ERASMUS UNIVERSITEIT ROTTERDAM

Faculteit der Economische Wetenschappen

Econometrisch Instituut

# Het Economische Lot Sizing probleem met een herproductie optie & gezamenlijke setupkosten

Bachelorscriptie Econometrie & Besliskunde

Abstract

In deze scriptie wordt het Economische Lot Sizing Probleem met een herproductie optie en gezamenlijke setupkosten behandeld, met als doel de huidige resultaten van de Silver Meal heuristiek te overtreffen. Hier zijn diverse heuristieken voor gebruikt. Het genetisch algoritme was in staat om een groot deel van de probleeminstanties optimaal op te lossen. Het maximum ligt echter nog aanzienlijk hoog wanneer het aantal startoplossingen klein zijn. Het gewogen genetisch algoritme resulteerde ook zeer goed, met als gemiddeld fout kleiner dan een 0,5%. Het maximum was veel lager vergeleken met het genetisch algoritme.

*Begeleider:* Dr. W. van den Heuvel

*Meelezer:* Dr. A.F. Gabor

*Auteur:* Bayram Kavi [294688]

17 augustus 2009
Rotterdam

# Inhoudsopgave

Hoofdstuk 1

# Inleiding

Tegenwoordig nemen steeds meer bedrijven gebruikte producten van klanten terug. De redenen hiervoor zijn wetgeving van de overheid, het milieubewust zijn van de klant en economische redenen. Dit heeft tot gevolg dat bedrijven de gebruikte producten kunnen herproduceren, zodat de herproduceerde producten weer verkocht kunnen worden. Voor de bedrijven kan dit voordelig zijn, want het herproduceren van gebruikte producten kan goedkoper zijn dan het maken van nieuwe producten. Milieubewuste klanten zullen deze producten als 'groene' producten zien wat weer concurrentievoordelen met zich mee kan brengen. Maar het nadeel van herproduceren is dat de bedrijven ook rekening moeten houden met de voorraadkosten van de producten die terugkomen. De huidige modellen voor het Economic Lot Sizing model moeten dus uitgebreid worden. Dit probleem is een uitbreiding van het klassieke Economic Lot Sizing probleem en wordt 'Economic Lot Sizing with a Remanufacturing option' genoemd (ELSR).

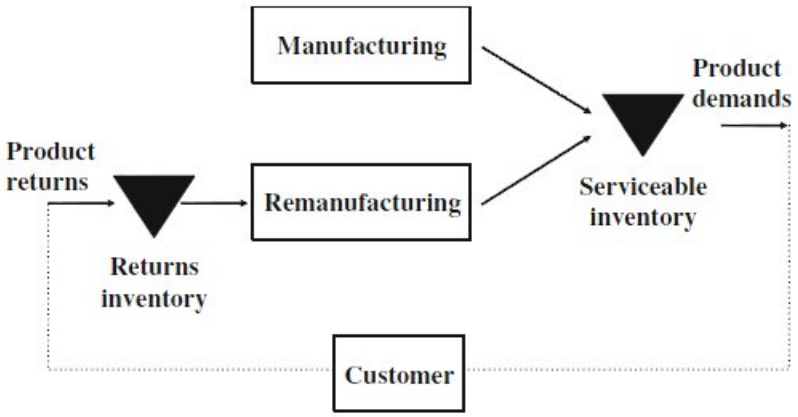
De modellen in de literatuur verschillen ook enigszins van elkaar. Er kan aangenomen worden dat er sprake is van aparte of gezamenlijke setupkosten. Herproductie kan dus plaatsvinden op dezelfde productielijn als reguliere productie of er wordt gebruik gemaakt van een aparte productielijn voor herproductie, waardoor de setupkosten voor herproductie apart genomen moeten worden. Door Richter en Sombrutzki (2000) is aangenomen dat er voldoende producten terugkomen (returns) om alle vraag te voorzien, zodat er alleen sprake is herproductie en geen productie. Richter en Weber (2001) nemen aan dat de returns in de eerste periode voldoende zijn om aan alle vraag te voldoen, wat ook in Li *et al.* (2006) is aangenomen. Al deze modellen zijn gebaseerd op aparte setupkosten. In deze scriptie gaan we uit van gezamenlijke setupkosten. De literatuur voor gezamenlijke setupkosten is voorlopig nog beperkt. In Teunter *et al.* (2006) zijn echter wel resultaten te vinden voor het probleem met gezamenlijke setupkosten. Voor het exact oplossen van het probleem is er gebruik gemaakt van het Dynamic Programming algoritme en er zijn ook bekende heuristieken zoals de Silver Meal, Least Unit Costs en Part Period Balancing gebruikt. De resultaten van de Silver Meal en Least Unit Costs zullen in deze scriptie besproken worden. Het doel van deze scriptie is om deze resultaten te verbeteren.

Deze scriptie is als volgt opgebouwd; in hoofdstuk 2 wordt een gedetailleerde beschrijving van het ELSR probleem gegeven. Vervolgens worden er in hoofdstuk 3 de verschillende methoden toegelicht. Deze methoden bestaan uit heuristieken die van een oneindige, discrete planningshorizon uitgaan en metaheuristieken. In hoofdstuk 4 worden de resultaten van de methoden behandeld met behulp van een gevoeligheidsanalyse, waarbij verschillende vraag-, return- en kostenpatronen gebruikt gaan worden. Tot slot worden de belangrijkste bevindingen samengevat in hoofdstuk 5.

Hoofdstuk 2

# Probleemformulering

Het probleem dat in deze scriptie behandeld wordt in de literatuur 'Economic Lot Sizing with a Remanufacturing option and joint setup costs' genoemd (ELSR). Deze wordt als volgt beschreven (Teunter *et al.* (2006); de vraag naar producten in elke periode is bekend, verder staan de returns per periode vast. Voor zowel de vraag als de returns is er sprake van een eindige, discrete planningshorizon. De returns kunnen herproduceerd worden om aan de vraag te voldoen in elk periode. Er wordt enkel geproduceerd als er geen voorraad meer is, dit wordt in de literatuur de zero-inventory eigenschap genoemd. Tevens kunnen klanten geen onderscheid maken tussen geproduceerde en herproduceerde producten, dit is aangenomen. Geproduceerde en herproduceerde producten worden aangeduid als servicables, producten die gereed zijn voor verkoop. Verder wordt aangenomen dat returns die arriveren in een periode direct herproduceerd kunnen worden om de vraag in de desbetreffende periode te voldoen. De setupkosten voor productie en herproductie staan vast. Tot slot zijn er voorraadkosten voor de returns en servicables die in voorraad worden gehouden. In figuur 1 staat dit schematisch weergegeven.




**Figuur 1 - Inventory systeem met herproductie**

Om het ELSR probleem te modelleren worden de volgende parameters en beslissingsvariabelen gebruikt.

Parameters:

- $T$ horizon van het model
- $t$ index voor horizon, $t = 1,\ldots, T$
- $d_t$ vraag in periode $t$
- $r_t$ returns in het begin van periode $t$
- $K$ gezamenlijke setupkosten
- $h^r$ voorraadkosten van de returns per periode
- $h^s$ voorraadkosten van de servicables per periode

Beslissingsvariabelen:

- $x_t^r$ aantal herproduceerde producten in periode $t$
- $x_t^m$ aantal geproduceerde producten in periode $t$
- $I^r$ aantal returns in voorraad aan het einde van periode $t$
- $I^s$ aantal servicables in voorraad aan het einde van periode $t$
- $\delta_t$ indicator variabele voor productie in periode $t$
- $M$ een groot geheeltallig getal

Met behulp van de bovenstaande notatie wordt de ELSR gemodelleerd als een gemixte geheeltallig lineair programmering probleem.

$$\min\left(\sum_{t=1}^{T}\left(K\delta_t + h^r I_t^r + h^s I_t^s\right)\right)$$

s.t.

$$I_t^r = I_{t-1}^r + r_t - x_t^r \qquad \text{voor } t = 1,\ldots,T, \tag{1}$$

$$I_t^s = I_{t-1}^s + x_t^r + x_t^m - d_t \qquad \text{voor } t = 1,\ldots,T, \tag{2}$$

$$M_t\delta_t \geq x_t^r + x_t^m \qquad \text{voor } t = 1,\ldots,T, \tag{3}$$

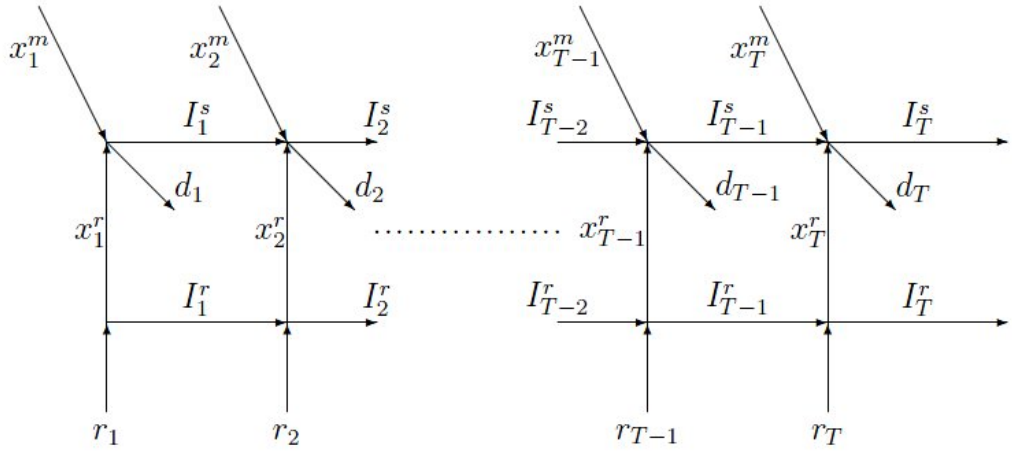
$$\delta_t \in \{0,1\} \tag{4}$$

$$x_t^r, x_t^m, I_t^r, I_t^s \geq 0 \tag{5}$$

$$M_t = \sum_{i=t}^{T} d_t \qquad \text{voor } t = 1,\ldots,T. \tag{6}$$

In de doelstellingsfunctie hebben we de setupkosten, voorraadkosten van de returns en de voorraadkosten van de servicables die geminimaliseerd worden. De eerste twee restricties zorgen dat voorraad van returns en servicables in balans blijven. Restrictie 1 corrigeert het aantal returns $I_t^r$, door aan de returns uit de vorige periode $I_{t-1}^r$ de returns in de desbetreffende periode $r_t$ toe te voegen minus het aantal producten dat herproduceerd is $x_t^r$. En restrictie 2 corrigeert het huidige aantal servicables $I_t^s$ door aan het aantal servicables uit de vorige periode $I_{t-1}^s$ de totale productie $x_t^r + x_t^m$ toe te voegen minus de vraag in de desbetreffende periode $d_t$. Restrictie 3 houdt de productieperioden (setups) bij, zodat de totale productie (herproductie en productie) nooit de vraag in periode *t* te boven gaat. Verder is de indicator variabele een binaire variabele die aangeeft of er geproduceerd wordt of niet, de overige beslissingsvariabelen moeten groter of gelijk zijn aan 0.

De bovenstaande wiskundige beschrijving kan ook als een network flow probleem gepresenteerd worden zoals in figuur 2. De stroom van de bovenste lijnen geven de productieperioden weer, de verticale lijnen in het midden de herproductieperioden. Vervolgens staan de bovenste horizontale lijnen voor de voorraad van servicables en de onderste horizontale lijnen geven de voorraad van de returns weer. Tot slot geven de onderste verticale lijnen het aantal returns in periode *t* weer.




**Figuur 2 - Network flow representatie van het ELSR probleem**

In tabel 1 staat een voorbeeld van het ELSR probleem uitgewerkt. Hierbij zijn de setupkosten gelijk aan 20, de voorraadkosten van de returns bedragen 0,2 per periode en de voorraadkosten van de servicables bedragen 1 per periode. Zoals te zien is in de

onderstaande tabel wordt in periode 1 en 4 geproduceerd. De totale kosten voor de onderstaande probleeminstantie bedraagt 63,8.

| $T$ | $D$ | $R$ | $x_t^r$ | $x_t^m$ |
|---|---|---|---|---|
| 1 | 5 | 4 | 4 | 13 |
| 2 | 8 | 1 | 0 | 0 |
| 3 | 4 | 2 | 0 | 0 |
| 4 | 7 | 3 | 6 | 7 |
| 5 | 6 | 5 | 0 | 0 |

**Tabel 1 - Voorbeeld ELSR**

Hoofdstuk 3

# Methodologie

In dit hoofdstuk worden de zeven methoden beschreven die in het vervolg toegepast gaan worden op het Lot Sizing probleem met returns. De methoden zijn te splitsen in twee categorieën, namelijk methoden die van een oneindige, discrete planningshorizon uitgaan zoals de Silver Meal, Least Unit Costs en Marginal Costs Difference heuristiek. Het voordeel van deze methoden is dat het veranderen van het aantal perioden geen invloed heeft op de uitkomst van de heuristieken. De tweede categorie bestaat uit metaheuristieken zoals het genetisch algoritme en het greedy algoritme. Van beide metaheuristieken worden twee varianten beschreven.

## 3.1 Silver Meal (Least Period Costs)

De Silver Meal- of Least Period Costs heuristiek is een bekende heuristiek van Silver en Meal (1973) die bij het originele Lot Sizing probleem gebruikt wordt (Silver *et al.* 2006) en goede resultaten levert (Teunter *et al.* 2006). De Silver-Meal heuristiek kiest het aantal perioden dat de kosten per periode minimaliseert. Er wordt in periode 1 gestart en telkens wanneer er aan het stopcriterium (zie formule (5)) wordt voldaan, wordt er een beslissing genomen.
Een nadeel van deze heuristiek is dat het echte minimum gemist kan worden, want als eenmaal aan het stopcriterium wordt voldaan staat de beslissing vast en wordt er niet meer naar de eerstvolgende periode gekeken. Het idee voor Lot Sizing met returns en gezamenlijke setupkosten is hetzelfde, enkel de kostenberekening is anders. Met behulp van de kostenberekening kunnen we ook precies het stopcriterium definiëren.

Voor de kostenberekening zijn de volgende variabelen van belang,

- $l$ startperiode
- $k$ eindperiode
- $m$ overgebleven return uit vorige Lot Sizing beslissing

$C_{l,k}(m)$ zijn de totale kosten in interval $[l, k]$ als de overgebleven returns, bepaald door de vorige Lot Sizing beslissing, aan het eind van periode $l-1$ gelijk zijn aan $m$. Hierbij moet de order die geplaatst wordt in periode $l$ voldoende zijn om de vraag te voldoen tot en

met periode *k*. Er blijven enkel returns over in periode *l* als de voorraad *m* aan het einde van periode *l* - 1 plus de returns $R_l$ in periode *l* groter zijn dan de ordergrootte $\sum_{i=l}^{k} D_i$. Dit wordt als volgt geformuleerd $(m + R_l - \sum_{i=l}^{k} D_i)^+$. Deze kosten worden enkel meegenomen als deze positief zijn, met $(x)^+ = \max\{0, x\}$. Als er returns overblijven aan het eind van periode *l* dan blijven deze in voorraad tot periode *k*. Uiteindelijk worden de voorraadkosten voor de returns op de volgende manier berekend

$$h^r(k-l+1)(m+R_l-\sum_{i=l}^{k} D_i)^+ \tag{1}$$

Er moet nog rekening gehouden worden met de returns die na periode *l* in voorraad blijven. Deze worden als volgt berekend

$$h^r \sum_{i=l+1}^{k} (k+1-i)R_i \tag{2}$$

Verder zijn er voorraadkosten voor de servicables, dit zijn producten die al geproduceerd zijn en in de volgende perioden verkocht gaan worden. De voorraadkosten hiervoor zijn

$$h^s \sum_{i=l+1}^{k} (i-1)D_i \tag{3}$$

Tot slot moet er nog rekening gehouden worden met de setupkosten. De totale kostenformule is dus

$$C_{l,k}(m) = K + h^r\left((k-l+1)\left(m + R_l - \sum_{i=1}^{k} D_i\right)^+ + \sum_{i=l+1}^{k}(k+1-i)R_i\right) + h^s \sum_{i=l+1}^{k}(i-1)D_i \tag{4}$$

Met behulp van de bovenstaande totale kostenformule wordt het stopcriterium voor de Silver-Meal heuristiek als volgt gedefinieerd

$$\frac{C_{l,k+1}(m)}{k+1-l} > \frac{C_{l,k}(m)}{k-l} \tag{5}$$

Het stopcriterium vergelijkt het verschil in kosten per periode, waarbij *k* - *l* gelijk staat aan het aantal perioden en *k* + 1 – *l* wanneer er een nieuwe setup plaatsvindt in periode *k* + 1.

## 3.2 Least Unit Costs

De Least Unit Costs heuristiek is een soortgelijk heuristiek als de Silver Meal. Deze is ook toegepast op het originele Lot Sizing probleem. Deze heuristiek doet het iets slechter dan de Silver Meal heuristiek (Teunter *et al.* 2006). Het verschil met de Silver Meal heuristiek is dat de Least Unit Cost heuristiek als stopcriterium kosten per product in de desbetreffende periode in acht neemt in plaats van kosten per periode. Het aantal producten in een periode wordt met de onderstaande formule berekend

$$\sum_{i=l}^{k} D_i \tag{6}$$

Voor de kostenberekening wordt gebruik gemaakt van formule (4), omdat de originele totale kostenformule van de Least Unit Costs heuristiek niet toepasbaar is op het probleem waarbij er sprake is van returns. Het stopcriterium voor de Least Unit Costs heuristiek wordt dan

$$\frac{C_{l,k+1}(m)}{\sum_{i=l}^{k+1} D_i} > \frac{C_{l,k}(m)}{\sum_{i=l}^{k} D_i} \tag{7}$$

## 3.3 Marginal Cost Difference

De Marginal Cost Difference heuristiek van Groff (1979) is ook een van de heuristieken die op het Lot Sizing probleem zonder return toegepast is. Het stopcriterium van deze heuristiek vergelijkt de marginale kosten verschillen van de voorraadkosten $\Delta H_{l,k}$ en de setupkosten $\Delta K_{l,k}$. Om deze heuristiek toe te passen op het Lot Sizing probleem met returns wordt voor de kostenberekening gebruik gemaakt van formule (4) en de marginale voorraadkosten formule $\Delta H_{l,k} = h\left(\sum_{i=l}^{k} D_i\right)/2$ wordt aangepast. De marginale setupkosten formule $\Delta K_{l,k} = K/(k-l) - K/(k+1-l)$ van Groff kan wel gebruikt worden voor het Lot Sizing probleem met returns.

De marginale setupkosten $\Delta K_{l,k}$ en de marginale voorraadkosten $\Delta H_{l,k}$ worden als volgt gedefinieerd

$$\Delta K_{l,k} = \frac{K}{k-l} - \frac{K}{k+1-l} \tag{7}$$

$$\Delta H_{l,k} = C_{l,k+1}(m) - C_{l,k}(m) \tag{8}$$

Voor de marginale setupkosten hebben we het verschil tussen de totale kosten genomen van periode *l* tot *k*+1 en *l* tot *k*. Hierbij is gebruik gemaakt van formule (4). Het verschil in deze totale kosten is exact gelijk aan de marginale voorraadkosten. Het stopcriterium van de Marginal Cost Difference heuristiek van Groff is

$$\Delta H_{l,k} - \Delta K_{l,k} < 0 \leq \Delta H_{l,k+1} - \Delta K_{l,k+1} \tag{9}$$

Voor de volledigheid van het stopcriterium wordt $\Delta K_{1,1}$ gelijkgesteld aan oneindig en $\Delta H_{1,1}$ gelijkgesteld aan nul.

## 3.4 Genetisch Algoritme

Het eerste genetisch algoritme (Van den Heuvel 2006) dat toegepast gaat worden op het probleem bestaat alleen uit het combineren van oplossingen. De parameters voor dit genetisch algoritme zijn het aantal startoplossingen *n* en het aantal iteraties *m*.

Het genetisch algoritme gaat als volgt te werk, het aantal startoplossingen (deze worden random gegenereerd) worden alleen gecombineerd. De startoplossingen geven weer in welke perioden er geproduceerd wordt. Hierbij geeft een 1 aan dat er in de desbetreffende periode geproduceerd wordt, en een 0 geeft aan dat er niet geproduceerd wordt. In de eerste periode zal er altijd geproduceerd worden, zodat het een toegelaten oplossingen is, voor de overige perioden worden uniform verdeelde pseudorandom getallen getrokken, door deze vervolgens af te ronden is het bekend of er geproduceerd wordt of niet. Door de zero-inventory eigenschap zijn we met deze oplossingen in staat om alle productiehoeveelheden te berekenen. Dit komt doordat er enkel geproduceerd wordt als er geen voorraad meer is. Wanneer de productieperioden bekend zijn, staan dus alle productiehoeveelheden vast.

Nu er *n* toegelaten startoplossingen zijn, kan de combinatieprocedure van start gaan. Voor de combinatieprocedure is er één regel die de oplossingen gaat combineren. De combinatieregel wordt als volgt geformuleerd; Stel er zijn twee kandidaat-oplossingen $(y_t^1, y_t^2)$. Als voor beide oplossingen op dezelfde plek de waarden gelijk zijn aan elkaar, dan verandert de waarde niet. Als de waarden niet gelijk aan elkaar zijn, wordt het met kans ½, de waarde 1 en met kans ½, wordt de waarde 0.
In figuur 3 staat de combinatieregel voor het genetisch algoritme.

$$y_t^1 = 1, y_t^2 = 1 \Rightarrow y_t^3 = 1$$
$$y_t^1 = 0, y_t^2 = 1 \Rightarrow y_t^3 = \begin{cases} 0 \text{ met kans } ½ \\ 1 \text{ met kans } ½ \end{cases}$$
$$y_t^1 = 1, y_t^2 = 0 \Rightarrow y_t^3 = \begin{cases} 0 \text{ met kans } ½ \\ 1 \text{ met kans } ½ \end{cases}$$
$$y_t^1 = 0, y_t^2 = 0 \Rightarrow y_t^3 = 0$$

**Figuur 3 - Combinatieregel voor GA**

In elke iteratie van het genetisch algoritme worden alle $n$ startoplossingen met elkaar gecombineerd. Na elke iteratie worden er dus $½n(n-1)$ oplossingen toegevoegd aan de $n$ startoplossingen. Van deze $n + ½n(n-1)$ oplossingen worden vervolgens de beste $n$ oplossingen gekozen die gebruikt gaan worden in de eerstvolgende iteratie. In elk iteratie worden de kosten voor de $½n(n-1)$ oplossingen berekend. Bij het selecteren van de beste $n$ oplossingen wordt er gekeken naar de totale kosten van elke oplossing.
Het genetisch algoritme heeft 2 stopcriteria, wanneer er $m$ iteraties zijn bereikt of als de kosten van de beste $n$ oplossingen aan elkaar gelijk zijn.

Genetisch Algoritme

1. Genereer $n$ startoplossingen en bereken de totale kosten van deze $n$ oplossingen
2. Combineer alle $n$ oplossingen met elkaar door de combinatieregel toe te passen. Het totale aantal oplossingen worden dan $n + ½n(n-1)$
3. Bereken de totale kosten van alle gecombineerde oplossingen $½n(n-1)$
4. Selecteer de beste $n$ oplossingen op basis van de totale kosten
5. Als alle $n$ oplossingen niet gelijk zijn aan elkaar of $m$ aantal iteraties niet zijn bereikt.
   Ga terug naar stap 2 met $n$ oplossingen
6. Kies beste oplossing uit de $n$ oplossingen
   STOP

## 3.5 Gewogen Genetisch Algoritme

Het tweede genetisch algoritme wordt uitgebreid met het mutatieproces en het combinatieproces krijgt een ander vorm. Er zal gecombineerd worden met gewichten die aan de oplossingen toegekend gaan worden. Hierdoor neemt het aantal parameters toe. Naast het aantal startoplossingen $n$ en het aantal iteraties $m$ moet het percentage oplossingen dat gecombineerd wordt *PercComb* en het percentage oplossingen dat gemuteerd gaat worden *PercMut* gegeven worden.

Eerst wordt er net als bij het genetische algoritme $n$ startoplossingen gegenereerd, door voor elk van de $n$ oplossingen $(T-1)$ uniform verdeelde pseudorandom getallen te generen en deze vervolgens af te ronden. De waarden voor de eerste perioden zijn allen 1 zodat er toegelaten oplossingen gegenereerd worden. Om aan deze oplossingen gewichten toe te kennen worden eerst de totale kosten van elk oplossing berekend. Voor het toekennen van de gewichten aan de oplossingen wordt de volgende formule gebruikt

$$w_i = \frac{\exp(-c_i/1000)}{\sum_i \exp(-c_j/1000)} \quad [1] \qquad (2)$$

Hierbij is $w_i$ het gewicht van oplossing $i$. Nu er gewichten zijn toegekend aan de $n$ oplossingen worden er $(1 - PercComb) \times n$ oplossingen met kans $w_i$ geselecteerd uit de $n$ startoplossingen. Deze $n$ oplossingen worden als nieuwe set oplossingen gedefinieerd. Vervolgens worden er $(PercComb \times n)$ oplossingen aan de nieuwe set oplossingen toegevoegd door deze willekeurig gekozen oplossingen met kans $w_i$ met elkaar te combineren. Bij het combineren van de oplossingen wordt weer gebruikt gemaakt van de combinatieregel zie figuur 3.

Voor het muteren worden er $(PercMut \times n)$ oplossingen uit de nieuwe set oplossingen toegevoegd door deze willekeurig te muteren. De mutatieregel is als volgt; we kiezen willekeurig een productieperiode, deze veranderen we in een periode waarin niet geproduceerd wordt. Vervolgens kiezen we een willekeurig periode waarin niet geproduceerd wordt en deze veranderen we vervolgens in een productieperiode. Hierbij wordt geen gebruik gemaakt van de gewichten $w_i$.

Nu de oplossingen zijn gecombineerd en gemuteerd, worden de kosten van alle oplossingen berekend. De beste oplossing wordt bewaard en aan de huidige set oplossingen toegevoegd.

[1] De totale kosten van elk oplossing is een getal in de duizenden, vandaar dat er gedeeld wordt door 1000 om vervolgens het exponentieel van de kosten te nemen.

Het stopcriterium voor het genetisch algoritme is het aantal iteraties *m*. Na elke iteratie wordt de beste oplossing bewaard. De beste oplossing van alle iteraties is de uiteindelijke oplossing van het probleem.

Gewogen Genetisch Algoritme

1 Genereer *n* startoplossingen

2 Bereken kosten van de *n* staroplossingen

3 Bepaal beste oplossing

4 Voeg (1 - *PercComb*) x *n* oplossingen toe uit huidige set oplossingen aan nieuwe set oplossingen met kans $w_i$

5 Voeg *PercComb* x *n* oplossingen toe uit huidige set oplossingen aan nieuwe set door deze willekeurig met elkaar te combineren met kans $w_i$

6 Kies *PercMut* x *n* oplossingen uit nieuwe set oplossingen en muteer deze

7 Bereken kosten van de nieuwe set oplossingen en bewaar beste oplossing

8 Als maximum aantal iteraties is bereikt, selecteer beste oplossing van nieuwe set oplossingen
STOP

9 Ga naar Stap 4 met nieuwe set oplossingen

## 3.6 Add Production Period

De Add Production Period heuristiek maakt gebruik van de greedy benadering. Het doel van deze heuristiek is om op een systematisch wijze productieperioden toe te voegen waarbij de periode die gekozen wordt om te produceren de totale kosten zal verminderen. Bij toevoegen van een productieperiode zal de samenstelling van de geproduceerde en de herproduceerde producten veranderen en daardoor ook de totale kosten. Dit komt doordat het mogelijk is om tussen de productieperioden in (indien deze niet direct achterelkaar plaatsvinden) nog een productieperiode toe te voegen. Hierdoor veranderen de geproduceerde en herproduceerde producten voor zowel de eerste als

laatste periode. Het probleem wordt dus niet opgesplitst in subproblemen, waardoor er dus ook geen subproblemen geoptimaliseerd worden.

De Add Production Period heuristiek start met één productieperiode in periode 1, alle overige perioden zijn niet mogelijk zoals eerder vermeld. Vervolgens worden de totale kosten berekend. Bij het toevoegen van productieperiode worden voor elke mogelijke periode de totale kosten berekend en er wordt gekozen voor de periode die de totale kosten minimaliseert. Dit wordt herhaald totdat het toevoegen van een productieperiode geen kostenbesparing meer oplevert.

Add Production Period (Greedy Algoritme)

1 Bereken de totale kosten met één productieperiode in periode 1
Bewaar de totale kosten

2 Voeg indien mogelijk één productieperiode toe, anders STOP

3 Bereken de totale kosten voor alle mogelijke perioden waar de productieperiode toegevoegd kan worden.
Selecteer de periode met de minimale totale kosten

4 Als het toevoegen van periode hogere totale kosten meebrengt
STOP, beste oplossing gevonden.

5 Bewaar de totale kosten.
Ga naar stap 2 met huidige productieperioden.

## 3.7 Remove Production Period

Het principe van de Remove Production Period heuristiek is hetzelfde als van de Add Production Period heuristiek. Zoals de naam al luidt verwijderen we in deze heuristiek productieperioden in plaats van toevoegen. Een bijkomend voordeel van deze heuristiek in tegenstelling tot het Add Production Period is dat na het verwijderen van een productieperiode meer returns beschikbaar zijn. Maar ook hier kunnen we niet spreken van optimalisatie van subproblemen, omdat na het verwijderen van productieperioden de samenstelling van de returns weer zal veranderen.

De heuristiek start met het maximaal aantal productieperioden en vervolgens verwijderen we systematisch productieperioden totdat de totale kosten niet meer dalen. Een

productieperiode wordt pas verwijderd als deze voor de hoogste kostenbesparing zorgt tussen alle productieperioden die verwijderd kunnen worden. In elk stap wordt één productieperiode verwijderd. De heuristiek stopt als we geen productieperiode meer kunnen verwijderen, dat wil zeggen als het verwijderen van een productieperiode geen kostenbesparing meer levert.

Remove Production Period (Greedy Algoritme)

1 Bereken de totale kosten wanneer in alle perioden geproduceerd wordt.
Bewaar de totale kosten.

2 Verwijder indien mogelijk één productieperiode, anders STOP

3 Bereken de totale kosten voor alle mogelijke perioden waar één productieperiode verwijderd kan worden.
Selecteer de periode met de minimale totale kosten.

4 Als het verwijderen van periode hogere totale kosten meebrengt
STOP beste oplossing gevonden.

5 Bewaar de totale kosten.
Ga naar stap 2 met huidige productieperioden.

Hoofdstuk 4

# Resultaten

De methoden die in het vorige hoofdstuk beschreven zijn, worden in dit hoofdstuk getest. De methoden zullen getest worden met behulp van een gevarieerde dataset. Deze bestaat uit verschillende vraag- en returnpatronen. Deze worden als eerst beschreven. De prestaties van de methoden worden beoordeeld aan de hand van een gevoeligheidsanalyse. De optimale oplossingen zijn met de dataset geleverd, waardoor het mogelijk is om de oplossingen van de methoden te vergelijken met de optimale oplossingen. Het verschil in procenten met de optimale oplossingen (dit wordt 'error' genoemd) worden bepaald en hiervan worden enkele karakteristieken zoals het gemiddelde, standaarddeviatie en maximum berekend.

Na de beschrijving van de dataset worden de resultaten van de Silver Meal en Least Unit Costs heuristiek in het kort besproken (voor meer details zie Teunter *et al.* (2006)). Vervolgens bespreken we de resultaten van de Marginal Costs Difference heuristiek. Van het genetisch en gewogen genetisch algoritme worden de resultaten met drie verschillende parameters besproken. De laatste sectie zal de resultaten van de Add en Remove Production Period bevatten.

## 4.1 Dataset

De dataset is geleverd door Dr. W. van den Heuvel van de Erasmus Universiteit Rotterdam. De dataset bevat verschillende vormen van zowel de vraag als de returns. In de dataset komen 10 verschillende patronen van de vraag voor. Voor de returns vinden we 22 verschillende patronen. Voor elk patroon zijn er 4 realisaties gegenereerd, waardoor het totaal aantal patronen voor de vraag 40 wordt en aantal patronen voor de returns 88.
De kosten parameters zijn als volgt; de voorraadkosten voor de servicables zijn gelijk aan 1. Voor de returns variëren de voorraadkosten tussen 0.2, 0.5 en 0.8. Nog een kostenpost die we behandelen zijn de setupkosten. Deze kosten bedragen 200, 500 of 2000. In tabel 2 staat een gedetailleerde beschrijving van de vraag- en returnpatronen en de kostenparameters.
In totaal zijn er 40 (vraagpatronen) maal 88 (return patronen) maal 3 (setupkosten) maal 3 (voorraadkosten), dus 31680 probleeminstanties om de verschillende methoden te

testen. In de volgende subsecties zullen we de resultaten van de verschillende methoden bespreken.

| Vraag patroon | | | | | | Return patroon | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $\mu$ | $\sigma$ | $\tau$ | $a$ | $c$ | $d$ | $\mu$ | $\sigma$ | $\tau$ | $a$ | $c$ | $d$ |
| Stationair | | | | | | Stationair | | | | | |
| 100 | 10 | 0 | 0 | na | na | 30 | 3 | 0 | 0 | na | na |
| 100 | 20 | 0 | 0 | na | na | 30 | 6 | 0 | 0 | na | na |
| Positieve trend | | | | | | 50 | 5 | 0 | 0 | na | na |
| 100 | 10 | 10 | 0 | na | na | 50 | 10 | 0 | 0 | na | na |
| 100 | 10 | 20 | 0 | na | na | 70 | 7 | 0 | 0 | na | na |
| Negatieve trend | | | | | | 70 | 14 | 0 | 0 | na | na |
| 210 | 10 | -10 | 0 | na | na | Positieve trend | | | | | |
| 320 | 10 | -20 | 0 | na | na | 30 | 3 | 3 | 0 | na | na |
| Seizoensgebonden (piek in het midden) | | | | | | 30 | 3 | 6 | 0 | na | na |
| 100 | 10 | 0 | 20 | 12 | 1 | 70 | 7 | 7 | 0 | na | na |
| 100 | 10 | 0 | 40 | 12 | 1 | 70 | 7 | 14 | 0 | na | na |
| Seizoensgebonden (dal in het midden) | | | | | | Negatieve Trend | | | | | |
| 100 | 10 | 0 | 20 | 12 | 3 | 63 | 3 | -3 | 0 | na | na |
| 100 | 10 | 0 | 40 | 12 | 3 | 96 | 3 | -6 | 0 | na | na |
| | | | | | | 147 | 7 | -7 | 0 | na | na |
| | | | | | | 224 | 7 | -14 | 0 | na | na |
| Kosten Parameters | | | | | | Seizoensgebonden (piek in het midden) | | | | | |
| Parameters | | | Waarden | | | 30 | 3 | 0 | 6 | 12 | 1 |
| K | | | 200 | 500 | 2000 | 30 | 3 | 0 | 12 | 12 | 1 |
| $h^r$ | | | 0.2 | 0.5 | 0.8 | 70 | 7 | 0 | 14 | 12 | 1 |
| $h^s$ | | | | 1 | | 70 | 7 | 0 | 28 | 12 | 1 |
| | | | | | | Seizoensgebonden (dal in het midden) | | | | | |
| | | | | | | 30 | 3 | 0 | 6 | 12 | 3 |
| | | | | | | 30 | 3 | 0 | 12 | 12 | 3 |
| | | | | | | 70 | 7 | 0 | 14 | 12 | 3 |
| | | | | | | 70 | 7 | 0 | 28 | 12 | 3 |

**Tabel 2 - De vraag- en returnpatronen zijn met de volgende formule gegenereerd $D_t=\mu+\tau(t-1)+asin[(2\pi t/c)+d(\pi/2)]+\varepsilon_t$ met μ de startwaarde van het patroon, τ de trend level, a de amplitudo van de cyclus, c de cycluslengte en $\varepsilon_t$ (t = 1….T) is een onafhankelijk normaal verdeelde random variabele met standaarddeviatie σ.**

## 4.2 Silver Meal en Least Unit Costs

De resultaten van de Silver Meal en Least Unit Costs heuristiek zijn gegeven. Zoals eerder vermeld willen we deze resultaten verbeteren. Een beknopt schema met de resultaten van alle patronen staat in tabel 3.

| | Percentage kosten error | |
|---|---|---|
| | Silver Meal | Least Unit Costs |
| Gemiddelde (%) | 3.1 | 4.2 |
| Standaarddeviatie (%) | 4.9 | 6.5 |
| Maximum (%) | 25.1 | 37.0 |
| Percentage < 1% | 59.40% | 49.27% |
| Percentage < 5% | 77.99% | 73.13% |
| Percentage < 10% | 89.74% | 84.24% |
| Gemiddelde uitvoertijd (s) | 0.02 | 0.02 |

**Tabel 3 - Resultaten van Silver Meal en Least Unit Cost heuristiek.**

We zien dat de Silver Meal heuristiek het op alle fronten beter doet dan de Least Unit Costs heuristiek. Het gemiddelde is 1% lager, standaarddeviatie 1.5% en het maximum is 12% lager. De gemiddelde uitvoertijd van de heuristieken zijn wel gelijk aan elkaar, deze is 0.02 seconde. Om meer inzicht te verkrijgen in de prestaties van de heuristieken staan in tabel 4 de karakteristieken voor de diverse setupkosten. We zien voor beide heuristieken dat naarmate de setupkosten stijgen, de resultaten slechter worden. Dit kan als volgt verklaard worden, naarmate de setupkosten hoger worden telt een extra productieperiode veel zwaarder mee, omdat de setupkosten hoger zijn. Ook het feit dat we een eindig aantal perioden hebben heeft invloed op de resultaten. Doordat de dataset 12 perioden bevat vallen de kosten veel hoger uit als er in de laatste periode aan het stopcriterium wordt voldaan. In praktijk zal dit niet zo zijn, tenzij er expliciet gekozen wordt om voor een vast aantal perioden te produceren.

| | Silver Meal | | | Least Unit Costs | | |
|---|---|---|---|---|---|---|
| | K=200 | K=500 | K=2000 | K=200 | K=500 | K=2000 |
| Gemiddelde (%) | 1,1 | 1,9 | 6,1 | 1,5 | 2,4 | 8,8 |
| Standaarddeviatie (%) | 2,7 | 2,7 | 6,6 | 3,1 | 3,4 | 8,6 |
| Maximum (%) | 23,6 | 23,3 | 25,1 | 34,0 | 27,5 | 37,0 |

**Tabel 4 - Gevoeligheidsanalyse Silver Meal & Least Unit Costs Heuristiek gesorteerd op setupkosten (K = 200, 500 en 2000)**

In Appendix A tabel A.1 staat een uitgebreide tabel, waarin resultaten te vinden zijn voor de verschillende vraag- returnpatronen en verschillende kosten parameters.

## 4.2 Marginal Cost Difference

De resultaten van de Marginal Cost Difference heuristiek die uitgevoerd is op het Lot Sizing probleem zonder returns waren zeer goed en vergelijkbaar met de Silver Meal en Least Unit Costs heuristiek (Baker 1989). Dit kunnen we helaas niet zeggen als we de heuristiek aanpassen en uitvoeren voor het Lot Sizing probleem met returns. In tabel 5

staan de resultaten voor alle patronen. Het gemiddelde, standaarddeviatie en maximum zijn meer dan 3 keer hoger dan de Silver Meal heuristiek. We zien dat slechts 4,02% van de probleeminstanties een fout hebben lager dan 1%. Met een maximum fout van 65,3% kunnen we zeggen dat de resultaten echter tegenvallen. Het feit dat de prestaties van de heuristiek tegenvallen komt doordat er teveel productieperioden plaatsvinden. Het stopcriterium werkt dus niet goed bij het probleem met returns. Het feit dat de formule voor de marginale kostenverschillen van de voorraadkosten aangepast is kan invloed hebben op de tegenvallende resultaten van de heuristiek. In Appendix A tabel A.1 staan tevens de resultaten van de heuristiek voor de verschillende patronen, waaruit weer duidelijk te zien is dat de Marginal Cost Difference heuristiek het altijd slechter doet dan de Silver Meal heuristiek.

| | Percentage kosten error |
|---|---|
| | Marginal Cost Difference |
| Gemiddelde (%) | 10,7 |
| Standaarddeviatie (%) | 9,6 |
| Maximum (%) | 65,3 |
| Percentage < 1% | 4,02% |
| Percentage < 5% | 30,21% |
| Percentage < 10% | 64,21% |
| Gemiddelde uitvoertijd (s) | 0.02 |

**Tabel 5 - Resultaten Marginal Cost Difference heuristiek**

## 4.3 Genetisch Algoritme

Het genetisch algoritme is de eerste metaheuristiek en de eerste variant die we gaan toepassen. Deze is driemaal uitgevoerd op het probleem met verschillende paramaters. Het aantal iteraties is constant gehouden, deze is gelijk genomen aan de lengte van de horizon 12. Het aantal startoplossingen is gevarieerd, hierbij is gekozen voor 2, 4 en 6 maal de lengte van de horizon wat neerkomt op 24, 48 en 72 startoplossingen. Deze methode levert zeer goede resultaten voor het Lot Sizing probleem met returns en aparte setupkosten (Van den Heuvel 2006). De resultaten voor het probleem met returns en gezamenlijke setupkosten zijn echter voortreffelijk. De resultaten zijn te vinden in tabel 6. Hierbij staat GA2 voor het genetisch algoritme met 2 maal, GA4 voor 4 maal en tot slot GA6 voor 6 maal de lengte van de horizon. We zien duidelijk dat het gemiddelde nu veel lager ligt, zelfs onder de 1% en deze daalt naarmate het aantal startoplossingen toeneemt. Hetzelfde geldt voor de standaarddeviatie. We zien dat het maximum van GA2 nog best hoog ligt, maar het percentage oplossingen met een fout kleiner dan 1% ligt tegen de 90% aan en bijna alle oplossingen hebben een fout kleiner dan 5%. Het maximum, wat erg hoog ligt, kunnen we wijten aan het feit dat de startoplossingen random gegenereerd worden. De resultaten van GA4 en GA6 zijn zoals verwacht veel

beter. Het gemiddelde, maximum en standaarddeviatie worden kleiner en meer dan 99% procent van de oplossingen hebben een fout kleiner dan 1%.

| | Percentage kosten error | | |
|---|---|---|---|
| | GA2 | GA4 | GA6 |
| Gemiddelde (%) | 0,282 | 0,024 | 0,004 |
| Standaarddeviatie (%) | 0,93 | 0,29 | 0,13 |
| Maximum (%) | 50,0 | 22,9 | 9,4 |
| Percentage < 1% | 89,26% | 99,15% | 99,90% |
| Percentage < 5% | 99,76% | 99,97% | 99,98% |
| Percentage < 10% | 99,97% | 99,99% | 100,00% |
| Gemiddelde uitvoertijd (s) | 0,26 | 0,58 | 1,24 |

**Tabel 6 – Resultaten genetisch algoritme**

De uitvoertijden van het genetisch algoritme liggen wat hoger vergeleken met de vorige heuristieken. Dit komt doordat dit algoritme geavanceerder is. Naarmate het aantal startoplossingen hoger wordt neemt de uitvoertijd toe, doordat er meer oplossingen met elkaar gecombineerd worden. Voor het genetisch algoritme kunnen we geen onderscheid maken als we de resultaten sorteren op setupkosten. De fouten stijgen niet naarmate de setupkosten hoger worden zoals bij de Silver Meal heuristiek (zie Appendix B tabel B.1).

## 4.4 Gewogen Genetisch Algoritme

De tweede variant van het genetisch algoritme is het gewogen genetisch algoritme. Voor het gewogen genetisch algoritme zijn de parameters als volgt gekozen; het aantal startoplossingen varieert tussen 24, 48 en 72 zoals bij het genetisch algoritme. Het aantal iteraties is 50. Hiervoor is gekozen, omdat het gewogen genetisch algoritme niet zoals het genetisch algoritme alle oplossingen met elkaar combineert, maar de beste *n* oplossingen met elkaar combineert, rekening houdend met het gewicht. Het percentage dat gecombineerd wordt is 80% en het percentage dat gemuteerd wordt is 30%. In tabel 7 zijn de resultaten van het gewogen genetisch algoritme te vinden. Naarmate de startoplossingen toenemen worden de resultaten veel beter. Tevens valt op dat het maximum veel lager ligt vergeleken met het genetisch algoritme, maar het gemiddelde en standaarddeviatie zijn wel wat hoger dan het genetisch algoritme. Omdat het maximum van het gewogen genetisch algoritme veel lager ligt hebben alle oplossingen een fout kleiner dan 10% en 99% van de oplossingen hebben een fout kleiner dan 5%. De percentages oplossingen met een fout kleiner dan 1% zijn vergelijkbaar met het genetisch algoritme. De gemiddelde uitvoertijd van het gewogen genetisch algoritme 2 en 4 liggen wat hoger dan het genetisch algoritme en de uitvoertijd van het gewogen genetisch algoritme 6 ligt 0.2 seconde lager dan het genetisch algoritme. De uitvoertijden zijn

uiteindelijk wel vergelijkbaar met elkaar, het verschil is minimaal. In Appendix B tabel B.1 zijn de resultaten voor elk patroon afzonderlijk te vinden.

| | Percentage kosten error | | |
|---|---|---|---|
| | GGA2 | GGA4 | GGA6 |
| Gemiddelde (%) | 0,32 | 0,17 | 0,11 |
| Standaarddeviatie (%) | 0,83 | 0,60 | 0,90 |
| Maximum (%) | 9,44 | 6,42 | 7,36 |
| Percentage < 1% | 87,63 | 93,14 | 95,39 |
| Percentage < 5% | 99,76 | 99,93 | 99,95 |
| Percentage < 10% | 100,00 | 100,00 | 100,00 |
| Gemiddelde uitvoertijd (s) | 0,48 | 0,65 | 1,03 |

**Tabel 7 – Resultaten gewogen genetisch algoritme**

## 4.5 Add en Remove Production Period

De tweede soort metaheuristiek maakt gebruikt van de greedy benadering. Ook hierbij hebben we twee varianten toegepast. De eerste variant zal systematisch productieperioden toevoegen en de tweede variant zal systematisch productieperioden verwijderen. Zoals eerder beschreven wordt hierbij gekeken naar de totale kosten. In tabel 8 zijn de resultaten van de gevoeligheidsanalyse te vinden voor beide varianten.

| | Percentage kosten error | |
|---|---|---|
| | Add Period | Remove Period |
| Gemiddelde (%) | 2,4 | 2,9 |
| Standaarddeviatie (%) | 2,9 | 3,3 |
| Maximum (%) | 33,4 | 25,7 |
| Percentage < 1% | 46,71% | 37,92% |
| Percentage < 5% | 85,86% | 79,38% |
| Percentage < 10% | 97,92% | 95,94% |
| Gemiddelde uitvoertijd (s) | 0,02 | 0,03 |

**Tabel 8 – Resultaten Add en Remove Production Period**

De prestatie van beide heuristieken is niet te vergelijken met de resultaten van het genetisch algoritme. Het gemiddelde en standaarddeviatie liggen namelijk veel hoger. Vooral het percentage oplossingen met een fout kleiner dan 1% is erg laag voor beide methoden. Maar het percentage oplossingen met een fout kleiner dan 10% voor beide methoden is hoger dan 95%. De resultaten van beide methoden zijn vergelijkbaar met de Silver Meal heuristiek, waarbij alleen het percentage oplossingen met een fout kleiner dan 1% slechter is in vergelijking met de Silver Meal heuristiek.

Als we de resultaten sorteren op setupkosten (tabel 9) zien we dat beide methoden het tegenovergestelde van elkaar presteren. De add production period methode presteert goed wanneer de setupkosten hoog zijn en de remove production period methode presteert goed wanneer de setupkosten erg laag zijn. Dit verschil kan verklaard worden, namelijk wanneer de setupkosten hoog zijn is het gebruikelijk dat er weinig productieperioden zijn. Dat maakt het voor de add production period methode mogelijk om goede resultaten te leveren. Als het optimaal is om dan in 2 perioden te produceren is het zeer waarschijnlijk dat de add production period methode deze zal vinden, omdat deze met 1 productieperiode begint en telkens een productieperiode toevoegt als de totale kosten niet hoger worden. Het feit dat de remove production perioden slechter presteert met hoge setupkosten komt doordat deze veel meer iteraties nodig heeft dan de add production period methode. Als de optimale oplossing bestaat uit 2 productieperioden dan heeft de remove production period methode $T$ - 2 iteraties nodig, hierbij is het mogelijk dat de productieperiode die de probleeminstantie optimaliseert al verwijderd is. De add production period methode zal altijd de optimale oplossing vinden als er sprake is van slechts twee productieperioden, wat in de eerste iteratie al zal gebeuren. Hetzelfde principe geldt voor lage setupkosten waar de remove production period methode beter werkt dan de add production period methode.

| | Add Production Period | | | Remove Production Period | | |
|---|---|---|---|---|---|---|
| | K=200 | K=500 | K=2000 | K=200 | K=500 | K=2000 |
| Gemiddelde (%) | 3,7 | 2,2 | 1,3 | 1,5 | 3,3 | 4,0 |
| Standaarddeviatie (%) | 3,7 | 2,3 | 1,9 | 1,9 | 2,8 | 4,3 |
| Maximum (%) | 33,4 | 15,6 | 9,2 | 13,0 | 18,2 | 25,7 |

**Tabel 9 - Gevoeligheidsanalyse Add en Remove Production Period gesorteerd op setupkosten (K = 200, 500 en 2000)**

In Appendix C tabel C.1 zijn de resultaten per patroon te vinden. De vraag en return patronen voor beide methoden verschillen minimaal van elkaar.

Hoofdstuk 5

# Conclusie

In deze scriptie hebben we voor het Economic Lot Sizing probleem met returns en gezamenlijke setupkosten diverse methoden toegepast om de huidige resultaten die verkregen zijn met de Silver Meal en Least Unit Costs te verbeteren. Hiervoor hebben we diverse methoden gebruikt die onder te verdelen zijn in twee categorieën, namelijk de methoden die uitgaan van een oneindige, discrete planningshorizon en metaheuristieken. Het voordeel van de eerste categorie methoden is dat deze toepasbaar zijn voor een oneindige planningshorizon, dat wil zeggen dat de uitkomsten niet veranderen als er een verandering optreedt in de planningshorizon. Door het toepassen van de metaheuristieken zijn we te weten gekomen dat de prestaties van de metaheuristieken beter zijn. Dit komt doordat we gebruik hebben gemaakt van een eindige horizon. Het nadeel van deze metaheuristieken is dat de resultaten enkel geldig zijn voor de gegeven planningshorizon en dat deze voor een korter of langer planningshorizon andere resultaten zullen leveren.

Tussen de methoden die van een oneindige, discrete planningshorizon uitgaan hebben we de resultaten van de Silver Meal heuristiek niet kunnen overtreffen. De Marginal Cost Difference heuristiek deed het slechter dan de Least Unit Costs heuristiek. De resultaten van de Marginal Costs Difference heuristieken laten ons zien dat er teveel productieperioden plaatsvinden waardoor de kosten veel hoger uit vallen. Om de resultaten van deze heuristiek te verbeteren is verder onderzoek nodig.
Verder zijn er drie verschillende soorten metaheuristieken toegepast op het probleem. De resultaten van Add en Remove Production Period zijn vergelijkbaar met de resultaten van de Silver Meal heuristiek, alleen het percentage oplossingen met een fout kleiner dan 1% is in vergelijking met de Silver Meal heuristiek slechter. Een groot deel van de oplossingen 95%, had een fout kleiner dan 10%. Tevens zou een combinatie van de Add en Remove Production Period interessante resultaten kunnen leveren. Achteraf bekeken zien we dat de Add Production Period methode beter presteert indien er sprake is van hoge setupkosten en de Remove Production Period methode presteert beter als de setupkosten laag zijn.
We hebben gezien dat het genetisch algoritme over het algemeen betere resultaten levert. Naarmate het aantal startoplossingen toenemen, komen de oplossingen zeer dicht bij het optimum te liggen. Het gemiddelde fout van de GA6 was zelfs 0.004%. Voor het maximum van het genetisch algoritme zien we een dalende lijn naarmate het aantal

startoplossingen toeneemt. Echter neemt de rekentijd van het algoritme wel toe als het aantal startoplossingen toeneemt.
De resultaten van het gewogen genetisch algoritme hadden in tegenstelling tot het genetisch algoritme een veel lager maximum. Een stijging in het aantal startoplossingen had geen significant effect op de maximum fout van de methode, maar wel voor de gemiddelde fout en de standaarddeviatie. Het gewogen genetisch algoritme met 72 startoplossingen gaf als gemiddeld fout een percentage van 0,11. De keuze tussen het genetisch en gewogen genetisch algoritme is moeilijk. Het is een afweging tussen een laag maximum of een laag gemiddelde fout.

Tot slot is het nog interessant om de keuze tussen de Silver Meal heuristiek en het genetisch algoritme te bespreken. Een groot verschil tussen beide methoden is zoals eerder besproken de planningshorizon. Als er sprake is van een onbepaald planningshorizon waarvoor geproduceerd moet worden is het verstandiger om gebruik te maken van het Silver Meal heuristiek. De rekentijd van deze heuristiek is aanzienlijk lager dan het genetisch algoritme. Hierdoor kunnen de wijzigingen in de planningshorizon zeer snel bijgewerkt worden. Een verandering bij het genetisch algoritme kan als gevolg hebben dat de eerdere oplossingen niet meer bruikbaar zijn waardoor de methode opnieuw uitgevoerd moet worden.
Als er echter enkel voor een bepaalde horizon geproduceerd moet worden is het verstandiger om het genetisch algoritme te gebruiken. De resultaten voor een bepaalde planningshorizon zijn bij het genetisch algoritme zeer dicht bij het optimum en het maximum fout van deze methode is in alle gevallen lager dan 10%. Bij de Silver Meal kan de maximum fout oplopen tot 25%.

Hieruit kunnen we concluderen dat in het geval van een eindig planningshorizon het beter is om het genetisch algoritme te gebruikten en in het geval een oneindige planningshorizon het Silver Meal heuristiek te gebruiken.

# Bibliografie


[1] Baker, K.R. (1989), Lot-sizing procedures and a standard data set: a reconciliation of the literature. *Journal of Manufacturing and Operations Management,* 2:199-221

[2] Silver, E.A., Pyke, D.F. & Peterson, R. (1998), Inventory Management and Production Planning and Scheduling. *Wiley*.

[3] Teunter R. H., Bayındır Z. P. & van den Heuvel W. (2006). Dynamic lot sizing with product returns and remanufacturing. *International Journal of Production Research,* 44(20):4377-4400.

[4] Van den Heuvel, W. (2006), The Economic Lot-Sizing Problem – new Results and Extension. *Rotterdam School of Management (RSM) Erasmus University, Erasmus Research Institute of Management (ERIM).*

# APPENDIX A

| | Percentage cost error | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Average | | | Standard Deviation | | | Maximum | | |
| | SM | LUC | MCD | SM | LUC | MCD | SM | LUC | MCD |
| Alle patronen | 3,0 | 4,2 | 10,7 | 4,9 | 6,5 | 9,6 | 25,1 | 37,0 | 65,3 |
| Vraag patroon | | | | | | | | | |
| Stationair | 3,3 | 3,5 | 9,5 | 5,6 | 5,7 | 9,3 | 23,3 | 25,4 | 51,4 |
| - kleine variantie | 2,8 | 2,4 | 8,4 | 5,6 | 4,9 | 6,8 | 20,3 | 22,8 | 44,5 |
| - grote variantie | 3,8 | 4,6 | 10,5 | 5,6 | 6,1 | 11,1 | 23,3 | 25,4 | 51,4 |
| Positieve trend | 2,0 | 2,4 | 17,1 | 3,3 | 3,7 | 11,1 | 22,6 | 22,6 | 65,3 |
| - kleine trend | 1,3 | 3,4 | 11,1 | 2,2 | 4,7 | 6,2 | 22,6 | 22,6 | 39,7 |
| - grote trend | 2,7 | 1,4 | 23,2 | 4,1 | 1,6 | 11,7 | 11,1 | 17,7 | 65,3 |
| Negatieve trend | 3,2 | 3,0 | 7,6 | 4,0 | 4,0 | 5,2 | 18,3 | 18,3 | 31,2 |
| - kleine trend | 3,6 | 2,5 | 7,9 | 4,8 | 2,3 | 4,4 | 18,3 | 15,3 | 23,0 |
| - grote trend | 2,8 | 3,5 | 7,3 | 3,0 | 5,1 | 5,9 | 16,5 | 18,3 | 31,2 |
| Seizoensmatig (piek) | 3,8 | 4,9 | 12,3 | 5,8 | 6,6 | 10,0 | 23,6 | 34,0 | 40,4 |
| - kleine amplitudo | 3,4 | 4,5 | 10,7 | 5,8 | 7,0 | 9,3 | 20,9 | 26,1 | 40,0 |
| - grote amplitudo | 4,2 | 5,3 | 14,0 | 5,8 | 6,2 | 10,4 | 23,6 | 34,0 | 40,4 |
| Seizoensmatig (dal) | 2,9 | 7,3 | 7,2 | 5,1 | 9,6 | 7,2 | 25,1 | 37,0 | 52,1 |
| - kleine amplitudo | 3,3 | 4,6 | 6,7 | 5,7 | 7,5 | 6,2 | 19,4 | 26,7 | 44,6 |
| - grote amplitudo | 2,4 | 9,9 | 7,7 | 4,4 | 10,6 | 8,0 | 25,1 | 37,0 | 52,1 |
| Return patroon | | | | | | | | | |
| Stationair | 2,7 | 3,8 | 10,8 | 4,8 | 6,3 | 9,1 | 22,4 | 30,5 | 49,0 |
| - kleine variantie | 2,7 | 3,8 | 10,7 | 4,8 | 6,3 | 9,1 | 21,9 | 30,3 | 48,5 |
| - grote variantie | 2,7 | 3,8 | 11,0 | 4,8 | 6,4 | 9,1 | 22,4 | 30,5 | 49,0 |
| Positieve trend | 3,9 | 5,1 | 11,1 | 5,3 | 6,8 | 10,6 | 22,6 | 33,2 | 65,3 |
| - kleine trend | 3,5 | 4,5 | 10,7 | 5,1 | 6,6 | 10,2 | 19,5 | 29,6 | 54,3 |
| - grote trend | 4,3 | 5,8 | 11,6 | 5,4 | 7,0 | 11,0 | 22,6 | 33,2 | 65,3 |
| Negatieve trend | 3,3 | 4,8 | 10,0 | 5,0 | 6,7 | 9,9 | 23,6 | 37,0 | 52,1 |
| - kleine trend | 3,5 | 4,8 | 10,6 | 5,2 | 6,8 | 10,4 | 23,6 | 35,5 | 52,1 |
| - grote trend | 3,1 | 4,8 | 9,5 | 4,7 | 6,6 | 9,5 | 21,0 | 37,0 | 50,7 |
| Seizoensmatig (piek) | 2,6 | 3,9 | 11,4 | 4,7 | 6,4 | 9,4 | 21,2 | 30,3 | 49,6 |
| - kleine amplitudo | 2,6 | 3,8 | 11,2 | 4,7 | 6,3 | 9,3 | 21,2 | 30,3 | 49,6 |
| - grote amplitudo | 2,7 | 3,9 | 11,6 | 4,8 | 6,4 | 9,5 | 19,5 | 30,3 | 49,3 |
| Seizoensmatig (dal) | 2,7 | 3,7 | 10,2 | 4,8 | 6,3 | 8,7 | 25,1 | 30,1 | 48,0 |
| - kleine amplitudo | 2,7 | 3,7 | 10,3 | 4,8 | 6,3 | 8,8 | 23,6 | 30,1 | 47,2 |
| - grote amplitudo | 2,7 | 3,6 | 10,1 | 4,8 | 6,2 | 8,7 | 25,1 | 30,1 | 48,0 |
| Setupkosten | | | | | | | | | |
| 200 | 1,1 | 1,5 | 12,5 | 2,7 | 3,1 | 9,7 | 23,6 | 34,0 | 65,3 |
| 500 | 1,9 | 2,4 | 6,6 | 2,7 | 3,4 | 6,8 | 23,3 | 27,5 | 37,5 |
| 2000 | 6,1 | 8,8 | 13,1 | 6,6 | 8,6 | 10,4 | 25,1 | 37,0 | 52,1 |
| Voorraadkosten | | | | | | | | | |
| 0.2 | 2,0 | 2,9 | 10,6 | 4,0 | 5,4 | 9,0 | 19,6 | 32,4 | 51,4 |
| 0.5 | 3,2 | 4,6 | 10,3 | 5,2 | 6,9 | 9,3 | 23,6 | 37,0 | 52,1 |
| 0.8 | 3,8 | 5,1 | 11,3 | 5,3 | 6,9 | 10,3 | 25,1 | 35,5 | 65,3 |

**Tabel A.1 - Resultaten gevoeligheidsanalyse Silver Meal (SM), Least Unit Costs (LUC) en Marginal Cost Difference (MCD)**

# APPENDIX B

| | Percentage cost error | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Gemiddelde | | | Standaarddeviatie | | | Maximum | | |
| | GA2 | GA4 | GA6 | GA2 | GA4 | GA6 | GA2 | GA4 | GA6 |
| Alle patronen | 0,282 | 0,024 | 0,004 | 0,93 | 0,29 | 0,13 | 50,0 | 22,9 | 9,4 |
| Vraag patroon | | | | | | | | | |
| Stationair | 0,362 | 0,031 | 0,004 | 1,26 | 0,44 | 0,13 | 50,0 | 22,9 | 9,4 |
| - kleine variantie | 0,388 | 0,033 | 0,003 | 1,02 | 0,40 | 0,08 | 15,9 | 17,6 | 2,3 |
| - grote variantie | 0,335 | 0,029 | 0,004 | 1,46 | 0,47 | 0,17 | 50,0 | 22,9 | 9,4 |
| Positieve trend | 0,251 | 0,023 | 0,003 | 0,72 | 0,23 | 0,11 | 12,1 | 6,6 | 7,4 |
| - kleine trend | 0,234 | 0,018 | 0,004 | 0,70 | 0,20 | 0,14 | 12,1 | 6,6 | 7,4 |
| - grote trend | 0,267 | 0,029 | 0,003 | 0,75 | 0,25 | 0,08 | 9,6 | 4,1 | 2,7 |
| Negatieve trend | 0,182 | 0,011 | 0,004 | 0,52 | 0,17 | 0,13 | 8,6 | 10,3 | 8,1 |
| - kleine trend | 0,171 | 0,011 | 0,006 | 0,55 | 0,20 | 0,18 | 8,6 | 10,3 | 8,1 |
| - grote trend | 0,192 | 0,011 | 0,001 | 0,49 | 0,12 | 0,04 | 8,0 | 2,9 | 1,3 |
| Seizoensmatig (piek) | 0,299 | 0,026 | 0,002 | 0,76 | 0,21 | 0,07 | 12,0 | 5,1 | 4,0 |
| - kleine amplitudo | 0,329 | 0,030 | 0,003 | 0,82 | 0,24 | 0,07 | 9,1 | 5,1 | 2,5 |
| - grote amplitudo | 0,270 | 0,022 | 0,002 | 0,69 | 0,18 | 0,07 | 12,0 | 3,8 | 4,0 |
| Seizoensmatig (dal) | 0,317 | 0,029 | 0,005 | 1,15 | 0,30 | 0,18 | 46,5 | 13,3 | 9,4 |
| - kleine amplitudo | 0,389 | 0,045 | 0,007 | 1,09 | 0,39 | 0,22 | 31,6 | 13,3 | 9,4 |
| - grote amplitudo | 0,244 | 0,013 | 0,003 | 1,20 | 0,18 | 0,12 | 46,5 | 5,7 | 6,7 |
| Return patroon | | | | | | | | | |
| Stationair | 0,267 | 0,026 | 0,005 | 0,89 | 0,34 | 0,15 | 36,3 | 22,9 | 7,9 |
| - kleine variantie | 0,286 | 0,028 | 0,002 | 1,06 | 0,41 | 0,06 | 36,3 | 22,9 | 2,5 |
| - grote variantie | 0,248 | 0,023 | 0,007 | 0,68 | 0,25 | 0,20 | 6,7 | 10,3 | 7,9 |
| Positieve trend | 0,277 | 0,020 | 0,002 | 0,89 | 0,22 | 0,08 | 29,3 | 6,6 | 4,0 |
| - kleine trend | 0,259 | 0,023 | 0,001 | 0,75 | 0,22 | 0,03 | 8,6 | 5,1 | 1,2 |
| - grote trend | 0,295 | 0,018 | 0,004 | 1,01 | 0,22 | 0,10 | 29,3 | 6,6 | 4,0 |
| Negatieve trend | 0,318 | 0,027 | 0,003 | 1,10 | 0,36 | 0,13 | 46,5 | 17,6 | 9,4 |
| - kleine trend | 0,332 | 0,037 | 0,006 | 1,19 | 0,48 | 0,18 | 46,5 | 17,6 | 9,4 |
| - grote trend | 0,303 | 0,018 | 0,001 | 1,00 | 0,19 | 0,04 | 31,6 | 4,3 | 2,1 |
| Seizoensmatig (piek) | 0,265 | 0,021 | 0,004 | 0,70 | 0,21 | 0,15 | 9,6 | 7,2 | 9,4 |
| - kleine amplitudo | 0,268 | 0,020 | 0,004 | 0,68 | 0,19 | 0,18 | 6,1 | 3,1 | 9,4 |
| - grote amplitudo | 0,262 | 0,022 | 0,004 | 0,72 | 0,24 | 0,11 | 9,6 | 7,2 | 5,5 |
| Seizoensmatig (dal) | 0,292 | 0,025 | 0,004 | 1,02 | 0,23 | 0,12 | 50,0 | 4,3 | 8,1 |
| - kleine amplitudo | 0,272 | 0,021 | 0,001 | 0,73 | 0,21 | 0,03 | 7,9 | 4,3 | 1,7 |
| - grote amplitudo | 0,312 | 0,029 | 0,006 | 1,24 | 0,25 | 0,17 | 50,0 | 4,3 | 8,1 |
| Setupkosten | | | | | | | | | |
| 200 | 0,293 | 0,031 | 0,004 | 0,82 | 0,26 | 0,12 | 12,1 | 5,3 | 9,4 |
| 500 | 0,340 | 0,032 | 0,005 | 0,82 | 0,30 | 0,15 | 13,7 | 13,3 | 8,1 |
| 2000 | 0,213 | 0,009 | 0,002 | 1,11 | 0,29 | 0,11 | 50,0 | 22,9 | 9,4 |
| Voorraadkosten | | | | | | | | | |
| 0.2 | 0,278 | 0,020 | 0,005 | 0,88 | 0,26 | 0,18 | 36,3 | 17,6 | 9,4 |
| 0.5 | 0,290 | 0,026 | 0,003 | 1,06 | 0,27 | 0,11 | 50,0 | 13,3 | 9,4 |
| 0.8 | 0,278 | 0,026 | 0,003 | 0,82 | 0,33 | 0,09 | 29,3 | 22,9 | 7,4 |

**Tabel B.1 – Resultaten gevoeligheidsanalyse genetisch algoritme waarbij GAx staat voor het aantal startoplossingen xT, het aantal iteraties voor alle drie de methoden is 12.**

| | Percentage cost error | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Gemiddelde | | | Standaarddeviatie | | | Maximum | | |
| | GGa2 | GGa4 | GGa6 | GGa2 | GGa4 | GGa6 | GGa2 | GGa4 | GGa6 |
| Alle patronen | 0,32 | 0,17 | 0,11 | 0,83 | 0,60 | 0,49 | 9,44 | 6,42 | 7,36 |
| Vraag patroon | | | | | | | | | |
| Stationair | 0,39 | 0,21 | 0,14 | 0,97 | 0,70 | 0,59 | 6,8 | 6,1 | 6,8 |
| - kleine variantie | 0,43 | 0,23 | 0,17 | 1,05 | 0,76 | 0,67 | 6,8 | 6,1 | 5,8 |
| - grote variantie | 0,34 | 0,18 | 0,12 | 0,88 | 0,63 | 0,50 | 6,8 | 5,5 | 6,8 |
| Positieve trend | 0,30 | 0,18 | 0,13 | 0,77 | 0,65 | 0,55 | 7,8 | 6,4 | 6,8 |
| - kleine trend | 0,18 | 0,09 | 0,05 | 0,62 | 0,47 | 0,36 | 7,8 | 6,4 | 6,8 |
| - grote trend | 0,41 | 0,28 | 0,22 | 0,89 | 0,77 | 0,67 | 4,8 | 4,6 | 4,0 |
| Negatieve trend | 0,17 | 0,09 | 0,05 | 0,51 | 0,36 | 0,27 | 6,0 | 5,0 | 4,2 |
| - kleine trend | 0,15 | 0,06 | 0,03 | 0,51 | 0,33 | 0,21 | 6,0 | 5,0 | 4,2 |
| - grote trend | 0,20 | 0,12 | 0,08 | 0,51 | 0,39 | 0,32 | 5,4 | 3,4 | 2,6 |
| Seizoensmatig (piek) | 0,43 | 0,23 | 0,14 | 0,93 | 0,66 | 0,50 | 8,0 | 5,3 | 7,4 |
| - kleine amplitudo | 0,52 | 0,28 | 0,17 | 1,05 | 0,74 | 0,58 | 8,0 | 5,0 | 7,4 |
| - grote amplitudo | 0,34 | 0,18 | 0,10 | 0,78 | 0,56 | 0,40 | 5,4 | 5,3 | 3,4 |
| Seizoensmatig (dal) | 0,32 | 0,15 | 0,09 | 0,87 | 0,58 | 0,46 | 9,4 | 6,1 | 5,9 |
| - kleine amplitudo | 0,43 | 0,21 | 0,13 | 1,02 | 0,70 | 0,57 | 8,9 | 5,9 | 5,9 |
| - grote amplitudo | 0,20 | 0,09 | 0,06 | 0,66 | 0,43 | 0,32 | 9,4 | 6,1 | 5,6 |
| Return patroon | | | | | | | | | |
| Stationair | 0,28 | 0,16 | 0,10 | 0,74 | 0,57 | 0,45 | 6,1 | 6,1 | 5,3 |
| - kleine variantie | 0,29 | 0,16 | 0,10 | 0,75 | 0,56 | 0,45 | 6,1 | 6,1 | 5,1 |
| - grote variantie | 0,27 | 0,17 | 0,11 | 0,73 | 0,57 | 0,46 | 5,8 | 4,6 | 5,3 |
| Positieve trend | 0,36 | 0,19 | 0,13 | 0,93 | 0,67 | 0,56 | 9,4 | 6,4 | 6,8 |
| - kleine trend | 0,34 | 0,17 | 0,13 | 0,88 | 0,61 | 0,53 | 8,0 | 5,3 | 4,3 |
| - grote trend | 0,37 | 0,20 | 0,13 | 0,98 | 0,73 | 0,59 | 9,4 | 6,4 | 6,8 |
| Negatieve trend | 0,35 | 0,18 | 0,11 | 0,90 | 0,63 | 0,49 | 8,0 | 6,0 | 7,4 |
| - kleine trend | 0,40 | 0,21 | 0,13 | 0,96 | 0,69 | 0,54 | 8,0 | 5,5 | 6,8 |
| - grote trend | 0,30 | 0,16 | 0,09 | 0,83 | 0,57 | 0,44 | 7,8 | 6,0 | 7,4 |
| Seizoensmatig (piek) | 0,29 | 0,14 | 0,10 | 0,75 | 0,53 | 0,45 | 6,2 | 5,0 | 4,0 |
| - kleine amplitudo | 0,28 | 0,14 | 0,11 | 0,74 | 0,53 | 0,48 | 6,2 | 5,0 | 4,0 |
| - grote amplitudo | 0,30 | 0,14 | 0,09 | 0,75 | 0,53 | 0,43 | 5,4 | 4,2 | 3,7 |
| Seizoensmatig (dal) | 0,34 | 0,18 | 0,11 | 0,87 | 0,62 | 0,49 | 6,8 | 5,0 | 5,8 |
| - kleine amplitudo | 0,35 | 0,17 | 0,12 | 0,87 | 0,60 | 0,50 | 6,8 | 4,6 | 5,0 |
| - grote amplitudo | 0,34 | 0,19 | 0,11 | 0,87 | 0,64 | 0,49 | 6,4 | 5,0 | 5,8 |
| Setupkosten | | | | | | | | | |
| 200 | 0,45 | 0,25 | 0,17 | 1,00 | 0,73 | 0,61 | 6,8 | 6,1 | 5,8 |
| 500 | 0,47 | 0,25 | 0,16 | 0,93 | 0,70 | 0,56 | 9,4 | 6,4 | 6,8 |
| 2000 | 0,05 | 0,01 | 0,01 | 0,31 | 0,18 | 0,13 | 6,8 | 5,5 | 7,4 |
| Voorraadkosten | | | | | | | | | |
| 0.2 | 0,31 | 0,17 | 0,11 | 0,83 | 0,61 | 0,50 | 6,2 | 6,1 | 5,3 |
| 0.5 | 0,32 | 0,17 | 0,11 | 0,83 | 0,59 | 0,47 | 7,8 | 6,2 | 5,4 |
| 0.8 | 0,33 | 0,18 | 0,12 | 0,85 | 0,61 | 0,49 | 9,4 | 6,4 | 7,4 |

**Tabel B.2 – Resultaten gewogen genetisch algoritme waarbij GGax staat het aantal startoplossingen xT, aantal iteraties m is 50, het percentage dat gecombineerd word is 80% en het percentage dat gemuteerd word is 30% voor alle drie de methoden.**

# APPENDIX C

| | Percentage cost error | | | | | |
|---|---|---|---|---|---|---|
| | Gemiddelde | | Standaarddeviatie | | Maximum | |
| | Add | Remove | Add | Remove | Add | Remove |
| Alle patronen | 2,4 | 2,9 | 2,9 | 3,3 | 33,4 | 25,7 |
| Vraag patroon | | | | | | |
| Stationair | 2,5 | 3,2 | 3,8 | 3,5 | 33,4 | 24,6 |
| - kleine variantie | 2,9 | 3,5 | 4,1 | 3,6 | 23,0 | 24,6 |
| - grote variantie | 2,0 | 2,9 | 3,4 | 3,3 | 33,4 | 22,1 |
| Positieve trend | 3,0 | 2,7 | 2,2 | 3,7 | 22,3 | 16,0 |
| - kleine trend | 3,0 | 2,3 | 2,1 | 3,0 | 22,3 | 16,0 |
| - grote trend | 3,0 | 3,1 | 2,3 | 4,3 | 11,7 | 11,7 |
| Negatieve trend | 2,6 | 2,3 | 2,5 | 3,2 | 32,2 | 16,7 |
| - kleine trend | 2,7 | 2,0 | 2,8 | 2,7 | 32,2 | 16,7 |
| - grote trend | 2,6 | 2,7 | 2,1 | 3,6 | 9,8 | 13,8 |
| Seizoensmatig (piek) | 2,1 | 3,3 | 3,2 | 3,2 | 27,7 | 25,7 |
| - kleine amplitudo | 2,3 | 3,8 | 3,3 | 3,4 | 24,7 | 25,7 |
| - grote amplitudo | 1,9 | 2,9 | 3,0 | 2,9 | 27,7 | 19,2 |
| Seizoensmatig (dal) | 1,7 | 3,1 | 2,5 | 2,9 | 18,1 | 22,7 |
| - kleine amplitudo | 1,9 | 3,0 | 2,9 | 2,9 | 18,1 | 20,2 |
| - grote amplitudo | 1,4 | 3,2 | 2,0 | 2,9 | 17,2 | 22,7 |
| Return patroon | | | | | | |
| Stationair | 2,1 | 2,7 | 2,5 | 3,0 | 12,9 | 22,8 |
| - kleine variantie | 2,1 | 2,6 | 2,5 | 2,9 | 12,3 | 19,2 |
| - grote variantie | 2,1 | 2,7 | 2,5 | 3,0 | 12,9 | 22,8 |
| Positieve trend | 2,8 | 3,1 | 3,1 | 3,7 | 22,3 | 22,7 |
| - kleine trend | 2,6 | 3,1 | 2,9 | 3,6 | 19,0 | 22,7 |
| - grote trend | 3,0 | 3,1 | 3,3 | 3,7 | 22,3 | 20,2 |
| Negatieve trend | 2,9 | 3,5 | 3,7 | 3,8 | 33,4 | 25,7 |
| - kleine trend | 3,0 | 3,6 | 4,0 | 3,9 | 33,4 | 25,7 |
| - grote trend | 2,8 | 3,5 | 3,4 | 3,8 | 32,2 | 20,0 |
| Seizoensmatig (piek) | 2,1 | 2,6 | 2,5 | 2,9 | 13,6 | 16,1 |
| - kleine amplitudo | 2,1 | 2,6 | 2,5 | 2,9 | 13,2 | 15,5 |
| - grote amplitudo | 2,1 | 2,7 | 2,6 | 2,9 | 13,6 | 16,1 |
| Seizoensmatig (dal) | 2,2 | 2,8 | 2,7 | 3,2 | 12,4 | 24,6 |
| - kleine amplitudo | 2,2 | 2,7 | 2,6 | 3,1 | 12,1 | 23,6 |
| - grote amplitudo | 2,3 | 2,9 | 2,7 | 3,4 | 12,4 | 24,6 |
| Setupkosten | | | | | | |
| 200 | 3,7 | 1,5 | 3,7 | 1,9 | 33,4 | 13,0 |
| 500 | 2,2 | 3,3 | 2,3 | 2,8 | 15,6 | 18,2 |
| 2000 | 1,3 | 4,0 | 1,9 | 4,3 | 9,2 | 25,7 |
| Voorraadkosten | | | | | | |
| 0.2 | 2,1 | 3,0 | 2,7 | 3,2 | 13,6 | 25,7 |
| 0.5 | 2,4 | 3,0 | 2,9 | 3,3 | 23,6 | 24,6 |
| 0.8 | 2,7 | 2,8 | 3,2 | 3,4 | 33,4 | 20,6 |

**Tabel C.1 – Resultaten gevoeligheidsanalyse Add Production Period en Remove Production Period heuristiek.**