8
Referências Bibliográficas

ANDRADE, E.Q., 1993, **Instabilidade e Vibrações de Colunas Esbeltas sobre Base Elástica**. 149 f. Dissertação de Mestrado – PUC-Rio, Rio de Janeiro.

BOUC, R., 1972, **Sur la Methode de Galerkin-Urabe pour les Systemes Differentiels Periodiques.** International Journal of Non-Linear Mechanics, Vol. 7, pp. 175-188.

BOYCE, W. E.; DIPRIMA, R. C., 1998, **Equações Diferenciais Elementares e Problemas de Valores de Contorno**. 6. ed. Tradução Engenheiro Horacio Macedo. Rio de Janeiro: Livros Técnicos e Científicos. 532p. Título original: Elementary Differential Equations and Boundary Value Problems.

BIRKHOFF, G. D., 1927, **Dynamical Systems**. Providence: A.M.S. Publications.

CLARK, S. K.,1972**, Dynamics of Continuous Elements**. New Jersey: Prentice-Hall. 219p.

CLOUGH, R.W.; PENZIEN, J., 1993, **Dynamics of Structures**. 2. ed. New York: McGraw-Hill.738p.

COSKUN, I.; ENGIN, H., 1999, **Non-Linear Vibrations of a Beam on an Elastic Foundation**. Journal of Sound and Vibration, Vol. 223, pp. 335-354.

DANIEL, J., 1986, **Dynamic Buckling of Guyed Stacks, Masts and Columns with Constant Inertia and Algebric Polynomials Generated by use of Operator T_n Attached to Bessel Functions**. Great Britain: Journal of the Franklin Institute, Vol. 321, pp. 95-107.

DAVISSON, M.T.; ROBINSON, K. E., 1965, **Bending and Buckling of Partially Embedded Piles**, 6th ICSMFE, Vol. 2, pp.243-246.

FARSHAD, M., 1994, **Stability of Structures**. Amsterdam: Elsevier.425p.

LEIPHOLZ, H., 1970**, Stability Theory**. New York, Academic Press. 277 p.

LYAPUNOV, A. M., 1949, **Problém Géneral de la Stabilité du Mouvement. Annals of Mathematical Studies**. Vol.17. Princeton: Princeton University Press.

LYNCH, S., 2001, **Dynamical Systems with Applications using MAPLE**. Boston: Birkhäuser.

MEIROVITCH, L., 1975, **Elements of Vibration Analysis.** New York: Mc-Graw-Hill.

PASQUETTI, E., 2003, **Estabilidade Estática e Dinâmica de Torres Estaiadas**. 99 f. Dissertação de Mestrado - PUC-Rio, Rio de Janeiro.

PERKO, L., 2001, **Differential Equations and Dynamical Systems**. 3.ed. New York: Springer. 525p.

POINCARÉ, H., 1890, **Sur Les Équation de la Dynamique et le Problème des Trois Corps**. [S.I.]. Acta Math 13 pp 1-270.

POINCARE, H., 1880-1890, **Mémoire Sur les Cóurbes Définies par les Équations Différentielles I-IV, Ouevre I**. Paris: Gauthier-Villar.

POINCARE, H., 1899, **Les Méthodes Nouvelles de la Méchanique Celeste, 3 vos**. Paris: Gauthier-Villar.

POULOS, H.G.; DAVIS, E.H., 1980, **Pile Foundation Analysis and Design**. New York: John Wiley and Sons.

PRADO, Z. J. G. M. D., 2001, **Acoplamento e Interação Modal na Instabilidade Dinâmica de Cascas Cilíndricas**. 222 f. Tese de Doutorado – PUC-Rio, Rio de Janeiro.

SANTEE, M. D., 1999, **Vibrações Não-Lineares e Instabilidade de Elementos Estruturais Sensíveis a Imperfeições**. 244 f. Tese de Doutorado – PUC-Rio, Rio de Janeiro.

TERZAGHI, K., 1955, **Evaluation of Coefficients of Subgrade Reaction**. Geotechnique, Vol. 5, pp. 297-326.

THOMSON, W. T., 1978, **Teoria da Vibração com Aplicações**. Tradução Engenheiro Cássio Sigaud. Rio de Janeiro: Interciência. 462 p. Título original: Theory of Vibration with Applications.

TIMOSHENKO, S. P.; GERE, J.M., 1961, **Theory of Elastic Stability**. New York: Mc-Graw-Hill.

URABE, T.; REITER, A., 1966, **Numerical Computation of Nonlinear Forced Oscillations by Galerkin's Procedure**. Journal of Mathematical Analysis and Applications, Vol. 14, pp. 107-140.

VANDEGHEN, A.; ALEXANDRE, M., 1969, **Vibration des Grandes Cheminées en Acier sous L´Action du Vent**. I.A.B.S.E Publicações, Vol. 29-1, pp.95-132.

VIDVASAGAR, M., 1978, **Nonlinear Systems Analysis**. New Jersey: Prentice-Hall. 302p.

WADEE, M. K.; HIGUCHI, Y.; HUNT, G. W., 1998, **Galerkin Approximations to Static and Dynamic Localization Problems**. Int. J. Solids Struct. [S.I.].

YAMAN, M., 2003, **The Analysis of the Orientation Effect of Non-Linear Flexible Systems on Performance of the Pendulum Absorver**. International Journal of Non-Linear Mechanics, Vol. 39, pp. 741-752.

9
Apêndice

O presente apêndice tem como objetivo principal descrever os comandos do programa MAPLE 7 utilizados na análise paramétrica das diversas situações envolvendo a rigidez, a profundidade da fundação, as condições de apoio e de carregamento. O desenvolvimento do programa é dividido em duas partes: análise linear e análise não-linear. Como visto ao longo da tese, cada uma requer abordagem diferenciada e uma especial atenção com a precisão dos valores encontrados, pois as expressões são muito sensíveis às variações dos parâmetros.

9.1.
Análise Linear

O estudo paramétrico realizado na análise linear consiste em quatro etapas:

- Cálculo da carga crítica da estaca parcialmente enterrada;
- Uso do Método de Ritz para cálculo das freqüências de vibração da coluna semi-enterrada carregada;
- Cálculo da primeira freqüência de vibração da estaca semi-enterrada com carregamento estático no topo;
- Representação dos modos de vibração e criação do arquivo para exportação de dados.

Os cálculos apresentados referem-se à coluna biapoiada com a profundidade da fundação atingindo a metade da altura, rigidez igual a 2.000 e carregamento estático igual a 50% da carga crítica.

###Cálculo da carga crítica da estaca bi-apoiada, com base elástica até metade da profundidade da estaca.###

```
> restart:
> with(DEtools):
> with(LinearAlgebra):
> Digits:=64:
```

#Definições:
#w1(x) = expressão para os deslocamentos no trecho desenterrado;
#w2(x) = expressão para os deslocamentos no trecho enterrado;
#lambda = parâmetro adimensional de carga;
#apenas para facilitar a visualização das expressões foi usada no programa a variável x ao invés da letra grega ζ, que representa a coordenada adimensional do eixo axial.

#Rigidez do Solo:

```
> K:=2000:
```

#Definição das Equações Diferenciais:

```
>
eq1:=diff(w1(x),x,x,x,x)+(lambda^2*Pi^2*diff(w1(x),x,x)
)=0:
> sol1:=dsolve(eq1):
> w[1]:=rhs(sol1):
> w[1,x]:=diff(w[1],x):
> w[1,xx]:=diff(w[1],x,x):
> w[1,xxx]:=diff(w[1],x,x,x):
>
eq2:=diff(w2(x),x,x,x,x)+(lambda^2*Pi^2*diff(w2(x),x,x)
)+(Pi^4*K*w2(x))=0:
> sol2[inicial]:=dsolve(eq2):
>
sol2a:=subs(_C1=_C5,_C2=_C6,_C3=_C7,_C4=_C8,sol2[inicia
l]):
> sol2:=rhs(sol2a):
> w[2]:=sol2:
> w[2,x]:=diff(sol2,x):
> w[2,xx]:=diff(sol2,x,x):
> w[2,xxx]:=diff(sol2,x,x,x):
```

#Condições de contorno definidas pelos apoios:

```
> exp1:=eval(w[1],x=1/2):
> exp2:=eval(w[1,xx],x=1/2):
> exp3:=eval(w[2],x=0):
> exp4:=eval(w[2,xx],x=0):
```

#Condições de continuidade:

```
> w[1,x=0]:=eval(w[1],x=0):
> w[2,x=a]:=eval(w[2],x=1/2):
> w[1,x_x=0]:=eval(w[1,x],x=0):
> w[2,x_x=a]:=eval(w[2,x],x=1/2):
> w[1,xx_x=0]:=eval(w[1,xx],x=0):
> w[2,xx_x=a]:=eval(w[2,xx],x=1/2):
> w[1,xxx_x=0]:=eval(w[1,xxx],x=0):
> w[2,xxx_x=a]:=eval(w[2,xxx],x=1/2):
> exp5:=w[1,x=0]-w[2,x=a]:
> exp6:=w[1,x_x=0]-w[2,x_x=a]:
> exp7:=w[1,xx_x=0]-w[2,xx_x=a]:
> exp8:=w[1,xxx_x=0]-w[2,xxx_x=a]:
>
#solve({exp1,exp2,exp3,exp4,exp5,exp6,exp7,exp8},{_C1,_
C2,_C3,_C4,_C5,_C6,_C7,_C8}):
```

#Construção da matriz dos coeficientes do sistema homogêneo de equações:

```
> a11:=coeff(exp1,_C1):
> a21:=coeff(exp2,_C1):
> a31:=coeff(exp3,_C1):
> a41:=coeff(exp4,_C1):
> a51:=coeff(exp5,_C1):
> a61:=coeff(exp6,_C1):
> a71:=coeff(exp7,_C1):
> a81:=coeff(exp8,_C1):
> a12:=coeff(exp1,_C2):
> a22:=coeff(exp2,_C2):
> a32:=coeff(exp3,_C2):
> a42:=coeff(exp4,_C2):
> a52:=coeff(exp5,_C2):
> a62:=coeff(exp6,_C2):
> a72:=coeff(exp7,_C2):
> a82:=coeff(exp8,_C2):
> a13:=coeff(exp1,_C3):
> a23:=coeff(exp2,_C3):
> a33:=coeff(exp3,_C3):
> a43:=coeff(exp4,_C3):
> a53:=coeff(exp5,_C3):
> a63:=coeff(exp6,_C3):
> a73:=coeff(exp7,_C3):
> a83:=coeff(exp8,_C3):
> a14:=coeff(exp1,_C4):
> a24:=coeff(exp2,_C4):
> a34:=coeff(exp3,_C4):
> a44:=coeff(exp4,_C4):
> a54:=coeff(exp5,_C4):
> a64:=coeff(exp6,_C4):
```

```
> a74:=coeff(exp7,_C4):
> a84:=coeff(exp8,_C4):
> a15:=coeff(exp1,_C5):
> a25:=coeff(exp2,_C5):
> a35:=coeff(exp3,_C5):
> a45:=coeff(exp4,_C5):
> a55:=coeff(exp5,_C5):
> a65:=coeff(exp6,_C5):
> a75:=coeff(exp7,_C5):
> a85:=coeff(exp8,_C5):
> a16:=coeff(exp1,_C6):
> a26:=coeff(exp2,_C6):
> a36:=coeff(exp3,_C6):
> a46:=coeff(exp4,_C6):
> a56:=coeff(exp5,_C6):
> a66:=coeff(exp6,_C6):
> a76:=coeff(exp7,_C6):
> a86:=coeff(exp8,_C6):
> a17:=coeff(exp1,_C7):
> a27:=coeff(exp2,_C7):
> a37:=coeff(exp3,_C7):
> a47:=coeff(exp4,_C7):
> a57:=coeff(exp5,_C7):
> a67:=coeff(exp6,_C7):
> a77:=coeff(exp7,_C7):
> a87:=coeff(exp8,_C7):
> a18:=coeff(exp1,_C8):
> a28:=coeff(exp2,_C8):
> a38:=coeff(exp3,_C8):
> a48:=coeff(exp4,_C8):
> a58:=coeff(exp5,_C8):
> a68:=coeff(exp6,_C8):
> a78:=coeff(exp7,_C8):
> a88:=coeff(exp8,_C8):
>
A:=Matrix([[a11,a12,a13,a14,a15,a16,a17,a18],[a21,a22,a
23,a24,a25,a26,a27,a28],[a31,a32,a33,a34,a35,a36,a37,a3
8],[a41,a42,a43,a44,a45,a46,a47,a48],[a51,a52,a53,a54,a
55,a56,a57,a58],[a61,a62,a63,a64,a65,a66,a67,a68],[a71,
a72,a73,a74,a75,a76,a77,a78],[a81,a82,a83,a84,a85,a86,a
87,a88]]):
> det:=simplify(Determinant(A)):
```

#Determinação do valor do parâmetro de carga a partir da verificação do momento em que ocorre a troca de sinal do determinante:

```
> for lambda from 2.51681 by 0.000001 to 2.51682 do
determ[lambda]:=print(simplify(Re(det))) end do;
```

###Método de Ritz para cálculo das freqüências de vibração de uma coluna semi-enterrada carregada###

```
> restart:
> with(DEtools):
> with(LinearAlgebra):
> Digits:=64:
```

#Definições:
#f(x) = expressão para os deslocamentos da estaca;
#Omega = parâmetro de freqüência (s).

#Equação da energia:

```
>
 eq:=int(diff(f(x),`$`(x,2))^2-
Pi^2*lambda^2*(diff(f(x), x))^2-Pi^4*Omega^4*(f(x)^2),
x=0..1)+int(Pi^4*K*f(x)^2,x = 0 .. 1/2):
>
eq:=subs(f(x)=A1*sin(Pi*x)+A2*sin(2*Pi*x)+A3*sin(3*Pi*x
)+A4*sin(4*Pi*x),eq):
> deq1:=diff(eq,A1):
> deq2:=diff(eq,A2):
> deq3:=diff(eq,A3):
> deq4:=diff(eq,A4):
> solve({deq1,deq2,deq3,deq4},{A1,A2,A3,A4}):
> a11:=coeff(deq1,A1):
> a12:=coeff(deq1,A2):
> a13:=coeff(deq1,A3):
> a14:=coeff(deq1,A4):
> a21:=coeff(deq2,A1):
> a22:=coeff(deq2,A2):
> a23:=coeff(deq2,A3):
> a24:=coeff(deq2,A4):
> a31:=coeff(deq3,A1):
> a32:=coeff(deq3,A2):
> a33:=coeff(deq3,A3):
> a34:=coeff(deq3,A4):
> a41:=coeff(deq4,A1):
> a42:=coeff(deq4,A2):
> a43:=coeff(deq4,A3):
> a44:=coeff(deq4,A4):
>
A:=Matrix([[a11,a12,a13,a14],[a21,a22,a23,a24],[a31,a32
,a33,a34],[a41,a42,a43,a44]]):
> det:=Determinant(A):
> K:=2000:lambda:=.5*2.516811209107670508859218:
> plot(det,Omega=0..8):
> sols:=fsolve(det,Omega=0..8):
```

#Determinação dos modos de vibração:
#Substituindo-se o valor da freqüência na matriz A, obtendo-se:

```
> Omega:=sols[1]:
> b11:=simplify(a11):
> b12:=simplify(a12):
> b13:=simplify(a13):
> b14:=simplify(a14):
> b21:=simplify(a21):
> b22:=simplify(a22):
> b23:=simplify(a23):
> b24:=simplify(a24):
> b31:=simplify(a31):
> b32:=simplify(a32):
> b33:=simplify(a33):
> b34:=simplify(a34):
> b41:=simplify(a41):
> b42:=simplify(a42):
> b43:=simplify(a43):
> b44:=simplify(a44):
>
B:=Matrix([[b11,b12,b13,b14],[b21,b22,b23,b24],[b31,b32
,b33,b34],[b41,b42,b43,b44]]):
```

#Eliminação da última linha e da 3a. coluna:

```
>
W:=Matrix([[b11,b12,b14],[b21,b22,b24],[b31,b32,b34]]):
> V:=Vector([A1,A2,A4]):
> Q:=W.V:
> R:=Vector([-b13,-b23,-b33]):
> sols:=solve({Q[1]=R[1], Q[2]=R[2],
Q[3]=R[3]},{A1,A2,A4}):
> sols[1]:
> sols[2]:
> sols[3]:
> V:=subs(sols[1],sols[2],sols[3],V):
> ampnorm:=sqrt(((V[1])^2)+((V[2])^2)+((V[3])^2)+1):
> Vnorm:=V/ampnorm:
```

#Observação: Todas as constantes acima estão sendo multiplicadas por A3, e que consideraremos igual a 1.

```
> A3:=1/ampnorm:
> A:=Vector([Vnorm[1],Vnorm[2],A3,Vnorm[3]]):
> ee1:=A1=A[1]:
> ee2:=A2=A[2]:
> ee3:=A3=A[3]:
> ee4:=A4=A[4]:
```

```
>
f(x):=A1*sin(Pi*x)+A2*sin(2*Pi*x)+A3*sin(3*Pi*x)+A4*sin
(4*Pi*x):
> f(x):=subs(ee1,ee2,ee3,ee4,f(x)):
> with(plottools):plot([f(x),x,x=0..1]):
```

###Cálculo da primeira freqüência de vibração da estaca semi-enterrada biapoiada com carregamento estático no topo.###

```
> restart:
> with(DEtools):
> with(LinearAlgebra):
> Digits:=64:
```

#Definições:
#s = parâmetro de freqüência de vibração da estaca;
#lambda = valor originário de um percentual da carga critico aplicado no topo da estaca (50%);
#f1(x) = expressão para os deslocamentos do trecho desenterrado da estaca;
#f2(x) = expressão para os deslocamentos do trecho enterrado da estaca;

#Resolução para os casos em que: s<= (K-(λ^4/4))^1/4:
#Definição do intervalo de variação de s, baseando-se nos valores aproximados obtidos por Ritz:

```
> inf:=2.045482:
> sup:=2.045483:
> inter:=0.0000001:
```

#Determinação do valor do parâmetro de freqüência a partir da verificação do momento em que ocorre a troca de sinal do determinante:

```
> for s from inf by inter to sup do
K:=2000:lambda_crit:=2.516811209107670508859218:lambda:
=0.5*lambda_crit:
eq1:=diff(f1(x),x,x,x,x)+(lambda^2*Pi^2*diff(f1(x),x,x)
)-(s^4*Pi^4*f1(x))=0:
sol1:=dsolve(eq1): finicial[1]:=rhs(sol1):
f[1]:=_C1*sin(1/2*Pi*sqrt(2*lambda^2+2*sqrt(lambda^4+4*
s^4))*x)+_C2*cos(1/2*Pi*sqrt(2*lambda^2+2*sqrt(lambda^4
+4*s^4))*x)+_C3*sinh(1/2*Pi*sqrt(2)*sqrt(-
lambda^2+sqrt(lambda^4+4*s^4))*x)+_C4*cosh(1/2*Pi*sqrt(
2)*sqrt(-lambda^2+sqrt(lambda^4+4*s^4))*x):
f[1,x]:=diff(f[1],x):
f[1,xx]:=diff(f[1],x,x):
f[1,xxx]:=diff(f[1],x,x,x):
eq2:=diff(f2(x),x,x,x,x)+(lambda^2*Pi^2*diff(f2(x),x,x)
)+((K-s^4)*Pi^4*f2(x))=0:
sol2[inicial]:=dsolve(eq2):
sol2a:=subs(_C1=_C5,_C2=_C6,_C3=_C7,_C4=_C8,sol2[inicia
```

```
l]):   sol2:=rhs(sol2a):    finicial[2]:=sol2:
g:=alpha^4+(lambda^2*Pi^2)*alpha^2+(K-s^4)*Pi^4=0:
R1:=solve(g,alpha):              a:=Re(R1[1]):
b:=Im(R1[1]):
if (a>0) and (b>0) then
f[2]:=(exp(a*x))*(_C5*sin(b*x)+_C6*cos(b*x))+(exp(-
a*x))*(_C7*sin(b*x)+_C8*cos(b*x)) else if (a>0) and
(b<0) then f[2]:=(exp(a*x))*(_C5*sin(-b*x)+_C6*cos(-
b*x))+(exp(-a*x))*(_C7*sin(-b*x)+_C8*cos(-b*x)) else if
(a<0) and (b>0) then
f[2]:=(exp(a*x))*(_C5*sin(b*x)+_C6*cos(b*x))+(exp(-
a*x))*(_C7*sin(b*x)+_C8*cos(b*x)) else if (a<0) and
(b<0) then f[2]:=(exp(a*x))*(_C5*sin(-b*x)+_C6*cos(-
b*x))+(exp(-a*x))*(_C7*sin(-b*x)+_C8*cos(-b*x)) end if:
end if:end if: end if:
f[2,x]:=diff(f[2],x):
f[2,xx]:=diff(f[2],x,x):
f[2,xxx]:=diff(f[2],x,x,x):
exp1:=eval(f[1],x=1/2):
exp2:=eval(f[1,xx],x=1/2):
exp3:=eval(f[2],x=0):
exp4:=eval(f[2,xx],x=0):
f[1,x=0]:=eval(f[1],x=0):
f[2,x=a]:=eval(f[2],x=1/2):
f[1,x_x=0]:=eval(f[1,x],x=0):
f[2,x_x=a]:=eval(f[2,x],x=1/2):
f[1,xx_x=0]:=eval(f[1,xx],x=0):
f[2,xx_x=a]:=eval(f[2,xx],x=1/2):
f[1,xxx_x=0]:=eval(f[1,xxx],x=0):
f[2,xxx_x=a]:=eval(f[2,xxx],x=1/2):
exp5:=f[1,x=0]-f[2,x=a]:
exp6:=f[1,x_x=0]-f[2,x_x=a]:
exp7:=f[1,xx_x=0]-f[2,xx_x=a]:
exp8:=f[1,xxx_x=0]-f[2,xxx_x=a]:
a11:=coeff(exp1,_C1):
a21:=coeff(exp2,_C1):a31:=coeff(exp3,_C1):a41:=coeff(ex
p4,_C1):a51:=coeff(exp5,_C1):a61:=coeff(exp6,_C1):a71:=
coeff(exp7,_C1):a81:=coeff(exp8,_C1):a12:=coeff(exp1,_C
2):a22:=coeff(exp2,_C2):a32:=coeff(exp3,_C2):a42:=coeff
(exp4,_C2):a52:=coeff(exp5,_C2):a62:=coeff(exp6,_C2):a7
2:=coeff(exp7,_C2):a82:=coeff(exp8,_C2):a13:=coeff(exp1
,_C3):a23:=coeff(exp2,_C3):a33:=coeff(exp3,_C3):a43:=co
eff(exp4,_C3):a53:=coeff(exp5,_C3):a63:=coeff(exp6,_C3)
:a73:=coeff(exp7,_C3):a83:=coeff(exp8,_C3):a14:=coeff(e
xp1,_C4):a24:=coeff(exp2,_C4):a34:=coeff(exp3,_C4):a44:
=coeff(exp4,_C4):a54:=coeff(exp5,_C4):a64:=coeff(exp6,_
C4):a74:=coeff(exp7,_C4):a84:=coeff(exp8,_C4):a15:=coef
f(exp1,_C5):a25:=coeff(exp2,_C5):a35:=coeff(exp3,_C5):a
45:=coeff(exp4,_C5):a55:=coeff(exp5,_C5):a65:=coeff(exp
6,_C5):a75:=coeff(exp7,_C5):a85:=coeff(exp8,_C5):a16:=c
oeff(exp1,_C6):a26:=coeff(exp2,_C6):a36:=coeff(exp3,_C6
```

```
):a46:=coeff(exp4,_C6):a56:=coeff(exp5,_C6):a66:=coeff(
exp6,_C6):a76:=coeff(exp7,_C6):a86:=coeff(exp8,_C6):a17
:=coeff(exp1,_C7):a27:=coeff(exp2,_C7):a37:=coeff(exp3,
_C7):a47:=coeff(exp4,_C7):a57:=coeff(exp5,_C7):a67:=coe
ff(exp6,_C7):a77:=coeff(exp7,_C7):a87:=coeff(exp8,_C7):
a18:=coeff(exp1,_C8):a28:=coeff(exp2,_C8):a38:=coeff(ex
p3,_C8):a48:=coeff(exp4,_C8):a58:=coeff(exp5,_C8):a68:=
coeff(exp6,_C8):a78:=coeff(exp7,_C8):a88:=coeff(exp8,_C
8):
A:=Matrix([[a11,a12,a13,a14,a15,a16,a17,a18],[a21,a22,a
23,a24,a25,a26,a27,a28],[a31,a32,a33,a34,a35,a36,a37,a3
8],[a41,a42,a43,a44,a45,a46,a47,a48],[a51,a52,a53,a54,a
55,a56,a57,a58],[a61,a62,a63,a64,a65,a66,a67,a68],[a71,
a72,a73,a74,a75,a76,a77,a78],[a81,a82,a83,a84,a85,a86,a
87,a88]]): det[s]:= print (simplify(Determinant(A)), s)
end do:
```

#Observações:
#1) Conforme visto durante a tese cada trecho, seja enterrado ou desenterrado, requer um estudo de sua respectiva resposta para a equação diferencial proposta. O programa neste caso, está preparado para qualquer solução dentro da restrição em s<= (K-(λ^4/4))^1/4. Aconselha-se traçar, portanto, o gráfico das freqüência x rigidez da fundação pelo método de Ritz:
#2) Os valores explicitados no final do programa são: o valor do determinante e o seu respectivo parâmetro de freqüência.

###Representação dos modos de vibração e criação do arquivo para exportação de dados###

```
> restart:
> with(DEtools):
> with(LinearAlgebra):
> Digits:=64:
```

#Resolução para os casos em que: s<= (K-(λ^4/4)^1/4:
#1a. freqüência de vibração - Sol. exata:

```
> s1:=2.045482:
> s:=s1:
> K:=2000:
>lambda_crit:=2.516811209107670508859218:
>lambda:=0.5*lambda_crit:
>
eq1:=diff(f1(x),x,x,x,x)+(lambda^2*Pi^2*diff(f1(x),x,x)
)-(s^4*Pi^4*f1(x))=0:
sol1:=dsolve(eq1): finicial[1]:=rhs(sol1):
```

```
>
f[1]:=_C1*sin(1/2*Pi*sqrt(2*lambda^2+2*sqrt(lambda^4+4*
s^4))*x)+_C2*cos(1/2*Pi*sqrt(2*lambda^2+2*sqrt(lambda^4
+4*s^4))*x)+_C3*sinh(1/2*Pi*sqrt(2)*sqrt(-
lambda^2+sqrt(lambda^4+4*s^4))*x)+_C4*cosh(1/2*Pi*sqrt(
2)*sqrt(-lambda^2+sqrt(lambda^4+4*s^4))*x):
> f[1,x]:=diff(f[1],x):
> f[1,xx]:=diff(f[1],x,x):
> f[1,xxx]:=diff(f[1],x,x,x):
>
eq2:=diff(f2(x),x,x,x,x)+(lambda^2*Pi^2*diff(f2(x),x,x)
)+((K-s^4)*Pi^4*f2(x))=0:
> sol2[inicial]:=dsolve(eq2):
>
sol2a:=subs(_C1=_C5,_C2=_C6,_C3=_C7,_C4=_C8,sol2[inicia
l]):   > sol2:=rhs(sol2a):    finicial[2]:=sol2:
> g:=alpha^4+(lambda^2*Pi^2)*alpha^2+(K-s^4)*Pi^4=0:
R1:=solve(g,alpha):              a:=Re(R1[1]):
b:=Im(R1[1]):
if (a>0) and (b>0) then
f[2]:=(exp(a*x))*(_C5*sin(b*x)+_C6*cos(b*x))+(exp(-
a*x))*(_C7*sin(b*x)+_C8*cos(b*x)) else if (a>0) and
(b<0) then f[2]:=(exp(a*x))*(_C5*sin(-b*x)+_C6*cos(-
b*x))+(exp(-a*x))*(_C7*sin(-b*x)+_C8*cos(-b*x)) else if
(a<0) and (b>0) then
f[2]:=(exp(a*x))*(_C5*sin(b*x)+_C6*cos(b*x))+(exp(-
a*x))*(_C7*sin(b*x)+_C8*cos(b*x)) else if (a<0) and
(b<0) then f[2]:=(exp(a*x))*(_C5*sin(-b*x)+_C6*cos(-
b*x))+(exp(-a*x))*(_C7*sin(-b*x)+_C8*cos(-b*x)) end if:
end if:end if: end if:
> f[2,x]:=diff(f[2],x):
> f[2,xx]:=diff(f[2],x,x):
> f[2,xxx]:=diff(f[2],x,x,x):
> exp1:=eval(f[1],x=1/2):
> exp2:=eval(f[1,xx],x=1/2):
> exp3:=eval(f[2],x=0):
> exp4:=eval(f[2,xx],x=0):
> f[1,x=0]:=eval(f[1],x=0):
> f[2,x=a]:=eval(f[2],x=1/2):
> f[1,x_x=0]:=eval(f[1,x],x=0):
> f[2,x_x=a]:=eval(f[2,x],x=1/2):
> f[1,xx_x=0]:=eval(f[1,xx],x=0):
> f[2,xx_x=a]:=eval(f[2,xx],x=1/2):
> f[1,xxx_x=0]:=eval(f[1,xxx],x=0):
> f[2,xxx_x=a]:=eval(f[2,xxx],x=1/2):
> exp5:=f[1,x=0]-f[2,x=a]:
> exp6:=f[1,x_x=0]-f[2,x_x=a]:
> exp7:=f[1,xx_x=0]-f[2,xx_x=a]:
> exp8:=f[1,xxx_x=0]-f[2,xxx_x=a]:
```

```
> a11:=coeff(exp1,_C1):
a21:=coeff(exp2,_C1):a31:=coeff(exp3,_C1):a41:=coeff(ex
p4,_C1):a51:=coeff(exp5,_C1):a61:=coeff(exp6,_C1):a71:=
coeff(exp7,_C1):a81:=coeff(exp8,_C1):a12:=coeff(exp1,_C
2):a22:=coeff(exp2,_C2):a32:=coeff(exp3,_C2):a42:=coeff
(exp4,_C2):a52:=coeff(exp5,_C2):a62:=coeff(exp6,_C2):a7
2:=coeff(exp7,_C2):a82:=coeff(exp8,_C2):a13:=coeff(exp1
,_C3):a23:=coeff(exp2,_C3):a33:=coeff(exp3,_C3):a43:=co
eff(exp4,_C3):a53:=coeff(exp5,_C3):a63:=coeff(exp6,_C3)
:a73:=coeff(exp7,_C3):a83:=coeff(exp8,_C3):a14:=coeff(e
xp1,_C4):a24:=coeff(exp2,_C4):a34:=coeff(exp3,_C4):a44:
=coeff(exp4,_C4):a54:=coeff(exp5,_C4):a64:=coeff(exp6,_
C4):a74:=coeff(exp7,_C4):a84:=coeff(exp8,_C4):a15:=coef
f(exp1,_C5):a25:=coeff(exp2,_C5):a35:=coeff(exp3,_C5):a
45:=coeff(exp4,_C5):a55:=coeff(exp5,_C5):a65:=coeff(exp
6,_C5):a75:=coeff(exp7,_C5):a85:=coeff(exp8,_C5):a16:=c
oeff(exp1,_C6):a26:=coeff(exp2,_C6):a36:=coeff(exp3,_C6
):a46:=coeff(exp4,_C6):a56:=coeff(exp5,_C6):a66:=coeff(
exp6,_C6):a76:=coeff(exp7,_C6):a86:=coeff(exp8,_C6):a17
:=coeff(exp1,_C7):a27:=coeff(exp2,_C7):a37:=coeff(exp3,
_C7):a47:=coeff(exp4,_C7):a57:=coeff(exp5,_C7):a67:=coe
ff(exp6,_C7):a77:=coeff(exp7,_C7):a87:=coeff(exp8,_C7):
a18:=coeff(exp1,_C8):a28:=coeff(exp2,_C8):a38:=coeff(ex
p3,_C8):a48:=coeff(exp4,_C8):a58:=coeff(exp5,_C8):a68:=
coeff(exp6,_C8):a78:=coeff(exp7,_C8):a88:=coeff(exp8,_C
8):
>
A:=Matrix([[a11,a12,a13,a14,a15,a16,a17,a18],[a21,a22,a
23,a24,a25,a26,a27,a28],[a31,a32,a33,a34,a35,a36,a37,a3
8],[a41,a42,a43,a44,a45,a46,a47,a48],[a51,a52,a53,a54,a
55,a56,a57,a58],[a61,a62,a63,a64,a65,a66,a67,a68],[a71,
a72,a73,a74,a75,a76,a77,a78],[a81,a82,a83,a84,a85,a86,a
87,a88]]):
```

#Configuração dos Modos:
#Criação da Matriz B que representa os valores da Matriz A, após a substituição dos valores de todos os parâmetros.

```
> b11:=simplify(a11):
> b12:=simplify(a12):
> b13:=simplify(a13):
> b14:=simplify(a14):
> b15:=simplify(a15):
> b16:=simplify(a16):
> b17:=simplify(a17):
> b18:=simplify(a18):
> b21:=simplify(a21):
> b22:=simplify(a22):
> b23:=simplify(a23):
> b24:=simplify(a24):
> b25:=simplify(a25):
```

```
> b26:=simplify(a26):
> b27:=simplify(a27):
> b28:=simplify(a28):
> b31:=simplify(a31):
> b32:=simplify(a32):
> b33:=simplify(a33):
> b34:=simplify(a34):
> b35:=simplify(a35):
> b36:=simplify(a36):
> b37:=simplify(a37):
> b38:=simplify(a38):
> b41:=simplify(a41):
> b42:=simplify(a42):
> b43:=simplify(a43):
> b44:=simplify(a44):
> b45:=simplify(a45):
> b46:=simplify(a46):
> b47:=simplify(a47):
> b48:=simplify(a48):
> b51:=simplify(a51):
> b52:=simplify(a52):
> b53:=simplify(a53):
> b54:=simplify(a54):
> b55:=simplify(a55):
> b56:=simplify(a56):
> b57:=simplify(a57):
> b58:=simplify(a58):
> b61:=simplify(a61):
> b62:=simplify(a62):
> b63:=simplify(a63):
> b64:=simplify(a64):
> b65:=simplify(a65):
> b66:=simplify(a66):
> b67:=simplify(a67):
> b68:=simplify(a68):
> b71:=simplify(a71):
> b72:=simplify(a72):
> b73:=simplify(a73):
> b74:=simplify(a74):
> b75:=simplify(a75):
> b76:=simplify(a76):
> b77:=simplify(a77):
> b78:=simplify(a78):
> b81:=simplify(a81):
> b82:=simplify(a82):
> b83:=simplify(a83):
> b84:=simplify(a84):
```

```
> b85:=simplify(a85):
> b86:=simplify(a86):
> b87:=simplify(a87):
> b88:=simplify(a88):
>
B:=Matrix([[b11,b12,b13,b14,b15,b16,b17,b18],[b21,b22,b
23,b24,b25,b26,b27,b28],[b31,b32,b33,b34,b35,b36,b37,b3
8],[b41,b42,b43,b44,b45,b46,b47,b48],[b51,b52,b53,b54,b
55,b56,b57,b58],[b61,b62,b63,b64,b65,b66,b67,b68],[b71,
b72,b73,b74,b75,b76,b77,b78],[b81,b82,b83,b84,b85,b86,b
87,b88]]):
```

#Eliminação da última linha e da 3a. coluna (considerar-se-á C3=1):

```
>
W:=Matrix([[b11,b12,b14,b15,b16,b17,b18],[b21,b22,b24,b
25,b26,b27,b28],[b31,b32,b34,b35,b36,b37,b38],[b41,b42,
b44,b45,b46,b47,b48],[b51,b52,b54,b55,b56,b57,b58],[b61
,b62,b64,b65,b66,b67,b68],[b71,b72,b74,b75,b76,b77,b78]
]):
> V:=Vector([_C1,_C2,_C4,_C5,_C6,_C7,_C8]):
> Q:=W.V:
> R:=Vector([-b13,-b23,-b33,-b43,-b53,-b63,-b73]):
> sols:=solve({Q[1]=R[1], Q[2]=R[2], Q[3]=R[3],
Q[4]=R[4], Q[5]=R[5], Q[6]=R[6], Q[7]=R[7]},{
_C1,_C2,_C4,_C5,_C6,_C7,_C8}):
>
V:=subs(sols[1],sols[2],sols[3],sols[4],sols[5],sols[6]
,sols[7],V):
```

#Normalização do Vetor V:

```
>
ampnorm:=sqrt(((V[1])^2)+((V[2])^2)+((V[3])^2)+((V[4])
^2)+((V[5])^2)+((V[6])^2)+((V[7])^2)+1)):
> Vnorm:=V/ampnorm:
```

#Observação: Todas as constantes acima estão sendo multiplicadas por _C3, a qual é considerada igual à unidade.

```
> C3:=1/ampnorm:
>
C:=Vector([Vnorm[1],Vnorm[2],C3,Vnorm[3],Vnorm[4],Vnorm
[5],Vnorm[6],Vnorm[7]]):
>
> ee1:=_C1=C[1]:
> ee2:=_C2=C[2]:
> ee3:=_C3=C[3]:
> ee4:=_C4=C[4]:
> ee5:=_C5=C[5]:
> ee6:=_C6=C[6]:
```

```
> ee7:=_C7=C[7]:
> ee8:=_C8=C[8]:
```

#Substituição dos valores das constantes nas expressões para os deslocamentos:

```
> f[1]:=subs(ee1,ee2,ee3,ee4,ee5,ee6,ee7,ee8,f[1]):
> f[2]:=subs(ee1,ee2,ee3,ee4,ee5,ee6,ee7,ee8,f[2]):
> with(plottools):
>plot([Re(f[1]),x,x=0..0.5]):
>plot([Re(f[2]),x,x=0..0.5]):
```

#Determinação do arquivo para exportação dos valores encontrados nos gráficos anteriores

#Para o gráfico de f1(x):

#Definição dos limites e do intervalo de variação de x:

```
> inf:=0:
> inter:=0.01:
> sup:=0.5:
```

#Definçäo dos dados para construção da tabela e do gráfico:

```
> cont:=0:
for x from inf by inter to sup do
   cont:=cont+1
end do: cont:
> F1:=array(1..cont):
> X:=array(1..cont):
> cont:=0:
```

#Arquivo de saída:

```
> fd := fopen("c:\\saidaf1.dat",WRITE,BINARY):
> for x from inf by inter to sup do
    cont:=cont+1;
    X[cont]:=x;
    F1[cont]:=simplify(f[1]);
    fprintf(fd,"%f  , %f \n",X[cont],Re(F1[cont]));
end do:
> fclose(fd);
```

#Para o gráfico de f2(x):

#Definição dos limites e do intervalo de variação de x:

```
> inf:=0:
> inter:=0.01:
> sup:=0.5:
```

#Definição dos dados para construção da tabela e do gráfico:

```
> cont:=0:
for x from inf by inter to sup do
   cont:=cont+1
end do: cont:
```

```
> F2:=array(1..cont):
> X:=array(1..cont):
> cont:=0:

#Arquivo de saída:
> fd := fopen("c:\\saidaf2.dat",WRITE,BINARY):
> for x from inf by inter to sup do
    cont:=cont+1;
    X[cont]:=x;
    F2[cont]:=simplify(f[2]);
    fprintf(fd,"%f  , %f \n",X[cont],Re(F2[cont]));
end do:
> fclose(fd);
```

9.2. Análise Não-Linear

A análise não-linear consistiu em duas etapas: o estudo da relação freqüência e amplitude na vibração livre e o estudo da ressonância não-linear para a vibração forçada. Construiu-se, portanto, dois programas que possibilitaram estes estudos:

- Análise não-linear da vibração livre: resolução da equação diferencial não-linear pelo método do Balanço Harmônico.
- Análise não-linear da vibração forçada: resolução da equação diferencial não-linear pelo método do Balanço Harmônico. Têm-se os carregamentos estático e dinâmico atuando e consideram-se os efeitos de forças de dissipação.

Os cálculos apresentados referem-se à coluna biapoiada com a profundidade da fundação atingindo a metade da altura, rigidez igual a 2.000 e carregamento estático igual a 50% da carga crítica. Para o caso de vibração forçada, a amplitude da força harmônica considerada é de 0,5 e o fator de amortecimento é igual à 0,01.

###Análise não-linear da vibração livre: resolução da equação diferencial não-linear pelo método do balanço harmônico###

```
> restart;
> with(DEtools):
> with(LinearAlgebra):
> Digits:=32
```

#apenas para facilitar a visualização das expressões foi usada no programa a variável x ao invés da letra grega ζ, que representa a coordenada adimensional do eixo axial.

#Equação original da energia:

```
>
eq_orig:=int((E*J/2)*(diff(w(x),x,x)^2+diff(w(x),x,x)^2
*diff(w(x), x)^2+1/4*(diff(w(x),x,x)^2*diff(w(x),
x)^4))-(P*Pi^2/2)*(diff(w(x),x)^2+1/4*diff(w(x),x)^4)-
(M*omega^4*Pi^4/2)*w(x)^2,x=0..1)+int(k*Pi^4*w(x)^2/2,x
=0..1/2):
```

#Substituições:

```
> P/(E*J)=lambda^2:
> (M*omega^4)/(E*J)=omega^4:
> k/(E*J)=K:
```

#Desmembrando a integral acima de forma tendo-se separadamente os termos da parte desenterrada e enterrada da estaca:

```
>
eq:=int((diff(w1(x,t),x,x)^2+diff(w1(x,t),x,x)^2*diff(w
1(x,t), x)^2+1/4*(diff(w1(x,t),x,x)^2*diff(w1(x,t),
x)^4))-
(lambda^2*Pi^2)*(diff(w1(x,t),x)^2+1/4*diff(w1(x,t),x)^
4),x=0..1/2)+int((diff(w2(x,t),x,x)^2+diff(w2(x,t),x,x)
^2*diff(w2(x,t),
x)^2+1/4*(diff(w2(x,t),x,x)^2*diff(w2(x,t), x)^4))-
(lambda^2*Pi^2)*(diff(w2(x,t),x)^2+1/4*diff(w2(x,t),x)^
4),x=0..1/2)-(M^4*Pi^4* int(diff(w1(x,t),t)^2,
x=0..1/2)+(M^4*Pi^4* int(diff(w2(x,t),t)^2,
x=0..1/2)))+int(K*Pi^4*w2(x,t)^2,x = 0 .. 1/2):
> K:=2000:
> lambda_crit:=2.516811209107670508859218:
> lambda:=0.5*lambda_crit:
```

#Expressões de w1 e w2 obtidas a partir do programa para a análise linear:

```
> w1(x,t):=C(t)*(-
.74226998589049586523524259304748*sin(2.247231744935770
6756377467276767*Pi*x)+.30368363027451847980453138713522
*cos(2.247231744935770675637746727676 7*Pi*x)+.42359673
16369279094513811017838 8*sinh(1.316523043830711889835 44
```

```
07656926*Pi*sqrt(2)*x)-
.42116186221485765029734644136053*cosh(1.31652304383071
18898354407656926*Pi*sqrt(2)*x)):
> w2(x,t):=C(t)*(exp(-
14.690669053927442687368922937633*x)*(-
.86993267354275547566290318475451e-
4*sin(14.954278128496788954221564677256*x)-
.13710967143026508747295478552280e-
4*cos(14.954278128496788954221564677256*x))+exp(14.6906
690539274426873689229376 33*x)*(-
.86993267354275547566290318475451e-
4*sin(14.954278128496788954221564677256*x)+.13710967143
026508747295478552280e-
4*cos(14.954278128496788954221564677256*x))):
```

#Substituição das expressões de w1 e w2 multiplicadas pela amplitude C(t) na expressão do Funcional:

```
> eq1:=simplify(eq):
```

#Eliminando os termos imaginários, tem-se:

```
> eq1 :=
331.55224356674151466587840759916*C(t)^2+1999.873329687
733976446973461550 3*C(t)^4+9061.881801817506165303992655
2239*C(t)^6-
18.939518226994037017095151352587*M^4*diff(C(t),t)^2:
```

#Construção da equação de movimento (equação diferencial não-linear):

```
> eq1a:=subs(diff(C(t),t)=AA,C(t)=C, eq1):
> eq1b:=diff(eq1a, C):
> eq1c:=subs(C=C(t), eq1b):
> eq1d:=diff(eq1a, AA):
> eq1e:=subs(AA=diff(C(t), t), eq1d):
> eq1f:=eq1c-(diff(eq1e, t)):
> eq2:=simplify(subs(C(t)=A1*sin(omega*t),eq1f)):
> eq3:=subs((cos(omega*t))^2=(1-sin(omega*t)^2),
(cos(omega*t))^4=(1-sin(omega*t)^2)^2, eq2):
> eq4 := expand(eq3):
```

#Substituição das expressões equivalentes de (sin(omega*t))^5 e (sin(omega*t))^3:

```
>
eq5:=simplify(subs((sin(omega*t))^3=3/4*sin(omega*t)-
1/4*sin(3*omega*t),(sin(omega*t))^5=5/8*sin(omega*t)-
5/16*sin(3*omega*t)+1/16*sin(5*omega*t), eq4)):
```

#Obtenção da expressão para o parâmetro da freqüência em função da amplitude A1:

```
> eq6:=expand(coeff(eq5, sin(omega*t))):
> isolate(eq6, omega^2):
```

###Análise não-linear da vibração forçada: resolução da equação diferencial não-linear pelo método do balanço harmônico. Têm-se os carregamentos estático e dinâmico atuantes e consideram-se os efeitos de forças de dissipação###

```
> restart:
> with(DEtools):
> with(LinearAlgebra):
> Digits:=32:
```

#A partir do funcional já utilizado para a vibração livre tem-se, acrescentando as parcelas referentes a força de dissipação, a seguinte expressão:

```
>
eq:=int((diff(w1(x,t),x,x)^2+diff(w1(x,t),x,x)^2*diff(w
1(x,t), x)^2+1/4*(diff(w1(x,t),x,x)^2*diff(w1(x,t),
x)^4))-
(lambda^2*Pi^2)*(diff(w1(x,t),x)^2+1/4*diff(w1(x,t),x)^
4),x=0..1/2)+int((diff(w2(x,t),x,x)^2+diff(w2(x,t),x,x)
^2*diff(w2(x,t),
x)^2+1/4*(diff(w2(x,t),x,x)^2*diff(w2(x,t), x)^4))-
(lambda^2*Pi^2)*(diff(w2(x,t),x)^2+1/4*diff(w2(x,t),x)^
4),x=0..1/2)-(M^4*Pi^4* int(diff(w1(x,t),t)^2,
x=0..1/2)+(M^4*Pi^4* int(diff(w2(x,t),t)^2,
x=0..1/2)))+int(K*Pi^4*w2(x,t)^2,x = 0 ..
1/2)+beta[1]*int((diff(w1(x,t), t))^2,
x=0..1/2)+beta[1]*int((diff(w2(x,t), t))^2,
x=0..1/2)+2*int(p(t)*w1(x,t),
x=0..1/2)+2*int(p(t)*w2(x,t), x=0..1/2):
> K:=2000:
> lambda_crit:=2.516811209107670508859218:
> lambda:=0.5*lambda_crit:
```

#Expressões de w1 e w2 obtidas a partir do programa para a análise linear:

```
> w1(x,t):=C(t)*(-
.74226998589049586523524259304748*sin(2.247231744935770
67563774672767*Pi*x)+.30368363027451847980453138713522
*cos(2.247231744935770675637746727676 7*Pi*x)+.42359673
16369279094513811017838 8*sinh(1.31652304383071188983544
07656926*Pi*sqrt(2)*x)-
.42116186221485765029734644136053*cosh(1.31652304383071
18898354407656926*Pi*sqrt(2)*x)):
> w2(x,t):=C(t)*(exp(-
14.690669053927442687368922937633*x)*(-
.86993267354275547566290318475451e-
4*sin(14.954278128496788954221564677256*x)-
.13710967143026508747295478552280e-
4*cos(14.954278128496788954221564677256*x))+exp(14.6906
69053927442687368922937633*x)*(-
.86993267354275547566290318475451e-
4*sin(14.954278128496788954221564677256*x)+.13710967143
```

```
026508747295478552280e-
4*cos(14.954278128496788954221564677256*x))):
> p(t):=A0*sin(omega*t):
```

#Substituição das expressões de w1 e w2 multiplicadas pela amplitude C na expressão do Funcional:

```
> eq1:=simplify(eq):
```

#Determinação da Equação de Movimento :

```
> eq1a:=subs(diff(C(t),t)=AA,C(t)=C, eq1):
> eq1b:=diff(eq1a, C):
> eq1c:=subs(C=C(t), eq1b):
> eq1d:=diff(eq1a, AA):
> eq1e:=subs(AA=diff(C(t), t), eq1d):
> eq1f:=eq1c-(diff(eq1e, t)):
```

#Retirando-se os termos com números imaginários desprezíveis, tem-se:

```
> eq1g :=
663.10448713348302933175681519832*C(t)+7999.49331875093
590578789384620l2*C(t)^3+54371.290810905036991823955931
344*C(t)^5-
.569956780643459531161667144690003*A0*sin(omega*t)+37.87
903645398807403419030270516 6*M^4*diff(C(t),`$`(t,2))-
.388865516061182613139300938222 80*beta[1]*diff(C(t),`$`
(t,2)):
>
eq2:=simplify(subs(C(t)=A1*sin(omega*t)+A2*cos(omega*t)
,eq1g)):
```

#Substituições de seno e co-seno:

```
> eq3:=
simplify(subs((cos(omega*t))^3=3/4*cos(omega*t)+1/4*cos
(3*omega*t),(cos(omega*t))^5=5/8*cos(omega*t)+5/16*cos(
3*omega*t)+1/16*cos(5*omega*t),
(cos(omega*t))^2=1/2+1/2*(cos(2*omega*t)),
(cos(omega*t))^4=3/8+1/2*(cos(2*omega*t))+1/8*(cos(4*om
ega*t)),eq2)):
> eq3a:=collect(eq3, A1):
> eq3b:=collect(eq3a, A2):
> eq3c:=subs(sin(omega*t)*cos(2.*omega*t)=1/2*(sin(-
omega*t)+sin(3*omega*t)),
sin(omega*t)*cos(4.*omega*t)=1/2*(sin(-
3*omega*t)+sin(5*omega*t)),eq3b):
> eq4:=collect(eq3c, sin(omega*t)):
> eq5:=collect(eq4, cos(omega*t)):
> eqa:=coeff(eq5, sin(omega*t)):
> eqb:=coeff(eq5, cos(omega*t)):
```

###Implementação do Método de Newton Rapson:###

```
> restart:
> with(linalg):
> Digits:=32:

#Equações obtidas na etapa anterior:
> eqa:=
33982.056756815648119889972457090*A1*A2^4+33982.0567568
156481198899724570 89*A1^5+(5999.619989063201929340920 38
46510*A1+67964.113513631296239779944914180*A1^3)*A2^2+5
999.619989063201929340920 3846509*A1^3-
.569956780643459531161667144690 03*A0+(663.1044871334830
2933175681519832-
37.879036453988074034190302705166*M^4*omega^2+.38886551
606118261313930093822280*beta[1]*omega^2)*A1:
> eqb:=
(67964.113513631296239779944914180*A1^2+5999.6199890632
019293409203846509)*A2^3+(33982.056756815648119889972245
7090*A1^4+5999.619989063201929340920 3846510*A1^2+.38886
551606118261313930093822280*beta[1]*omega^2-
37.879036453988074034190302705166*M^4*omega^2+663.10448
713348302933175681519832)*A2+33982.05675681564811988997
2457090*A2^5:
> mf:= matrix(2,1,[eqa,eqb]):
> mk:= matrix(2,2,[ [diff(eqa,A1), diff(eqa,A2)] ,
[diff(eqb,A1), diff(eqb,A2)] ]):
> M:=0.702:
> omega:=8.490187813*.9:
> A0:=.5:
> beta[1]:=8.490187813*0.47:
>erro:=1e-10:
> imk:=inverse(mk):
> A1:=0.1: A2:=0.1: saida:=0:
> for i from 1 to 50 while (saida=0) do
  dC:=multiply(imk,mf):
  A1:=A1-dC[1,1]: A2:=A2-dC[2,1]:
  #print (mf[1,1], mf[2,1], dC[1,1], dC[2,1]):
 if (abs(dC[1,1])<erro and abs(dC[2,1])<1e^erro ) then
saida := 1: end if:
 end do:
 print (i,A1,A2):
```

#Observação:
#O valor de omega (parâmetro de freqüência) baseia-se no valor encontrado para a freqüência natural na vibração livre. A freqüência natural (neste caso, **W**n =

8,490187813) é obtida na etapa anterior, a partir da expressão final de omega, fazendo-se A1=0. Neste exemplo, (**W/W**n) = 0,9.