Wetenschappelijk Rekenen

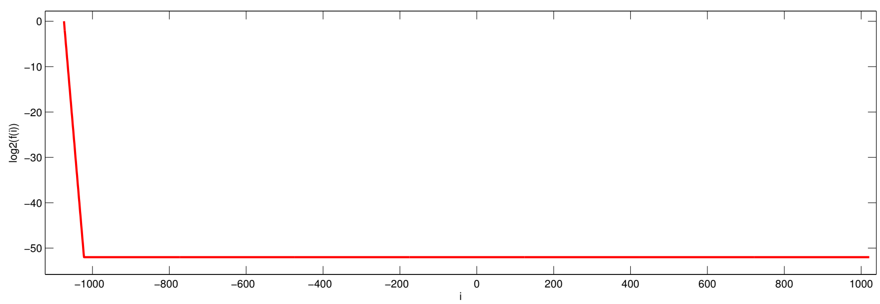
Examen - Bacheloropleiding informatica

Oefeningen – 10 juni 2014

1. In de oefeninglessen hebben we gezien dat we de machine-epsilon bekomen bij het berekenen van $|3(\frac{4}{3} - 1) - 1|$. Beschouw de volgende functie:

$$f(i) = \left| \frac{3\,(\frac{4\,.\,2^i}{3} - 2^i) - 2^i}{2^i} \right|$$

waarbij i een geheel getal is. Deze functie is geïmplementeerd in het bestand `vraag1.m` en is te vinden op Indianio. Wanneer we bijvoorbeeld $f(0)$, $f(10)$ of $f(-20)$ evalueren bekomen we opnieuw de machine-epsilon, maar is dit het geval voor elke $i \in \mathbb{Z}$? De volgende grafiek toont alle eindige waarden van $f(i)$, na het nemen van de `log2`:



Verklaar het verloop van de grafiek als i de verzameling der gehele getallen doorloopt. Maak hierbij gebruik van IEEE-standaarden.

Oplossing: De breuk $\frac{4}{3}$ is niet exact voorstelbaar en er treedt een afrondingsfout op. Aangezien de waarden hier dicht bij 1 liggen komt de relatieve fout en de absolute fout overeen bij het berekenen van $|3(\frac{4}{3} - 1) - 1|$. Deze fout is gelijk aan de machine-epsilon.

Indien we vermenigvuldigen met 2^i komen de relatieve fout en absolute fout niet meer overeen (tenzij voor $i = 0$). Bijvoorbeeld voor $i = 100$:

```
>> x = abs((3.*(4.*(2.^100)/3-2.^100)-2.^100))
x =
     2.814749767106560e+14
```

Dit is een absolute fout, indien we delen door 2^{100} bekomen we de relatieve fout:

```
>> x / 2^100
ans =
     2.220446049250313e-16
```

Dit is opnieuw de machine-epsilon. De waarde van de machine-epsilon is 2^{-52}, wat overeenstemt met 52 beschikbare bits voor precisie. Dit verklaart waarom `log2(f(i))` gelijk is aan `-52` voor het grootste deel van de eindige waarden.

Opmerking: de waarde van de machine-epsilon verschilt afhankelijk van welke definitie men toepast. Volgens de definitie in MATLAB is de machine-epsilon gelijk aan 2^{-52}. In de cursus gebruikt men een andere definitie en is de machine-epsilon gelijk aan $\frac{2^{-52}}{2} = 2^{-53}$.

Wanneer i gelijk is aan 1022 en 1023, dan zal bij de berekening van $4 \cdot 2^i$ het OFL bereikt worden want $4 \cdot 2^i \geq 2^{1024}$. We bekomen een `Inf` in de teller en als gevolg zal `f(1022)` en `f(1023)` beide `Inf` opleveren. Wanneer i hoger is dan 1024 zal er in de teller en noemer een `Inf` voorkomen vanwege het overschrijden van OFL en krijgen we `NaN` als resultaat.

Het kleinste genormaliseerd getal is 2^{-1022}. Hier is `log2(f(-1022))` gelijk aan -52. Waarden kleiner dan 2^{-1022} en groter dan nul zijn subnormale getallen. Deze getallen zijn niet genormaliseerd en hebben leidende cijfers die nul zijn. Dit heeft als gevolg dat deze getallen minder precisie hebben. Hoe meer leidende nullen, hoe minder bits er beschikbaar zijn voor precisie.

Bijvoorbeeld voor $i = -1060$:

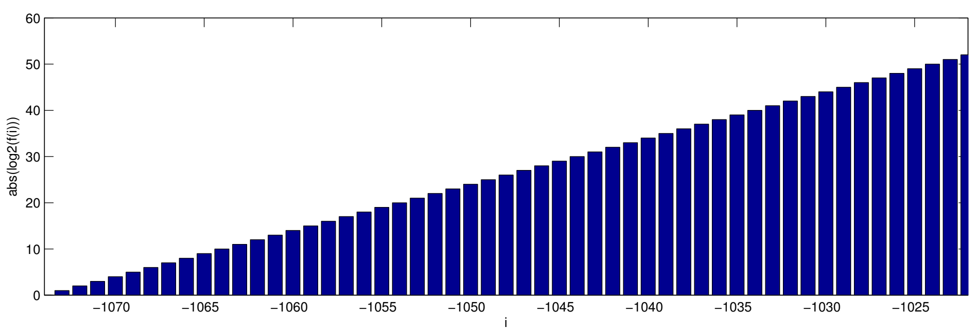
$$\mathsf{fl}(2^{-1060}) = \underbrace{000000000000}_{\text{teken en exponent}} \underbrace{0000000000000000000000000000000000000100000000000000}_{\text{mantisse}}$$

We berekenen de teller van f:

$$\mathsf{fl}\left(\left|3\,(\frac{4\,.\,2^i}{3} - 2^i) - 2^i\right|\right) = \underbrace{000000000000}_{\text{teken en exponent}} \underbrace{0000000000000000000000000000000000000000000000000001}_{\text{mantisse}}$$

Dit is een absolute afrondingsfout, we delen door 2^{-1060} en we bekomen de relatieve afrondingsfout: 2^{-14}. Dit correspondeert met de 14 bits precisie die we overhouden bij dit subnormaal getal in plaats van 52 bits precisie bij normale getallen.

Voor elke leidende nul in de mantisse verliezen we een bit precisie.



Dit verklaart de stijgende lijn in de grafiek uit de opgave aan de linkerkant. Voor $i = -1074$ is $f(i) = 1 = 2^0$ en hebben we nul bits precisie. Merk op dat $-1022 - 52 = -1074$. Het getal 2^{-1074} is het kleinste positieve machinegetal. Voor $i < -1074$ bekomen we `NaN` voor $f(i)$ vanwege $\frac{0}{0}$.

2. Gegeven, enkele punten van de functie f:

x	-2	-0.6	0.8	2.2	3.6	5
$f(x)$	15.9	3.7	7	4	-4.4	-5.2

Benader f met een functie $g(x) = a\,e^{-x} + b\,\sin(x) + c\,\cos(x)$. Bepaal a, b en c zodat de benadering $g(x)$ van $f(x)$ optimaal is volgens het principe der kleinste kwadraten. Plot vervolgens de datapunten van f en de benadering g in het interval $[-2, 5]$.

Oplossing:

We passen data-fitting toe. De drie basisfuncties zijn e^{-x}, $\sin(x)$ en $\cos(x)$.

Er zijn 6 datapunten die ons 6 vergelijkingen opleveren:

$$
\begin{aligned}
a\,e^{2} + b\,\sin(-2) + c\,\cos(-2) &= 15.9\\
a\,e^{0.6} + b\,\sin(-0.6) + c\,\cos(-0.6) &= 3.7\\
a\,e^{-0.8} + b\,\sin(0.8) + c\,\cos(0.8) &= 7\\
a\,e^{-2.2} + b\,\sin(2.2) + c\,\cos(2.2) &= 4\\
a\,e^{-3.6} + b\,\sin(3.6) + c\,\cos(3.6) &= -4.4\\
a\,e^{-5} + b\,\sin(5) + c\,\cos(5) &= -5.2
\end{aligned}
$$

In matrixnotatie is dit

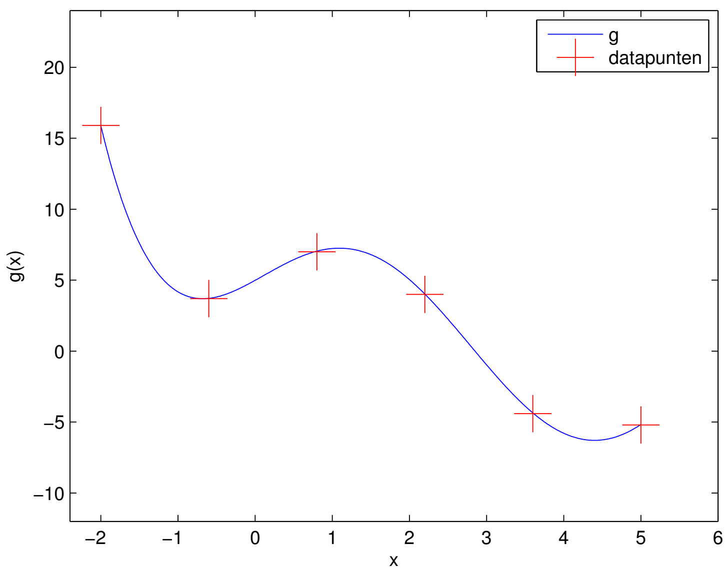
$$
\begin{bmatrix}
e^{2} & \sin(-2) & \cos(-2)\\
e^{0.6} & \sin(-0.6) & \cos(-0.6)\\
e^{-0.8} & \sin(0.8) & \cos(0.8)\\
e^{-2.2} & \sin(2.2) & \cos(2.2)\\
e^{-3.6} & \sin(3.6) & \cos(3.6)\\
e^{-5} & \sin(5) & \cos(5)
\end{bmatrix}
\begin{bmatrix} a\\ b\\ c \end{bmatrix}
=
\begin{bmatrix} 15.9\\ 3.7\\ 7\\ 4\\ -4.4\\ -5.2 \end{bmatrix}.
$$

Hierop passen we kleinste kwadraten toe in Matlab:

```
>> x = [-2 -0.6 0.8 2.2 3.6 5]';
>> y = [15.9 3.7 7 4 -4.4 -5.2]';
>> A = [exp(-x) , sin(x) , cos(x)]
A =
   7.389056098930650 -0.909297426825682 -0.416146836547142
   1.822118800390509 -0.564642473395035 0.825335614909678
   0.449328964117222 0.717356090899523 0.696706709347165
   0.110803158362334 0.808496403819590 -0.588501117255346
   0.027323722447293 -0.442520443294852 -0.896758416334147
   0.006737946999085 -0.958924274663138 0.283662185463226

>> coef = A \ y
coef =
   3.001790011604405
   6.006140110917550
   1.985037158083168

>> %alternatief:
>> [Q,R] = qr(A); coef = R \ (Q'*y);
```

Dus $a = 3.001790011604405$, $b = 6.006140110917550$ en $c = 1.985037158083168$.
We plotten g in het interval $[-3, 5]$, samen met de datapunten:

```
g = @(t) coef(1).*exp(-t) + coef(2).*sin(t) + coef(3).*cos(t);
xx = linspace(-2,5,1000);
plot(xx, g(xx));
hold on;
plot(x,y , 'Marker' , '+' , 'linestyle' , 'none' , 'color' , 'red' , 'Markersize' , 20);
axis(1.2 * axis());
xlabel('x'); ylabel('g(x)');
legend('g' , 'datapunten');
```



3. (a) Bepaal de coëfficiënten van de kwadratuurformule

$$\int_{-4}^{6} f(x)\,dx \approx Q(f) = w_1 f\left(1-\sqrt{15}\right) + w_2 f(\beta) + w_3 f\left(1+\sqrt{15}\right)$$

zodat de graad wordt gemaximaliseerd.

(b) Bepaal de constanten p en α in de uitdrukking

$$\int_{-4}^{6} f(x)\,dx = Q(f) + \alpha f^{(p)}(\xi)$$

met $\xi \in [-4, 6]$. Wat is de graad van Q ?

Oplossing:

(a) Er zijn 4 coëfficiënten om te bepalen, dus we leggen 4 vergelijkingen op:

```
Q := f -> w[1]*f(1-sqrt(15))+w[2]*f(beta)+w[3]*f(1+sqrt(15)):
solve({int(1, x = -4 .. 6) = Q(x->1),
       int(x, x = -4 .. 6) = Q(x->x),
       int(x^2, x = -4 .. 6) = Q(x->x^2),
       int(x^3, x = -4 .. 6) = Q(x -> x^3)});
```

$$\left\{\beta = 1, w_1 = \frac{25}{9}, w_2 = \frac{40}{9}, w_3 = \frac{25}{9}\right\}$$

```
assign(%);
```

$$Q(f) = \frac{1}{9}\left(25 f\left(1-\sqrt{15}\right) + 40 f(1) + 25 f\left(1+\sqrt{15}\right)\right)$$

(b) We bepalen de graad van Q. De graad is minstens 3. We gaan na of de graad hoger is:

```
simplify(int(x^4, x = -4 .. 6) = Q(x -> x^4));
```

$$1760 = 1760$$

```
simplify(int(x^5, x = -4 .. 6) = Q(x -> x^5));
```

$$\frac{21280}{3} = \frac{21280}{3}$$

```
simplify(int(x^6, x = -4 .. 6) = Q(x -> x^6));
```

$$\frac{296320}{7} \neq 38760$$

De graad is 5. De p in de foutterm is dus gelijk aan 6. We bepalen α door de f in

$$\int_{-4}^{6} f(x)\,dx = Q(f) + \alpha f^{(6)}(\xi)$$

in te vullen met $f(x) = x^6$.

$$\frac{296320}{7} = 38760 + \alpha\, 6!$$

$$\alpha = \frac{625}{126}$$

Besluit:

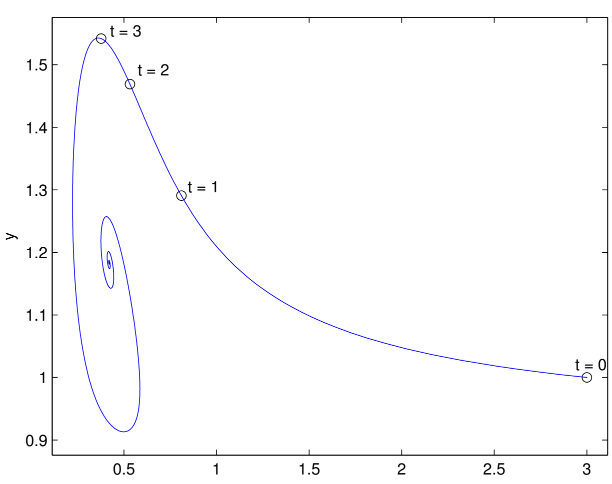
$$\int_{-4}^{6} f(x)\,dx = Q(f) + \frac{625}{126} f^{(6)}(\xi)$$

Zie het bestand `vraag3.maple.txt`.

4. Gegeven is het beginwaardeprobleem

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 2x(1-x) - \frac{xy}{x+0.6} \\ -\frac{y}{4} + \frac{xy}{x+0.6} - \frac{y^2}{5(y^2+\frac{1}{16})} \end{bmatrix}, \quad \begin{bmatrix} x(0) \\ y(0) \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \end{bmatrix}.$$

Naarmate de tijd t toeneemt zal $\begin{bmatrix} x(t) \\ y(t) \end{bmatrix}$ convergeren naar een bepaald punt. Zie de figuur.



Dit punt noemt men een stabiel evenwichtspunt. Ga hierbij als volgt te werk om dit evenwichtspunt te berekenen:

(a) Gebruik `ode45` om vanuit de beginwaarde een schatting te bekomen van dit evenwichtspunt. Bereken de waarde waarbij $t = 50$ en gebruik dit in 4b.

(b) Maak gebruik van de schatting uit 4a om de waarde van dit evenwichtspunt te verbeteren. Hierbij kan je gebruik maken van de eigenschap dat de afgeleide van een functie naar de tijd in een evenwichtstpunt nul is. Bereken het punt nauwkeurig tot minstens 13 cijfers na de komma.

Oplossing: We beginnen met de differentiaalvergelijking te implementeren als de functie f. Deze functie is van de vorm $\mathbf{y}' = f(t, \mathbf{y})$.

$$f\left(t, \begin{bmatrix} x \\ y \end{bmatrix}\right) = \begin{bmatrix} 2x(1-x) - \frac{xy}{x+0.6} \\ -\frac{y}{4} + \frac{xy}{x+0.6} - \frac{y^2}{5(y^2+\frac{1}{16})} \end{bmatrix}$$

Voor stap (a) berekenen we de schatting als volgt:

```
[t,Y] = ode45(@f , [0,50] , [3;1] );
```

De laatste waarde van de vector `t` is 50, we vinden de schatting op de laatste lijn van `Y`.

```
schatting = Y(end,:)'
```

```
schatting =
   0.419404975628702
   1.181531848199056
```

Deze waarde voor het evenwichtspunt gaan we nu verbeteren door de methode van Newton toe te passen. In het evenwichtspunt $[x_e,\, y_e]^T$ geldt dat de afgeleide naar de tijd nul is. Dus

$$\begin{bmatrix} 2x_e(1-x_e) - \frac{x_e y_e}{x_e+0.6} \\ -\frac{y_e}{4} + \frac{x_e y_e}{x_e+0.6} - \frac{y_e^2}{5(y_e^2+\frac{1}{16})} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

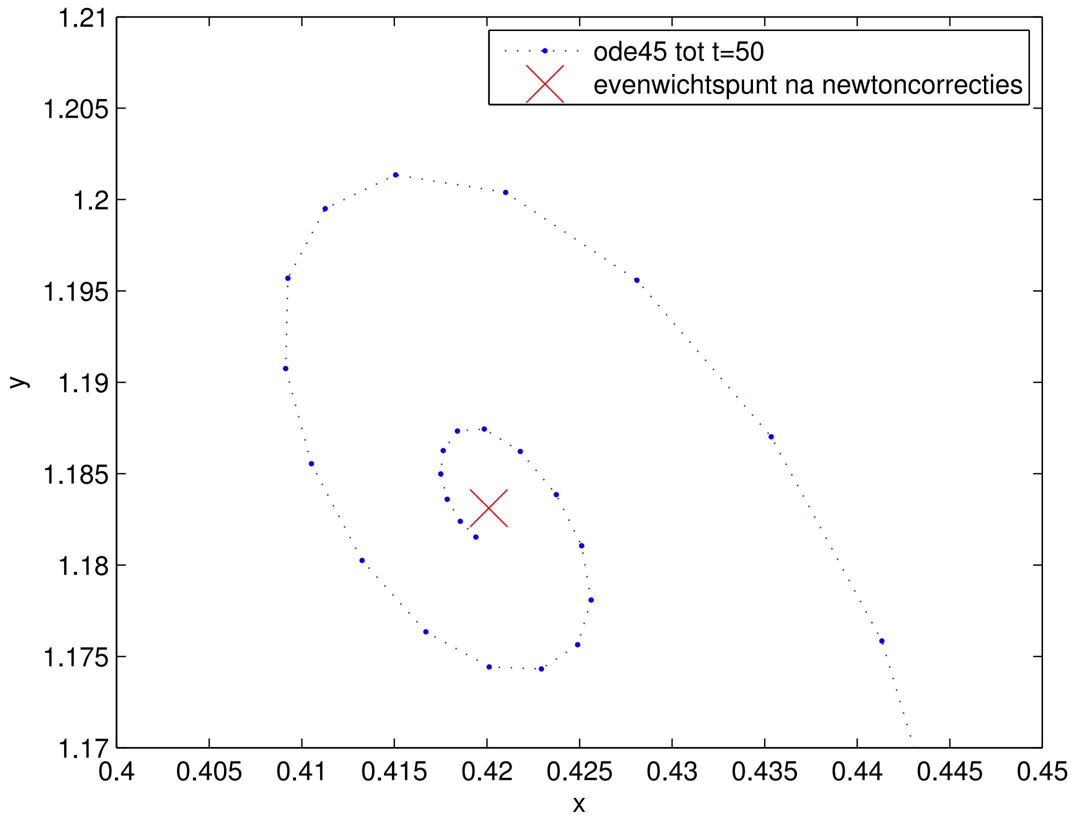
Het evenwichtspunt vinden we als een nulpunt van f. Merk op dat f onafhankelijk is van t. We steunen op het bestand *NewtonMethode.m* uit de oefeningles van hoofdstuk 5 (opgenomen in het bestand *vraag4.m*).

```
evenwichtspunt = NewtonMethode(@(x) f(-1,x) , @(x) Jf(x), schatting)
```

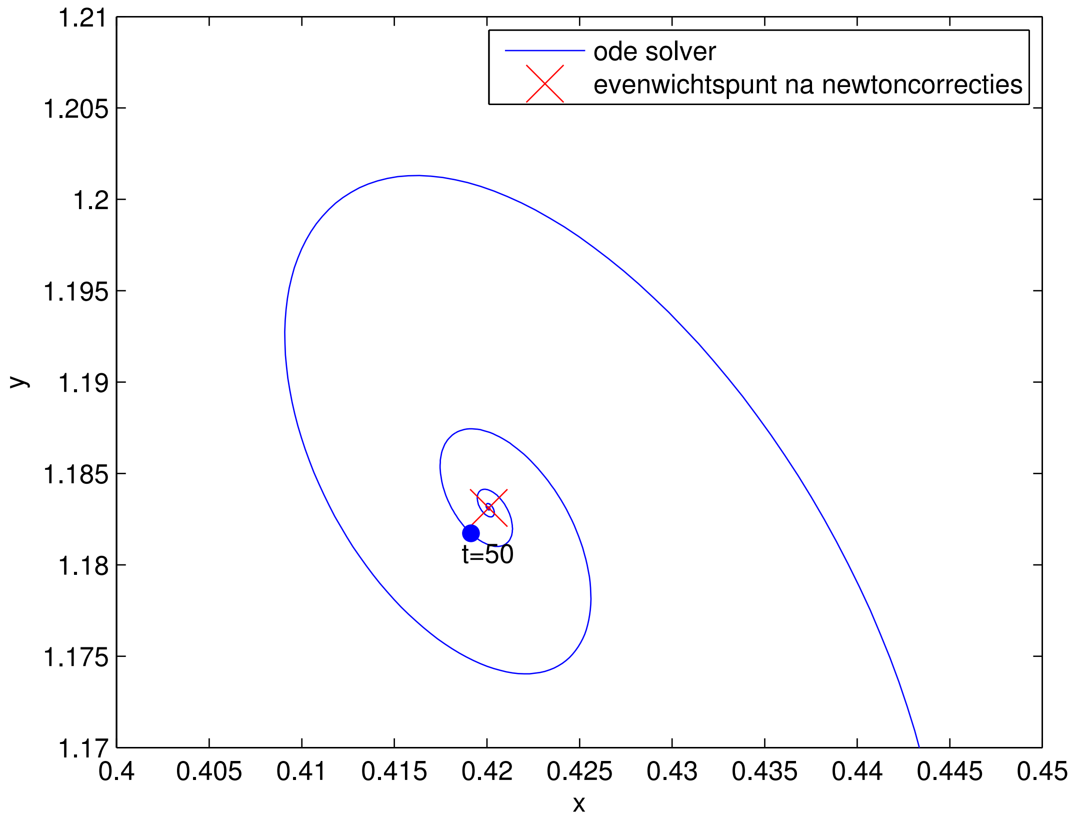
```
Correcties via Newton's Methode:

it      deltanorm           f(X)
  1  1.58724678e-03  1.26692574e-06
  2  3.23584104e-06  2.94375635e-13
  3  5.85763256e-13  5.55111512e-17
  4  1.22339794e-16  5.55111512e-17

evenwichtspunt =
   0.420095593848104
   1.183115859137302
```

Aangezien f onafhankelijk is van t vullen we voor t een willekeurige waarde in. De jacobiaan `Jf` hebben we berekend met de hulp van Maple. Als stopvoorwaarde gebruiken we $\|\delta\|_\infty < 10^{-13}$, met δ de correctie in een Newtoniteratiestap.

Het resultaat van stap (a) en stap (b) is te zien op de volgende figuur.



Ter verificatie kan men een *ODE solver* toepassen met hogere eindwaarden van t om de convergentie naar het berekende punt te bevestigen:



De volgende code, ook terug te vinden in het bestand *vraag4.m*, lost de volledige oefening op:

```
function vraag4
    [t,Y] = ode45(@f , [0,50] , [3;1] );

    plot(Y(:,1) , Y(:,2) , 'linestyle' , ':' , 'Marker' , '.');
    hold on; xlabel('x'); ylabel('y');

    format long;
    schatting = Y(end,:)'
    evenwichtspunt = NewtonMethode(@(x) f(-1,x) , @(x) Jf(x), schatting)

    plot(evenwichtspunt(1) , evenwichtspunt(2) , 'Marker' , 'X' , 'linestyle' , 'none' , 'color' ,

    axis([0.40 0.45 1.17 1.21])
    legend('ode45␣tot␣t=50' , 'evenwichtspunt␣na␣newtoncorrecties');
end


function Y = f(t,Y)
    x = Y(1);
    y = Y(2);
    Y(1) = 2*x*(1-x) - (x*y)/(x+0.6);
    Y(2) = -y/4 + (x*y)/(x+0.6) - (0.2*y*y)/(y*y + 1/16);
```

```
end
function jac = Jf(Y)
    jac = zeros(2,2);
    x = Y(1);
    y = Y(2);
    jac(1,1) = 2 - 4*x - y/(x + 0.6) + (x*y)/(x+0.6)^2;
    jac(1,2) = - x / (x+ 0.6);
    jac(2,1) = y/(x+0.6) - (x*y)/(x+ 0.6)^2;
    jac(2,2) = -0.25 + x/(x+0.6) - 0.4*y / (y^2 + 1/16) + (0.4*y^3) / (y^2 + 1/16)^2;

end

function [X,k] = NewtonMethode(F, Jf, X)
    deltaTol = 10^-13;
    maxIt = 50;
    k = 0;
    delta=Inf;
    Fx = F(X);

    disp('it␣␣␣␣␣␣deltanorm␣␣␣␣␣␣␣␣␣␣␣f(X)␣␣');
    while (delta > deltaTol) && (k <= maxIt)
        delta = (Jf(X) \ -Fx);
        X = X + delta;
        Fx = F(X);
        delta = norm(delta,Inf);
        k = k+1;
        fprintf('%3i␣%15.8e␣%15.8e␣␣␣\n' ,k , delta, norm(Fx,Inf));
    end
end
```