Simulatie van kansexperimenten
met de TI-83 Plus

Koen Stulens
Limburgs Universitair Centrum
Diepenbeek, België

1. Inleiding

De **TI-83 (Plus)** is een uitstekende technologie om de oorspronkelijke ideeën over kansen te demonstreren en deze te linken aan de theoretische kansrekening.

Het werk *Ars Conjectandi* van Jacob Bernoulli was een mijlpaal in de ontwikkeling van de kansrekening. Het werd na zijn dood in 1713 gepubliceerd door zijn neef Nikolaus Bernoulli.

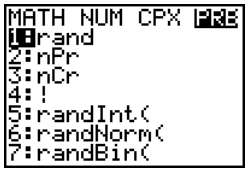
In dit werk was er voor het eerst sprake van de wet der grote aantallen : de relatieve frequentie van een gebeurtenis *benadert* de theoretische kans op die gebeurtenis als men het experiment een voldoende aantal keer herhaalt.

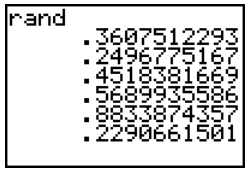
Het is dit principe dat zal gebruikt worden in de hierna volgende simulaties.

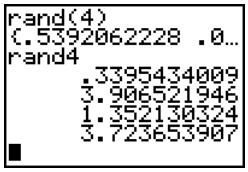
2. Toevalsgetallen

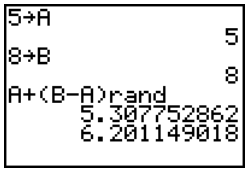
Het commando **rand** (**MATH<PRB> 1 : rand**) genereert de volgende toevalsgetallen :

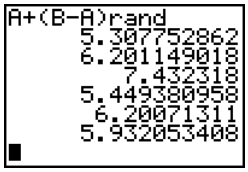
rand	→	een willekeurig getal x tussen 0 en 1 ($0 < x < 1$)
rand(4)	→	een lijst van 4 willekeurige getallen tussen 0 en 1
rand4	→	een willekeurig getal x tussen 0 en 4 ($0 < x < 4$)
A+(B-A)rand	→	een willekeurig getal x tussen A en B ($A < x < B$)







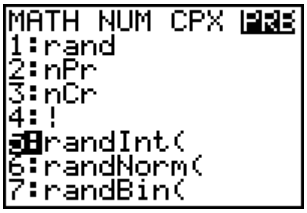


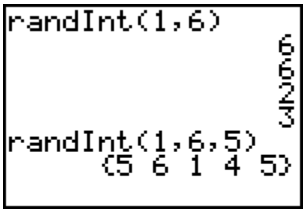


Het commando **rand** genereert toevalsgetallen vertrekkende van een startwaarde. Standaard staat deze startwaarde ingesteld op 0. Vertrekkende van eenzelfde startwaarde bekom je telkens dezelfde rij toevalsgetallen. Om een andere rij toevalsgetallen te bekomen kun je andere startwaarde toekennen aan **rand** met bv, het commando **144 → rand**.

Het commando **randInt** (**MATH<PRB> 5:randInt**) genereert de volgende toevalsgetallen :

randInt(1,6) → een willekeurig geheel getal tussen 1 en 6 (grenzen inbegrepen)
randInt(1,6,5) → een lijst van 5 willekeurige gehele getallen tussen 1 en 6



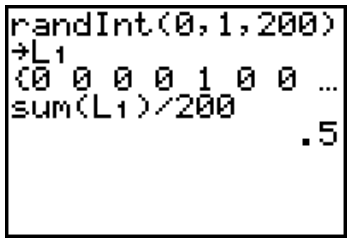


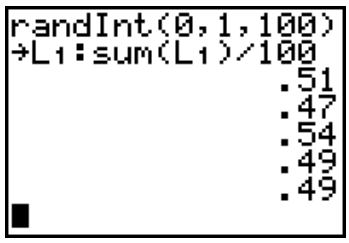
3. Het opwerpen van een munstuk

Om het opwerpen van een muntstuk te simuleren, coderen we kop met **1** en kruis met **Ø**.

a. Simuleer tweehonderd keer opwerpen van een muntstuk : **randInt(Ø,1,2ØØ) → L1**.

b. Bepaal het aantal keer kop en de relatieve frequentie van kop. De onderstaande schermafdrukken tonen dat de relatieve frequentie van de gebeurtenis kop ½ op de lange duur benadert.





Om het vorige resultaat visueel voor te stellen, definiëren we de volgende lijsten :

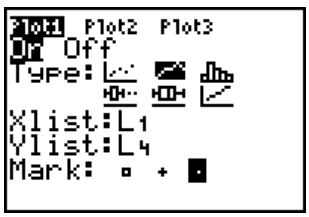
L1 = seq(X,X,1,2ØØ)

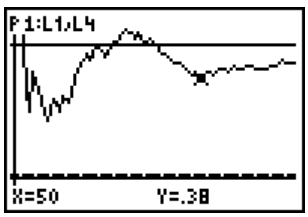
L2 = randInt(Ø,1,2ØØ)

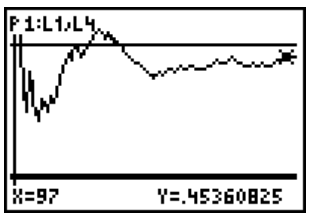
L3 = cumSum(L2)

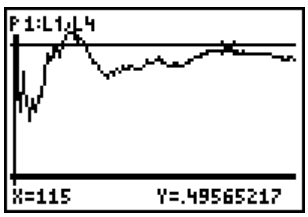
L4 = L3/L1

Y1 = 1/2







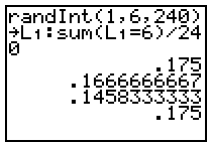


Bovenstaande statistische plots tonen dat meer worpen niet automatisch een betere benadering geven. Als we het munstuk blijven opwerpen zal de relatieve frequentie ½ zo dicht benaderen als we maar willen maar niemand kan ons zeggen hoeveel worpen we hiertoe moeten uitvoeren.

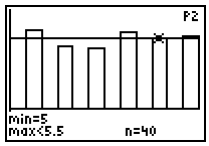
Klasopgave

Laat elke leerling 100 worpen simuleren met een muntstuk en bepaal de relatieve frequentie van het aantal keer munt van de hele klas. Laat iedereen lukraak een startwaarde kiezen voor de toevalsgenerator. Welk resultaat verwacht je ?

4. Het werpen van dobbelstenen



a. Het 240 keer werpen van een dobbelsteen kan je simuleren met **randInt(1,6,24Ø) → L1**.



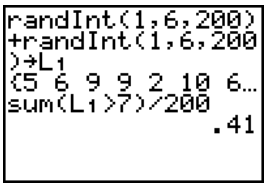
b. Bepaal de relatieve frequentie van de gebeurtenis zes ogen.

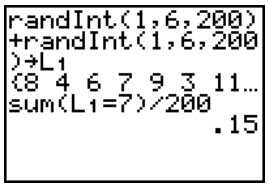
Het hiernaast afgebeelde histogram is opnieuw een visualisatie van Bernoulli's wet der grote aantallen.

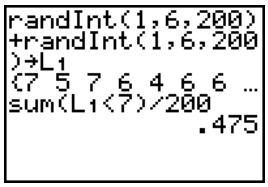
c. Het commando **randInt(1,6)+randInt(1,6)** simuleert het tellen van de ogen van een worp met twee dobbelstenen.

Dice 1	Dice 2	Dice 1	Dice 2	Dice 1	Dice 2	Dice 1	Dice 2	Dice 1	Dice 2	Dice 1	Dice 2
1	1	2	1	3	1	4	1	5	1	6	1
1	2	2	2	3	2	4	2	5	2	6	2
1	3	2	3	3	3	4	3	5	3	6	3
1	4	2	4	3	4	4	4	5	4	6	4
1	5	2	5	3	5	4	5	5	5	6	5
1	6	2	6	3	6	4	6	5	6	6	6

Voer aan de hand van het **TEST**-menu de onderstaande simulaties uit.







5. Opdracht

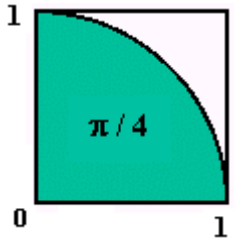
Bepaal d.m.v. simulatie een benadering voor het getal π = 3.14159265358979... .

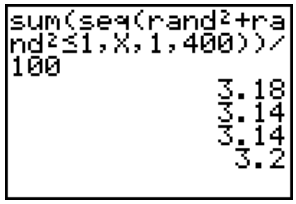
a. Voer met het commando **rand** een test uit die nagaat of een willekeurig punt, **(rand,rand)**, uit het interval $[0,1]\times[0,1]$ op een afstand kleiner dan of gelijk aan 1 van de oorsprong ligt.

b. Voer de test uit punt a. vierhonderd keer met het **seq**-commando.

c. Ga na hoeveel punten uit b. op een afstand kleiner of gelijk aan 1 van de oorsprong liggen.

d. Bepaal met het resultaat uit c. een benadering voor π.





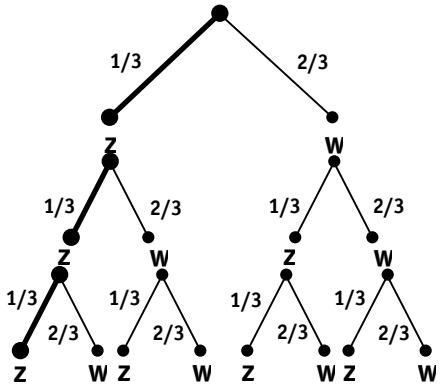
6. Trekkingen

Heel wat problemen uit de kansrekening kunnen herleid worden tot het trekken van ballen uit een urne. Er zijn twee soorten van trekkingen : met teruglegging en zonder teruglegging.

Veronderstel dat in een urne 10 witte en 5 zwarte ballen liggen. Stel de witte ballen voor met de getallen 1, 2, ..., 10 en de zwarte met 11, 12, ..., 15.

De vraag die we ons stellen is wat de kans is op drie zwarte ballen na drie trekkingen.

6.1 Trekking met teruglegging



$$P(3x\ zwart) \parallel \frac{1}{3}\frac{1}{3}\frac{1}{3} \approx 0.037$$

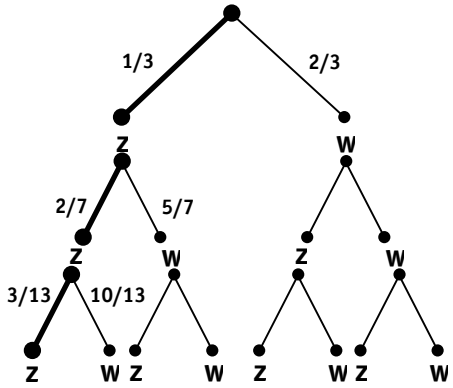
Het trekken van drie ballen kan in dit geval eenvoudig met **`randInt(1,15,3)`**.

`sum(randInt(1,15,3)>10)` test hoeveel ballen van de drie er zwart zijn. Het simuleren van de kans op drie zwarte ballen verloopt als volgt :

```
seq(sum(randInt(
1,15,3)>10),X,1,
100)→L1:sum(L1=3
)/100
                .03
                .03
                .05
```

```
                .04
                .05
                .03
                .04
                .05
                .03
                .05
```

6.2 Trekking zonder teruglegging



$$P(3x\ zwart) \parallel \frac{1}{3}\frac{2}{7}\frac{3}{13} \approx 0.022$$

Om een trekking zonder teruglegging te simuleren moeten we de vorige procedure verfijnen.

Het onderstaande korte programma simuleert een trekking zonder teruglegging van drie getallen uit 15 getallen.

```
ClearList L1,L2
Repeat sum(ΔList(L2)=Ø)=Ø
randInt(1,15,3)→L1
L1→L2
SortA(L2)
End
```

Het trekken van drie ballen (**randInt(1,15,3)→L1**) wordt herhaald tot de conditie van de **Repeat**-lus (**sum(ΔList(L2)=Ø)=Ø**) waar is. In dat geval zijn alle elementen van de lijst **ΔList(L2)=Ø** gelijk aan nul. M.a.w. geen element van **ΔList(L2)** is nul; wat wil zeggen dat alle elementen van **L1** verschillend zijn.

Voor het uitvoeren van de simulatie plaatsen we bovenstaand programma in een lus, bv. een **For**-lus.

```
ClearList L1,L2,L3
For (I,1,1ØØ)
Repeat sum(ΔList(L2)=Ø)=Ø
randInt(1,15,3)→L1
L1→L2
SortA(L2)
End
sum(L1>1Ø)→L3(I)
End
sum(L3=3)/1ØØ→N
Disp N
```

```
prgmWITHOUT
                0
             Done
              .02
             Done
              .03
             Done
```

```
             Done
              .02
             Done
              .01
             Done
              .01
             Done
```

Een uitgebreider programma i.v.m. het uitvoeren van trekkingen vind je in de appendix (**DRAWING**).

7. Opdrachten

1. We gooien met drie muntstukken. Hoe groot is de kans op het gooien van minstens 1 keer munt ? Simuleer dit.

2. In een doos zitten 20 gloeilampen waarvan 15 goede en 5 slechte. Men neemt lukraak in één greep 3 lampen uit de doos. Wat is de kans dat
   (a) de 3 lampen slecht zijn ?
   (b) de 3 lampen goed zijn ?
   (c) één lamp slecht is en twee lampen goed zijn ?

   Wijzigen de oplossingen indien we de lampen één na één uit de doos nemen ?

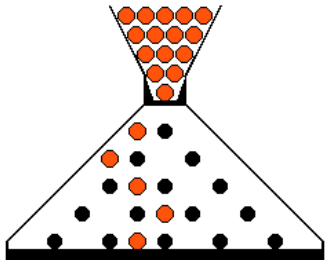
   Benader de kans op 3 slechte lampen via simulatie.

3. Een spel bestaat uit het lukraak kiezen van 3 getallen tussen 0 en 99 (herhaling is toegestaan). Men wint het spel als de som van deze drie getallen deelbaar is door 5. Bepaal de experimentele kans op winst d.m.v. simulatie.

4. Benader $\int_0^1 e^{-x^2}\,dx$ met simulatie.

5. **Galton-bord**

Een Galton-bord is een machine die toelaat een bal te rollen doorheen rijen pinnen. De pinnen zijn op het bord bevestigd zoals aangegeven op de figuur rechts afgebeeld.

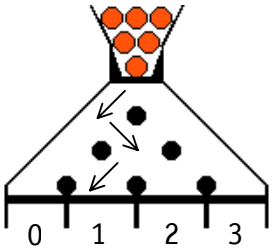


Wanneer de bal een pin raakt, is de kans om rechts te vallen gelijk aan de kans om links te vallen. Indien de bal alle rijen van pinnen gepasseerd is, valt de bal in één van de gleuven.

De vraag die we ons stellen is op hoeveel manieren de bal in een bepaalde gleuf kan vallen.

a. Beschouw een Galton-bord met drie rijen pinnen.

Op hoeveel manieren kan iedere gleuf bereikt worden ?
(de aangeduide weg op de figuur noteren we met LRL)



Gleuf x	0	1	2	3
Mogelijke wegen naar x				

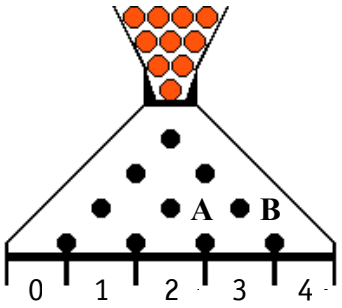
Indien we het aantal keren dat een bal naar rechts moet vallen om in een bepaalde gleuf terecht te komen tellen, kunnen we stellen dat het aantal mogelijke wegen om gleuf x te bereiken gelijk is aan : $\binom{3}{x}$.

Daar de kans op iedere weg gelijk is, is volgens het Laplace-model de kans op om in gleuf x terecht te komen gelijk aan $\frac{\#\text{ wegen naar } x}{\#\text{ wegen}}$.

Wat geeft dit voor een Galton-bord met drie rijen pinnen ?

b. We breiden het Galton-bord uit met één rij pinnen.

Hoeveel wegen leiden naar gleuf 3 ?



→ Aantal wegen tot punt **A** .....
→ Aantal wegen tot punt **B** .....
→ Aantal wegen tot gleuf 3 .....

Gleuf x	0	1	2	3	4
Mogelijke wegen naar x					

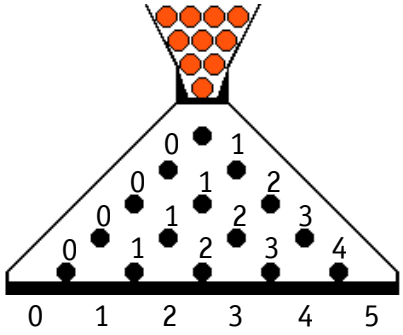
Wat besluit je in dit geval i.v.m. de kansen om in gleuf x te belanden ?

c. ***Algemeen***

Wat is de kans om in gleuf x te belanden voor een Galton-bord met n rijen pinnen ?

d. **Simulatie van het Galton-bord**

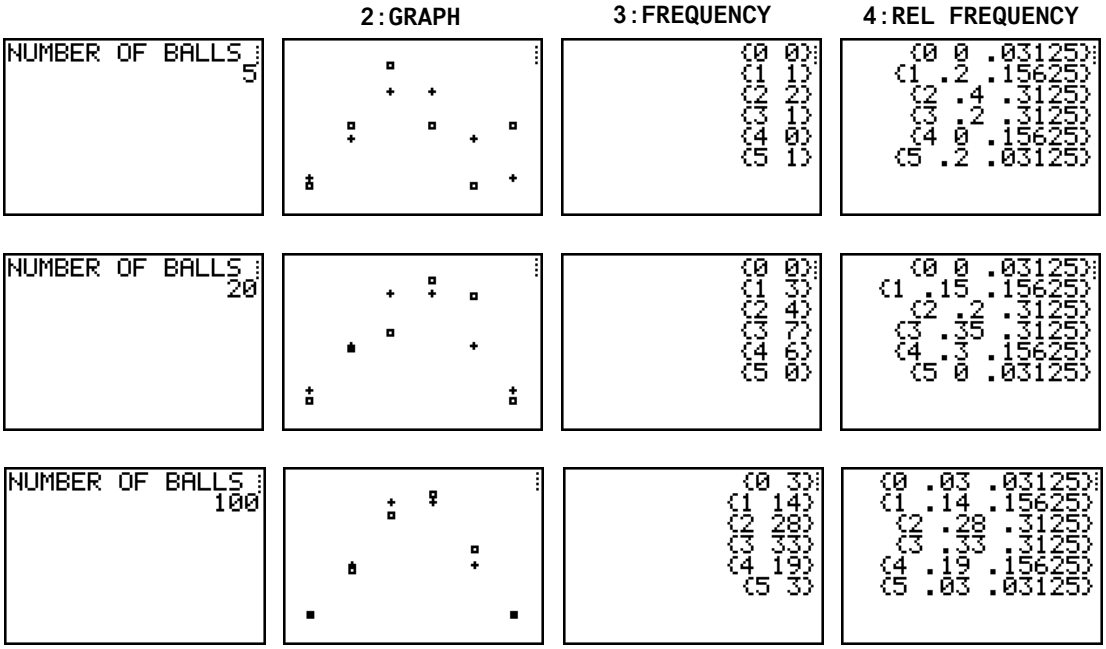
In wat volgt simuleren we een Galton-bord met vijf rijen pinnen.



Het vallen naar rechts coderen we met 1 en het vallen naar links met 0. Het onderstaande programma simuleert het rollen van één bal.

```
Ø→P
For(J,1,5)
P+randInt(Ø,1)→P
End
```

Om meer dan één bal te simuleren, moet je bovenstaand programma in een lus plaatsen (zie appendix : **GALTON**)



Appendix : TI-83 Plus programs

a. Galton's pinball machine

```
ClrList L1,L2,L3,L4,L5
Input "NUMBER OF ROWS   ",R
Input "NUMBER OF BALLS  ",N
seq(X,X,Ø,R)→L1
For(I,1,R+1)
Ø→L2(I)
End
N→M
Lbl H
For(I,1,N)
Ø→P
For(J,1,R)
P+randInt(Ø,1)→P
End
P→L5(I):L2(P+1)+1→L2(P+1)
End
L2/M→L3
binompdf(R,.5)→L4
Lbl G
ClrHome
Menu("GALTON","EXTRA BALLS",A,"GRAPH",B,"FREQUENCY",C,"REL
FREQUENCY",D,"TOTAL OF BALLS",E,"END",F)
Lbl A
Input "NUMBER OF BALLS  ",N
ClrList L5
M+N→M
Goto H
Lbl B
PlotsOff
FnOff
Plot1(Scatter,L1,L3,□):Plot2(Scatter,L1,L4,+)
AxesOff:ZoomStat:DispGraph
Pause
Goto G
Lbl C
For(K,1,R+1)
Disp {L1(K),L2(K)}
Pause
End
Goto G
Lbl D
For(K,1,R+1)
Disp {L1(K),L3(K),L4(K)}
Pause
End
Goto G
Lbl E
Disp "NUMBER OF BALLS",M
Pause
Goto G
Lbl F
```

```
Stop
```

b. Drawing with or without replacement

```
Menu("REPLACEMENT","WITHOUT",A,"WITH",B)
Lbl A
Disp "DRAW"
Input R
Disp "OUT OF"
Input N
Repeat sum(ΔList(L2)=Ø)=Ø
randInt(1,N,R)→L1
L1→L2
SortA(L2)
End
Disp L1
Stop
Lbl B
Disp "DRAW"
Input R
Disp "OUT OF"
Input N
randInt(1,N,R)→L1
Disp L1
Stop
```

Literatuur

a. *Handleiding TI-83*, Texas Instruments (education.ti.com), 1996

b. *Statistiek met een grafisch rekentoestel*, Guido Herweyers – Koen Stulens, Acco Leuven, 2000

c. *The Practice of Statistics, TI-83 Graphing Calculator Enhanced*, D.S.Yates, D.S. Moore, G.P. McCabe, W.H. Freeman and Company, New York, 1999.