkel uitgevoerd met Multilevel Quasi-Monte Carlo tot een nauwkeurigheid $\varepsilon = 10^{-5}$.
De diagnose van deze berekening is gegeven in figuur 5.14. Dit resultaat toont de robuustheid van het selectie-algoritme. Er wordt een levelstructuur opgebouwd die een stijgende kost heeft door de verdubbeling van de snedes ten opzichte van het vorige level. Deze verdubbeling zorgt voor de daling in de variantie van de correctieterm [2]. De rekenkost voor dit systeem heeft een asymptotische complexiteit $\mathcal{O}(\epsilon^{-p})$ met $p < 2$. De verdeling van de levensduur van dit systeem is weergegeven in figuur 5.14f. Deze is berekend met de methode beschreven in sectie 4.4.3. Hierbij is er de kans op een levensduur lager dan 3 bijna onbestaande. De levensduur van het systeem is tussen 3 en 5.

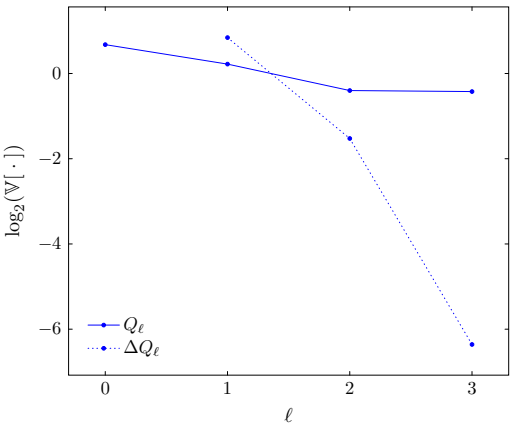
5.1.5 Rekenkost in relatie met het aantal componenten

Als finale deel van deze sectie wordt er gekeken naar de relatie tussen de rekenkost van de berekening en het aantal componenten in een systeem. Omdat in het slechtste geval het aantal snedes van het systeem exponentieel toeneemt als het aantal componenten toeneemt en de rekenkost afhankelijk is van het aantal snedes is dit een interessante invalshoek.
Om een indruk te krijgen van de rekenkost in functie van het aantal componenten wordt er gestart van een systeem met 10 componenten. Dit systeem wordt uitgebreid naar een groter systeem door telkens 5 componenten toe te voegen. Het systeem

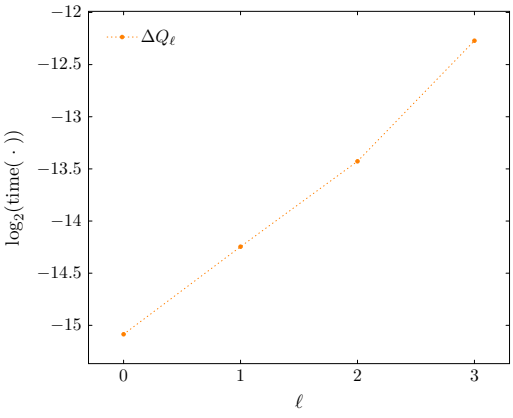
wordt vervolgens gesimuleerd met Multilevel Monte Carlo en Multilevel Quasi Monte Carlo met een tolerantie gelijk aan $\varepsilon = 5 \cdot 10^{-3}$. Het grootste systeem dat gesimuleerd wordt heeft 75 componenten.
Figuur 5.15 toont de rekenkost van de simulatie voor een systeem dat groeit van 10 componenten naar 75 componenten. Hierbij is duidelijk dat bij grotere systemen Multilevel Monte Carlo en Multilevel Quasi Monte Carlo een lagere rekenkost hebben dan Monte Carlo. De figuur bevestigt dat het gebruik van Quasi-Monte Carlo de efficiëntie van de berekening verhoogt.



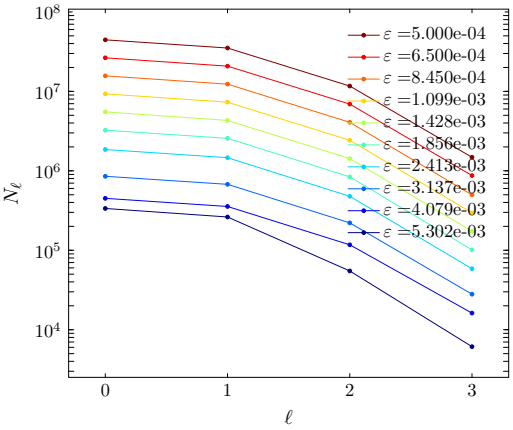
(a) Verwachte waarde in elk level



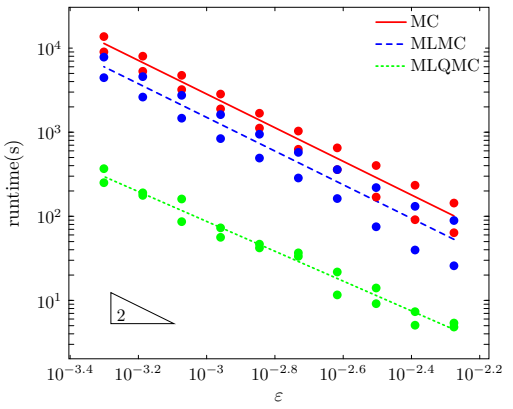
(b) Variantie in elk level



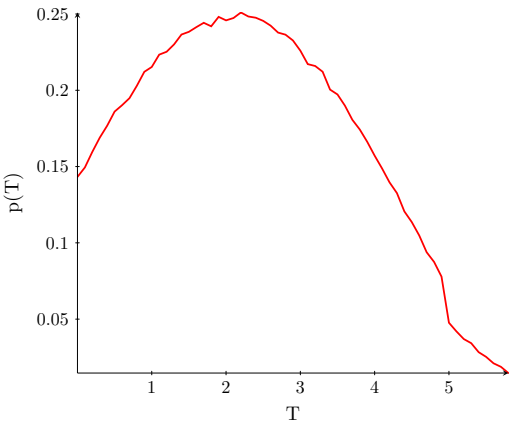
(c) Kost voor elk level



(d) Aantal simulaties

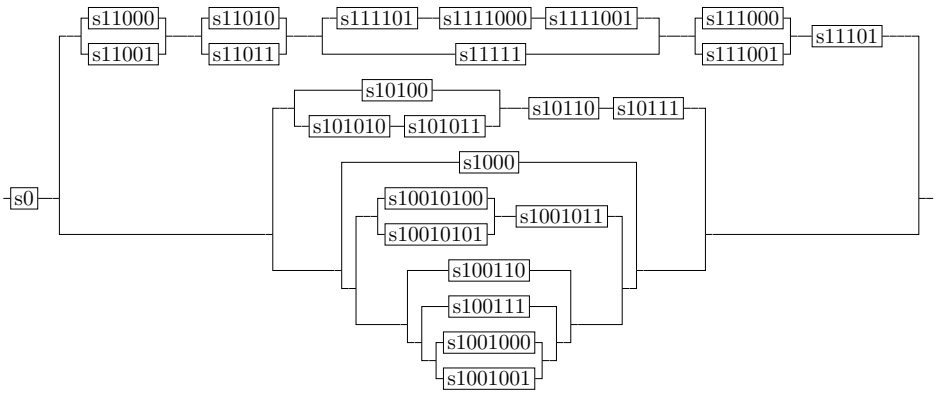


(e) Rekenkost van Monte Carlo, Multilevel Monte Carlo, Multilevel Quasi-Monte Carlo

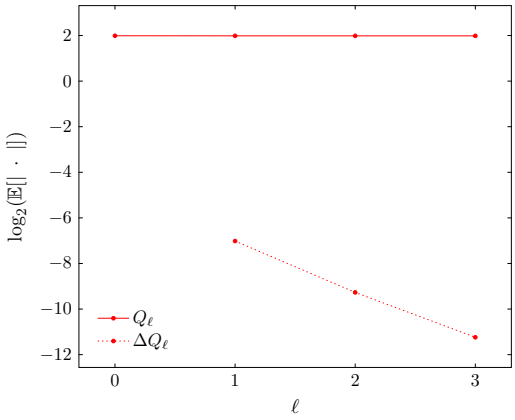


(f) Verdeling van de levensduur

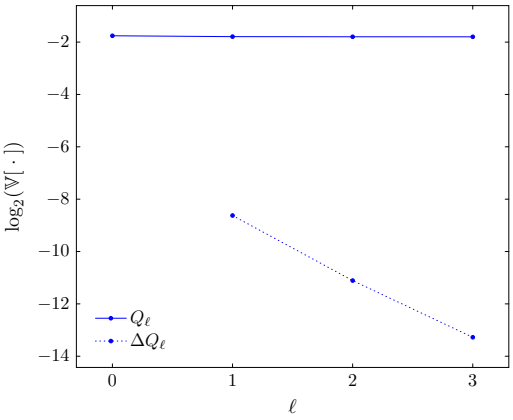
Figuur 5.12: Uniforme verdeling



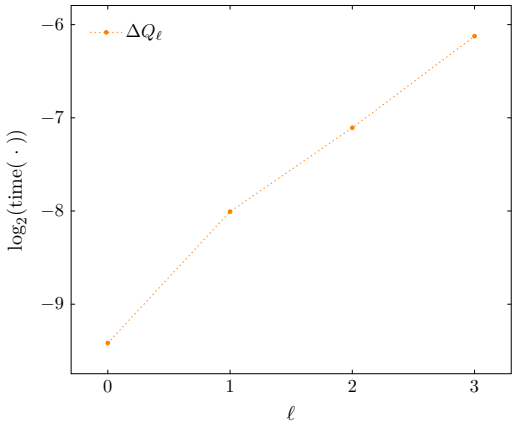
Figuur 5.13: Systeem met componenten een levensduur volgens een onafhankelijke en gelijke verdeling



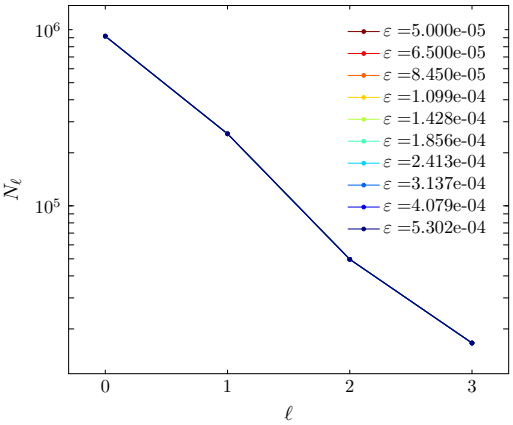
(a) Verwachte waarde in elk level



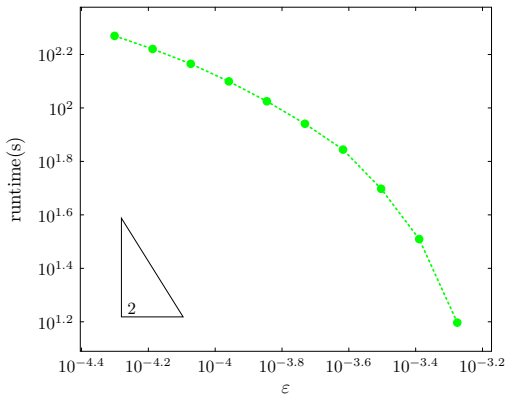
(b) Variantie in elk level



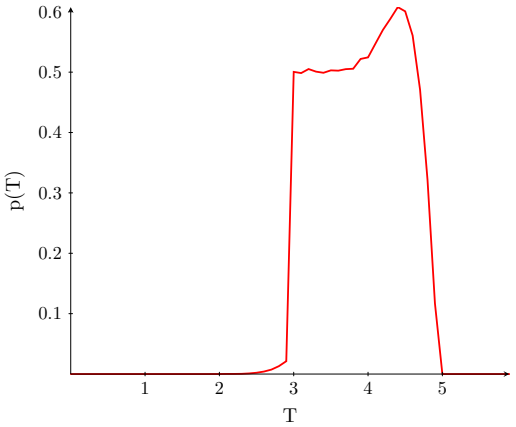
(c) Kost voor elk level



(d) Aantal simulaties

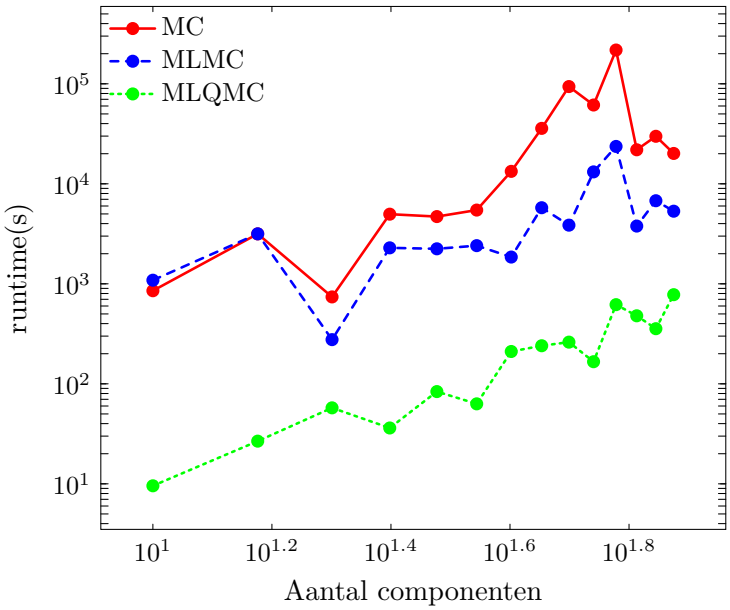


(e) Rekenkost van MLQMC



(f) Verdeling van de levensduur

Figuur 5.14: systeem met componenten met i.i.d. levensduur, resultaat van de schatter



Figuur 5.15: Rekenkost vs. aantal componenten

5.2 Herstelbare systemen

Zoals eerder vermeld wordt de overstap gemaakt naar systemen met herstelbare componenten. Deze overstap laat zien dat Multilevel Monte Carlo toepasbaar is voor uiteenlopende scenario's. Systemen met herstelbare componenten kunnen niet gesimuleerd worden met op de manier die vermeldt staat in sectie 2.4. Door het herstelproces verandert de toestand van de componenten en wordt het simulatieproces een stochastisch proces van falen en herstellen. Dit zorgt voor een bijkomende willekeurigheid in de rekenkost van de MC simulatie.
De duur van het herstelproces is bepaald door een exponentiële verdeling met intensiteitsparameter λ. Een grotere waarde voor λ betekent een grotere intensiteit per tijdsduur en dus indiceert dat het herstelproces van kortere duur is, dit zorgt dat het component zich minder lang in een gefaalde toestand bevindt. De kans is dus kleiner dat bijkomende componenten falen en dat het systeem faalt. Een snel herstelproces kan zorgen voor een langere levensduur.

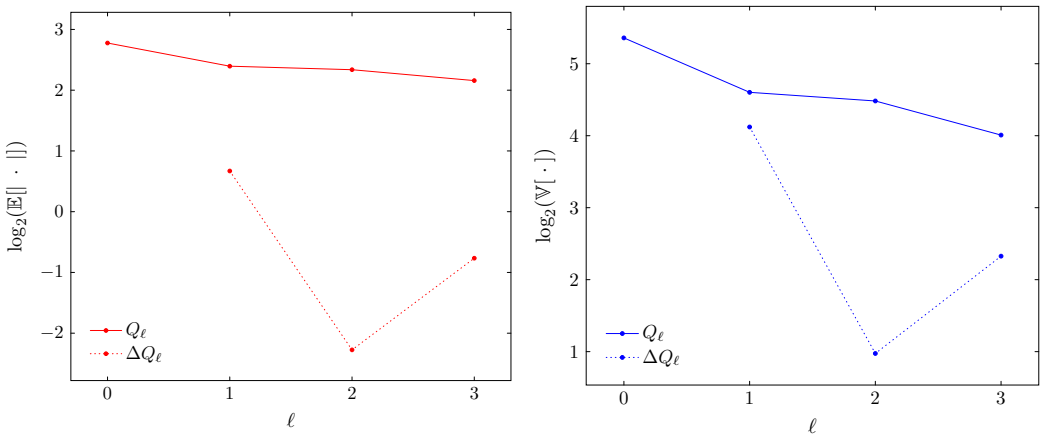
5.2.1 Herstelbaar systeem 1: 20 componenten

Herneem het systeem voorgesteld in sectie 5.1.1. Het herstelproces gebeurt volgens een exponentiële verdeling met $\lambda = 1$. De berekening is uitgevoerd met een opgegeven nauwkeurigheid $\varepsilon = 5 \cdot 10^{-3}$. De diagnose en de vergelijking in de rekenkost worden gegeven in figuur 5.16. De afname van de verwachte waarde gebeurt volgens $\alpha = 0.717734$. De daling bevestigd dat het selectie-algoritme de belangrijkheid van de snedes goed weet in te schatten [2]. Figuur 5.16b toont de afname van de variantie ($\beta = 0.897262$), en figuur 5.16c de toename van de kost ($\gamma = 4.15769$). Dit toont de kwaliteit van het selectie-algoritme : de variantie daalt per level, er zijn dus minder simulaties nodig op de hogere levels, waar de kost veel hoger is.
De levensduur van het systeem is $\hat{T} = 4.4658$ met een variantie $\mathbb{V} = 4.97296 \cdot 10^{-5}$. Vergeleken met de berekening in sectie 5.1.1 is de levensduur nu hoger. Het toevoegen van het herstelproces heeft er dus voor gezorgd dat het systeem een langere levensduur heeft.
De vergelijking van de rekenkost voor MC en Multilevel Monte Carlo is gegeven in figuur 5.16e. Hierin is een duidelijke versnelling te zien bij het berekenen van de levensduur. Multilevel Monte Carlo heeft maar een fractie tussen 0.0168-0.0224 van de rekentijd van Monte Carlo nodig voor een berekening met dezelfde nauwkeurigheid. Dit is een versnelling van 44 tot 60 keer.

5.2.2 Herstelbaar systeem 2: 25 componenten

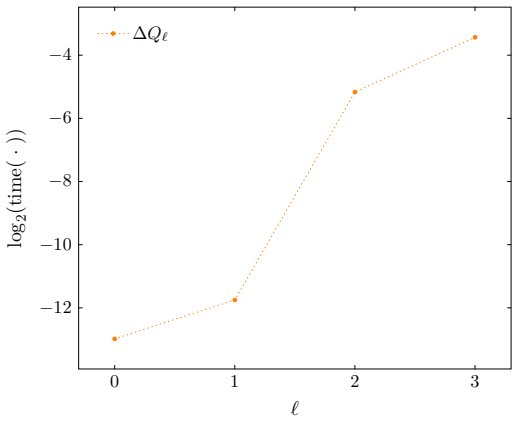
Er wordt een systeem opgebouwd van 25 componenten volgens de methode vermeld in het begin van dit hoofdstuk. De componenten hebben een Weibull-verdeling met vormparameter $k = 3$ en een schaalparameter uit de uniforme verdeling U(2,10). Het herstelproces is bepaald door een exponentiële verdeling met intensiteitsparameter

$\lambda = 1$. De berekening is uitgevoerd met een opgegeven nauwkeurigheid $\varepsilon = 10^{-3}$. De diagnose en de vergelijking in de rekenkost worden gegeven in figuur 5.17. Er is een sterke daling van de verwachte waarde en de variantie van de correctieterm ($\alpha = 3.35289$, $\beta = 3.84361$). Door het herstelproces is er een sterke stijging van de kost, $\gamma = 3.15062$. De variantie daalt per level, er zijn dus minder simulaties nodig op de hogere levels, waar de kost veel hoger is.
De vergelijking van de rekenkost voor MC en Multilevel Monte Carlo is gegeven in figuur 5.16e. Hierin is Multilevel Monte Carlo duidelijk sneller dan Monte Carlo.

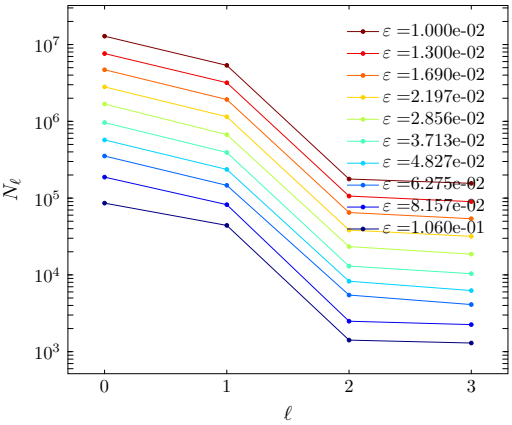


(a) Verwachte waarde in elk level

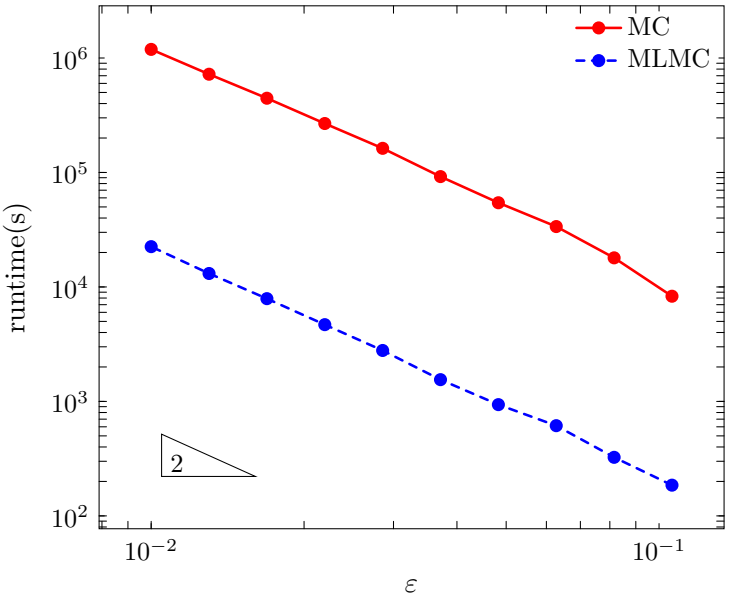
(b) Variantie in elk level



(c) Kost voor elk level

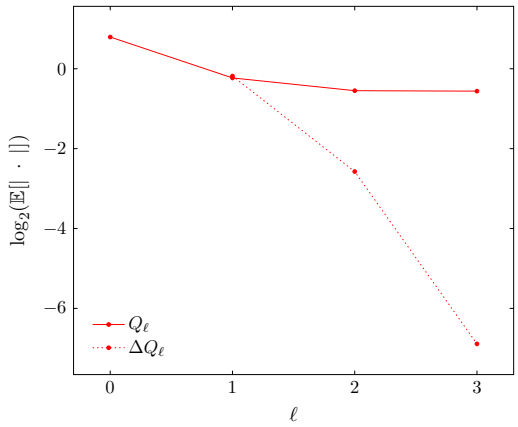


(d) Aantal simulaties

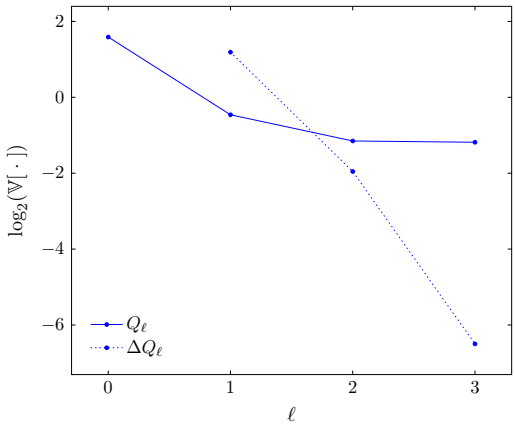


(e) Rekenkost Multilevel Monte Carlo, Multilevel Quasi-Monte Carlo, Monte Carlo

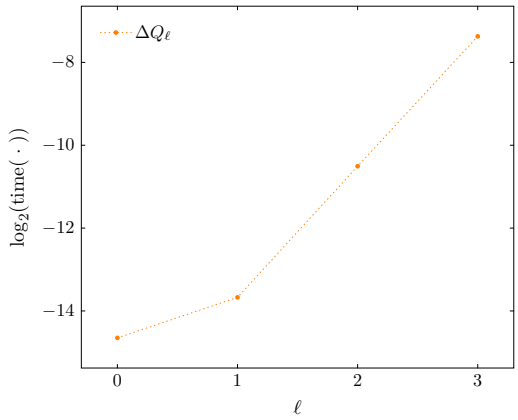
Figuur 5.16: Herstelbaar systeem 1, diagnose en rekenkost van de schatter



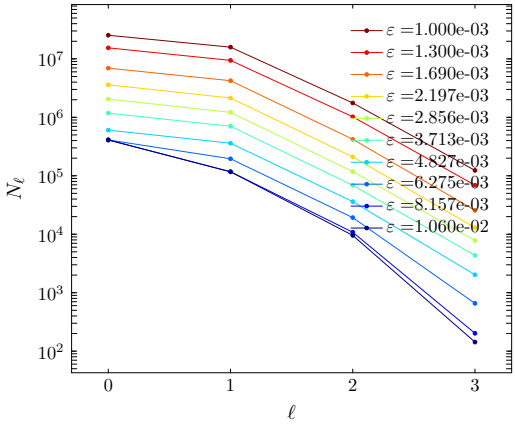
(a) Verwachte waarde in elk level



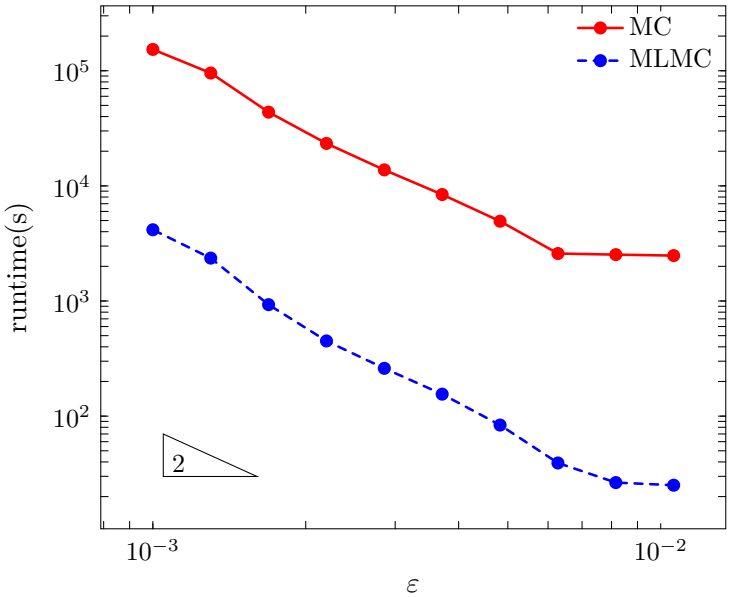
(b) Variantie in elk level



(c) Kost voor elk level



(d) Aantal simulaties



(e) Rekenkost Multilevel Monte Carlo, Multilevel Quasi-Monte Carlo, Monte Carlo

Figuur 5.17: Herstelbaar systeem 2, diagnose en rekenkost van de schatter

5.3 Toepassing

De implementatie van Multilevel Monte Carlo wordt in deze sectie toegepast op 2 reële systemen. Hierbij wordt de toepasbaarheid van de methode bekeken en wordt er aandacht besteed aan de rekenkost van Monte Carlo, Multilevel Monte Carlo en Multilevel Quasi-Monte Carlo. Een eerste deel behandelt een brandalarm waarbij de veroudering van de componenten de grootste oorzaak is van het falen. Het tweede deel bespreekt de berekening van de levensduur van een waterkrachtcentrale. In deze toepassing hebben de componenten een verschillende verdeling van de levensduur.

5.3.1 Toepassing 1: Brandalarm

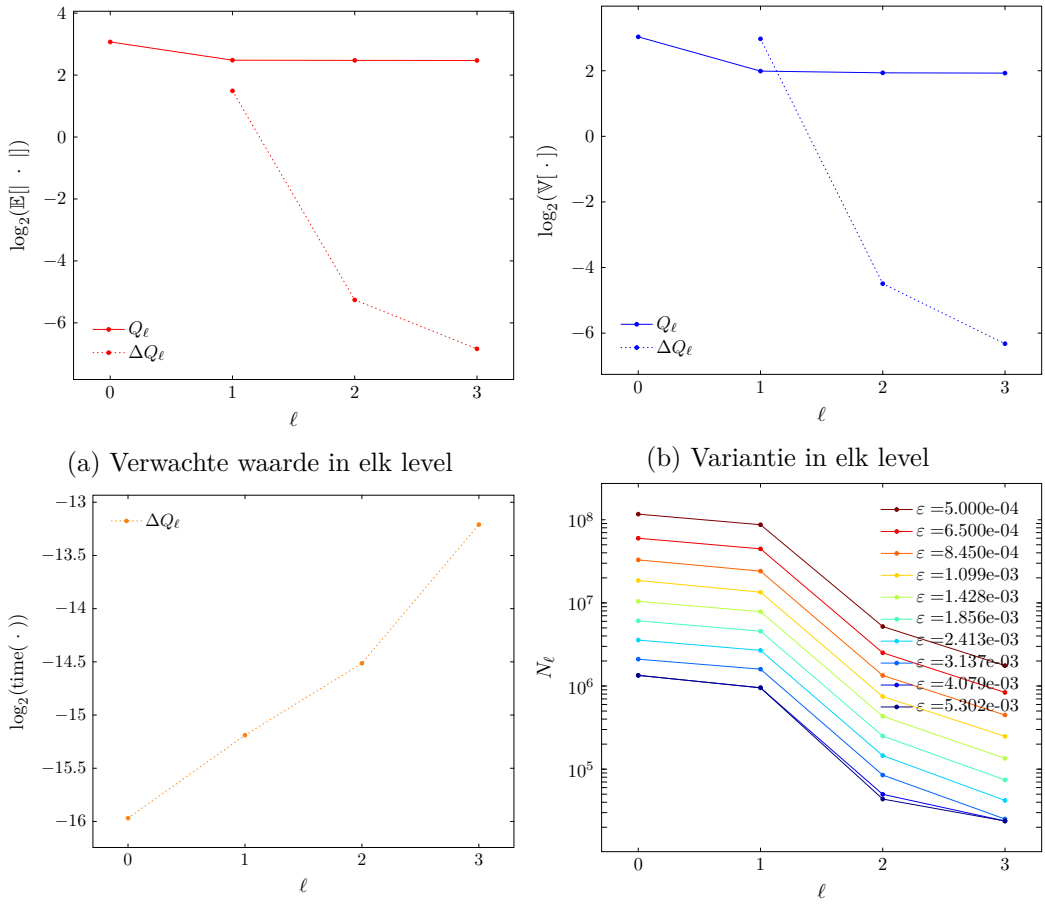
De eerste toepassing, reeds vermeld in hoofdstuk 2, is het brandalarm. Hierbij is de verdeling van de levensduur van de componenten een Weibull-verdeling, Weibull(k, λ) met k de vormparameter en λ de schaalparameter. De parameters die gebruikt zijn tijdens de simulatie zijn gegeven in tabel 5.1. De vormparameter is hier 3 omdat

Tabel 5.1: Parameters van de verdeling van de levensduur van de componenten van het brandalarm

Component	Verdeling	Parameters
Voeding	Weibull	$k = 3$, $\lambda = 10$
Start relais	Weibull	$k = 3$, $\lambda = 10$
Rookdetector	Weibull	$k = 3$, $\lambda = 7$
Thermische zekering	Weibull	$k = 3$, $\lambda = 7$
2-van-de-3 schakelaar	Weibull	$k = 3$, $\lambda = 8$
Handmatige schakelaar	Weibull	$k = 3$, $\lambda = 8$
Druksensor	Weibull	$k = 3$, $\lambda = 10$
Operator	Weibull	$k = 3$, $\lambda = 15$

er wordt verondersteld dat de componenten robuust zijn tegen externe factoren. De voorname factor van falen is de veroudering.
De berekening is uitgevoerd met een opgegeven nauwkeurigheid $\varepsilon = 1 \cdot 10^{-4}$. De schatter geeft het resultaat weergegeven in figuur 5.18. Hierin is er een daling te zien bij de verwachte waarde van de correctieterm ($\alpha = 2.6536$). De variantie van de correctieterm daalt met $\beta = 2.97822$ en het aantal simulaties in de hogere levels is lager dan in de lage levels. De kost stijgt per level met een parameter $\gamma = 0.951298$. De diagnose van de Multilevel Quasi-Monte Carlo wordt gegeven in bijlage A.7. De levensduur van het brandalarm is $\hat{T}_S = 6.20453$ waarbij de RMSE$= 6.55 \cdot 10^{-5}$. Dit betekent dat men verwacht dat het brandalarm 6.2 jaar de functie kan uitvoeren.
In de vergelijking van de rekenkost, figuur 5.19, is er een duidelijke daling van de rekenkost door gebruik te maken van Multilevel Monte Carlo. De complexiteit van de berekening is $\mathcal{O}(\varepsilon^{-2})$. De ratio tussen de rekenkost voor Monte Carlo en Multilevel

Monte Carlo is 0.281-0.370. Dit betekent dat Multilevel Monte Carlo 2.70 à 3.55 keer sneller is dan Monte Carlo. Door de hoge waarde van β en de grootte van het systeem wordt er bij Multilevel Quasi-Monte Carlo een complexiteit $\mathcal{O}(\varepsilon^{-p})$ met $p < 2$ waargenomen. De rekenkost van Multilevel-Quasi Monte Carlo is bij een nauwkeurigheid $\varepsilon = 10^{-4}$ slechts 1% van de rekenkost van Monte Carlo. Bij deze nauwkeurigheid duurt de Monte Carlo berekening 3 uur en 25 minuten. Door de MLQMC methode te gebruiken reduceert dit naar slecht 2 minuten!



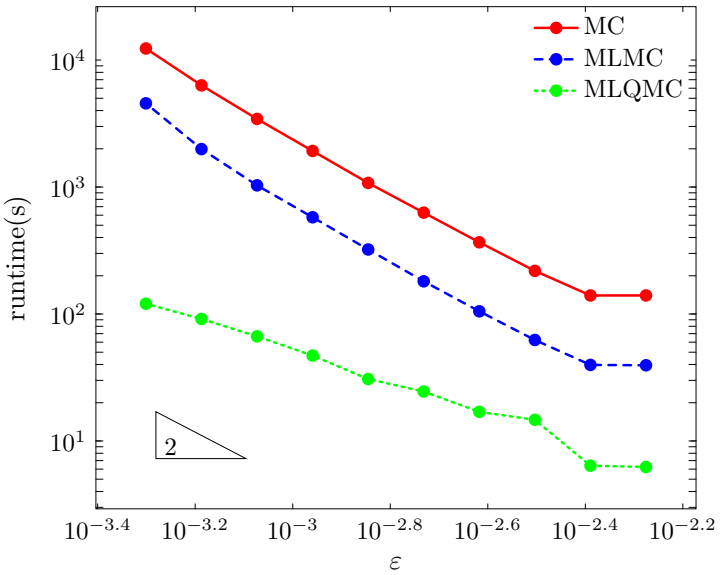
(a) Verwachte waarde in elk level

(b) Variantie in elk level

(c) Kost voor elk level

(d) Aantal simulaties

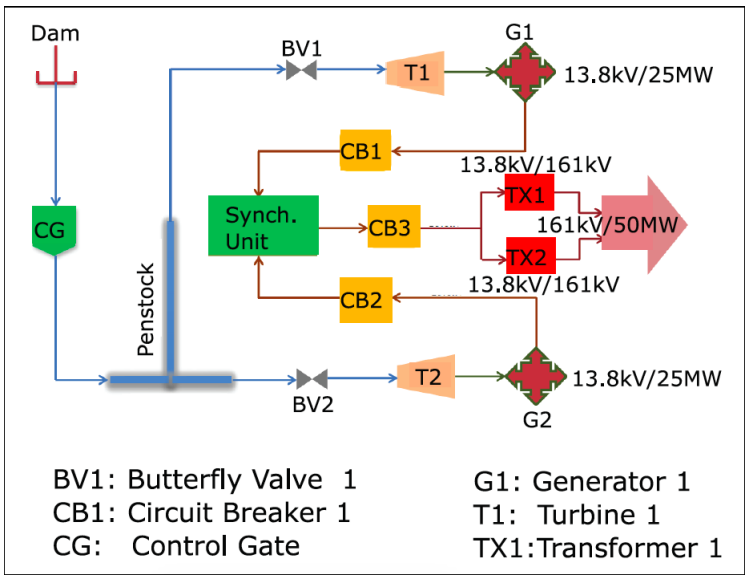
Figuur 5.18: Brandalarm, diagnose en resultaat van de schatter



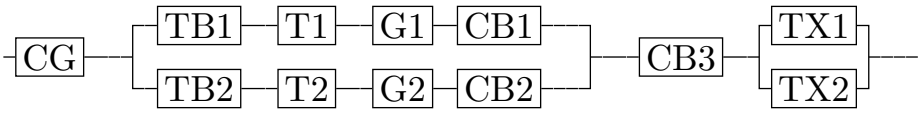
Figuur 5.19: Brandalarm, rekenkost Multilevel Monte Carlo, Multilevel Quasi-Monte Carlo, Monte Carlo

5.3.2 Toepassing 2: Hydro-elektrische energie centrale

Deze toepassing is een casus uit George-Williams et al. [9]. Het model dat onderzocht wordt is de Bumbana waterkrachtcentrale, een 50 MW centrale in Sierra Leone. De centrale bestaat uit twee eenheden met de volgende onderdelen: Vlinderventiel (BV), Turbine (T), generator (G), zekering (CB). De energie van deze eenheden worden gesynchroniseerd en gestuurd naar een transformator (TX). Voordat het water bij de energiecentrale komt, gaat deze door een debietregelaar (CG). Een schematische voorstelling en het betrouwbaarheidsblokdiagram zijn gegeven in figuur 5.20. Het element voor de synchronisatie is zeer betrouwbaar, de zekering in dit element is bepalend voor de betrouwbaarheid. Tabel 5.2 geeft de verdeling van de levensduur



(a) schematische weergave [9]



(b) betrouwbaarheidsblokdiagram

Figuur 5.20: Waterkrachtcentrale

van de componenten en de parameters voor deze verdelingen. Als extra uitbreiding op het onderzoek zijn de componenten in dit systeem niet onafhankelijk. De relaties in dit systeem zijn:

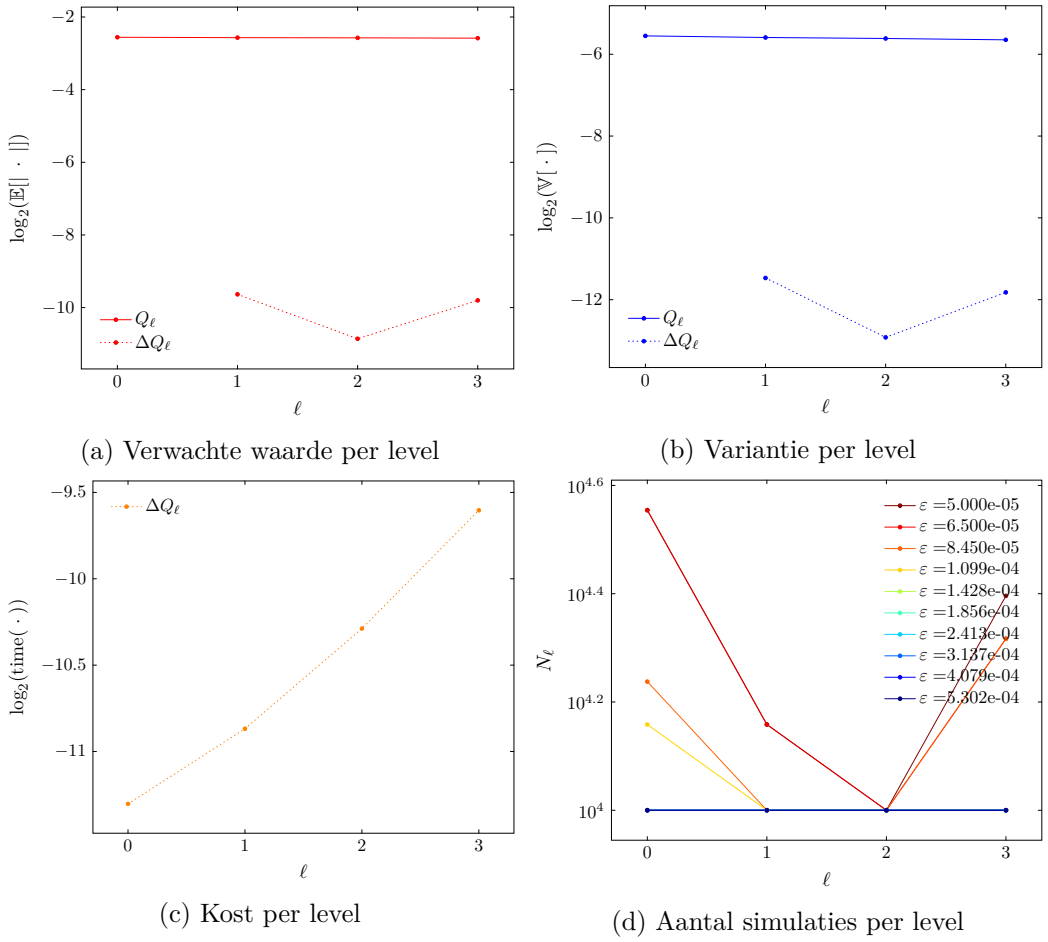
- onzuiverheden in de constructie of dam kunnen defecten veroorzaken in de bij de debietregelaar (CG) en in turbine 2 (T2) volgens een intensiteit van 2t per jaar, met t de tijd in werking. Turbine 1 (T1) is immuun tegen deze onzuiverheden.

- De zekering CD3 kan op 2 manieren falen. Dit falen is wordt bepaald door 2 onafhankelijke verdelingen. (zekering CD3$^{(1)}$ en zekering CD3$^{(2)}$)
- Het falen van de zekering in het synchronisatie element (CB3) veroorzaakt een storing in de zekering van eenheid 1 (CB1) of een storing in transformator van eenheid 2 (TX2) afhankelijk van de manier waarop CD3 faalt.
- Het falen van de zekering (CB1) uit eenheid 1 heeft als gevolg dat transformator 1 (TX1) faalt.

Tabel 5.2: Parameters van de verdeling van de levensduur van de componenten van de waterkrachtcentrale

Component	Verdeling	Parameters
Debietregelaar (CG)	Weibull	$k = 1.8$, $\lambda = 3$
Vlinderventiel (BV)	Weibull	$k = 2.3$, $\lambda = 1.8$
Turbine (T)	Weibull	$k = 3$, $\lambda = 4$
Generator (G)	Weibull	$k = 2.6$, $\lambda = 2.1$
Zekering (CB1,CB2)	Exponential	$\mu = 4$
Zekering(CB3)$^{(1)}$	Exponential	$\mu = 3.85$
Zekering(CB3)$^{(2)}$	Weibull	$k = 3$, $\lambda = 1.44$
Transformator(TX)	Gamma	$k = 1$, $\lambda = 3$
onzuiverheden	Weibull	$k = 2$, $\lambda = 1$

Figuur 5.21 geeft de resultaten van de simulatie met Multilevel Quasi-Monte Carlo voor een nauwkeurigheid $\epsilon = 3 \cdot 10^{-3}$. Volgens de schatter is de levensduur van deze waterkrachtcentrale $\hat{T}_S = 0.00135075$. De variantie op deze waarde is $V = 4.997 \cdot 10^{-7}$. De fout is dus kleiner dan de opgegeven nauwkeurigheid. $\sqrt{7.07 \cdot 10^{-4}} < 1 \cdot 10^{-3}$. Er zijn hier 4 levels waarbij de correctieterm en de variantie een stijging vertonen tussen level 2 en level 3. Dit samen met de stijging van de kost zorgt voor een situatie die niet optimaal is. Figuur 5.22 toont de vergelijking tussen Multilevel Monte Carlo, Multilevel Quasi-Monte Carlo en Monte Carlo. De rekenkost van Multilevel Monte Carlo is ondanks de stijgende variantie lager dan de Monte Carlo methode. De reden hiervoor is de lagere variantie in de correctieterm ten opzichte van de variantie in de levensduur. Door de introductie van de Quasi-Monte Carlo is er een significante verbetering van de rekenkost waarbij de complexiteit in deze situatie lager is dan $\mathcal{O}(\varepsilon^{-2})$.
De verwachte levensduur van de waterkrachtcentrale is laag waardoor dit niet praktisch is om te gebruiken. Een mogelijk interessantere inferentie is het bepalen van de betrouwbaarheid van het systeem op een specifiek tijdstip t. Deze betrouwbaarheid wordt gegeven door de signatuur van het systeem, $\mathrm{P}(T_S > t)$ [28]. Door gebruik te maken van de simulaties, zoals besproken in sectie 4.4.3 kan er een signatuur worden opgesteld waar deze betrouwbaarheid van afgelezen kan worden. Figuur 5.23 geeft de



(a) Verwachte waarde per level

(b) Variantie per level

(c) Kost per level

(d) Aantal simulaties per level

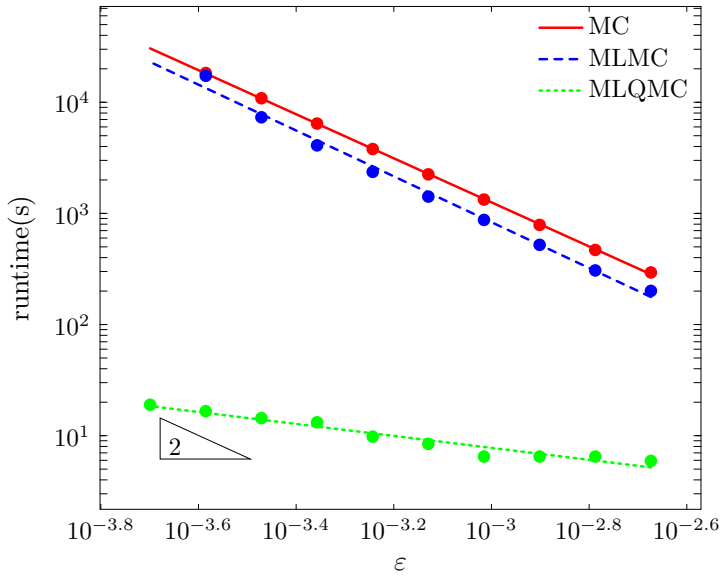
Figuur 5.21: Waterkrachtcentrale, diagnose van de Multilevel Quasi-Monte Carlo berekening

signatuur van de waterkrachtcentrale gebaseerd op de numerieke resultaten. Hieruit wordt bijvoorbeeld geconcludeerd dat er 30% kans is dat het systeem langer werkt dan 1 jaar.

5.4 Besluit van dit hoofdstuk

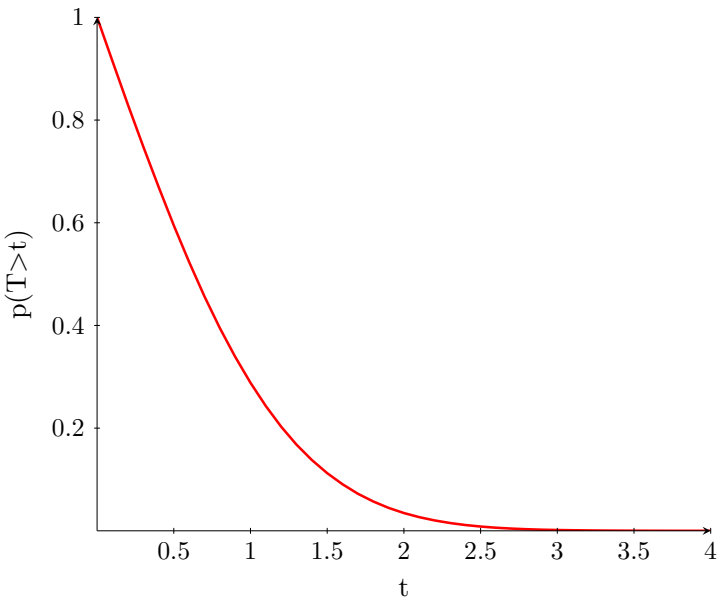
5.4.1 Multilevel Monte Carlo

Niet-herstelbare systemen zijn gebruikt voor de implementatie te voltooien zoals voorgesteld in [2]. Hierbij zijn 3 systemen gebruikt om de implementatie te beoordelen en de rekenkost te analyseren. Over de 3 systemen wordt hetzelfde besluit genomen dat het gebruik van Multilevel Monte Carlo zorgt voor een verlaging van de rekenkost. Deze rekenkost is afhankelijk van de variantie van de levensduur van het systeem en dus de variantie van de levensduur van de componenten. Een lagere



Figuur 5.22: Waterkrachtcentrale, rekenkost van Monte Carlo, Multilevel Monte Carlo en Multilevel Quasi-Monte Carlo

variantie van de levensduur zorgt dat er minder simulaties nodig zijn waardoor de kost voor de berekening lager ligt.
Er is een verschil in de kost voor het opstarten van de berekening tussen Monte Carlo en Multilevel Monte Carlo. Deze kost is in de vergelijking niet meegerekend maar wordt enkel belangrijk wanneer de nauwkeurigheid groot is en een hoge variantie toegelaten is en wanneer het systeem klein is. Bij deze specifieke gevallen is de fractie van de rekenkost die besteed wordt aan de set-up een groot deel van de uiteindelijke berekening. In de gevallen waar deze thesis zich op richt: nauwkeurige metingen ($\varepsilon < 1 \cdot 10^{-2}$) en relatief grote systemen (aantal componenten ≥ 20) is de fractie van de kost voor de opstart te verwaarlozen bij de uiteindelijke vergelijking.
Uit de diagnostische analyse van de berekening kunnen de voorwaarden gecontroleerd worden die opgesteld zijn in het theorema. Door de levels op te bouwen zoals voorgesteld in [2] wordt er een algemene daling gezien van de verwachte waarde van de correctieterm en de variantie van de correctieterm. Deze dalingen zijn niet monotoon, mogelijks is er een hogere variantie of hogere verwachte waarde bij een bepaald level. Dit heeft echter geen significante gevolgen voor de conclusie over de rekenkost van de Multilevel Monte Carlo berekening. Het selectie-algoritme zorgt consistent voor een kost die kwadratisch stijgt met de levels. De verlaagde kost op lagere levels samen met een verlaagde variantie op de hogere levels zorgt voor de efficiëntie van Multilevel Monte Carlo.
In de bestudeerde gevallen verloopt de rekenkost van Multilevel Monte Carlo hetzelfde als rekenkost van Monte Carlo maar er is een verschil van een constante factor.



Figuur 5.23: Waterkrachtcentrale, overlevingssignatuur P(T>t)

5.4.2 Multilevel Quasi-Monte Carlo

De resultaten van de Multilevel Quasi-Monte Carlo schatter tonen dat de rekenkost wordt verlaagd door de introductie van Quasi-Monte Carlo in de Multilevel Monte Carlo berekening. De daling van de rekenkost bevestigd de verwachtingen die vooraf aan het onderzoek zijn gemaakt. De implementatie van MLQMC maakt gebruik van deterministisch goed gekozen punten die gebaseerd zijn op een rang-1 roosterregel met een standaard genererende vector. Door de componenten te ordenen volgens de totale orde senstiviteit worden er meer uniforme waarden genomen bij de componenten die de grootste invloed hebben. De invoer van deze ordening zorgt voor een grotere opstarttijd voordat de simulatie begint. Deze heeft echter geen invloed op de conclusie die genomen is uit de resultaten. De resultaten die gepresenteerd zijn in deze masterproef hebben steeds de efficiëntie van de methodes bevestigd. Multilevel Quasi-Monte Carlo is een snellere methode dan Multilevel Monte Carlo waarbij de rekenkost sneller verloopt bij een MLQMC in specifieke gevallen (het systeem van 20 componenten). Bij het systeem van 50 componenten is er geen verlaging van de complexiteit. Dit komt overeen met andere onderzoeken naar Quasi-Monte Carlo waarbij er opgemerkt wordt dat Quasi-Monte Carlo minder effcieënt is bij zeer hoog-dimensionale problemen [10].

Hoofdstuk 6

Besluit

Het doel van de masterproef is een efficiënte berekening van de levensduur van een systeem waarvan de verdeling van de levensduur van de componenten gekend is en waarvan een betrouwbaarheidsdiagram kan worden opgesteld. Het onderzoek is vertrokken van een algoritme geïntroduceerd door L.J.M. Aslett in [2] dat zich baseert op de verzameling van minimale snedes. Een systeem is volledig bepaald door deze verzameling en kan voorgesteld worden door een serieschakeling van de snedes uit deze verzameling. De verzameling van minimale snedes wordt opgesteld door het MOCUS algoritme toe te passen.
Multilevel Monte Carlo maakt gebruik van de benaderingen van de levensduur op verschillende levels waarbij de levels met een hogere kost ook een nauwkeurigere benadering zijn. Met de verzameling uit het MOCUS algoritme kan het selectie-algoritme uit hoofdstuk 4 zo een levelstructuur opbouwen. Deze structuur definieert op ieder level een deelverzameling van de verzameling van snedes waarbij de deelverzameling van een bepaald level ook tot de verzameling van alle hogere levels behoort. Het resultaat is een Multilevel Monte Carlo schatter die de verwachtte waarde van de levensduur berekent met dezelfde nauwkeurigheid als Monte Carlo maar met een lagere rekenkost.
De toegevoegde waarde van deze masterproef is de evolutie naar Multilevel Quasi-Monte Carlo. Het doel van deze evolutie is de rekenkost verder verlagen waarbij het ultieme doel een verlaging van de complexiteit van de rekenkost is. De toevoeging van de Quasi-Monte Carlo gebeurt door de levensduur van de componenten te bepalen volgens een rang-1 roosterregel met een standaard genererende vector. Vooraf aan de berekening gebeurt er een ordening van de componenten. De ordening gebeurt naar aanleiding van de eigenschap dat Quasi-Monte Carlo punten meer uniform verdeeld zijn in de lagere dimensies.
Deze masterproef stelt een algoritme voor dat de levelstructuur van een systeem construeert en de Multilevel (Quasi-)Monte Carlo berekening uitvoert voor een opgegeven nauwkeurigheid ε. Dit algoritme is getest op verschillende willekeurige systemen waarbij er vertrokken is van 1 component. Vervolgens worden willekeurige componenten vervangen door een serie- of parallel-schakeling of wordt er een brug gevormd tussen twee componenten. De resultaten tonen het verwachte resultaat

waarbij de Multilevel Monte Carlo berekening een rekenkost heeft die lager ligt dan de Monte Carlo berekening. De Multilevel Quasi-Monte Carlo heeft een nog lagere rekenkost dan de Multilevel Monte Carlo waarbij er een verlaging van de complexiteit te zien is zodat de asymptotische complexiteit van de rekenkost $\mathcal{O}(\varepsilon^{-p})$ met $p < 2$ is. Als laatste onderdeel van deze thesis is de implementatie toegepast op twee reële toepassingen een brandalarm en een waterkrachtcentrale. In beide gevallen ligt de rekenkost van MLMC lager dan de Monte Carlo simulatie. De rekenkost van de klassieke MC method voor het berekenen van de levensduur van een brandalarm voor een nauwkeurigheid van $\varepsilon = 5 \cdot 10^{-4}$ is 3 uur 25 minuten. Onze MLQMC methode reduceert dit naar slecht 2 minuten! Multilevel Quasi-Monte Carlo is hier een zeer efficiënt algoritme om te gebruiken, de asymptotische complexiteit van de rekenkost in beide toepassingen is lager dan $\mathcal{O}(\varepsilon^{-2})$.
Dit onderzoek ligt aan het begin van vele mogelijkheden. Er kan verder naar toepassingen gekeken worden en hoe er omgegaan wordt met afhankelijkheden tussen de componenten, bijvoorbeeld interne afhankelijkheden en “gemeenschappelijke oorzaak van falen” (“common-cause failures”).
Een andere richting voor verder onderzoek is de optimalisatie van de MLQMC methode. De implementatie van MLQMC is hier gebeurt met een rang-1 roosterregel met een standaard genererende vector. De gunstige resultaten die hier bekomen zijn wekken de interesse om dit verder te bestuderen. Een mogelijkheid is het gebruiken van andere soorten Quasi-Monte Carlo regels of andere genererende vectoren. Een andere invalshoek is het bestuderen van de complexiteit van de rekenkost van de MLQMC en de situaties waarin de complexiteit $\mathcal{O}(\varepsilon^{-p})$ met $p < 2$ is en de mogelijkheid om deze situaties te definiëren.

Bijlagen

Bijlage A

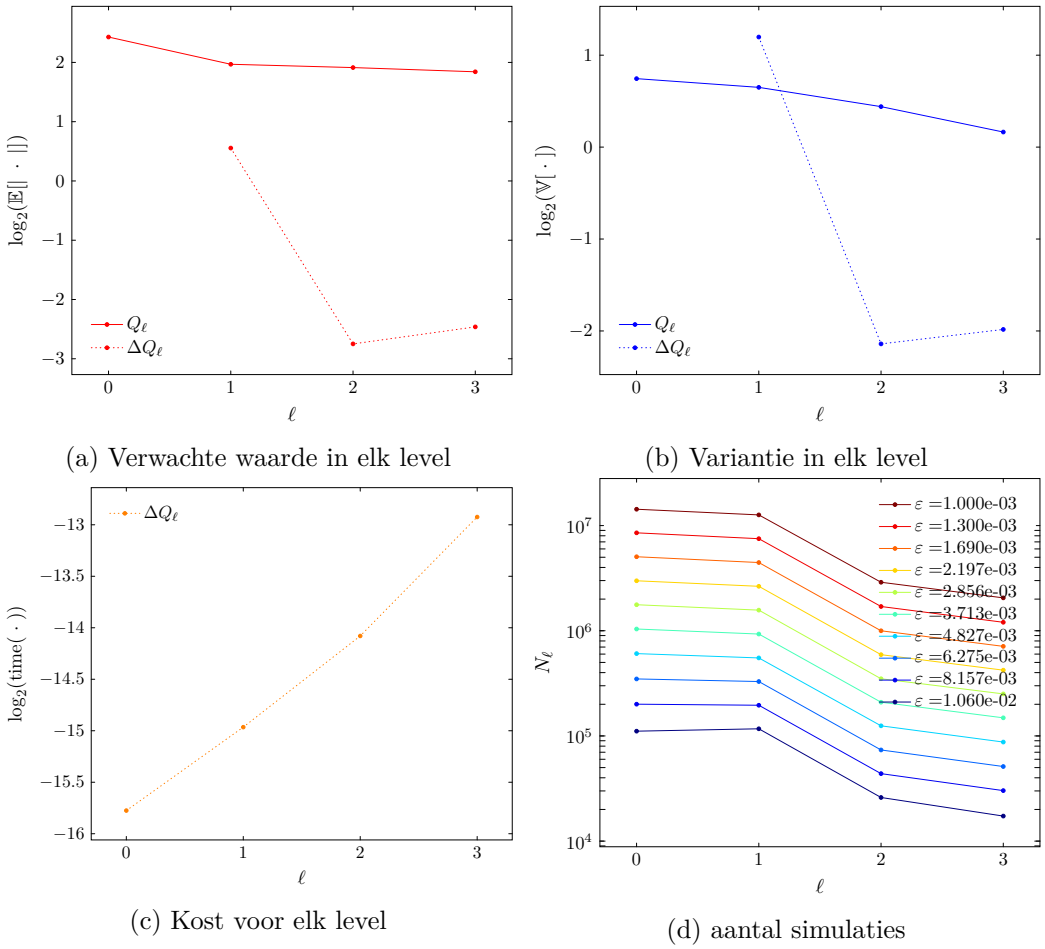
Numerieke resultaten

In deze bijlage worden de bijkomende simulaties besproken die nodig waren voor de analyse en bespreking van de resultaten in hoofdstuk 5. Het eerste deel bespreekt het systeem met 20 componenten. De tweede sectie geeft de bijkomende berekeningen van het systeem met 30 componenten. Het derde deel is de bijkomende simulatie voor het systeem met 50 componenten. Bijlage A.4 tot A.6 geven de diagnose van de MLQMC berekeningen die zijn weggelaten bij de resultaten.

A.1 Systeem met 20 componenten

A.1.1 vormparameter $k = 3$

Deze sectie geeft de diagnostische analyse en de resultaten van de berekening van de levensduur van het systeem met 20 componenten besproken in sectie 5.1.1 waarbij de vormparameter gewijzigd is naar $k = 3$. De Multilevel (Quasi-)Monte Carlo simulatie wordt uitgevoerd voor een nauwkeurigheid $\varepsilon = 10^{-3}$. Het resultaat van de berekening wordt gegeven in figuur A.1. Linksboven wordt de gemiddelde waarde $\mathbb{E}[\hat{T}_\ell]$ en de correctiefactor $\mathbb{E}[\hat{T}_\ell - \hat{T}_{\ell-1}]$ in ieder level ℓ gegeven in een semi-logaritmische grafiek. Rechtsboven is er een semi-logaritmische grafiek van de variantie in ieder level $\mathbb{V}[\hat{T}_\ell]$ en de variantie van de correctiefactor $\mathbb{V}[\hat{T}_\ell - \hat{T}_{\ell-1}]$. Figuur A.1c toont de kost in elk level $C[\hat{T}_\ell]$ en figuur A.1d toont het aantal simulaties N_ℓ uitgevoerd in level ℓ.
De figuren tonen het gewenste gedrag zoals vermeld in het theorema van Multilevel Monte Carlo. De variantie van de correctieterm is lager dan de variantie van de levensduur in de hoge levels. Dit zorgt voor een verlaging van de rekenkost. De simulatie geeft een schatting voor de parameters β en γ. Deze zijn voor dit systeem $\beta = 1.59093$ en $\gamma = 1.02179$. Het aantal simulaties dat berekend is, is hoger voor de lagere levels. Dit is verbonden met het feit dat $\beta > \gamma$, het grootste deel van de rekenkost ligt dus vooral bij de lage levels.
Figuur A.3 toont de rekenkost van het Multilevel Monte Carlo in functie van de nauwkeurigheid ε. Hierbij is de rekenkost van Monte Carlo gegeven door het product van het aantal simulaties op het laagste level met de kost van de simulatie op het hoogste niveau: $N_0 C_L$. De rekenkost van de Multilevel (Quasi-)Monte Carlo simulatie wordt gegeven door de som van het product van het aantal simulaties op elk level met



(a) Verwachte waarde in elk level

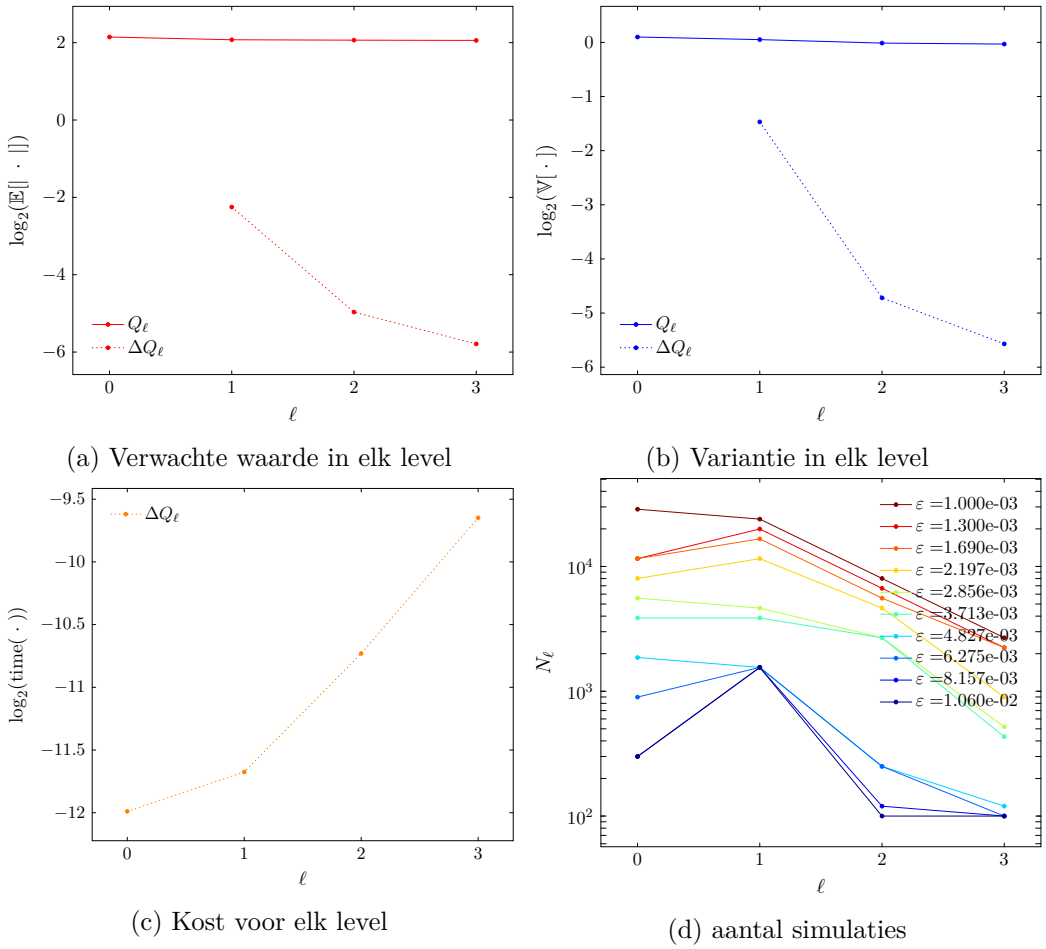
(b) Variantie in elk level

(c) Kost voor elk level

(d) aantal simulaties

Figuur A.1: 20 componenten ($k = 3$), resultaat van de Multilevel Monte Carlo schatter

de kost van een simulatie op dat level. De scatterpunten zijn de gemeten waarden van de rekenkost. De lijnen zijn regressies uitgevoerd op 3 simulaties.
De berekening met Multilevel Quasi-Monte Carlo is uitgevoerd door gebruik te maken van een rang-1 roosterregel. Hierbij zijn de componenten gerangschikt volgens de invloed van de variantie van de levensduur van het component op de variantie van de levensduur van het systeem. De diagnostische analyse is weergegeven in figuur A.2. De resultaten tonen het gewenste gedrag waarbij de daling van de variantie groter is dan de stijging van de kost. De rekenkost van MC en MLMC is kwadratisch ($\mathcal{O}(\varepsilon^{-2})$). Dit is te zien in figuur A.3. De rekenkost van MLQMC is duidelijk lager waarbij de rekenkost toeneemt volgens $\mathcal{O}(\varepsilon^{-1.2})$ wat een duidelijke verbetering is van de complexiteit.



(a) Verwachte waarde in elk level

(b) Variantie in elk level

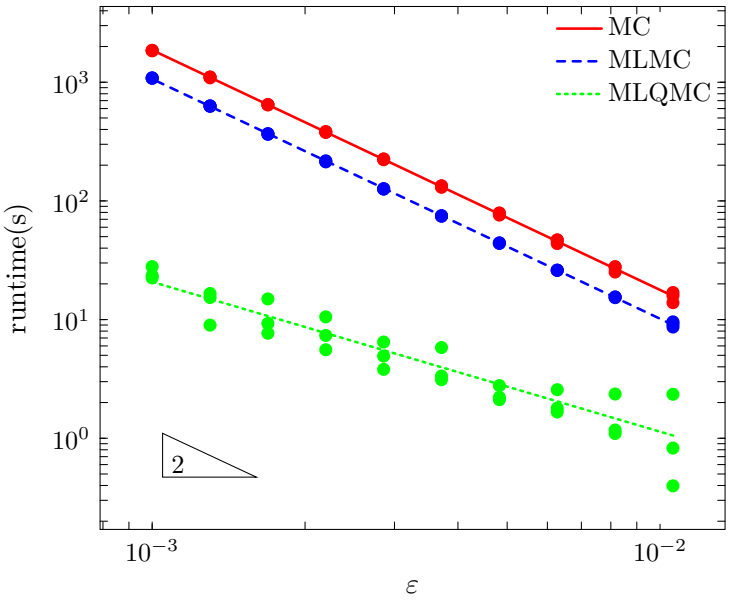
(c) Kost voor elk level

(d) aantal simulaties

Figuur A.2: 20 componenten ($k = 3$), resultaat van de Multilevel Quasi-Monte Carlo schatter

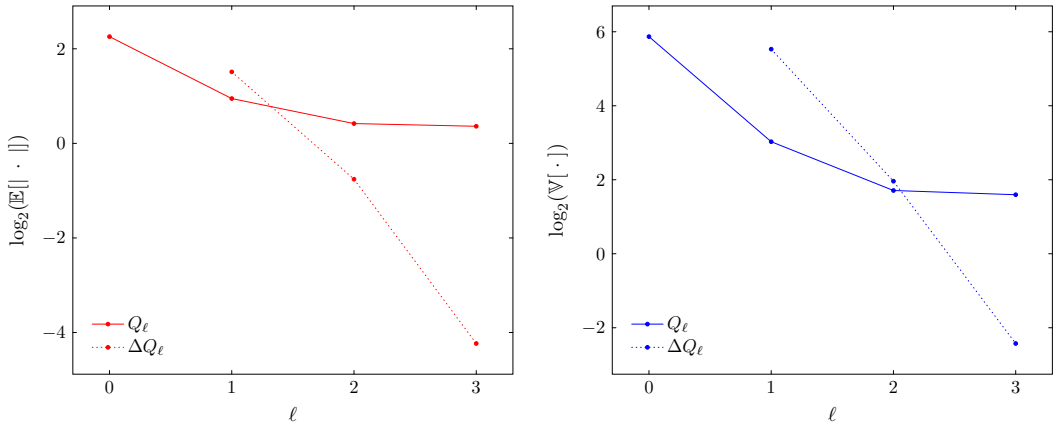
A.1.2 vormparameter $k = 0.5$

Deze sectie geeft de diagnostische analyse en de resultaten van de berekening waarbij de vormparameter gewijzigd is naar $k = 0.5$. De Multilevel (Quasi-)Monte Carlo berekening wordt uitgevoerd voor een nauwkeurigheid $\varepsilon = 10^{-3}$. Het resultaat van de simulatie wordt gegeven in figuur A.4.
De gemiddelde waarde en variantie van de correctieterm daalt (figuur A.4a-A.4b) en de kost van de berekening in een level stijgt (figuur A.4c). Dit bevestigt de voorwaarden in het theorema van Multilevel Monte Carlo. De simulatie geeft een schatting voor de parameters $\beta = 3.98549$ en $\gamma = 1.0381$. In de figuur is het opmerkelijk dat de variantie van de correctieterm bij level $\ell = 1$ en level $\ell = 2$ hoger is dan de variantie van de levensduur op dit level. De methode zou hier efficiënter worden als de levelstructuur wordt aangepast en de levensduur wordt berekend door enkel gebruik te maken van de laatste twee levels ($\mathbb{E}[T_s] = \mathbb{E}[\hat{T}_2] + \mathbb{E}[\hat{T}_3 - \hat{T}_2]$). De diagnostische analyse van Multilevel Quasi-Monte Carlo berekening is weergegeven



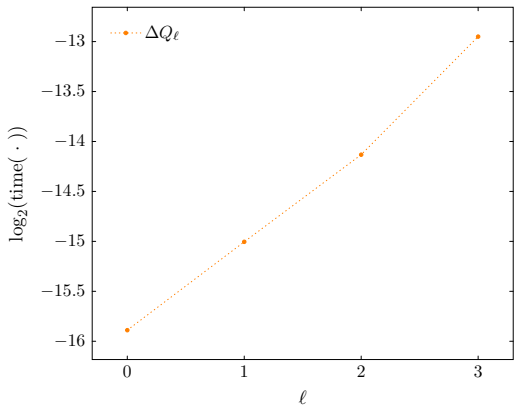
Figuur A.3: 20 componenten ($k = 3$), rekenkost

in figuur A.5. De resultaten tonen het gewenste gedrag waarbij de parameters geschat zijn: $\beta = 3.46474$ en $\gamma = 0.99965$. Figuur A.6 toont de rekenkost van Monte Carlo, Multilevel Monte Carlo en Multilevel Quasi-Monte Carlo in functie van de nauwkeurigheid ε. De Multilevel Monte Carlo methode is 2.5 à 3 keer sneller dan Monte Carlo. Het verloop van de rekenkost is $\mathcal{O}(\varepsilon^{-2})$. Er een duidelijke reductie van de rekenkost als Quasi-Monte Carlo simulaties worden gebruikt. de berekening met MLQMC is tien keer sneller dan MLMC. De rekenkost van MLQMC heeft volgens de berekeningen op dit systeem een complexiteit $\mathcal{O}(\varepsilon^{-1.8})$

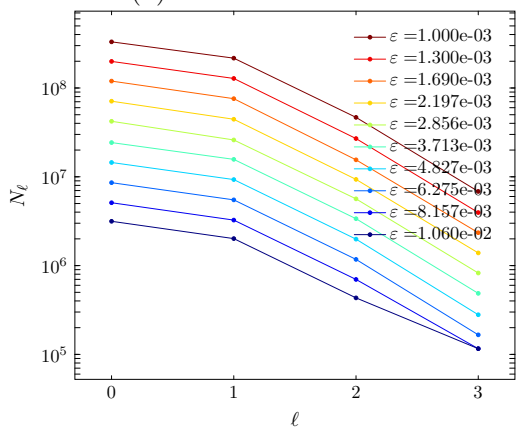


(a) Verwachte waarde in elk level

(b) Variantie in elk level

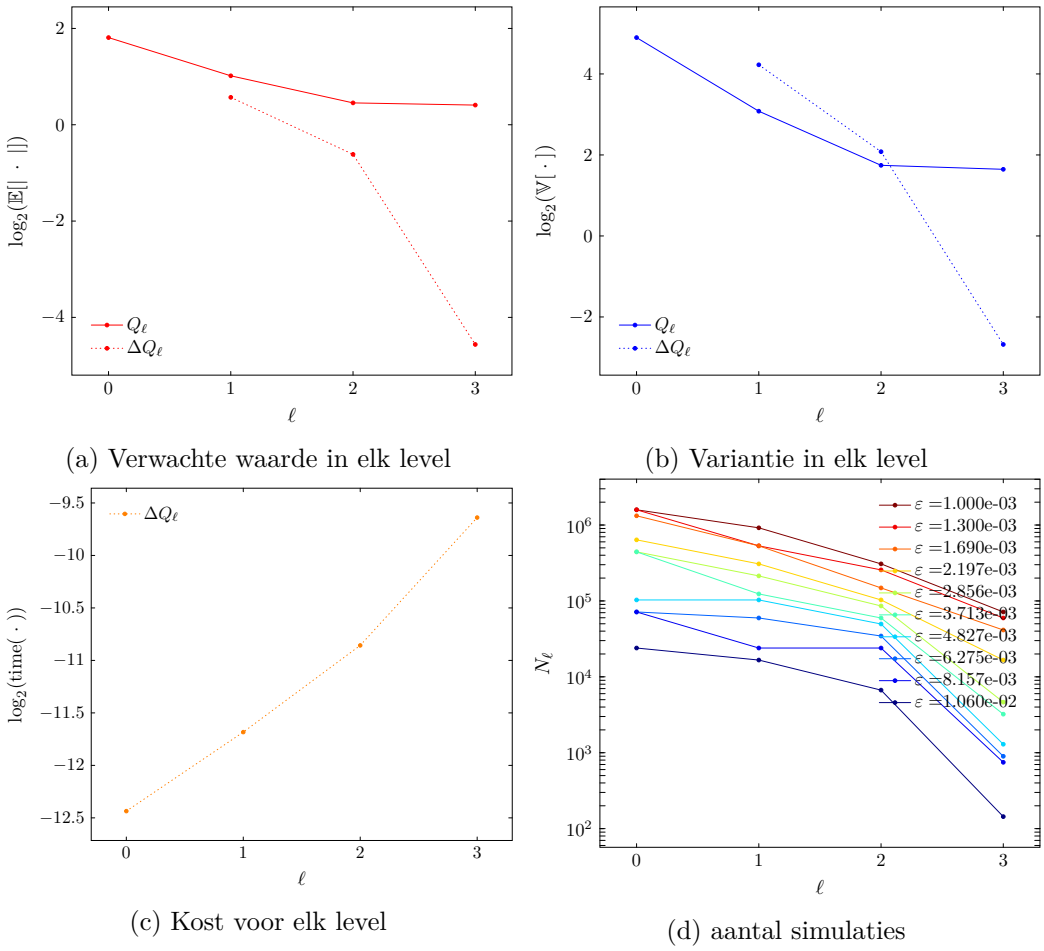


(c) Kost voor elk level



(d) aantal simulaties

Figuur A.4: 20 componenten ($k = 0.5$), resultaat van de Multilevel Monte Carlo schatter



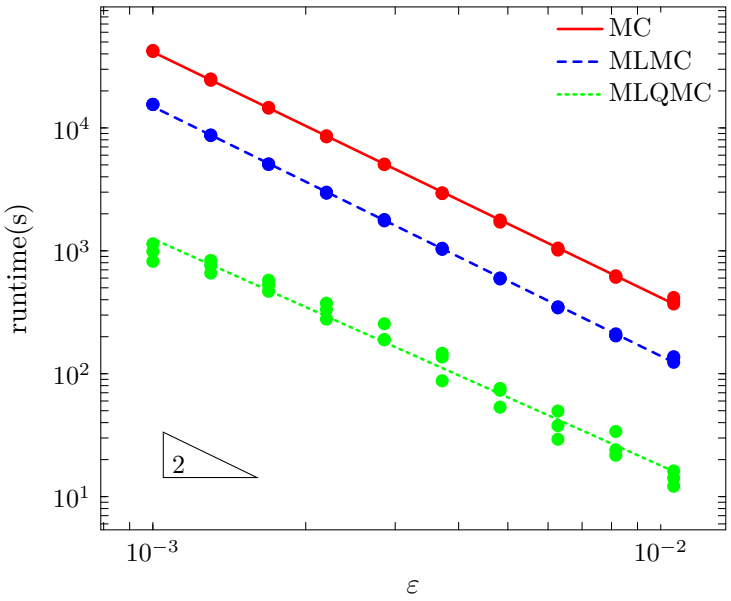
(a) Verwachte waarde in elk level

(b) Variantie in elk level

(c) Kost voor elk level

(d) aantal simulaties

Figuur A.5: 20 componenten ($k = 0.5$), resultaat van de Multilevel Quasi-Monte Carlo schatter



Figuur A.6: 20 componenten ($k = 3$), rekenkost

A.2 Systeem met 30 componenten

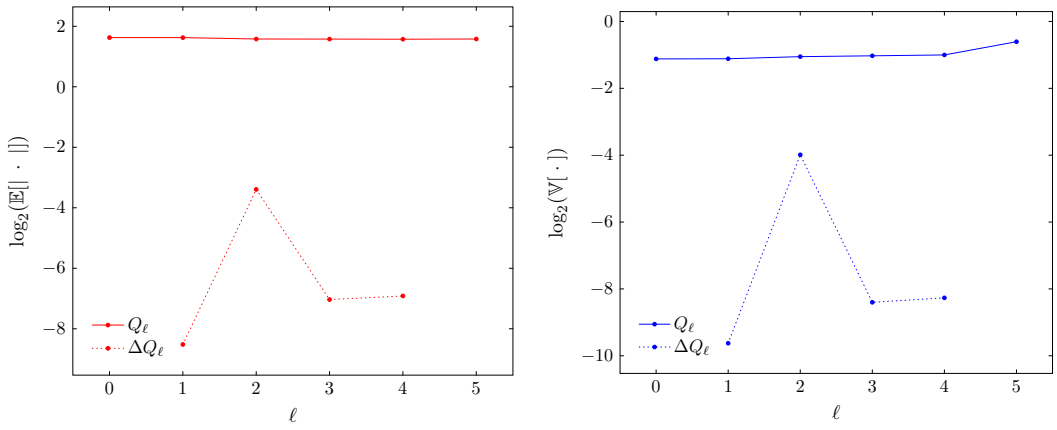
Deze sectie geeft de diagnostische analyse en de resultaten van de berekening van de levensduur van het systeem met 30 componenten besproken in sectie 5.1.2. De vormparameter is opnieuw gewijzigd naar $k = 3$ (sectie A.2.1) en naar $k = 0.5$ (sectie A.2.2).

A.2.1 vormparameter $k = 3$

De berekeningen zijn uitgevoerd met een nauwkeurigheid $\varepsilon = 10^{-3}$. Het resultaat van de schatter wordt gegeven in figuur A.7.
De daling van de variantie van de correctieterm is hier niet aanwezig. De simulatie geeft een schatting voor de parameters β en γ. Deze zijn voor dit systeem $\beta = 0.177414$ en $\gamma = 1.02418$. De waarde voor γ toont dat het selectie-algoritme consistent zorgt voor de verdubbeling in de kost ten opzichte van een vorig level. De situatie hier is dat $\beta < \gamma$, dit is niet de ideale situatie voor Multilevel Monte Carlo. In de figuur is de variantie van de correctieterm op level 1 zeer laag. Dit verklaart de lage waarde voor β. Figuur A.12 toont de rekenkost van het Multilevel Monte Carlo in functie van de nauwkeurigheid ε.
De diagnostische analyse van Multilevel Quasi-Monte Carlo schatter is weergegeven in figuur A.8. De resultaten tonen het gewenste gedrag waarbij de parameters geschat zijn: $\beta = 1.30849$ en $\gamma = 1.01873$, hier daalt de variantie wel waarbij de waarde van β licht hoger is dan γ. In de vergelijking tussen Monte Carlo, Multilevel Monte Carlo en Multilevel Quasi-Monte Carlo in figuur A.9 is er een duidelijke reductie van de rekenkost als Multilevel (Quasi-)Monte Carlo simulaties worden gebruikt. De rekenkost van elke methode groeit volgens $\mathcal{O}(\varepsilon^{-2})$.

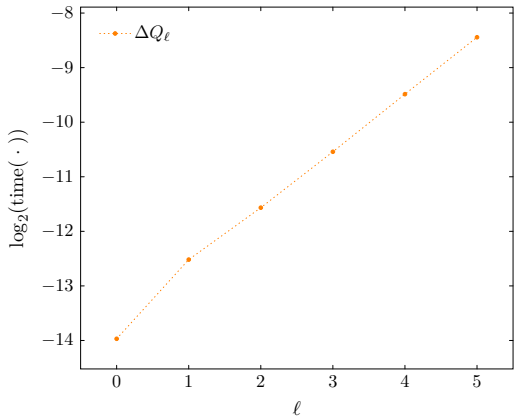
A.2.2 vormparameter $k = 0.5$

De Multilevel (Quasi-)Monte Carlo berekening wordt uitgevoerd voor een nauwkeurigheid $\varepsilon = 5 \cdot 10^{-3}$. Het resultaat van de MLMC schatter wordt gegeven in figuur A.10. De figuren tonen het gedrag zoals vermeld in het theorema van Multilevel Monte Carlo. De schatter geeft een schatting voor de parameters β en γ. Deze zijn voor dit systeem $\beta = 1.94468$ en $\gamma = 1.03996$. Het aantal uitgevoerde simulaties is het hoogst bij level $\ell = 0$ en daalt bij de hogere levels. Verbonden met het feit dat $\beta > \gamma$ wordt de rekenkost vooral bepaald door de lage levels. De verwachte complexiteit is $\mathcal{O}(\varepsilon^{-2})$. Figuur A.12 toont de rekenkost van het Multilevel Monte Carlo in functie van de nauwkeurigheid ε en bevestigt deze verwachting.
De diagnostische analyse van Multilevel Quasi-Monte Carlo simulatie is weergegeven in figuur A.5. De resultaten tonen het gewenste gedrag waarbij de parameters geschat zijn: $\beta = 2.52899$ en $\gamma = 1.04073$. In de vergelijking tussen Standaard Monte Carlo, Multilevel Monte Carlo en Multilevel Quasi-Monte Carlo in figuur A.6 is er een duidelijke reductie van de rekenkost als Quasi-Monte Carlo simulaties worden gebruikt. De MLQMC berekening heeft een rekenkost die groeit volgens $\mathcal{O}(\varepsilon^{-2})$.

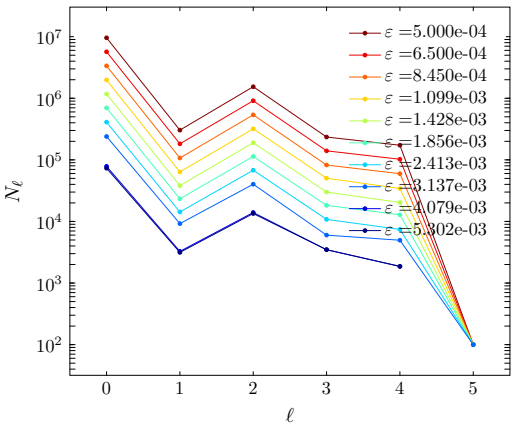


(a) Verwachte waarde in elk level

(b) Variantie in elk level

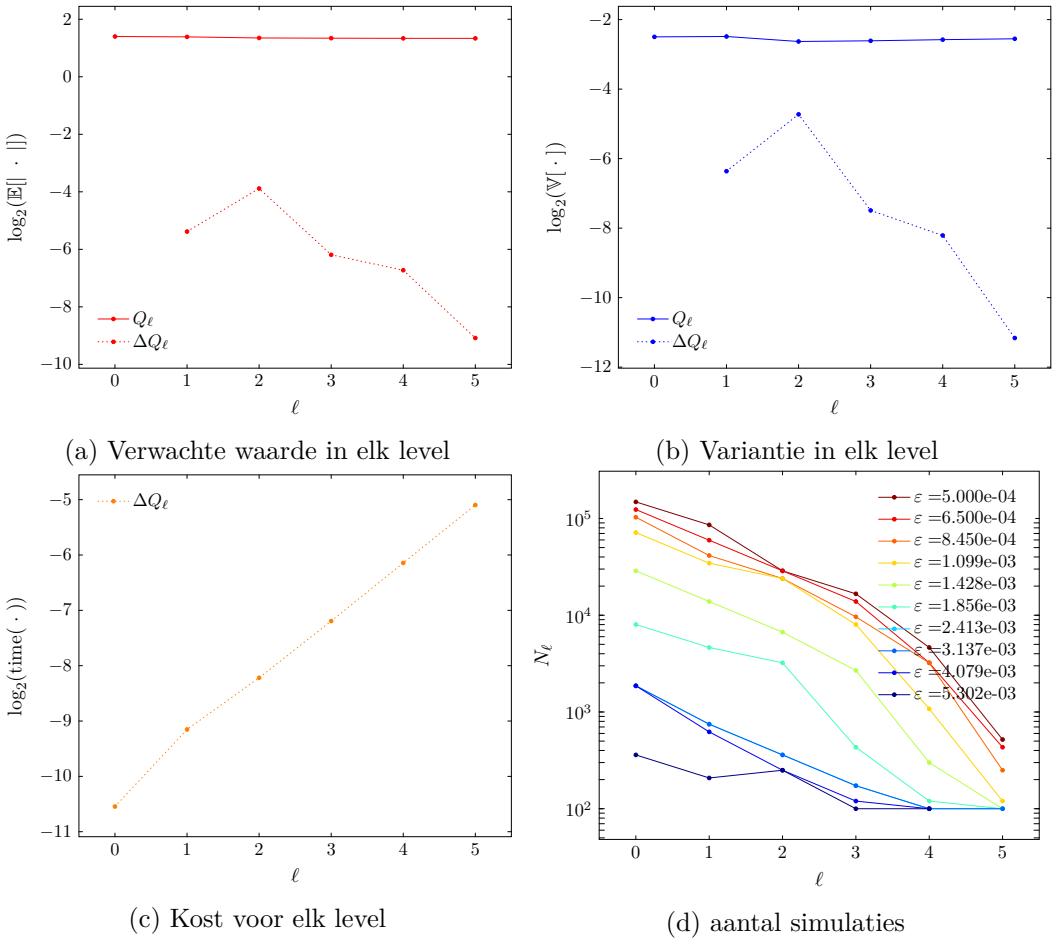


(c) Kost voor elk level



(d) aantal simulaties

Figuur A.7: 30 componenten ($k = 3$), diagnose van de Multilevel Monte Carlo simulatie



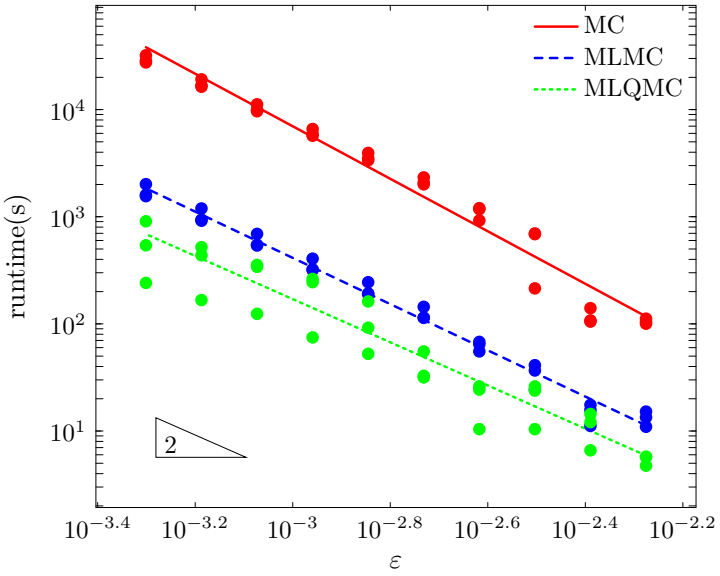
(a) Verwachte waarde in elk level

(b) Variantie in elk level

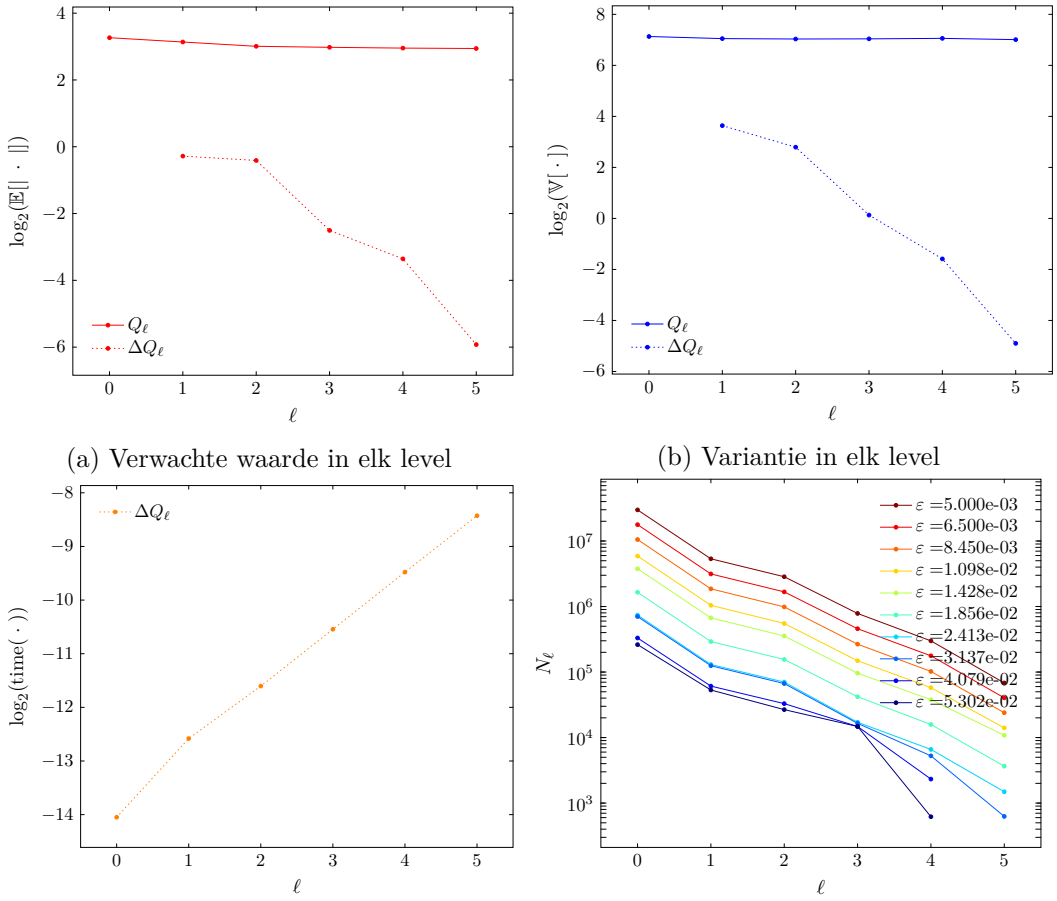
(c) Kost voor elk level

(d) aantal simulaties

Figuur A.8: 30 componenten ($k = 3$), diagnose van de Multilevel Quasi-Monte Carlo simulatie



Figuur A.9: 30 componenten ($k = 3$), rekenkost



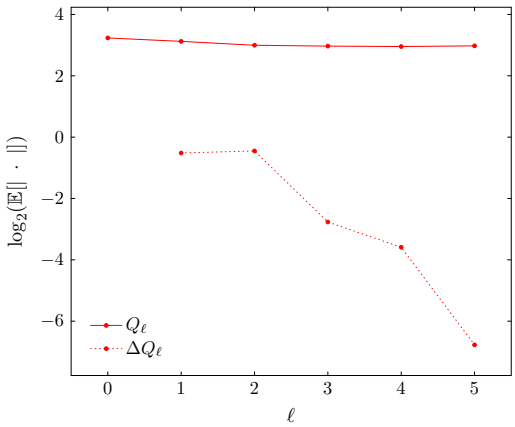
(a) Verwachte waarde in elk level

(b) Variantie in elk level

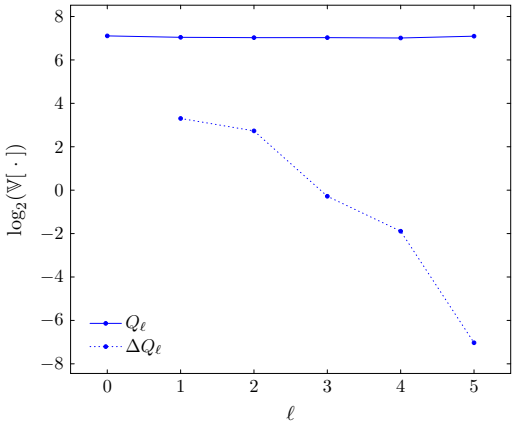
(c) Kost voor elk level

(d) aantal simulaties

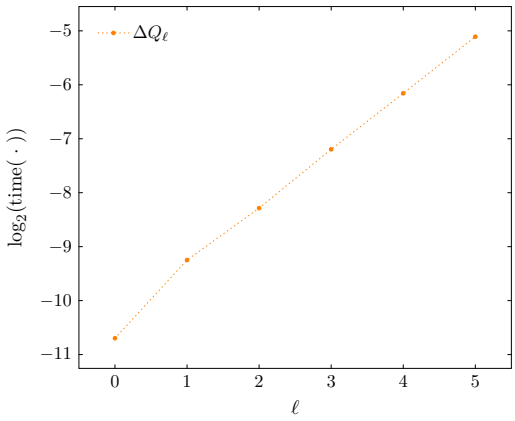
Figuur A.10: 30 componenten ($k = 0.5$), resultaat van de MLMC schatter



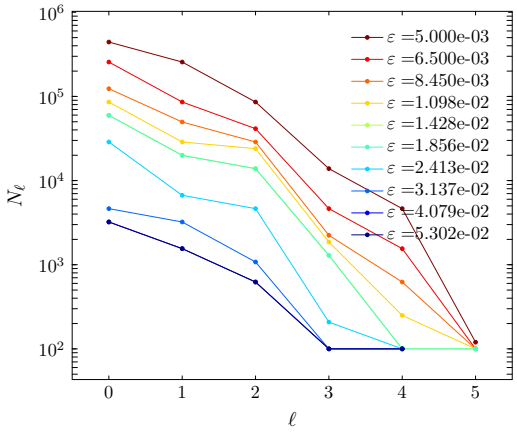
(a) Verwachte waarde in elk level



(b) Variantie in elk level

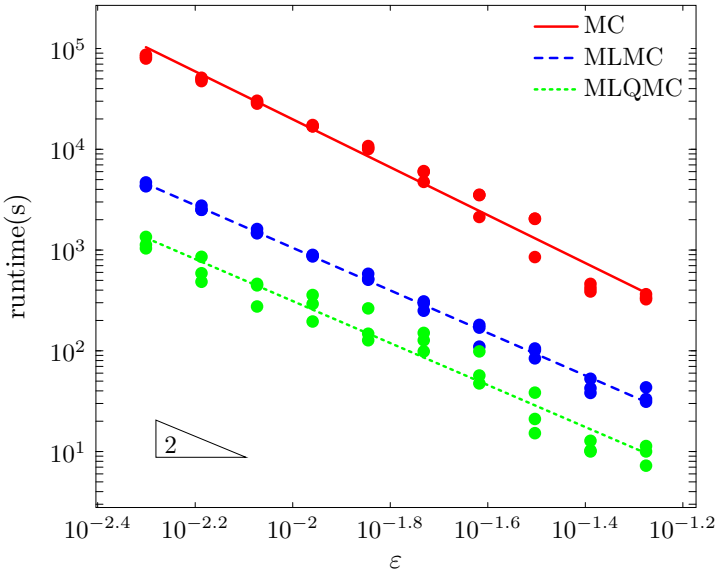


(c) Kost voor elk level



(d) aantal simulaties

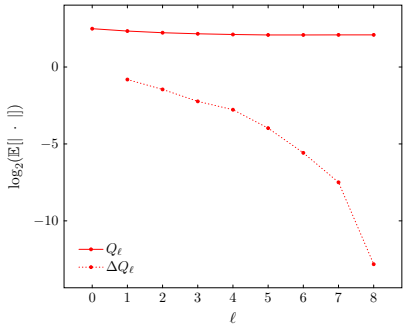
Figuur A.11: 30 componenten ($k = 0.5$), resultaat van de MLQMC schatter



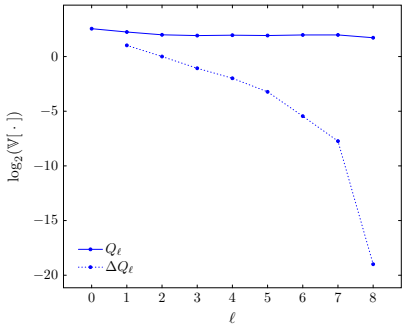
Figuur A.12: 30 componenten ($k = 0.5$), rekenkost

A.3 Systeem met 50 componenten

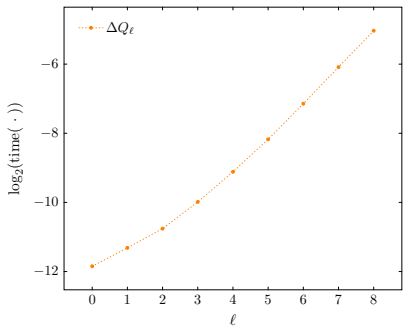
De berekening van de levensduur van het systeem in sectie 5.1.3 is hier herhaald voor een vormparameter $k = 1$. De Multilevel Monte Carlo berekening wordt uitgevoerd voor een nauwkeurigheid $\varepsilon = 5 \cdot 10^{-3}$. Het resultaat van de schatter wordt gegeven in figuur A.13.
De figuren tonen het gewenste gedrag zoals vermeld in het theorema van Multilevel Monte Carlo. De simulatie geeft een schatting voor de parameters β en γ. Deze zijn voor dit systeem $\beta = 1.88118$ en $\gamma = 0.976373$, $\beta > \gamma$ dus de grootste fractie van de rekenkost is bij de lage levels. Figuur A.15 toont de rekenkost van het Multilevel Monte Carlo in functie van de nauwkeurigheid ε.



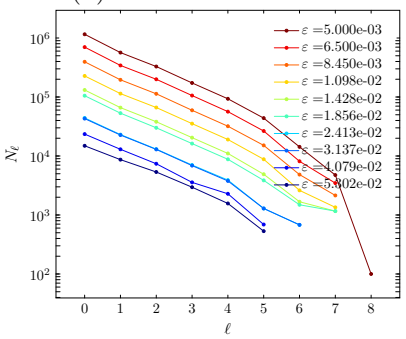
(a) Verwachte waarde in elk level



(b) Variantie in elk level



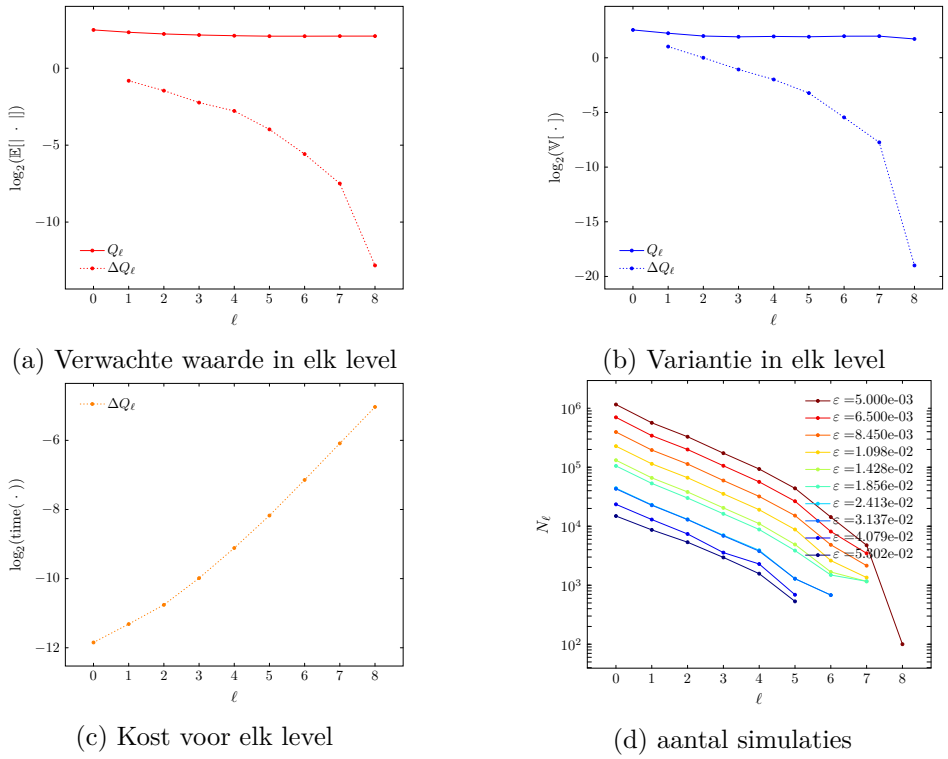
(c) Kost voor elk level



(d) aantal simulaties

Figuur A.13: 50 componenten ($k = 1$), resultaat van de Multilevel Monte Carlo schatter

De diagnose van de Multilevel Quasi-Monte Carlo berekening is weergegeven in figuur A.14. De resultaten tonen het gewenste gedrag waarbij parameter $\beta = 1.64271$ en parameter $\gamma = 1.0235$. In figuur A.15 wordt de rekenkost van Monte Carlo, Multilevel Monte Carlo en Multilevel Quasi-Monte Carlo weergegeven. Er is een bijkomende reductie van de rekenkost als Quasi-Monte Carlo simulaties worden gebruikt. In dit geval is er geen verlaging van de complexiteit.



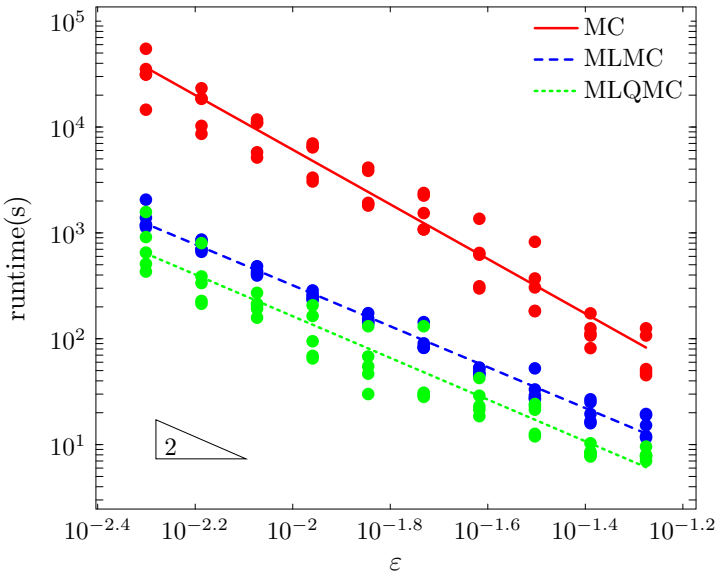
(a) Verwachte waarde in elk level

(b) Variantie in elk level

(c) Kost voor elk level

(d) aantal simulaties

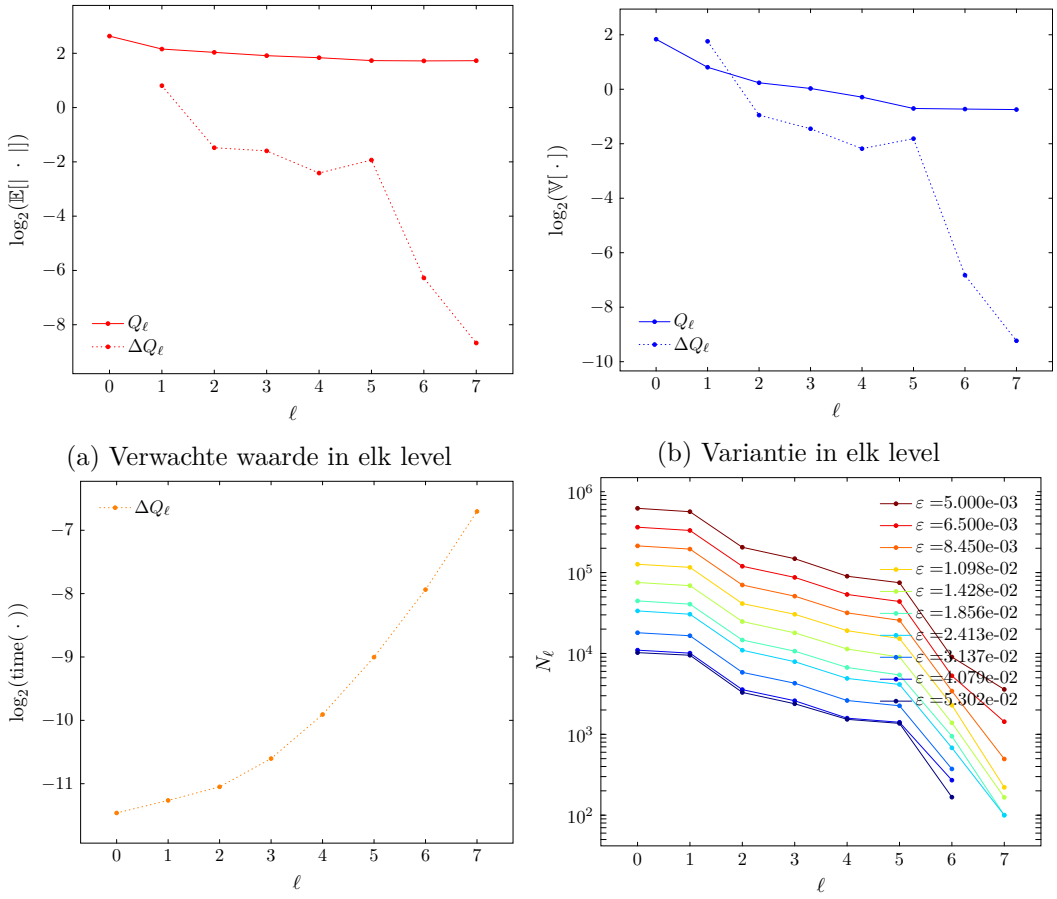
Figuur A.14: 50 componenten ($k = 1$), resultaat van de Multilevel Quasi-Monte Carlo schatter



Figuur A.15: 50 componenten ($k = 1$), rekenkost

A.4 Systeem met 70 componenten

Een bijkomend systeem waarop de implementatie is uitgevoerd is een systeem van 70 componenten. Hierbij is opnieuw vertrokken van een systeem met 1 component waarbij er het component telkens vervangen wordt door een serieschakeling of parallelschakeling van twee componenten volgens een bepaalde kans. Figuur A.16 toont de afname van de variantie en de toename van de kost per level. De verwachte waarde van de correctieterm daalt en het aantal simulaties is groter op de lage levels. Figuur A.17 toont de vergelijking tussen de standaard Monte Carlo, Multilevel Monte Carlo en Multilevel Quasi-Monte Carlo. Hierbij is er een duidelijke reductie van de rekenkost als men overgaat op de Multilevel setting en gebruik maakt van Quasi-Monte Carlo.



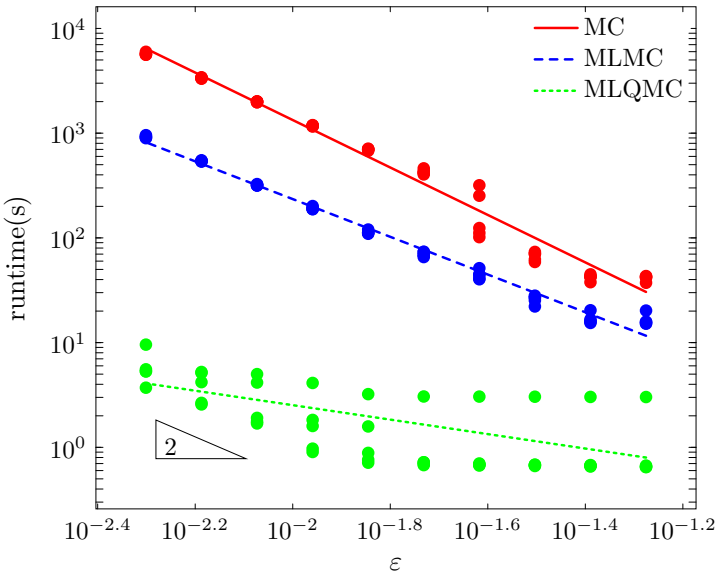
(a) Verwachte waarde in elk level

(b) Variantie in elk level

(c) Kost voor elk level

(d) aantal simulaties

Figuur A.16: 70 componenten, resultaat van de Multilevel Monte Carlo simulatie

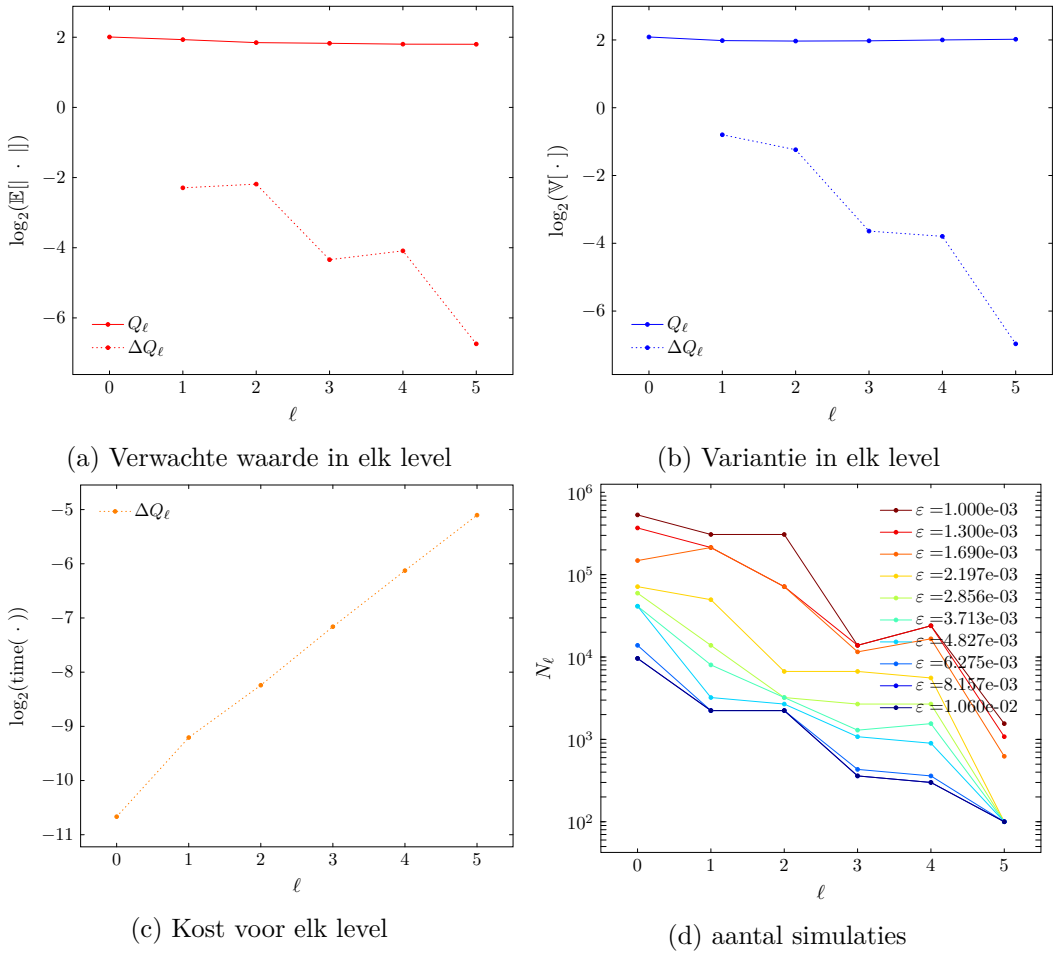


Figuur A.17: 70 componenten, rekenkost

A.5 MLQMC van het systeem met 30 componenten ($k = 1$)

Deze bijlage toont de diagnose van de MLQMC berekening van de levensduur van het systeem met 30 componenten en een vormparameter $k = 1$. De waarde en de variantie van de correctieterm neemt af en de kost van een simulatie verdubbelt ten opzichte van het vorige level. De parameters van de voorwaarden van het theorema zijn:

- $\alpha = 0.955502$
- $\beta = 1.33157$
- $\gamma = 1.03898$



(a) Verwachte waarde in elk level

(b) Variantie in elk level

(c) Kost voor elk level

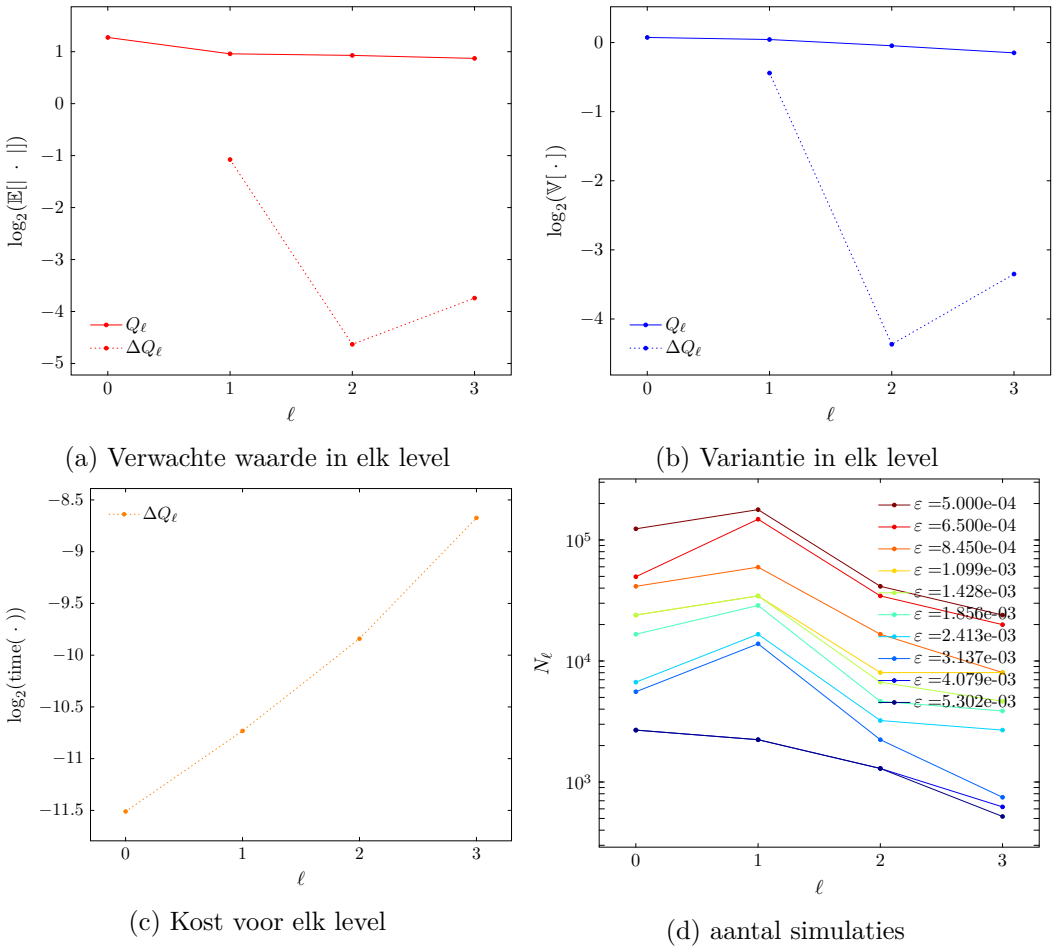
(d) aantal simulaties

Figuur A.18: 30 componenten, diagnose van de Multilevel Quasi-Monte Carlo berekening

A.6 MLQMC van het systeem met uniforme verdelingen

Deze bijlage toont de diagnose van de MLQMC berekening van de levensduur van het systeem met 30 componenten en een vormparameter $k = 1$. De waarde en de variantie van de correctieterm neemt af en de kost van een simulatie verdubbelt ten opzichte van het vorige level. De parameters van de voorwaarden van het theorema zijn:

- $\alpha = 1.33294$
- $\beta = 1.45419$
- $\gamma = 1.02905$



(a) Verwachte waarde in elk level

(b) Variantie in elk level

(c) Kost voor elk level

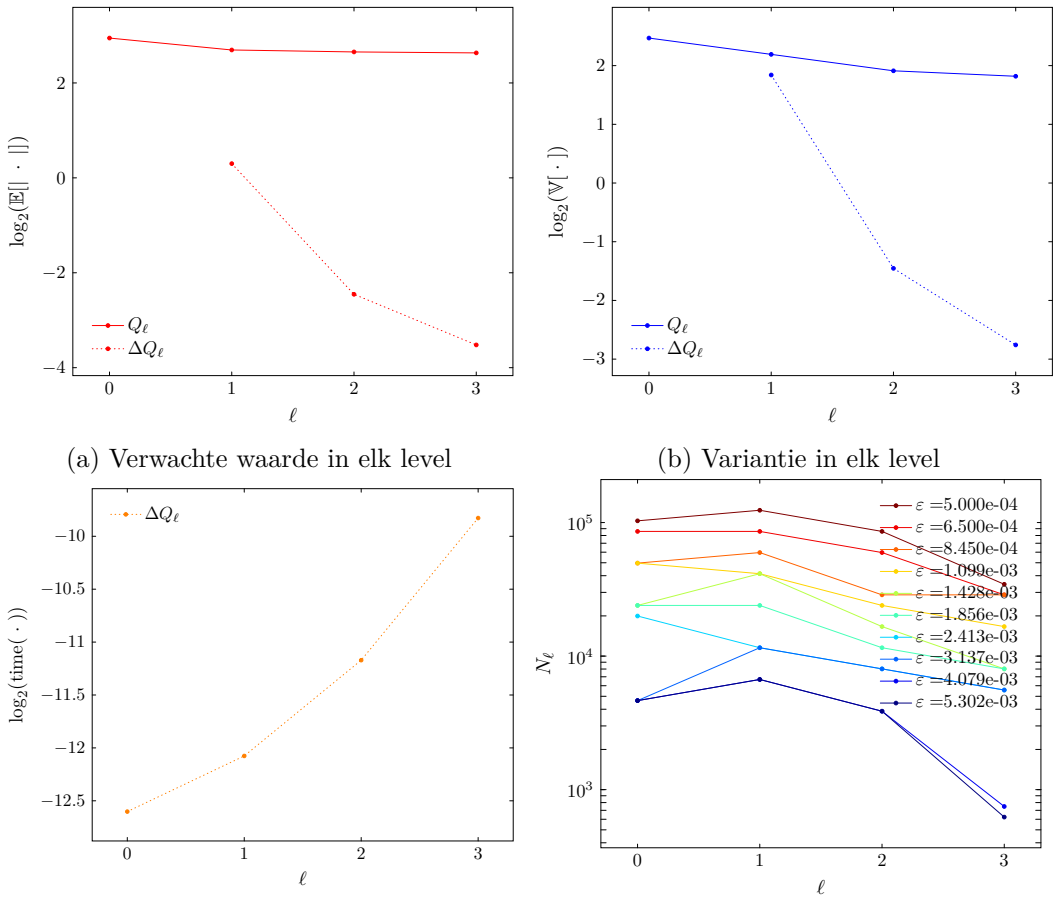
(d) aantal simulaties

Figuur A.19: Uniforme verdeling, diagnose van de Multilevel Quasi-Monte Carlo berekening

A.7 Brandalarm

Hier wordt de diagnose van de MLQMC berekening van de levensduur het brandalarm gegeven. De waarde en de variantie van de correctieterm neemt af en de kost van een simulatie verdubbelt ten opzichte van het vorige level. De parameters van de voorwaarden van het theorema zijn:

- $\alpha = 1.91012$
- $\beta = 2.29931$
- $\gamma = 1.12429$



(a) Verwachte waarde in elk level

(b) Variantie in elk level

(c) Kost voor elk level

(d) aantal simulaties

Figuur A.20: Brandalarm, diagnose Multilevel Quasi-Monte Carlo

Bijlage B

Code in Julia

Dit deel van de bijlage geeft een volledig script dat gebruikt kan worden om de levenduur van een willekeurig systeem G te bepalen via Multilevel (Quasi-)Monte Carlo. Voor de uitvoering van dit script moet er gebruik gemaakt worden van enkele pakketten:

- MultilevelEstimators.jl [22]: dit is het pakket van de MLMC schatter.
- Reporter.jl [22]: dit pakket wordt gebruikt voor het opstellen van de figuren van de diagnose.
- JLD2 [12]: dit wordt gebruikt voor het opslaan en uitlezen van de informatie van het systeem en van de resultaten van de schatter.

Bijkomend moeten de modules Systems.jl en MultilevelSystems.jl uit [26] worden geïnstalleerd. Deze modules bevatten de code voor het opbouwen en definiëren van de systemen en de simulatiefunctie.

script

```
using JLD2,Systems,MultilevelEstimators,MultilevelSystems,Reporter
import MultilevelSystems.level_levensduur,
                            MultilevelSystems.sample_levensduur

## Voorbereiding
start = time();
# willekeurig systeem
G = generateSystem("G",20,0.4,0.5,1.);
println("Generated a system. Number of components: ", G.aantal);
println("system: ",G.circuit);
T = Component[];
# bepalen levelstructuur en ordening
Levels,T,varT = levels_for_QMC(G);
println("Levels en Sampling voor QMC voltooid");
```

```
println(T);println(varT);
println("Aantal levels: ",length(Levels));
lijst = G.lijst;
# simulatiefunctie
level_levensduur(level,W)=MultilevelSystems.lvl_levensduur(level
+one(level),W,G,Levels,lijst);
levelQ_levensduur(level,W) = MultilevelSystems.lvl_levensduur(level
+one(level),W,G,Levels,T);
# verdelingen van de componenten
function distribution_levensduur(G::System,lijst)
    distributions = [MultilevelEstimators.Weibull()
     for i in 1:length(lijst)];
    for i = 1:length(lijst);
    k = lijst[i].shape;
    l = lijst[i].scale;
     distributions[i] = MultilevelEstimators.Weibull(k,l);
    end
    return distributions;
end
# bepaling duur van de set-up
elapsed = time() - start;
display(elapsed);
# nauwkeurigheid
tol=1e-3;
id = 1;

## MLMC berekening
distributions = distribution_levensduur(G,lijst);
distributionsQ = distribution_levensduur(G,T);
estimator = MultilevelEstimators.Estimator(ML(), QMC(),
   levelQ_levensduur, distributionsQ,
   name="levensduur_$(id)_MLQMC", max_index_set_param=
   length(Levels)-1, nb_of_warm_up_samples = 100);
h1 = run(estimator, tol);

## MLQMC berekening
estimator = MultilevelEstimators.Estimator(ML(), MC(),
   level_levensduur, distributions,
   name="levensduur_$(id)_MLMC", max_index_set_param=
   length(Levels)-1, nb_of_warm_up_samples = 100);
h2 = run(estimator, tol);

## diagnose
report(h1);report(h2);
```

```
## opslaan van het systeem en extra informatie
systeem = Dict(:systeem => G,:levels => Levels, :lijst => T,
                 :invloed => varT,:setup_time => elapsed);
@save "system_$(id).jld2" systeem;
```

Bibliografie

[1] I. Antonov and V. Saleev. An Economic Method of Computing LPτ-sequences. *USSR Computational Mathematics and Mathematical Physics*, 19(1):252 – 256, 1979.

[2] L. J. Aslett, T. Nagapetyan, and S. J. Vollmer. Multilevel Monte Carlo for Reliability Theory. *Reliability Engineering & System Safety*, 165:188–196, 2017.

[3] L. J. M. Aslett, F. P. A. Coolen, and S. P. Wilson. Bayesian Inference for Reliability of Systems and Networks Using the Survival Signature. *Risk Analysis*, 35(9):1640–1651, 2015.

[4] M. B. Giles. Multilevel Monte Carlo Path Simulation. *Operations Research*, 56:607–617, 06 2008.

[5] R. Barlow and F. Proschan. *Statistical Theory of Reliability and Life Testing: Probability Models*. To Begin With, 1981.

[6] S. Board. *Quality Vocabulary. Availability, Reliability and Maintainability terms. Guide to Concepts and Related Definitions*. BSI, 1991.

[7] R. E. Caflisch. Monte Carlo and Quasi-Monte Carlo Methods. *Acta Numerica*, 7:1–49, 1998.

[8] J. Dick and F. Y. Kuo. Constructing Good Lattice Rules with Millions of Points. In H. Niederreiter, editor, *Monte Carlo and Quasi-Monte Carlo Methods 2002*, pages 181–197, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[9] H. George-Williams, G. Feng, F. PA Coolen, M. Beer, and E. Patelli. Extending the Survival Signature Paradigm to Complex Systems with Non-Repairable Dependent Failures. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, page 1748006X1880808, 11 2018.

[10] M. B. Giles. Multilevel Monte Carlo Methods. *Acta Numerica*, 24:259–328, 2015.

[11] S. Heinrich. Monte Carlo Complexity of Global Solution of Integral Equations. *Journal of Complexity*, 14(2):151 – 175, 1998.

[12] T. E. Holy and contributors. *JLD2 Package*. JuliaIO, 2017.

[13] S. Joe and F. Y. Kuo. Remark on Algorithm 659: Implementing Sobol's Quasi-random Sequence Generator. *ACM Trans. Math. Softw.*, 29:49–57, 03 2003.

[14] S. Joe and F. Y. Kuo. Constructing Sobol Sequences with Better Two-Dimensional Projections. *SIAM. J. Sci. Comput.*, 30, 01 2008.

[15] F. Kuo. *Lattice Rule Generating Vectors*, 2007.

[16] F. Y. Kuo and D. Nuyens. Practical Guide to Quasi-Monte Carlo Methods. 2016.

[17] F. Y. Kuo and I. H. Sloan. Lifting the Curse of Dimensionality. *Notices of the AMS*, 52:1320–1329, 2005.

[18] P. L Ecuyer. Quasi-Monte Carlo Methods in Finance. *Proceedings - Winter Simulation Conference*, 2:1645– 1655 vol.2, 01 2005.

[19] W. Morokoff. Generating Quasi-Random Paths for Stochastic Processes. *SIAM Review*, 40(4):765–788, 1998.

[20] D. Nuyens and R. Cools. Fast Algorithms for Component-by-Component Construction of Rank-1 Lattice Rules in Shift-Invariant Reproducing Kernel Hilbert Spaces. *Mathematics of Computation*, 75(254):903–920, 2006.

[21] M. Rausand and A. Hoyland. *System Reliability Theory: Models, Statistical Methods, and Applications, Second Edition*, volume 396. Wiley, 01 2004.

[22] P. Robbe. *MultilevelEstimators.jl Documentation.* KU Leuven dept. Computerwetenschappen, 2018.

[23] A. Saltelli, D. Albrecht, S. Tarantola, and F. Ferretti. A New Sample-based Algorithms to Compute the Total Sensitivity Index. *arXiv e-prints*, page arXiv:1703.05799, Mar 2017.

[24] S. Shetty. Determining the Availability and Reliability of Storage Configurations, 2002.

[25] Szechtman. Control variates techniques for monte carlo simulation. In *Proceedings of the 2003 Winter Simulation Conference, 2003.*, volume 1, pages 144–149 Vol.1, Dec 2003.

[26] M. Vromans. *MLMC voor de Analyse van de Betrouwbaarheid van Systemen: code.* KU Leuven dept. Computerwetenschappen, 2019.

[27] B. J. Waterhouse and M. B. Giles. Multilevel Quasi-Monte Carlo Path Simulation. *Radon Series Comp. Appl. Math*, 8:1–18, 2009.

[28] W. Zamojski, Mazurkiewicz, J. J., Sugier, T. Walkowiak, and J. Kacprzyk. *Complex Systems and Dependability*, volume 170. Springer-Verlag Berlin Heidelberg, 2012.

[29] E. Zio. *The Monte Carlo Simulation Method for System Reliability and Risk Analysis*. Springer, 01 2013.

Fiche masterproef

Student: Michiel Vromans

Titel: Multilevel Monte Carlo voor de Analyse van de Betrouwbaarheid van Systemen

Engelse titel: Multilevel Monte Carlo for System Reliability

UDC: 621.3

Korte inhoud:
De meest algemeen toepasbare techniek voor de berekening van de gemiddelde tijd tot aan falen van systemen is Monte Carlo waarbij er gebruik gemaakt wordt van de snedes van het systeem. De rekenkost van deze techniek verloopt volgens $\mathcal{O}(\varepsilon^{-2})$, wat voor kleine nauwkeurigheden zeer hoog is. Deze masterproef bestudeert het toepassen van Multilevel Monte Carlo voor de bepaling van de gemiddelde tijd tot aan falen waarbij er een structuur in de verzameling van snedes wordt opgebouwd met een groeiende kost en een dalende variantie. Door de levensduur op de verschillende levels te combineren wordt een berekening bekomen met dezelfde nauwkeurigheid maar met een lagere rekenkost. In dit werk wordt het algoritme uitgebreid tot Multilevel Quasi-Monte Carlo. Door deze uitbreiding wordt er in de resultaten een lagere rekenkost waargenomen voor het bepalen van de gemiddelde tijd tot aan falen van systemen dan in Multilevel Monte Carlo. Afhankelijk van de grootte van de dimensies en de variantie van de ingangsvariabelen wordt er een complexiteit $\mathcal{O}(\varepsilon^{-p})$ met $p < 2$ waargenomen.

Thesis voorgedragen tot het behalen van de graad van Master of Science in de ingenieurswetenschappen: wiskundige ingenieurstechnieken

Promotoren: Prof. dr. ir. Stefan Vandewalle
Prof. dr. ir. David Moens

Assessoren: Prof. Nigel Smart
Prof. dr. Ir. Dirk Nuyens

Begeleider: Ir. Pieterjan Robbe