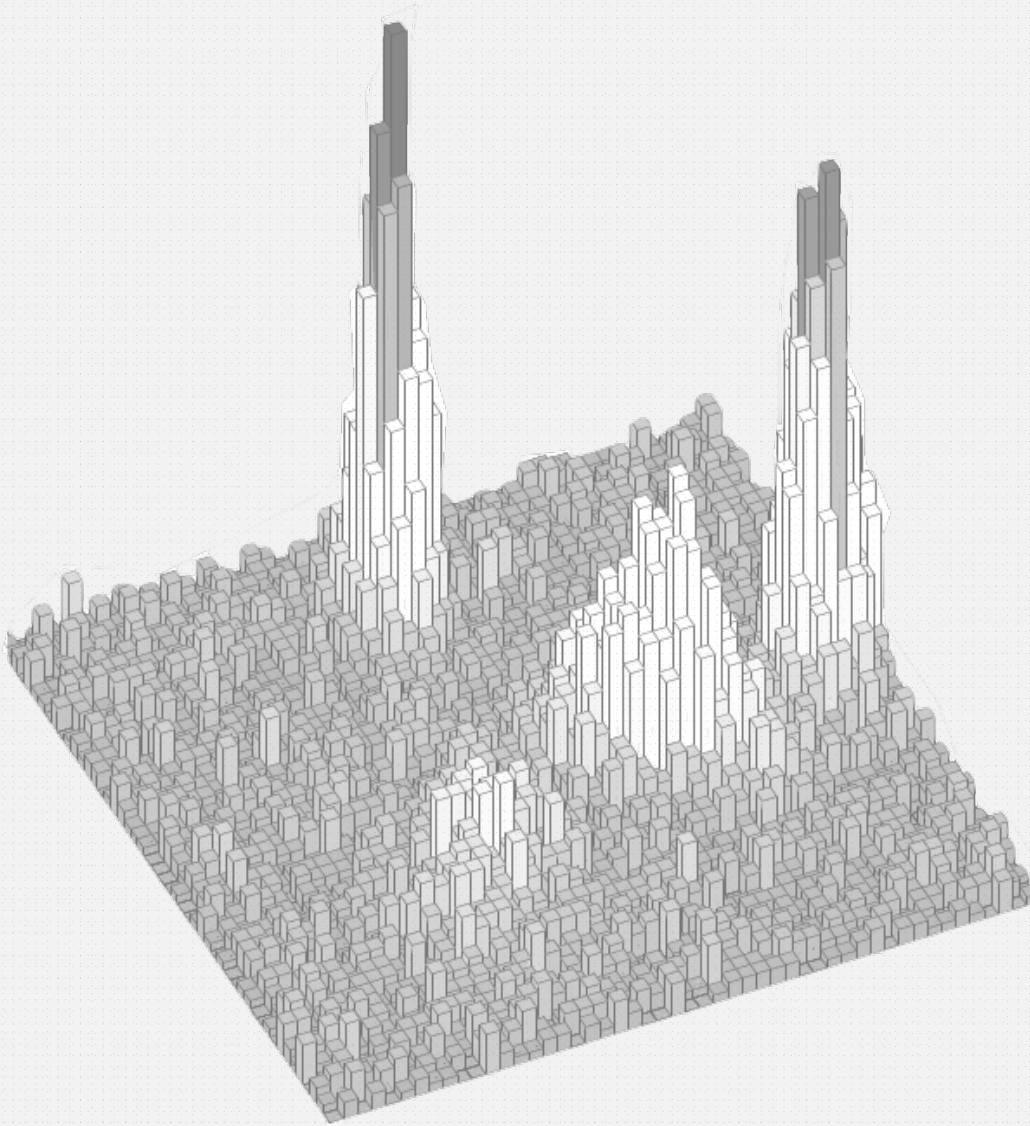


# Practicum 2 - Monte Carlo-Simulaties

Bruno Vandekerkhove



# Inhoudsopgave

<b>Simuleren van Sparen en Beleggen</b>	<b>1</b>
1.1 Opdracht 1	1
1.2 Opdracht 2	1
1.3 Opdracht 3	1
1.4 Opdracht 4	1
1.5 Opdracht 5	2
1.6 Opdracht 6	2
1.7 Opdracht 7	2
1.8 Opdracht 8	3
1.9 Opdracht 9	3
1.10 Opdracht 10	5
1.11 Opdracht 11	5
1.12 Opdracht 12	5
<b>Bronnen</b>	<b>5</b>
<b>Evaluatie</b>	<b>6</b>

## Simuleren van Sparen en Beleggen

De broncode bevindt zich in de **src** folder. Het algemene script (**src/s0216676\_script**) is opgedeeld in secties, één per opgave. Elke opgave wordt hieronder afzonderlijk beantwoord. Aan het einde van elk antwoord wordt (indien nodig) de broncode weergegeven.

### Opdracht 1

Hier is een implementatie :

```
1 function [yield, invested, value] = s0216676_simulateSavingInvesting(budget, rate, months)
2     value = repelem(budget * 1.02 .^ (0:floor(months/12)), 1, 12);
3     invested = sum(value(1:months));
4     for j = 13:12:months % Consider each month of january
5         win = sum(value(j-12:j-1) .* (((12:-1:1)/12) * (rate/100))); % Calculate savings
6         value(j) = value(j) + (win - 0.15 * (win > 980) * (win - 980));
7         value(j-12:j) = cumsum(value(j-12:j)); % Accumulate sums
8     end
9     value = value(1:months); value(j:end) = cumsum(value(j:end));
10    yield = value(months) / invested - 1;
11 end
```

### Opdracht 2

Het resultaat van de gegeven code is te zien in figuur 1. De totale investering bedraagt zo'n 96091 euro. De relatieve winsten bedragen 5.46%, 11.34%, 23.34% en 50.82%.

### Opdracht 3

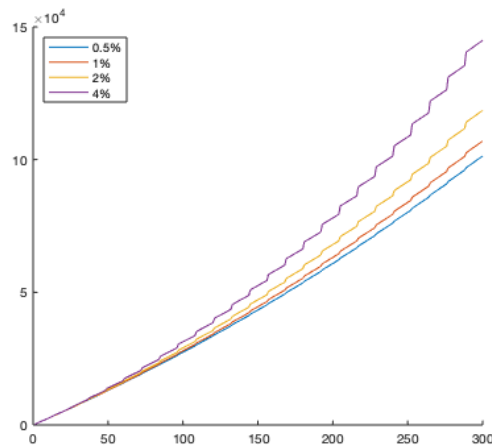
Het bestand wordt ingeladen.

```
1 load('Funds.mat')
```

### Opdracht 4

Men kan de parameters berekenen door eerst  $\sigma$  te bepalen op basis van de rendementen, het gemiddelde te berekenen van diezelfde rendementen en vervolgens  $\mu$  daaruit te bepalen.

De bekomen parameters bedragen  $\mu = 3.208565 \times 10^{-3}$ ,  $\sigma = 3.070426 \times 10^{-2}$  (voor EUN5) en  $\mu = 7.857782 \times 10^{-3}$ ,  $\sigma = 3.277546 \times 10^{-2}$  (voor VWRL).



Figuur 1: Simulatie van spaarrekeningen met verschillende rentevoeten.

```

1 function [mu,sigma] = s0216676_estimateParameters(s)
2     rendements = log(s(2:end) ./ s(1:end-1)); % Base doesn't matter
3     sigma = std(rendements);
4     mu = mean(rendements) + 0.5 * sigma^2;
5 end

```

## Opdracht 5

De implementatie maakt gebruik van een eenvoudige `for` loop.

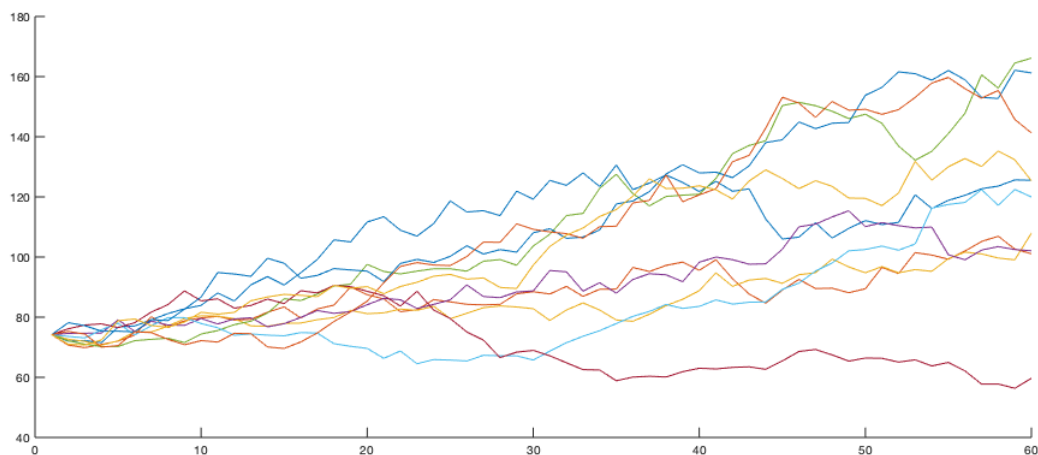
```

1 function [path] = s0216676_simulateFundPath(initialPrice, mu, sigma, months)
2     alpha = mu - 0.5 * sigma^2;
3     path = [initialPrice 2:months];
4     for t = 2:months
5         path(t) = path(t-1) * exp(alpha + sigma * randn);
6     end
7 end

```

## Opdracht 6

In figuur 2 worden de resulterende paden afgebeeld. Een aantal paden wijken af van de werkelijke koers afgebeeld in de opgave, maar de meesten komen realistisch over. De initiële waarde bedraagt telkens 74.19 euro wat overeenstemt met de waarde in dollar op 26 november 2019.



Figuur 2: Simulatie van spaarrekeningen met verschillende rentevoeten.

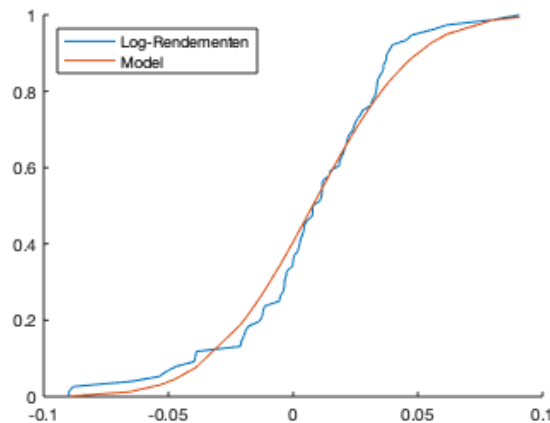
```

1 figure; hold all;
2 for i = 1:10
3     plot(s0216676_simulateFundPath(74.19, mu, sigma, 60)); % Converted to euros (26/11/2019)
4 end

```

## Opdracht 7

De cumulatieve distributiefuncties zijn afgebeeld in figuur 3. Op basis hiervan kan men met een zekere graad van vertrouwen beweren dat de log-rendementen effectief normaal verdeeld is. Normaal gezien maakt men gebruik van specifieke testen om dit te kwantificeren.



Figuur 3: Cumulatieve distributiefuncties van de log-rendementen en het model.

```

1 [f,x] = ecdf(log(VWRL(2:end) ./ VWRL(1:end-1)));
2 [f_norm] = normcdf(x, mu, sigma);
3 figure; hold all;
4 plot(x, f, x, f_norm);
5 legend('Log-Rendementen', 'Model', 'Location', 'NorthWest');

```

## Opdracht 8

Het kon ook met een lus, maar ik deed het deze keer met ingebouwde functies.

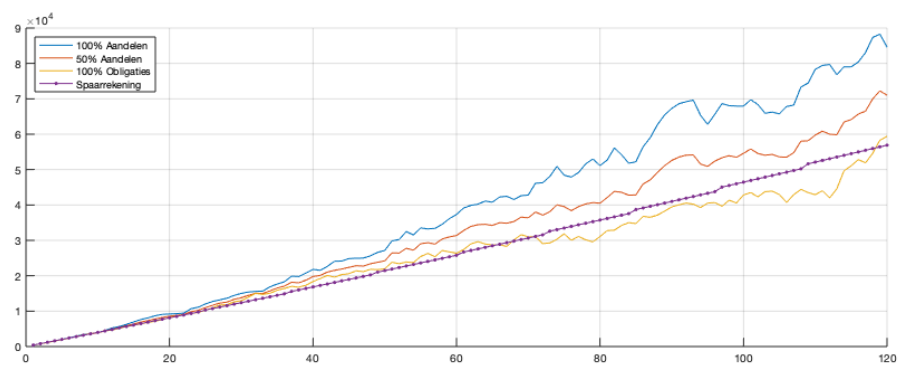
```

1 function [yield, invested, value, units] = ...
2     s0216676_simulateFundInvestingPath(budget, pricePath, alpha)
3 months = size(alpha,1);
4 budgets = repelem(budget * (1.02 .^ (0:floor(months/12))), 12, 2);
5 invested = sum(budgets(1:months,1));
6 units = (budgets(1:months,:) .* [alpha (1-alpha)] - 6) / 1.0035;
7 for i = 1:2
8     units(units(:,i) < 0, 3-i) = (budgets(units(:,i) < 0) - 6) / 1.0035;
9     units(units(:,i) < 0, i) = 0;
10 end
11 units = cumsum(units ./ pricePath);
12 value = units .* pricePath;
13 yield = sum(value(end,:)) / invested - 1;

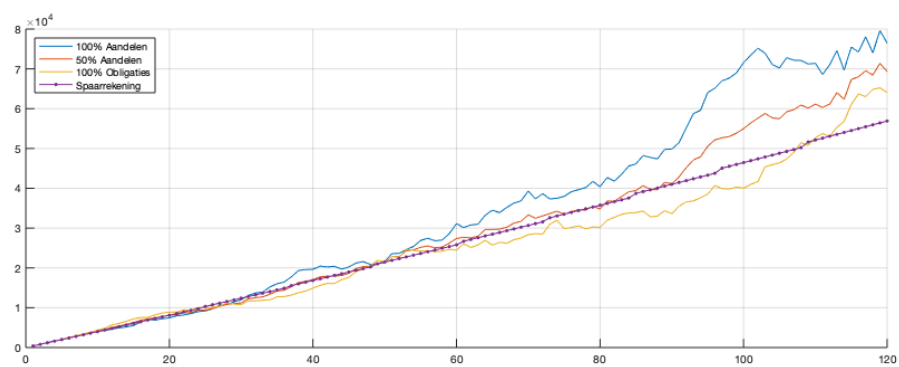
```

## Opdracht 9

De resulterende figuren zijn afgebeeld op de volgende pagina.

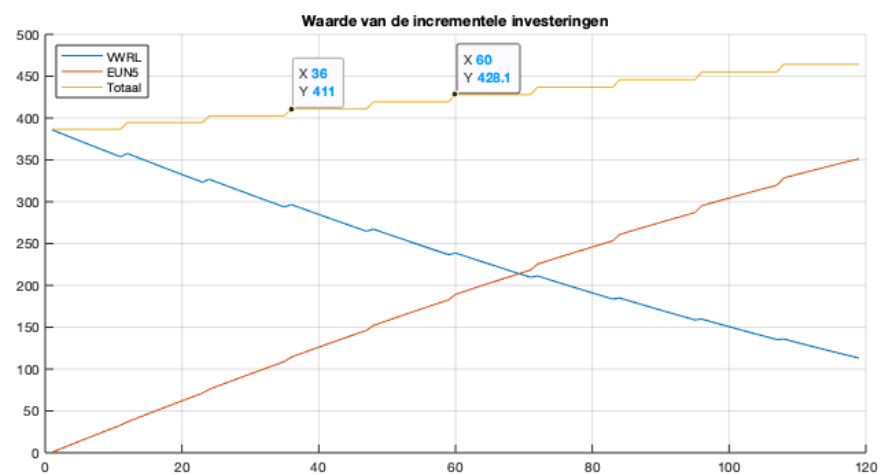


(a) Iteratie 1



(b) Iteratie 2

Figuur 4: Waarde van vier type investeringen doorheen een periode van 120 maanden.



Figuur 5: Waarde van de investering op maandelijkse basis.

Opdracht 10

Opdracht 11

Opdracht 12

## Evaluatie

Ik spendeerde ongeveer 17 uur aan het practicum en zo'n 3 uur aan het verslag. De moeilijkheidsgraad lag volgens mij goed. De terminologie was meer dan duidelijk en de taak (net zoals het andere practicum) best interessant.