Esempio Metodo basato su Taylor

```
> restart:
> with(plots):
> with(Student[VectorCalculus]):
```

Esempio di equazione differenziale

```
> f := (x,y) -> x*y^2-y ;
  a := 0 ;
  b := 6 ;
```

$$f := (x, y) \to x\,y^2 + Student:\text{-}VectorCalculus:\text{-}`\text{-}`(y)$$

$$a := 0$$

$$b := 6 \qquad (1)$$

```
> SetCoordinates(cartesian[x,y]);
```

$$cartesian_{x,\,y} \qquad (2)$$

```
> VF := VectorField( <1,f(x,y)>, output=plot, view=[a..b, 0..1],
  color="NavyBlue",
                     fieldoptions=[fieldstrength=fixed,arrows=SLIM,
  grid=[40,40]] ):
```

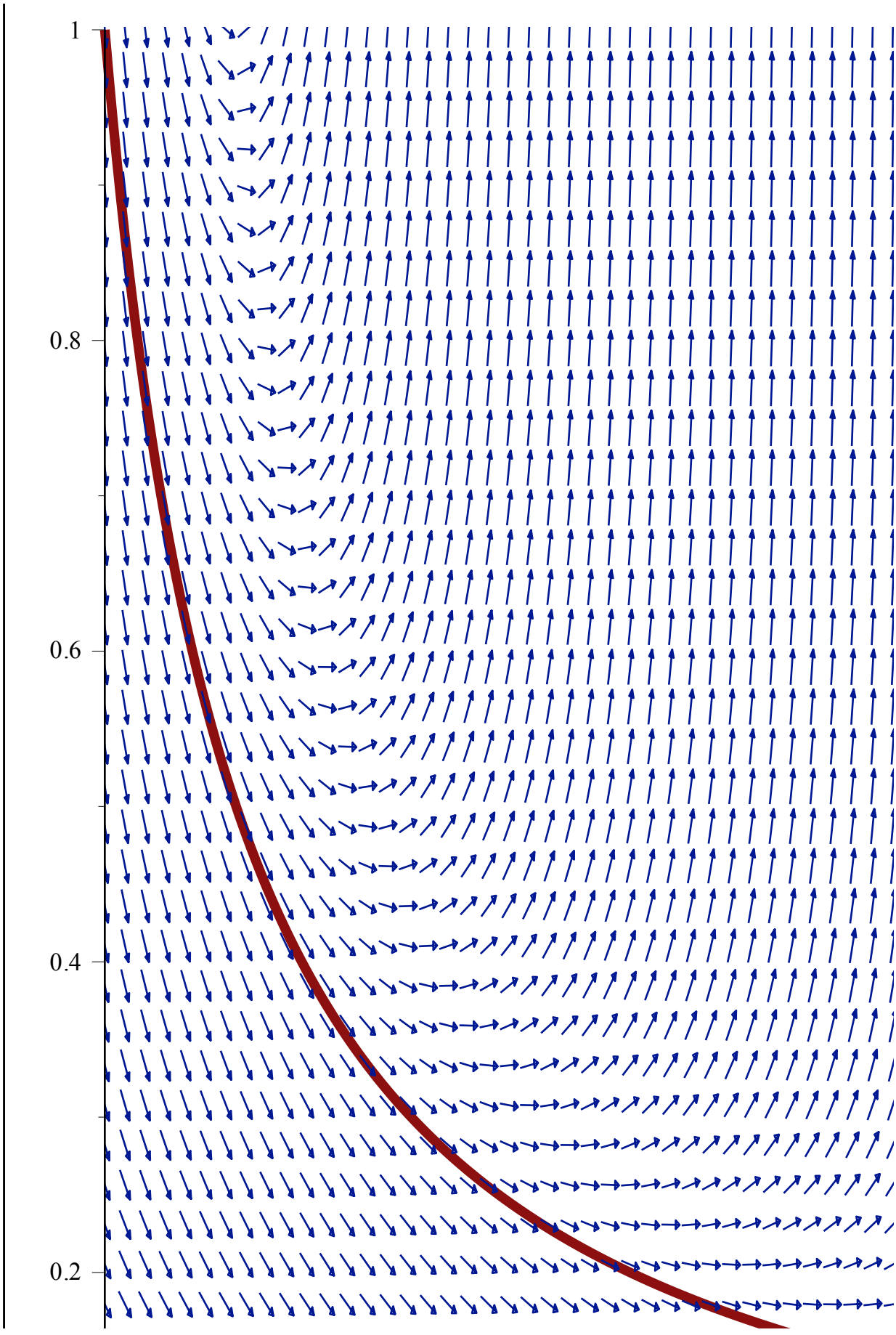
Soluzione esatta dell'ODE

```
> ESATTA := simplify(dsolve( {diff(y(x),x) - f(x,y(x)), y(a)=1} )) ;
```

$$ESATTA := y(x) = \frac{1}{x+1} \qquad (3)$$

Plot della soluzione in [a,b]

```
> AA := plot( subs(ESATTA,y(x)), x=a..b,thickness=5 ) :
  display(AA,VF) ;
```


1
0.8
0.6
0.4
0.2

Derivate di y(x) = f1(x,y(x))

```
> #SUBS := D(y)(x) = f1(x,y(x)) ;
> #Dy := f1(x,y(x)) ;
> #DDy :=  subs(SUBS,simplify(diff(Dy,x))) ;
> #DDDy :=  subs(SUBS,simplify(diff(DDy,x))) ;
> #DDDDy :=  subs(SUBS,simplify(diff(DDDy,x))) ;
```

Derivate di y(x) = x*y(x)-y(x)^2

```
> fun := x*y(x)^2- y(x) ;
```

$$fun := x\,y(x)^2 - y(x) \tag{4}$$

```
> SUBS := diff(y(x), x) = fun ;
```

$$SUBS := \frac{d}{dx}\,y(x) = x\,y(x)^2 - y(x) \tag{5}$$

```
> Dy := fun ;
```

$$Dy := x\,y(x)^2 - y(x) \tag{6}$$

```
> DDy := simplify(subs(SUBS,simplify(diff(Dy,x)))) ;
```

$$DDy := y(x)^2 + 2\,x^2\,y(x)^3 - 3\,x\,y(x)^2 + y(x) \tag{7}$$

```
> DDDy := simplify(subs(SUBS,simplify(diff(DDy,x)))) ;
```

$$DDDy := 6\,x\,y(x)^3 - 5\,y(x)^2 + 6\,x^3\,y(x)^4 - 12\,x^2\,y(x)^3 + 7\,x\,y(x)^2 - y(x) \tag{8}$$

```
> DDDDy := simplify(subs(SUBS,simplify(diff(DDDy,x)))) ;
```

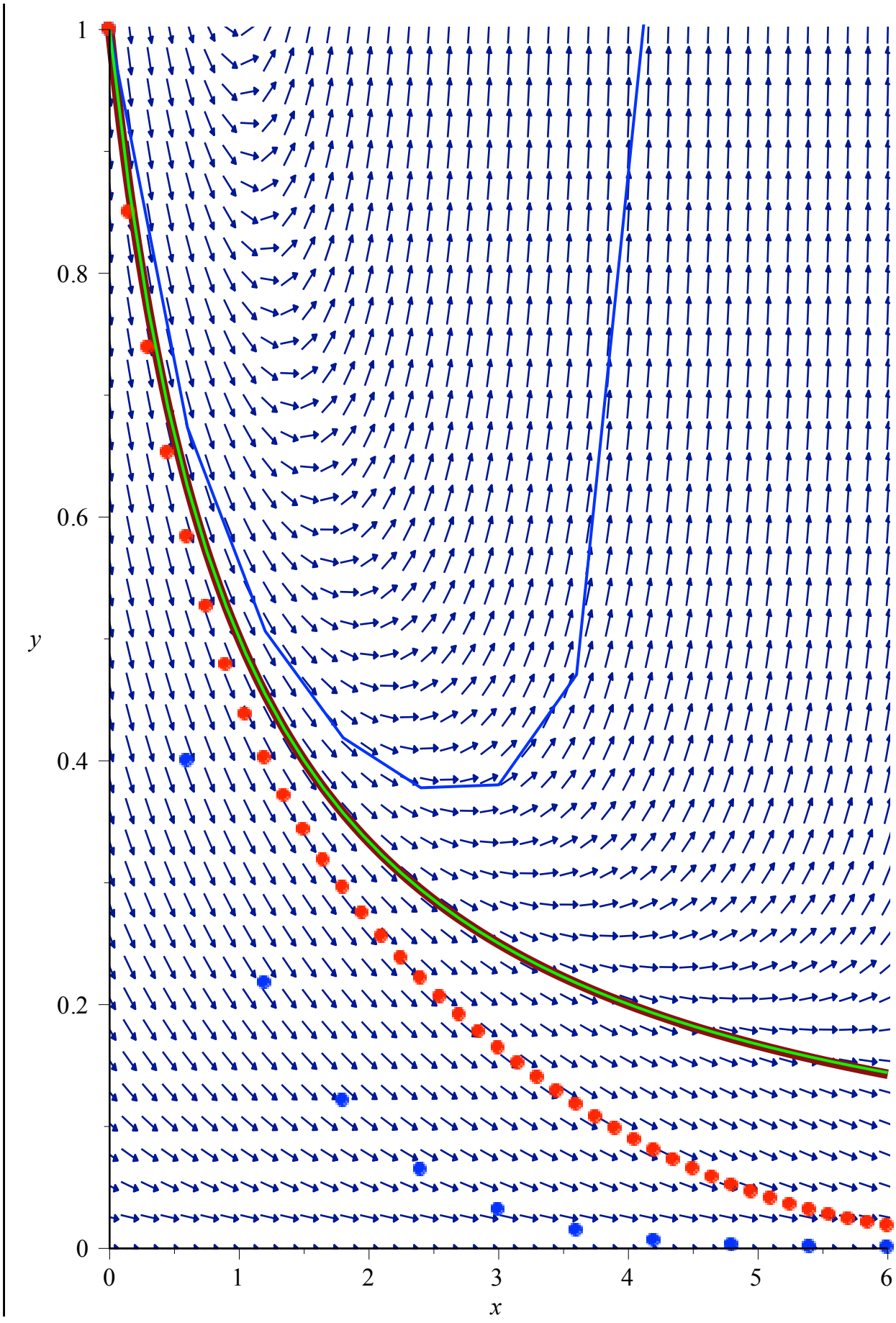
$$DDDDy := 6\,y(x)^3 + 36\,x^2\,y(x)^4 - 52\,x\,y(x)^3 + 17\,y(x)^2 + 24\,x^4\,y(x)^5 - 60\,x^3\,y(x)^4 + 50\,x^2\,y(x)^3 - 15\,x\,y(x)^2 + y(x) \tag{9}$$

```
> TaylorSolver := proc( n::integer )
    local X, Y, h, k, xk, yk, x0, y0, y1, y2, y3, y4 ;
    h := (b-a)/n ;
    X := [a] ;
    Y := [1] ; # dato iniziale
    for k from 1 to n do
      # calcolo yk
      x0 := X[-1] ;
      y0 := Y[-1] ;
      y1 := subs(y(x)=y0,x=x0,Dy) ; # valore della derivata
      y2 := subs(y(x)=y0,x=x0,DDy) ; # valore della derivata
      y3 := subs(y(x)=y0,x=x0,DDDy) ; # valore della derivata
      y4 := subs(y(x)=y0,x=x0,DDDDy) ; # valore della derivata
      # avanzo usando la espansione di Taylor al 4 ordine
      yk := evalf(y0 + h * y1 + (h^2/2!) * y2 + (h^3/3!) * y3 +
  (h^4/4!)*y4) ;
      xk := evalf(x0 + h) ;
      # aggiungo alla lista la soluzione
      X := [op(X),xk] ;
      Y := [op(Y),yk] ;
    end ;
    X,Y ;
  end proc :
```

Funzione Eulero, calcola la soluzione in [a,b] (Eulero a precisione infinita)

```
> EuleroSolver := proc( n::integer )
    local X, Y, h, k, xk, yk, x0, y0, y1 ;
    h := (b-a)/n ;
    X := [a] ;
    Y := [1] ; # dato iniziale
    for k from 1 to n do
      # calcolo yk
```

```
      x0 := X[-1] ;
      y0 := Y[-1] ;
      y1 := subs(y(x)=y0,x=x0,Dy) ; # valore della derivata
      # avanvo usando la espansione di Taylor al 1 ordine
      yk := evalf(y0 + h * y1) ;
      xk := evalf(x0 + h) ;
      # aggiungo alla lista la soluzione
      X := [op(X),xk] ;
      Y := [op(Y),yk] ;
    end ;
    X,Y ;
  end proc :
> TX10,TY10  := TaylorSolver(10) :
  TX40,TY40 := TaylorSolver(40) :
> X10,Y10 := EuleroSolver(10) :
  X40,Y40 := EuleroSolver(40) :
> BB := plot( [seq([X10[k],Y10[k]],k=1..nops(X10))], x=0..6, y=0..1,
  color=blue, style=point, symbol=solidcircle ):
  CC := plot( [seq([X40[k],Y40[k]],k=1..nops(X40))], x=0..6, y=0..1,
  color=red, style=point, symbol=solidcircle ):
  DD := plot( [seq([TX10[k],TY10[k]],k=1..nops(X10))], x=0..6, y=0.
  .1,color=blue ):
  EE := plot( [seq([TX40[k],TY40[k]],k=1..nops(X40))], x=0..6, y=0.
  .1,color=green ):
  display(VF,AA,BB,CC,DD,EE) ;
```


1
0.8
0.6
0.4
0.2
0
y
0
1
2
3
4
5
6
x