Entropie en Informatie: Modellen van dobbelstenen met gebroken symmetrie

Reinier Jonker, Menno de Bell en Andries van der Leden
onder begeleiding van Prof. F.A. Bais

Juni 2007

Samenvatting

Een dobbelsteen is een object met een zeer grote symmetriegroep. Bij het werpen van een homogene dobbelsteen is de kans voor elke zijde om boven te komen even groot. Als de symmetrie echter wordt gebroken zijn de kansen niet meer gelijk.

Dit onderzoek betreft modellen met een minimaal aantal parameters van zeszijdige dobbelstenen met gebroken symmetrie. Hiervoor wordt gebruik gemaakt van het maximale-entropie-beginsel. Het blijkt mogelijk om modellen te maken uitgaande van dit principe. Het blijkt dat om een goede kansverdeling te verkrijgen in de meeste gevallen vier of vijf statistische gegevens van dobbelsteenworpen nodig zijn.

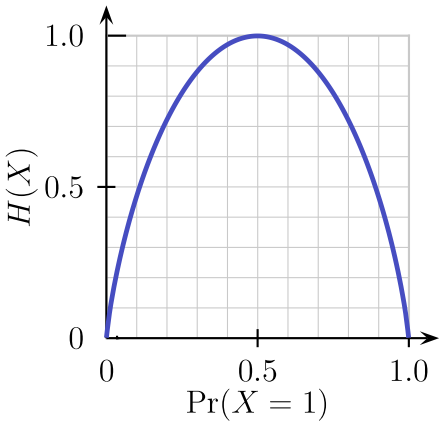
Inhoudsopgave

1 Inleiding

Entropie en informatie zijn onlosmakelijk met elkaar verbonden. Voor elk systeem dat informatie bevat kan een entropie gedefiniëerd worden. De entropie is de hoeveelheid informatie die in het systeem zit. Claude E. Shannon definiëerde entropie als volgt:

$$H(X) = \sum_{i=1}^{n} p(x_i) \log_2(\frac{1}{p(x_i)}) = -\sum_{i=1}^{n} p(x_i) \log_2(p(x_i)) \tag{1}$$

Zo kunnen we bijvoorbeeld een munt omhoog gooien waarbij zowel de kopzijde als de muntzijde boven kan komen. Als we niets weten is de beste aanname dat de kans op beide mogelijkheden even groot is. In dit geval is de entropie maximaal: het systeem bevat veel informatie. Als we echter weten dat bijvoorbeeld één kant van de munt zwaarder is, dan neemt de kans dat de andere zijde bovenop komt toe. We weten dus al meer over het systeem en het heeft ons minder te vertellen. Het zou echter ook kunnen dat door een vreemde onbekende invloed het van te voren zeker is dat de munt met de muntzijde boven komt. In dit geval weten we alles van tevoren en geeft het systeem ons geen informatie: de entropie is minimaal. De entropie is dus een symmetrische functie rondom $P_{\text{kop}} = P_{\text{munt}}$, zoals te zien in figuur 1.



Figuur 1: Grafiek van de entropie als functie van de kans dat de muntzijde boven komt.

In dit project kijken we naar dobbelstenen, platonische lichamen met aan elke zijde een nummer. Na het werpen komt slechts één nummer 'boven' liggen. De kans dat een bepaalde zijde boven ligt is bij een homogene dobbelsteen gelijk voor elke zijde. Als er echter ongelijkheden zijn, wordt deze

symmetrie gebroken. De kans wordt dan afhankelijk van welke zijde het betreft en de dobbelsteen krijgt als symmetriegroep een ondergroep van de groep voor een homogene dobbelsteen.

Wij hebben modellen gemaakt van dobbelstenen met onregelmatigheden die de symmetrie breken. Uitgaande van het maximale-entropie-beginsel hebben we modellen gemaakt om met zo min mogelijk gegevens de kansverdeling van een dobbelsteen te kunnen voorspellen.

Het project bestaat uit vier onderdelen:

1. Er wordt gekeken naar het verband tussen entropie en informatie, het maximale-entropie-beginsel en het definiëren van entropiefuncties voor perfecte en imperfecte dobbelstenen.

2. Aan de hand van de symmetrie beginselen worden imperfecties geclassificeerd als een vorm van symmetriebreking. Dit is in detail uitgewerkt voor de gewone dobbelsteen.

3. Het bouwen van een 'event generator' die worpen genereert voor dobbelstenen met gegeven kansen en data teruggeeft in de vorm van gemiddelden.

4. Gebruikmakend van de data uit de event generator en het maximale-entropie-beginsel wordt vervolgens analytisch en numeriek de kansverdeling van een aantal imperfecte dobbelstenen gemodelleerd.

2 Entropie en informatie

2.1 Eenheid van informatie

Entropie en informatie zijn twee termen die onlosmakelijk met elkaar verbonden zijn. In 1948 vond C.E. Shannon [1] een algemene uitdrukking voor de informatietheorie waarbij hij al opmerkte dat deze veel wegheeft van de entropiefunctie van Boltzmann.

Deze functie is een logaritmische functie waarbij het afhangt van de eenheid welk grondgetal er gebruikt wordt. 'Binary digits' (beter bekend als bits) gebruiken bijvoorbeeld het grondgetal 2, maar 'decimal digits' (dets) gebruikt het grondgetal 10. Daaruit volgt dat $\log_2 M = \log_{10} M \frac{log(10)}{log(2)} \approx 3,32 \log_{10} M$ en zie je dat 1 decimal digit ongeveer gelijk is aan $3\frac{1}{3}$ bits.

2.2 Informatiefunctie

C. E. Shannon vroeg zich in 1948 af hoe een bron mathematisch beschreven kan worden en hoeveel informatie er is geproduceerd in een bron. Een belangrijk punt hierin is de beschikbare kennis die je van de bron hebt, bijvoorbeeld de kansen van symbolen die je verstuurd.

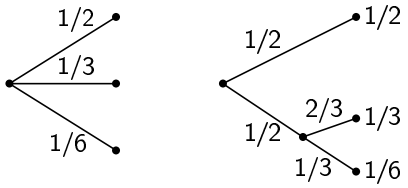
In een taal bijvoorbeeld worden reeksen letters verstuurd en in het Nederlands is de kans op een **E** veel groter dan die van **Q**. Door deze verschillende kansen zit er in een bepaalde reeks letters minder informatie dan als de kansen gelijk waren. Hier kan gebruik van gemaakt worden om dezelfde informatie met minder eenheden over te dragen. Zo wordt bij Morsecode de **E** een **.** en de **Q** wordt **- - . -**.

Een bron maakt een set symbolen met een bepaalde waarschijnlijkheid. Dit geldt voor een taal, maar ook voor een dobbelsteen. Een dobbelsteen heeft de getallenset $\{1, 2, ..., 6\}$ met bijbehorende kansen $\{p_1, p_2, ..., p_6\}$. Bij een normale dobbelsteen is elke p_i gelijk aan $\frac{1}{6}$ en in dit geval is de informatie van het systeem maximaal.

Stel $\{p_1, p_2, ..., p_N\}$ is een set met kansen behorend bij bepaalde gebeurtenissen. Kun je dan bij een meting iets zeggen over een informatiefunctie $H(p_1, p_2, ..., p_N)$? Dat kan met de volgende redelijke, zo niet noodzakelijke condities:

1. H continu is voor alle p_i.
2. alle p_i gelijk zijn ($p_i = \frac{1}{N}$), dan is H monotoom stijgend.
3. een keuze wordt afgebroken in twee succesvolle keuzes, dan is H de som van de individuele H's, $H(x, y) = H(x) + H_x(y)$, waarbij $H(x, y)$

de gezamenlijke entropie ('joint entropy') is en $H_x(y)$ de conditionele entropie ('conditional entropy').



Figuur 2: Kansboom van het voorbeeld.

Een voorbeeld van regel (3) is $p_1 = \frac{1}{2}$, $p_2 = \frac{1}{3}$ en $p_3 = \frac{1}{6}$ voor de situaties 1, 2 en 3. Elke kans kan afzonderlijk beschouwd worden, maar situatie 2 en 3 kunnen ook samen als één situatie '2 of 3' gezien worden, waarop de kans dan gelijk is aan $p_2 + p_3 = \frac{1}{2}$. De situatie '2 of 3' kan weer verdeeld worden in situatie 2 met een kans van $\frac{2}{3}$ situatie 3 met een kans van $\frac{1}{3}$. Uiteindelijk volgt dan uit regel (3) dat:

$$H(\frac{1}{2}, \frac{1}{3}, \frac{1}{6}) = H(\frac{1}{2}, \frac{1}{2}) + \frac{1}{2}H(\frac{2}{3}, \frac{1}{3}) \tag{2}$$

Voor de laatste H staat een factor $\frac{1}{2}$ omdat dit gebeurt in de helft van de gevallen. Hoe ziet H er dan uit?

Laat $H(\frac{1}{N}, \frac{1}{N}, ..., \frac{1}{N}) = A(n)$. Uit de derde conditie volgt dat door de juiste keuze s^m afgebroken kan worden tot een serie van m met s gelijke kansen, dus $A(s^m) = mA(s)$. Zo kan ook $A(t^n) = nA(t)$. We kiezen een n en zoeken een m zodat geldt:

$$s^m \leq t^n \leq s^{m+1} \tag{3}$$

Neem hiervan de logaritmes en deel vervolgens door $n \log(s)$:

$$\frac{m}{n} \leq \frac{\log(t)}{\log(s)} \leq \frac{m}{n} + \frac{1}{n} \quad \text{of} \quad \left|\frac{m}{n} - \frac{\log(t)}{\log(s)}\right| < \epsilon \tag{4}$$

waarbij ϵ erg klein is. H is monotoom stijgend, dus $A(n)$ ook en dus met (3):

$$A(s^m) \leq A(t^n) \leq A(s^{m+1}) \tag{5}$$

$$mA(s) \leq nA(t) \leq (m+1)A(s) \tag{6}$$

Vervolgens delen $nA(s)$:

$$\frac{m}{n} \leq \frac{A(t)}{A(s)} \leq \frac{m}{n} + \frac{1}{n} \quad \text{of} \quad \left|\frac{m}{n} - \frac{A(t)}{A(s)}\right| < \epsilon \tag{7}$$

$$\left|\frac{A(t)}{A(s)} - \frac{\log(t)}{\log(s)}\right| < 2\epsilon \quad A(t) = K \log t \tag{8}$$

Waar K positief moet zijn om nog aan de tweede conditie te voldoen. Stel dat we nu een keuze hebben uit n mogelijkheden met de kansen $p_i = \frac{n_i}{\sum n_i}$ waar de n_i integers zijn. We kunnen $\sum n_i$ mogelijkheden afbreken naar n mogelijkheden met de kansen $p_1, p_2, ..., p_n$ en dan kan er dus een keuze worden gemaakt van n_i met gelijke kansen. Met behulp van de derde conditie krijgen we een uitdrukking voor de totale keuze voor $\sum n_i$:

$$K \log(\sum n_i) = H(p_1, ..., p_n) + K \sum p_i \log(n_i) \tag{9}$$

$$\begin{aligned} H &= K(\sum p_i \log(\sum n_i) - \sum p_i \log(n_i)) \\ &= -K \sum p_i \log(\frac{n_i}{\sum n_i}) = -K \sum p_i \log p_i \end{aligned} \tag{10}$$

Uit deze set met condities volgt dus:

$$H = -k \sum_{i=1}^{N} p_i \log(p_i) \tag{11}$$

waarbij de k een bepaalde constante is die in de meeste gevallen gelijk aan 1 wordt gesteld. Deze functie is de algemene functie voor informatie en het komt overeen met de entropiefunctie van Boltzmann. We kunnen zeggen dat $H = -k \sum_{i=1}^{N} p_i \log(p_i)$ de informatie van de set kansen $\{p_1, p_2, ..., p_N\}$ is.

Laten we een voorbeeld van informatie met grondgetal 2 van een zeszijdige dobbelsteen geven. We nemen eerst een perfecte dobbelsteen waarbij $p_i = \frac{1}{N} = \frac{1}{6}$:

$$H = -\sum_{i=1}^{6} p_i \log_2(p_i) = -\sum_{i=1}^{6} \frac{1}{6} \log_2(\frac{1}{6}) = -\log_2(\frac{1}{6}) \approx 2,6 \tag{12}$$

Een dobbelsteen met andere kansen, bijvoorbeeld $\{\frac{3}{10}, \frac{1}{10}, \frac{1}{10}, \frac{1}{10}, \frac{1}{10}, \frac{3}{10}\}$ heeft een informatie van

$$H = -\sum_{i=1}^{6} p_i \log_2(p_i) = -(\frac{6}{10} \log_2(\frac{3}{10}) + \frac{4}{10} \log_2(\frac{1}{10})) \approx 2,4 \tag{13}$$

De informatie is dus afgenomen. De informatie kan je dus interpreteren als volgt: Hoe hoger de informatie, hoe minder je weet over het systeem. Later wordt de log van de informatie functie vervangen door ln oftewel een log met grondgetal e, dit zodat het rekenwerk verderop (bij de modeloptimalisatie) iets simpeler wordt.

3 Symmetriegroepen van platonische dobbelstenen

3.1 De normale kubische dobbelsteen

De symmetriegroep van een object bestaat uit alle rotaties die je op het object kan uitvoeren zonder dat het object daarbij een andere vorm in de ruimte inneemt. Een groep wordt genoteerd als $G = \{g_i\}$ waarbij $\mathbf{G}$ de groep is en g_i de elementen waarbij i van 1 tot het aantal elementen N loopt. Een twee-dimensionale driehoek heeft een symmetriegroep die bestaat uit drie elementen, een rotatie over 120 graden, eentje over 240 graden en een rotatie over 360 graden, ofwel geen rotatie. Dit laatste 'triviale' element wordt de eenheid genoemd, en vaak aangegeven met $\mathbf{e}$. Het aanwezig zijn van het triviale element is een van de eisen waar een symmetriegroep aan moet voldoen, voor de eenheid geldt:

$$\mathbf{e}g_i = g_i\mathbf{e} = g_i \tag{14}$$

Als tweede moet het resultaat van twee elementen gelijk zijn aan het resultaat een enkel (ander) element van de groep:

$$g_1 g_2 = g_3 \tag{15}$$

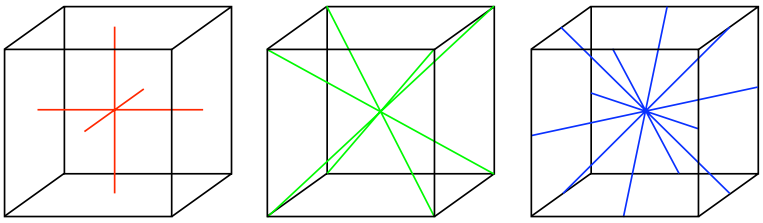
$$g_1, g_2 \in G \Rightarrow g_3 \in G \tag{16}$$

Een derde belangrijke conditie is het bestaan van inverse elementen, die de actie van een bepaald element teniet doen.

$$gg^{-1} = g^{-1}g = \mathbf{e} \tag{17}$$

Als we bij een dobbelsteen stellen dat het nummer op een zijde geen invloed heeft op de symmetrie (en we in feite kijken naar een 'kale' kubus) bevat de symmetriegroep maar liefst 24 elementen. In fysische zin vormen deze 24 elementen niet alleen de 24 rotaties die men kan doen op dobbelsteen, maar ook de 24 verschillende oriëntaties die een dobbelsteen bezit. Het aantal oriëntaties komt direct naar voren uit het feit dat er zes manieren zijn om een bepaalde zijde 'boven' te kiezen waarna er nog vier mogelijkheden zijn om de zijde 'voor' te kiezen. Bij de 24 rotaties wordt de dobbelsteen geroteerd om 13 verschillende assen, er zijn 3 viervoudige assen (over 90°), 4 drievoudige assen (over 120°) en 6 tweevoudige assen (over 180°). Deze assen staan afgebeeld in Figuur 3.

De symmetriegroep van de dobbelsteen is de zogenaamde Orthogonale groep en wordt aangegeven met $\mathbf{O}$. Ondanks dat deze groep vaak de kubische



Figuur 3: drie assen met vier rotaties, vier assen met drie rotaties en zes assen met twee rotaties

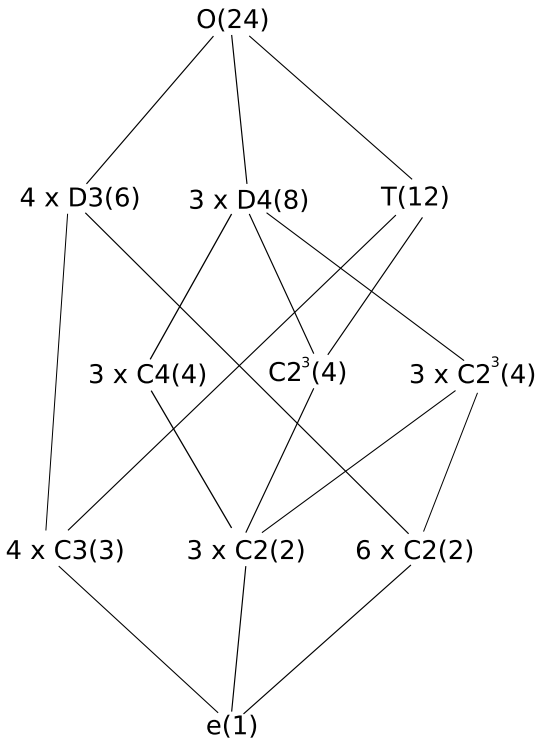
groep wordt genoemd, is er een scala aan platonische lichamen die dezelfde symmetriegroep hebben. Een opmerkelijk voorbeeld hiervan is de octagon, de 8-zijdige broer van de kubus.

3.2 Ondergroepen

De symmetriegroep van een dobbelsteen kan worden gebroken door een onregelmatigheid in te voeren. Je kunt hierbij denken aan een productiefout of een kwaadwillende casino-eigenaar. Stel dat we een plakje van de onderzijde van een dobbelsteen afzagen en vervangen door een metalen plaatje. Dan is elke rotatie die de onderzijde naar een andere kant roteert niet langer geldig. De overige rotaties (in dit geval de rotaties om de verticale rotatieas) vormen samen een ondergroep. Voor een complete tabel met de groepen en de bijbehorende elementen zie bijlage 1.

Een ondergroep moet zoals elke groep voldoen aan de eerder genoemde drie condities. De kleinste ondergroep is de groep die enkel bestaat uit de eenheid en waarbij alle symmetriën zijn gebroken. Ondergroepen komen voor in verschillende soorten en maten, en het komt vaak voor dat een grote ondergroep andere, kleinere ondergroepen geheel omvat. Zo bevat de tetrahedrale groep (met 12 elementen) maar liefst vijf verschillende ondergroepen, waarvan de grootste op zijn beurt nog eens drie kleinere groepen omvat.

Met behulp van deze relaties tussen subgroepen kan een 'groepenboom' (zie Fig. 3.2) worden gemaakt. In een dergelijke groepenboom staat de grootste groep bovenaan, met aftakking naar al zijn directe ondergroepen (diegene die niet worden omvat door een andere ondergroep) die weer aftakkingen hebben naar hún directe ondergroepen. Onderaan de boom komen alle aftakkingen samen bij de triviale groep, met als enige element de eenheid.



Figuur 4: Groepenboom van de groep S_4

3.3 Kansverdeling

In veel gevallen heeft het breken van de symmetriegroep van een dobbelsteen naar een kleinere ondergroep als gevolg dat de kansverdeling veranderd. Bij een perfecte dobbelsteen met een volledige symmetriegroep is de kans p_i dat een bepaalde zijde boven komt te liggen voor elk nummer gelijk dankzij de rotaties die elke zijde kunnen laten overgaan in een andere. Er geldt dus: $p_1 = p_2 = p_3 = p_4 = p_5 = p_6 = q_1$.

Bij de meeste ondergroepen is hier echter geen sprake van. Neem bijvoorbeeld de eerder genoemde dobbelsteen met het metalen plaatje. Hier zijn alleen rotaties die de zijkanten van de dobbelsteen in elkaar laten overgaan; de overige twee vlakken blijven onder elke rotatie stationair. In plaats van één enkele kans die geldt voor elke zijde is er sprake van drie verschillende kansen, één voor de zijkanten, een ander voor de bovenkant en een laatste voor de onderkant: $p_1 = q_1$, $p_2 = p_3 = p_4 = p_5 = q_2$ en $p_6 = q_3$. Over het algemeen geldt dat een dobbelsteen met veel onregelmatigheden ook veel verschillende kansen kent. Zie figuur (3.3) voor een compleet overzicht van kansverdelingen bij de behorende symmetriegroepen.

	p1	p2	p3	p4	p5	p6
O(24)	q1	q1	q1	q1	q1	q1
T(12)	q1	q1	q1	q1	q1	q1
D4(8)	q2	q1	q1	q1	q1	q2
D4(8)	q1	q2	q1	q1	q2	q1
D4(8)	q1	q1	q2	q2	q1	q1
D3(6)	q1	q1	q1	q1	q1	q1
D3(6)	q1	q1	q1	q1	q1	q1
D3(6)	q1	q1	q1	q1	q1	q1
D3(6)	q1	q1	q1	q1	q1	q1
C2^3(4)	q2	q1	q1	q1	q1	q2
C2^3(4)	q1	q2	q1	q1	q2	q1
C2^3(4)	q1	q1	q2	q2	q1	q1
C2^3(4)	q1	q2	q3	q3	q2	q1
C4(4)	q2	q1	q1	q1	q1	q3
C4(4)	q1	q2	q1	q1	q3	q1
C4(4)	q1	q1	q2	q3	q1	q1
C3(3)	q1	q1	q1	q2	q2	q2
C3(3)	q1	q1	q2	q1	q2	q2
C3(3)	q1	q2	q2	q1	q1	q2
C3(3)	q1	q2	q1	q2	q1	q2
C2(2)	q1	q2	q3	q3	q2	q1
C2(2)	q1	q2	q3	q3	q2	q1
C2(2)	q1	q2	q3	q3	q2	q1
C2(2)	q1	q2	q3	q3	q2	q1
C2(2)	q1	q2	q3	q3	q2	q1
C2(2)	q1	q2	q3	q3	q2	q1
C2(2)	q3	q1	q2	q2	q1	q4
C2(2)	q1	q3	q2	q2	q4	q1
C2(2)	q1	q2	q3	q4	q2	q1

Figuur 5: Kansverdeling voor de verschillende ondergroepen van de orthogonale groep.

4 Het analytische model

4.1 Inleiding

De bedoeling van het onderzoek is om uiteindelijk een model van een dobbelsteen te maken aan de hand van een minimale set gegevens van gegenereerde dobbelsteenworpen. Dit model is gemaakt aan de hand van de informatiefunctie en bijbehorende voorwaarden, door een Lagrangiaan op te stellen waarmee het maximum van de informatiefunctie binnen de voorwaarden berekend kan worden.

Met de voorwaarden (*constraints*) blijft er een ruimte van mogelijke oplossingen over. Omdat het systeem naar een maximale informatie zoekt, is de meest waarschijnlijke oplossing het maximum van de informatiefunctie.

Dit probleem kan op twee manieren berekend worden, namelijk analytisch en numeriek. Doordat er steeds meer beperkende voorwaarden aan het probleem worden toegevoegd wordt het steeds lastiger een oplossing te vinden.

4.2 Beperkende voorwaarden

Er zijn beperkende voorwaarden nodig om elke kans van alle zijden van een dobbelsteen vast te leggen. Hoe meer voorwaarden er gebruikt worden, des te meer de kansen vast komen te liggen. Het maximale aantal beperkende voorwaarden dat nodig is is gelijk aan het aantal zijden van de dobbelsteen. De eerste voorwaarde is de meest triviale, namelijk dat de som van alle kansen bij elkaar 1 is, als je met een dobbelsteen werpt komt er altijd wel een getal boven. De voorwaarde ziet er dan als volgt uit:

$$\sum_{i=1}^{N} p_i = 1 \tag{18}$$

De p_i in deze formule is de kans van de zijde i, in totaal zijn er N zijdes, de totale kans van alle zijdes bij elkaar is 1, bij een dobbelsteen is N dus zes. Om de voorwaarde in de Lagrangiaan in te voeren moet het omgeschreven worden tot $\lambda(\sum_{i=1}^{N} p_i - 1)$, λ is een onbekende parameter die aan het systeem wordt toegevoegd en is later nodig voor het opschrijven van de vergelijking van de p_i. Vervolgens worden de andere voorwaarden gemaakt aan de hand van de eerste voorwaarde. De tweede voorwaarde ziet er als volgt uit:

$$\sum_{i=1}^{N} i p_i = I \tag{19}$$

De I aan het eind van de formule staat voor de gemiddelde waarde, de i staat voor de zijdes met bijbehorende kansen p_i. Een simpel voorbeeld hiervan is bijvoorbeeld de perfecte dobbelsteen, namelijk: $\frac{1}{6}+\frac{2}{6}+\frac{3}{6}+\frac{4}{6}+\frac{5}{6}+1 = I = 3,5$.

Om de overige voorwaarden te maken wordt er telkens een i toegevoegd in de som, zodat de zesde voorwaarde er als volgt uit ziet:

$$\sum_{i=1}^{N} i^5 p_i = I_5 \tag{20}$$

waarin I_5 bij een normale dobbelsteen de waarde $\frac{1^5}{6} + \frac{2^5}{6} + \frac{3^5}{6} + \frac{4^5}{6} + \frac{5^5}{6} + \frac{6^5}{6} = 2033,5$ heeft.

4.3 Analytische oplossing

4.3.1 Eén beperkende voorwaarde

Bij de eerste beperkende voorwaarde wordt er alleen gewerkt met de informatie functie en de wetenschap dat de som van alle kansen gelijk aan 1 is. Zoals al vermeld is, ziet die voorwaarde er uit als $\sum_{i=1}^{N} p_i = 1$ en deze zal dan bij de Lagrangiaan toegevoegd worden. De Lagrangiaan krijgt dan de vorm:

$$L(p_i, \lambda) = -\sum_{i=1}^{N} p_i \ln(p_i) - \lambda(\sum_{i=1}^{N} p_i - 1) \tag{21}$$

om vervolgens het maximum te berekenen dient men de Lagrangiaan partieel te differentiëren naar p_i en λ en deze vervolgens gelijk aan 0 te stellen. De vergelijkingen zijn dan:

$$\frac{\partial L(p_i, \lambda)}{\partial p_i} = -\ln(p_i) - 1 - \lambda = 0 \tag{22}$$

$$\frac{\partial L(p_i, \lambda)}{\partial \lambda} = \sum_{i=1}^{N} p_i - 1 = 0 \tag{23}$$

uit vergelijking (22) volgt dan dat p_i eruit ziet als een e macht:

$$-ln(p_i) = 1 + \lambda \quad \rightarrow \quad p_i = e^{-(1+\lambda)} \tag{24}$$

wat hierbij opvalt is dat er geen i afhankelijkheid in de som zit, dus krijgen we $p_i = \text{constant} = p$ en deze p_i vullen we vervolgens in (23) en doordat we kunnen schrijven dat $\sum_{i=1}^{N} p = Np$ volgt hieruit:

$$p = \frac{1}{N} = p_i \tag{25}$$

De conclusie bij één beperkende voorwaarde is dus dat de optimale keuze van de p_i gelijk is aan $\frac{1}{N}$ en bij een dobbelsteen dus gelijk aan $\frac{1}{6}$, omdat een dobbelsteen zes zijden heeft. Dit is echter geen verrassend resultaat, aangezien de informatie maximaal is bij gelijke kansen en de voorwaarde heeft er voor gezorgd dat de som van alle kansen gelijk is aan 1.

4.3.2 Twee beperkende voorwaarden

De tweede beperkende voorwaarde zorgt daarentegen voor meer problemen. Het begint weer met het opstellen van de Lagrangiaan en partieel differentiëren naar p_i, λ_1 en λ_2:

$$L(p_i, \lambda_1, \lambda_2) = -\sum_{i=1}^{N} p_i \ln(p_i) - \lambda_1(\sum_{i=1}^{N} p_i - 1) - \lambda_2(\sum_{i=1}^{N} ip_i - I) \tag{26}$$

$$\frac{\partial L(p_i, \lambda)}{\partial p_i} = -\ln(p_i) - 1 - \lambda_1 - \lambda_2 i = 0 \tag{27}$$

$$\frac{\partial L(p_i, \lambda)}{\partial \lambda_1} = \sum_{i=1}^{N} p_i - 1 = 0 \tag{28}$$

$$\frac{\partial L(p_i, \lambda)}{\partial \lambda_2} = \sum_{i=1}^{N} ip_i - I = 0 \tag{29}$$

In dit geval kunnen we een algemene oplossing van p_i opschrijven aan de hand van (27)

$$-ln(p_i) = 1 + \lambda_1 + \lambda_2 i \quad \rightarrow \quad p_i = e^{-(1+\lambda_1+\lambda_2 i)} \quad \rightarrow \quad p_i = \alpha e^{-\beta i} \tag{30}$$

De laatste formule hier van p_i is de algemene vorm van hoe deze eruit ziet. Nu kunnen α en β berekend worden uit de vergelijkingen (28) en (29).

$$\sum_{i=1}^{N} \alpha e^{-\beta i} = 1 \quad \text{en} \quad \sum_{i=1}^{N} \alpha e^{-\beta i} i = I \tag{31}$$

Deze vergelijkingen zijn op te lossen doordat deze sommen uitgeschreven kunnen worden als volgt:

$$\sum_{k=1}^{N} aq^{k-1} = \frac{a(q^N - 1)}{q - 1} \tag{32}$$

$$\sum_{k=0}^{N-1} (a + kr)q^k = \frac{a - [a + (N-1)r]q^N}{1 - q} + \frac{rq(1 - q^{N-1})}{(1-q)^2} \tag{33}$$

Hieruit volgt vervolgens dat α uit te drukken is in β

$$\alpha = \frac{e^{-\beta} - 1}{e^{-\beta(N+1)} - e^{-\beta}} \tag{34}$$

Door deze functie in het resultaat van (33) te stoppen krijg je een functie afhankelijk van β, N en I.

$$I = \frac{(-1 + e^{-\beta})(e^{-\beta}(1 - e^{-\beta N}) + e^{-\beta(N+1)}(-1 + e^{-\beta})N)}{(1 - e^{-\beta})^2(-e^{-\beta} + e^{-\beta(N+1)})} \tag{35}$$

Deze formule kan vervolgens vereenvoudigd worden tot de vergelijking:

$$I = \frac{1}{1 - e^{-\beta}} + N + \frac{N}{-1 + e^{-\beta N}} \tag{36}$$

Deze formule lijkt vrij simpel, maar is echter een groot probleem. We weten dat $N = 6$ voor een dobbelsteen, dus dan komt er in deze functie een zesde macht in voor. Daardoor blijkt dat e^{β} niet in één keer uit te rekenen is en wordt er gebruik gemaakt van $e^{\beta} = 1 + \beta + \frac{\beta^2}{2}$ $(-1 < \beta < 1)$ en vanaf de derde term wordt het weg gelaten, anders is het berekenen alsnog erg lastig. Hierdoor komt (36) er als volgt uit te zien:

$$I = \frac{1}{(\beta + \frac{\beta^2}{2})} + 6 + \frac{-1}{(\beta + \frac{6\beta^2}{2})} \tag{37}$$

Deze formule werkt alleen als $I > 3,5$ dus wordt er ook nog een formule opgesteld voor $I < 3,5$. Wat opvalt is dat als $\beta = 0$ ingevuld wordt dat er dan een probleem ontstaat, maar we weten dat $e^{-\beta}$ dan 1 wordt en dus alle p_i gelijk zijn aan α en dus alle kansen gelijk zijn dus moet het gemiddelde dan wel $3,5$ zijn. Voor $I < 3,5$ hebben we echter:

$$I = \frac{1}{1 - \frac{1}{(1 - \beta + \frac{\beta^2}{2})}} + 6 + \frac{6}{-1 + \frac{1}{1 - 6\beta + \frac{(6\beta)^2}{2}}} \tag{38}$$

Vervolgens kan hieruit β opgelost worden voor I > 3,5, I = 3,5 en I < 3,5. Daarna worden deze waarden ingevuld in (34) en dan kunnen de waarden voor α en β ingevuld worden in (30).

$$I > 3,5: \quad \beta = \frac{42 - 7I - \sqrt{5}\sqrt{252 - 72I + 5I^2}}{6I - 32} \tag{39}$$

$$I = 3,5: \quad \beta = 0 \tag{40}$$

$$I < 3,5: \quad \beta = \frac{7 - 7I + \sqrt{5}\sqrt{-7 + 2I + 5I^2}}{6I - 6} \tag{41}$$

Deze formules kloppen niet helemaal vanwege de vereenvoudiging die gemaakt wordt, maar een betere oplossing lijkt niet mogelijk te zijn in dit geval. De derde beperkende voorwaarde lijkt vooralsnog niet oplosbaar te zijn.

Op bijlage 2 zijn de analytische waarden van I (I_{an})vergeleken met de ingevoerde I. Hieruit blijkt dat de analytische oplossing niet helemaal een goed antwoord geeft. De aanname die gemaakt is, geldt als $-1 < \beta < 1$ en zoals uit de tabel blijkt zitten de waarden van β ook tussen -1 en 1. Er is dus bij het oplossen iets verkeerd gegaan. Om na te gaan of de derde term, die door de vereenvoudiging niet is meegenomen, een significant verschil had gemaakt staat deze in de laatste kolom van de tabel. Deze kolom geeft de fout aan die je zou verwachten, maar kennelijk ligt het niet alleen aan de benadering die gedaan is, maar is er nog iets fout gegaan.

Wat wel klopt is dat de som van alle kansen 1 is in alle gevallen, maar door afronding in de tabel kan het zo zijn dat het optellen van alle p_i niet 1 geeft. Ook zien we dat het verschil tussen I en I_{an} steeds groter wordt naarmate het gemiddelde meer afwijkt van de eerlijke dobbelsteen, dus met een correctieterm is I_{an} aan te passen waardoor deze wel overeenkomt met I.

5 Numerieke berekening

5.1 Toepassingen

Voor verschillende doeleinden wordt gebruik gemaakt van computerprogramma's. Ten eerste moeten we een aantal 'worpen' kunnen genereren met vooraf bepaalde kansen (event generator). Ten tweede moeten we de parameters die in onze voorwaarden worden gebruikt uit een reeks worpen kunnen halen en tot slot moeten we natuurlijk op grond van de parameters kansen kunnen uitrekenen.

Onze programma's zijn geschreven in C. Voor het maken van plots wordt gebruik gemaakt van GNUplot en de `gplot`-library van Maurits Wijzenbeek en Eric Hennes.

5.2 Het genereren van worpen

Voor het genereren van worpen hebben wij een computerprogramma geschreven dat als input vraagt om de (relatieve) kans van elke zijde, het aantal te genereren worpen en de bestandsnaam voor de output. Van de gegeven kansen wordt een array met cumulatieve kansen gemaakt en voor elke worp wordt een willekeurig getal tussen 0 en 1 gekozen. Vervolgens wordt per zijde het willekeurige getal vergeleken met de cumulatieve kans voor die zijde. De eerste zijde die voldoet, is de geworpen zijde.

De willekeurige getallen komen uit `/dev/random`, de standaard pseudo-random number generator (PRNG) van UNIX-systemen. Op het gebruikte platform maakt deze gebruik van het Yarrow-algoritme (Kelsey et al., 1999) [3]. Dit geeft voor ons voldoende willekeurige getallen.

Na het genereren van de worpen geeft het programma de n^{e}-machts gemiddelden I_1 tot en met I_5. Deze worden berekend tijdens het genereren van de worpen. De n^{e}-machts gemiddelden kunnen worden gebruikt bij het modelleren. Daarnaast hebben wij een programma geschreven

De broncode van het programma `gendice` is te vinden in bijlage 4.

5.3 Numerieke berekening van de kansen

In de numerieke berekening moet worden voldaan aan de zelfde Lagrangiaan met de zelfde algemene oplossingsvormen als in de analytische berekening. Nu wordt het maximum echter numeriek benaderd door iteratief de waarden van parameters aan te passen om de afwijking kleiner te maken dan een bepaalde grenswaarde.

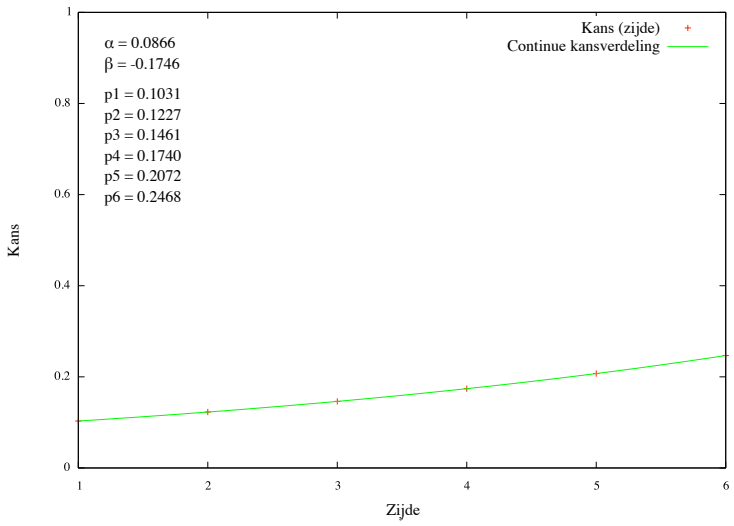
5.3.1 Twee beperkende voorwaarden

Het eenvoudigste geval is wederom de situatie met twee beperkende voorwaarden. De oplossingen voldoen hierbij aan formule (30), met de voorwaarden (28) en (29). We beginnen onze benadering van de oplossing met de 'eerlijke' dobbelsteen, $\alpha = \frac{1}{6}$ en $\beta = 0$. Daarna passen we stapsgewijs β aan in de juiste richting volgens:

$$\beta_{k+1} = \beta_k + \frac{I_k}{I} \tag{42}$$

Hierbij is β_k de tweede parameter uit de algemene oplossing voor iteratie k, I_k is de gemiddelde worp van de oplossing uit de huidige iteratie en I is het gemiddelde zoals ingevoerd. Na elke stap berekenen we α rechtsreeks uit de som van de kansen voor $\alpha = 1$.

Voor de meeste gekozen waarden van I geeft het programma binnen enkele tientallen iteraties een oplossing waarvoor I minder dan 10^{-10} af ten opzichte van de gekozen waarde. Een tabel met oplossingen voor $2,5 < I < 4,5$ is te vinden in bijlage 3.



Figuur 6: Grafiek van de meest waarschijnlijke kansverdeling als alleen bekend is dat de gemiddelde worp I 4 is.

5.3.2 Meer beperkende voorwaarden

Met meer dan twee beperkende voorwaarden wordt de algemene oplossing groter. Als we zes voorwaarden nemen, bestaande uit de triviale eis dat de

som van de kansen 1 is en vijf gegeven n^{e}-machts worpgemiddelden I_n, dan krijgen we de Lagrangiaan uit vergelijking (43), analoog aan vergelijking (26) voor twee voorwaarden.

$$L = -\sum_{i=1}^{N} p_i \ln(p_i) - \lambda_1(\sum_{i=1}^{N} p_i - 1) - \lambda_2(\sum_{i=1}^{N} ip_i - I_1) - \lambda_3(\sum_{i=1}^{N} i^2 p_i - I_2)$$
$$- \lambda_4(\sum_{i=1}^{N} i^3 p_i - I_3) - \lambda_5(\sum_{i=1}^{N} i^4 p_i - I_4) - \lambda_6(\sum_{i=1}^{N} i^5 p_i - I_5) \tag{43}$$

Om de meest waarschijnlijke oplossing met de gegeven parameters te bereiken zoeken we wederom het maximum van de Lagrangiaan. Dit blijkt van de vorm (44), een verdere generalisatie van vergelijking (30) voor meer voorwaarden. Met minder voorwaarden kunnen we één of meer van de parameters λ_2 t/m λ_6 0 stellen.

$$p_i = e^{1+\lambda_1+\lambda_2 i+\lambda_3 i^2+\lambda_4 i^3+\lambda_5 i^5+\lambda_6 i^6} \tag{44}$$

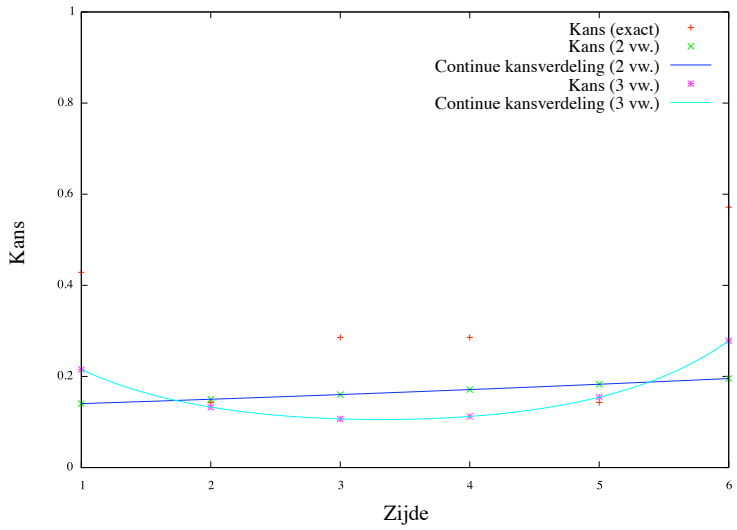
We berekenen dit numeriek door middel van de Gradiëntmethode. Deze werkt als volgt. We nemen beginwaarden voor de λ_i-parameters, in ons geval de oplossing voor een eerlijke dobbelsteen. Vervolgens berekenen we de afgeleide van de de Lagrangiaan naar alle λ_i-parameters. We trekken nu van elke parameter een parameter-afhankelijke constante maal de afgeleide af. Dit herhalen wij totdat de gemiddelden I_k minder dan een gekozen grensfactor afwijken van de ingevoerde waarden.

$$L(\vec{\lambda} - \gamma\nabla\vec{\lambda}) \geqslant L(\lambda_i) \tag{45}$$

De constante moet positief zijn, maar kleiner dan 1. In ons geval blijkt dat deze in de beginsituatie kleiner dan $0,00001$ moet zijn voor de drie lagere-orde-parameters en kleiner dan $0,000001$ voor de drie hogere-orde-parameters. Als hier niet aan is voldaan worden de parameters binnen enkele iteraties oneindig, omdat de correctieterm zo groot wordt dat de afgeleide in de volgende iteratie groter wordt.

Bij elke iteratie wordt elke afgeleide vergeleken met de dezelfde afgeleide in de vorige iteratie. Als het verschil minder is dan een zekere grenswaarde wordt de stapgrootte voor die parameter verkleind.

De broncode van het programma `numeric` is te te vinden in bijlage 6.



Figuur 7: Vergelijking van een twee- en drie-voorwaarden-model bij een sterk onregelmatige dobbelsteen.

6 Conclusie

In dit onderzoek zijn de mogelijke symmetriegroepen van zeszijdige dobbelstenen met eventuele onregelmatigheden in kaart gebracht.

Het blijkt mogelijk analytisch een model te maken met twee beperkende voorwaarden. Er is echter wel sprake van een grote fout voor sterk asymmetrische dobbelstenen (20% bij $I = 2,50$). Deze is niet te verklaren uit de benadering die is toegepast.

Een model met een derde beperkende voorwaarde blijkt analytisch niet te berekenen. Numeriek is dit wel mogelijk. Het programma kan rekenen met maximaal zes voorwaarden. Het blijkt echter dat in de meeste situaties met minder dan zes voorwaarden (vijf n^{e}-machts gemiddelden) de meest waarschijnlijke kansverdeling zeer sterk afwijkt van de daadwerkelijke kansen. Het gebruik van alleen n^{e}-machtsparameters als beperkende voorwaarden is dus vermoedelijk niet de beste werkwijze om een model met zo min mogelijk parameters te maken.

Daarnaast blijkt de kwaliteit van het model ongeacht het aantal parameters zeer sterk af te hangen van de precisie waarmee de parameters berekend worden. Voor een model met twee beperkende voorwaarden bleek dit geen probleem te vormen. Echter, voor meerdere voorwaarden werd het moeilijker een goed model te produceren.

7 Discussie

Om het gedrag van dobbelstenen beter te modelleren moeten geometrische eigenschappen in het model worden meegenomen. Dit kan in de vorm van extra beperkende voorwaarden. Vervolgens zou dan een vergelijkend onderzoek kunnen worden gedaan naar de complexiteit van diverse modellen door een extra term mee te nemen in de Lagrangiaan die het aantal parameters meeneemt. Zodoende kan een model met een kleiner aantal parameters een grotere entropie worden toegekend, waarmee het als een beter model wordt gekwalificeerd. Hierbij zou bijvoorbeeld gebruik gemaakt kunnen worden van de methode 'minimum description length' (J. Rissanen, 1978) [4].

Een vervolgonderzoek zou bovendien ook modellen kunnen maken van dobbelstenen van andere vormen, zoals de tetraëder, de octaëder, de dodecaëder en de icosaëder.

Voor het maken van betere modellen is in elk geval verder onderzoek en meer tijd nodig.

Referenties

[1] C.E. Shannon: *A Mathematical Theory of Communication*, The Bell System Technical Journal (Vol. 27 1948)

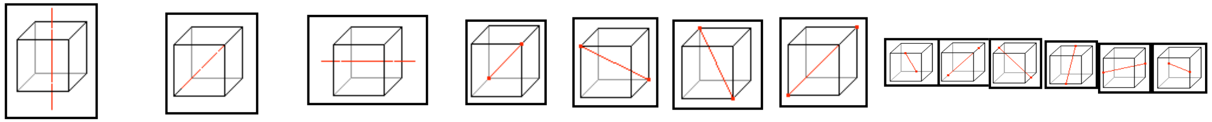
[2] Alan D. Thomas en G. V. Wood: *Group Tables*, Shiva Mathematics Series, Shiva Publishers Ltd., 1980

[3] J. Kelsey, B. Schneier en N. Ferguson: *Yarrow-160: Notes on the Design and Analysis of the Yarrow Cryptographic Pseudorandom Number Generator*, Sixth Annual Workshop on Selected Areas in Cryptography, Springer Verlag, augustus 1999

[4] J. Rissanen, *'Modelling By Shortest Data Description'* , Automatica (Vol. 14 1978)

Wij willen Prof. F.A. Bais graag hartelijk danken voor zijn steun, begeleiding en aandacht bij dit project.

Bijlage 1: Elementen van de ondergroepen



bovenkant	1	1	1	1	4	6	3	2	6	5	2	3	2	4	4	5	3	5	2	3	4	5	6	6
voorkant	2	3	5	4	2	2	2	6	5	1	3	1	4	1	6	3	6	4	1	5	5	6	4	3
rechterkant	3	5	4	2	1	4	6	3	3	3	1	2	6	5	2	6	5	1	4	1	6	4	5	2
linkerkant	4	2	3	5	6	3	1	4	4	4	6	5	1	2	5	1	2	6	3	6	1	3	2	5
achterkant	5	4	2	3	5	5	5	1	2	6	4	6	3	6	1	4	1	3	6	2	2	1	3	4
onderkant	6	6	6	6	3	1	4	5	1	2	5	4	5	3	3	2	4	2	5	4	3	2	1	1
O(24)	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
T(12)	x		x			x			x		x	x	x	x	x	x	x	x						
D4(8)	x	x	x	x		x			x														x	x
D4(8)	x		x		x	x	x		x											x	x			
D4(8)	x		x			x		x	x	x									x			x		
D3(6)	x										x	x									x	x	x	
D3(6)	x												x	x						x		x		x
D3(6)	x														x	x			x	x			x	
D3(6)	x																x	x	x		x			x
C2^3(4)	x		x																				x	x
C2^3(4)	x					x														x	x			
C2^3(4)	x								x										x			x		
C2^3(4)	x		x			x			x															
C4(4)	x	x	x	x																				
C4(4)	x				x	x	x																	
C4(4)	x							x	x	x														
C3(3)	x										x	x												
C3(3)	x												x	x										
C3(3)	x														x	x								
C3(3)	x																x	x						
C2(2)	x																		x					
C2(2)	x																			x				
C2(2)	x																				x			
C2(2)	x																					x		
C2(2)	x																						x	
C2(2)	x																							x
C2(2)	x		x																					
C2(2)	x					x																		
C2(2)	x								x															

Bijlage 2: Tabel analytische waarden

I	β	α	p_1	p_2	p_3	p_4	p_5	p_6	$\sum p_i$	I_{an}	ΔI	$\frac{1}{3}\frac{b^3}{b^2}$
2.50	0.1770	0.2960	0.248	0.208	0.174	0.146	0.122	0.102	1	2.9933	-0.4933	0.059
2.55	0.1636	0.2843	0.241	0.205	0.174	0.148	0.125	0.107	1	3.0304	-0.4804	0.055
2.60	0.1509	0.2735	0.235	0.202	0.174	0.150	0.129	0.111	1	3.0658	-0.4658	0.050
2.65	0.1389	0.2635	0.229	0.200	0.174	0.151	0.132	0.115	1	3.0996	-0.4496	0.046
2.70	0.1275	0.2543	0.224	0.197	0.173	0.153	0.134	0.118	1	3.1318	-0.4318	0.042
2.75	0.1166	0.2458	0.219	0.195	0.173	0.154	0.137	0.122	1	3.1627	-0.4127	0.039
2.80	0.1063	0.2378	0.214	0.192	0.173	0.155	0.140	0.126	1	3.1922	-0.3922	0.035
2.85	0.0964	0.2304	0.209	0.190	0.173	0.157	0.142	0.129	1	3.2204	-0.3704	0.032
2.90	0.0870	0.2235	0.205	0.188	0.172	0.158	0.145	0.133	1	3.2475	-0.3475	0.029
2.95	0.0780	0.2170	0.201	0.186	0.172	0.159	0.147	0.136	1	3.2734	-0.3234	0.026
3,00	0.0694	0.2110	0.197	0.184	0.171	0.160	0.149	0.139	1	3.2983	-0.2983	0.023
3.05	0.0611	0.2053	0.193	0.182	0.171	0.161	0.151	0.142	1	3.3222	-0.2722	0.020
3.10	0.0532	0.2000	0.190	0.180	0.170	0.162	0.153	0.145	1	3.3451	-0.2451	0.018
3.15	0.0456	0.1949	0.186	0.178	0.170	0.162	0.155	0.148	1	3.3671	-0.2171	0.015
3.20	0.0383	0.1902	0.183	0.176	0.170	0.163	0.157	0.151	1	3.3883	-0.1883	0.013
3.25	0.0313	0.1857	0.180	0.174	0.169	0.164	0.159	0.154	1	3.4087	-0.1587	0.010
3.30	0.0246	0.1815	0.177	0.173	0.169	0.164	0.160	0.157	1	3.4283	-0.1283	0.008
3.35	0.0181	0.1775	0.174	0.171	0.168	0.165	0.162	0.159	1	3.4472	-0.0972	0.006
3.40	0.0118	0.1737	0.172	0.170	0.168	0.166	0.164	0.162	1	3.4655	-0.0655	0.004
3.45	0.0058	0.1701	0.169	0.168	0.167	0.166	0.165	0.164	1	3.4830	-0.0330	0.002
3.50	0	0.167	0.167	0.167	0.167	0.167	0.167	0.167	1	3.5	0	-
3.55	-0.0058	0.1633	0.164	0.165	0.166	0.167	0.168	0.169	1	3.5170	0.0330	-0.002
3.60	-0.0118	0.1599	0.162	0.164	0.166	0.168	0.170	0.172	1	3.5345	0.0655	-0.004
3.65	-0.0181	0.1564	0.159	0.162	0.165	0.168	0.171	0.174	1	3.5528	0.0972	-0.006
3.70	-0.0246	0.1528	0.157	0.160	0.164	0.169	0.173	0.177	1	3.5717	0.1283	-0.008
3.75	-0.0313	0.1491	0.154	0.159	0.164	0.169	0.174	0.180	1	3.5913	0.1587	-0.010
3.80	-0.0383	0.1454	0.151	0.157	0.163	0.170	0.176	0.183	1	3.6117	0.1883	-0.013
3.85	-0.0456	0.1416	0.148	0.155	0.162	0.170	0.178	0.186	1	3.6329	0.2171	-0.015
3.90	-0.0532	0.1378	0.145	0.153	0.162	0.170	0.180	0.190	1	3.6549	0.2450	-0.018
3.95	-0.0611	0.1338	0.142	0.151	0.161	0.171	0.182	0.193	1	3.6778	0.2721	-0.020
4,00	-0.0694	0.1298	0.139	0.149	0.160	0.171	0.184	0.197	1	3.7017	0.29828	-0.023
4.05	-0.0780	0.1257	0.136	0.147	0.159	0.172	0.186	0.201	1	3.7266	0.3234	-0.026
4.10	-0.0870	0.12158	0.133	0.145	0.159	0.172	0.188	0.206	1	3.7525	0.3475	-0.029
4.15	-0.0964	0.1173	0.129	0.142	0.157	0.173	0.190	0.209	1	3.7796	0.3704	-0.032
4.20	-0.1063	0.1130	0.126	0.140	0.155	0.173	0.192	0.214	1	3.8078	0.3922	-0.035
4.25	-0.1166	0.1086	0.122	0.137	0.154	0.173	0.195	0.219	1	3.8373	0.4127	-0.039
4.30	-0.1275	0.1042	0.118	0.134	0.153	0.173	0.197	0.224	1	3.8682	0.43184	-0.042
4.35	-0.13892	0.0997	0.115	0.132	0.152	0.174	0.200	0.229	1	3.9004	0.4496	-0.046
4.40	-0.1509	0.0951	0.111	0.129	0.150	0.174	0.202	0.235	1	3.9342	0.4658	-0.050
4.45	-0.1636	0.0904	0.107	0.125	0.148	0.174	0.205	0.241	1	3.9696	0.4804	-0.055
4.50	-0.1770	0.0857	0.102	0.122	0.146	0.174	0.208	0.248	1	4.0067	0.4933	-0.059

Bijlage 3: Tabel numerieke waarden (twee voorwaarden)

I	β	α	p_1	p_2	p_3	p_4	p_5	p_6	$\sum p_i$	I_{num}
2.50	0.371049	0.503607	0.34749	0.23977	0.16545	0.11416	0.07877	0.05435	1	2.500000
2.55	0.349568	0.477021	0.33630	0.23709	0.16714	0.11784	0.08307	0.05857	1	2.550000
2.60	0.328605	0.451955	0.32537	0.23425	0.16864	0.12141	0.08741	0.06293	1	2.600000
2.65	0.308113	0.428286	0.31472	0.23127	0.16994	0.12488	0.09176	0.06743	1	2.650000
2.70	0.288048	0.405905	0.30432	0.22815	0.17105	0.12824	0.09615	0.07208	1	2.700000
2.75	0.268371	0.384714	0.29416	0.22492	0.17198	0.13150	0.10055	0.07688	1	2.750000
2.80	0.249045	0.364627	0.28424	0.22158	0.17273	0.13465	0.10497	0.08183	1	2.800000
2.85	0.230039	0.345565	0.27455	0.21813	0.17331	0.13769	0.10940	0.08692	1	2.850000
2.90	0.211319	0.327460	0.26508	0.21459	0.17371	0.14062	0.11384	0.09215	1	2.900000
2.95	0.192858	0.310247	0.25583	0.21096	0.17395	0.14344	0.11828	0.09754	1	2.950000
3.00	0.174629	0.293870	0.24678	0.20724	0.17403	0.14615	0.12273	0.10307	1	3.000000
3.05	0.156605	0.278277	0.23794	0.20345	0.17396	0.14874	0.12718	0.10874	1	3.050000
3.10	0.138764	0.263421	0.22929	0.19958	0.17372	0.15121	0.13162	0.11457	1	3.100000
3.15	0.121081	0.249260	0.22083	0.19565	0.17334	0.15357	0.13606	0.12054	1	3.150000
3.20	0.103535	0.235754	0.21257	0.19166	0.17281	0.15581	0.14049	0.12667	1	3.200000
3.25	0.086105	0.222867	0.20448	0.18761	0.17213	0.15793	0.14490	0.13295	1	3.250000
3.30	0.068771	0.210568	0.19657	0.18351	0.17131	0.15993	0.14930	0.13938	1	3.300000
3.35	0.051513	0.198825	0.18884	0.17936	0.17036	0.16180	0.15368	0.14596	1	3.350000
3.40	0.034311	0.187610	0.18128	0.17517	0.16926	0.16355	0.15803	0.15270	1	3.400000
3.45	0.017146	0.176899	0.17389	0.17094	0.16803	0.16517	0.16237	0.15961	1	3.450000
3.50	0.000000	0.166667	0.16667	0.16667	0.16667	0.16667	0.16667	0.16667	1	3.500000
3.55	-0.017146	0.156892	0.15961	0.16237	0.16517	0.16803	0.17094	0.17389	1	3.550000
3.60	-0.034311	0.147554	0.15270	0.15803	0.16355	0.16926	0.17517	0.18128	1	3.600000
3.65	-0.051513	0.138634	0.14596	0.15368	0.16180	0.17036	0.17936	0.18884	1	3.650000
3.70	-0.068771	0.130114	0.13938	0.14930	0.15993	0.17131	0.18351	0.19657	1	3.700000
3.75	-0.086105	0.121978	0.13295	0.14490	0.15793	0.17213	0.18761	0.20448	1	3.750000
3.80	-0.103535	0.114210	0.12667	0.14049	0.15581	0.17281	0.19166	0.21257	1	3.800000
3.85	-0.121081	0.106797	0.12054	0.13606	0.15357	0.17334	0.19565	0.22083	1	3.850000
3.90	-0.138764	0.099724	0.11457	0.13162	0.15121	0.17372	0.19958	0.22929	1	3.900000
3.95	-0.156605	0.092979	0.10874	0.12718	0.14874	0.17396	0.20345	0.23794	1	3.950000
4.00	-0.174629	0.086551	0.10307	0.12273	0.14615	0.17403	0.20724	0.24678	1	4.000000
4.05	-0.192858	0.080428	0.09754	0.11828	0.14344	0.17395	0.21096	0.25583	1	4.050000
4.10	-0.211319	0.074599	0.09215	0.11384	0.14062	0.17371	0.21459	0.26508	1	4.100000
4.15	-0.230039	0.069056	0.08692	0.10940	0.13769	0.17331	0.21813	0.27455	1	4.150000
4.20	-0.249045	0.063787	0.08183	0.10497	0.13465	0.17273	0.22158	0.28424	1	4.200000
4.25	-0.268371	0.058786	0.07688	0.10055	0.13150	0.17198	0.22492	0.29416	1	4.250000
4.30	-0.288048	0.054043	0.07208	0.09615	0.12824	0.17105	0.22815	0.30432	1	4.300000
4.35	-0.308113	0.049551	0.06743	0.09176	0.12488	0.16994	0.23127	0.31472	1	4.350000
4.40	-0.328605	0.045302	0.06293	0.08741	0.12141	0.16864	0.23425	0.32537	1	4.400000
4.45	-0.349568	0.041288	0.05857	0.08307	0.11784	0.16714	0.23709	0.33630	1	4.450000

Bijlage 4: Broncode gendice.c

```
/*  gendice.c -- Generate dice throws
 *
 *  (c) 2007 Reinier Jonker, Menno de Bell and Andries van der Leden
 */
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>

/* Prototypes */
double rnddouble ();

/* Globals */
double maxint;                          /* Largest possible integer */
FILE *devrnd;                           /* Pointer to /dev/random */

/* Main function */
int main(int argc, char *argv[]) {
    int i, j;                           /* Iterators */
    int nthrows;                        /* The number of throws */
    double p[6];                        /* Probability for each number */
    double pcumlt[6];
    double q;                           /* Chosen random value; 0. < q < 1. */
    double avg[6];                      /* nth average */

    int nthrown[6];                  /* The numer of times each number was
                                                thrown */

    /* Initialise globals */
    maxint = pow(2, 8 * sizeof(unsigned int)) - 1;
    devrnd = fopen("/dev/random", "r");

    FILE *outfile;                      /* Output file */

    /* Initialise with default parameters */
    p[0] = 1./6.;
    p[1] = 1./6.;
    p[2] = 1./6.;
    p[3] = 1./6.;
```

```
p[4] = 1./6.;
p[5] = 1./6.;

/* Normalise probabilities */
q = 0;
for(i = 0; i < 6; i++) {
    printf("Relative probability of side %d: ", i + 1);
    scanf("%lf", &p[i]);
    q = q + p[i];
}
for(i = 0; i < 6; i++) {
    p[i] = p[i] / q;
}

/* Read the number of throws from the first argument or
    substitute the default */
if(argc > 1) {
    if(sscanf(argv[1], "%d", &nthrows) != 1) {
        nthrows = 1000;
    }
}
else {
    nthrows = 1000;
}

/* Make an array with cumulative probabilities and
    initialise statistics */
pcumlt[0] = p[0];
nthrown[0] = 0;
avg[0] = 0;
for(i = 1; i < 6; i++) {
    pcumlt[i] = pcumlt[i - 1] + p[i];
    nthrown[i] = 0;
    avg[0] = 0;
}

/* Open output file, if available use the second argument
        as output file path */
if(argc > 2) {
    outfile = fopen(argv[2], "a");
}
```

```
    else {
        outfile = fopen("out.txt", "a");
    }

    /* Return an error if the file cannot be written */
    if(outfile == NULL) {
        printf("Error\n");
        return 1;
    }

    /* Generate a list of throws */
    for(i = 0; i < nthrows; i++) {
        q = rnddouble();
        for(j = 0; j < 6; j++) {
            if(q < pcumlt[j]) {
                nthrown[j]++;
                fprintf(outfile, "%d\n", j + 1);
                break;
            }
        }
    }

    /* Close output file */
    fclose(outfile);

    /* Show statistics */
    for(i = 0; i < 6; i++) {
        printf("%d: %d/%d\n", i + 1, nthrown[i], nthrows);
        for(j = 0; j < 6; j++) {
            avg[j] = avg[j] + (((pow(i + 1, j + 1) *
                                 (double) nthrown[i])) / (double) nthrows);
        }
    }
    for(i = 0; i < 6; i++) {
        printf("Average of n^%d: %f\n", i + 1, avg[i]);
    }

    /* Close /dev/random file pointer */
    fclose(devrnd);
    return 0;
}
```

```
/* Use an integer from /dev/random to generate a random
    double between 0. and 1. */

double rnddouble () {
    unsigned int r;
    fread(&r, sizeof(unsigned int), 1, devrnd);
    return(r / maxint);
}
```

Bijlage 5: Broncode numeric2.c

```
/* Calculate a two-parameter numeric model for six-sided dice
 *
 * (c) 2007 Reinier Jonker, Menno de Bell and Andries van der Leden
 */

/* Standard library and gplot includes */
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "gplot.h"

/* Prototypes */
void probabilities(double, double);
double csump(double, double);
double cavg(double, double);

/* Main function */
int main(int argc, char *argv[]) {
   int result;
   double avg;

   printf("Gemiddelde worp: ");
   while(scanf("%lg", &avg)) {
      if(avg >= 1. && avg <= 6.) {
         probabilities(0.000000000001, avg);
      }
      printf("Gemiddelde worp: ");
   }
}

void probabilities(double threshold, double avg) {
   int i;

   double alpha = 1./6.;
   double beta = 0.;
   double d1;                 /* First order approximaition */

   double pthr = 1. + threshold; /* Positive and negative limits */
```

```
double nthr = 1. - threshold;

int limit = 5000000;       /* Maximum number of iterations */

/* Plot variables */
float side[nsides];
float pres[nsides];
char *plotfunc;

/* Numerical calculation */
for(i = 0; i < limit; i++) {
   d1 = cavg(alpha, beta) / avg;
   if(d1 > pthr || d1 < nthr) {
      beta = beta + d1 - 1.;
   }
   else {
      break;
   }

   /* Calculate alhpa explicitly */
   alpha = 1. / csump(1., beta);
}

/* Prepare plot data and print results */
printf("%d iteraties\nalpha = %.10f; beta = %.10f\n", i, alpha, beta);
for(i = 0; i < nsides; i++) {
   side[i] = i + 1;
   pres[i] = alpha * exp(-(i + 1) * beta);

   printf("p(%d) = %.10f\n", i + 1, pres[i]);
}

printf("\nGemiddelde: %.10f; Som: %.10f\n", cavg(alpha, beta), csump(alpha, beta))

/* Make plot */
asprintf(&plotfunc, "%.10f * exp(%.10f * x)", alpha, - beta);
gpl_axis(X, 1., nsides, "Zijde");
gpl_axis(Y, 0., 1., "Kans");
gpl_style(POINTS);
gpl_data(nsides, side, pres, "Kans (zijde)");
gpl_style(LINES);
```

```
   gpl_func(plotfunc, "Continue kansverdeling");
   gpl_show();

   free(plotfunc);
}

/* Calculate the sum of the probabilities for given alpha and beta */
double csump(double alpha, double beta) {
   int i;
   double sum = 0.;

   for(i = 1; i <= nsides; i++) {
      sum = sum + exp( - i * beta);
   }

   return alpha * sum;
}

/* Calculate the average throw for given alpha and beta */
double cavg(double alpha, double beta) {
   int i;
   double avg = 0.;

   for(i = 1; i <= nsides; i++) {
      avg = avg + i * exp(- i * beta);
   }

   return alpha * avg;
}
```

Bijlage 6: Broncode numeric.c

```
/*  numeric.c -- Calculate a numeric model for six-sided dice
 *
 *  (c) 2007 Reinier Jonker, Menno de Bell and Andries van der Leden
 */
/* Standard library and gplot includes */
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "gplot.h"

/* Prototypes */
void probabilities(double avg[]);
double csump(double avg[]);
double cavg(int power, double avg[]);
double pi(int side, double lambda[]);
double lagr(double lambda[], double avg[]);
double dlagdp(int side, double lambda[]);
double dlagdl(int order, double lambda[], double avg[]);

/* Main function */
int main(int argc, char *argv[]) {
    int result;
    double avg[6];

    /* The first 'average' is actually the condition that the sum
        of properties is 1. */
    avg[0] = 1.;

    /* Ask for nth-power averages. Zeroes will be ignored. */
    printf("Worpgemiddelden: ");
    while(scanf("%lg %lg %lg %lg %lg", &avg[1], &avg[2], &avg[3],
            &avg[4], &avg[5]) == 5) {
        probabilities(avg);
        printf("Worpgemiddelden: ");
    }

    /* We always succeed in programme execution */
    return 0;
```

```
}

/* Calculate the probabilities given the nth-power averages */
void probabilities(double avg[]) {
    int i, j;                   /* Iterators */

    int limit = 5000000;        /* Maximum number of iterations */

    double step[6] = {0.00001, 0.00001, 0.00001, 0.000001, 0.000001, 0.000001};
                                    /* Initial step sizes */
    double delta[6];            /* Gradient-based increments for the
                                       lambda parameters */

    double lambda[6] = {0, 0, 0, 0, 0, 0};
                                    /* Initialise lambdas to 0 */
    double dL;                  /* Partial derivative of L in the current
                                       iteration */

    double hist[6] = {1., 1., 1., 1., 1., 1.};
                                    /* Store partial derivative for the next
                                       iteration */

    /* Plot variables */
    float side[6];
    float pres[6];
    char *plotlabel;

    /* Initial condition: p(i) = 1/N */
    lambda[0] = -log(1./6.);

    /* Numerical calculation */
    for(i = 0; i < limit; i++) {

        /* Calculate gradient */
        for(j = 0; j < 6; j++) {
            dL = dlagdl(j, lambda, avg);

            /* Halve step size if the sign of the partial derivative
                changes */
            if((dL * hist[j]) < 0) {
                step[j] = 0.5 * step[j];
```

```
        }
        hist[j] = dL;
        delta[j] = step[j] * dL;
    }
    /* Adjust lambdas */
    for(j = 0; j < 6; j++) {
        lambda[j] = lambda[j] - delta[j];
    }
}

/* Prepare plot data and print results */
printf("%d iterations\nlambda1 = %G; lambda2 = %G; lambda3 = %G;
        lambda4 = %G; lambda5 = %G; lambda6 = %G\n", i, lambda[0],
        lambda[1], lambda[2], lambda[3], lambda[4], lambda[5]);
for(i = 0; i < 6; i++) {
    side[i] = i + 1;
    pres[i] = pi(i + 1, lambda);

    printf("p(%d) = %f\n", i + 1, pres[i]);
}

printf("\n<W>: %f; <W^2>: %f; <W^3>: %f; <W^4>: %f; <W^5>: %f; Som: %f\n",
         cavg(1, lambda), cavg(2, lambda), cavg(3, lambda),
         cavg(4, lambda), cavg(5, lambda), csump(lambda));

/* Make plot */
asprintf(&plotlabel, "lambda1 = %G\nlambda2 = %G\nlambda2 = %G\n
    lambda3 = %G\nlambda4 = %G\nlambda5 = %G\nlambda6 = %G",
    lambda[0], lambda[1], lambda[2], lambda[3], lambda[4], lambda[5]);

gpl_style(POINTS);
gpl_axis(X, 1., 6., "Zijde");
gpl_axis(Y, 0., 1., "Kans");
gpl_data(6, side, pres, "Kans (zijde)");
gpl_style(LINES);
gpl_command(plotlabel);
gpl_show();

/* Print final step sizes */
for(i = 0; i < 6; i++) {
    printf("step[%d] = %G\n", i, step[i]);
```

```
    }

}

/* Calculate the sum of the probabilities for given lambdas */
double csump(double lambda[]) {
    int i;
    double sum = 0.;

    for(i = 1; i < 7; i++) {
        sum = sum + pi(i, lambda);
    }

    return sum;
}

/* Calculate the average throw for given lambdas */
double cavg(int power, double lambda[]) {
    int i;
    double avg = 0.;

    for(i = 1; i < 7; i++) {
        avg = avg + pow(i, power) * pi(i, lambda);
    }

    return avg;
}

/* Current p(i) for given lambdas */
double pi(int side, double lambda[]) {
    int i;
    double arg = 0.;
    for(i = 0; i < 6; i++) {
        arg = arg - pow(side, i) * lambda[i];
    }
    return exp(arg);
}

/* The Lagrangian for given lambdas and averages */
double lagr(double lambda[], double avg[]) {
    int i, j;
```

```
    double p;
    double logp;
    double L = 0.;

    for(i = 1; i < 7; i++) {
        /* Calculate log(p) and p for this side */
        logp = 0;
        for(j = 0; j < 6; j++) {
            logp = logp - lambda[j] * pow(i, j);
        }
        p = exp(logp);
        L = L - p * logp;
        for(j = 0; j < 6; j++) {
            L = L - pow(i, j) * lambda[j] * p;
        }
    }

    /* Calculate the Lagrangian */
    for(j = 0; j < 6; j++) {
        L = L - lambda[j] * avg[j];
    }
    return L;
}

/* Differentiate the Lagrangian with respect to lambda(order) */
double dlagdl(int order, double lambda[], double avg[]) {
    int i;
    double res = 0.;
    for(i = 1; i < 7; i++) {
        res = res - pow(i, order) * pi(i, lambda);
    }
    return res + avg[order];
}

/* Differentiate the Lagrangian with respect to p(side).
    Not actually used. */
double dlagdp(int side, double lambda[]) {
    int i;
    double p;
    double logp = 0.;
    for(i = 0; i < 6; i++) {
```

```
        logp = logp - pow(side, i) * lambda[i];
    }
    return - logp - lambda[0] - lambda[1] * side;
}
```