

Oscillations amorties par frottement turbulent

restart

faute de mesures on en construit une simulation par intégration numérique

$$\text{equa} := \frac{d^2}{dt^2} x(t) + 2 \beta \left(\frac{d}{dt} x(t) \right) \left| \frac{d}{dt} x(t) \right| + \omega_0^2 x(t) = 0$$

$$\frac{d^2}{dt^2} x(t) + 2 \beta \left(\frac{d}{dt} x(t) \right) \left| \frac{d}{dt} x(t) \right| + \omega_0^2 x(t) = 0 \quad (1)$$

$$\text{ini1} := x(0) = 1; \text{ini2} := D(x)(0) = 0$$

$$x(0) = 1$$

$$D(x)(0) = 0 \quad (2)$$

$$\omega_0 := \text{evalf}(2 \text{ Pi}); \beta := 0.12$$

$$6.283185308$$

$$0.12 \quad (3)$$

$$\text{sol} := \text{dsolve}(\{\text{equa}, \text{ini1}, \text{ini2}\}, \{x(t)\}, \text{type} = \text{numeric})$$

$$\text{proc}(x_rkf45) \dots \text{end proc} \quad (4)$$

$$\text{tpts} := \text{Array}(1..1001, \text{datatype} = \text{float}_8)$$

$$\left[\begin{array}{l} 1 .. 1001 \text{ Array} \\ \text{Data Type: float}_8 \\ \text{Storage: rectangular} \\ \text{Order: Fortran_order} \end{array} \right] \quad (5)$$

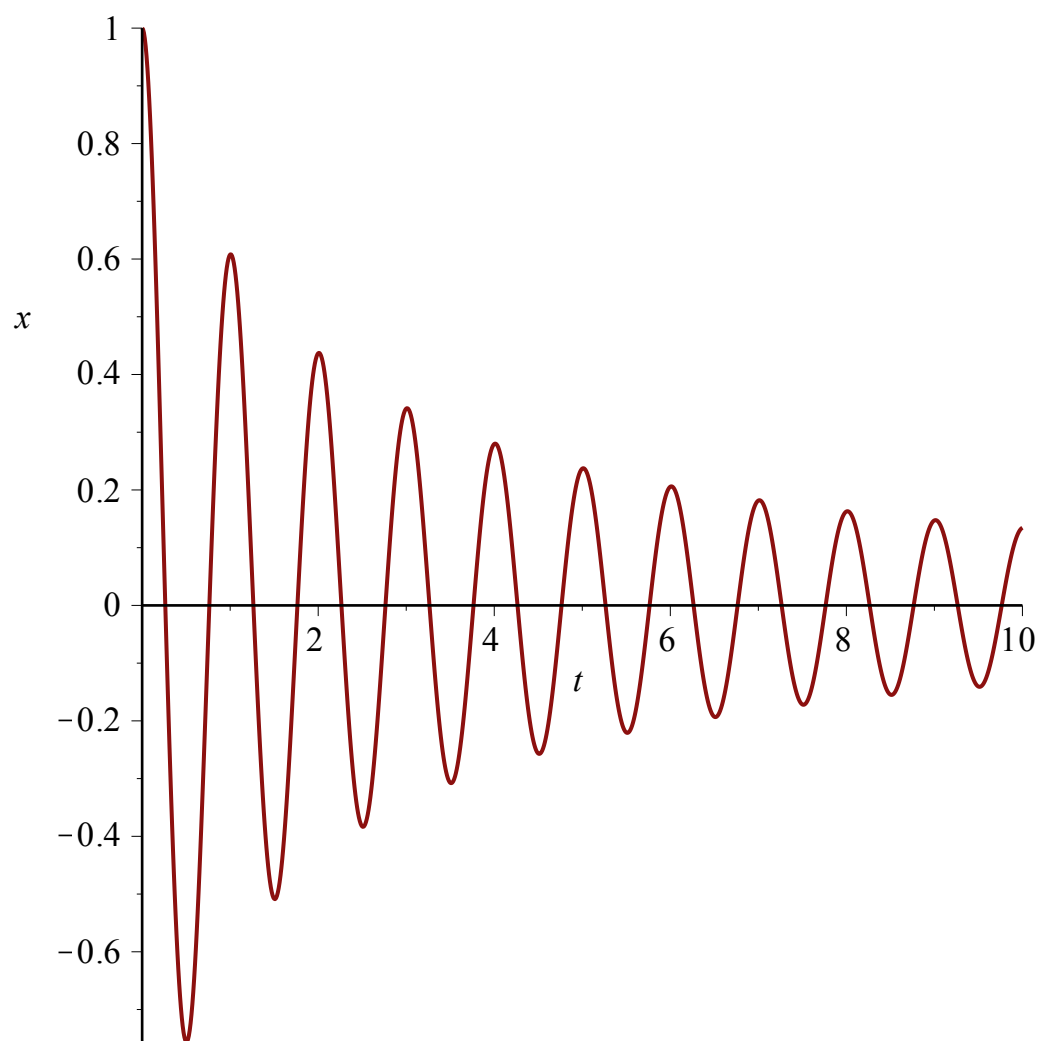
$$\text{xpts} := \text{Array}(1..1001, \text{datatype} = \text{float}_8)$$

$$\left[\begin{array}{l} 1 .. 1001 \text{ Array} \\ \text{Data Type: float}_8 \\ \text{Storage: rectangular} \\ \text{Order: Fortran_order} \end{array} \right] \quad (6)$$

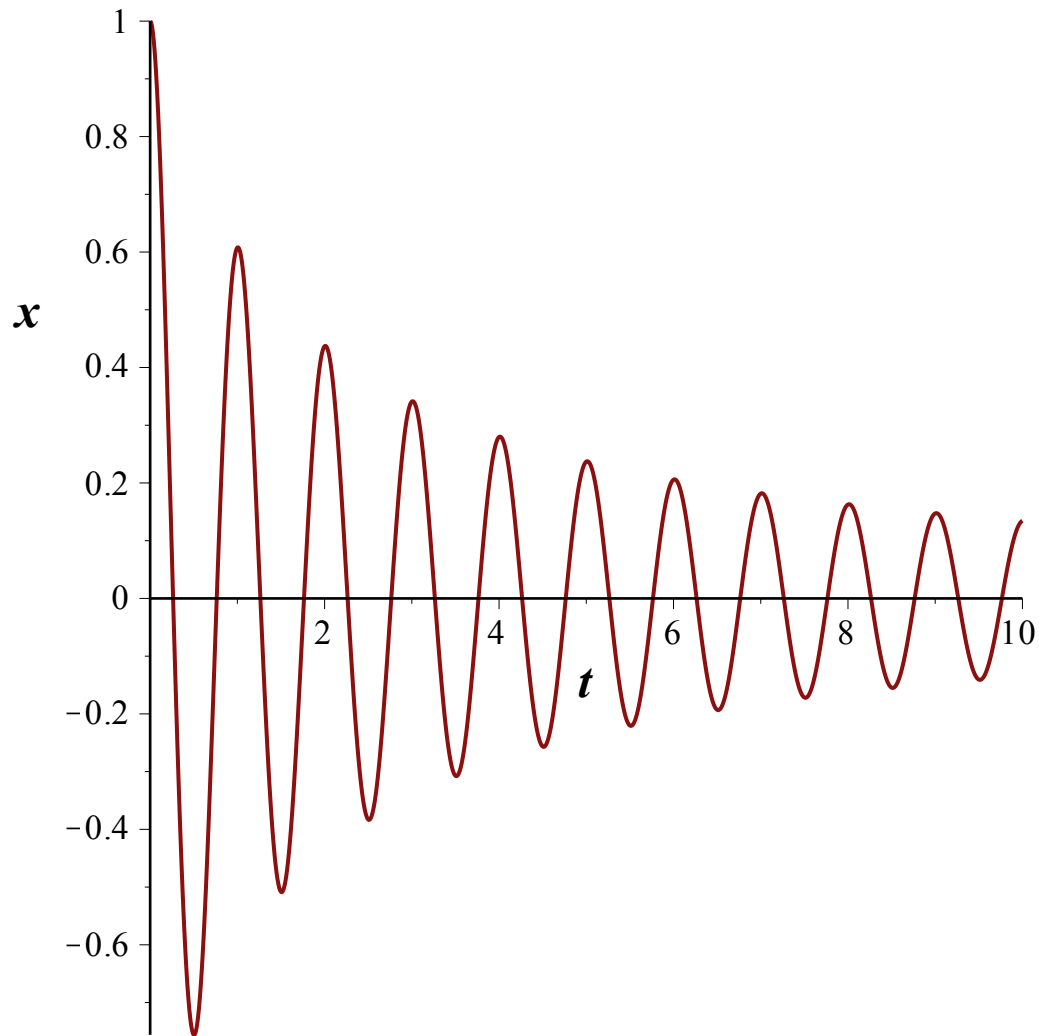
$$\text{for } i \text{ from } 1 \text{ to } 1001 \text{ do } \text{tpts}[i] := \text{rhs}\left(\text{sol}\left(\text{evalf}\left(\frac{i-1}{100.}\right)\right)[1]\right); \text{xpts}[i]$$

$$:= \text{rhs}\left(\text{sol}\left(\text{evalf}\left(\frac{i-1}{100.}\right)\right)[2]\right) \text{end:}$$

$$\text{plots}[\text{odeplot}](\text{sol}, [t, x(t)], 0..10, \text{numpoints} = 1000)$$



`plot(tpts, xpts, labels = [t, x], labelfont = [TIMES, BOLDITALIC, 14])`



#` L'amortissement fluide "turbulent" est :

• au début plus rapide que l'amortissement "visqueux" (grande amplitude ; grandes vitesses ; grandes turbulences)

• à la fin moins rapide (petites vitesses ; peu de turbulences)

on calcule des estimations de la dérivée et de la dérivée seconde

$dxpts := \text{Array}(1 \dots 1001, \text{datatype} = \text{float}_8)$

$$\left[\begin{array}{l} 1 \dots 1001 \text{ Array} \\ \text{Data Type: float}_8 \\ \text{Storage: rectangular} \\ \text{Order: Fortran_order} \end{array} \right] \quad (7)$$

$ddxpts := \text{Array}(1 \dots 1001, \text{datatype} = \text{float}_8)$

$$\left[\begin{array}{l} 1 \dots 1001 \text{ Array} \\ \text{Data Type: float}_8 \\ \text{Storage: rectangular} \\ \text{Order: Fortran_order} \end{array} \right] \quad (8)$$

$dxpts[1] := 0. :$

$cf := \text{CurveFitting}[\text{LeastSquares}](tpts[1 \dots 7], xpts[1 \dots 7], v, \text{curve} = 1 + a v^2 + b v^3 + c v^4)$

$$-19.7425456122476 v^2 + 0.231858380330651 v^3 + 91.8681637123044 v^4 + 1. \quad (9)$$

$$ddxpts[1] := subs\left(v = 0.00, \frac{d}{d v} \frac{d}{d v} cf\right) \\ -39.4850912244951 \quad (10)$$

$$dxpts[2] := subs\left(v = 0.01, \frac{d}{d v} cf\right) \\ -0.394413882076003 \quad (11)$$

$$ddxpts[2] := subs\left(v = 0.01, \frac{d}{d v} \frac{d}{d v} cf\right) \\ -39.3609379252205 \quad (12)$$

$$dxpts[3] := subs\left(v = 0.02, \frac{d}{d v} cf\right) \\ -0.786483813194712 \quad (13)$$

$$ddxpts[3] := subs\left(v = 0.02, \frac{d}{d v} \frac{d}{d v} cf\right) \\ -39.0163010330364 \quad (14)$$

$$dxpts[4] := subs\left(v = 0.03, \frac{d}{d v} cf\right) \\ -1.17400495742703 \quad (15)$$

$$ddxpts[4] := subs\left(v = 0.03, \frac{d}{d v} \frac{d}{d v} cf\right) \\ -38.4511805479427 \quad (16)$$

for i from 5 to 998 do

$cf := \text{CurveFitting}[\text{LeastSquares}]([tpts[i-3], tpts[i-2], tpts[i-1], tpts[i], tpts[i+1], tpts[i+2], tpts[i+3]], [xpts[i-3], xpts[i-2], xpts[i-1], xpts[i], xpts[i+1], xpts[i+2], xpts[i+3]], v, \text{curve} = a + b v + c v^2 + d v^3);$

$dxpts[i] := subs\left(v = \text{evalf}\left(\frac{i-1}{100.}\right), \frac{d}{d v} cf\right);$

$ddxpts[i] := subs\left(v = \text{evalf}\left(\frac{i-1}{100.}\right), \frac{d}{d v} \frac{d}{d v} cf\right)$

end:

$cf := \text{CurveFitting}[\text{LeastSquares}]([tpts[995], tpts[996], tpts[997], tpts[998], tpts[999], tpts[1000], tpts[1001]], [xpts[995], xpts[996], xpts[997], xpts[998], xpts[999], xpts[1000], xpts[1001]], v, \text{curve} = a + b v + c v^2 + d v^3 + e v^4)$

$$79750.1721796882 - 31909.3687948423 v + 4785.13131511572 v^2 \\ - 318.746041739434 v^3 + 7.95765605210480 v^4 \quad (17)$$

$$dxpts[999] := subs\left(v = 9.98, \frac{d}{d v} cf\right) \\ 0.174663888654322 \quad (18)$$

$$ddxpts[999] := subs\left(v = 9.98, \frac{d}{d v} \frac{d}{d v} cf\right) \\ -5.22163890117554 \quad (19)$$

$$dxpts[1000] := subs\left(v = 9.99, \frac{d}{d v} cf\right)$$

(20)

$$ddxpts[1000] := subs\left(v = 9.99, \frac{d}{d v} \frac{d}{d v} cf\right)$$

(21)

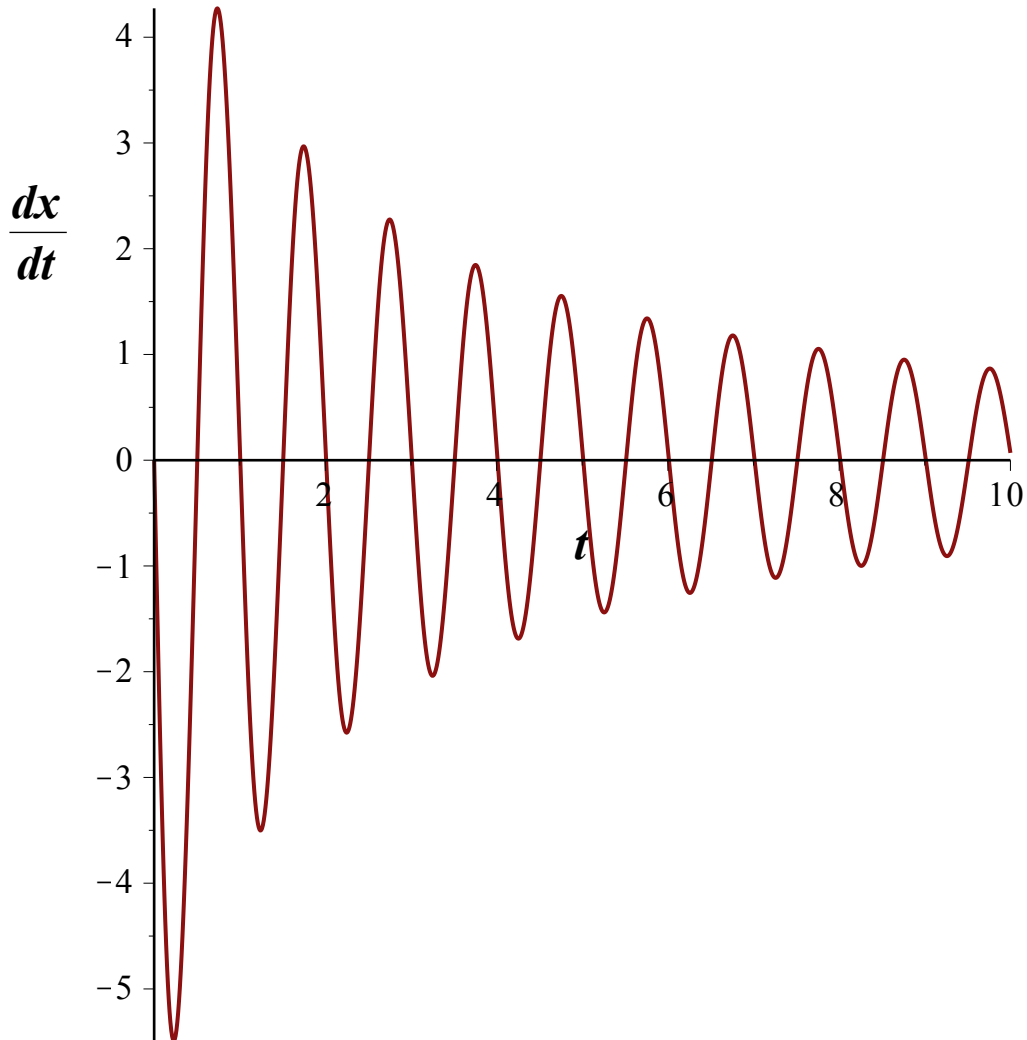
$$dxpts[1001] := subs\left(v = 10.00, \frac{d}{d v} cf\right)$$

(22)

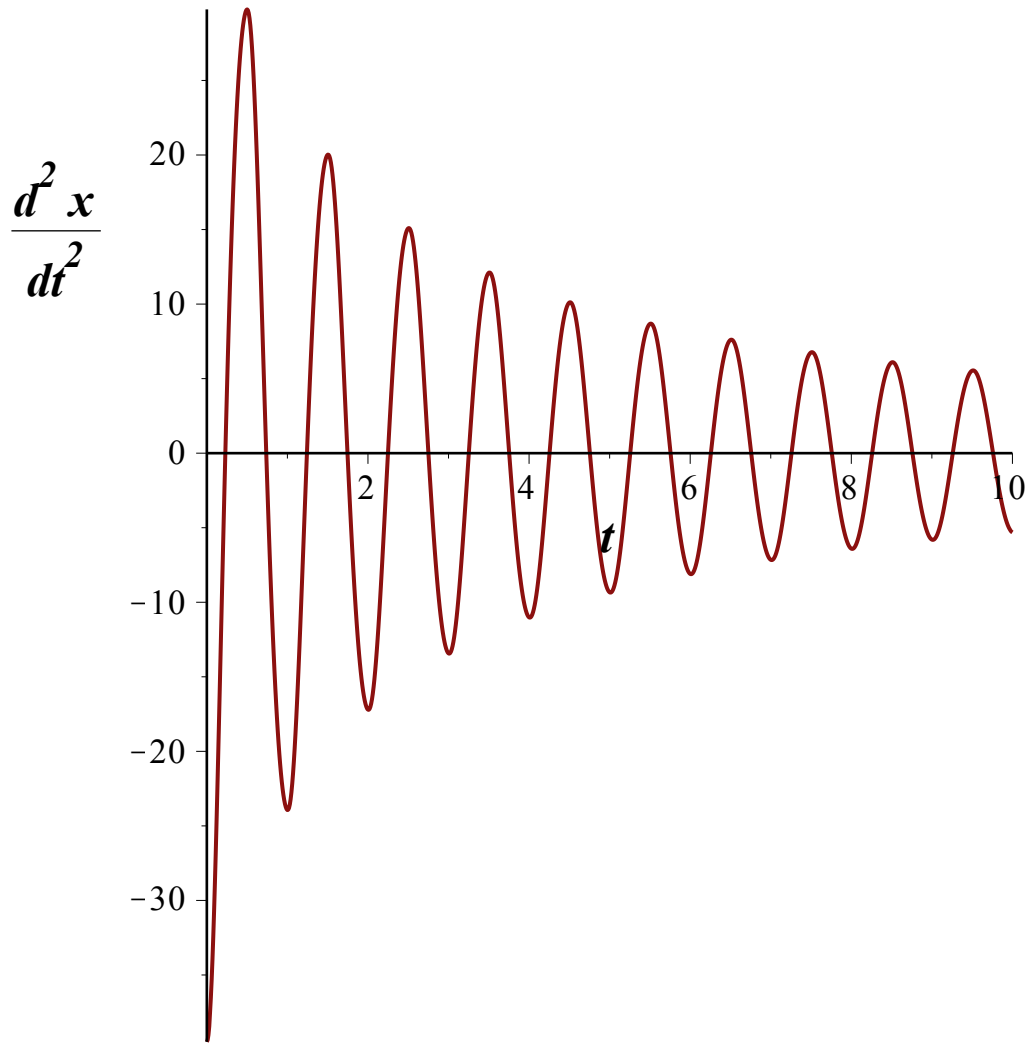
$$ddxpts[1001] := subs\left(v = 10.00, \frac{d}{d v} \frac{d}{d v} cf\right)$$

(23)

$$plot\left(tpts, dxpts, labels = \left[t, \frac{dx}{dt}\right], labelfont = [TIMES, BOLDITALIC, 14]\right)$$



$$plot\left(tpts, ddxpts, labels = \left[t, \frac{d^2x}{dt^2}\right], labelfont = [TIMES, BOLDITALIC, 14]\right)$$



il reste à combiner les trois fonctions pour voir si cela permet de retrouver les paramètres
 # on calcule la période et ω (compatible avec ω_0)

$$ti := tpts[27] + (tpts[28] - tpts[27]) \frac{xpts[27]}{xpts[27] - xpts[28]}$$

(24)

$$tf := tpts[977] + (tpts[978] - tpts[977]) \frac{xpts[977]}{xpts[977] - xpts[978]}$$

(25)

$$T := evalf\left(\frac{tf - ti}{9.5}\right)$$

(26)

$$\omega := evalf\left(\frac{2 \text{ Pi}}{T}\right)$$

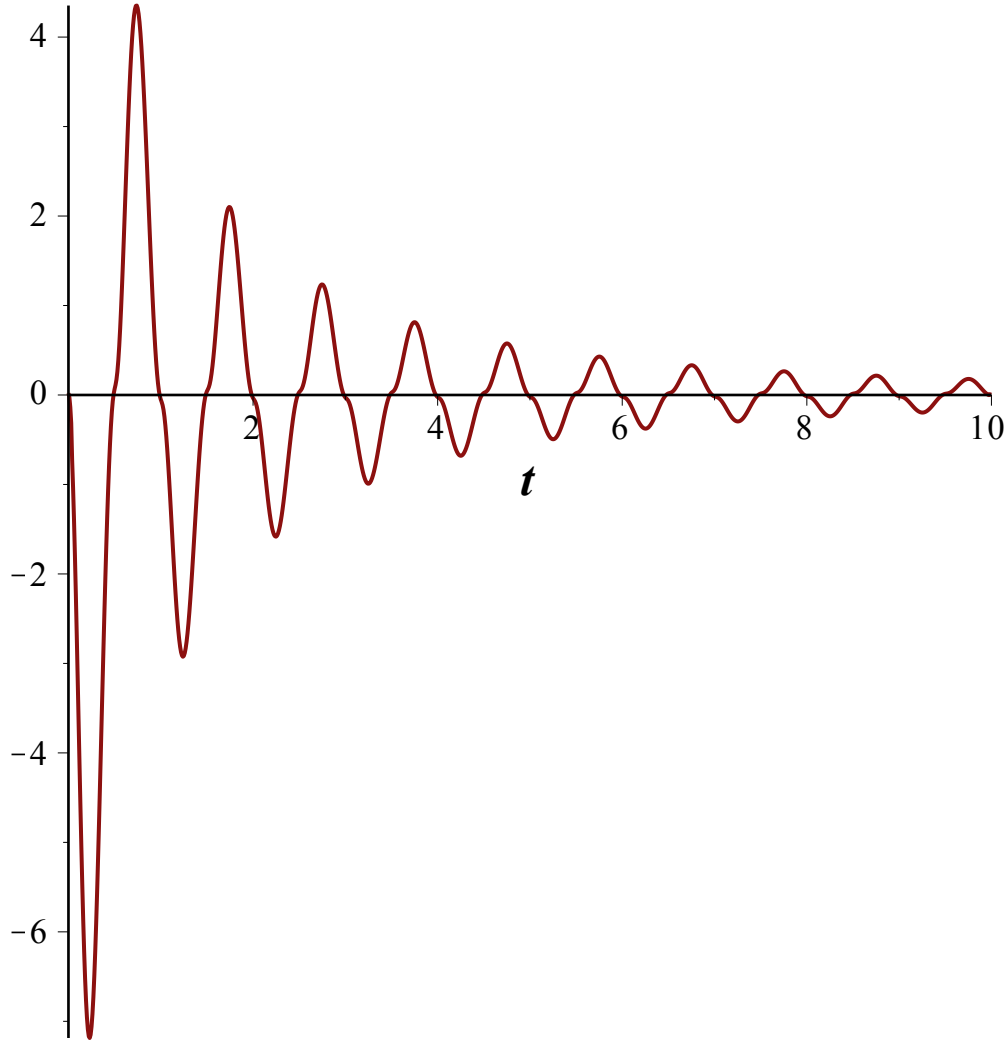
(27)

$$ffpts := \text{Array}(1..1001, datatype = float_8)$$

$$\left[\begin{array}{l} 1 \dots 1001 \text{ Array} \\ \text{Data Type: float}_8 \\ \text{Storage: rectangular} \\ \text{Order: Fortran_order} \end{array} \right]$$

(28)

for i **from** 1 **to** 1001 **do** $ffpts[i] := -evalf(ddxpts[i] + \omega^2 xpts[i])$ **end**:
 $plot(tpts, ffpts, labels = [t, ""], labelfont = [TIMES, BOLDITALIC, 14])$

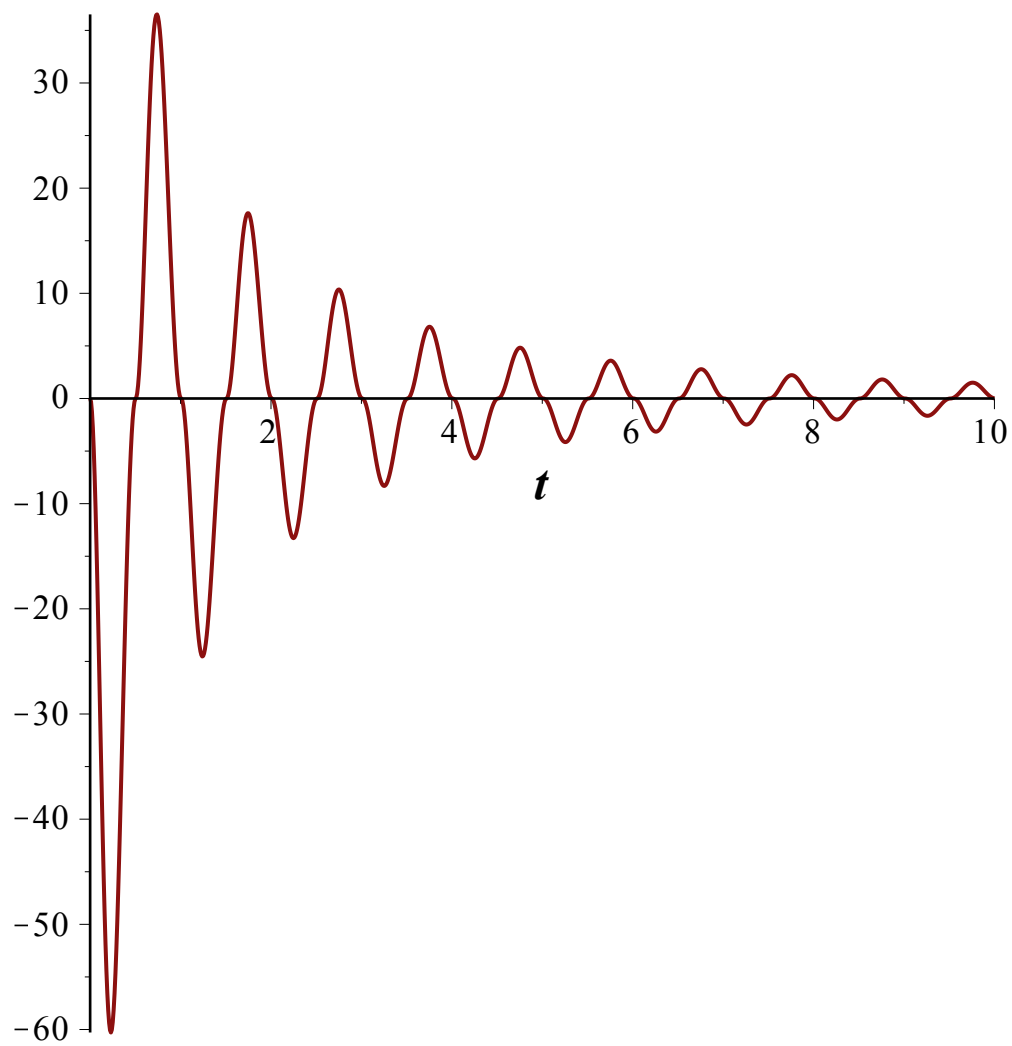


$vvpts := Array(1 \dots 1001, datatype = float_8)$

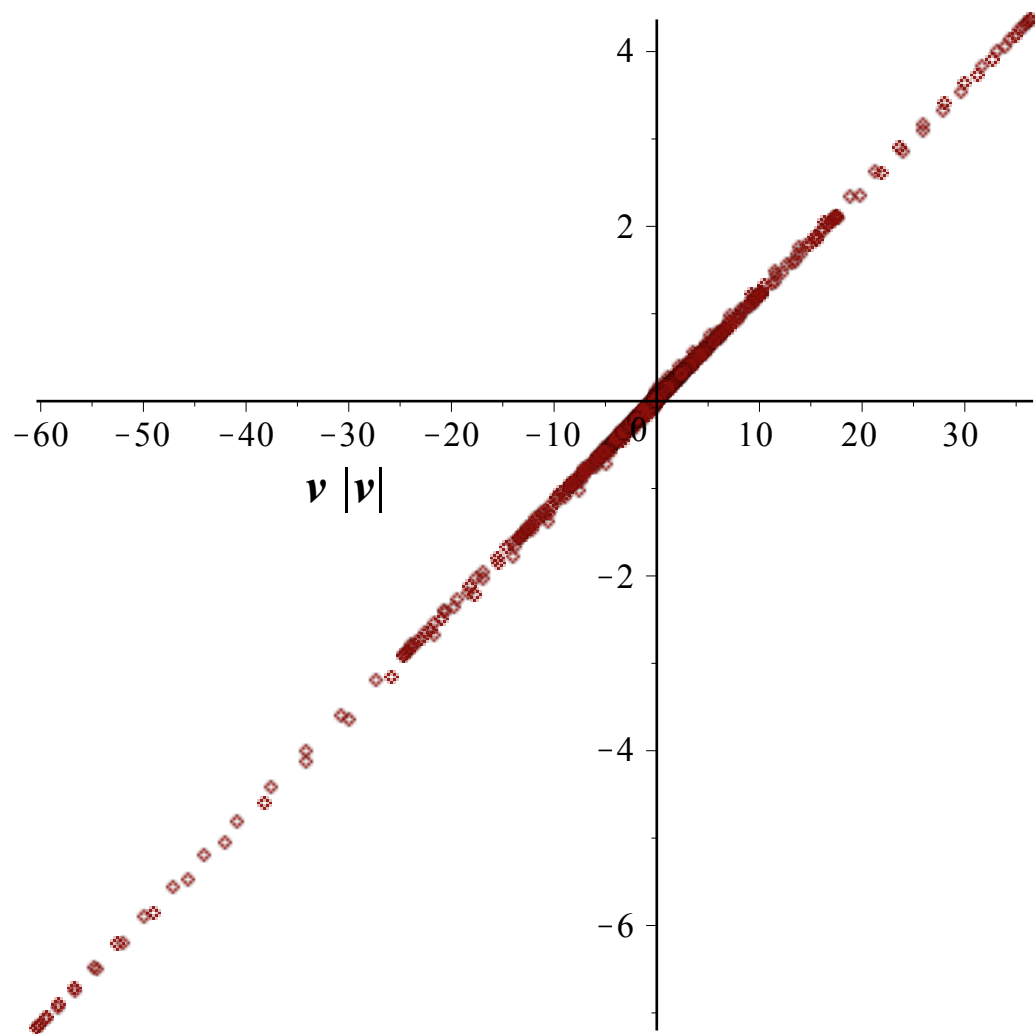
$$\left[\begin{array}{l} 1 \dots 1001 \text{ Array} \\ \text{Data Type: float}_8 \\ \text{Storage: rectangular} \\ \text{Order: Fortran_order} \end{array} \right]$$

(29)

for i **from** 1 **to** 1001 **do** $vvpts[i] := evalf(2 \cdot d xpts[i] \cdot \text{abs}(dxpts[i]))$ **end**:
 $plot(tpts, vvpts, labels = [t, ""], labelfont = [TIMES, BOLDITALIC, 14])$



on vérifie que la forme est compatible (il faut au moins 50 points par période)
 # si la courbe est correcte mais n'est pas une droite, on teste v^k en échelle logarithmique
`plot(vvpts,ffpts, labels = ['v |v|', ""], labelfont = [TIMES, BOLDITALIC, 14])`



on calcule β (à comparer au 0.12 utilisé pour générer la simulation)

`CurveFitting[LeastSquares](vvpts[1..1001],ffpts[1..1001],v,curve=a v)`

[0.119452794956248 v](#)

(30)