9
Referências Bibliográficas

1 GOTTLIEB, H. P. W. Frequencies of oscillators with fractional-power non-linearities. *Journal of Sound and Vibration*, v. 261, p. 557–566, 2003.

2 REN, Y. The receptance-based perturbative multi-harmonic balance method for the calculation of the aperiodic steady state response of non-linear systems. *Journal of Sound and Vibration*, v. 181, n. 2, p. 331–339, 1995.

3 LAU, S. L.; YUEN, S. W. Solution diagram of non-linear dynamic systems by the ihb method. *Journal of Sound and Vibration*, v. 167, n. 2, p. 303–316, 1993.

4 XU, L.; LU, M. W.; CAO, Q. Bifurcation and chaos of a harmonically excited oscillator with both stiffness and viscous damping piecewise linearities by incremental harmonic balance method. *Journal of Sound and Vibration*, v. 264, p. 873–882, 2003.

5 MICKENS, R. E. Comments on the method of harmonic balance. *Journal of Sound and Vibration*, v. 94, n. 3, p. 456–460, 1984.

6 NAYFEH, A. H.; BALACHANDRAN, B. *Applied Nonlinear Dynamics*. New York: John Wiley & Sons, 1995.

7 PARKER, T. S.; CHUA, L. O. *Practical Numerical Algorithms for Chaotic Systems*. New York: Springer-Verlag, 1991.

8 LEWANDOESKI, R. Non-linear, steady-state vibration of structures by harmonic balance method/finite element method. *Computer & Structures*, v. 44, p. 287–296, 1992.

9 TONGUE, B. H. Characteristics of numerical simulations of chaotic systems. *American Society of Mechanical Engineers, Journal of Applied Mechanics*, v. 54, p. 695–699, 1987.

10 LEUNG, A. Y. T. Rigorous studies of a duffing oscillatorion. *Journal of Sound and Vibration*, v. 149, n. 1, p. 147–149, 1991.

11 LEUNG, A. Y. T.; FUNG, T. C. Geometrically non-linear vibration of spinning structures by fynite element method. *Journal of Sound and Vibration*, v. 139, n. 1, p. 43–62, 1990.

12 HASSAN, A.; BURTON, T. D. Extraneous solutions predicted by the harmonic balance method. *Journal of Sound and Vibration*, v. 182, n. 4, p. 523–539, 1995.

13 RAPP, P. E.; MEES, A. I. Spurious predictions of limit cycles in a non-linear feedback system by the describing funtion method. *International Journal Control*, v. 26, n. 6, p. 821–829, 1977.

14 CHEUNG, Y. K.; CHEN, S. H.; LAU, S. L. Application of the incremental harmonic balance method to cubic non-linearity systems. *Journal of Sound and Vibration*, v. 140, n. 2, p. 273–286, 1990.

15 LAU, S. L.; CHEUNG, Y. K.; WU, S. Incremental harmonic balance method with multiple time scales for aperiodic vibration of nonlinear systems. *Journal of Sound and Vibration*, v. 50, p. 871–876, 1983.

16 WONG, C. W.; ZHANG, W. S.; LAU, S. L. Periodic forced vibration of unsymmetrical piecwise-linear systems by incremental harmonic balance method. *Journal of Sound and Vibration*, v. 149, n. 1, p. 91–105, 1991.

17 FERRI, A. A. On the equivalence of the incremental harmonic balance method and the harmonic balance-newton raphson method. *Journal of Applied Mechanics*, v. 53, p. 455–457, 1986.

18 LEWANDOESKI, R. Computational formulation for periodic vibration of geometrically nonlinear structures-part 1: theoretical background. *International Journal of Solids Structures*, v. 34, n. 15, p. 1925–1947, 1997.

19 HASSAN, A. On the local stability of the approximate harmonic balance solutions. *Nonlinear Dynamics*, v. 10, p. 105–133, 1996.

20 HAMDAN, M. N.; AL-QAISIA, A. A.; AL-BEDOOR, B. O. Comparasion of analytical techniques for nonlinear vibrations of a parametrically excited cantileved. *International Journal of Mechanical Sciences*, v. 43, p. 1521–1542, 2001.

21 HASSAN, A. A second comparasion of two higher order perturbation schemes. *Journal of Sound and Vibration*, v. 184, n. 5, p. 907–928, 1995.

22 HAMDAN, M. N.; BURTON, T. D. On the steady state response ans stability of non-linear oscillators using harmonic balance. *Journal of Sound and Vibration*, v. 166, n. 2, p. 255–266, 1993.

23 WU, B. S.; LIM, C. W.; MA, Y. F. Analytical aproximation to large-amplitude oscillation od a non-linear conservative system. *International Journal of Non-Linear Mechanics*, v. 38, p. 1037–1043, 2003.

24 NAYFEH, A. H.; MOOK, D. T. *Nonlinear Oscillations*. New York: John Wiley & Sons, 1979.

25 MICKENS, R. E.; COOPER, K. Generalized harmonic balance/numerical method for determining analytical approximations to the periodic solutions of the $x^{4/3}$ potential. *Journal of Sound and Vibration*, v. 250, n. 5, p. 951–954, 2002.

26 GONçALVES, P. B.; MACHADO, V. L. S. S. A computational strategy for determining bifurcation diagrams for non-linear oscillating systems. *Journal of the Brazilian Society of Mechanical Sciences*, v. 19, n. 4, p. 344–352, 1995.

27 DOOREN, R. V.; JANSSEN, H. A continuation algorithm for discovering new chaotic motions in forced duffing systems. *Journal of Computational and Applied Mathematics*, v. 66, p. 527–541, 1996.

28 NAYFEH, A. H.; CHIN, C.-M. *Perturbation Methods with Maple*. [S.l.]: Dynamics Press, Inc., 1999.

29 SANCHEZ, N. E. The method of multiple scales: Asymptotic solutions and normal forms for nonlinear oscillatory problems. *Journal Symbolic Computation*, v. 19, p. 344–352, 1996.

30 HISTORY/MATHEMATICIANS/EULER.HTML http://www-groups.dcs.st-and.ac.uk/. 7/5/2005.

31 HTTP://WWW.DANG.SE/TEXTER/MOON.TXT. 7/5/2005.

32 HAGEDON, P. *Oscilações não-lineares*. São Paulo: Edgard Blücher LTDA, 1984.

33 GIORGILLI, A. Small denominators and the exponential stability: from poincaré to the present time. *Journal of Applied Mechanics*, v. 53, p. 455–457, 1998.

34 NAYFEH, A. H. *Perturbation Methods*. New York: John Wiley & Sons, 1973.

35 VALéRY, R. R. Averaging method for strongly non-linear oscillators with periodic excitations. *International Journal of Non-Linear Mechanics*, v. 29, p. 737–753, 1994.

36 NAYFEH, A. H. *Problems in Perturbation*. New York: John Wiley & Sons, 1993.

37 HASSAN, A. The kbm derivative expasion method is equivalent to the multiple-time-scales method. *js*, v. 200, n. 4, p. 433–440, 1997.

38 MINORU, U.; RETTER, A. Numerical computation of nonlinear forced oscillations by galerkin's procedure. *Journal of Mathematical Analisys and Applications*, v. 14, p. 107–140, 1966.

39 NAYFEH, A. H. The response of single degree of freedom systems with quadratic and cubic non-linearities to a subharmonic excitation. *Journal of Sound and Vibration*, v. 89, n. 4, p. 457–470, 1983.

40 ATADAN, A. S.; HUSEYIN, K. An intrinsic method of harmonic analisys for non-linear oscillations (a perturbation technique). *Journal of Sound and Vibration*, v. 95, n. 4, p. 525–530, 1984.

41 BAJKOWSKI, J.; SZEMPLINSKA-STUPNICKA, W. Internal resonances efects - simulation versus analytical methods results. *Journal of Sound and Vibration*, v. 104, n. 2, p. 259–275, 1986.

42 BAJKOWSKI, J.; SZEMPLINSKA-STUPNICKA, W. The 1/2 subharmonic resonance and its transiction to chaotic motion in a non-linear oscillator. *International Journal of Non-Linear Mechanics*, v. 21, n. 5, p. 401–419, 1986.

43 NAYFEH, A. H.; ZAVODNEY, L. D. The response of two-degree-of-freedom systems with quadratic non-linearities to a combination parametric resonance. *Journal of Sound and Vibration*, v. 107, n. 2, p. 329–350, 1986.

44 LEUNG, A. Y. T.; FUNG, T. C. Construction of chaotic regions. *Journal of Sound and Vibration*, v. 131, n. 3, p. 445–455, 1989.

45 NAYFEH, A. H.; ZAVODNEY, L. D.; SANCHEZ, N. E. The response of a single-degree-of-freedom system with quadratic and cubic non-linearities to a principal parametric resonance. *Journal of Sound and Vibration*, v. 129, n. 3, p. 417–442, 1989.

46 VIRGIN, L. N.; CARTEE, L. A. A note on the escape from a potencial well. *International Journal of Non-Linear Mechanics*, v. 26, n. 3/4, p. 449–452, 1991.

47 BURTON, T. D.; RAHMAN, Z. On the multi-scale analisys of strongly non-linear forced oscillators. *International Journal of Non-Linear Mechanics*, v. 21, n. 2, p. 135–146, 1986.

48 SARMA, M. S. Applicability of the perturbation technique to the period solution of $\ddot{x} + \alpha x + \beta x^2 + \gamma x^3 = 0$. *js*, v. 180, p. 177–184, 1995.

49 CHEN, S. H.; CHEUNG, Y. K. An elliptic perturbtaion method for certain strongly non-linear oscillators. *Journal of Sound and Vibration*, v. 192, p. 453–464, 1996.

50 BLAIR, K. B.; KROUSGRILL, C. M.; N., F. T. Harmonic balance and continuation tecniques in the dynamic analysis of duffing's equation. *Journal of Sound and Vibration*, v. 202, n. 5, p. 717–731, 1997.

51 FRANCIOSI, C.; TOMASIELLO, S. The use of mathematica for the analysis of strongly nonlinear two-degree-of-freedom systems by means of the modified lindstedt-poincaré method. *js*, v. 211, p. 145–156, 1998.

52 CHEN, S. H.; CHEUNG, Y. K. A modified lindstedt-poincaré method for a strongly non-linear two degree-of-freedom system. *js*, v. 193, p. 751–762, 1996.

53 KHANIN, R.; CARTMELL, M.; GILBERT, A. A computerised implementation of the multiple scales perturbation method using mathematica. *Computer & Structures*, v. 76, p. 565–575, 2000.

54 ANDRIANOV, I.; AWREJCEWICZ, J. A role of initial conditions choice on the results obtainded using different perturbation methods. *Journal of Sound and Vibration*, v. 236, n. 1, p. 161–165, 2000.

55 AMORE, P.; ARANDA, A.; PACE, A. de; LóPEZ, J. A. Comparative study of quantum anharmonic potentials. *Physics Letters A*, v. 329, p. 451–458, 2004.

56 HU, H. A classical perturbation technique which is valid for large parameters. *Journal of Sound and Vibration*, v. 269, p. 409–412, 2004.

57 MICKENS, R. E. Oscillations in an $x^{4/3}$ potential. *Journal of Sound and Vibration*, v. 246, n. 2, p. 375–378, 2001.

58 MICKENS, R. E. Analysis of non-linear oscillators having non-polynomial elastic terms. *Journal of Sound and Vibration*, v. 255, n. 5, p. 789–792, 2002.

59 HE, J.-H. Homotopy perturbation technique. *Comput. Methods Appl. Mech. Engrg.*, v. 178, p. 257–262, 1999.

60 HE, J.-H. The homotopy perturbation method for nonlinear oscillators with discontinuities. *Applied Mathematics and Computation*, v. 151, p. 287–292, 2004.

61 HE, J.-H. New interpretation of homotopy perturbation method. *International Journal of Modern Physics B*, v. 20, n. 18, p. 2561–2568, 2006.

62 BELéNDEZ, A.; BELéNDEZ, T.; MáRQUEZ, A.; NEIPP, C. Application of he's homotopy perturbation method to conservative truly nonlinear oscillators. *Journal of Sound and Vibration*, v. 37, p. 770–780, 2008.

63 CRAIG, R. R.; KURDILA, A. J. *Fundamentals of structural dynamics*. New Jersey: John Wiley & Sons, 2006.

64 SWAMY, N. S.; NATARAJA, H. R.; SAI, K. S.; RAO, N. On the periodic solution for $\ddot{x} + x^{1/(2n+1)} = 0$. *Journal of Sound and Vibration*, v. 261, p. 952–954, 2003.

65 GE, Z. M.; LIN, T. N. Regular and chaotic dynamic analysis and control of an elliptical pendulum on a vibrating basement. *Journal of Sound and Vibration*, v. 230, p. 1045–1068, 2000.

66 PI, Y.-L.; BRADFORD, M. A. Dynamic buckling of shallow pin-ended arches under a sudden central concentrated load. *manuscript*.

67 DYM, C. L.; SHAMES, I. H. *Solid Mechanics: A Variational Approach*. Tokio: McGraw-Hill, 1973.

68 SATHYAMMORTHY, M. *Nonlinear analisys of structures*. Madras, India: C. R. C. Press, 1997.

69 CHIA, C. *Nonlinear analisys of plates*. New York: McGraw-Hill, 1980.

70 PINTO, O. C. *Controle ativo de vibrações não-lineares de estruturas frexíveis*. Tese (Doutorado) — Departamento de Engenharia Civil, Pontifícia Universidade Católica do Rio de Janeiro, 1999.

71 BUFFONI, S. *Estudo da Flambagem de Armaduras Longitudinais em Pilares de Concreto Armado*. Tese (Doutorado) — Departamento de Engenharia Civil, Pontifícia Universidade Católica do Rio de Janeiro, 2004.

72 SOLIMAN, M. S.; GONçALVES, P. B. Chaotic behavior resulting in transient and steady state instabilities of pressure-loaded shallow spherical shells. *Journal of Sound and Vibration*, v. 259, p. 497–512, 2003.

73 BURTON, T. D.; HAMDAN, M. N. Analysis of non-linear autonomous conservative oscillators by a time transformation method. *Journal of Sound and Vibration*, v. 87, n. 4, p. 543–554, 1983.

74 BURTON, T. D. A perturbation method for certain non-linear oscillators. *International Journal of Non-Linear Mechanics*, v. 19, n. 5, p. 397–407, 1984.

75 HE, J.-H. A new perturbation technique which is also valid for large parameters. *Journal of Sound and Vibration*, v. 229, n. 5, p. 1257–1263, 2000.

76 HU, H.; XIONG, Z. G. Comparasion of two lindsted-poincaré-type perturbation methods. *Journal of Sound and Vibration*, v. 278, p. 437–444, 2004.

77 SANCHEZ, N. E. A view to the new perturbation technique valid for large parameters. *js*, v. 282, p. 1309–1316, 2005.

78 NAYFEH, A. H. Resolving controversies in the application of the method of multiple scales and the generalized method of averaging. *Nonlinear Dynamics*, v. 40, p. 61–102, 2005.

79 CARRERA, E. A study on arc-lehgth-type methods and their operation failures illustrated by a simple model. *Computer & Structures*, v. 50, n. 2, p. 217–229, 1994.

80 CRISFIELD, M. A. A fast incremental-iterative solution procedure that handles snap-through. *Computer & Structures*, v. 13, p. 55–62, 1981.

81 RICHARDS, D. *Advanced Mathematical Methods with Maple*. Cambridge: Cambridge University Press, 2002.

82 LAU, S. L.; CHEUNG, Y. K.; WU, S. Amplitude incremental variational principle for nonlinear vibration of elastic systems. *Journal of Applied Mechanics*, v. 48, p. 959–964, 1981.

83 RAND, R. H. Lecture notes on nonlinear vibrations. 2003.

84 THOMPSON, J. M. T.; STEWART, H. B. *Nonlinear dynamics and chaos*. Chichester: John Wiley & Sons Ltd., 1986.

85 HAYASHI, C. *Nonlinear Oscillations in Physical Systems*. New Jersey: Princeton University Press, 1964.

86 BOYCE, W. E.; DIPRIMA, R. C. *Elementary Differential Equations and Boundary Value Problems (eighth ed.)*. New York: Wiley Inc., 2005.

87 KREYSZIG, E. *Advanced Engineering Mathematics*. New York: John Wiley & Sons, 1993.

88 QAISI, M. I. A power series approach for the study of periodic motion. *Journal of Sound and Vibration*, v. 196, p. 401–406, 1996.

89 MICKENS, R. E. Iteration method solutions for conservative and limit-cycle $x^{1/3}$ force oscillators. *Journal of Sound and Vibration*, v. 292, p. 964–968, 2006.

90 HORSSEN, W. van. On the periods of the periodic solutions of the non-linear oscillator equation. *Journal of Sound and Vibration*, v. 260, p. 961–964, 2003.

91 AWREJCEWICZ, J.; ANDRIANOV, I. V. Oscillations of non-linear system with restoring force close to $sign(x)$. *Journal of Sound and Vibration*, v. 252, p. 962–966, 2002.

92 LIPSCOMB, T.; MICKENS, R. E. Exact solution to the antisymmetric, constant force oscillator equation. *Journal of Sound and Vibration*, v. 169, p. 138–140, 1994.

93 PILIPCHUK, V. N. An explicit form general solution for oscillators with a non-smooth restoring force $\ddot{x} + sign(x)f(x) = 0$. *Journal of Sound and Vibration*, v. 226, p. 795–798, 1999.

94 POTTI, P. K. G.; SARMA, M. S.; RAO, B. N. On the exact periodic solution for $\ddot{x} + sign(x) = 0$. *Journal of Sound and Vibration*, v. 220, p. 380–383, 1999.

95 HU, H.; XIONG, Z. G. Oscillations in a $x^{(2m+2)/(2n+1)}$ potencial. *Journal of Sound and Vibration*, v. 259, p. 977–980, 2003.

96 HU, H. Solutions of nonlinear oscillators with fractional powers by an iteration procedure. *Journal of Sound and Vibration*, v. 294, p. 608–614, 2006.

97 RAMOS, J. I. Piecewise-linearized methods for oscillators with fractional-power nonlinearities. *Journal of Sound and Vibration*, v. 300, p. 502–521, 2007.

98 BELéNDEZ, A.; PASCUAL, C.; ORTUñO, M.; BELéNDEZ, T.; GALLEGO. Application of he's homotopy perturbation method to obtain higher-order approximations to a nonlinear oscillator with discontinuities. *Nonlinear Analysis: Real world Applications*, 2007.

A
Programa em Maple: Lindsted Poincaré modificado - vibração forçada

```
> restart;
```

Ordem da solução

```
> nt:=2;
```

$$nt := 2$$

A.1
Rotinas do método da perturbação

```
> solucao_aproximada_frequencia:=proc(pot1,pot2,epsilon)
> global eq,omega,omega0,_X,i,__omega;
> __omega:=Omega^pot1:
> for i from 1 to pot2 do
> __omega:=__omega-epsilon^i*_e[i];
> od;
> eq:=subs(omega0^pot1=__omega,eq);
> end proc:
> solucao_aproximada_tempo:=proc(pot,epsilon)
> global eq,tau,X,XX,__x,i;
> X:=0: XX:=0:
> for i from 0 to pot do
> X:=X+epsilon^i*__x[i](t);
> XX:=XX+epsilon^i*__x[i];
> od;
> eq:=subs(x(t)=X,eq);
> end proc:
```

```
> monta_equacoes:=proc(pot)
> global eq,_eq; local i,j,AA,eq1;
> eq1:=expand(eq):
> for i from pot by -1 to 1 do
> eq1:=subs((mu^i)=AA[i],eq1): _eq[i]:=diff(eq1,AA[i]);
AA[i]:=0;
> od;
> _eq[0]:=eq1;
> for i from 1 to pot do
> for j from 0 to i-1 do
> _eq[i]:=subs(__x[j](t)=xxx[j](t),_eq[i]);
> od:
> od:
> end proc:
> resolva:=proc(i)
> global _eq,__x,t;
> __x[i]:=rhs(dsolve(_eq[i],__x[i](t)));
> end proc:
> resolva_eqd:=proc(eq_,ivar_)
> global omega,t,__x,solu;
> local eq1,eq2,i,ii,AAAA,BBBB,D1,D2,solu1,op2;
> ii:=nops(op(1,eq_))-2;
> op2:=op(2,eq_);
> eq1:=subs({diff(__x[ivar_](t),t$2)=AAAA,__x[ivar_](t)=BBBB
> },op(1,eq_)):
> D2:=diff(eq1,AAAA);
> D1:=diff(eq1,BBBB);
> eq1:=subs({AAAA=0,BBBB=0},eq1)=op2:
> solu:=0:
> if(ii>1)then
> for i from 1 to ii do
> eq2:=D2*diff(__x[ivar_](t),`$`(t,2))+D1*__x[ivar_](t)=
-op(i,op(1,eq1));
> solu1:=rhs(dsolve(eq2,__x[ivar_](t)));
> if(i>1)then
> solu1:=subs(cat(_C,2*ind-1)=0,cat(_C,2*ind)=0,solu1);
> end if:
> solu:=solu+solu1;
> od;
> else
> eq2:=D2*diff(__x[ivar_](t),`$`(t,2))+D1*__x[ivar_](t)=
-op(1,eq1);
> rhs(dsolve(eq2,__x[ivar_](t)));
> solu:=solu+%;
> end if;
> solu;
> end proc:
```

Rotina para resolver equações diferenciais lineares com muitos termos não-homogeneos.

```
> resolva_eqd_old:=proc(eq_,ivar_)
> global Omega,t,__x,solu,ind,SIMPLIFY;
> local eq1,eq2,i,ii,CCC,DDD,v1,v2,solu1;
> eq1:=subs(diff(__x[ivar_](t),t$2)=CCC,op(1,eq_)):
> v1:=diff(eq1,CCC);
> eq1:=subs(__x[ivar_](t)=DDD,eq1):
> v2:=diff(eq1,DDD);
> eq1:=subs(CCC=0,DDD=0,eq1);
> eq1:=SIMPLIFY(eq1);
> ii:=nops(eq1);
> solu:=0:
> if(ii>1)then
> for i from 1 to ii do
> eq2:=v1*diff(__x[ivar_](t),`$`(t,2))+v2*__x[ivar_](t)=
-op(i,eq1);
> solu1:=rhs(dsolve(eq2,__x[ivar_](t)));
> if(i>1)then
> solu1:=subs(cat(_C,2*ind-1)=0,cat(_C,2*ind)=0,solu1);
> end if:
> solu:=solu+%;
> od;
> else
> eq2:=v1*diff(__x[ivar_](t),`$`(t,2))+v2*__x[ivar_](t)=-eq1;
> rhs(dsolve(eq2,__x[ivar_](t)));
> solu:=solu+%;
> end if;
> solu;
> end proc:
> elimina_secular_term:=proc(i)
> global Omega,t,__x,_e; local AA,AAA;
> subs(Omega*t=AAA,__x[i]);
> diff(%,t);
> subs(sin(AAA)=AA,%); diff(%,AA);
> _e[i]:=rhs(isolate(simplify(%=0),_e[i]));
> end proc:
> aplica_condicoes_iniciais:=proc(eq_,icte)
> global __x,_C,t;
> t:=0;
> solve(eq_,cat('_C',icte));
> end proc:
> EXPANDE:=proc(eqq)
> subs({sin=SINN,cos=COSS},eqq);
> expand(%);
> subs({SINN=sin,COSS=cos},%);
> end proc:
```

A.2
Equação de Duffing

```
> printlevel:=2;
```

$$printlevel := 2$$

```
> eqd:=diff(x(t),t$2)+2*mu*zeta*omega0*diff(x(t),t)+
omega0^2*x(t)+alpha*x(t)^2+beta*x(t)^3=mu*F(t);
```

$$eqd := (\tfrac{d^2}{dt^2}\,\mathrm{x}(t)) + 2\,\mu\,\zeta\,\omega 0\,(\tfrac{d}{dt}\,\mathrm{x}(t)) + \omega 0^2\,\mathrm{x}(t) + \alpha\,\mathrm{x}(t)^2 + \beta\,\mathrm{x}(t)^3 = \mu\,\mathrm{F}(t)$$

Analisando a nao linearidade quadrática

```
> beta:=0; zeta:=0; eqd:=subs(F(t)=F*cos(Omega*t),eqd);
```

$$\beta := 0$$

$$\zeta := 0$$

$$eqd := (\tfrac{d^2}{dt^2}\,\mathrm{x}(t)) + \omega 0^2\,\mathrm{x}(t) + \alpha\,\mathrm{x}(t)^2 = \mu\,F\cos(\Omega\,t)$$

```
> eq:=eqd: mu:=alpha:
> solucao_aproximada_frequencia(2,nt,mu);
```

$$(\tfrac{d^2}{dt^2}\,\mathrm{x}(t)) + (\Omega^2 - \alpha\,\_e_1 - \alpha^2\,\_e_2)\,\mathrm{x}(t) + \alpha\,\mathrm{x}(t)^2 = \alpha\,F\cos(\Omega\,t)$$

Freqüência da resposta - Ω.

```
> i:='i':
> Omega^2=omega0^2+sum(mu^i*_e[i],'i'=1..nt);
> omega0^2=Omega^2-sum(mu^i*_e[i],'i'=1..nt);
```

$$\Omega^2 = \omega 0^2 + \alpha\,\_e_1 + \alpha^2\,\_e_2$$

$$\omega 0^2 = \Omega^2 - \alpha\,\_e_1 - \alpha^2\,\_e_2$$

```
> eq:=expand(eq);
```

$$eq := (\tfrac{d^2}{dt^2}\,\mathrm{x}(t)) + \mathrm{x}(t)\,\Omega^2 - \mathrm{x}(t)\,\alpha\,\_e_1 - \mathrm{x}(t)\,\alpha^2\,\_e_2 + \alpha\,\mathrm{x}(t)^2 = \alpha\,F\cos(\Omega\,t)$$

```
> solucao_aproximada_tempo(nt,mu);
```

$$(\tfrac{\partial^2}{\partial t^2}\,\%1) + \%1\,\Omega^2 - \%1\,\alpha\,\_e_1 - \%1\,\alpha^2\,\_e_2 + \alpha\,\%1^2 = \alpha\,F\cos(\Omega\,t)$$

$$\%1 := \_\_x_0(t) + \alpha\,\_\_x_1(t) + \alpha^2\,\_\_x_2(t)$$

```
> eqd;
```

$$(\tfrac{d^2}{dt^2}\,\mathrm{x}(t)) + \omega 0^2\,\mathrm{x}(t) + \alpha\,\mathrm{x}(t)^2 = \alpha\,F\cos(\Omega\,t)$$

```
> X;
```

$$\_\_x_0(t) + \alpha\,\_\_x_1(t) + \alpha^2\,\_\_x_2(t)$$

```
> eq:=expand(eq):
> monta_equacoes(nt):
> for i from 0 to nt do
> _eq[i]; od;
```

$$(\tfrac{d^2}{dt^2}\,\_\_x_0(t)) + \Omega^2\,\_\_x_0(t) = 0$$

$$(\tfrac{d^2}{dt^2}\,\_\_x_1(t)) + \Omega^2\,\_\_x_1(t) - \_e_1\,xxx_0(t) + xxx_0(t)^2 = F\cos(\Omega\,t)$$

$$(\tfrac{d^2}{dt^2}\,\_\_x_2(t)) + \Omega^2\,\_\_x_2(t) - \_e_1\,xxx_1(t) - \_e_2\,xxx_0(t) + 2\,xxx_0(t)\,xxx_1(t) = 0$$

```
> eqd:
> subs(x(t)=X,eqd):
> auxaux:=collect(expand(%),mu):
> for i from 1 to nt do
> eqaux[i]:=subs(__x[i](t)=0,op(1,_eq[i]));
> eqaux[i]:=expand(eqaux[i])-op(2,_eq[i]);
> od:
```

A.2.1
Resolve as equações

```
> RETIRA_TERMOS_SECULARES:=proc()
> global __x,ind,t,_e,eq,eq1,eq2;
> local i,AAA,BBB,solu;
> eq:=__x[ind]:
> for i from 1 to 10*ind do
> eq:=subs(sin(i*Omega*t)=AAA[i],eq);
> eq:=subs(cos(i*Omega*t)=BBB[i],eq);
> od:
> solu[1]:=0: solu[2]:=0:
> eq:=diff(eq,t);
> #SENOS
> eq1:=diff(eq,AAA[1]);
> if(eq1<>0)then
> solu[1]:=expand(solve(eq1=0,_e[ind]));
> end if;
> #COSSENOS
> eq2:=diff(eq,BBB[1]);
> if(eq2<>0)then
> solu[2]:=expand(solve(eq2=0,_e[ind]));
> end if;
> _e[ind]:=solu[1];
> if(expand(eq1)=0)then
> if(expand(eq2)<>0)then
> _e[ind]:=solu[2];
> end if;
> else
> _e[ind]:=solu[2];
> end if;
> end proc:
> unassign('_e'); unassign('__x');
> ind:=0;
> __x[ind]:=a*subs({mu=1,F=1},op(2,eqd));
> __v[ind]:=diff(__x[ind],t);
```

$$ind := 0$$

$$__x_0 := a\cos(\Omega\, t)$$

$$__v_0 := -a\sin(\Omega\, t)\,\Omega$$

```
> printlevel:=2;
```

```
> for ind from 1 to nt do
> cat("EQUACAO ",ind);
> cat("SIMPLIFICANDO A EQUACAO ",ind);
> eq:=expand(eqaux[ind]);
> eq:=subs({seq(xxx[j](t)=__x[j],j=0..ind-1)},eq);
> _eq[ind]:=diff(__x[ind](t),t$2)+Omega^2*__x[ind](t)+
EXPANDE(combine(e
> q,trig))=0;
> "RESOLVENDO A EQUACAO";
> __x[ind]:=resolva_eqd(_eq[ind],ind);
> "IMPONDO AS CONDICOES INICIAIS";
> __x[ind]:=subs({_C1=0,_C2=0},__x[ind]);
> cat("SIMPLIFICANDO A SOLUCAO ",ind);
> __x[ind]:=collect(EXPANDE(__x[ind]),{sin,cos});
> __v[ind]:=diff(__x[ind],t);
> "RETIRANDO OS TERMOS SECULARES";
> RETIRA_TERMOS_SECULARES();
> __x[ind]:=EXPANDE(__x[ind]);
> od;
```

$$printlevel := 2$$

"EQUACAO 1"

"SIMPLIFICANDO A EQUACAO 1"

$$eq := -\_e_1\, xxx_0(t) + xxx_0(t)^2 - F\cos(\Omega\, t)$$

$$eq := -\_e_1\, a\cos(\Omega\, t) + a^2\cos(\Omega\, t)^2 - F\cos(\Omega\, t)$$

$$\_eq_1 := (\tfrac{d^2}{dt^2}\ \_\_x_1(t)) + \Omega^2\ \_\_x_1(t) - \_e_1\, a\cos(\Omega\, t) + \frac{1}{2}\, a^2\cos(2\,\Omega\, t) + \frac{a^2}{2} - F\cos(\Omega\, t) = 0$$

"RESOLVENDO A EQUACAO"

$$\_\_x_1 := \sin(\Omega\, t)\ \_C2 + \cos(\Omega\, t)\ \_C1 + \frac{1}{2}\,\frac{\_e_1\, a\,(\cos(\Omega\, t) + \sin(\Omega\, t)\,\Omega\, t)}{\Omega^2} + \frac{1}{6}\,\frac{a^2\cos(2\,\Omega\, t)}{\Omega^2} - \frac{a^2}{2\,\Omega^2} + \frac{1}{2}\,\frac{F\,(\cos(\Omega\, t) + \sin(\Omega\, t)\,\Omega\, t)}{\Omega^2}$$

"IMPONDO AS CONDICOES INICIAIS"

$$\_\_x_1 := \frac{1}{2}\,\frac{\_e_1\, a\,(\cos(\Omega\, t) + \sin(\Omega\, t)\,\Omega\, t)}{\Omega^2} + \frac{1}{6}\,\frac{a^2\cos(2\,\Omega\, t)}{\Omega^2} - \frac{a^2}{2\,\Omega^2} + \frac{1}{2}\,\frac{F\,(\cos(\Omega\, t) + \sin(\Omega\, t)\,\Omega\, t)}{\Omega^2}$$

"SIMPLIFICANDO A SOLUCAO 1"

$$\_\_x_1 := (\frac{1}{2}\,\frac{\_e_1\, a\, t}{\Omega} + \frac{F\, t}{2\,\Omega})\sin(\Omega\, t) + (\frac{1}{2}\,\frac{\_e_1\, a}{\Omega^2} + \frac{F}{2\,\Omega^2})\cos(\Omega\, t) + \frac{1}{6}\,\frac{a^2\cos(2\,\Omega\, t)}{\Omega^2} - \frac{a^2}{2\,\Omega^2}$$

$$--v_1 := (\frac{1}{2}\,\frac{-e_1\,a}{\Omega} + \frac{F}{2\,\Omega})\sin(\Omega\,t) + (\frac{1}{2}\,\frac{-e_1\,a\,t}{\Omega} + \frac{F\,t}{2\,\Omega})\cos(\Omega\,t)\,\Omega$$
$$-(\frac{1}{2}\,\frac{-e_1\,a}{\Omega^2} + \frac{F}{2\,\Omega^2})\sin(\Omega\,t)\,\Omega - \frac{1}{3}\,\frac{a^2\sin(2\,\Omega\,t)}{\Omega}$$

"RETIRANDO OS TERMOS SECULARES"

$$--x_1 := \frac{1}{6}\,\frac{a^2\cos(2\,\Omega\,t)}{\Omega^2} - \frac{a^2}{2\,\Omega^2}$$

"EQUACAO 2"

"SIMPLIFICANDO A EQUACAO 2"

$$eq := \frac{F\,xxx_1(t)}{a} - \_e_2\,xxx_0(t) + 2\,xxx_0(t)\,xxx_1(t)$$

$$eq := \frac{F\,(\frac{1}{6}\,\frac{a^2\cos(2\,\Omega\,t)}{\Omega^2} - \frac{a^2}{2\,\Omega^2})}{a} - \_e_2\,a\cos(\Omega\,t)$$
$$+2\,a\cos(\Omega\,t)\,(\frac{1}{6}\,\frac{a^2\cos(2\,\Omega\,t)}{\Omega^2} - \frac{a^2}{2\,\Omega^2})$$

$$\_eq_2 := (\tfrac{d^2}{dt^2}\,--x_2(t)) + \Omega^2\,--x_2(t) + \frac{1}{6}\,\frac{F\,a\cos(2\,\Omega\,t)}{\Omega^2} - \frac{F\,a}{2\,\Omega^2}$$
$$-\_e_2\,a\cos(\Omega\,t) - \frac{5}{6}\,\frac{a^3\cos(\Omega\,t)}{\Omega^2} + \frac{1}{6}\,\frac{a^3\cos(3\,\Omega\,t)}{\Omega^2} = 0$$

"RESOLVENDO A EQUACAO"

$$--x_2 := 5\sin(\Omega\,t)\,\_C2 + 5\cos(\Omega\,t)\,\_C1 + \frac{1}{18}\,\frac{F\,a\cos(2\,\Omega\,t)}{\Omega^4} + \frac{F\,a}{2\,\Omega^4}$$
$$+\frac{1}{2}\,\frac{-e_2\,a\,(\cos(\Omega\,t) + \sin(\Omega\,t)\,\Omega\,t)}{\Omega^2} + \frac{5}{12}\,\frac{a^3\,(\cos(\Omega\,t) + \sin(\Omega\,t)\,\Omega\,t)}{\Omega^4}$$
$$+\frac{1}{48}\,\frac{a^3\cos(3\,\Omega\,t)}{\Omega^4}$$

"IMPONDO AS CONDICOES INICIAIS"

$$--x_2 := \frac{1}{18}\,\frac{F\,a\cos(2\,\Omega\,t)}{\Omega^4} + \frac{F\,a}{2\,\Omega^4} + \frac{1}{2}\,\frac{-e_2\,a\,(\cos(\Omega\,t) + \sin(\Omega\,t)\,\Omega\,t)}{\Omega^2}$$
$$+\frac{5}{12}\,\frac{a^3\,(\cos(\Omega\,t) + \sin(\Omega\,t)\,\Omega\,t)}{\Omega^4} + \frac{1}{48}\,\frac{a^3\cos(3\,\Omega\,t)}{\Omega^4}$$

"SIMPLIFICANDO A SOLUCAO 2"

$$--x_2 := (\frac{1}{2}\,\frac{-e_2\,a\,t}{\Omega} + \frac{5\,a^3\,t}{12\,\Omega^3})\sin(\Omega\,t) + (\frac{1}{2}\,\frac{-e_2\,a}{\Omega^2} + \frac{5\,a^3}{12\,\Omega^4})\cos(\Omega\,t)$$
$$+\frac{1}{18}\,\frac{F\,a\cos(2\,\Omega\,t)}{\Omega^4} + \frac{F\,a}{2\,\Omega^4} + \frac{1}{48}\,\frac{a^3\cos(3\,\Omega\,t)}{\Omega^4}$$

$$--v_2 := (\frac{1}{2}\,\frac{-e_2\,a}{\Omega} + \frac{5\,a^3}{12\,\Omega^3})\sin(\Omega\,t) + (\frac{1}{2}\,\frac{-e_2\,a\,t}{\Omega} + \frac{5\,a^3\,t}{12\,\Omega^3})\cos(\Omega\,t)\,\Omega$$
$$-(\frac{1}{2}\,\frac{-e_2\,a}{\Omega^2} + \frac{5\,a^3}{12\,\Omega^4})\sin(\Omega\,t)\,\Omega - \frac{1}{9}\,\frac{F\,a\sin(2\,\Omega\,t)}{\Omega^3} - \frac{1}{16}\,\frac{a^3\sin(3\,\Omega\,t)}{\Omega^3}$$

"RETIRANDO OS TERMOS SECULARES"

$$--x_2 := \frac{1}{18}\,\frac{F\,a\cos(2\,\Omega\,t)}{\Omega^4} + \frac{F\,a}{2\,\Omega^4} + \frac{1}{48}\,\frac{a^3\cos(3\,\Omega\,t)}{\Omega^4}$$

A.2.2
Relação freqüência-deslocamento

```
> for i from 1 to nt do
> _e[i]; j:='j':
> __eq[i]:=omega0^2+sum(mu^j*_e[j],'j'=1..i)-Omega^2;
> V[i]:=__eq[i]:
> H[i]:=diff(V[i],omega);
> od:
> for i from 1 to nt do
> __eq[i];
> od;
```

$$\omega 0^2 - \frac{\alpha\,F}{a} - \Omega^2$$

$$\omega 0^2 - \frac{\alpha\,F}{a} - \frac{5\,\alpha^2\,a^2}{6\,\Omega^2} - \Omega^2$$

```
> unassign('Omega'):
> for i from 1 to nt do isolate(__eq[i],Omega^2); od;
```

$$\Omega^2 = \omega 0^2 - \frac{\alpha\,F}{a}$$

$$\Omega^2 = \omega 0^2 - \frac{\alpha\,F}{a} - \frac{5\,\alpha^2\,a^2}{6\,\Omega^2}$$

A.2.3
Soluções

```
> v0:=0; omega0:='omega0':
> for i from 1 to nt do
> j:='j':
> __eq[i]:=omega0+sum(mu^j*_e[j],'j'=1..i)-Omega;
> __X[i]:=sum(mu^j*__x[j],'j'=0..i);
> od;
```

$$v0 := 0$$

$$j := j$$

$$--eq_1 := \omega 0 - \frac{\alpha\,F}{a} - \Omega$$

$$--X_1 := a\cos(\Omega\,t) + \alpha\,(\frac{1}{6}\,\frac{a^2\cos(2\,\Omega\,t)}{\Omega^2} - \frac{a^2}{2\,\Omega^2})$$

$$j := j$$

$$--eq_2 := \omega 0 - \frac{\alpha\,F}{a} - \frac{5\,\alpha^2\,a^2}{6\,\Omega^2} - \Omega$$

$$\_\_X_2 := a\cos(\Omega\,t) + \alpha\,(\frac{1}{6}\,\frac{a^2\cos(2\,\Omega\,t)}{\Omega^2} - \frac{a^2}{2\,\Omega^2})$$
$$+\,\alpha^2\,(\frac{1}{18}\,\frac{F\,a\cos(2\,\Omega\,t)}{\Omega^4} + \frac{F\,a}{2\,\Omega^4} + \frac{1}{48}\,\frac{a^3\cos(3\,\Omega\,t)}{\Omega^4})$$

A.2.4
Curva de ressonância

```
> A:='A': Omega:='Omega': F:='F': F0:='F0':
> alpha:='alpha': omega0:='omega0':
> __eq[1];
```

$$\omega 0 - \frac{\alpha\,F}{a} - \Omega$$

```
> F0;
```

$$F0$$

```
> F:=F0/mu; __eq[nt];
```

$$F := \frac{F0}{\alpha}$$

$$\omega 0 - \frac{F0}{a} - \frac{5\,\alpha^2\,a^2}{6\,\Omega^2} - \Omega$$

```
> __X[1];
```

$$a\cos(\Omega\,t) + \alpha\,(\frac{1}{6}\,\frac{a^2\cos(2\,\Omega\,t)}{\Omega^2} - \frac{a^2}{2\,\Omega^2})$$

```
> with(Optimization);
> FindAmplitude:=proc(_a)
> global __x,a,mu,t,Omega,XXX;
> local i,T,max,min,dt;
> a:=_a; XXX:=0; max:=0;
> for i from 0 to nt do
> XXX:=XXX+mu^i*__x[i];
> od;
> T:=evalf(2*Pi/Omega);
> max:=Maximize(XXX,t=0..4*T)[1];
> min:=abs(Minimize(XXX,t=0..4*T)[1]);
> dt:=T/500;
> for t from 0 by dt to 4*T do
> if(abs(evalf(XXX))>max)then
> max:=evalf(XXX);
> end if;
> od;
> t:='t':
> if(max>min) then
> max;
> else
> min;
> end if;
> end proc:
```

[*ImportMPS*, *Interactive*, *LPSolve*, *LSSolve*, *Maximize*, *Minimize*, *NLPSolve*, *QPSolve*]

```
> CurvaResonancia:=proc(vi,delta1,vf,parte)
> global __eq,nt,a,Amp,solu,Omega,d,cont,delta,fd,txt;
> local i,j,k,kk,init,RA2,RA3,aux1,soluu;
> Omega:=vi; delta:=delta1;
> txt:=sprintf("CurvaRessonancia_F0=%2.3g_alpha=%2.3g_beta=
> %2.3g_omega0=%2.3g_OmegaxA_nt=%d.dat",
> F0,alpha,beta,omega0,nt);
> fd:=fopen(txt,WRITE,BINARY);
> fprintf(fd,"   Omega      a          Amp          a[1]
> a[2]\n");
> for i from 1 by 1 while Omega<= vf do
> a:='a':
> __eq[nt];
> if(omega0-Omega<>0)then
> soluu:=solve(__eq[nt],a);
> k:=nops([soluu]):
> if(k=1)then
> solu[1]:=soluu;
> else
> solu:=soluu;
> end if;
> for j from 1 to k do
> if(Im(solu[j])=0)then
> Amp:=FindAmplitude(Re(solu[j]));
> aux1:=subs({cos(Omega*t)=AAA,cos(2*Omega*t)=BBB,
> cos(3*Omega*t)=CCC},__X[nt]);
> RA2:=evalf(diff(aux1,BBB)/diff(aux1,AAA));
> RA3:=evalf(diff(aux1,CCC)/diff(aux1,AAA));
> fprintf(fd,"%7.4f %14.10f %14.10f %14.10f
> %14.10f\n",Omega,a,Amp,RA2,RA3);
> end if;
> od;
> end if;
> Omega:=Omega+delta;
> od;
> fclose(fd);
> end proc:
> __X[nt];
```

$$a\cos(\Omega t) + \alpha\left(\frac{1}{6}\frac{a^2\cos(2\Omega t)}{\Omega^2} - \frac{a^2}{2\Omega^2}\right) + \alpha^2\left(\frac{1}{18}\frac{F0\,a\cos(2\Omega t)}{\alpha\Omega^4} + \frac{F0\,a}{2\Omega^4\alpha} + \frac{1}{48}\frac{a^3\cos(3\Omega t)}{\Omega^4}\right)$$

```
> F0:=1; omega0:=1; alpha:=1; F;
```

$$F0 := 1$$
$$\omega 0 := 1$$
$$\alpha := 1$$
$$1$$

```
> __eq[1]; __eq[2];
```

$$1 - \frac{1}{a} - \Omega$$

$$1 - \frac{1}{a} - \frac{5\,a^2}{6\,\Omega^2} - \Omega$$

```
> t:='t':
> eqd;
```

$$(\tfrac{d^2}{dt^2}\,\mathrm{x}(t)) + \mathrm{x}(t) + \mathrm{x}(t)^2 = \cos(\Omega\,t)$$

```
> NT:=nt;
```

$$NT := 2$$

```
> for i from 1 to NT do
> nt:=i; a:='a': t:='t':
> CurvaResonancia(0.01,0.01,2.0);
> od;
```

$$nt := 1$$
$$a := a$$
$$t := t$$
$$nt := 2$$
$$a := a$$
$$t := t$$

```
> t:='t': a:='a': Omega:='Omega';
> __X[1]; collect(__X[2],cos);
```

$$\Omega := \Omega$$

$$a\cos(\Omega\,t) + \frac{1}{6}\,\frac{a^2\cos(2\,\Omega\,t)}{\Omega^2} - \frac{a^2}{2\,\Omega^2}$$

$$a\cos(\Omega\,t) + (\frac{a^2}{6\,\Omega^2} + \frac{a}{18\,\Omega^4})\cos(2\,\Omega\,t) + \frac{a}{2\,\Omega^4} - \frac{a^2}{2\,\Omega^2} + \frac{1}{48}\,\frac{a^3\cos(3\,\Omega\,t)}{\Omega^4}$$

```
> __eq[1]; __eq[2];
```

$$1 - \frac{1}{a} - \Omega$$

$$1 - \frac{1}{a} - \frac{5\,a^2}{6\,\Omega^2} - \Omega$$

B
Programa em Maple: Método de Taylor - vibração livre

```
> restart;
> with(linalg):
```

B.1
Solução

```
> eqd:=diff(u(t),t$2)+omega0^2*u(t)+alpha*u(t)^2+beta*u(t)^3=F(t);
```

$$eqd := (\tfrac{d^2}{dt^2}\,\mathrm{u}(t)) + \omega 0^2\,\mathrm{u}(t) + \alpha\,\mathrm{u}(t)^2 + \beta\,\mathrm{u}(t)^3 = \mathrm{F}(t)$$

```
> eqd:=subs({alpha=0,F(t)=0},eqd);
```

$$eqd := (\tfrac{d^2}{dt^2}\,\mathrm{u}(t)) + \omega 0^2\,\mathrm{u}(t) + \beta\,\mathrm{u}(t)^3 = 0$$

nt = maior potência da série + 1

```
> nt:=4+1;
```

$$nt := 5$$

```
> eqd;
```

$$(\tfrac{d^2}{dt^2}\,\mathrm{u}(t)) + \omega 0^2\,\mathrm{u}(t) + \beta\,\mathrm{u}(t)^3 = 0$$

```
> isolate(%,diff(u(t),`$`(t,2)));
```

$$\tfrac{d^2}{dt^2}\,\mathrm{u}(t) = -\omega 0^2\,\mathrm{u}(t) - \beta\,\mathrm{u}(t)^3$$

```
> u2:=rhs(%);
```

$$u2 := -\omega 0^2\,\mathrm{u}(t) - \beta\,\mathrm{u}(t)^3$$

```
> taylor(u(t),t=0,4);
```

$$\mathrm{u}(0) + \mathrm{D}(u)(0)\,t + \frac{1}{2}\,(\mathrm{D}^{(2)})(u)(0)\,t^2 + \frac{1}{6}\,(\mathrm{D}^{(3)})(u)(0)\,t^3 + \mathrm{O}(t^4)$$

série de taylor

```
> i:='i': unassign('c');
> U:=sum(c[i]*t^i,i=0..nt-1);
```

$$U := c_0 + c_1\,t + c_2\,t^2 + c_3\,t^3 + c_4\,t^4$$

```
> dU:=diff(U,t);
```

$$dU := c_1 + 2\,c_2\,t + 3\,c_3\,t^2 + 4\,c_4\,t^3$$

ausência de amortecimento

```
> v0:=0;
```

$$v0 := 0$$

derivadas da série polinomial

```
> dt[0]:=U;
> for i from 1 to nt-1 do
> dt[i]:=diff(dt[i-1],t);
> od:
```

$$dt_0 := c_0 + c_1\,t + c_2\,t^2 + c_3\,t^3 + c_4\,t^4$$

```
> eq[0]:=u0; eq[1]:=v0; eq[2]:=u2;
> for i from 3 to nt-1 do
> diff(eq[i-1],t):
> eq[i]:=expand(subs(diff(u(t),`$`(t,2))=eq[2],%));
> od:
```

$$eq_0 := u0$$

$$eq_1 := 0$$

$$eq_2 := -\omega 0^2\,\mathrm{u}(t) - \beta\,\mathrm{u}(t)^3$$

```
> for i from 0 to nt-1 do
> EQ[i]:=expand(subs({diff(u(t),t)=v0,u(t)=u0},eq[i]));
> od:
```

coeficientes da série

```
> for i from 0 to nt-1 do
> c[i]:=EQ[i]/i!;
> od:
> U;
> dU;
```

$$u0 + (-\frac{1}{2}\,\omega 0^2\,u0 - \frac{1}{2}\,\beta\,u0^3)\,t^2 + (\frac{1}{24}\,\omega 0^4\,u0 + \frac{1}{6}\,\omega 0^2\,\beta\,u0^3 + \frac{1}{8}\,\beta^2\,u0^5)\,t^4$$

$$2\,(-\frac{1}{2}\,\omega 0^2\,u0 - \frac{1}{2}\,\beta\,u0^3)\,t + 4\,(\frac{1}{24}\,\omega 0^4\,u0 + \frac{1}{6}\,\omega 0^2\,\beta\,u0^3 + \frac{1}{8}\,\beta^2\,u0^5)\,t^3$$

B.2
Relação freqüência-amplitude

```
> omega:='omega':
> subs({t=T/4,v0=0},U)=0:
> EQ:=subs(T=2*Pi/omega,%):
> EQ:=expand(%/u0);
```

$$EQ := 1 - \frac{\pi^2\,\omega 0^2}{8\,\omega^2} - \frac{u0^2\,\pi^2\,\beta}{8\,\omega^2} + \frac{\pi^4\,\omega 0^4}{384\,\omega^4} + \frac{u0^2\,\pi^4\,\omega 0^2\,\beta}{96\,\omega^4} + \frac{u0^4\,\pi^4\,\beta^2}{128\,\omega^4} = 0$$

B.3
Exemplo

```
> u0:='u0': omega0:='omega0': omega:='omega': beta:='beta':
b:='b':
> omega0:=1.2; beta:=0.0001; u0:=0.3;
```

$$\omega 0 := 1.2$$
$$\beta := 0.0001$$
$$u0 := 0.3$$

```
> U;
```

$$0.3 - 0.2160013500\,t^2 + 0.02592064800\,t^4$$

B.3.1
Verificação da solução: integração numérica

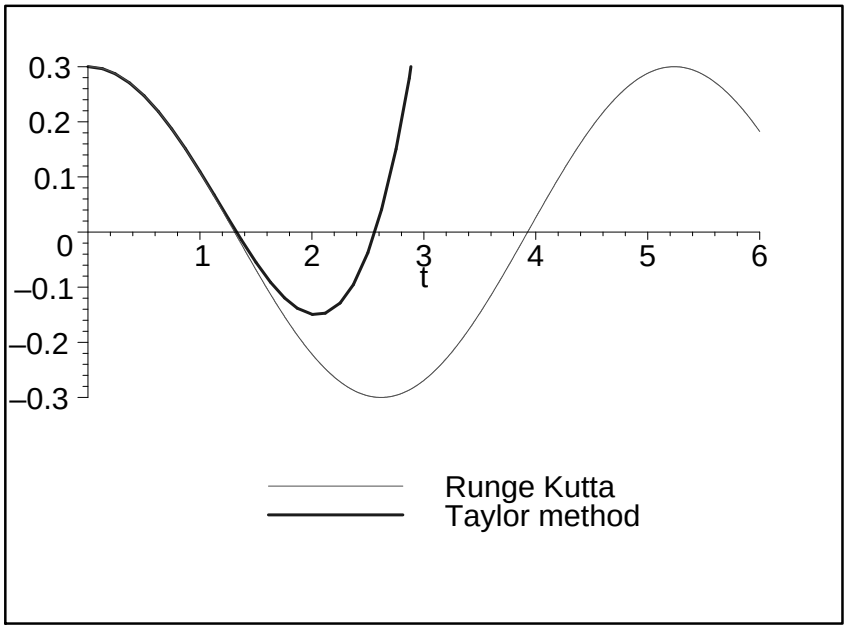
```
> eqd2:=y(t)=diff(x(t),t),diff(y(t),t)=-omega0^2*x(t)-beta*x(t)^3;
> init:=x(0)=u0,y(0)=v0;
```

$$eqd2 := \mathrm{y}(t) = \tfrac{d}{dt}\,\mathrm{x}(t),\ \tfrac{d}{dt}\,\mathrm{y}(t) = -1.44\,\mathrm{x}(t) - 0.0001\,\mathrm{x}(t)^3$$
$$init := \mathrm{x}(0) = 0.3,\ \mathrm{y}(0) = 0$$

```
> f:=dsolve({eqd2,init},type=numeric,method=classical[rk4],
maxfun=999999,output=procedurelist):
> vi:=0; vf:=6;
> #escrevendo na lista
> delta:=0.01: cont:=0:
> for k from vi by delta to vf do
> cont:=cont+1;
> od: cont;
> d1:=array(1..cont,1..2): d2:=array(1..cont,1..2):
> cont:=0:
> for k from vi by delta to vf do
> cont:=cont+1;
> d1[cont,1]:=k;
> d1[cont,2]:=eval(x(t),f(k));
> d2[cont,1]:=k;
> d2[cont,2]:=eval(y(t),f(k));
> od:
> d1g:=convert(d1,listlist):
> d2g:=convert(d2,listlist):
```

$$vi := 0$$
$$vf := 6$$
$$601$$

```
> plot([d1g,U],t=0..vf,y=-u0..u0,color=[red,blue],thickness=
[2,4],legend=["Runge Kutta","Taylor method"]);
```



B.3.2
Transformação da solução em série de Taylor em uma série de Fourier

```
> printlevel:=1;
> unassign('a'); unassign('w');
> unassign('A'); unassign('_omega'); unassign('omega');
> omega0:='omega0': beta:='beta': delta:= 'delta':
> b:='b': u0:='u0':
```

$$printlevel := 1$$

número de harmônicos

```
> nh:=nops(U)-1;#(nt-1)/2;
> i:='i': omega:='omega':
> xx:=sum(A[i]*cos((2*i-1)*omega*t),i=1..nh);
```

$$nh := 2$$

$$xx := A_1 \cos(\omega\, t) + A_2 \cos(3\, \omega\, t)$$

série de cada harmônico

```
> a*cos(x*t);
> taylor(%,t,nt):
> serie:=convert(%,polynom):
```

$$a \cos(x\, t)$$

```
> serie2:=subs({seq((x^2)^i=x2^i,i=1..nt)},serie):
> EXPANDE:=proc(equacao)
> local k,i,soma:
> k:=nops(equacao);
> soma:=0:
> for i from 1 to k do
> soma:=soma+expand(op(i,equacao));
> od;
> soma;
> end proc:
```

substituição dos harmônicos na fórmula da série

```
> for i from 1 to nh do
> for j from 1 to 1 do
> expand(subs(x=(2*i-1)*omega,serie));
> subs(a=A[i],%);
> parte[i]:=%:
> od;
> cat("harmonico ",i),"ok";
> od;
> UHBM:=0:
> for i from 1 to nh do
> UHBM:=UHBM+collect(parte[i],t);
> od:
> UHBM:=collect(UHBM,t):
```

"harmonico 1", "ok"

"harmonico 2", "ok"

```
> xx;
```

$$A_1 \cos(\omega\, t) + A_2 \cos(3\, \omega\, t)$$

primeiras equações

```
> p0:=subs(t=0,U):
> p1:=subs(t=0,UHBM):
> eq[1]:=p0;
> eqhbm[1]:=p1;
```

$$eq_1 := u0$$

$$eqhbm_1 := A_1 + A_2$$

demais equações

```
> cont:=1:
> for j from 2 by 2 to nt-1 do
> if(cont>nh)then break; end if;
> for k from 1 by 1 to 1 do
> cont:=cont+1:
> subs({seq(t^i=0,i=2..j-2),seq(t^i=0,i=j+1..2*nt)},U)-p0:
> eq[cont]:=subs(t=1,%);
> subs({seq(t^i=0,i=2..j-2),seq(t^i=0,i=j+1..nt)},UHBM)-p1:
> eqhbm[cont]:=subs(t=1,%);
> od:
> t^j,cat("equacao ",cont),"ok";
> od;
```

t^2, "equacao 2", "ok"

t^4, "equacao 3", "ok"

```
> neq:=cont;
```

$$neq := 3$$

Determinação das constantes

```
> eq[1]-eqhbm[1]=0;
> A[1]:=solve(%,A[1]);
```

$$u0 - A_1 - A_2 = 0$$

$$A_1 := u0 - A_2$$

```
> expand(eq[2]-eqhbm[2]=0);
```

$$-\frac{\omega 0^2\, u0}{2} - \frac{\beta\, u0^3}{2} + \frac{\omega^2\, u0}{2} + 4\, A_2\, \omega^2 = 0$$

```
> M:=Matrix(nh-1): V:=Matrix(1..nh-1,1):
> for i from 2 to nh do
> cont:=1:
> for j from 2 by 1 to nh do
> M[i-1,cont]:=diff(expand(eq[i]-eqhbm[i]),A[j]);
> cont:=cont+1:
> od:
> V[i-1,1]:=-subs({seq(A[jj]=0,jj=2..nh)},expand(eq[i]
-eqhbm[i]));
> od:
> M;
```

$$\left[\begin{array}{c} 4\,\omega^2 \end{array}\right]$$

```
> V;
```

$$\left[\begin{array}{c} \frac{\omega 0^2\, u0}{2} + \frac{\beta\, u0^3}{2} - \frac{\omega^2\, u0}{2} \end{array}\right]$$

```
> R:=evalm(inverse(M)&*V):
> for i from 2 to nh do
> A[i]:=R[i-1,1];
> od:
```

freqüência da resposta

```
> _eq:=expand(eq[nh+1]-eqhbm[nh+1]):
> omega0:=1.2; beta:=0.0001; u0:=0.3;
```

$$\omega 0 := 1.2$$
$$\beta := 0.0001$$
$$u0 := 0.3$$

```
> aux:=fsolve(_eq,omega);
```

$$aux := -1.200002812,\ -0.4000040625,\ 0.4000040625,\ 1.200002812$$

```
> aux[4];
```

$$1.200002812$$

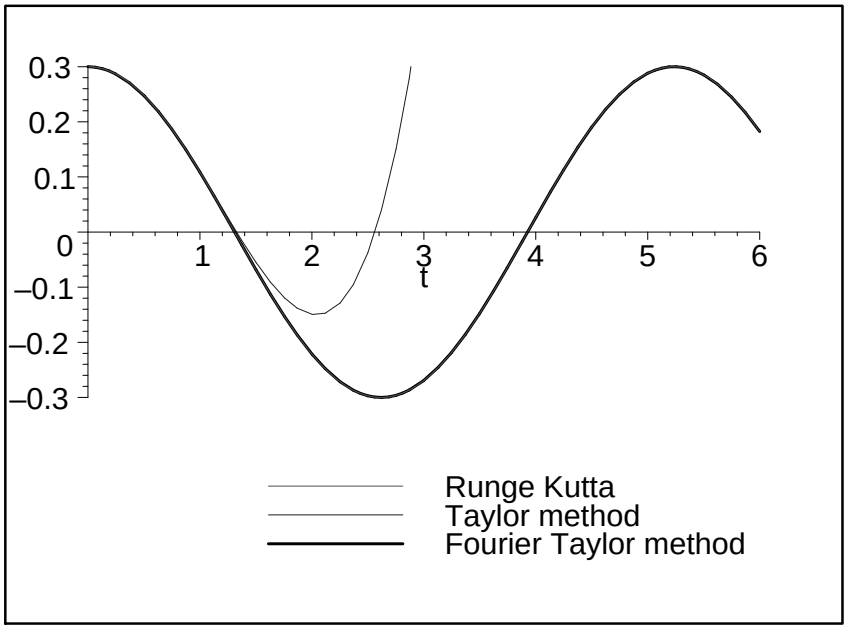
```
> omega:=%;
```

$$\omega := 1.200002812$$

```
> i:='i':
> xxx:=sum(A[i]*cos((2*i-1)*omega*t),i=1..nh);
```

$$xxx := 0.2999999414\cos(1.200002812\, t) + 0.5861083642\, 10^{-7}\cos(3.600008436\, t)$$

```
> plot([d1g,U,xxx],t=0..vf,y=-u0..u0,color=[red,blue,black],
thickness=[2,2,4],legend=["Runge Kutta","Taylor method",
"Fourier Taylormethod"]);
```




```
> u0:='u0': omega0:='omega0':
> omega:='omega': beta:='beta':
```


B.4
Transformação da solução em série de Taylor em uma solução de Lindstedt-Poincaré

```
> printlevel:=1;
> unassign('a'); unassign('w');
> unassign('A'); unassign('_omega'); unassign('omega');
> omega0:='omega0': beta:='beta': delta:= 'delta': u0:='u0':
```

$$printlevel := 1$$

```
> collect(U,{beta,t});
```

$$u0 + (-\frac{1}{2}\,\omega 0^2\,u0 - \frac{1}{2}\,\beta\,u0^3)\,t^2 + (\frac{1}{24}\,\omega 0^4\,u0 + \frac{1}{6}\,\omega 0^2\,\beta\,u0^3 + \frac{1}{8}\,\beta^2\,u0^5)\,t^4$$

```
> nops(%);
```

$$3$$

aproximação até β será correta

máxima potência em beta - se for número de termos-1 faltará uma equação

```
> pb:=%-2;
```

número de harmônicos

```
> nh:=pb+1; i:='i': omega:='omega':
```

retirada das potências de beta que não podem ser representadas corretamente pela solução de LP

```
> Ucopy:=subs({seq(beta^i=0,i=pb+1..10*pb)},U):
> xx:=sum(A[i]*cos((2*i-1)*omega*t),i=1..nh);
```

$$pb := 1$$

$$nh := 2$$

$$xx := A_1 \cos(\omega\, t) + A_2 \cos(3\, \omega\, t)$$

```
> xx2:=xx: ni:=0: j:='j':
> for i from 1 to nh do
> ni:=ni+(pb-(i-1)+1);
> xx2:=subs(A[i]=sum(a[i,j]*beta^j,j=i-1..pb),xx2);
> A[i]:=sum(a[i,j]*beta^j,j=i-1..pb);
> _omega[i]:=(2*i-1)*omega;
> od:
> xx2;
```

$$(a_{1,0} + a_{1,1}\, \beta) \cos(\omega\, t) + a_{2,1}\, \beta \cos(3\, \omega\, t)$$

```
> ni;
```

$$3$$

```
> j:='j':
> "número de incógnitas";
> ni:=ni+pb+1;
> xx2:=subs({omega=sum(w[j]*beta^j,j=0..pb)},xx2);
> omega:=sum(w[j]*beta^j,j=0..pb);
```

"número de incógnitas"

$$ni := 5$$

$$xx2 := (a_{1,0} + a_{1,1}\, \beta) \cos((w_0 + w_1\, \beta)\, t) + a_{2,1}\, \beta \cos(3\, (w_0 + w_1\, \beta)\, t)$$

$$\omega := w_0 + w_1\, \beta$$

```
> unassign('a'); unassign('w');
```

série de cada harmônico

```
> a*cos(x*t);
> taylor(%,t,nt);
> serie:=convert(%,polynom);
```

$$a \cos(x\, t)$$

$$a - \frac{a\, x^2}{2}\, t^2 + \frac{a\, x^4}{24}\, t^4 + \mathrm{O}(t^5)$$

$$serie := a - \frac{1}{2}\, a\, x^2\, t^2 + \frac{1}{24}\, a\, x^4\, t^4$$

```
> serie2:=subs({seq((x^2)^i=x2^i,i=1..nt)},serie);
```

$$serie2 := a - \frac{1}{2}\, a\, x2\, t^2 + \frac{1}{24}\, a\, x2^2\, t^4$$

```
> _omega[2];
```

$$3\, w_0 + 3\, w_1\, \beta$$

```
> pb;
```

$$1$$

```
> EXPANDE:=proc(equacao)
> local k,i,soma:
> k:=nops(equacao);
> soma:=0:
> for i from 1 to k do
> soma:=soma+expand(op(i,equacao));
> od;
> soma;
> end proc:
```

```
> expand(_omega[2]^2);
```

$$9\,{w_0}^2 + 18\,w_0\,w_1\,\beta + 9\,{w_1}^2\,\beta^2$$

```
> subs({seq(beta^i=0,i=pb+1..10*pb)},%);
```

$$9\,{w_0}^2 + 18\,w_0\,w_1\,\beta$$

substituição dos harmônicos na fórmula da série

```
> ULP:=0:
> for i from 1 to nh do
> for j from 1 to 1 do
> #expandindo omega^2
> expand(_omega[i]^2):
> #retirando as potencias em beta que nao interessam
> subs({seq(beta^i=0,i=pb+1..10*pb)},%);
> #substituindo o omega^2 com as potencias em beta que
interessam
> expand(subs(x2=%,serie2));
> #retirando as novas potencias em beta que nao interessam
> subs({seq(beta^i=0,i=pb+1..10*pb)},%);
> #inserindo a amplitude do harmonico
> subs(a=A[i],%);
> #retirando as novas potencias em beta que nao interessam
> parte[i]:=subs({seq(beta^i=0,i=pb+1..10*pb)},expand(%));
> od;
> cat("harmonico ",i),"made";
> od;
> ULP:=0:
> for i from 1 to nh do
> ULP:=ULP+collect(parte[i],t);
> od:
> ULP:=collect(ULP,t):
```

"harmonico 1", "made"

"harmonico 2", "made"

```
> sort(subs({seq(beta^i=0,i=pb+1..10*pb)},ULP),t,ascending);
```

$$a_{1,1}\,\beta + a_{2,1}\,\beta + a_{1,0} + (-\frac{1}{2}\,{w_0}^2\,a_{1,0} - \frac{1}{2}\,{w_0}^2\,a_{1,1}\,\beta - w_0\,w_1\,\beta\,a_{1,0} - \frac{9}{2}\,a_{2,1}\,\beta\,{w_0}^2)\,t^2 + (\frac{1}{24}\,{w_0}^4\,a_{1,0} + \frac{1}{24}\,{w_0}^4\,a_{1,1}\,\beta + \frac{1}{6}\,{w_0}^3\,w_1\,\beta\,a_{1,0} + \frac{27}{8}\,a_{2,1}\,\beta\,{w_0}^4)\,t^4$$

primeiras equações

```
> p0:=subs(t=0,Ucopy):
> p1:=subs(t=0,ULP):
> eq[1]:=p0;
> eqlp[1]:=p1;
```

$$eq_1 := u0$$

$$eqlp_1 := a_{2,1}\,\beta + a_{1,0} + a_{1,1}\,\beta$$

número de incógnitas

```
> ni;
```

5

demais equações

```
> cont:=1:
> for j from 2 by 2 to nt-1 do
> for k from 1 by 1 to 1 do
> cont:=cont+1:
> subs({seq(t^i=0,i=2..j-2),seq(t^i=0,i=j+1..2*nt)},
Ucopy)-p0:
> eq[cont]:=subs(t=1,%);
> subs({seq(t^i=0,i=2..j-2),seq(t^i=0,i=j+1..nt)},ULP)-p1:
> eqlp[cont]:=subs(t=1,%);
> od:
> t^j,cat("equacao ",cont),"ok";
> od;
```

t^2, “equacao 2”, “ok”

t^4, “equacao 3”, “ok”

```
> neq:=cont:
```

B.4.1
Determinação das constantes

```
> unassign('a'); unassign('w');
> i:='i':
> a[1,0]:=u0: w[0]:=omega0:
> printlevel:=1;
> for k from 1 to neq do
> cat("EQUACAO ",k);
> for KKK from 1 to 1 do
> variaveis:=1:
> p0:=subs(beta=0,eq[k]):
> p1:=subs(beta=0,eqlp[k]):
> #potencia beta^0
> if(expand(p0-p1)<>0)then
> if(k=1)then
> isolate(p0=p1,a[k,0]);
> a[k,k-1]:=expand(rhs(%));
> variaveis:=variaveis,sprintf("a[%d,%d]",k,k-1);
> else
> isolate(p0=p1,w[k-2]);
> w[k-2]:=expand(rhs(%));
> variaveis:=variaveis,sprintf("w[%d]",k-2);
> end if;
> end if;
> #potencia beta^j
> for j from 1 to pb do
> aux1:=subs(subs({seq(beta^i=0,i=j+1..pb),beta^j=AAA},
eq[k]))-p0:
> aux2:=subs(subs({seq(beta^i=0,i=j+1..pb),beta^j=AAA},
eqlp[k]))-p1:
> if(subs({AAA=1,beta=0},expand(aux1-aux2))<>0)then
> subs({AAA=1,beta=0},aux1=aux2);
> if(j+1<k)then
> isolate(%,w[j]);
> w[j]:=expand(rhs(%));
> variaveis:=variaveis,sprintf("w[%d]",j);
> else
```

```
> isolate(%,a[k,j]);
> a[k,j]:=expand(rhs(%));
> variaveis:=variaveis,sprintf("a[%d,%d]",k,j);
> end if;
> end if;
> od;
> od;
> "found",variaveis;
> od;
```

$$printlevel := 1$$

"EQUACAO 1"

"found", 1, "a[1,1]"

"EQUACAO 2"

"found", 1, "a[2,1]"

"EQUACAO 3"

"found", 1, "w[1]"

```
> xx2;
```

$$(u0 - \frac{u0^3\,\beta}{32\,\omega0^2})\cos((\omega0 + \frac{3\,u0^2\,\beta}{8\,\omega0})\,t) + \frac{1}{32}\,\frac{u0^3\,\beta\cos(3\,(\omega0 + \frac{3\,u0^2\,\beta}{8\,\omega0})\,t)}{\omega0^2}$$

```
> omega;
```

$$\omega0 + \frac{3\,u0^2\,\beta}{8\,\omega0}$$

solução de primeira ordem obtida com LP tradicional

```
> omega0+3/8*beta/omega0*x0^2-21/256*beta^2/omega0^3*x0^4
+81/2048*beta^3/omega0^5*x0^6;
```

$$\omega0 + \frac{3\,\beta\,x0^2}{8\,\omega0} - \frac{21\,\beta^2\,x0^4}{256\,\omega0^3} + \frac{81\,\beta^3\,x0^6}{2048\,\omega0^5}$$

```
> xx2;
```

$$(u0 - \frac{u0^3\,\beta}{32\,\omega0^2})\cos((\omega0 + \frac{3\,u0^2\,\beta}{8\,\omega0})\,t) + \frac{1}{32}\,\frac{u0^3\,\beta\cos(3\,(\omega0 + \frac{3\,u0^2\,\beta}{8\,\omega0})\,t)}{\omega0^2}$$

```
> U;
```

$$u0 + (-\frac{1}{2}\,\omega0^2\,u0 - \frac{1}{2}\,\beta\,u0^3)\,t^2 + (\frac{1}{24}\,\omega0^4\,u0 + \frac{1}{6}\,\omega0^2\,\beta\,u0^3 + \frac{1}{8}\,\beta^2\,u0^5)\,t^4$$

```
> sprintf("%d termos = %d harmonicos",nops(U),pb+1);
```

"3 termos = 2 harmonicos"

C
Programa em Maple: Fourier-Taylor - vibração forçada amortecida

```
> restart;
> with(linalg):
```

C.1
Solução


```
> eqd:=diff(u(t),t$2)+2*zeta*omega0*diff(u(t),t)+omega0^2*u(t)
+alpha*u(t)^2+beta*u(t)^3=F(t);
```

$$eqd := (\tfrac{d^2}{dt^2}\,\mathrm{u}(t)) + 2\,\zeta\,\omega 0\,(\tfrac{d}{dt}\,\mathrm{u}(t)) + \omega 0^2\,\mathrm{u}(t) + \alpha\,\mathrm{u}(t)^2 + \beta\,\mathrm{u}(t)^3 = \mathrm{F}(t)$$

```
> eqd:=subs({alpha=0,F(t)=F*cos(Omega*t)},eqd);
```

$$eqd := (\tfrac{d^2}{dt^2}\,\mathrm{u}(t)) + 2\,\zeta\,\omega 0\,(\tfrac{d}{dt}\,\mathrm{u}(t)) + \omega 0^2\,\mathrm{u}(t) + \beta\,\mathrm{u}(t)^3 = F\cos(\Omega\,t)$$

```
> nt:=7;
```

$$nt := 7$$

```
> eqd;
```

$$(\tfrac{d^2}{dt^2}\,\mathrm{u}(t)) + 2\,\zeta\,\omega 0\,(\tfrac{d}{dt}\,\mathrm{u}(t)) + \omega 0^2\,\mathrm{u}(t) + \beta\,\mathrm{u}(t)^3 = F\cos(\Omega\,t)$$

```
> isolate(%,diff(u(t),`$`(t,2)));
```

$$\tfrac{d^2}{dt^2}\,\mathrm{u}(t) = F\cos(\Omega\,t) - 2\,\zeta\,\omega 0\,(\tfrac{d}{dt}\,\mathrm{u}(t)) - \omega 0^2\,\mathrm{u}(t) - \beta\,\mathrm{u}(t)^3$$

```
> u2:=rhs(%);
```

$$u2 := F\cos(\Omega\,t) - 2\,\zeta\,\omega 0\,(\tfrac{d}{dt}\,\mathrm{u}(t)) - \omega 0^2\,\mathrm{u}(t) - \beta\,\mathrm{u}(t)^3$$

```
> diff(u2,t$2);
```

$$-F\cos(\Omega\,t)\,\Omega^2 - 2\,\zeta\,\omega 0\,(\tfrac{d^3}{dt^3}\,\mathrm{u}(t)) - \omega 0^2\,(\tfrac{d^2}{dt^2}\,\mathrm{u}(t)) - 6\,\beta\,\mathrm{u}(t)\,(\tfrac{d}{dt}\,\mathrm{u}(t))^2 - 3\,\beta\,\mathrm{u}(t)^2\,(\tfrac{d^2}{dt^2}\,\mathrm{u}(t))$$

```
> c[0]:=u0;
> c[1]:=v0;
> c[2]:=u2;
> for i from 3 to nt do
> c[i]:=diff(c[i-1],t);
> od:
```

$$c_0 := u0$$
$$c_1 := v0$$

$$c_2 := F\cos(\Omega\, t) - 2\,\zeta\,\omega 0\,(\tfrac{d}{dt}\,\mathrm{u}(t)) - \omega 0^2\,\mathrm{u}(t) - \beta\,\mathrm{u}(t)^3$$

```
> c0[0]:=u0;
> c0[1]:=v0;
> c0[2]:=subs({u(t)=u0,diff(u(t),t)=v0,
> cos(Omega*t+phi)=cos(phi),sin(Omega*t+phi)=sin(phi),
> cos(Omega*t)=1,sin(Omega*t)=0},u2);
```

$$c0_0 := u0$$

$$c0_1 := v0$$

$$c0_2 := F - 2\,\zeta\,\omega 0\,v0 - \omega 0^2\,u0 - \beta\,u0^3$$

```
> for i from 3 to nt do
> c0[i]:=expand(subs({u(t)=u0,diff(u(t),t)=v0,
> cos(Omega*t+phi)=cos(phi),sin(Omega*t+phi)=sin(phi),
> cos(Omega*t)=1,sin(Omega*t)=0,seq(diff(u(t),`$`(t,j))=
du[j](u0,v0,Omega),j=2..i-1)},c[i]));
> od:
```

série de taylor

```
> i:='i':
> U:=sum(c0[i]*t^i/i!,i=0..nt-1):
```


C.1.1
Transforma a solução em série de Taylor em uma série de Fourier

```
> printlevel:=1:
> unassign('a'); unassign('w');
> unassign('A'); unassign('B'); unassign('_omega');
> unassign('omega'); unassign('Omega');
> omega0:='omega0': beta:='beta': delta:= 'delta': F:='F':
zeta:='zeta':
> F0:='F0':
> b:='b': u0:='u0': phi:='phi': v0:='v0':
> nops(U)-2: (%-1)/2: round(%-.01):
```

número de harmônicos

```
> nh:=%;
> i:='i': omega:='omega':
> xx:=sum(A[i]*cos((2*i-1)*Omega*t)+B[i]*sin((2*i-1)*Omega*t)
,i=1..nh);
```

$$nh := 2$$

$$xx := A_1\cos(\Omega\, t) + B_1\sin(\Omega\, t) + A_2\cos(3\,\Omega\, t) + B_2\sin(3\,\Omega\, t)$$

```
> a*cos(x*t);
> taylor(%,t,nt):
> serie:=convert(%,polynom):
```

$$a\cos(x\, t)$$

```
> serie2:=subs({seq((x^2)^i=x2^i,i=1..nt)},serie):
> b*sin(x*t);
> taylor(%,t,nt):
> serie_s:=convert(%,polynom):
```

$$b\sin(x\, t)$$

```
> serie2_s:=subs({seq((x^3)^i=x3^i,i=1..nt)},serie_s):
```

```
> EXPANDE:=proc(equacao)
> local k,i,soma:
> k:=nops(equacao);
> soma:=0:
> for i from 1 to k do
> soma:=soma+expand(op(i,equacao));
> od;
> soma;
> end proc:
> for i from 1 to nh do
> for j from 1 to 1 do
> expand(subs(x=(2*i-1)*Omega,serie));
> subs(a=A[i],%);
> parte[i]:=%:
> expand(subs(x=(2*i-1)*Omega,serie_s));
> subs(b=B[i],%);
> parte[i]:=parte[i]+%:
> od;
> cat("harmonico ",i),"ok";
> od;
> UHBM:=0:
> for i from 1 to nh do
> UHBM:=UHBM+collect(parte[i],t);
> od:
> UHBM:=collect(UHBM,t):
```

"harmonico 1", "ok"

"harmonico 2", "ok"

```
> xx;
```

$$A_1\cos(\Omega\,t)+B_1\sin(\Omega\,t)+A_2\cos(3\,\Omega\,t)+B_2\sin(3\,\Omega\,t)$$

primeiras equações

```
> p0:=subs(t=0,U):
> p1:=subs(t=0,UHBM):
> eq[1]:=p0;
> eqhbm[1]:=p1;
```

$$eq_1 := u0$$

$$eqhbm_1 := A_1 + A_2$$

demais equações

```
> cont:=1:
> for j from 1 by 1 to nt do
> if(cont>2*nh+1)then break; end if;
> for k from 1 by 1 to 1 do
> cont:=cont+1:
> subs({seq(t^i=0,i=j+1..nt),t^j=1},U)-p0:
> eq[cont]:=subs(t=0,%);
> subs({seq(t^i=0,i=j+1..nt),t^j=1},UHBM)-p1:
> eqhbm[cont]:=subs(t=0,%);
> od:
> t^j,cat("equacao ",cont),"ok";
> od;
```

t, "equacao 2", "ok"

t^2, "equacao 3", "ok"

t^3, "equacao 4", "ok"

$$t^4, \text{“equacao 5”}, \text{“ok”}$$
$$t^5, \text{“equacao 6”}, \text{“ok”}$$

```
> neq:=cont;
```

$$neq := 6$$

```
> for i from 1 to 5 do
> eq[i]=eqhbm[i];
> od;
```

$$u0 = A_1 + A_2$$
$$v0 = B_1\,\Omega + 3\,B_2\,\Omega$$
$$\frac{F}{2} - \zeta\,\omega0\,v0 - \frac{\omega0^2\,u0}{2} - \frac{\beta\,u0^3}{2} = -\frac{1}{2}\,A_1\,\Omega^2 - \frac{9}{2}\,A_2\,\Omega^2$$
$$-\frac{1}{3}\,\zeta\,\omega0\,du_2(u0,\,v0,\,\Omega) - \frac{\omega0^2\,v0}{6} - \frac{\beta\,u0^2\,v0}{2} = -\frac{1}{6}\,B_1\,\Omega^3 - \frac{9}{2}\,B_2\,\Omega^3$$
$$-\frac{F\,\Omega^2}{24} - \frac{1}{12}\,\zeta\,\omega0\,du_3(u0,\,v0,\,\Omega) - \frac{1}{24}\,\omega0^2\,du_2(u0,\,v0,\,\Omega) - \frac{\beta\,u0\,v0^2}{4}$$
$$-\frac{1}{8}\,\beta\,u0^2\,du_2(u0,\,v0,\,\Omega) = \frac{1}{24}\,A_1\,\Omega^4 + \frac{27}{8}\,A_2\,\Omega^4$$

```
> xx2:=subs({seq(A[i]=cc[i](u0,v0,Omega),i=0..nh),
> seq(B[i]=dd[i](u0,v0,Omega),i=1..nh)},xx);
```

$$xx2 := cc_1(u0,\,v0,\,\Omega)\cos(\Omega\,t) + dd_1(u0,\,v0,\,\Omega)\sin(\Omega\,t) + cc_2(u0,\,v0,\,\Omega)\cos(3\,\Omega\,t) + dd_2(u0,\,v0,\,\Omega)\sin(3\,\Omega\,t)$$

Determinação das amplitudes

```
> nh;
```

$$2$$

```
> nops(xx);
```

$$4$$

```
> neq;
```

$$6$$

```
> M:=Matrix(2*nh): V:=Matrix(1..2*nh,1):
> for i from 1 to neq-2 do
> cont:=1:
> for j from 1 by 1 to nh do
> M[i,cont]:=diff(expand(eq[i]-eqhbm[i]),B[j]);
> cont:=cont+1:
> M[i,cont]:=diff(expand(eq[i]-eqhbm[i]),A[j]);
> cont:=cont+1:
> od:
> V[i,1]:=-subs({seq(A[jj]=0,jj=0..nh),
> seq(B[jj]=0,jj=1..nh)},expand(eq[i]-eqhbm[i]));
> od:
> cont;
```

$$5$$

```
> M;
```

$$\begin{bmatrix} 0 & -1 & 0 & -1 \\ -\Omega & 0 & -3\,\Omega & 0 \\ 0 & \frac{\Omega^2}{2} & 0 & \frac{9\,\Omega^2}{2} \\ \frac{\Omega^3}{6} & 0 & \frac{9\,\Omega^3}{2} & 0 \end{bmatrix}$$

```
> R:=evalm(inverse(M)&*V):
> cont:=1:
> for i from 1 by 2 to 2*nh do
> B[cont]:=R[i,1];
> A[cont]:=R[i+1,1];
> cont:=cont+1:
> od:
> _eq[1]:=eq[2*nh+1]-eqhbm[2*nh+1]:
> _eq[2]:=eq[2*nh+2]-eqhbm[2*nh+2]:
```

Equações que determinam os pontos fixos

```
> _eq[3]:=(u0-u0_0)^2+(v0-v0_0)^2+(Omega-Omega_0)^2-r^2;
> HH2:=Matrix([[diff(_eq[1],u0),diff(_eq[1],v0)],
> [diff(_eq[2],u0),diff(_eq[2],v0)]]):
> VV2:=Matrix([[-_eq[1]],[-_eq[2]]]):
> HH3:=Matrix(
> [[diff(_eq[1],u0),diff(_eq[1],v0),diff(_eq[1],Omega)],
> [diff(_eq[2],u0),diff(_eq[2],v0),diff(_eq[2],Omega)],
> [diff(_eq[3],u0),diff(_eq[3],v0),diff(_eq[3],Omega)]]):
> VV3:=Matrix([[-_eq[1]],[-_eq[2]],[-_eq[3]]]):
```

$$\_eq_3 := (u0 - u0\_0)^2 + (v0 - v0\_0)^2 + (\Omega - Omega\_0)^2 - r^2$$

```
> EXCUTE_NR:=proc(naprox)
> global HH2,VV2,u0,v0,DELTA;
> local i;
> for i from 1 to naprox do
> DELTA:=evalm(inverse(evalf(HH2))&*VV2);
> u0:=u0+evalf(DELTA[1,1]);
> v0:=v0+evalf(DELTA[2,1]);
> od:
> "residuo",evalf(DELTA[1,1]),evalf(DELTA[2,1]);
> end proc:
> EXCUTE_NR3:=proc(naprox)
> global HH3,VV3,u0,phi,Omega,DELTA,u00,phi0,Omega0,r;
> local i;
> for i from 1 to naprox do
> DELTA:=evalm(inverse(evalf(HH3))&*VV3);
> u0:=u0+evalf(DELTA[1,1]);
> phi:=phi+evalf(DELTA[2,1]);
> Omega:=Omega+evalf(DELTA[3,1]);
> od:
> "residuo",evalf(DELTA[1,1]),evalf(DELTA[2,1]),
> evalf(DELTA[3,1]);
> end proc:
```

```
> for i from 1 to 2 do
> aux:=HH2[i,1]:
> for j from 1 to nt do
> aux:=subs(diff(du[j](u0,v0,Omega),u0)=duu0[j],aux);
> od:
> HH2[i,1]:=subs({seq(du[ii](u0,v0,Omega)=du[ii],
> ii=1..nt)},aux):
> aux:=HH2[i,2]:
> for j from 1 to nt do
> aux:=subs(diff(du[j](u0,v0,Omega),v0)=duv0[j],aux);
> od:
> HH2[i,2]:=subs({seq(du[ii](u0,v0,Omega)=du[ii],
> ii=1..nt)},aux):
> VV2[i,1]:=subs({seq(du[ii](u0,v0,Omega)=du[ii],ii=1..nt)
> },VV2[i,1]):
> od:
> for i from 1 to 3 do
> aux:=HH3[i,1]:
> for j from 1 to nt do
> aux:=subs(diff(du[j](u0,v0,Omega),u0)=duu0[j],aux);
> od:
> HH3[i,1]:=subs({seq(du[ii](u0,v0,Omega)=du[ii],
> ii=1..nt)},aux):
> aux:=HH3[i,2]:
> for j from 1 to nt do
> aux:=subs(diff(du[j](u0,v0,Omega),v0)=duv0[j],aux);
> od:
> HH3[i,2]:=subs({seq(du[ii](u0,v0,Omega)=du[ii],
> ii=1..nt)},aux):
> aux:=HH3[i,3]:
> for j from 1 to nt do
> aux:=subs(diff(du[j](u0,v0,Omega),Omega)=duomega[j],aux);
> od:
> HH3[i,3]:=subs({seq(du[ii](u0,v0,Omega)=du[ii],
> ii=1..nt)},aux):
> VV3[i,1]:=subs({seq(du[ii](u0,v0,Omega)=du[ii],ii=1..nt)
> },VV3[i,1]):
> od:
```


Remontagem do problema

```
> unassign('cc'); unassign('dd');
> unassign('du'); unassign('duu0'); unassign('duv0');
> unassign('ccu0'); unassign('ccv0');
> unassign('ddu0'); unassign('ddv0');
> unassign('ccu0u0'); unassign('ccu0v0');
> unassign('ccv0u0'); unassign('ccv0v0');
> unassign('ddu0u0'); unassign('ddu0v0');
> unassign('ddv0u0'); unassign('ddv0v0');
> omega0:='omega0': beta:='beta': delta:= 'delta': F:='F':
> zeta:='zeta': F0:='F0': Omega:='Omega':
> b:='b': u0:='u0': phi:='phi': v0:='v0':
> xx2;
```

$$cc_1(u0,\ v0,\ \Omega)\cos(\Omega\,t) + dd_1(u0,\ v0,\ \Omega)\sin(\Omega\,t) + cc_2(u0,\ v0,\ \Omega)\cos(3\,\Omega\,t) + dd_2(u0,\ v0,\ \Omega)\sin(3\,\Omega\,t)$$

```
> xx2:=subs(
> {seq(cc[jj](u0,v0,Omega)=cc[jj],jj=0..nt),
> seq(dd[jj](u0,v0,Omega)=dd[jj],jj=1..nt)},xx2);
```

$$xx2 := cc_1\cos(\Omega\,t) + dd_1\sin(\Omega\,t) + cc_2\cos(3\,\Omega\,t) + dd_2\sin(3\,\Omega\,t)$$

solução

```
> CC[0]:=subs({seq(du[jj](u0,v0,Omega)=du[jj],
> jj=1..nt)},A[0]):
> for i from 1 to nh do;
> CC[i]:=subs({seq(du[jj](u0,v0,Omega)=du[jj],
> jj=1..nt)},A[i]);
> DD[i]:=subs({seq(du[jj](u0,v0,Omega)=du[jj],
> jj=1..nt)},B[i]);
> od:
> c0[0];
```

$$u0$$

```
> c0[2];
```

$$F - 2\,\zeta\,\omega0\,v0 - \omega0^2\,u0 - \beta\,u0^3$$

```
> c0[5];
```

$$-2\,\zeta\,\omega0\,du_4(u0,\ v0,\ \Omega) - \omega0^2\,du_3(u0,\ v0,\ \Omega) - 6\,\beta\,v0^3 - 18\,\beta\,u0\,v0\,du_2(u0,\ v0,\ \Omega) - 3\,\beta\,u0^2\,du_3(u0,\ v0,\ \Omega)$$

derivadas fundamentais

```
> DU[0]:=u0;
> DU[1]:=v0;
> for i from 2 to nt do;
> DU[i]:=expand(subs({seq(du[jj](u0,v0,Omega)=du[jj],
jj=2..i-1)},c0[i]));
> od:
```

$$DU_0 := u0$$
$$DU_1 := v0$$

```
> DU[2];
> DU[3];
> DU[4];
> DU[5];
```

$$F - 2\,\zeta\,\omega0\,v0 - \omega0^2\,u0 - \beta\,u0^3$$
$$-2\,\zeta\,\omega0\,du_2 - \omega0^2\,v0 - 3\,\beta\,u0^2\,v0$$
$$-F\,\Omega^2 - 2\,\zeta\,\omega0\,du_3 - \omega0^2\,du_2 - 6\,\beta\,u0\,v0^2 - 3\,\beta\,u0^2\,du_2$$
$$-2\,\zeta\,\omega0\,du_4 - \omega0^2\,du_3 - 6\,\beta\,v0^3 - 18\,\beta\,u0\,v0\,du_2 - 3\,\beta\,u0^2\,du_3$$

derivadas das derivadas fundamentais

```
> for i from 0 to 2 do;
> DUu0[i]:=diff(DU[i],u0);
> DUv0[i]:=diff(DU[i],v0);
> DUomega[i]:=diff(DU[i],Omega);
> od:
> for i from 3 to nt do;
> DUu0[i]:=subs({seq(diff(du[jj](u0,v0,Omega),u0)=duu0[jj],
jj=2..i),
```

```
>  seq(du[jj](u0,v0,Omega)=du[jj],jj=2..i)}, diff(c0[i],u0));
>  DUv0[i]:=subs({seq(diff(du[jj](u0,v0,Omega),v0)=duv0[jj],
jj=2..i),
>  seq(du[jj](u0,v0,Omega)=du[jj],jj=2..i)}, diff(c0[i],v0));
>  DUomega[i]:=subs({seq(diff(du[jj](u0,v0,Omega),Omega)
= duomega[jj],jj=2..i),seq(du[jj](u0,v0,Omega)=du[jj],
jj=2..i)}, diff(c0[i],Omega));
>  od:
```

Rotinas

```
>  COMPUTE_TRUE_DU:=proc()
>  global zeta,omega0,beta,F0,F,Omega,u0,v0,nt,du,DU,true_du;
>  local i,j;
>  true_du[0]:=DU[0];
>  true_du[1]:=DU[1];
>  true_du[2]:=DU[2];
>  for i from 3 to nt do
>  true_du[i]:=expand(subs({seq(du[j]=true_du[j],j=2..i-1)},
DU[i]));
>  od;
>  "ok";
>  end proc:
>  COMPUTE_TRUE_DDU:=proc()
>  global zeta,omega0,beta,F0,F,Omega,u0,v0,nt,
>  duu0,duv0,du,DUu0,DUv0,
>  DUomega,DU,true_du,true_duu0,true_duv0,true_duomega;
>  local i,j;
>  true_duu0[0]:=DUu0[0]; true_duv0[0]:=DUv0[0];
>  true_duomega[0]:=DUomega[0];
>  true_duu0[1]:=DUu0[1]; true_duv0[1]:=DUv0[1];
>  true_duomega[1]:=DUomega[1];
>  true_duu0[2]:=DUu0[2]; true_duv0[2]:=DUv0[2];
>  true_duomega[2]:=DUomega[2];
>  for i from 3 to nt do
>  true_duu0[i]:=expand(subs({seq(du[j]=true_du[j],j=2..i-1),
>  seq(duu0[j]=true_duu0[j], j=2..i-1)}, DUu0[i]));
>  true_duv0[i]:=expand(subs({seq(du[j]=true_du[j],j=2..i-1),
>  seq(duv0[j]=true_duv0[j], j=2..i-1)}, DUv0[i]));
>  true_duomega[i]:=expand(subs({seq(du[j]=true_du[j],
j=2..i-1),
>  seq(duomega[j]=true_duomega[j], j=2..i-1)}, DUomega[i]));
>  od;
>  "ok";
>  end proc:
>  COMPUTE_TRUE_VV:=proc()
>  global zeta,omega0,beta,F0,F,Omega,u0,v0,nt,duu0,duv0,du,
>  true_du,true_duu0,true_duv0,true_vv,VV2;
>  local i;
>  true_vv:=Matrix(2,1):
>  for i from 1 to 2 do
>  true_vv[i,1]:=subs({seq(du[j]=true_du[j],j=2..nt)},
VV2[i,1]);
```

```
> od:
> "ok";
> end proc:
> COMPUTE_TRUE_VV3:=proc()
> global zeta,omega0,beta,F0,F,Omega,u0,v0,nt,
> duu0,duv0,du,true_du,
> true_duu0,true_duv0,true_vv3,VV3;
> local i;
> true_vv3:=Matrix(3,1):
> for i from 1 to 2 do
> true_vv3[i,1]:=subs({seq(du[j]=true_du[j],j=2..nt)},
VV3[i,1]);
> od:
> true_vv3[3,1]:=VV3[3,1]:
> "ok";
> end proc:
> COMPUTE_TRUE_HH:=proc()
> global zeta,omega0,beta,F0,F,Omega,u0,v0,
> Omega_0,u0_0,v0_0,r,nt,
> duu0,duv0,du,true_du,true_duu0,true_duv0,true_hh,HH2;
> local i;
> true_hh:=Matrix(2):
> for i from 1 to 2 do
> true_hh[i,1]:=subs({seq(du[j]=true_du[j],j=2..nt),
> seq(duu0[j]=true_duu0[j],j=2..nt)},HH2[i,1]);
> od:
> for i from 1 to 2 do
> true_hh[i,2]:=subs({seq(du[j]=true_du[j],j=2..nt),
> seq(duv0[j]=true_duv0[j],j=2..nt)},HH2[i,2]);
> od:
> "ok";
> end proc:
> COMPUTE_TRUE_HH3:=proc()
> global zeta,omega0,beta,F0,F,Omega,u0,v0,
> Omega_0,u0_0,v0_0,r,nt,
> duu0,duv0,du,true_du,true_duu0,true_duv0,true_hh3,HH3;
> local i;
> true_hh3:=Matrix(3):
> for i from 1 to 2 do
> true_hh3[i,1]:=subs({seq(du[j]=true_du[j],j=2..nt),
> seq(duu0[j]=true_duu0[j],j=2..nt)},HH3[i,1]);
> od:
> for i from 1 to 2 do
> true_hh3[i,2]:=subs({seq(du[j]=true_du[j],j=2..nt),
> seq(duv0[j]=true_duv0[j],j=2..nt)},HH3[i,2]);
> od:
> for i from 1 to 2 do
> true_hh3[i,3]:=subs({seq(du[j]=true_du[j],j=2..nt),
> seq(duomega[j]=true_duomega[j],j=2..nt)},HH3[i,3]);
> od:
> true_hh3[3,1]:=HH3[3,1]: true_hh3[3,2]:=HH3[3,2]:
> true_hh3[3,3]:=HH3[3,3]:
> "ok";
> end proc:
```

```
> COMPUTE_ONE_STEP:=proc()
> global true_hh,true_vv,u0,v0,COMPUTE_TRUE_DU,
> COMPUTE_TRUE_VV,COMPUTE_TRUE_DDU,COMPUTE_TRUE_HH;
> COMPUTE_TRUE_DU();
> COMPUTE_TRUE_VV();
> COMPUTE_TRUE_DDU();
> COMPUTE_TRUE_HH();
> evalm(inverse(true_hh)&*true_vv);
> u0:=u0+%[1,1];
> v0:=v0+%%[2,1];
> evalf(%%%[1,1]),evalf(%%%[2,1]);
> end proc:
> COMPUTE_ONE_AUTOMATIC_STEP:=proc()
> global
> true_hh3,true_vv3,u0,v0,Omega,COMPUTE_TRUE_DU,
> COMPUTE_TRUE_VV3,COMPUTE_TRUE_DDU,COMPUTE_TRUE_HH3;
> local aux;
> COMPUTE_TRUE_DU();
> COMPUTE_TRUE_VV3();
> COMPUTE_TRUE_DDU();
> COMPUTE_TRUE_HH3();
> aux:=evalm(inverse(true_hh3)&*true_vv3);
> u0:=u0+%[1,1];
> v0:=v0+%%[2,1];
> Omega:=Omega+%%%[3,1];
> evalf(aux[1,1]),evalf(aux[2,1]),evalf(aux[3,1]);
> end proc:
> COMPUTE_TRUE_SOLUTION:=proc()
> global zeta,omega0,beta,F0,F,Omega,u0,v0,nt,nh,du,cc,dd,
> true_du,true_cc,true_dd,true_xx;
> local i,j;
> true_du[0]:=DU[0];
> true_du[1]:=DU[1];
> true_du[2]:=DU[2];
> true_cc[0]:=0:
> for i from 1 to nh do
> true_cc[i]:=expand(subs({seq(du[j]=true_du[j],
> j=2..nt)},CC[i]));
> true_dd[i]:=expand(subs({seq(du[j]=true_du[j],
> j=2..nt)},DD[i]));
> od;
> i:='i':
> true_xx:=true_cc[0]+sum(true_cc[i]*cos((2*i-1)*Omega*t)+
> true_dd[i]*sin((2*i-1)*Omega*t),i=1..nh);
> end proc:
```

C.1.2 Exemplo

```
> t:='t':
```

```
> eqd;
```

$$(\tfrac{d^2}{dt^2}\,\mathrm{u}(t)) + 2\,\zeta\,\omega 0\,(\tfrac{d}{dt}\,\mathrm{u}(t)) + \omega 0^2\,\mathrm{u}(t) + \beta\,\mathrm{u}(t)^3 = F\cos(\Omega\,t)$$

```
> zeta:=0.05; omega0:=1; alpha:=0; beta:=1;
> F:=1; Omega:=2;
```

$$\zeta := 0.05$$
$$\omega 0 := 1$$
$$\alpha := 0$$
$$\beta := 1$$
$$F := 1$$
$$\Omega := 2$$

```
> eqd;
```

$$(\tfrac{d^2}{dt^2}\,\mathrm{u}(t)) + 0.10\,(\tfrac{d}{dt}\,\mathrm{u}(t)) + \mathrm{u}(t) + \mathrm{u}(t)^3 = \cos(2\,t)$$

Chute inicial para as coordenadas do ponto fixo da solução periódica

```
> u0:=1.2;
> v0:=0.2;
```

$$u0 := 1.2$$
$$v0 := 0.2$$

```
> for i from 1 to 15 do
> COMPUTE_ONE_STEP();
> od:
> COMPUTE_ONE_STEP();
> u0,v0;
```

$$0.1673185730\,10^{-11},\ 0.1835355378\,10^{-9}$$
$$-1.933790430,\ 0.09771591379$$

```
> u0,v0;
```

$$-1.933790430,\ 0.09771591379$$

```
> XX:=COMPUTE_TRUE_SOLUTION();
```

$$XX := -1.858154359\cos(2\,t) + 0.02044165762\sin(2\,t)$$
$$-\,0.0756360712\cos(6\,t) + 0.009472099763\sin(6\,t)$$

Verificação da solução: integração numérica

```
> t:='t':
> eqd;
```

$$(\tfrac{d^2}{dt^2}\,\mathrm{u}(t)) + 0.10\,(\tfrac{d}{dt}\,\mathrm{u}(t)) + \mathrm{u}(t) + \mathrm{u}(t)^3 = \cos(2\,t)$$

```
> eqd2:=y(t)=diff(x(t),t),diff(y(t),t)=F*cos(Omega*t)
-2*omega0*zeta*dif
> f(x(t),t)
> +alpha*x(t)^2-omega0^2*x(t)-beta*x(t)^3;
> init:=x(0)=u0,y(0)=v0;
```

$$eqd2 := \mathrm{y}(t) = \tfrac{d}{dt}\,\mathrm{x}(t),\ \tfrac{d}{dt}\,\mathrm{y}(t) = \cos(2\,t) - 0.10\,(\tfrac{d}{dt}\,\mathrm{x}(t)) - \mathrm{x}(t) - \mathrm{x}(t)^3$$
$$init := \mathrm{x}(0) = -1.933790430,\ \mathrm{y}(0) = 0.09771591379$$

```
> f:=dsolve({eqd2,init
> },type=numeric,method=classical[rk4],maxfun=999999,
output=procedurelist):
```

```
> T:=evalf(2*Pi/Omega);
```

$$T := 3.141592654$$

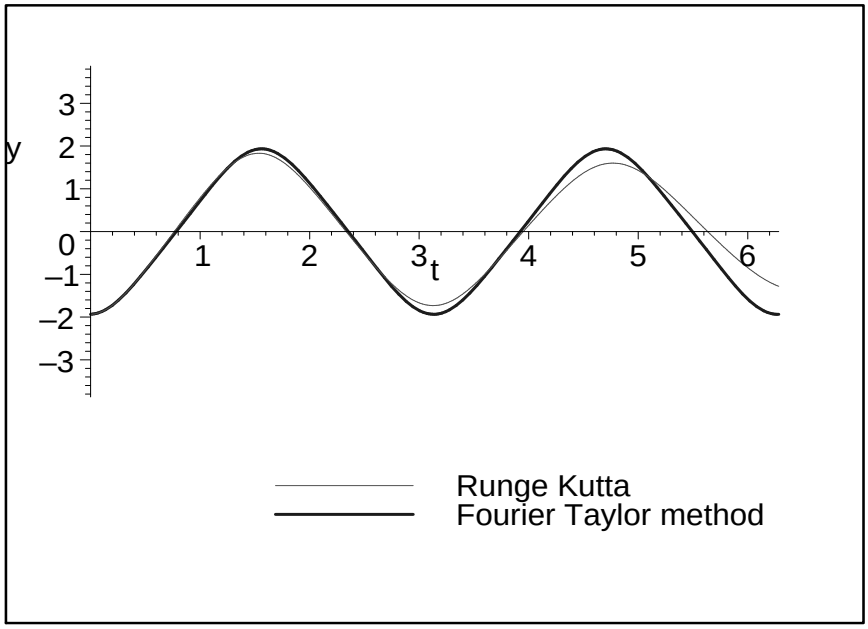
```
> vi:=0; vf:=evalf(2*T);
> #escrevendo na lista
> delta:=evalf(T/200): cont:=0:
> for k from vi by delta to vf do
> cont:=cont+1;
> od: cont;
> dd1:=array(1..cont,1..2): dd2:=array(1..cont,1..2):
> cont:=0:
> for k from vi by delta to vf do
> cont:=cont+1;
> dd1[cont,1]:=k;
> dd1[cont,2]:=eval(x(t),f(k));
> dd2[cont,1]:=k;
> dd2[cont,2]:=eval(y(t),f(k));
> od:
> d1g:=convert(dd1,listlist):
> d2g:=convert(dd2,listlist):
```

$$vi := 0$$

$$vf := 6.283185308$$

$$401$$

```
> plot([d1g,XX],t=0..vf,y=-2*u0..2*u0,color=[red,blue],
thickness=[2,4],legend=["Runge Kutta","Fourier Taylor
method"]);
```



C.2
Exportação de arquivo para programa em C++

```
> with(CodeGeneration):
> omega0:='omega0': beta:='beta': delta:= 'delta': F:='F':
> zeta:='zeta': F0:='F0': Omega:='Omega':
> b:='b': u0:='u0': phi:='phi': v0:='v0': x0:='x0':
> declarations:=[c::numeric, d::numeric, omega0::numeric,
> beta::numeric, Omega::numeric, F::numeric, t::numeric,
> u0::numeric, phi::numeric, u00::numeric, phi0::numeric,
> Omega0::numeric,
> du::numeric, duu0::numeric, duv0::numeric,
> duomega::numeric,
> cc::numeric, dd::numeric]:
> for i from 2 to nt do
> for k from 1 to 1 do
> nomes:=sprintf("duII%dJJ",i-1);
> du_toC[i]:=C(DU[i],resultname=nomes,declare=declarations,
precision=double,output=string,optimize=false):
> od:
> evalf(100*i/nt);
> od:
> for i from 2 to nt do
> for k from 1 to 1 do
> nomes:=sprintf("duu0II%dJJ",i-1);
> duu0_toC[i]:=C(DUu0[i],resultname=nomes,
> declare=declarations,precision=double,output=string,
optimize=false):
> nomes:=sprintf("duv0II%dJJ",i-1);
> duv0_toC[i]:=C(DUv0[i],resultname=nomes,
> declare=declarations, precision=double,output=string,
optimize=false):
> nomes:=sprintf("duomegaII%dJJ",i-1);
> duomega_toC[i]:=C(DUomega[i],resultname=nomes,
> declare=declarations, precision=double,output=string,
optimize=false):
> od:
> evalf(100*i/nt);
> od:
> for i from 1 to 3 do
> for k from 1 to 1 do
> nomes:=sprintf("vII%dJJ",i-1);
> v_toC[i]:=C(VV3[i,1],resultname=nomes,
> declare=declarations,precision=double,output=string,
optimize=true):
> for j from 1 to 3 do
> nomes:=sprintf("hII%dJJII%dJJ",i-1,j-1);
> m_toC[i,j]:=C(HH3[i,j],resultname=nomes,
> declare=declarations,precision=double,output=string,
optimize=true):
> od:
> od:
> evalf(100*i/3);
> od:
```

```
> for i from 1 to nh do
> for k from 1 to 1 do
> nomes:=sprintf("cII%dJJ",i-1);
> ci_toC[i]:=C(CC[i],resultname=nomes,
> declare=declarations,precision=double,output=string,
optimize=false):
> nomes:=sprintf("dII%dJJ",i-1);
> di_toC[i]:=C(DD[i],resultname=nomes,
> declare=declarations,precision=double,output=string,
optimize=false):
> od:
> evalf(100*i/nh);
> od:
> xx2;
```

$$cc_1 \cos(\Omega\, t) + dd_1 \sin(\Omega\, t) + cc_2 \cos(3\,\Omega\, t) + dd_2 \sin(3\,\Omega\, t)$$

```
> subs(t=0,xx2):
> dx0[0]:=%;
> x0:=%:
> for i from 1 to nt do
> subs(t=0,evalf(diff(xx2/(2*nh-1)^i,t$i))):
> dx0[i]:=collect(%,Omega);
> od:
```

$$dx0_0 := cc_1 + cc_2$$

```
> x0;
```

$$cc_1 + cc_2$$

```
> nomes:=sprintf("x0",i-1):
> x0_toC:=C(x0,resultname=nomes,
> declare=declarations,precision=double,output=string,
optimize=false):
> for i from 0 to nt do
> for k from 1 to 1 do
> nomes:=sprintf("dx0II%dJJ",i);
> dx0_toC[i]:=C(dx0[i],resultname=nomes,
> declare=declarations,precision=double,output=string,
optimize=false):
> od:
> evalf(100*i/nt);
> od:
```

C.2.1
Escreve o arquivo

```
> classname:="FTMapleForced_nt";
```

$$classname := \text{“FTMapleForced\_nt”}$$

```
> PRINT_HEADING := proc(fd)
> fprintf(fd,"//equações from Maple\n"):
> fprintf(fd,"\n"):
> fprintf(fd,"#include <math.h>\n"):
> fprintf(fd,"#include \"ftmaple.h\"\n"):
> fprintf(fd,"\n"):
> end proc:
> PRINT_LOCAL_VARS := proc(fd)
> local cont,i,j;
> fprintf(fd,"//local vars\n"):
> cont:=0:
> for i from 1 to 10*nh do
> cont:=cont+1:
> fprintf(fd,"static double t%d",cont):
> for j from 1 to nt do
> cont:=cont+1:
> fprintf(fd,",t%d",cont):
> od:
> fprintf(fd,";\n"):
> od:
> fprintf(fd,"\n\n"):
> end proc:
> PRINT_COMPUTE_DU := proc(fd) global ci_toC,di_toC; local
i;
> fprintf(fd,"void %s%d::ComputeDU()\n",classname,nh):
> fprintf(fd,"{\n"):
> for i from 2 to nt do
> fprintf(fd,"%s",du_toC[i]):
> od:
> fprintf(fd,"}\n"):
> end proc:
> PRINT_COMPUTE_DDU := proc(fd) global ci_toC,di_toC; local
i;
> fprintf(fd,"void %s%d::ComputeDDU()\n",classname,nh):
> fprintf(fd,"{\n"):
> for i from 2 to nt do
> fprintf(fd,"%s",duu0_toC[i]):
> fprintf(fd,"%s",duv0_toC[i]):
> fprintf(fd,"%s",duomega_toC[i]):
> od:
> fprintf(fd,"}\n"):
> end proc:
> PRINT_CONSTRUCTOR := proc(fd)
> fprintf(fd,"%s%d::%s%d()\n",classname,nh,classname,nh):
> fprintf(fd,"{\n"):
> fprintf(fd,"}\n"):
> fprintf(fd,"\n"):
> fprintf(fd,"%s%d::~%s%d()\n",classname,nh,classname,nh):
> fprintf(fd,"{\n"):
> fprintf(fd,"}\n"):
> fprintf(fd,"\n"):
> end proc:
```

```
> PRINT_SOLUTION := proc(fd) global ci_toC,di_toC; local i;
> fprintf(fd,"void %s%d::Solution()\n",classname,nh):
> fprintf(fd,"{\n"):
> for i from 1 to nh do
> fprintf(fd,"%s",ci_toC[i]):
> fprintf(fd,"%s",di_toC[i]):
> od:
> fprintf(fd,"}\n"):
> end proc:
> PRINT_EQ := proc(fd) global m_toC,v_toC; local i,j,k;
> fprintf(fd,"void %s%d::Eq()\n",classname,nh):
> fprintf(fd,"{\n"):
> fprintf(fd,"ComputeDU();\n"):
> fprintf(fd,"ComputeDDU();\n"):
> for i from 1 to 3 do
> fprintf(fd,"%s",v_toC[i]):
> od:
> for i from 1 to 3 do
> for j from 1 to 3 do
> fprintf(fd,"%s",m_toC[i,j]):
> od:
> od:
> fprintf(fd,"}\n"):
> fprintf(fd,"\n"):
> end proc:
> PRINT_COMPUTE_DX0 := proc(fd) global x0_toC,dx0_toC; local
i;
> fprintf(fd,"void %s%d::ComputeDX0()\n",classname,nh):
> fprintf(fd,"{\n"):
> fprintf(fd,"%s",x0_toC):
> for i from 0 to nt do
> fprintf(fd,"%s",dx0_toC[i]):
> od:
> fprintf(fd,"for(int i=0;i<nh;++i)\n"):
> fprintf(fd," dx0[i]*=pow(2*nh-1,i);\n"):
> fprintf(fd,"}\n"):
> end proc:
> fname:=sprintf("c:\\ftmaple_nl3_nt=%d.cpp",nh);
```

$$fname := \text{“c:\ftmaple\_nl3\_nt=2.cpp”}$$

writing...

```
> fname;
> fd:=fopen(fname,WRITE,BINARY);
> PRINT_HEADING(fd);
> PRINT_LOCAL_VARS(fd);
> PRINT_SOLUTION(fd);
> PRINT_COMPUTE_DX0(fd);
> PRINT_COMPUTE_DU(fd);
> PRINT_COMPUTE_DDU(fd);
> PRINT_EQ(fd);
> fclose(fd);
```

$$\text{“c:\ftmaple\_nl3\_nt=2.cpp”}$$

$$fd := 0$$

$$1$$

2
2
2
2
2
1

C.2.2
Arquivo exportado

```
    //equações from Maple
#include <math.h>
#include "ftmaple.h"
//local vars
static double t1,t2,t3,t4,t5,t6,t7,t8;
static double t9,t10,t11,t12,t13,t14,t15,t16;
static double t17,t18,t19,t20,t21,t22,t23,t24;
static double t25,t26,t27,t28,t29,t30,t31,t32;
static double t33,t34,t35,t36,t37,t38,t39,t40;
static double t41,t42,t43,t44,t45,t46,t47,t48;

void FTMapleForced_nt2::Solution()
{
c[0]=0.9e1/0.8e1 * u0 - pow(Omega, - 0.2e1) * ( - F/0.2e1 + zeta * omega0 *
v0 + omega0 * omega0 * u0/0.2e1 + beta * pow(u0,0.3e1)/0.2e1)/0.4e1;
d[0]=0.9e1/0.8e1/Omega * v0 - 0.3e1/0.4e1 * pow(Omega, - 0.3e1) * (zeta
* omega0 * du[1]/0.3e1 + omega0 * omega0 * v0/0.6e1 + beta * u0 * u0 *
v0/0.2e1);
c[1]= - u0/0.8e1 + pow(Omega, - 0.2e1) * ( - F/0.2e1 + zeta * omega0 * v0
+ omega0 * omega0 * u0/0.2e1 + beta * pow(u0,0.3e1)/0.2e1)/0.4e1;
d[1]= - 0.1e1/Omega * v0/0.24e2 + pow(Omega, - 0.3e1) * (zeta * omega0 *
du[1]/0.3e1 + omega0 * omega0 * v0/0.6e1 + beta * u0 * u0 * v0/0.2e1)/0.4e1;
}

void FTMapleForced_nt2::ComputeDX0()
{
x0=cc[0] + cc[1];
dx0[0]=cc[0] + cc[1];
dx0[1]=(0.3333333333e0 * dd[0] + 0.1e1 * dd[1]) * Omega;
dx0[2]=( - 0.1111111111e0 * cc[0] - 0.1e1 * cc[1]) * Omega * Omega;
dx0[3]=( - 0.3703703704e - 1 * dd[0] - 0.1e1 * dd[1]) * pow(Omega,0.3e1);
dx0[4]=(0.1234567901e - 1 * cc[0] + 0.1e1 * cc[1]) * pow(Omega,0.4e1);
dx0[5]=(0.4115226337e - 2 * dd[0] + 0.1e1 * dd[1]) * pow(Omega,0.5e1);
dx0[6]=( - 0.1371742112e - 2 * cc[0] - 0.1e1 * cc[1]) * pow(Omega,0.6e1);
dx0[7]=( - 0.4572473708e - 3 * dd[0] - 0.1e1 * dd[1]) * pow(Omega,0.7e1);
for(int i=0; i<nh; ++i)
```

```
dx0[i] * =pow(2 * nh - 1,i);
}

void FTMapleForced_nt2::ComputeDU()
{
du[1]=F - 0.2e1 * zeta * omega0 * v0 - omega0 * omega0 * u0 - beta *
pow(u0,0.3e1);
du[2]= - 0.2e1 * zeta * omega0 * du[1] - omega0 * omega0 * v0 - 0.3e1 * beta
* u0 * u0 * v0;
du[3]= - F * Omega * Omega - 0.2e1 * zeta * omega0 * du[2] - omega0 *
omega0 * du[1] - 0.6e1 * beta * u0 * v0 * v0 - 0.3e1 * beta * u0 * u0 * du[1];
du[4]= - 0.2e1 * zeta * omega0 * du[3] - omega0 * omega0 * du[2] - 0.6e1 *
beta * pow(v0,0.3e1) - 0.18e2 * beta * u0 * v0 * du[1] - 0.3e1 * beta * u0 *
u0 * du[2];
du[5]=F * pow(Omega,0.4e1) - 0.2e1 * zeta * omega0 * du[4] - omega0 *
omega0 * du[3] - 0.36e2 * beta * v0 * v0 * du[1] - 0.18e2 * beta * u0 *
pow(du[1],0.2e1) - 0.24e2 * beta * u0 * v0 * du[2] - 0.3e1 * beta * u0 * u0 *
du[3];
du[6]= - 0.2e1 * zeta * omega0 * du[5] - omega0 * omega0 * du[4] - 0.90e2 *
beta * v0 * pow(du[1],0.2e1) - 0.60e2 * beta * v0 * v0 * du[2] - 0.60e2 * beta
* u0 * du[1] * du[2] - 0.30e2 * beta * u0 * v0 * du[3] - 0.3e1 * beta * u0 * u0
* du[4];
}

void FTMapleForced_nt2::ComputeDDU()
{
duu0[1]= - omega0 * omega0 - 0.3e1 * beta * u0 * u0;
duv0[1]= - 0.2e1 * zeta * omega0;
duomega[1]=0;
duu0[2]= - 0.2e1 * zeta * omega0 * duu0[1] - 0.6e1 * beta * u0 * v0;
duv0[2]= - 0.2e1 * zeta * omega0 * duv0[1] - omega0 * omega0 - 0.3e1 * beta
* u0 * u0;
duomega[2]= - 0.2e1 * zeta * omega0 * duomega[1];
duu0[3]= - 0.2e1 * zeta * omega0 * duu0[2] - omega0 * omega0 * duu0[1] -
0.6e1 * beta * v0 * v0 - 0.6e1 * beta * u0 * du[1] - 0.3e1 * beta * u0 * u0 *
duu0[1];
duv0[3]= - 0.2e1 * zeta * omega0 * duv0[2] - omega0 * omega0 * duv0[1] -
0.12e2 * beta * u0 * v0 - 0.3e1 * beta * u0 * u0 * duv0[1];
duomega[3]= - 0.2e1 * F * Omega - 0.2e1 * zeta * omega0 * duomega[2] -
omega0 * omega0 * duomega[1] - 0.3e1 * beta * u0 * u0 * duomega[1];
duu0[4]= - 0.2e1 * zeta * omega0 * duu0[3] - omega0 * omega0 * duu0[2] -
0.18e2 * beta * v0 * du[1] - 0.18e2 * beta * u0 * v0 * duu0[1] - 0.6e1 * beta *
u0 * du[2] - 0.3e1 * beta * u0 * u0 * duu0[2];
duv0[4]= - 0.2e1 * zeta * omega0 * duv0[3] - omega0 * omega0 * duv0[2] -
0.18e2 * beta * v0 * v0 - 0.18e2 * beta * u0 * du[1] - 0.18e2 * beta * u0 * v0
* duv0[1] - 0.3e1 * beta * u0 * u0 * duv0[2];
duomega[4]= - 0.2e1 * zeta * omega0 * duomega[3] - omega0 * omega0 *
duomega[2] - 0.18e2 * beta * u0 * v0 * duomega[1] - 0.3e1 * beta * u0 * u0 *
```

```
duomega[2];
duu0[5]= - 0.2e1 * zeta * omega0 * duu0[4] - omega0 * omega0 * duu0[3] -
0.36e2 * beta * v0 * v0 * duu0[1] - 0.18e2 * beta * pow(du[1],0.2e1) - 0.36e2
* beta * u0 * du[1] * duu0[1] - 0.24e2 * beta * v0 * du[2] - 0.24e2 * beta * u0
* v0 * duu0[2] - 0.6e1 * beta * u0 * du[3] - 0.3e1 * beta * u0 * u0 * duu0[3];
duv0[5]= - 0.2e1 * zeta * omega0 * duv0[4] - omega0 * omega0 * duv0[3] -
0.72e2 * beta * v0 * du[1] - 0.36e2 * beta * v0 * v0 * duv0[1] - 0.36e2 * beta
* u0 * du[1] * duv0[1] - 0.24e2 * beta * u0 * du[2] - 0.24e2 * beta * u0 * v0 *
duv0[2] - 0.3e1 * beta * u0 * u0 * duv0[3];
duomega[5]=0.4e1 * F * pow(Omega,0.3e1) - 0.2e1 * zeta * omega0 *
duomega[4] - omega0 * omega0 * duomega[3] - 0.36e2 * beta * v0 * v0 *
duomega[1] - 0.36e2 * beta * u0 * du[1] * duomega[1] - 0.24e2 * beta * u0 *
v0 * duomega[2] - 0.3e1 * beta * u0 * u0 * duomega[3];
duu0[6]= - 0.2e1 * zeta * omega0 * duu0[5] - omega0 * omega0 * duu0[4] -
0.180e3 * beta * v0 * du[1] * duu0[1] - 0.60e2 * beta * v0 * v0 * duu0[2] -
0.60e2 * beta * du[1] * du[2] - 0.60e2 * beta * u0 * duu0[1] * du[2] - 0.60e2 *
beta * u0 * du[1] * duu0[2] - 0.30e2 * beta * v0 * du[3] - 0.30e2 * beta * u0 *
v0 * duu0[3] - 0.6e1 * beta * u0 * du[4] - 0.3e1 * beta * u0 * u0 * duu0[4];
duv0[6]= - 0.2e1 * zeta * omega0 * duv0[5] - omega0 * omega0 * duv0[4] -
0.90e2 * beta * pow(du[1],0.2e1) - 0.180e3 * beta * v0 * du[1] * duv0[1] -
0.120e3 * beta * v0 * du[2] - 0.60e2 * beta * v0 * v0 * duv0[2] - 0.60e2 * beta
* u0 * duv0[1] * du[2] - 0.60e2 * beta * u0 * du[1] * duv0[2] - 0.30e2 * beta
* u0 * du[3] - 0.30e2 * beta * u0 * v0 * duv0[3] - 0.3e1 * beta * u0 * u0 *
duv0[4];
duomega[6]= - 0.2e1 * zeta * omega0 * duomega[5] - omega0 * omega0 *
duomega[4] - 0.180e3 * beta * v0 * du[1] * duomega[1] - 0.60e2 * beta * v0 *
v0 * duomega[2] - 0.60e2 * beta * u0 * duomega[1] * du[2] - 0.60e2 * beta *
u0 * du[1] * duomega[2] - 0.30e2 * beta * u0 * v0 * duomega[3] - 0.3e1 * beta
* u0 * u0 * duomega[4];
}

void FTMapleForced_nt2::Eq()
{
ComputeDU();
ComputeDDU();
t1=Omega * Omega;
t4=zeta * omega0;
t8=omega0 * omega0;
t9=du[1];
t13=v0 * v0;
t16=u0 * u0;
t31=0.1e1/t1 * ( - F/0.2e1 + t4 * v0 + t8 * u0/0.2e1 + beta * t16 *
u0/0.2e1)/0.4e1;
t33=t1 * t1;
v[0]=F * t1/0.24e2 + t4 * du[2]/0.12e2 + t8 * t9/0.24e2 + beta * u0 *
t13/0.4e1 + beta * t16 * t9/0.8e1 + (0.9e1/0.8e1 * u0 - t31) * t33/0.24e2 +
0.27e2/0.8e1 * ( - u0/0.8e1 + t31) * t33;
t1=zeta * omega0;
```

```
t5=omega0 * omega0;
t6=du[2];
t9=v0 * v0;
t14=du[1];
t18=u0 * u0;
t19=beta * t18;
t23=0.1e1/Omega * v0;
t25=Omega * Omega;
t35=0.1e1/t25/Omega * (t1 * t14/0.3e1 + t5 * v0/0.6e1 + t19 * v0/0.2e1);
t38=t25 * t25;
t39=t38 * Omega;
v[1]=t1 * du[3]/0.60e2 + t5 * t6/0.120e3 + beta * t9 * v0/0.20e2 +
0.3e1/0.20e2 * beta * u0 * v0 * t14 + t19 * t6/0.40e2 + (0.9e1/0.8e1 *
t23 - 0.3e1/0.4e1 * t35) * t39/0.120e3 + 0.81e2/0.40e2 * ( - t23/0.24e2 +
t35/0.4e1) * t39;
t2=pow(u0 - u0_0,0.2e1);
t4=pow(v0 - v0_0,0.2e1);
t6=pow(Omega - Omega_0,0.2e1);
t7=r * r;
v[2]= - t2 - t4 - t6 + t7;
t5=omega0 * omega0;
t6=duu0[1];
t9=v0 * v0;
t16=u0 * u0;
t17=beta * t16;
t20=Omega * Omega;
t26=0.1e1/t20 * (t5/0.2e1 + 0.3e1/0.2e1 * t17)/0.4e1;
t28=t20 * t20;
h[0][0]= - zeta * omega0 * duu0[2]/0.12e2 - t5 * t6/0.24e2 - beta * t9/0.4e1 -
beta * u0 * du[1]/0.4e1 - t17 * t6/0.8e1 - (0.9e1/0.8e1 - t26) * t28/0.24e2 -
0.27e2/0.8e1 * ( - 0.1e1/0.8e1 + t26) * t28;
t5=omega0 * omega0;
t6=duv0[1];
t12=u0 * u0;
t16=Omega * Omega;
h[0][1]= - zeta * omega0 * duv0[2]/0.12e2 - t5 * t6/0.24e2 - beta * u0 *
v0/0.2e1 - beta * t12 * t6/0.8e1 - 0.5e1/0.6e1 * t16 * zeta * omega0;
t3=zeta * omega0;
t7=omega0 * omega0;
t8=duomega[1];
t11=u0 * u0;
t22= - F/0.2e1 + t3 * v0 + t7 * u0/0.2e1 + beta * t11 * u0/0.2e1;
t26=Omega * Omega;
t29=0.1e1/t26 * t22/0.4e1;
t31=t26 * Omega;
h[0][2]= - F * Omega/0.12e2 - t3 * duomega[2]/0.12e2 - t7 * t8/0.24e2 - beta
* t11 * t8/0.8e1 + 0.5e1/0.3e1 * Omega * t22 - (0.9e1/0.8e1 * u0 - t29) *
t31/0.6e1 - 0.27e2/0.2e1 * ( - u0/0.8e1 + t29) * t31;
```

```
t1=zeta * omega0;
t5=omega0 * omega0;
t6=duu0[2];
t13=beta * u0;
t14=duu0[1];
t21=u0 * u0;
t25=Omega * Omega;
h[1][0]= - t1 * duu0[3]/0.60e2 - t5 * t6/0.120e3 - 0.3e1/0.20e2 * beta * v0
* du[1] - 0.3e1/0.20e2 * t13 * v0 * t14 - t13 * du[2]/0.20e2 - beta * t21 *
t6/0.40e2 - t25 * (t1 * t14/0.3e1 + t13 * v0)/0.2e1;
t1=zeta * omega0;
t5=omega0 * omega0;
t6=duv0[2];
t9=v0 * v0;
t12=beta * u0;
t16=duv0[1];
t20=u0 * u0;
t21=beta * t20;
t24=0.1e1/Omega;
t26=Omega * Omega;
t34=0.1e1/t26/Omega * (t1 * t16/0.3e1 + t5/0.6e1 + t21/0.2e1);
t37=t26 * t26;
t38=t37 * Omega;
h[1][1]= - t1 * duv0[3]/0.60e2 - t5 * t6/0.120e3 - 0.3e1/0.20e2 * beta * t9 -
0.3e1/0.20e2 * t12 * du[1] - 0.3e1/0.20e2 * t12 * v0 * t16 - t21 * t6/0.40e2
- (0.9e1/0.8e1 * t24 - 0.3e1/0.4e1 * t34) * t38/0.120e3 - 0.81e2/0.40e2 * ( -
t24/0.24e2 + t34/0.4e1) * t38;
t1=zeta * omega0;
t5=omega0 * omega0;
t6=duomega[2];
t10=duomega[1];
t14=u0 * u0;
t15=beta * t14;
t18=Omega * Omega;
t20=0.1e1/t18 * v0;
t22=t18 * t18;
t31=t1 * du[1]/0.3e1 + t5 * v0/0.6e1 + t15 * v0/0.2e1;
t32=0.1e1/t22 * t31;
t35=0.1e1/t18/Omega;
t38=t35 * zeta * omega0 * t10;
t41=Omega * t22;
t45=0.1e1/Omega * v0;
t47=t35 * t31;
h[1][2]= - t1 * duomega[3]/0.60e2 - t5 * t6/0.120e3 - 0.3e1/0.20e2 * beta
* u0 * v0 * t10 - t15 * t6/0.40e2 - ( - 0.9e1/0.8e1 * t20 + 0.9e1/0.4e1 *
t32 - t38/0.4e1) * t41/0.120e3 - (0.9e1/0.8e1 * t45 - 0.3e1/0.4e1 * t47) *
t22/0.24e2 - 0.81e2/0.40e2 * (t20/0.24e2 - 0.3e1/0.4e1 * t32 + t38/0.12e2) *
t41 - 0.81e2/0.8e1 * ( - t45/0.24e2 + t47/0.4e1) * t22;
```

```
h[2][0]=0.2e1 * u0 - (double)(2 * u0_0);
h[2][1]=2 * v0 - 2 * v0_0;
h[2][2]=0.2e1 * Omega - (double)(2 * Omega_0);
}
```