Oppgave 11

Pris på avkastningsgaranti

```
<< "Statistics`ContinuousDistributions`"
<< "Graphics`Graphics`"
<< Graphics`Legend`
```

```
α = 0.20;
μ = 0.10;
σ = 0.20;
ν = μ - σ^2/2;
δ = 0.05;
γ = 0.03;
n = 100000;
T = 20;
```

```
(*Φ[x_]:=1/2 (1+Erf[x/√2]);*)
```

```
Φ[x_] := CDF[NormalDistribution[0, 1], x];
```

```
put[s0_, k_, t_] :=
  Module[{d1, d2}, d1 = (Log[s0/k] + δ t)/(σ √t) + (σ √t)/2; d2 = d1 - σ √t; k e^(-δ t) Φ[-d2] - s0 Φ[-d1]];
```

```
p = Which[γ < δ && γ > δ + Log[1 - α],
  x /. FindRoot[x == put[(1 - x) α, e^γ - (1 - x) (1 - α) e^δ, 1], {x, 0}], γ ≤ δ + Log[1 - α], Print
   "Garantien er gratis, fordi bankinnskuddet alene oppfyller garantien"], γ ≥ δ,
  Print["Ingen løsning når garantien er høyere enn den risikofrie avkastningen"]]
```

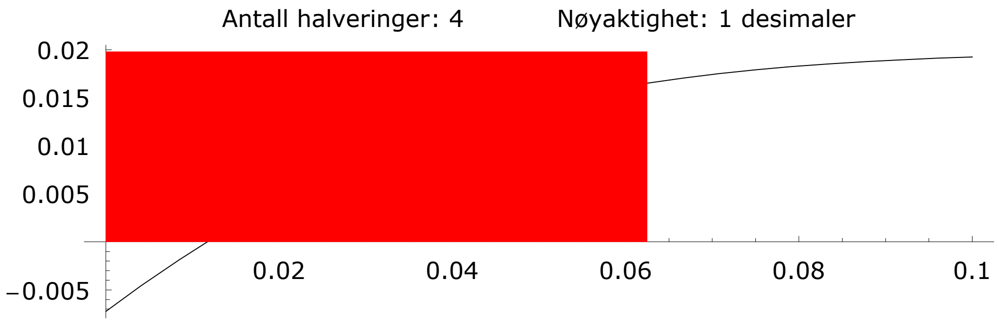
```
0.0117119
```

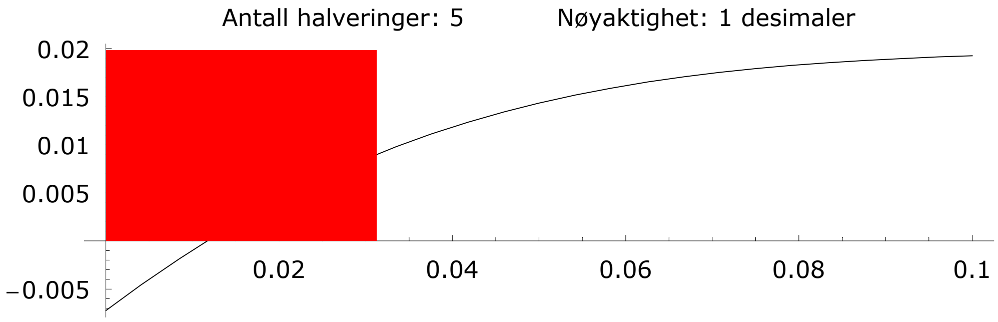
```
f[x_] := x - put[(1 - x) α, e^γ - (1 - x) (1 - α) e^δ, 1];
```

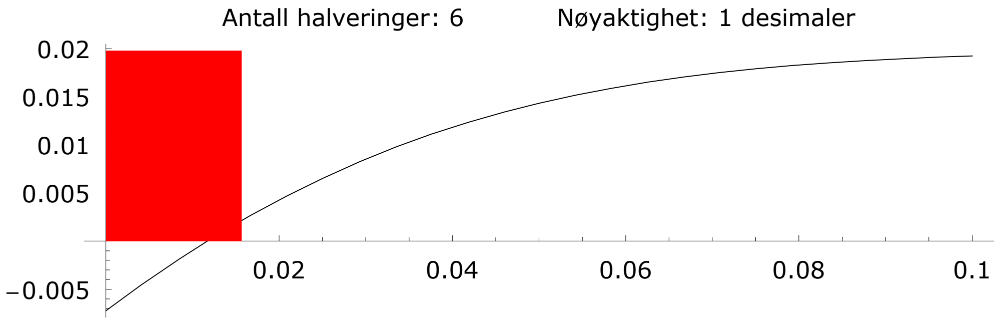
Grafisk fremstilling av hvordan metoden "bisection" virker.

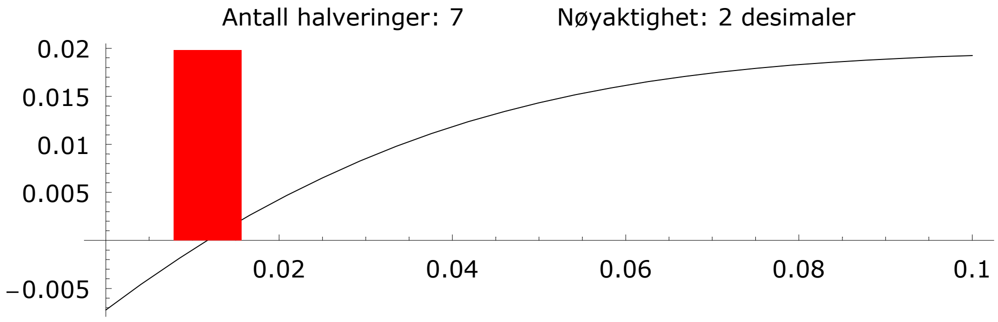
```
bisection[m_] := Module[{pMin, pMax, pTest}, pMin = 0;
   pMax = 1; pTest = pMax; Do[If[f[pTest] > 0, pMax = pTest, pMin = pTest];
    pTest = (pMin + pMax)/2, {m}]; N[{pMin, pMax}]];
```

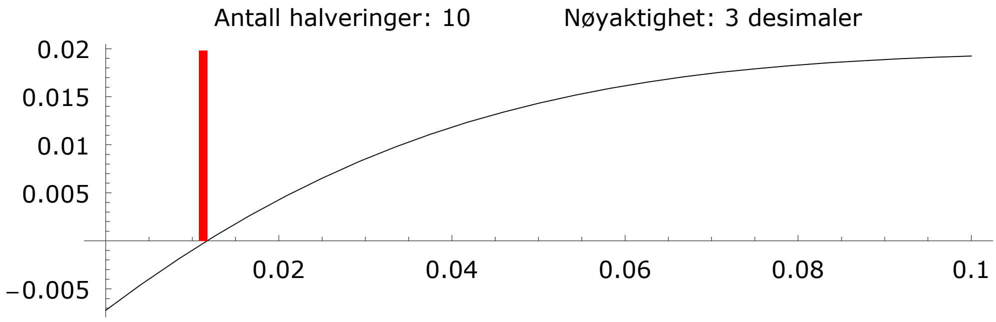
```
Do[Show[Plot[f[x], {x, 0, .1}, PlotRange → All,
   PlotLabel → "Antall halveringer:  " <> ToString[i - 1] <> "\tNøyaktighet:  " <>
     ToString[IntegerPart[Log[10, 2.^(i-1)]]] <> " desimaler",
   DisplayFunction → Identity], Graphics[{RGBColor[1, 0, 0],
    Rectangle[{bisection[i]⟦1⟧, 0}, {bisection[i]⟦2⟧, f[1]}]}],
  DisplayFunction → $DisplayFunction, DefaultFont → {"Verdana", 11},
  AspectRatio → .3, ImageSize → 500], {i, 5, 21}]
```

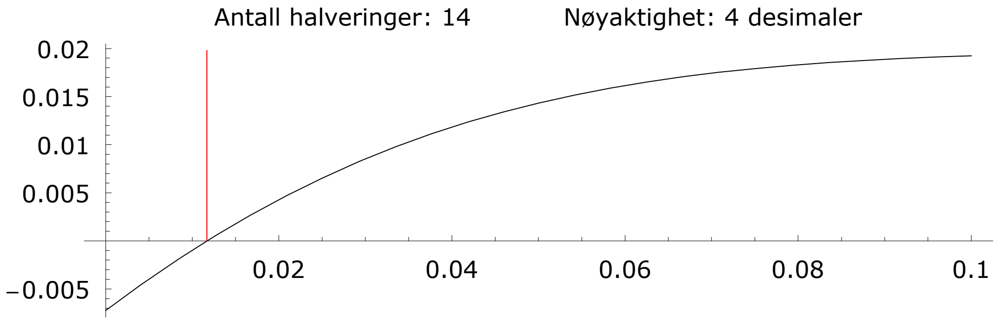


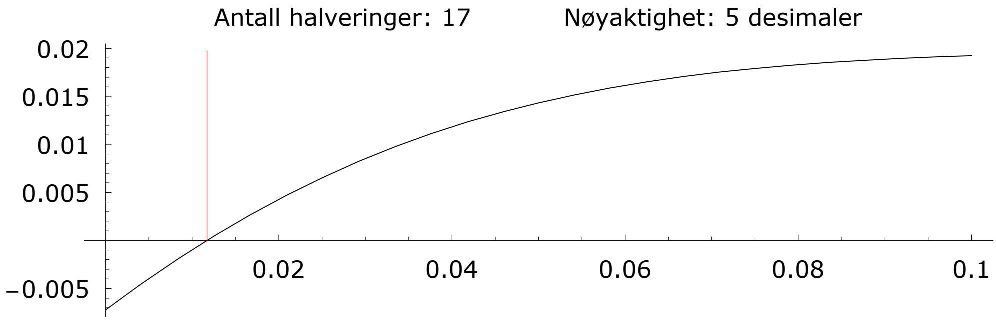


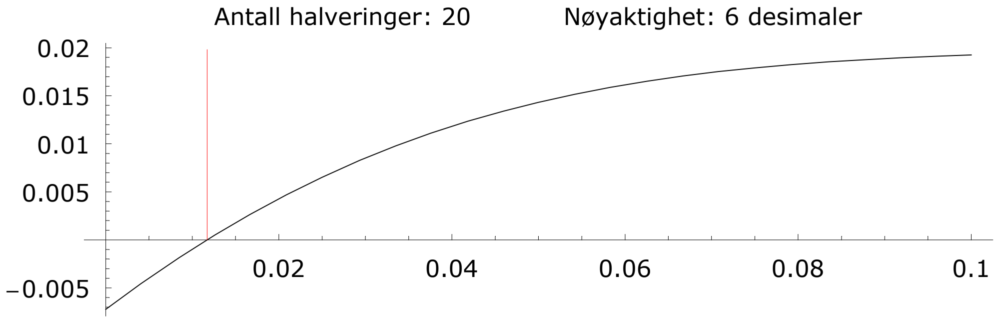












Sannsynlighetsfordelinger

```
Timing[a = Partition[α ⅇ^(γ+σ RandomArray[NormalDistribution[0,1],n T]) + (1 - α) ⅇ^δ, T];]
```

```
{5.778 Second, Null}
```

```
f[aSim_] := Fold[(1 + #1) #2 &, 0, aSim];
fg[aSim_] := Fold[(1 + #1) Max[ⅇ^γ, (1 - p) #2] &, 0, aSim];
```

```
(*Timing[fSim=Table[f[a[[i]]],{i,n}];]
  Timing[fgSim=Table[fg[a[[i]]],{i,n}];]*)
```

```
fCompile = Compile[{{matrise, _Real, 1}}, Fold[(1 + #1) #2 &, 0, matrise]];
```

```
fgCompile = Compile[{{matrise, _Real, 1}, {γc, _Real}, {pc, _Real}},
   Fold[(1 + #1) Max[ⅇ^(γc), (1 - pc) * #2] &, 0, matrise]];
```

```
Timing[fSimCompile = Table[fCompile[a[[i]]], {i, n}];]
```

```
{1.933 Second, Null}
```

```
Timing[fgSimCompile = Table[fgCompile[a[[i]], γ, p], {i, n}];]
```

```
{11.326 Second, Null}
```

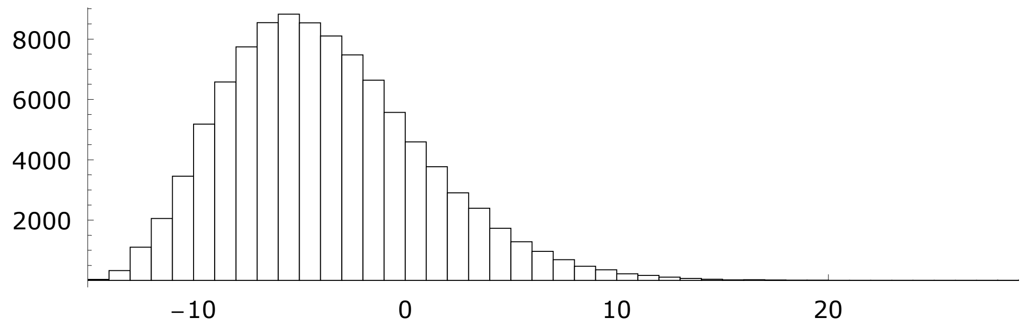
$$\Psi = 100\left(\frac{\text{fgSimCompile}}{\text{fSimCompile}} - 1\right);$$

```
pdf[data_, oppdeling_] := Histogram[data, BarStyle → RGBColor[1, 1, 1],
   HistogramCategories → Table[i, {i, -100, 100, oppdeling}],
   DefaultFont → {"Verdana", 11}, AspectRatio → .3, ImageSize → 500];
```

$$N\left[\frac{\text{Length[Select[}\Psi\text{, \#1 < 0. \&]]}}{n}\right]$$

```
0.80161
```

```
pdf[Ψ, 1];
```



Replikerende portefølje

```
h = 250;
k = ⅇ^γ - (1 - p) (1 - α) ⅇ^δ;
s0 = (1 - p) α;
```

Tid til utløp:

$$tT = 1 - \frac{\text{Range}[0, h-1]}{h};$$

$$s = \text{FoldList}\left[\#1\ \#2\ \&,\ s0,\ e^{\frac{\nu}{h} + \frac{\sigma\ \text{RandomArray[NormalDistribution[0,1],h-1]}}{\sqrt{h}}}\right];$$

Porteføljen tilpasses etterskuddsvis. På tid t finner jeg den sammensetningen som jeg skulle hatt på tid t-1 for at porteføljen skulle fått samme verdi som opsjonen på tid t. Aksjeandelen er lik den deriverte mhp aksjekursen.

```
at = -Φ[-(Log[s/k]/(σ √tT) + ((δ + σ^2/2) √tT)/σ)];

bt = (put[s, k, tT] - at s) ⅇ^(δ tT);

farge = {RGBColor[1, 0, 0], RGBColor[0, 1, 0], RGBColor[0, 0, 1]};

Plot[{s⟦Round[t]⟧, at⟦Round[t]⟧ s⟦Round[t]⟧, bt⟦Round[t]⟧},
  {t, 1, handledager}, PlotRange → {Automatic, {-.30, .30}},
  PlotStyle → farge, PlotLabel → "Replikerende  portefølje", Frame → True,
  PlotLegend → {" Aksjekurs", " Aksjer  i  replikerende  portefølje",
   " Obligasjoner  i  replikerende  portefølje"},
  LegendTextSpace → 20, LegendSize → 1.46, LegendPosition → {-.665, -.6},
  DefaultFont → {"Verdana", 11}, ImageSize → 500, AspectRatio → .3,
  LegendBackground → GrayLevel[.9], LegendShadow → {0, 0}];
```

Manglende selvfinansiering:

```
skalHaPåTidt = Delete[at s + bt ⅇ^(-δ tT), 1];
harPåTidt = Delete[at, -1] Delete[s, 1] + Delete[bt, -1] Delete[ⅇ^(-δ tT), 1];
tilskudd = skalHaPåTidt - harPåTidt;
Min[tilskudd]
Max[tilskudd]
```

```
-0.000258783
```

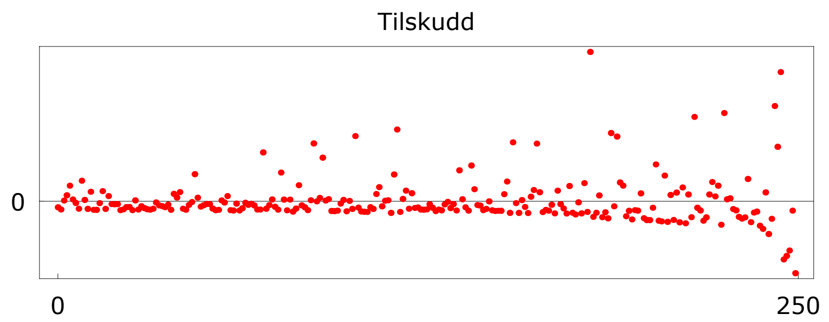
```
0.00053647
```

```
Table[at⟦i - 1⟧ s⟦i⟧ + bt⟦i - 1⟧ ⅇ^(-δ tT⟦i⟧), {i, 2, h}] == harPåTidt
Table[at⟦i⟧ s⟦i⟧ + bt⟦i⟧ ⅇ^(-δ tT⟦i⟧), {i, 2, h}] == skalHaPåTidt
```

```
True
```

```
True
```

```
ListPlot[tilskudd, Frame → True, PlotStyle → RGBColor[1, 0, 0],
  PlotRange → All, PlotLabel → "Tilskudd", DefaultFont → {"Verdana", 11},
  FrameTicks → {{{1, "0"}, {h, ToString[h]}}, Table[.005 i, {i, -5, 5}], None, None},
  ImageSize → 500, AspectRatio → .3];
```



Kontantverdi av tilskudd som andel av prisen på garantien:

```
beregnTilskudd := Module[{}, s = FoldList[#1 #2 &, s0, ℯ^(v/h + σ RandomArray[NormalDistribution[0,1],h-1]/√h)];
    at = -Φ[-(Log[s/k]/(σ √tT) + ((δ + σ^2/2) √tT)/σ)]; bt = (put[s, k, tT] - at s) ℯ^(δ tT);
    skalHaPåTidt = Delete[at s + bt ℯ^(-δ tT), 1];
    harPåTidt = Delete[at, -1] Delete[s, 1] + Delete[bt, -1] Delete[ℯ^(-δ tT), 1];
    tilskudd = skalHaPåTidt - harPåTidt; Delete[ℯ^(-δ Reverse[tT]), -1].tilskudd/p];

Timing[replTilskudd = Table[beregnTilskudd, {100}];]
```

```
{20.51 Second, Null}
```

```
pdf[replTilskudd, .02];
```