CURVAS PARAMETRIZADAS

ParamPlot - anima uma curva bidimensional parametrizada, ou um conjunto de duas curvas parametrizadas.

SINTAXE: ParamPlot([x(t),y(t)],t=a..b),

ParamPlot({[x1(t),y1(t)],[x2(t),y2(t)]},t=a..b),

ParamPlot([x(t),y(t)],t=a..b,ops)

PARAMETROS: x(t) - a expressão para a primeira componente da parametrização

y(t) - a expressão para a segunda componente da parametrização

t -a variavel independente da parametrização

a..b - a variação da variavel independente

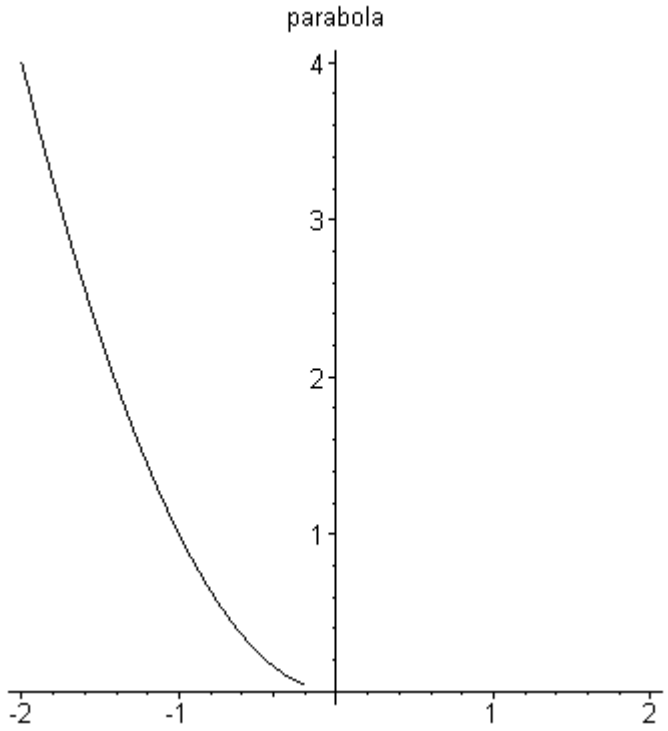
ops - qualquer numero de opcoes conveniente para o plot

Execute este prodecimento e faça os exemplos.

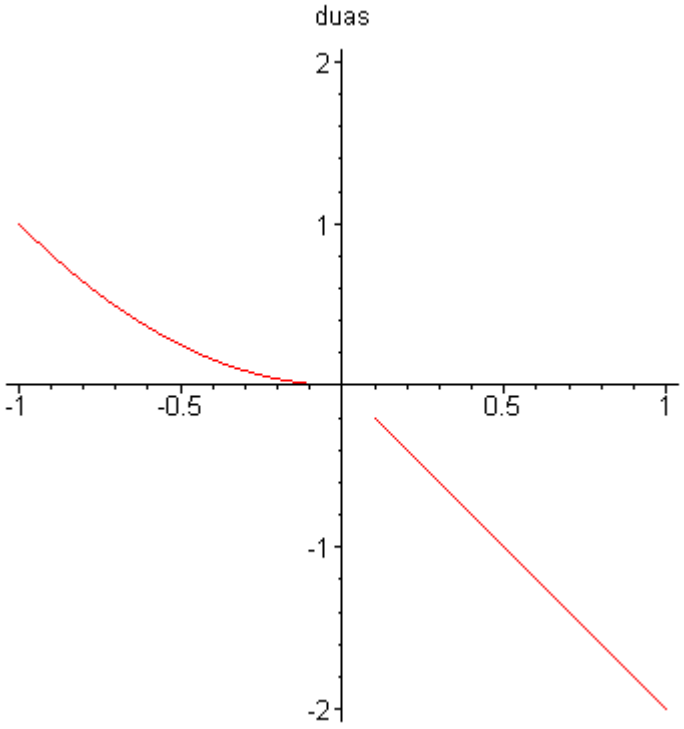
O Procedimento (execute-o)

Exemplos

> **ParamPlot([t,t^2],t=-2..2, title=' parabola');**



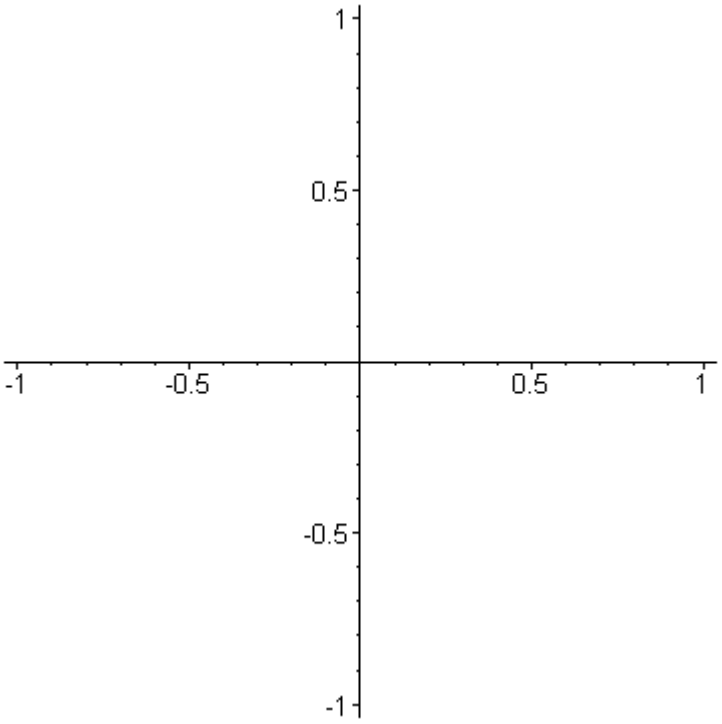
> **ParamPlot({[t,t^2],[-t,2*t]},t=-1..1, title=' duas');**



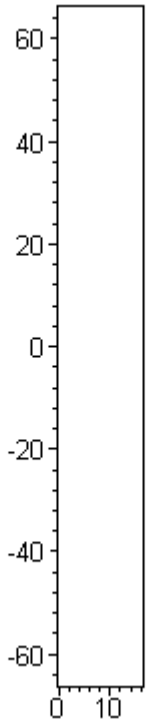
> **p:=[cos(2*t),sin(3*t)];**

$$p := [\cos(2\,t), \sin(3\,t)]$$

> **ParamPlot(p,t=0..2*Pi);**



> **ParamPlot([t^2,t^3],t=-4..4,axes=BOXED,scaling=CONSTRAINED);**



>

O Procedimento (execute-o)

```
> ParamPlot := proc(flist:{vector(algebraic),list(algebraic),
> set(list(algebraic)),set(vector(algebraic))},
> trange:name=range(constant))
> local xloc, td, tstart, plotlist, numpts, opargs, n, funclist,
> funcset,i,plotset;
> options `Copyright 1993 `;
> if type(flist,set) then
> funcset := map(convert,flist,list)
> else
> funcset := {convert(flist,list)}
> fi;
> numpts := 100;
> opargs := [];
> td := op(1,trange);
> for i from 1 to nops(funcset) do
> funclist := op(i,funcset);
> if nops(funclist)<>2 then
> ERROR(`first argument must be a two component vector or list.`) fi;
> if nops(indets(funclist,name) minus indets(funclist,constant))>1 then
> ERROR(`the first argument has parameters that must be defined.`) fi;
> if not(member(td,indets(funclist,name))) then
> ERROR(`second argument variable not present in the first.`) fi;
> od;
> if nargs > 2 then
> for n from 3 to nargs do
```

```
> if op(1,args[n])=numpoints then
> numpts := op(2,args[n])
> elif op(1,args[n])=frames then
> ERROR(`the frames option cannot be changed.`)
> else
> opargs := [opargs[],args[n]];
> fi
> od
> fi;
> tstart := op(1,op(2,trange));
> plotset := {};
> for i from 1 to nops(funcset) do
> funclist := op(i,funcset);
> plotlist := subs(td = tstart + xloc*(td - tstart),funclist);
> plotlist := [op(plotlist),xloc=0..1];
> plotset := plotset union {plotlist};
> od;
> plots[animate](plotset,trange,frames=32,numpoints=numpts,opargs[]);
> end:
>
```