# Flödespaketet

## Problemet

Differentialekvationen:

dx/dt = 1 −R*2*(x^2 − y^2)*(x^2 + y^2)^2,

dy/dt = −2*x*y*R*2(x^2 + y^2)^2

beskriver rörelsen hos en partikel i ett flöde av en vätska. I flödet så finns det en cylinder som påverkar partiklarnas bana. Uppgiften går ut på följande:

(1) Lös differentialekvationen för startpunkter där x=-4 samt y=0.2, 0.6, 1.0 och 1.4. Rita upp resultatet. Lös för tidsintervallet 0-12.

(2) Då x är en beroende variabel så kommer de olika kurvorna ha olika x-värden vid t=12. Beräkna vid vilket t kurvan med startpunkt (-4, 0.2) har samma x-värde som den med start i (-4, 1.4) har vid t=12.

(3) Konstruera ett paket bestående av 20 partiklar i en tjugohörning runt punkten (-4, 1) med radien 0.6. Beräkna arean av kurvan och studera hur den förändras när paketet rör sig längs flödet.

(4) Ge paketet en annan form. Se vad som händer.

# Lösning

## *1*

Löses av sektion 1 i Main.

Figurer: 1

För varje y-värde som är givet löses ekvationen med hjälp av ode45. Därefter ritas de utlösta y-värdena upp med avseende på de utlösta x-värdena. En variabel endpoint sparas som är den sista punkten som kurvan med startpunkt i y=0.2 når när t=12. Denna används i nästa del.

## *2*

Löses av sektion 2 i Main

Figurer: 4

**Definition:** tx = x för den kurvan som börjar i y=1.4 när t=12.

För att hitta det t där kurvan med ursprung i y=0.2 når tx så använder vi oss av ode45 igen. Denna gång med ett event som triggas när x=tx (vi fick reda på tx genom manuell körning av sektion 1 och stoppade in det i evtfun (se appendix A)). Eventet avbryter körningen och returnerar alla värden som har beräknats. Det sista av dessa är det vi är intresserade då den innehåller tidpunkten samt y-värdet där x=tx.

**Svar:** t är ungefär 15.60. Mer exakt svar finns i appendix B, fig9

## *3*

Löses av sektion 3 i Main.

Figurer: 2, 3, 5 och 6

Vi utför samma process för antal hörn i mängden {20, 40, 80}.

Först genereras polygonen med package. Därefter löses differentialekvationen för varje partikel i polygonen och vi får en datastruktur innehållandes polygonens tillstånd för alla t. Denna använder vi sedan för att beräkna arean av polygonen för varje t i en vektor. Till slut kan vi rita upp denna.

På detta sätt får vi areans kurva för 20, 40 och 80 hörn. I den kan vi se att arean varierar men gradvis ökar fram tills dess att den når ett ungefärligt konstant läge någonstans mellan 3 och 4 (beroende på vilken kurva vi kollar på). Därefter sker ingen eller en väldigt liten förändring när kurvorna planar ut.

## *4*

Löses även här av sektion 3 i Main.

Figurer: 8, 9, 10, 11

Samma process som i del 3 görs men på en annan polygon. Polygonen beräknas med:

```
x=xcenter+cos(angle)*r y=ycenter+sin(angle)*r/(i/2)
```

Intressant att notera för denna är att ett högre antal punkter gör en väldigt stor skillnad i arean. Sannolikt beror det på att detta är en mer oregelbunden form. Ojämnheterna i polygonen utgör en stor del av arean och när dessa jämnas ut så minskar den därför drastiskt.

# Appendix A: Kod

**Main.m**

```
t = [-8:0.05:12];
hold off;
format long
```

## Sektion 1

```
for y=[0.2 0.6 1.0 1.4]
    [T, XY] = ode45(@delta, t, [-4 y]);
    plot(XY(:,1), XY(:,2))
    if y==0.2
        endpoint = XY(end, :); % Används för att hitta t i sektionen efter
loopen.
    end
    hold on;
end
```

## Sektion 2

```
% Hitta t så att grafen med start i (-4, 0.2) har samma x som den med start
% i (-4, 1.4) har vid t=12
xlabel('X position')
xlabel('Y position')
op = odeset('Events', @evtfun);
[T, XY] = ode45(@delta, [12:20], [endpoint(1), endpoint(2)], op);
plot(XY(:,1), XY(:,2))
plot(XY(end, 1), XY(end, 2), 'O');
T(end);
```

## Sektion 3

```
% Generera en polygon med 20 hörn
points = package(-4, 1, 0.6, 20);
[T, XY] = calcpoints(t, points, @delta);
areacurve20 = zeros(size(t));
for i=1:length(t)
    areacurve20(i) = polygonarea(XY(1, i, :), XY(2, i, :));
end

% 40 hörn
points = package(-4, 1, 0.6, 40);
[T, XY] = calcpoints(t, points, @delta);
areacurve40 = zeros(size(t));
for i=1:length(t)
    areacurve40(i) = polygonarea(XY(1, i, :), XY(2, i, :));
end

% 80 hörn
points = package(-4, 1, 0.6, 80);
[T, XY] = calcpoints(t, points, @delta);
areacurve80 = zeros(size(t));
for i=1:length(t)
    areacurve80(i) = polygonarea(XY(1, i, :), XY(2, i, :));
end

hold off
subplot(2, 2, 1)
plot(XY(1, :, 1), areacurve40, 'g')
hold on
plot(XY(1, :, 1), areacurve20, 'r')
area = extrapolate(areacurve40, areacurve20);
areabetter = extrapolate(areacurve80, areacurve40);
```

```
plot(XY(1, :, 1), area)
subplot(2, 2, 2)
hold on
plot(XY(1, :, 1), area)
plot(XY(1, :, 1), areabetter, 'r')
```

**delta.m**

```
function res = delta(t, xy)
% Definierar differentialekvationen
R=2;
x = xy(1);
y = xy(2);
res = [1-(R^2*(x^2 - y^2)/(x^2 + y^2)^2); (-2*x*y*R^2)/(x^2 + y^2)^2];
```

**calcpoints.m**

```
function [T, AllXY] = calcpoints( t, points, func )
% Löser differentialekvationen func för ett givet intervall t och startpunkter
points.
AllXY = zeros(2, length(t), length(points));
i=1;
for point=points'
    [T, XY] = ode45(func, t, [point(1) point(2)]);
    AllXY(1:2, 1:length(t), i) = XY';
    i = i+1;
end
end
```

**package.m**

```
function res = package(x, y, r, n)
% Genererar ett paket av n partiklar i en cirkel runt punkten (x, y)
org_angle = 2*pi/n;
res = zeros(n, 2);
for i=[1:n]
    angle = org_angle*i;
    res(i, :) = [x+cos(angle)*r, y+sin(angle)*r]; % Beräkna punktkoordinater i
cirkeln. Ersätt med [x+cos(angle)*r y+sin(angle)*r/(i/2)] för resultatet i fig5
och fig6
end
```

**polygonarea.m**

```
function parea = polygonarea(x, y)
global globalvariabledonotuse;

% Omvandlar x och y till kolumnvektorer (1*1*n matriser tidigare)
x = x(:);
y = y(:);

x = [x(end); x; x(1)];
y = [y(end); y; y(1)];

parea = abs(sum(x(2:end-1).*(y(3:end)-y(1:end-2))))/2;

% Ritar var 40:de polygon
if mod(globalvariabledonotuse, 40) == 0
    plot(x, y, 'r')
end
globalvariabledonotuse = globalvariabledonotuse + 1;
```

**evtfun.m**

```
function [value, isterminal, direction] = evtfun(t, xy)
    % Avbryt ode45 när x=7.85 (antagligen överdriven precision här)
```

```
    value = 7.851531824785027-xy(1);
    isterminal = 1;
    direction = 0;
end
```

**extrapolate.m**

```
function res = extrapolate(val1, val2)
% val1 är uppskatning med hälften så lång steglängd som val2
res = val1 + (val1-val2)/3;
```

## Appendix B: Figurer

Fig1

Lösning av ekvationen för y0 = [0.2, 0.6, 1, 1.4]

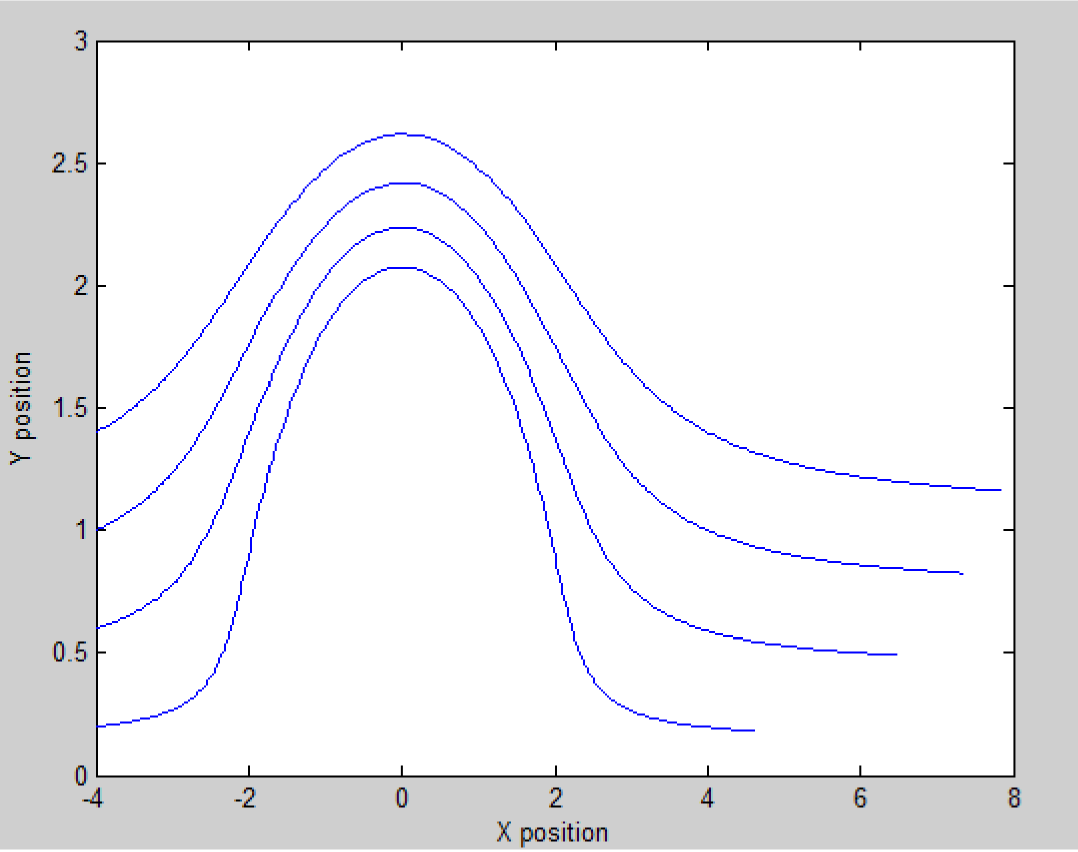



Fig2

Polygonernas area med avseende på x

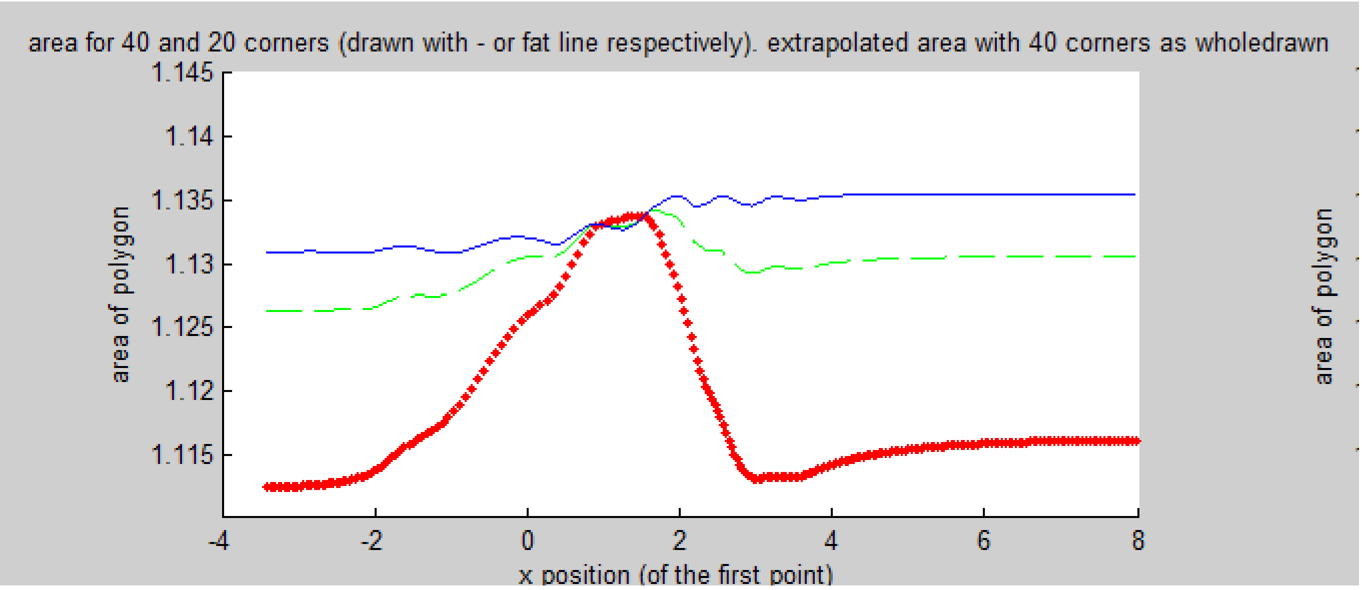
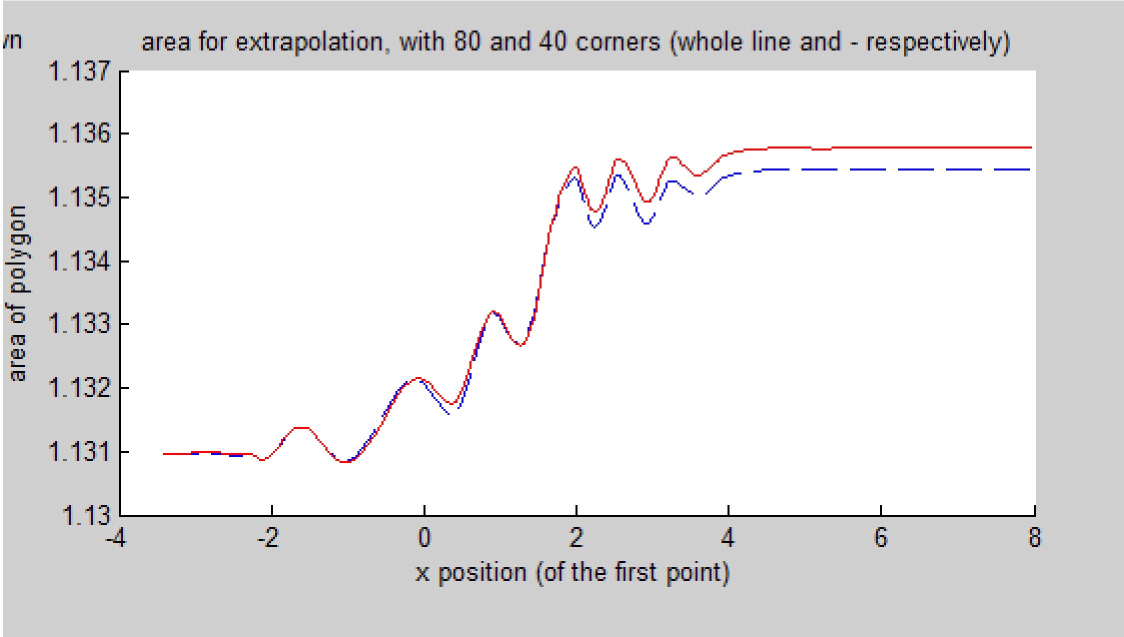
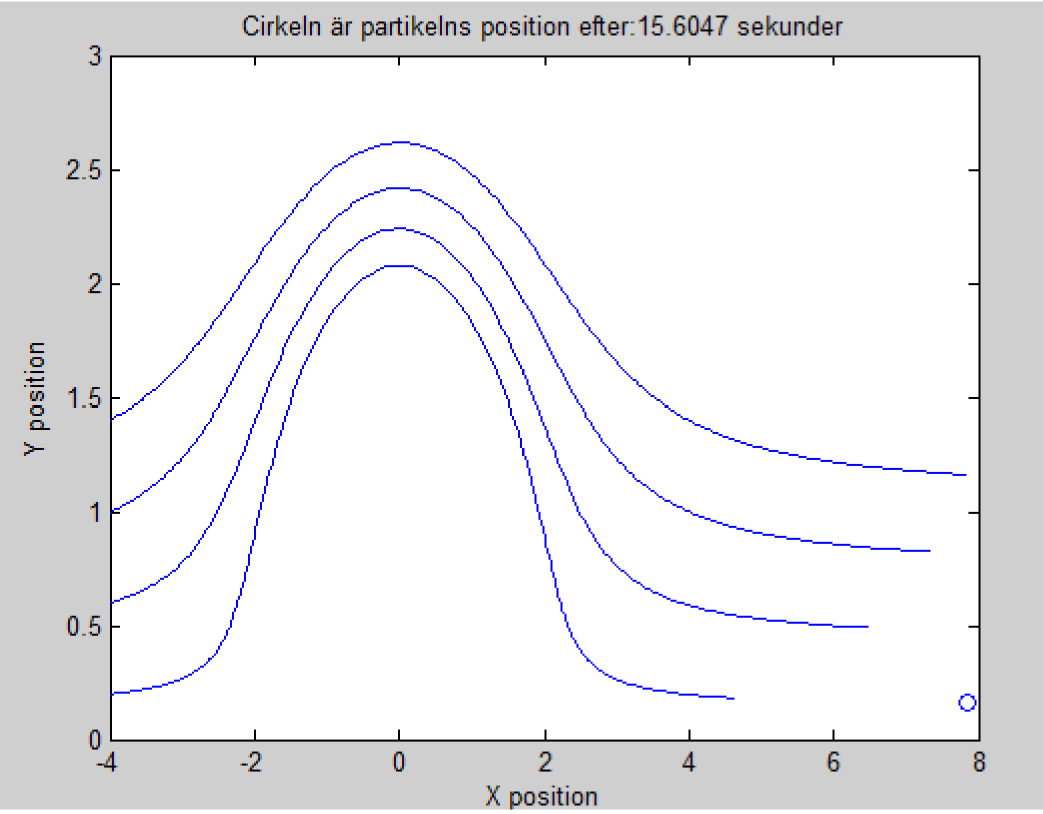


Fig3



/n
area for extrapolation, with 80 and 40 corners (whole line and - respectively)
1.137
1.136
1.135
1.134
1.133
1.132
1.131
1.13
area of polygon
-4
-2
0
2
4
6
8
x position (of the first point)


Fig4



Cirkeln är partikelns position efter:15.6047 sekunder
3
2.5
2
1.5
1
0.5
0
Y position
-4
-2
0
2
4
6
8
X position

Fig5
Polygonets deformering

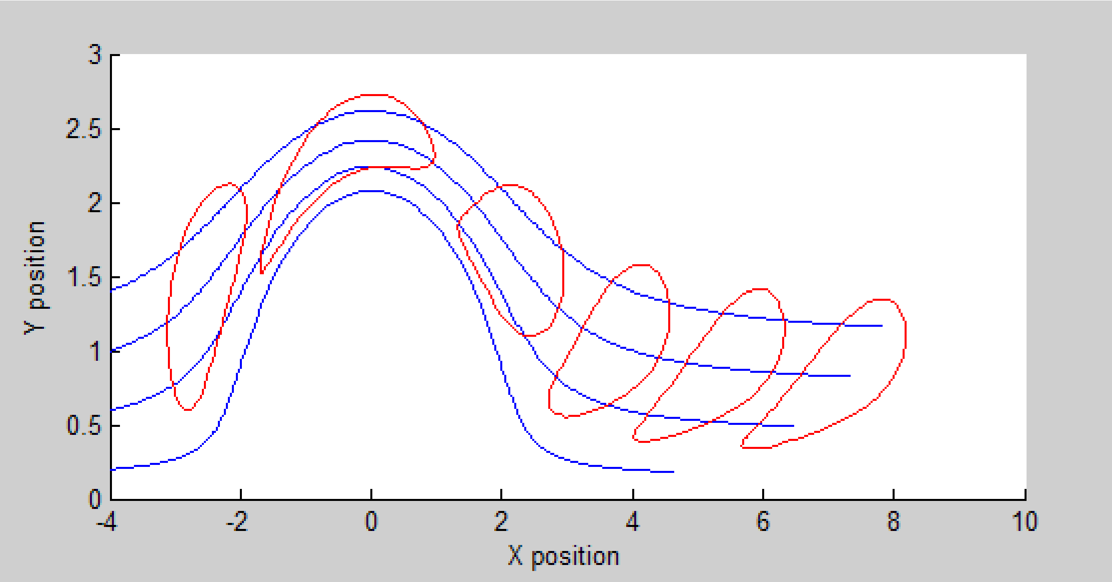



Fig6
Inzoomning på polygonen

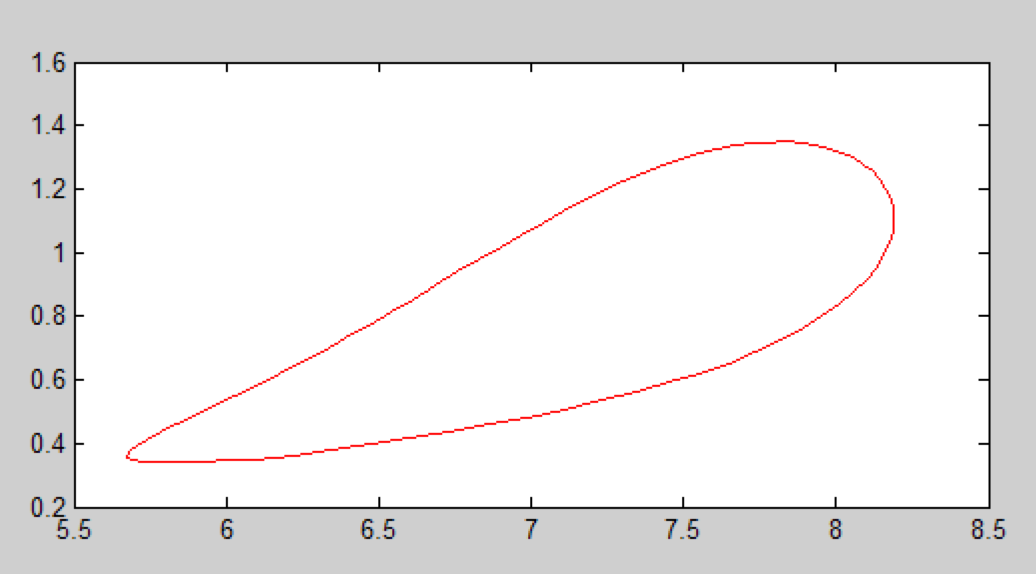



Fig7
Approximerad lösning för del 2

```
ans =

   15.604680365462242
```

Fig8
Den andra polygonens höjd beroende av x

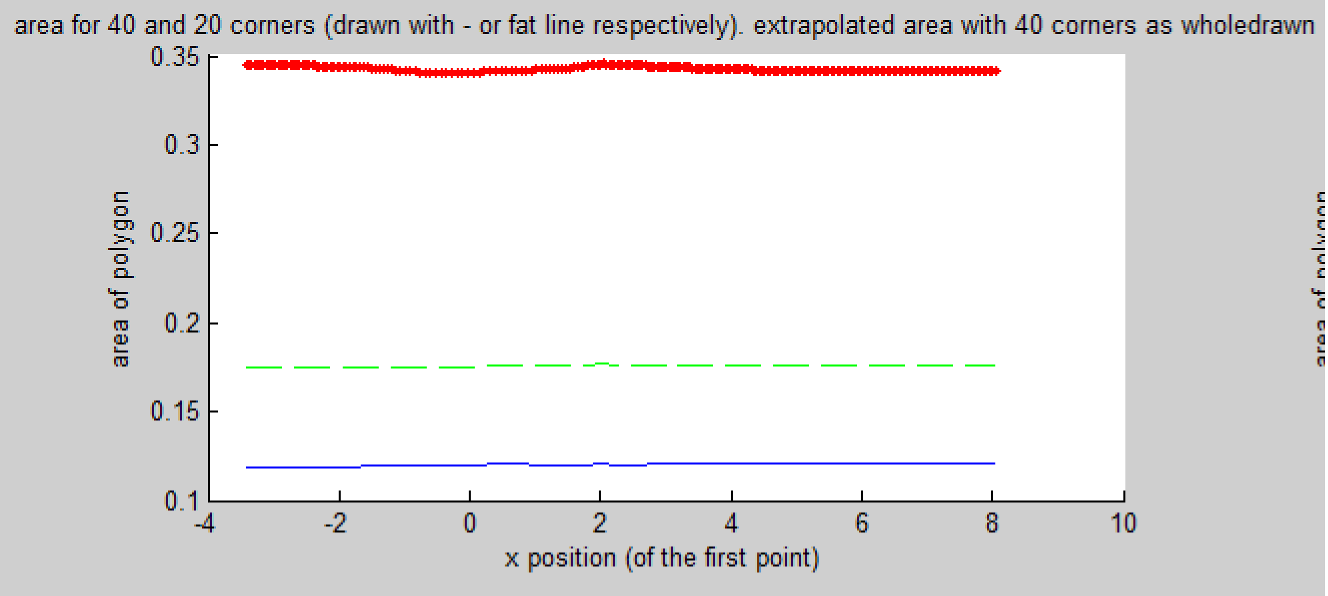



Fig9
Den andra polygonens höjd, extrapolerad

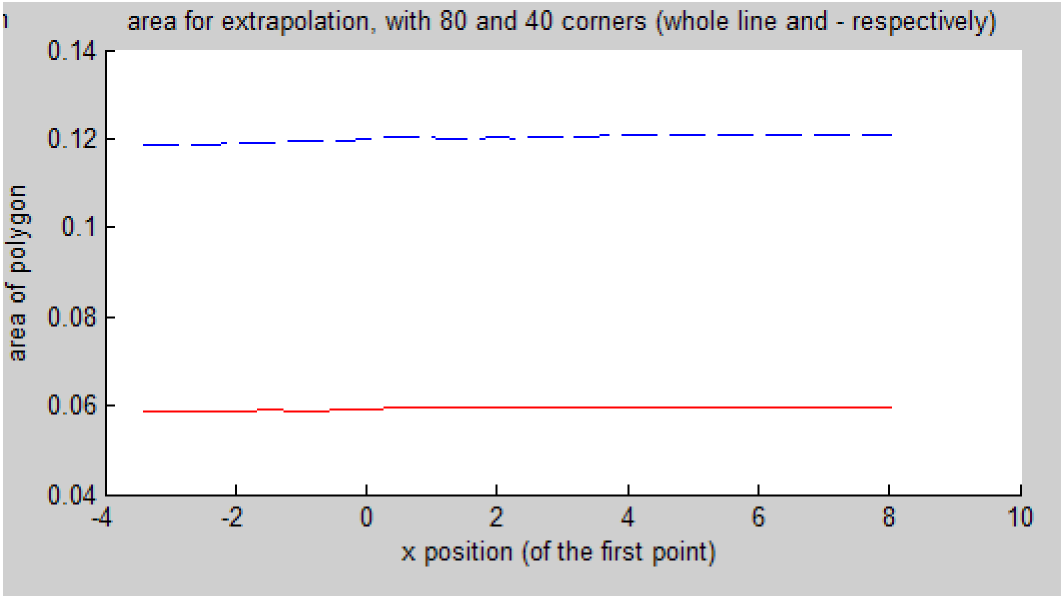



Fig10
Den andra polygonens deformering

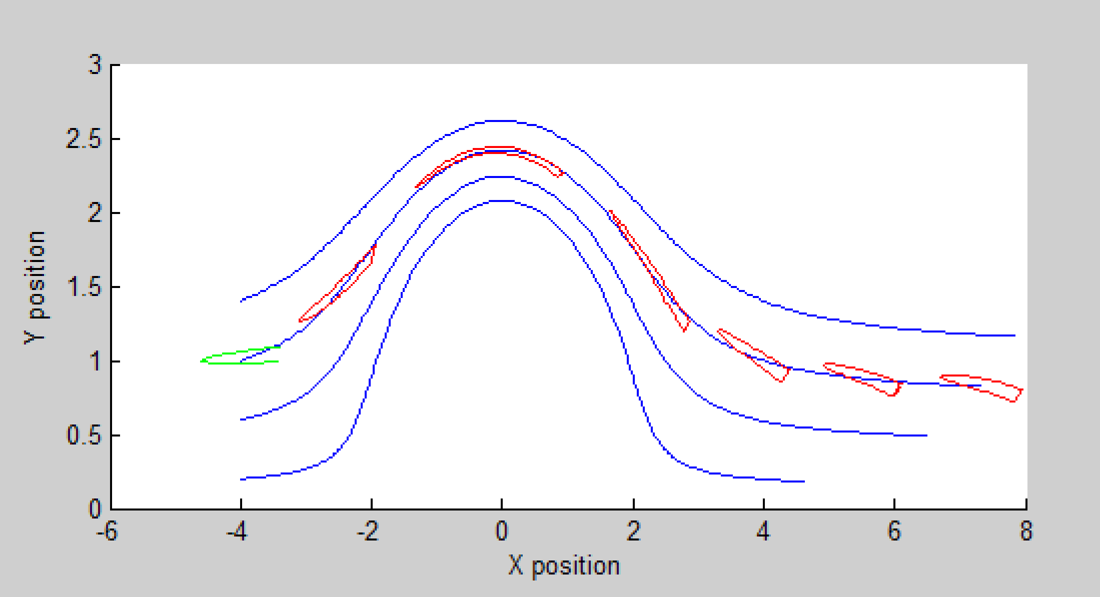



Fig11
Inzoomning på polygonerna i plot 3 i Fig5