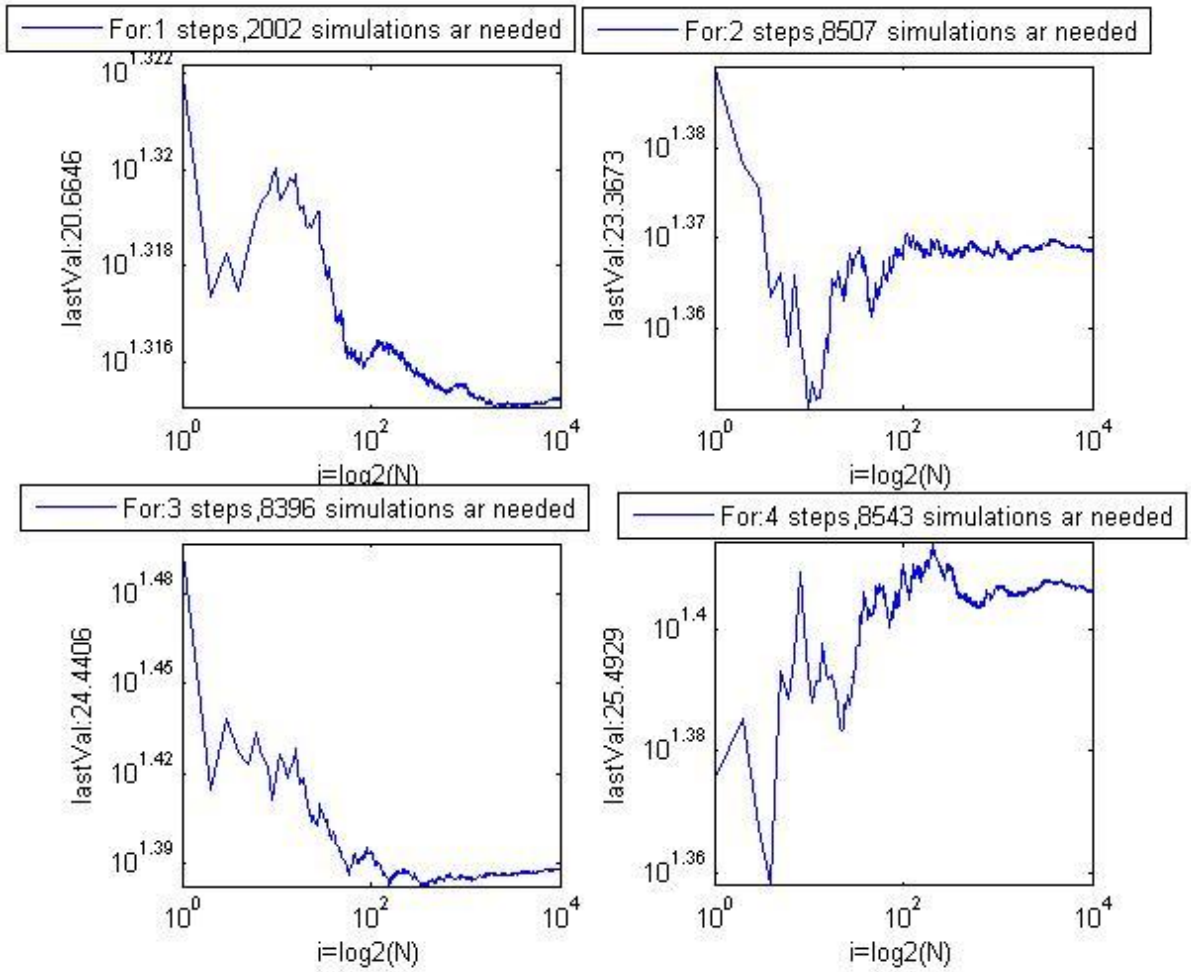
Verslag Modelleren en Simuleren practicum 2

Door Francis Duvivier

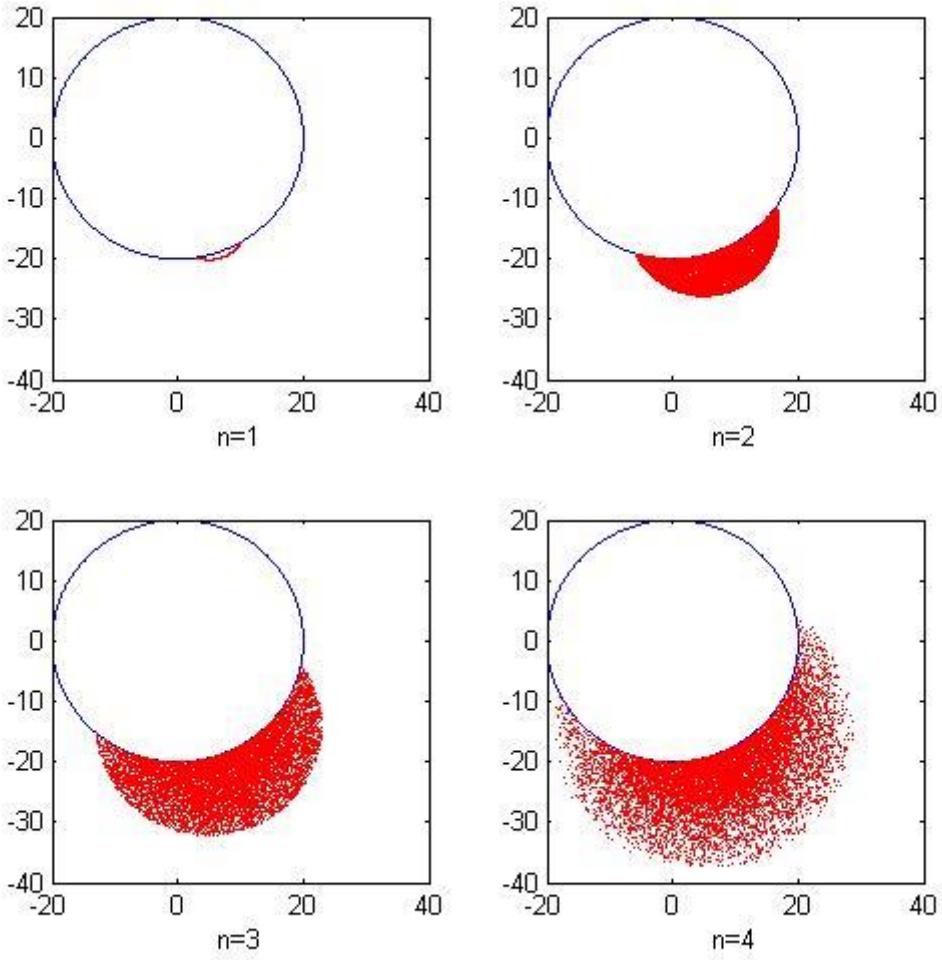
Voor Nico Achtsis en Dirk Nuyens

Opgave 1

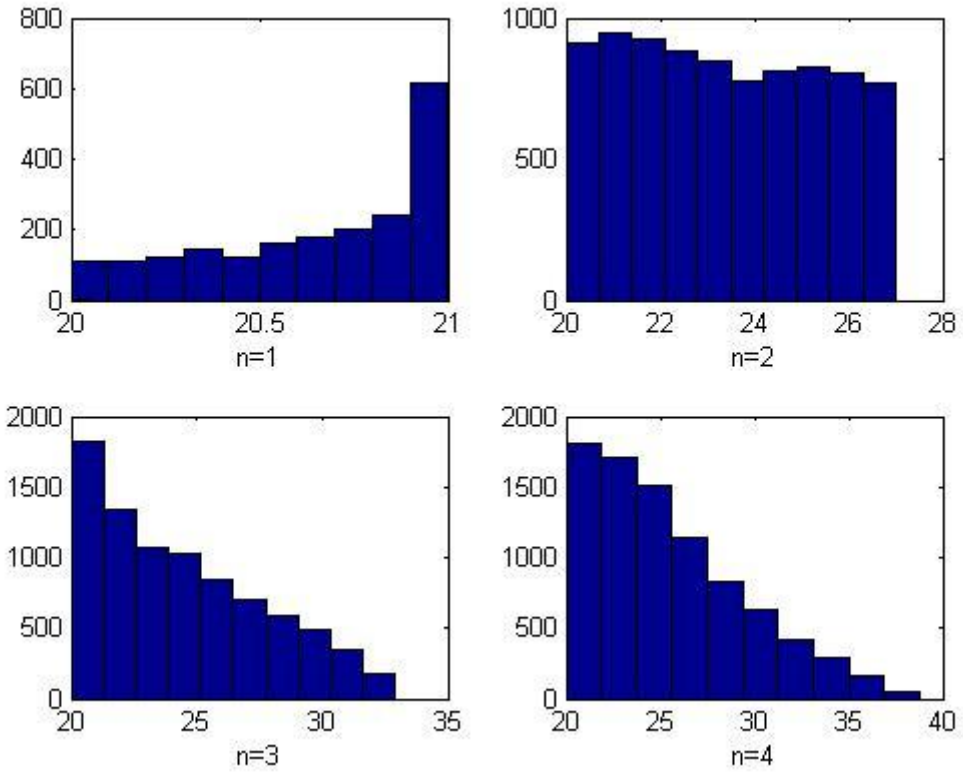
De oplossing kan u vinden in de bijgevoede matlab-bestanden en onderstaande afbeeldingen. Hieronder ziet u de convergentie-grafieken tesamen met het aantal simulaties nodig om een nauwkeurigheid van 1cm te bekomen.



Hieronder ziet u de eindpunten van de eend en de shietradius, de eindpunten waard de eend doodgeschoten word zijn er uit gelaten.

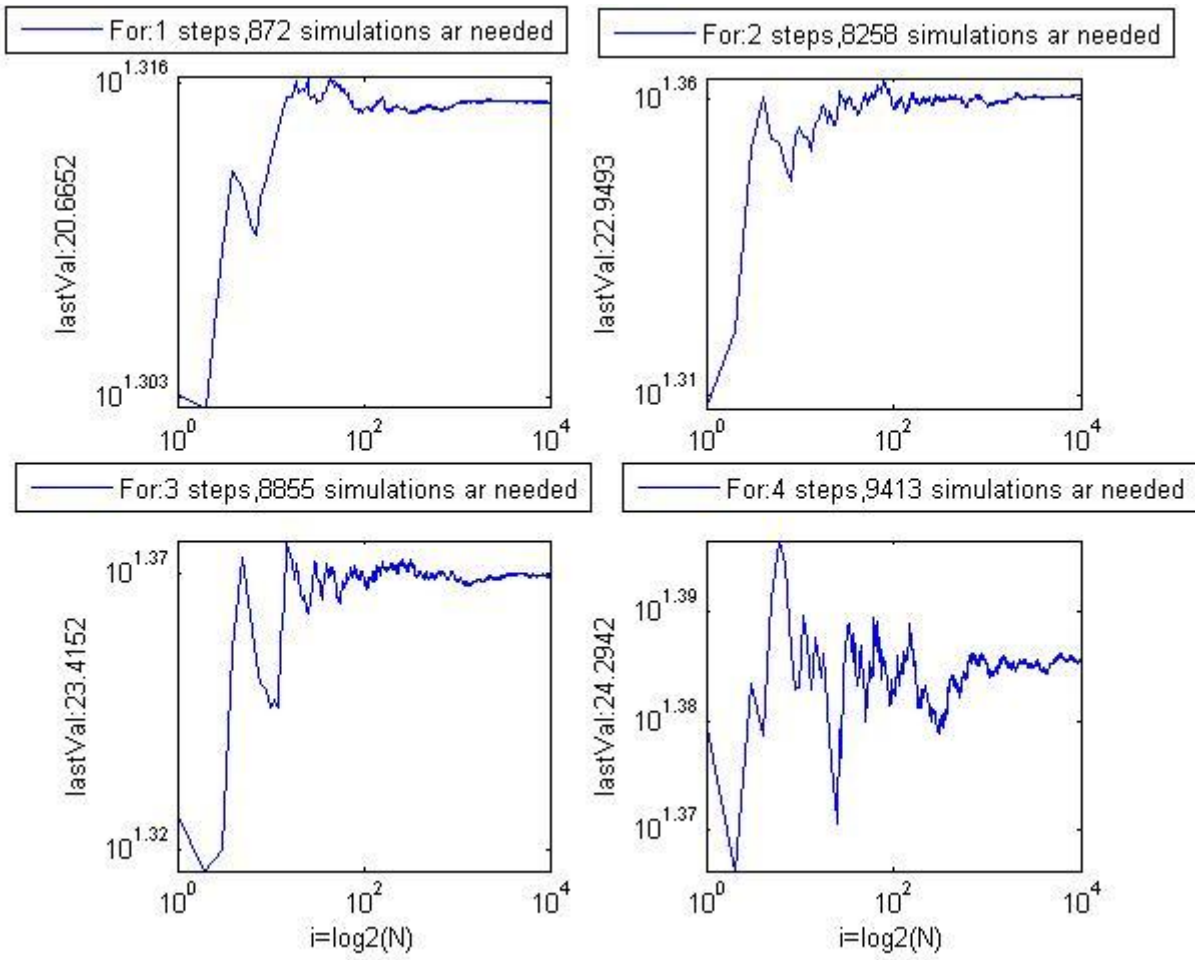


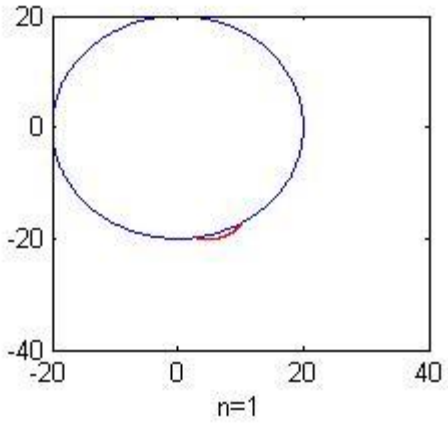
Hieronder ziet u de histogrammen van de afstanden voor Nn.



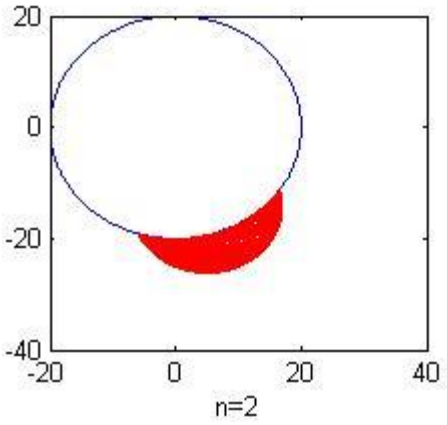
Opgave2

Hieronder ziet u dezelfde afbeeldingen, maar dan gemaakt zoals aangegeven in opgave2. De matlab code voor opgave 2 zit al verwerkt in die van opgave1. Dus opgave2.m roept gewoon opgave2b op. We verwachten hier een kleinere variantie omdat de oplossingen die bijkomen over het algemeen vrij dicht bij het gemiddelde zullen liggen want dat gemiddelde is niet ver buiten de shietradius omdat de meest waarschijnlijke eindafstand 20 meter is.

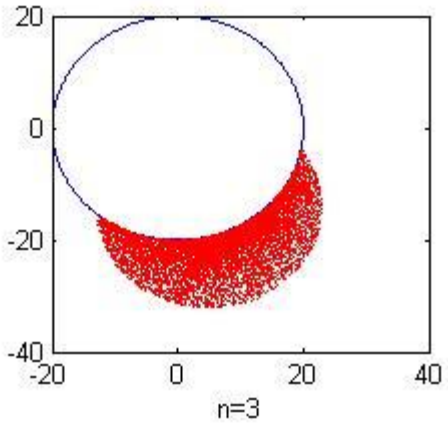




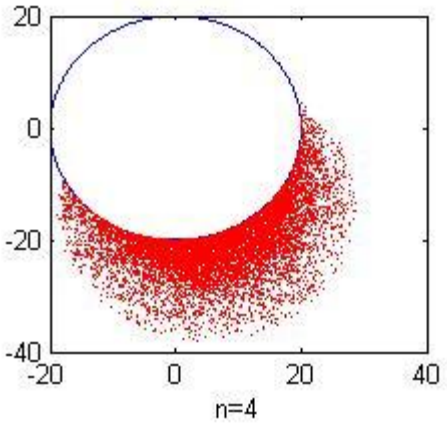
n=1



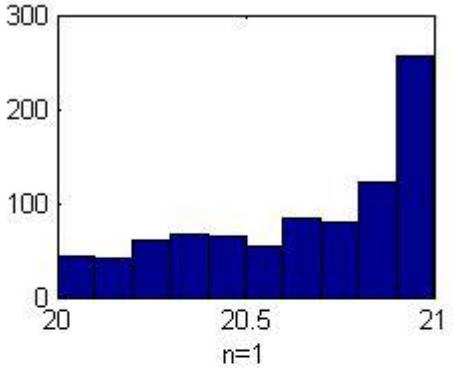
n=2



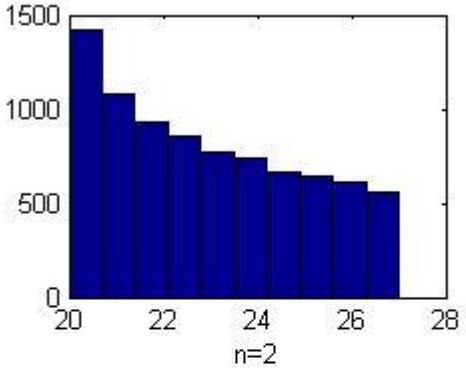
n=3



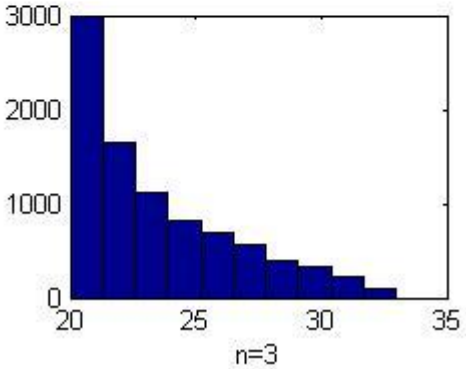
n=4



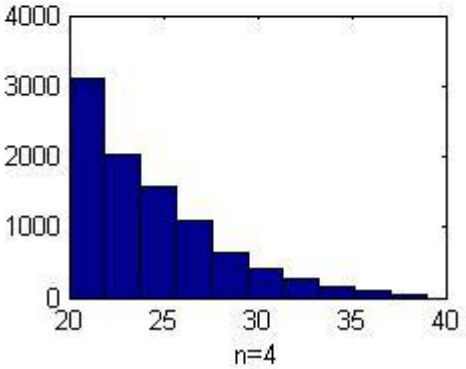
n=1



n=2



n=3



n=4

Bijlagen: matlab functies

```
function [distList posList] = opgave1a(n,N,escapeMode)
%OPGAVE1B Summary of this function goes here
%   Detailed explanation goes here
if(~exist('escapeMode'))
    escapeMode=0; %alle gevallen
end
stepDist=6;
distList=zeros(N,1);
posList=zeros(N,2);
i1=1;

while i1<=N
    init=InitialPosition(215207);
    randX=init(1);
    randY=init(2);
    for i2=1:n
        if (escapeMode==2 && i2==n&&
sqrt(randX^2+randY^2)>14&&sqrt(randX^2+randY^2)<26)
            newnewDist=0;
            newrandX =0;
            newrandY =0;
            while(newnewDist<20)
            newrandHoek=rand* 2 * pi;
            newrandX = randX+ cos(newrandHoek) * stepDist;
            newrandY = randY+ sin(newrandHoek) * stepDist;
            newnewDist=sqrt(newrandX^2+newrandY^2);
            end
            randX = newrandX;
            randY = newrandY;
        else
        randHoek=rand* 2 * pi;
        randX = randX+ cos(randHoek) * stepDist;
        randY = randY+ sin(randHoek) * stepDist;
        end
    end
    posList(i1,1)=randX;
    posList(i1,2)=randY;
    newDist=sqrt(randX^2+randY^2);
    if((escapeMode==2||escapeMode==1)&&newDist<20)
        i1=i1-1;
    else
        distList(i1)= newDist;
    end
    i1=i1+1;
end
end
```

```
function [ meanList ] = opgave1b( escapeMode )

%OPGAVE1B Summary of this function goes here
%   Detailed explanation goes here
if(~exist('escapeMode'))
    escapeMode=1; %alle gevallen
end

nbSims=10000;
nbSteps=4;
nbChecks=5;
nbSimsi1=zeros(nbSteps,1);
posList=zeros(nbSteps,nbSims, 2);
distList=zeros(nbSteps,nbSims);
meanList=zeros(nbSteps,nbSims);
for nbStepsi=1:nbSteps
    [ distList(nbStepsi,1:end)
posList(nbStepsi,1:end,1:end)]=opgave1a(nbStepsi,nbSims,escapeMode);
    EM=mean(distList(nbStepsi,1:end)); %the expected mean
    for i= nbSims -(0:nbSims-1)
        meanList(nbStepsi,i)=mean(distList(nbStepsi,1:i));
        if( nbSimsi1(nbStepsi)==0&&  abs(EM-meanList(nbStepsi,i))>0.01)
            nbSimsi1(nbStepsi)=i+1;
        end


    end

end
figure(1);
for nbStepsi=1:nbSteps
    subplot(2,2,nbStepsi);
    hold off;
    loglog(meanList(nbStepsi,1:end));
    xlabel('i=log2(N)');
    ylabel(strcat('lastVal: ',num2str(meanList(nbStepsi,end))));

    legend(strcat('For:  ', num2str(nbStepsi),' steps,  ',
num2str(nbSimsi1(nbStepsi)), ' simulations ar needed' ));
end

%now we drae the positions
figure(2);
for nbStepsi=1:nbSteps
    subplot(2,2,nbStepsi);
    hold off;
    drawCircle(20);
    hold on;
    scatter(posList(nbStepsi,1:end,1),posList(nbStepsi,1:end,2),0.9, 'r' );
        xlabel(strcat('n=', num2str(nbStepsi)));
    axis manual;
    axis([-20 40 -40 20]);

end

figure(3);
for nbStepsi=1:nbSteps
    subplot(2,2,nbStepsi);
    hold off;
    hist(distList(nbStepsi,1:nbSimsi1(nbStepsi)));
```

```
    xlabel(strcat('n=', num2str(nbStepsi)));

end


end

function  drawCircle( r )
ang=0:0.01:2*pi;
xp=r*cos(ang);
yp=r*sin(ang);
plot(xp,yp);
end


function [meanList]= opgave2
meanList=opgave1b(2);
end
```