

Maple 8 en g.d.v.

(gewone differentiaalvergelijkingen)

G. Sweers, maart 2003

Als je Maple niet vers start is het verstandig om met de grote schoonmaak te beginnen:

[> **restart;**

Voordat we verder gaan: vergeet niet dat Maple een uitgebreide **help**-voorziening heeft die via de balk te bereiken is.

Ook de cursor op een woord plaatsen en CTRL-F1 toetsen levert hulp bij het betreffende woord indien aanwezig.

[*Test dit door de cursor op **odeadvisor** te plaatsen en CTRL-F1 te typen.*

Het onderstaande is getest voor Maple 8. Het ode-gedeelte van bijvoorbeeld Maple 6 of oudere versies is deels noch forward, noch backward compatibel met Maple 8.

+ Gereedschap

+ Expliciete oplossingen

+ Functies en grafieken

+ Direct een grafiek

+ Numerieke methoden

+ Vectorveld en fasevlak

- Gereedschap

Het belangrijkste commando om differentiaalvergelijkingen met Maple aan te pakken is de volgende. Hiemeer worden een aantal kant-en-klare recepten klaargezet uit de 'package' [DEtools](#). Een overzicht van alle [packages](#) is beschikbaar via help.

```
[ > with(DEtools) :
```

Om de namen van deze gereedschapskist te zien kunt u de : vervangen door ; .

Nu kunnen we aan de slag met gewone differentiaalvergelijkingen (= ode = ordinary differential equations).

Een voorbeeld:

```
[ > gdv := D(y)(x) = x*y(x) ;
```

$$gdv := D(y)(x) = x y(x)$$

- Meer voorbeelden

Voorbeeld 1. Een eerste orde gewone differentiaalvergelijking.

```
[ > gdv1 := diff(y(x), x) = y(x) + x ;
```

$$gdv1 := \frac{d}{dx} y(x) = y(x) + x$$

Voorbeeld 2. Een tweede orde gewone differentiaalvergelijking.

```
[ > gdv2 := (D@@2)(y)(x) = y(x) * exp(y(x)^2) ;
```

$$gdv2 := (D^{(2)})(y)(x) = y(x) e^{(y(x)^2)}$$

Voorbeeld 3. Een andere eerste orde gewone differentiaalvergelijking.

```
[ > gdv3 := D(x)(t) = x(t)^2 + t^2 ;
```

$$gdv3 := D(x)(t) = x(t)^2 + t^2$$

Voorbeeld 4. Nog een gewone differentiaalvergelijkingen.

```
[ > gdv4 := D(y)(t) = y(t)^3 - t^2 ;
```

$$gdv4 := D(y)(t) = y(t)^3 - t^2$$

Voorbeeld 5. Een stelsel gewone differentiaalvergelijkingen.

```
[ > gdvstelsel := {D(u)(t) = v(t) + t, D(v)(t) = u(t) - v(t)} ;
```

$$gdvstelsel := \{D(u)(t) = v(t) + t, D(v)(t) = u(t) - v(t)\}$$

- Expliciete oplossingen

Om expliciete oplossingen van gewone differentiaalvergelijkingen te zoeken gebruiken we het commande `dsolve`. Een groot aantal opties is daarbij mogelijk maar voorlopig is het voldoende om het volgende te weten.

Gekopieerd uit help en geschreven voor g.d.v. voor y als functie van x:

dsolve - Solve ordinary differential equations (ODEs)

Calling Sequences

```
dsolve(ODE)
dsolve(ODE, y(x), extra_args)
dsolve({ODE, ICs}, y(x), extra_args)
dsolve({sysODE, ICs}, {funcs}, extra_args)
```

Parameters

ODE - an ordinary differential equation
y(x) - any indeterminate function of one variable
ICs - initial conditions
{sysODE} - a set with a system of ODEs
{funcs} - a set with indeterminate functions
extra_args - optional, depends on the type of problem being solved (see below)

* Herein {x, y(x)} represent any pair of independent and dependent variables.

De oplossingen van bovenstaande gdv vinden we via:

```
> dsolve(gdv);
```

$$y(x) = _CI e^{\left(\frac{x^2}{2}\right)}$$

En als we geïnteresseerd zijn in een bepaalde beginvoorwaarde:

```
> dsolve({gdv, y(0)=1}, y(x));
```

$$y(x) = e^{\left(\frac{x^2}{2}\right)}$$

- Meer voorbeelden

Voorbeeld 1a Voor de eerste extra differentiaalvergelijking.

```
> dsolve(gdv1);
```

$$y(x) = -x - 1 + e^x _CI$$

Voorbeeld 1b Voor de eerste met een extra beginvoorwaarde:

```
> dsolve({gdv1, y(0)=3}, y(x));
```

$$y(x) = -x - 1 + 4 e^x$$

Voorbeeld 2 Voor de bovenstaande tweede orde gewone differentiaalvergelijking.

```
> dsolve({gdv2, y(0)=0, D(y)(0)=1}, y(x));
```

$$y(x) = \text{RootOf} \left(\int_0^x \frac{1}{\sqrt{e^{(f^2)}}} d_f - x \right)$$

[Maple 8 levert een vreemde formule. Wat stelt deze uitdrukking voor? Maple 5 geeft overigens geen antwoord.

Voorbeeld 3. De eerste orde gewone differentiaalvergelijking uit gdv3.

[> `dsolve({gdv3, x(0)=0}, x(t));`

$$x(t) = -\frac{t \left(-\text{BesselJ}\left(\frac{-3}{4}, \frac{t^2}{2}\right) + \text{BesselY}\left(\frac{-3}{4}, \frac{t^2}{2}\right) \right)}{-\text{BesselJ}\left(\frac{1}{4}, \frac{t^2}{2}\right) + \text{BesselY}\left(\frac{1}{4}, \frac{t^2}{2}\right)}$$

Voorbeeld 4. En voor gdv4, een eerste orde gewone differentiaalvergelijking.

[> `dsolve(gdv4);`

[*Waarom gebeurt hier niets?*

Voorbeeld 5. Het stelsel gewone differentiaalvergelijkingen achtereenvolgens zonder en met randvoorwaarden levert de volgende oplossingen.

[> `dsolve(gdvtstelsel);`

$$\begin{aligned} \{u(t) = e^{\left(\frac{\sqrt{5}-1}{2}\right)t} C_2 + e^{\left(-\frac{\sqrt{5}+1}{2}\right)t} C_1 - 2 - t, \\ v(t) = \left(\frac{\sqrt{5}}{2} - \frac{1}{2}\right) e^{\left(\frac{\sqrt{5}-1}{2}\right)t} C_2 + \left(-\frac{1}{2} - \frac{\sqrt{5}}{2}\right) e^{\left(-\frac{\sqrt{5}+1}{2}\right)t} C_1 - 1 - t\} \end{aligned}$$

[> `dsolve(gdvtstelsel union {v(0)=1, u(0)=1}, {u(t), v(t)});`

{v(t) =

$$\begin{aligned} \left(\frac{\sqrt{5}}{2} - \frac{1}{2}\right) e^{\left(\frac{\sqrt{5}-1}{2}\right)t} \left(\frac{7\sqrt{5}}{10} + \frac{3}{2}\right) + \left(-\frac{1}{2} - \frac{\sqrt{5}}{2}\right) e^{\left(-\frac{\sqrt{5}+1}{2}\right)t} \left(-\frac{7\sqrt{5}}{10} + \frac{3}{2}\right) - 1 - t, \\ u(t) = e^{\left(\frac{\sqrt{5}-1}{2}\right)t} \left(\frac{7\sqrt{5}}{10} + \frac{3}{2}\right) + e^{\left(-\frac{\sqrt{5}+1}{2}\right)t} \left(-\frac{7\sqrt{5}}{10} + \frac{3}{2}\right) - 2 - t\} \end{aligned}$$

[*Waarom werkt het volgende niet?*

[> `dsolve({gdvtstelsel, v(0)=1, u(0)=1}, {u(t), v(t)});`

Error, (in dsolve/IC) required a specification of the indeterminate functions

- Functies en grafieken

Nadat we met `dsolve` (voor sommige) differentiaalvergelijkingen via Maple een oplossingen in de vorm van een gesloten formule hebben gevonden zouden we deze formule verder willen gebruiken; bijvoorbeeld om een grafiek bij zo'n functie te laten tekenen.

plot - create a two-dimensional plot of functions

Calling Sequence

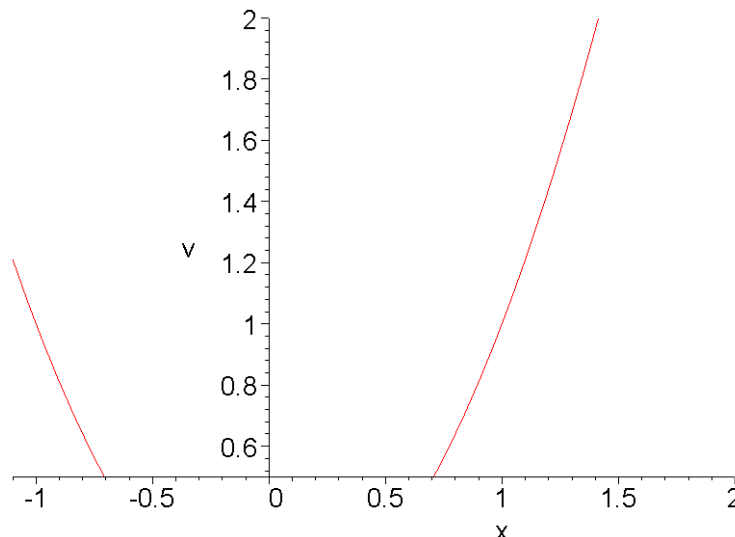
```
plot(f, h, v)
plot(f, h, v,...)
```

Parameters

f - function(s) to be plotted
h - horizontal range
v - (optional) vertical range

Een deel van een parabool vindt men door:

```
> plot(x^2, x=-1..2, v=.5..2);
```



Weliswaar hebben we hierboven enkele expliciete oplossingen van differentiaalvergelijkingen gevonden maar die zijn nog niet beschikbaar in de vorm van een functie. Een eerste stap daarvoor is het bewaren van de uitkomst:

```
> antwoord:=dsolve({gdv, y(0)=1}, y(x));
```

$$\text{antwoord} := y(x) = e^{\left(\frac{x^2}{2}\right)}$$

Vervolgens wat manipulaties om van deze uitdrukking tot een functie te komen:

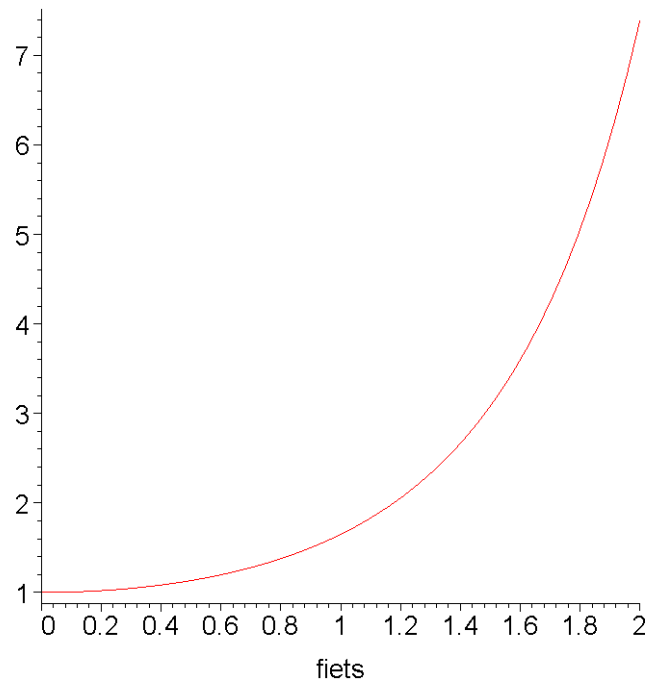
```
> Y:=unapply(subs(antwoord, y(x)), x);
```

$$Y := x \rightarrow e^{(1/2x^2)}$$

'subs(antwoord,y(x))' vertelt dat we y(x) vervangen door datgene wat in antwoord voor y(x) staat.

'unapply(f(x),x)' zegt dat het functievoorschrift f(x) vervangen wordt door een functie waar we ieder passend getal kunnen invullen. Die functie hebben we hier Y genoemd en kunnen we nu laten tekenen:

```
> plot(Y(fiets), fiets=0..2);
```

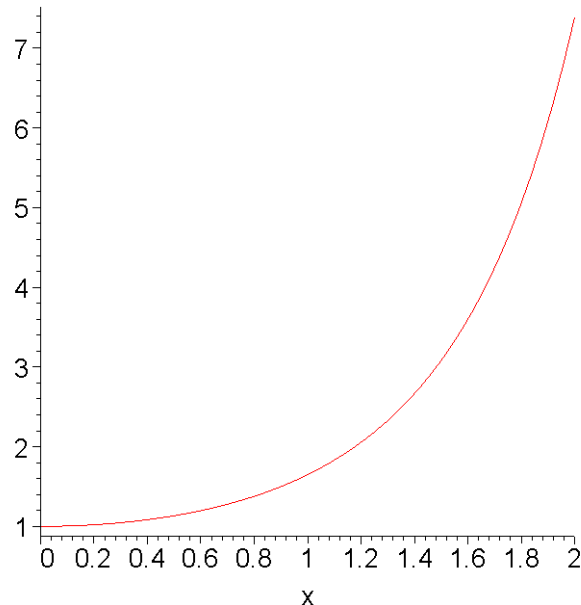


- Kan het simpeler?

Probeer ook eens direct $y(x)$ of $\text{subs}(\text{antwoord}, y(x))$ te laten tekenen.

```
> plot(y(x), x=0..2);
[ Plotting error, empty plot
```

```
> plot(subs(antwoord, y(x)), x=0..2);
```



```
> plot(subs(antwoord, y(t)), t=0..2);
[ Plotting error, empty plot
```

Kunt u deze resultaten verklaren?

- Direct een grafiek

Een aantal extra tekenopties krijgt u via het package [plots](#):

```
[ > with(plots) :  
  Warning, the name changecoords has been redefined
```

Het commando [odeplot](#) geeft nu mogelijkheden om direct numerieke benaderingen van gewone differentiaalvergelijkingen te tekenen.

plots[odeplot] - 2-D or 3-D plot of output from dsolve

Calling Sequence

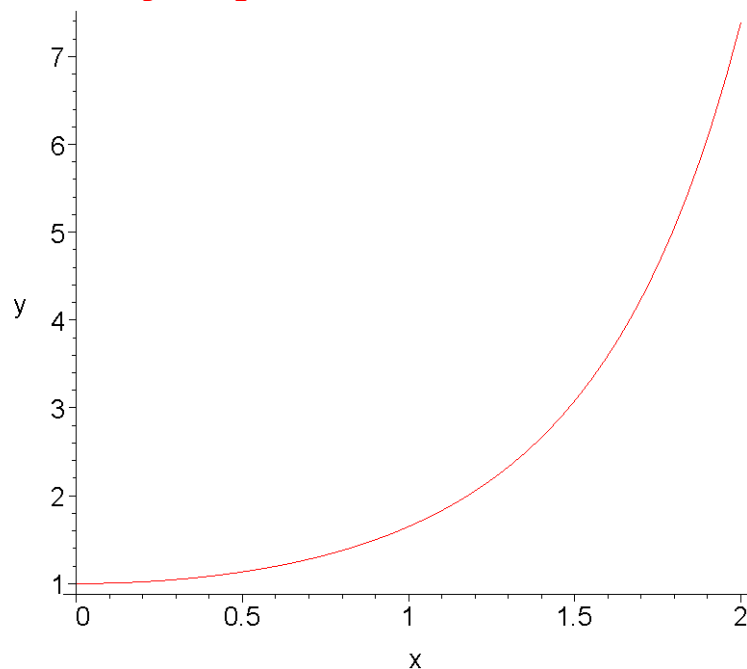
`odeplot(dsn, vars, range, options)`

Parameters

- `dsn` - output from a call to `dsolve(... , numeric)`
- `vars` - (optional list) axes and functions to plot
- `range` - (optional) parameter specification of type a..b
- `options` - (optional) equations that specify plot options; see `?plot[options]` and `?plot3d[options]`

De oplossing van het bovenstaande voorbeeld levert via `odeplot` de volgende schets.

```
[ > benadering:=dsolve({gdv,y(0)=1},y(x),numeric);  
                               benadering:=proc(x_rkf45) ... end proc  
[ > odeplot(benadering,[x,y(x)],0..2);
```



- Numerieke methoden

In de laatste regels van de vorige sectie hebben we de optie **numeric** ontmoet. Dit schept een groot aantal nieuwe mogelijkheden om een indruk van de oplossing te krijgen. Let op dat het hier om benaderingen gaat en niet over oplossingen in de vorm van expliciete standaardfuncties.

dsolve/numeric - find numerical solution of ordinary differential equations

Calling Sequence

```
dsolve(odesys, numeric, vars, options)  
dsolve(numeric, vars, procopts, options)
```

Parameters

odesys - set or list; ordinary differential equation(s) and initial/boundary conditions
numeric - name; instruct **dsolve** to find a numerical solution
vars - (optional) dependent variable or a set or list of dependent variables for **odesys**
procopts - (required if **odesys** is not present) options that specify a procedure defined system
options - (optional) equations of the form **keyword = value**

Standaard levert de optie **numeric** benaderingen via de *Fehlberg fourth-fifth order Runge-Kutta method*. Welke methode dit precies is doet er even niet toe. We kunnen heirmee een oplossing van `gdv4` numeriek benaderen. Expliciete oplossingsmethoden gaven geen resultaat.

```
> gdv4;
```

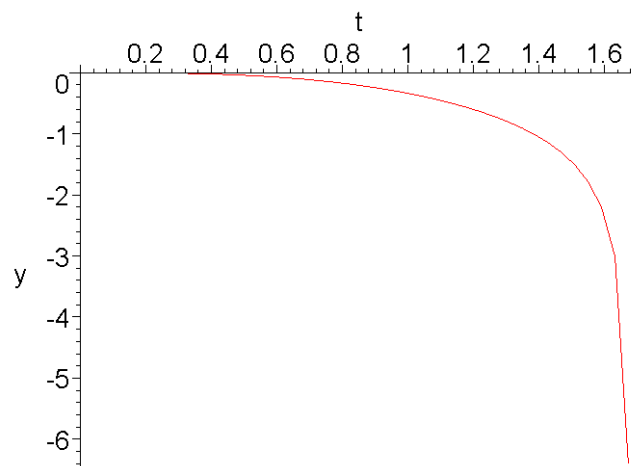
$$D(y)(t) = y(t)^3 - t^2$$

```
> benadering4 := dsolve ({gdv4, y(0)=0}, y(t), numeric, start=0);
```

```
benadering4 := proc(x_rkf45) ... end proc
```

```
> odeplot (benadering4, [t, y(t)], 0..2);
```

Warning, cannot evaluate the solution further right of 1.7142857, probably a singularity



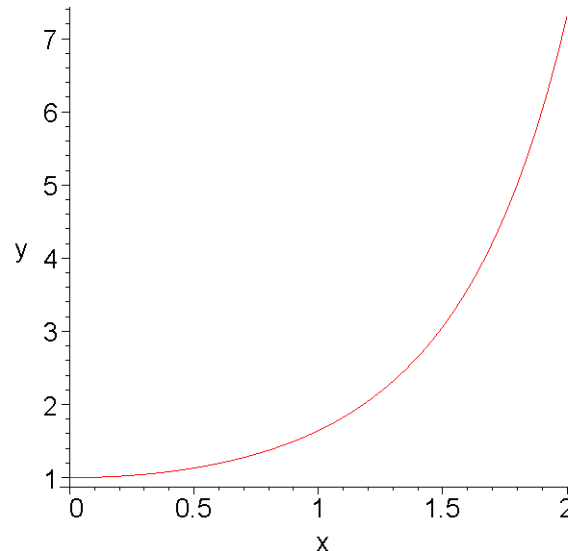
— Alternatieve numerieke methoden

Alternatieve methoden zijn bijvoorbeeld in Maple 8: [rosenbrock](#), [bvp](#), [dverk78](#), [lsode](#), [gear](#), [taylorseries](#), en [classical](#) waarbij we nog verder kunnen specificeren. De *forward Euler method* wordt als volgt aangeroepen.

```
> benadering2:=dsolve({gdv,y(0)=1},y(x),numeric,method=classical[foreuler],start=0);
```

```
benadering2 := proc(x_classical) ... end proc
```

```
> odeplot(benadering2, [x,y(x)], 0..2);
```

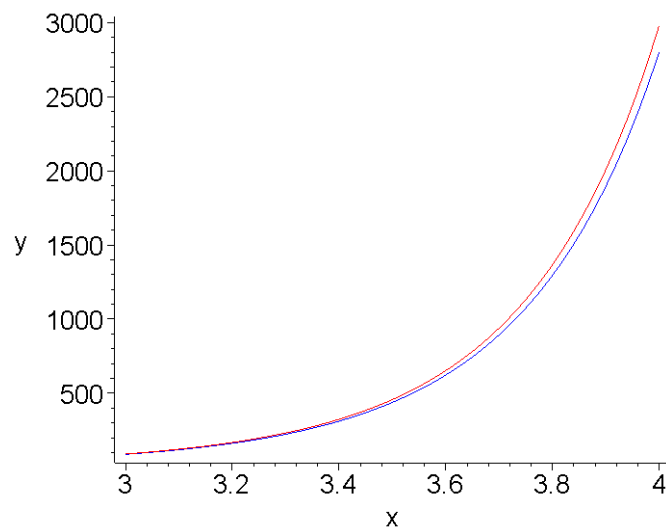


Vergelijk ook eens de beide benaderingen op [3,4]:

```
> p1:=odeplot(benadering, [x,y(x)], 3..4):
```

```
p2:=odeplot(benadering2, [x,y(x)], 3..4,color=blue):
```

```
display({p1,p2});
```



- Vectorveld en fasevlak

Een klassieke manier om zonder oplossingsmethoden toch een indruk te krijgen wat een oplossing van differentiaalvergelijkingen doen, is via een schets van het bijbehorende vectorveld. Dit tekenen gebruikt twee dimensies en dat beperkt de toepassing tot twee gevallen: 1) een eerste orde differentiaalvergelijking en 2) een stelsel van twee autonome eerste orde differentiaalvergelijkingen.

Om in een deel van het xy-vlak op plaats (x,y) een vector in de richting met richtingscoëfficiënt $f(x,y)$ te tekenen hebben we [DEplot](#) nodig.

DEtools[DEplot] - plot solutions to a system of DEs

Calling Sequences

`DEplot(deqns, vars, trange, options)`

`DEplot(deqns, vars, trange, inits, options)`

`DEplot(deqns, vars, trange, yrange, xrange, options)`

`DEplot(deqns, vars, trange, number, yrange, xrange, options)`

`DEplot(deqns, vars, trange, inits, xrange, yrange, options)`

Parameters

`deqns` - list or set of first order ordinary differential equations, or a single differential equation of any order

`vars` - dependent variable, or list or set of dependent variables

`trange` - range of the independent variable

`number` - equation of the form 'number'=integer indicating the number of differential equations when `deqns` is given as a function instead of expressions

`inits` - set or list of lists; initial conditions for solution curves

`yrange` - range of the first dependent variable

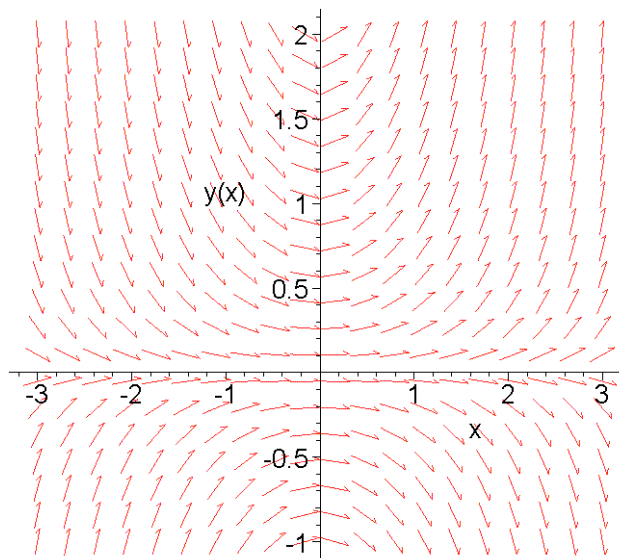
`xrange` - range of the second dependent variable

`options` - (optional) equations of the form keyword=value

- Vectorveld

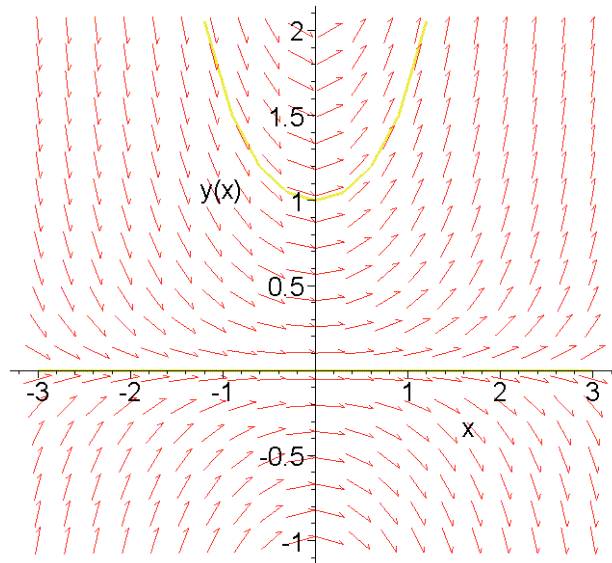
Voor het al eerder gebruikte voorbeeld wordt dat als volgt.

```
> DEplot(gdv, y(x), x=-3..3, y=-1..2);
```



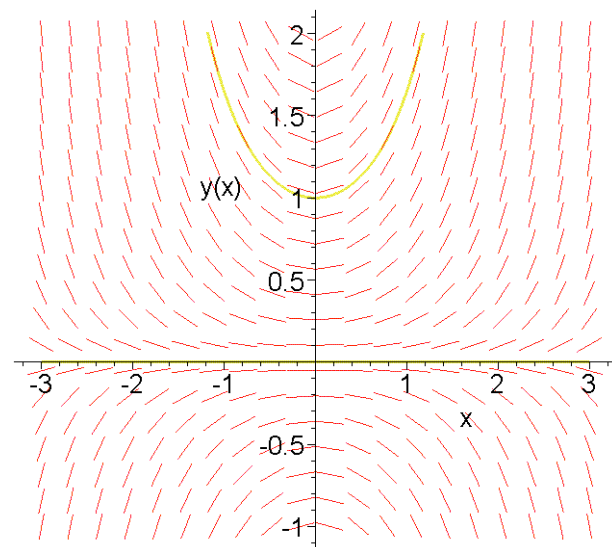
Het is zelfs mogelijk gelijktijdig numerieke benaderingen van een aantal oplossingskrommen te schetsen. Let op de plaats van de beginvoorwaarden en de haakjes.

```
> DEplot(gdv, y(x), x=-3..3, { [y(0)=0], [y(0)=1] }, y=-1..2);
```



De numerieke benadering die hierbij wordt gebruikt lijkt op *Euler forward* met standaard een vrij grove stapgrootte. De **stepsize** optie geeft de mogelijkheid deze stapgrootte te veranderen. Andere pijltjes zijn mogelijk met de optie **arrows** te veranderen naar **medium**, **large**, **line**, **small** of **none**.

```
> DEplot(gdv, y(x), x=-3..3, { [y(0)=0], [y(0)=1] }, y=-1..2, arrows
=line, stepsize=.01);
```



— Fasevlak

Een stelsel van twee eerste orde autonome differentiaalvergelijkingen, $x'(t)=f(x(t),y(t))$ en $y'(t)=g(x(t),y(t))$, levert oplossingen die van t afhangen. De oplossingskrommen $\{(x(t),y(t)); t \text{ binnen existentie-interval}\}$ hangen echter niet van t af. Een voorbeeld is het volgende.

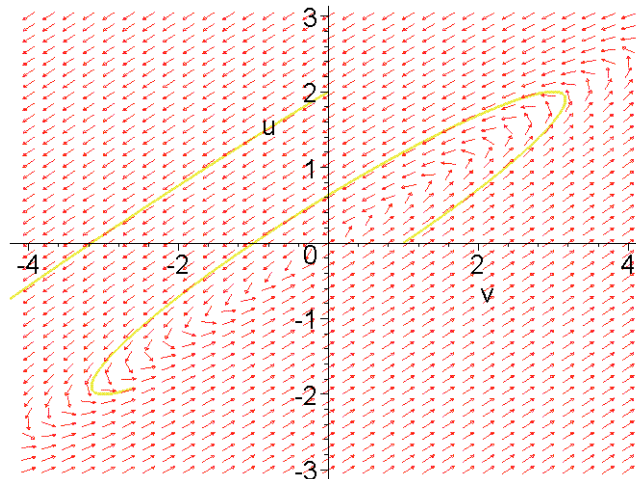
```
> autonoomgdvstelsel := {D(u)(t)=-3*u(t)+2*v(t),
D(v)(t)=-5*u(t)+3*v(t)};
```

```
autonoomgdvstelsel := {D(u)(t)=-3 u(t)+2 v(t), D(v)(t)=-5 u(t)+3 v(t)}
```

Het aantal pijltjes kan veranderd worden met **dirgrid**.

```
> DEplot(autonoomgdvstelsel, {u(t), v(t)}, t=0..5,
```

```
{[u(0)=2,v(0)=0],[u(0)=0,v(0)=1]}, u=-3..3,v=-4..4,
arrows=medium, stepsize=.01, dirgrid=[31,41]);
```

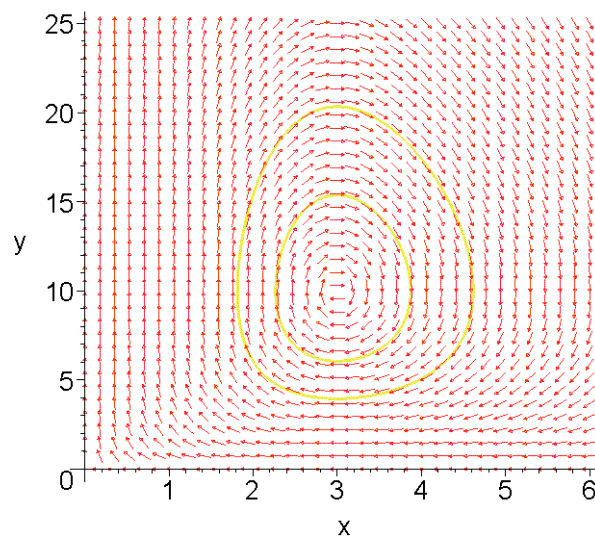


- Lotke-Volterra voorbeeld

De onderstaande stelsel differentiaalvergelijkingen modelleert de populaties roofdieren en prooidieren. Als er veel roofdieren zijn neemt het aantal prooidieren af waardoor de roofdieren honger lijden en hun aantal afneemt zodat het aantal prooidieren weer kan groeien waardoor etcetera

```
> LVstelsel := {D(x)(t) = -x(t) + 0.1*x(t)*y(t),
D(y)(t) = 3*y(t) - x(t)*y(t)};
LVstelsel := {D(x)(t) = -x(t) + 0.1*x(t)*y(t), D(y)(t) = 3*y(t) - x(t)*y(t)}
> DEplot(LVstelsel, {x(t), y(t)}, t=0..5,
{[x(0)=2.2, y(0)=5], [x(0)=2.4, y(0)=13]}, x=0..6,
y=0..25, arrows=medium, title='Lotke-Volterra',
stepsize=0.01, dirgrid=[35,35]);
```

Lotke-Volterra



```
>
```

[Welke variabele staat voor het aantal prooidieren?