Cranfield

College of Aeronautics Report No.9203
January 1992

DYNAMICS OF SHUTTLE BASED FLEXIBLE ANTENNA SYSTEM

M.Moch and C.L.Kirk

College of Aeronautics
Cranfield Institute of Technology
Cranfield, Bedford MK43 0AL. England

Cranfield

College of Aeronautics Report No.9203
January 1992

DYNAMICS OF SHUTTLE BASED FLEXIBLE ANTENNA SYSTEM

M.Moch and C.L.Kirk

College of Aeronautics
Cranfield Institute of Technology
Cranfield, Bedford MK43 0AL. England

ISBN 1 871564 379

£10.00

ACKNOWLEDGEMENT

I wish to thank Prof. C. L. Kirk
for his support during this thesis.

Special thanks also to Dipl Ing. F. Karl,
Institut für Flugzeugbau, Universität Stuttgart
for software support.

This thesis is in partial fulfilment of the degree of Diploma Ingenieur in the Instut fur Statik und Dynamik, University of Stuttgart, Germany.
The work was carried out in the period June 1 - December 31, 1991.

Cranfield, 11 January 1992

ABSTRACT

The aim of this thesis is to formulate a reduced dynamical model for the Shuttle based astro-mast antenna system by using a four degree of freedom model. Four degrees of freedom are considered sufficiently representative of the response, caused by the application of step function torques giving acceleration and deceleration of the antenna dish. The aim is to obtain a model that can be clearly understood in terms of dynamics and control by limiting the analysis to only four degrees of freedom.

This thesis formulates the corresponding dynamical model of the astro-mast antenna system. Using this dynamical model, an open loop pitching maneuver is simulated, eigenvalue analysis is applied, and residual oscillations at the end of the maneuver, are damped out by the d.c. servo motor using closed loop feedback control.

CONTENTS

NOTATION :

Θ_1, C_1, α, t, . . .	All *scalars* are written in normal print. In small or capital letters.
Θ, **b**, **p**, . . .	All *vectors* are written in bold print. In small latin letters or arbitrary greek letters.
M, **K**, **C**, . . .	All *matrices* are written in bold print and in capital latin letters.

c	real constant [-]
I	moment of inertia [kgm^2]
D	damping constant [Nmsec]
F	dissipative function [Nm/sec]
F	force [N]
L	length [m]
K	crossover point [-]
m	mass per length/unit [kg/m]
M	lumped mass [kg]
M	torque of servo motor or bending moment [Nm]
V	strain energy [Nm]
W	work [Nm]
Q	generalized exciting force [N]
R	length ratio [-]
t	time [sec]
T	kinetic energy [Nm]
TE	total energy [Nm]
x,y,z	coordinates [-]

α	pitch angle [rad]
β	damping ratio [-]
ε	error [-]
λ	stiffness of torsional springs [Nm/rad]
Λ	normalized stiffness of torsional springs [$1/sec^2$]
τ	discrete time value [sec]
Θ	generalized angle [rad]
φ	phase lag [rad]
Φ	shape function [-]
σ	real part of complex eigenvalue [1/sec]
ω	natural frequency or complex part of complex eigenvalue [rad/sec]

$\mathbf{A}$, $\mathbf{B}$	state space differential equation system matrices
$\mathbf{B}$	control influence matrix
$\mathbf{C}$	structural damping matrix
$\mathbf{G}$	gain matrix
$\mathbf{K}$	stiffness matrix
$\mathbf{M}$	mass matrix
$\mathbf{W}$	weigting matrix

$\mathbf{c}$	vector of constants
$\mathbf{Q}$	vector of generalized exciting forces
$\mathbf{p}$	parameter vector
$\mathbf{u}$	control vector
$\mathbf{x}$	configuration vector
$\boldsymbol{\Upsilon}$	object vector
$\boldsymbol{\Lambda}$	Lagrange multiplier vector
$\boldsymbol{\Theta}$	vector of generalized coordinates
$\boldsymbol{\Phi}$	(right) eigenvector
$\boldsymbol{\Psi}$	left eigenvector

1. INTRODUCTION

The aim of this thesis is to formulate a reduced dynamical model for the Shuttle based astro-mast antenna system by using a four degree of freedom system. These are considered sufficently representative of the response, caused by the application of step function torques giving acceleration and deceleration of the antenna dish. The aim is to obtain a model that can be clearly understood in terms of dynamics and control by limiting the analysis to only four degrees of freedom.

The aim of such antenna experiments was to perform accurate maneuvers through a prescribed angle, so that the antenna is pointed to a target in space, and to verify mathematical modelling and simulation. Vibration due to the maneuver must not be excessive and at the end of the maneuver it is required to use active vibration control, feeding back the systems displacements and velocities into the gimbal servo motor, in order to reduce the residual vibration, caused by the maneuver, to zero as quickly as possible.

This thesis does not examine yawing motions, which is the subject of another investigation. Normally a maneuver, involving pitch and yaw, does perform, say, pitch first, followed by the yaw maneuver, after the oscillations from the first maneuver have beeen damped out, since it is much more complex to consider both maneuvers simultaneously.

It is assumed that the Shuttle does not rotate due to the small torque transmitted from the base of the mast. The degree of rotation is deteremined as a check.

2. DYNAMIC MODEL OF ASTRO-MAST ANTENNA SYSTEM

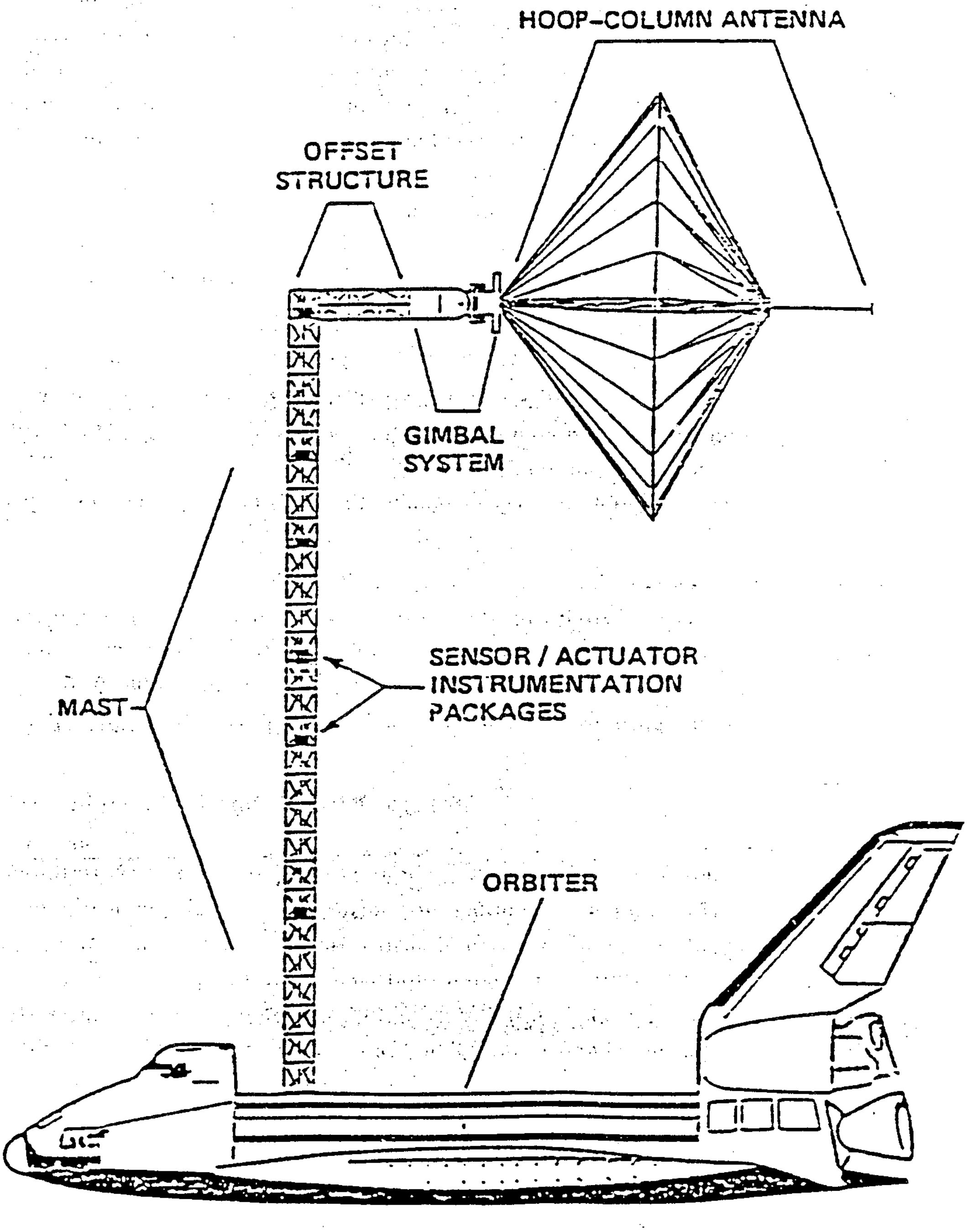


Fig. 2.1 : ASTRO-MAST ANTENNA SYSTEM, MOUNTED ON THE SHUTTLE

2.1 INTRODUCTION

The purpose of this section, is to define a dynamic model of the astro-mast antenna system. As a dynamic model, we choose an aquivalent double pendulum system, for which the Lagrangian approach is applicable.

The astro-mast system and the antenna system are linked together by the servo motor pivot. For the purpose of dynamic modelling, both systems can be considered on their own.

2.2 NATURAL FREQUENCIES OF ASTRO-MAST

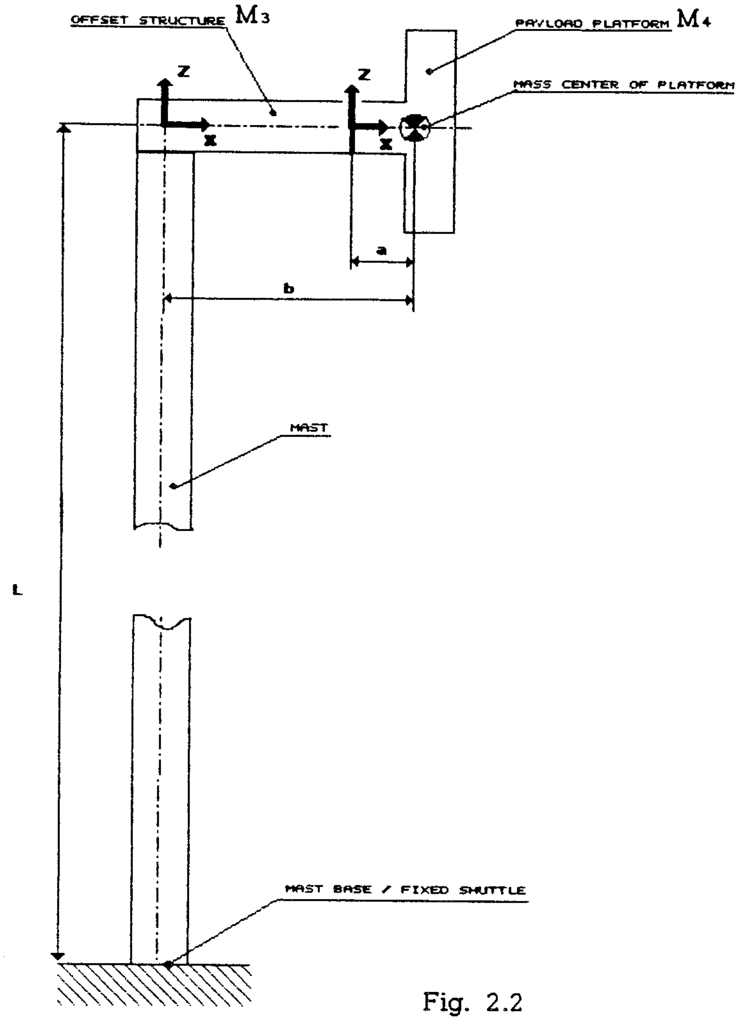


Fig. 2.2

The astro-mast consists of the mast, the offset structure and the payload platform (fig. 2.2).
Data from NASA report (Ref. 1):

length of mast :	$L = 60.693$ m
mass per length/unit:	m_1, m_2 see later discussions.
(constant) bending stiffness of mast:	$EI_y = 32.39$ E6 Nm^2
natural frequencies of mast only:	$\omega_1 = 1.294$ rad/sec
	ω_2 0 9.022 rad/sec
mass of offset structure:	$m_3 = 102.409$ kg
mass of payload platform:	$m_4 = 172.176$ kg
moment of inertia of offset structure (left frame of ref.):	$I_3 = 810.683$ kgm^2
moment of inertia of platform (right frame of ref.):	$I_4 = 31.54$ kgm^2
lengths:	$a = 0.358$ m
	$b = 3.327$ m

The natural frequencies of the astro-mast (including offset structure and payload platform) are unknown. The astro-mast can be idealized as a flexible cantilever beam with a point mass and rotary inertia at the free end:

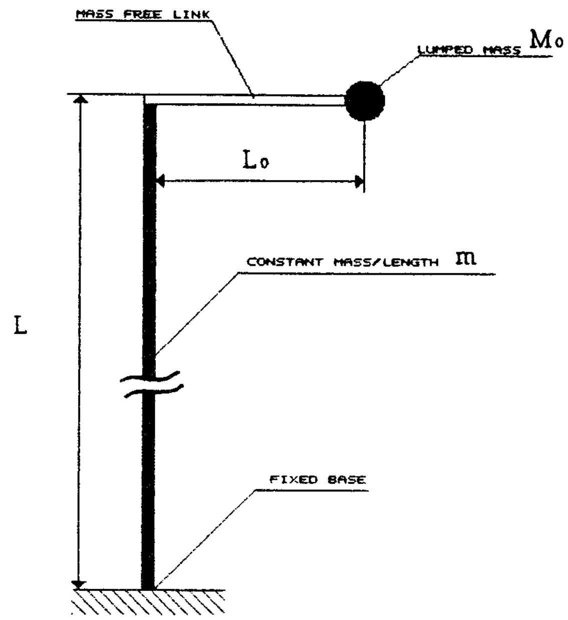


Fig. 2.3 : IDEALIZED ASTRO-MAST

We will use this idealized astro-mast, to evaluate the first 2 natural frequencies and mode shapes. First, we need to determine the parameters L_o , M_o and m of the idealized structure.

M_o is equal to the total mass of offset structure and payload platform:

$$M_o = m_3 + m_4 \quad (2.1)$$

$$M_o = 274.585 \text{ kg}$$

The total moment of inertia of the offset structure and payload platform with respect to the longitudinal axis of the mast (fig. 2.2) is:

$$I_p = I_3 + I_{4s} + m_4 b^2 \quad (2.2)$$

where I_{4s} is the moment of inertia of the payload platform with respect to the center of mass of the platform:

$$I_{4s} = I_4 - m_4 a^2 \qquad (2.3)$$

Thus, the total moment of inertia is:

$$I_p = I_3 + I_4 + m_4(b^2 - a^2) \qquad (2.4)$$

$$I_p = 2725.777 \text{ kgm}^2$$

For same moment of inertia, the length of the mass free link must be:

$$L_o = \sqrt{I/M_o} \qquad (2.5)$$

$$L_o = 3.150 \text{ m}$$

The natural frequencies of the uniform mast are known. We idealize the mast by a flexible cantilever beam, which has the same natural frequencies. Natural frequencies for the uniform cantilever beam are given by:

$$\omega_n = (a_n L)^4 \sqrt{I/mL^4} \qquad (2.6)$$

Where:

$a_1L = 1.875$ for the first mode.

$a_2L = 4.694$ for the second mode.

For given bending stiffness EI and natural frequencies ω_1, ω_2, the corresponding mass/length of the beam is:

$$m_1 = (1.875)^8 EI / (\omega_1^2 L^4) \quad \text{first mode}$$
$$m_2 = (4.694)^8 EI / (\omega_2^2 L^4) \quad \text{second mode} \qquad (2.7)$$

$m_1 = 17.623$ kg/m $\qquad$ $m_2 = 14.238$ kg/m

One would expect, that the masses m_1 for the first mode and m_2 for the second mode are the same. These are not the same, because the mass/length of the original mast is not constant, due to several instrumentation packages mounted on the mast. This results in slightly different natural frequencies, which we would obtain from the idealization as flexible cantilever beam with constant mass/length. We compensate this error, if we use m_1 for the first mode, and m_2 for the second mode.

Now, the idealized astro-mast, as flexible cantilever beam with additional lumped mass and rotary inertia at the free end, is defined:

length of idealized mast = length of original mast :	$L = 60.693$ m
Bending stiffness of idealized mast = bending stiffness of original mast :	$EI = 32.39\,Nm^2$
Mass/length of idealized mast is constant, for first mode:	$m_1 = 17.623$ kg/m
for second mode:	$m_2 = 14.238$ kg/m
Tip mass :	$M_0 = 274.5$ kg
Tip inertia :	$I_p = 2725.77$ kgm^2

The natural frequencies are the roots of the harmonic function (appendix A) :

$$\begin{aligned} &1 + \cos(aL)\cosh(aL) + \\ &+ (M\omega^2/EIa^3)\ [\cos(aL)\sinh(aL) - \sin(aL)\cosh(aL)] + \\ &- (I_p\omega^2/EIa)\ [\cos(aL)\sinh(aL) + \sin(aL)\cosh(aL)] + \\ &+ (M I_p\omega^2/EIm)\ [1 - \cos(aL)\cosh(aL)] = 0 \end{aligned} \tag{2.8}$$

Where:

$$a^4 = \omega^2 m / EI \tag{2.8a}$$

The corresponding shape function is:

$$\Phi(x) = A\{\sin(ax) - \sinh(ax) + k[\cos(ax) - \cosh(ax)]\} \tag{2.9}$$

Where k is a real constant:

$$\begin{aligned} k = &\{EIa[\sin(al) + \sinh(aL)] + \omega^2 I_p[\cos(aL) - \cosh(aL)]\} / \\ &\{\omega^2 I_p[\sin(aL) + \sinh(aL)] - EIa[\cos(aL) + \cosh(aL)]\} \end{aligned} \tag{2.9a}$$

The natural frequencies for the first 2 modes are listed in the table below. For comparison, the natural frequencies of the single mast are listed too:

tip mass tip inertia	natural frequency first mode	relatio to case $M_0 = I = 0$	natural frequency second mode	relatio to case $M_0 = I = 0$
$M_0 = 0$ $I = 0$	1.294 rad/s	100 %	9.022 rad/s	100 %
$M_0 = 274.5$ kg $I = 0$	0.905 rad/s	69.9 %	7.162 rad/s	79.4 %
$M_0 = 0$ $I = 2725.77$ kgm^2	1.291 rad/s	99.7 %	8.664 rad/s	96.0 %
$M_0 = 274.5$ kg $I = 2725.77$ kgm^2	0.904 rad/s	69.9 %	7.001 rad/s	77.6 %

The natural frequencies of the astro-mast are:

First mode: $\omega_1 = 0.904$ rad/sec $= 0.144$ Hz

Second mode: $\omega_2 = 7.001$ rad/sec $= 1.114$ Hz

The shape function of the second mode is depicted in fig. 2.4 , which shows the flexible cantilever beam, fixed on the left end. The dimensionless coordinate is $X = x/L$.

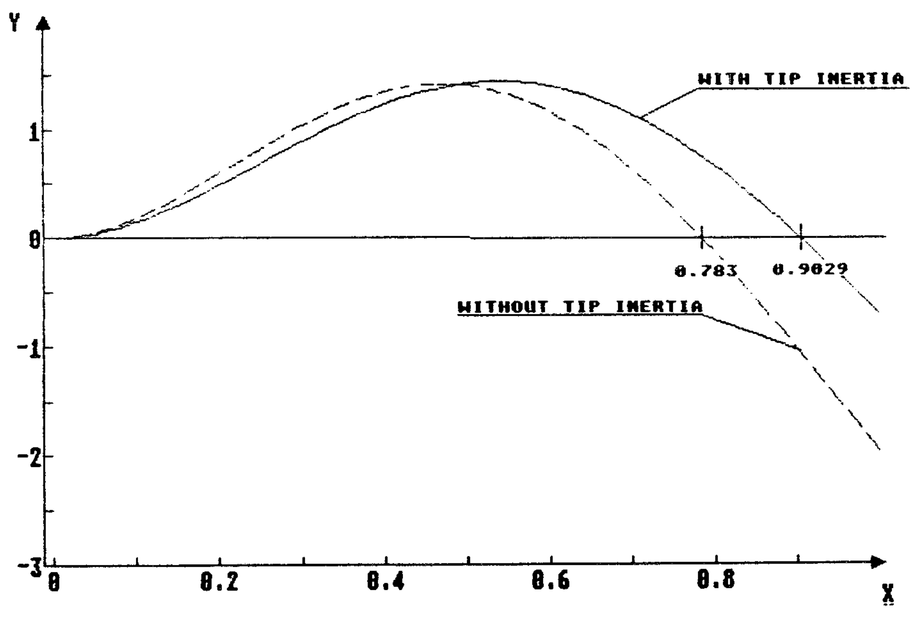


Fig. 2.4

A characteristic value of the second mode shape is the crossover point. The crossover point for the flexible mast is K = 0.9029 L .

	M_o = 0 I = 0 ω = 9.022 rad/s	M_o = 274.5 kg I = 2725.77 kgm^2 ω = 7.001 rad/s
crossoverpoint	0.783 L	0.9029 L

2.3 MAST DYNAMIC MODEL

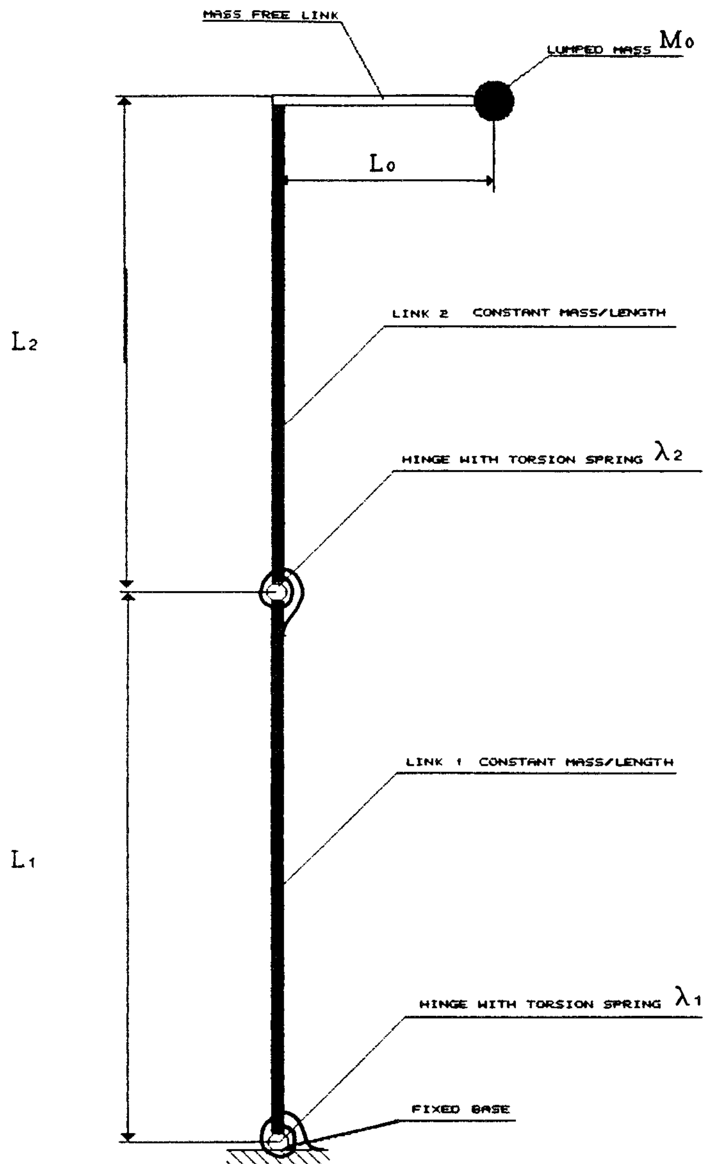


Fig. 2.5

As a dynamic model of the astro mast, we choose a discrete double pendulum, which is the most simply model, to describe 2 modes of vibration. The offset structure with payload platform is replaced by a single lumped mass, connected to the mast by a massless link. This configuration provides the additional mass and inertia at the top. The dynamic model is prescribed by 7 parameters:

2 mass parameters: Mass/length m of links L_1, L_2 and lumped mass M_0.

2 stiffness parameters: Stiffness of torsion springs λ_1, λ_2.

3 geometry parameters: Lengths of links L_1, L_2, L_0.

The parameters have to be evaluated in order to obtain the same dynamic characteristics in the first 2 modes of vibration, as the original mast. The length of the mast is given by:

$$L_1 + L_2 = L = 60.693 \text{ m} \tag{2.10}$$

Where we introduce the dimensionless length ratio:

$$R = (L_1 / L_2) \tag{2.11}$$

Both links of the dynamic mast model should have the same constant mass/length. We choose it as the algebratic middle of the masses from section 2.3 for the first and the second mode of the mast:

$$m = (m_1 + m_2)/2 = 15.931 \text{ kg/m} \tag{2.12}$$

The lumped mass M_0 and the length L_0 of the massless link are given by eqs. (2.1) & (2.5) :

$$M_0 = 274.58 \text{ kg}$$
$$L_0 = 3.150 \text{ m}$$

Up to now, 3 parameters of the dynamic model are not determined: R, λ_1, λ_2 . We will evaluate these parameters to obtain the natural frequencies ω_1 , ω_2 and the crossover point K. Applying the Lagrangian approach on the model, yields to the natural frequencies as a function of the unknown parameters.

MODES OF VIBRATION OF THE DYNAMIC MAST MODEL:

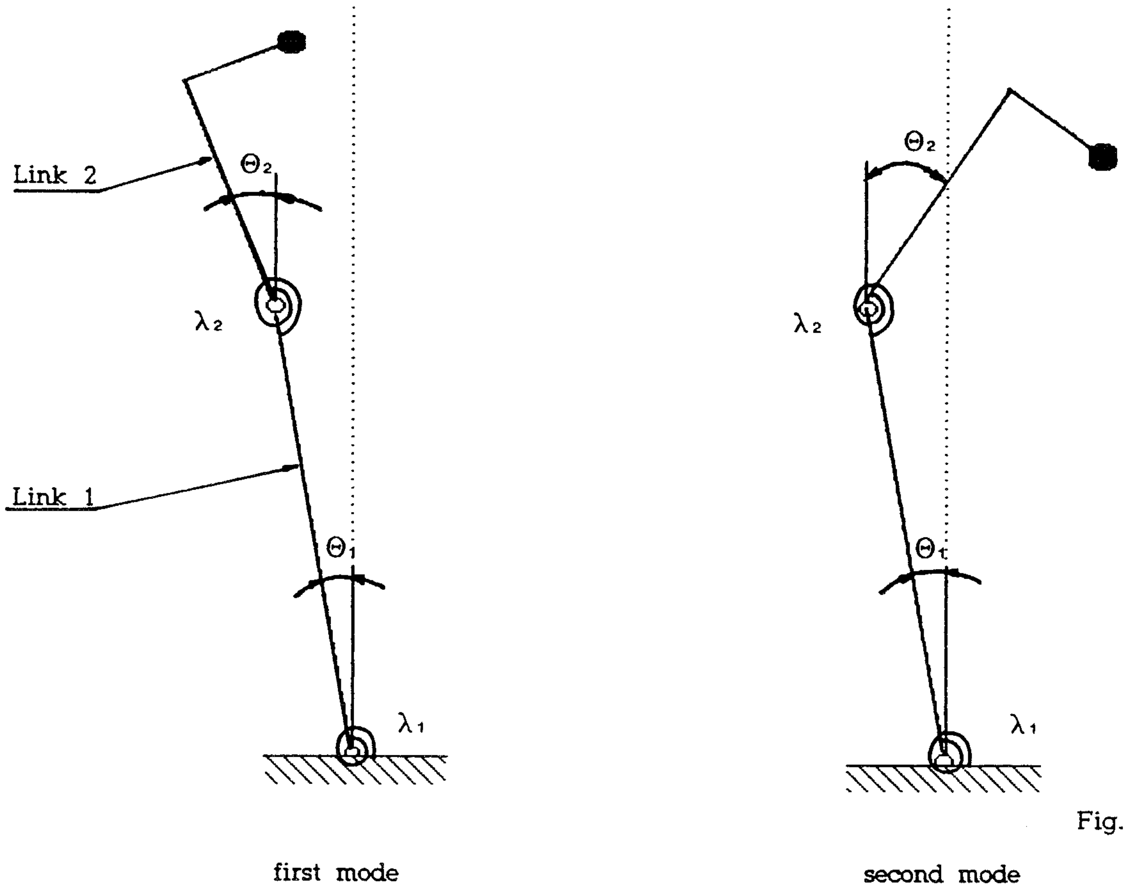


Fig. 2.6

Θ_1 and Θ_2 are the Lagrangian coordinates, which we assume as small. K denotes the crossover point.

Kinetic energies of dynamic mast model:

KINETIC ENERGY OF RIGID LINK 1:

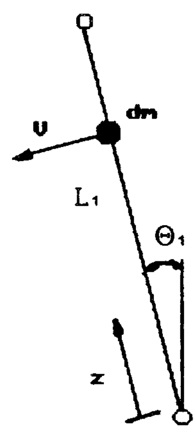


Fig. 2.7

The kinetic energy of a mass element on the link is :

$$dT_1 = 1/2\ V^2 dm \tag{2.13}$$

Where :

$$dm = mdZ \qquad m = \text{mass/length unit} \tag{2.13a}$$

For small angle Θ_1, the linearized equation for velocity is:

$$V = (\dot{\Theta}_1 Z)^2 \tag{2.14}$$

Integrating from $Z = 0$ to $Z = L_1$ yields the kinetic energy of link 1:

$$T_1 = \int_0^{L1} 1/2\ m \dot{\Theta}_1^2 Z^2 dZ \tag{2.15}$$

$$T_1 = 1/6\, m \dot{\Theta}_1^2 L_1^3$$

KINETIC ENERGY OF RIGID LINK 2 :

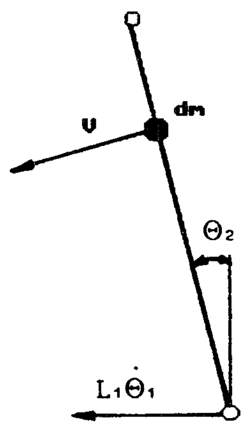


Fig. 2.8

Since we assume small angles Θ_1, Θ_2 , vertical velocities can be negelected. Thus, the velocity of a mass element on link 2 is:

$$V(Z) = \dot{\Theta}_1 L_1 + \dot{\Theta}_2 Z \tag{2.16}$$

The kinetic energy of link 2 is obtained by integrating from $Z = 0$ to $Z = L_2$:

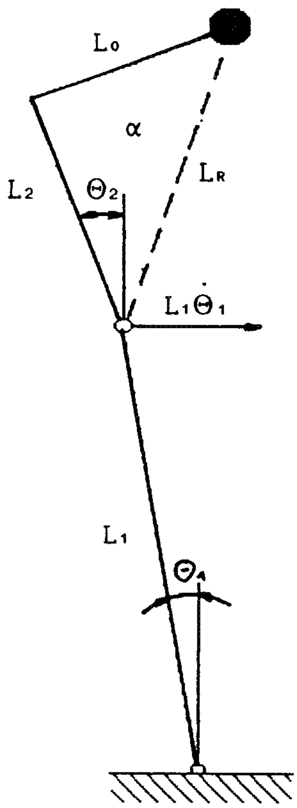
$$T_2 = 1/2\, m \int_0^{L2} V^2 dZ \tag{2.17}$$

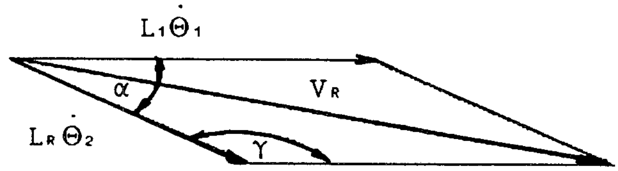
$$T_2 = 1/2\, m \int (\dot{\Theta}_1^2 L_1^2 + \dot{\Theta}_2^2 Z^2 + 2\dot{\Theta}_1\dot{\Theta}_2 L_1 Z)\, dZ$$

(2.17)

$$T_2 = 1/2\, m L_2^3\, [1/3\, \dot{\Theta}_1^2 + \dot{\Theta}_1\dot{\Theta}_2 (L_1/L_2) + \dot{\Theta}_1^2 (L_1/L_2)^2\,]$$

KINETIC ENERGY OF LUMPED MASS M_0:





$$\gamma = \pi - (\alpha + \Theta_2)$$

$$\cos(\gamma) = -\cos(\alpha + \Theta_2) \qquad (2.18)$$

Fig. 2.9

The resultant velocity of the lumped mass is:

$$V_r^2 = L_1^2\dot{\Theta}_1^2 + L_r^2\dot{\Theta}_2^2 + 2L_1L_r\dot{\Theta}_1\dot{\Theta}_2\cos(\alpha+\Theta_2) \qquad (2.19)$$

Since $\alpha >> \Theta_2$, we can neglect Θ_2 in the argument of the trigonometric function:

$$\cos(\alpha+\Theta_2) \approx \cos(\alpha) \qquad \text{for } \alpha >> \Theta_2 \qquad (2.20)$$

The length L_r is denoted by the theorem of Pythagoras:

$$L_r^2 = L_2^2 + L_0^2 \qquad (2.21)$$

Since $L_r \cos(\alpha) = L_2$:

$$V_r^2 = L_1^2\dot{\Theta}_1^2 + (L_2^2 + L_o^2)\dot{\Theta}_2^2 + 2L_1L_2\dot{\Theta}_1\dot{\Theta}_2 \qquad (2.22)$$

The corresponding kinetic energy is :

$$T_3 = 1/2\, M_o V_r^2 \qquad (2.23)$$

$$T_3 = 1/2 M_o L_2^2 \{(L_1/L_2)^2\dot{\Theta}_1^2 + (1+(L_o/L_2)^2)\dot{\Theta}_2^2 + 2(L_1/L_2)\dot{\Theta}_1\dot{\Theta}_2\}$$

TOTAL KINETIC ENERGY OF MAST DYNAMIC MODEL:

We replace L_1 and L_2 by the dimensionless length ratio R and the total length of the mast:

$$R = (L_1/L_2) \qquad L = L_1 + L_2$$

$$L_1 = RL/(R+1) \qquad L_2 = L/(R+1) \qquad (2.24)$$

Hence, the kinetic energies of the rigid links 1 and 2 and the lumped mass M_o are

$$T_1 = 1/6\, mL^3 [R/(R+1)]^3 \dot{\Theta}_1^2 \qquad (25.1)$$

$$T_2 = 1/2 mL^3[1/(R+1)]^3 [R^2\dot{\Theta}_1^2 + 1/3\,\dot{\Theta}_2^2 + R\dot{\Theta}_1\dot{\Theta}_2] \qquad (25.2)$$

$$T_3 = 1/2 M_o L^2 [1/(R+1)]^2 [R^2\dot{\Theta}_1^2 + (1+(L_o(R+1)/L)^2)\dot{\Theta}_2^2 + 2R\dot{\Theta}_1\dot{\Theta}_2] \qquad (25.3)$$

The total kinetic energy is the sum of all corresponding single kinetic energies:

$$T = T_1 + T_2 + T_3 \qquad (2.26)$$

Introducing eqs. (2.25.1) - (2.25.3) into eq. (2.26), and ordering to the Lagrangian coordinates yields :

$$\begin{aligned} T/mL^3 = & \left[R^3/6(R+1)^3 + R^2/2(R+1)^3 + (M_o/mL)(R^2/2(R+1)^2)\right]\dot{\Theta}_1^2 \qquad (2.27) \\ & + \left[R^3/6(R+1)^3 + (M_o/mL)(1+(L_o(R+1)/L)^2)\right]\dot{\Theta}_2^2 \\ & + \left[R/2(R+1)^3 + (M_o/mL)(R/(R+1)^2)\right]\dot{\Theta}_1\dot{\Theta}_2 \end{aligned}$$

Where, for convienience, we abbreviate the factors of the Lagrangian coordinates:

$$T/mL^3 = A\dot{\Theta}_1^2 + B\dot{\Theta}_2^2 + C\dot{\Theta}_1\dot{\Theta}_2 \qquad (2.27)$$

Where A, B, C are functions of the unknown length ratio R:

$$A = A(R) = R^3/6(R+1)^3 + R^2/2(R+1)^3 + (M_0/mL)(R^2/2(R+1)^2) \qquad (2.27.1)$$

$$B = B(R) = R^3/6(R+1)^3 + (M_0/mL)(1+(L_0(R+1)/L)^2) \qquad (2.27.2)$$

$$C = C(R) = R/2(R+1)^3 + (M_0/mL)(R/(R+1)^2) \qquad (2.27.3)$$

Strain energy mast dynamic model:

The strain energy is provided by 2 torsional springs:

$$V = 1/2\,\lambda_1\Theta_1^2 + 1/2\,\lambda_2(\Theta_2-\Theta_1)^2 \qquad (2.28)$$

We introduce normalized spring constants:

$$\Lambda_1 = \lambda_1/mL^3 \quad \text{and} \quad \Lambda_2 = \lambda_2/mL^3 \qquad (2.29)$$

Thus the total strain energy of the mast dynamic model is denoted by:

$$V/mL^3 = 1/2\,\Lambda_1\Theta_1^2 + 1/2\,\Lambda_2(\Theta_2-\Theta_1)^2 \qquad (2.30)$$

Differential equation of motion:

The system of Lagrange differential equations for linear, conservative systems is denoted by:

$$(d/dt)\,(\partial T/\partial\dot{\Theta}_i) + (\partial V/\partial\Theta_i) = 0 \qquad i = 1, 2 \ldots n \qquad (2.31)$$

The partial derviatives of the total kinetic energy T and the total strain energy V are:

$$(d/dt)(\partial T/\partial\dot{\Theta}_1) = 2A\ddot{\Theta}_1 + C\ddot{\Theta}_2 \qquad (2.32.1)$$

$$(d/dt)(\partial T/\partial\dot{\Theta}_2) = 2B\ddot{\Theta}_2 + C\ddot{\Theta}_1 \qquad (2.32.2)$$

$$(\partial V/\partial\Theta_1) = \Lambda_1\Theta_1 + \Lambda_2(\Theta_1 - \Theta_2) \qquad (2.33.1)$$

$$(\partial V/\partial\Theta_2) = \Lambda_2(\Theta_2 - \Theta_1) \qquad (2.33.2)$$

Introducing eqs. (2.32) - (2.33) into eq. (2.31) yields the system of second order differential equations:

$$2A\ddot{\Theta}_1 + C\ddot{\Theta}_2 + (\Lambda_1 + \Lambda_2)\Theta_1 - \Lambda_2\Theta_2 = 0$$
$$C\ddot{\Theta}_1 + 2B\ddot{\Theta}_2 - \Lambda_1\Theta_1 + \Lambda_2\Theta_2 = 0 \qquad (2.34)$$

Natural frequencies of mast dynamic model:

A solution of the system of second order differential equations is:

$$\Theta_1 = \bar{\Theta}_1\, e^{i\omega t} \qquad \Theta_2 = \bar{\Theta}_2\, e^{i\omega t} \qquad (2.35.1)$$

Where $\bar{\Theta}_1, \bar{\Theta}_2$ are amplitudes of displacement. Differentiating with respect to time gives:

$$\ddot{\Theta}_1 = -\bar{\Theta}_1\omega^2 e^{i\omega t} \qquad \ddot{\Theta}_2 = -\bar{\Theta}_2\omega^2 e^{i\omega t} \qquad (2.35.2)$$

We introduce eqs. (2.35) into the system of differential equations (2.34):

$$(2A\omega^2 - \Lambda_1 - \Lambda_2)\bar{\Theta}_1 + (C\omega^2 + \Lambda_2)\bar{\Theta}_2 = 0$$
$$(C\omega^2 + \Lambda_2)\bar{\Theta}_1 + (2B\omega^2 - \Lambda_2)\bar{\Theta}_2 = 0 \qquad (2.36)$$

$\bar{\Theta}_1, \bar{\Theta}_2$ can be eliminated. The natural frequencies ω_i are the roots of the fourth degree polynominal:

$$(4AB - C^2)\omega^4 - 2((A+B+C)\Lambda_2 + B\Lambda_1)\,\omega^2 + \Lambda_1\Lambda_2 = 0 \qquad (2.37)$$

Thus, the natural frequencies of the first and second mode are:

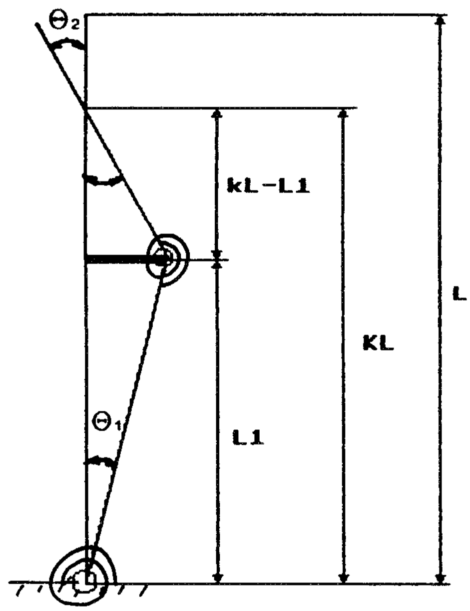
$$\omega^2_1 = (((A+B+C)\Lambda_2 + B\Lambda_1) - SQR(((A+B+C)\Lambda_2 + B\Lambda_1)^2 - (AB - C^2)\Lambda_1\Lambda_2)) / (AB - C^2) \qquad (2.38.1)$$

$$\omega^2_2 = (((A+B+C)\Lambda_2 + B\Lambda_1) + SQR(((A+B+C)\Lambda_2 + B\Lambda_1)^2 - (AB - C^2)\Lambda_1\Lambda_2)) / (AB - C^2) \qquad (2.38.2)$$

Geometry of second mode shape:

Until now, we have 2 equations to determine the 3 unknown: Λ_1, Λ_2, R. The third equation is provided by the geometry of the second mode shape, where we demand that the crossover point should be the same as of the original mast.

GEOMETRY SECOND MODE SHAPE:



$$-(kL - L_1)\Theta_1 = L_1\Theta_1 \qquad (2.39.1)$$

$$kL/L_1 - 1 = -\Theta_1/\Theta_2 \qquad (2.39.2)$$

Where k is the crossover point, evaluated in section 2.3 .

Fig. 2.10

The displacement ratio Θ_1/Θ_2 can be elininated:

$$\Theta_1/\Theta_2 = (C\omega_2^2 + \Lambda_2) / (-2A\omega_2^2 + \Lambda_1 + \Lambda_2) \qquad (2.40)$$

Hence:

$$(-2A\omega_2 + \Lambda_1 + \Lambda_2)(k\,(R+1)/R - 1) + C\omega_2 + \Lambda_2 = 0 \qquad (2.41)$$

Numerical results:

To evaluate the missing 3 parameters Λ_1, Λ_2, R of the dynamic mast model, we have to solve the system of 3 nonlinear equations:

$$(A+B+C)\Lambda_2 + B\Lambda_1 - SQR(((A+B+C)\Lambda_2 + B\Lambda_1)^2 - (AB - C^2)\Lambda_1\Lambda_2) - (AB - C^2)\omega_1^2 = 0 \qquad (2.42.1)$$

$$(A+B+C)\Lambda_2 + B\Lambda_1 + SQR(((A+B+C)\Lambda_2 + B\Lambda_1)^2 - (AB - C^2)\Lambda_1\Lambda_2) - (AB - C^2)\omega_2^2 = 0 \qquad (2.42.2)$$

$$(-2A\omega_2^2 + \Lambda_1 + \Lambda_2)(k\,(R+1)/R - 1) + C\omega_2^2 + \Lambda_2 = 0 \qquad (2.42.3)$$

where A, B, C are functions of R:

$$A = R^3/6(R+1)^3 + R^2/2(R+1)^3 + (M_o/mL)(R^2/2(R+1)^2) \qquad (2.27.1)$$

$$B = R^3/6(R+1)^3 + (M_o/mL)(1 + (L_o(R+1)/L)^2) \qquad (2.27.2)$$

$$C = R/2(R+1)^3 + (M_o/mL)(R/(R+1)^2) \qquad (2.27.3)$$

The roots are found numerically, using standard software. (Damped Newton iteration, using Jacoby matrix).

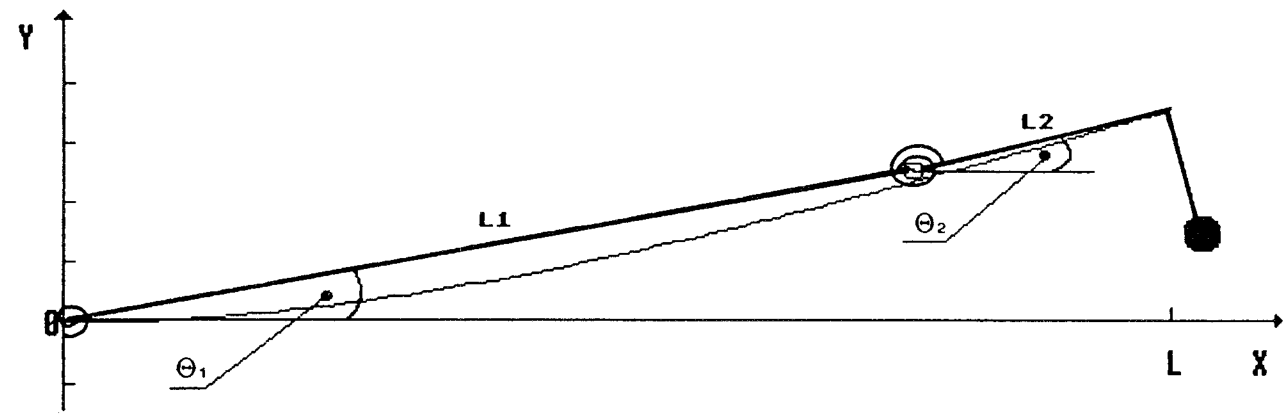
For the constants:

length of mast:	$L = 60.693$ m
length of link to tip mass:	$L_0 = 3.150$ m
mass/length of mast:	$m = 15.931$ kg/m
tip mass:	$M_0 = 274.58$ kg
natural frequencies:	$\omega_1 = 0.904$ rad/s
	$\omega_2 = 7.001$ rad/s
crossover point:	$k = 0.9029L$

We obtain the roots:

Torsional spring constants:	$\lambda_1 = 1.867\ 10^6$ Nm/rad
	$\lambda_2 = 1.150\ 10^6$ Nm/rad
Relatio L_1/L_2:	$R = 3.237$

Which defines the mast dynamic model completely.



First mode shape of flexible cantilever beam and equivalent double pendulum.

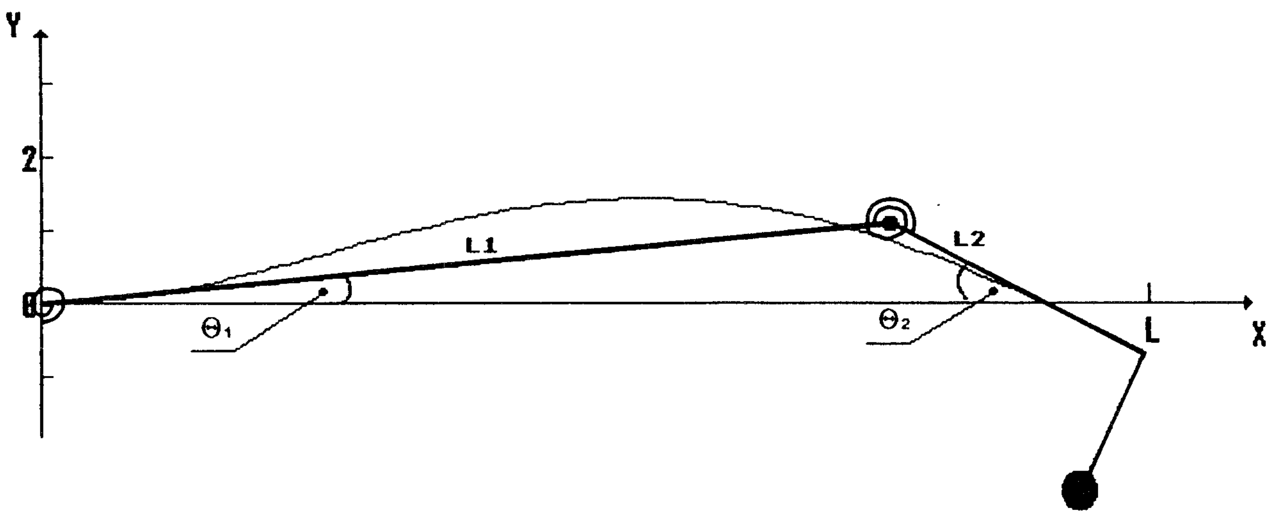


Fig. 2.11

Second mode shape of flexible cantilever beam and equivalent double pendulum.

2.4 ANTENNA DYNAMIC MODEL:

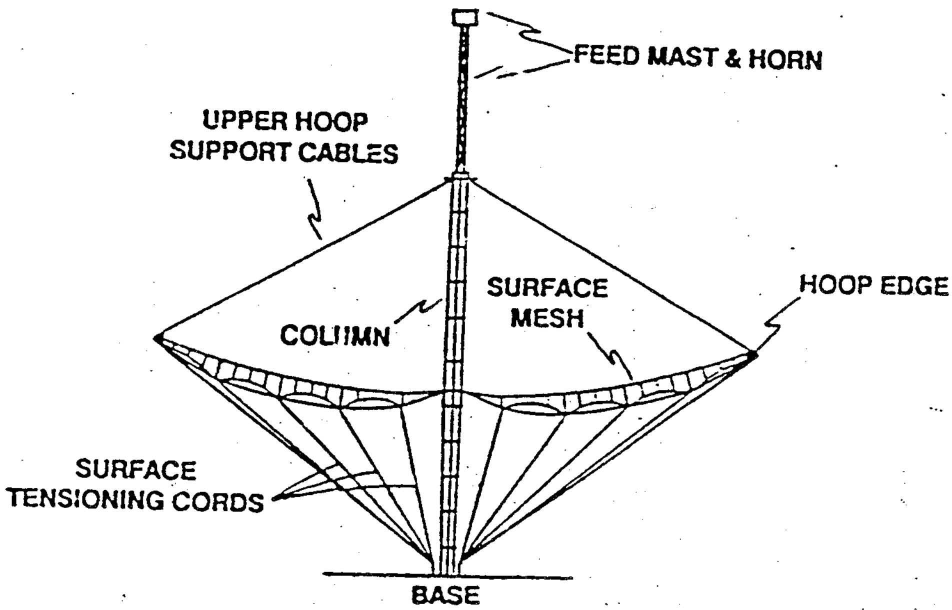


Figure 6-2. Diametrical cross section view of hoop — column antenna.

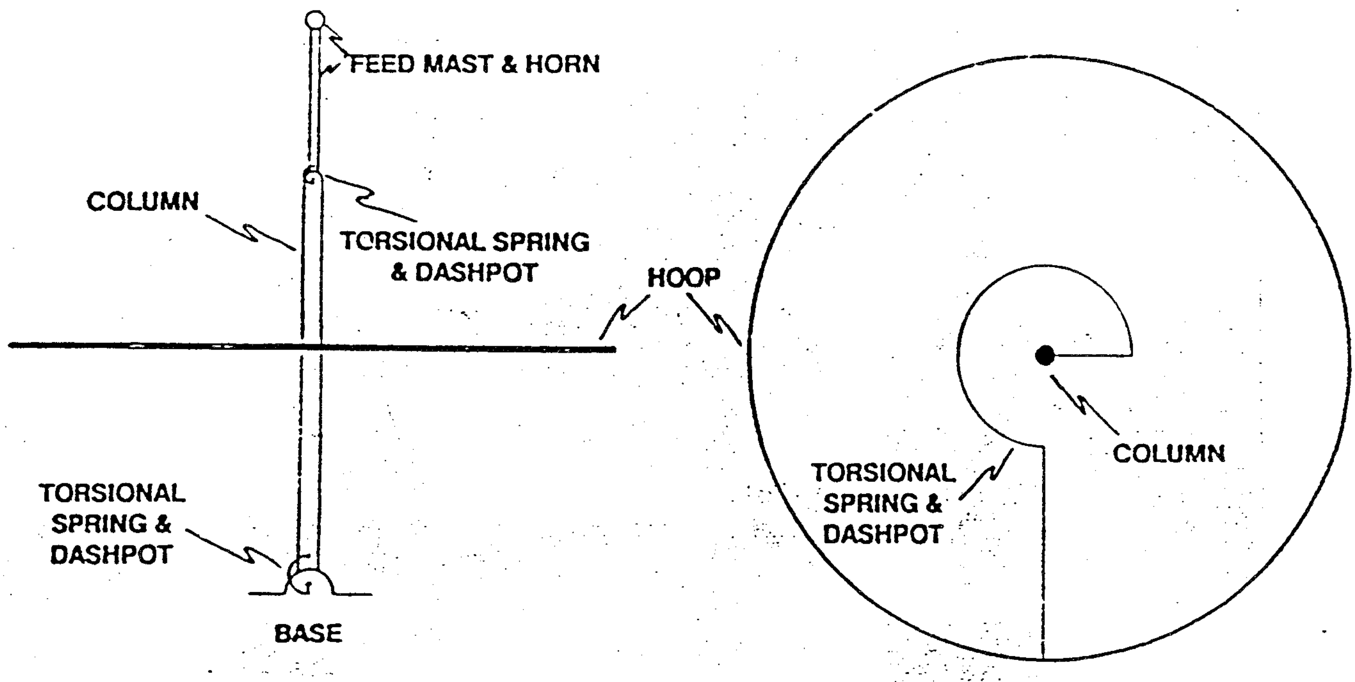


Figure 6-4. Three rigid-body antenna idealization, side and top views.

The Charles Stark Draper Laboratory (ref. 1) idealizes the antenna as three-rigid body system (Fig. 6-4), where two bending modes and one torsional mode is considered. It is clear, that the torsional mode has no meaning for the vibrations of the pure pitch maneuver. The model for the bending modes is a discrete double pendulum for the first 2 modes of vibration. For simplification, we consider only the first mode, which can be modelled by a single pendulum:

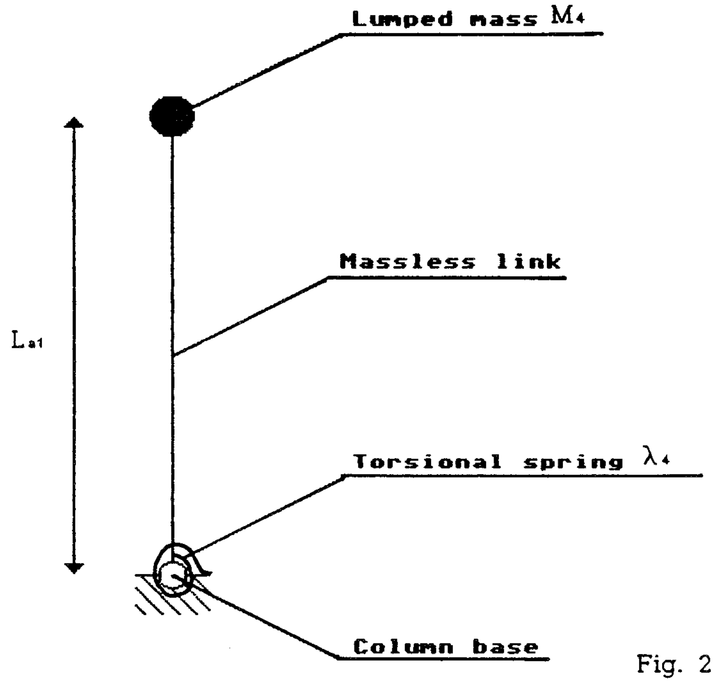


Fig. 2.13

We choose λ_4 equal to the stiffness of the column/base spring k_1, and the lumped mass M_4 equal to the total mass of the antenna (column + feed mast + hoop) :

$$\lambda_4 = k_1 = 7.144\ 10^4 \text{ Nm/rad} \qquad (2.28)$$

$$M_4 = m_c + m_f + m_h \qquad (2.29)$$

$$M_4 = 126.951\,\text{kg} + 117.234\,\text{kg} + 118.337\,\text{kg}$$

$$M_4 = 362.522\ \text{kg}$$

The length L_{a1} of the massless link is deteremined, in order to obtain the same natural frequency of the first bending mode. The natural frequency of a single pendulum is denoted by:

$$\omega_0 = \lambda / m L^2 \qquad (2.30)$$

For ω_0 = 1.508 rad/sec, we obtain the length of the massless rigid link:

$$L_{a1} = SQR(\lambda_4/M_4\,\omega_0) \tag{2.31}$$

$$L_{a1} = 4.246 \text{ m}$$

The column base is connected to the pitch gimbal hinge by a beam, which has the length L_{23} = 5.22 m and moment of inertia I_p = 1620 kgm^2 . We replace this beam by a massless link with tip mass M_3. The tip mass provides the same moment of inertia.

Hence the dynamic model for the system of antenna and beam is defined by a discrete double pendulum:

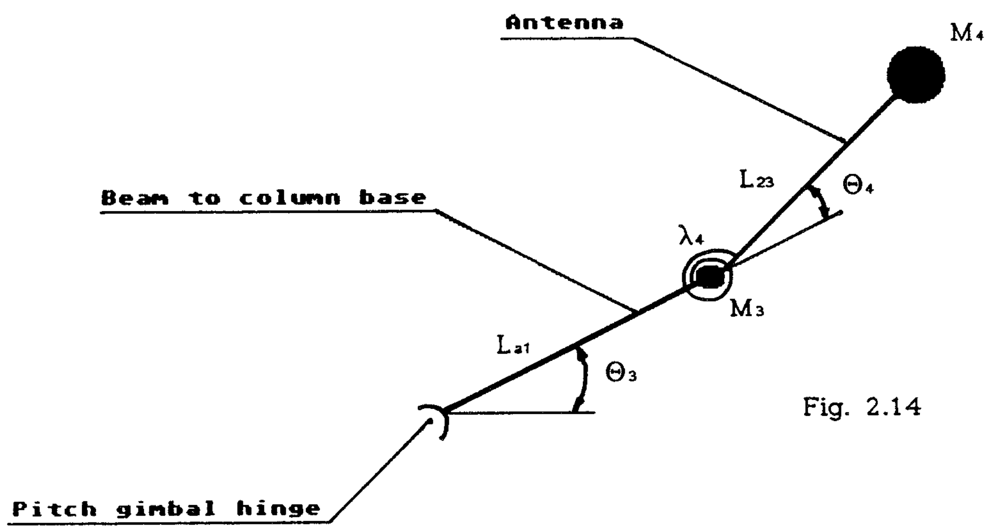


Fig. 2.14

L_{23} = 5.220 m

L_{a1} = 11.432 m

M_3 = 59.453 kg

M_4 = 362.522 kg

λ_4 = 7.144 10^4 Nm/rad

3. PITCH MANEUVER

3.1 INTRODUCTION

Having defined a dynamic model for the first 4 modes of vibration of the astro-mast antenna system, now we use this dynamic model, to investigate the vibrational characteristics. First, we are interested in the vibrations, which are caused by a pitch maneuver of the antenna. The pitch maneuver is decribed by the pitch angle α.

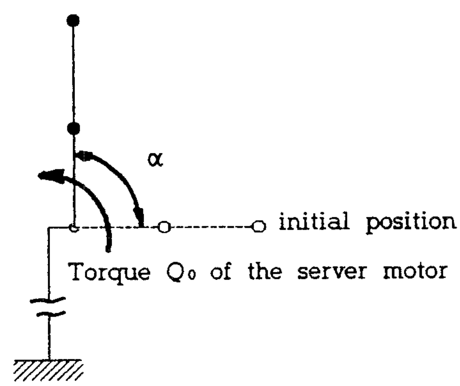


Fig. 3.1

The maneuver is achieved by a step functional torque, applied by the d.c. servo motor at the gimbal. We assume constant acceleration and constant deceleration. Thus the time dependent torque is a step function. In the report of the Charles Draper Laboratory (see ref. 1), page 82, the maximum available servo motor torque is 25 foot-pounds = 33.89 Nm, thus we assume a constant torque of +/- 30 Nm.

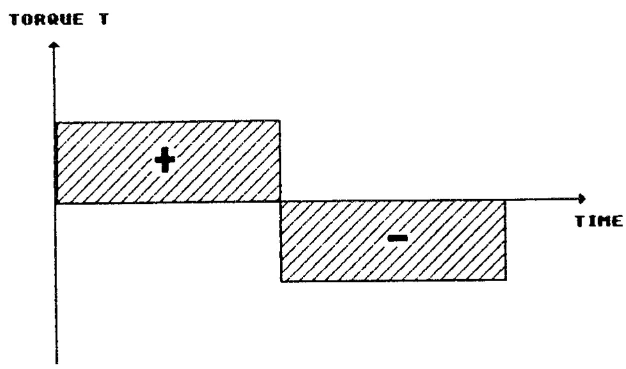


Fig. 3.2

Also, we assume zero vibrations initially. In practice, this means, all vibrations caused by previous maneuvers are damped out, so far to have negligible magnitudes.

The motion of the antenna model is described by the time history of the generalized angles Θ_i, used to model the system as shown in the previous section. We apply the Lagrangian approach to obtain the equation of motion. The approach is as follows:

1. *Determine the kinetic and the potential energies as a function of generalized angles:* Θ_1 , Θ_2 , Θ_3 , Θ_4 .
2. *Apply the Lagrangian equation to obtain the differential equations of motion.*
3. *Solve the equations of motion under step function servo motor excitation, causing angular acceleration of the antenna.*

The differential equation of motion will be valid for arbitrary pitch angles α. As an example, we will solve the differential equations for a pitch maneuver from $0 \le \alpha \le 90^\circ$.

3.2 DETERMINATION OF THE KINETIC AND POTENTIAL ENERGIES

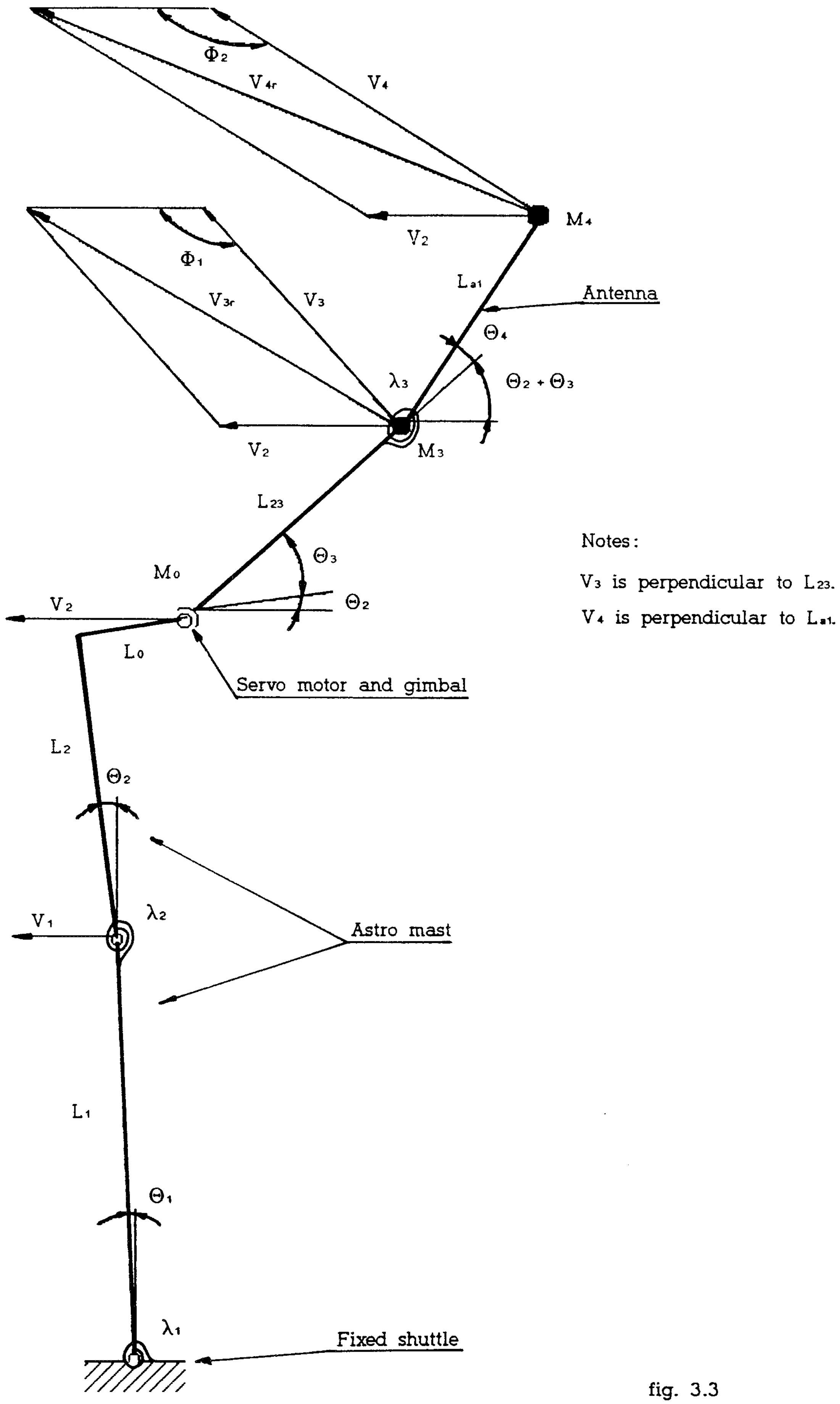


fig. 3.3

VELOCITIES V_i AND GENERALIZED ANGLES Θ_i OF THE DYNAMIC MODEL

The system can be divided in 2 parts: The astro-mast and the antenna. Both are linked together by the servo motor gimbal pivot. In section 2, we have already determined the equations of motion of the astro-mast dynamic model. Now we investigate the dynamics of the antenna, to obtain finally the vibrational response of the complete system, caused by the servomotor torque. As in section 2.4 defined, the antenna dynamic model consists of two massless links L_{23}, L_{41} and two lumped masses M_3, M_4.

The motion of the antenna dynamic model is described by the generalized angles Θ_3 and Θ_4. Θ_4 can be assumed to be small, but Θ_3 is a large angle, because it is the sum of the pitch angle α and vibrational displacements, as shown in fig. 3.5:

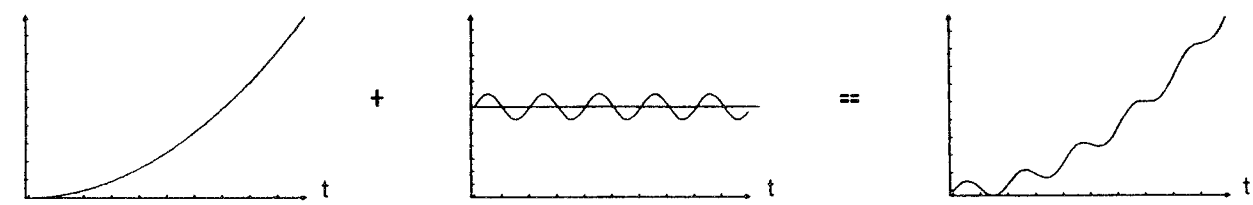


fig. 3.4

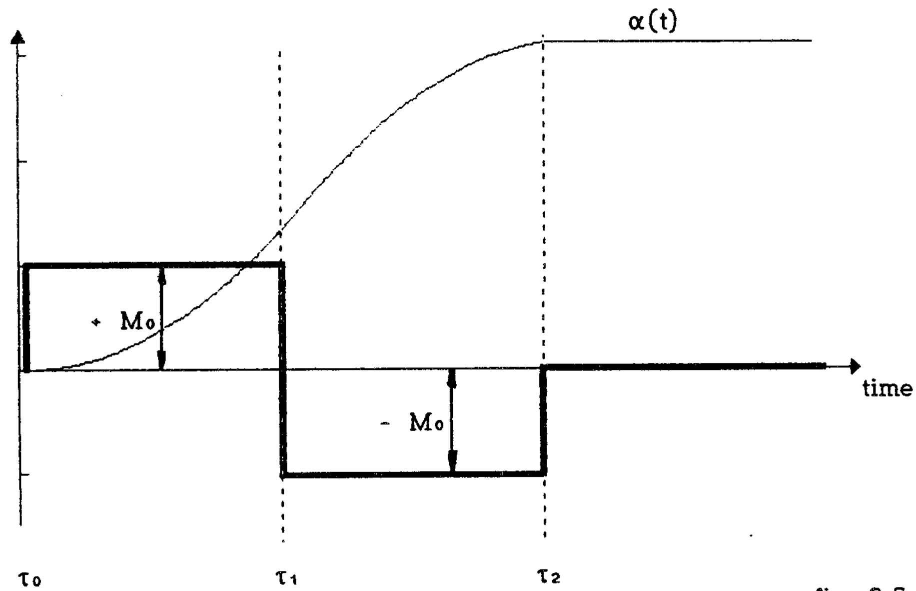


fig. 3.5

For constant torque M_0, the required pitch angle α is defined by a quadratic function:

$$\alpha(t) = \alpha_0 + 1/2\, M_0 t^2 / I_A \qquad \tau_0 <= t <= \tau_1 \qquad \text{Accel.}$$

$$\alpha(t) = \alpha_0 + M_0 / I_A \,(\tau_1^2 - 2\tau_1 t + 1/2\, t^2) \qquad \tau_1 < t <= \tau_2 \qquad \text{Decel.} \qquad (3.1)$$

Where α_0 is the initial pitch angle $\alpha_0 = 0$.

Where M_0 is the constant torque of the servo motor and I_A is the moment of inertia of the antenna system. The motion is a function of time and we define 3 discrete time values:

τ_0 : dish in initial position, start of acceleration.

τ_1 : end of acceleration, start of decceleration.

τ_2 : end of deceleration, dish in final position.

Since Θ_3 is a large angle rotation, we are not allowed to use linearized equations. Thus, the absolute velocities of the the lumped masses M_3 and M_4 are obtained using the cosine rule (see fig. 3.3), as:

$$V_{3r}^2 = V_2^2 + V_3^2 - 2\,V_2 V_3 \cos(\Phi_1) \tag{3.2}$$

$$V_{4r}^2 = V_2^2 + V_4^2 - 2\,V_2 V_4 \cos(\Phi_2) \tag{3.4}$$

V_2 is the horizontal velocity of the gimbal, which we have determined in section 2 as:

$$V_2 = \dot{\Theta}_1 L_1 + \dot{\Theta}_2 L_2 \tag{3.5}$$

The vertical velocity of the gimbal was found to be negligible.
The relative velocities V_3 and V_4 are:

$$V_3 = (\dot{\Theta}_2 + \dot{\Theta}_3)\,L_{23} \tag{3.6}$$

$$V_4 = V_3 + (\dot{\Theta}_2 + \dot{\Theta}_3 + \dot{\Theta}_4)\,L_{a1} = (\dot{\Theta}_2 + \dot{\Theta}_3)L_{23} + (\dot{\Theta}_2 + \dot{\Theta}_3 + \dot{\Theta}_4)L_{a1} \tag{3.7}$$

We eliminate the angles Φ_1 and Φ_2 by using trigonometric relations:

$$\Phi_1 = \pi/2 + \Theta_2 + \Theta_3 \qquad \cos(\Phi_1) = -\sin(\Theta_2 + \Theta_3) \tag{3.8}$$

$$\Phi_2 = \pi/2 + \Theta_2 + \Theta_3 + \Theta_4 \qquad \cos(\Phi_2) = -\sin(\Theta_2 + \Theta_3 + \Theta_4) \tag{3.9}$$

We introduce eqs. (3.5) - (3.9) into eqs. (3.2) and (3.4) to obtain the absolute velocities of the masses M_3, M_4, as a function of the generalized angles Θ_i:

$$V_{3r}^2 = (\dot{\Theta}_1 L_1 + \dot{\Theta}_2 L_2)^2 + (\dot{\Theta}_2 + \dot{\Theta}_3)^2 L_{23}^2 + 2\,(\dot{\Theta}_1 L_1 + \dot{\Theta}_2 L_2)(\dot{\Theta}_2 + \dot{\Theta}_3)\,L_{23} \sin(\Theta_2 + \Theta_3) \tag{3.2.a}$$

$$V_{4r}^2 = (\dot{\Theta}_1 L_1 + \dot{\Theta}_2 L_2)^2 + \{(\dot{\Theta}_2 + \dot{\Theta}_3)L_{23} + (\dot{\Theta}_2 + \dot{\Theta}_3 + \dot{\Theta}_4)L_{a1}\}^2 + \tag{3.4.a}$$
$$+ 2(\dot{\Theta}_1 L_1 + \dot{\Theta}_2 L_2)\{(\dot{\Theta}_2 + \dot{\Theta}_3)L_{23} + (\dot{\Theta}_2 + \dot{\Theta}_3 + \dot{\Theta}_4)L_{a1}\}\sin(\Theta_2 + \Theta_3 + \Theta_4)$$

For convenience, we arrange these equations in the following way:

$$V_{3r}^2 = L_1^2 \dot{\Theta}_1^2 + \tag{3.2.b}$$
$$+ \{L_2^2 + L_{23}^2 + 2L_2L_{23}\sin(\Theta_2 + \Theta_3)\}\dot{\Theta}_2^2 +$$
$$+ L_{23}^2 \dot{\Theta}_3^2 +$$
$$+ \{2L_1L_2 + 2L_1L_{23}\sin(\Theta_2 + \Theta_3)\}\dot{\Theta}_1\dot{\Theta}_2 +$$
$$+ \{2L_1L_{23}\sin(\Theta_2 + \Theta_3)\}\dot{\Theta}_1\dot{\Theta}_3 +$$
$$+ \{2L_{23}^2 + 2L_2L_{23}\sin(\Theta_2 + \Theta_3)\}\dot{\Theta}_2\dot{\Theta}_3$$

$$V_{4r}^2 = L_1^2 \dot{\Theta}_1^2 + \tag{3.4.b}$$
$$+ \{L_2^2 + L_{23}^2 + L_{a1}^2 + 2L_{23}L_{a1} + 2(L_2L_{23} + L_2L_{a1})\sin(\Theta_2 + \Theta_3 + \Theta_4)\}\dot{\Theta}_2^2 +$$
$$+ \{L_{23}^2 + L_{a1}^2 + 2L_{23}L_{a1}\}\dot{\Theta}_3^2 +$$
$$+ L_{a1}^2 \dot{\Theta}_4^2 +$$
$$+ \{2L_1L_2 + 2(L_1L_{23} + L_1L_{a1})\sin(\Theta_2 + \Theta_3 + \Theta_4)\}\dot{\Theta}_1\dot{\Theta}_2 +$$
$$+ \{2(L_1L_{23} + L_1L_{a1})\sin(\Theta_2 + \Theta_3 + \Theta_4)\}\dot{\Theta}_1\dot{\Theta}_3 +$$
$$+ \{2L_1L_{a1}\sin(\Theta_2 + \Theta_3 + \Theta_4)\}\dot{\Theta}_1\dot{\Theta}_4 +$$
$$+ \{2L_{23}^2 + 2L_{a1}^2 + 4L_{23}L_{a1} + (L_2L_{23} + L_2L_{a1})\sin(\Theta_2 + \Theta_3 + \Theta_4)\}\dot{\Theta}_2\dot{\Theta}_3 +$$
$$+ \{2L_{a1}^2 + 2L_{23}L_{a1} + 2L_2L_{a1}\sin(\Theta_2 + \Theta_3 + \Theta_4)\}\dot{\Theta}_2\dot{\Theta}_4 +$$
$$+ \{2L_{a1}^2 + 2L_{23}L_{a1}\}\dot{\Theta}_3\dot{\Theta}_4$$

The total kinetic energy is the sum of all the compounds:

$$T = T_1 + T_2 + T_3 + T_4 + T_5 \tag{3.10}$$

where:

T_1 = kinetic energy of rigid link 1

T_2 = kinetic energy of rigid link 2

T_3 = kinetic energy of lumped gymbal mass M_0

T_4 = kinetic energy of lumped mass M_3

T_5 = kinetic energy of lumped mass M_4

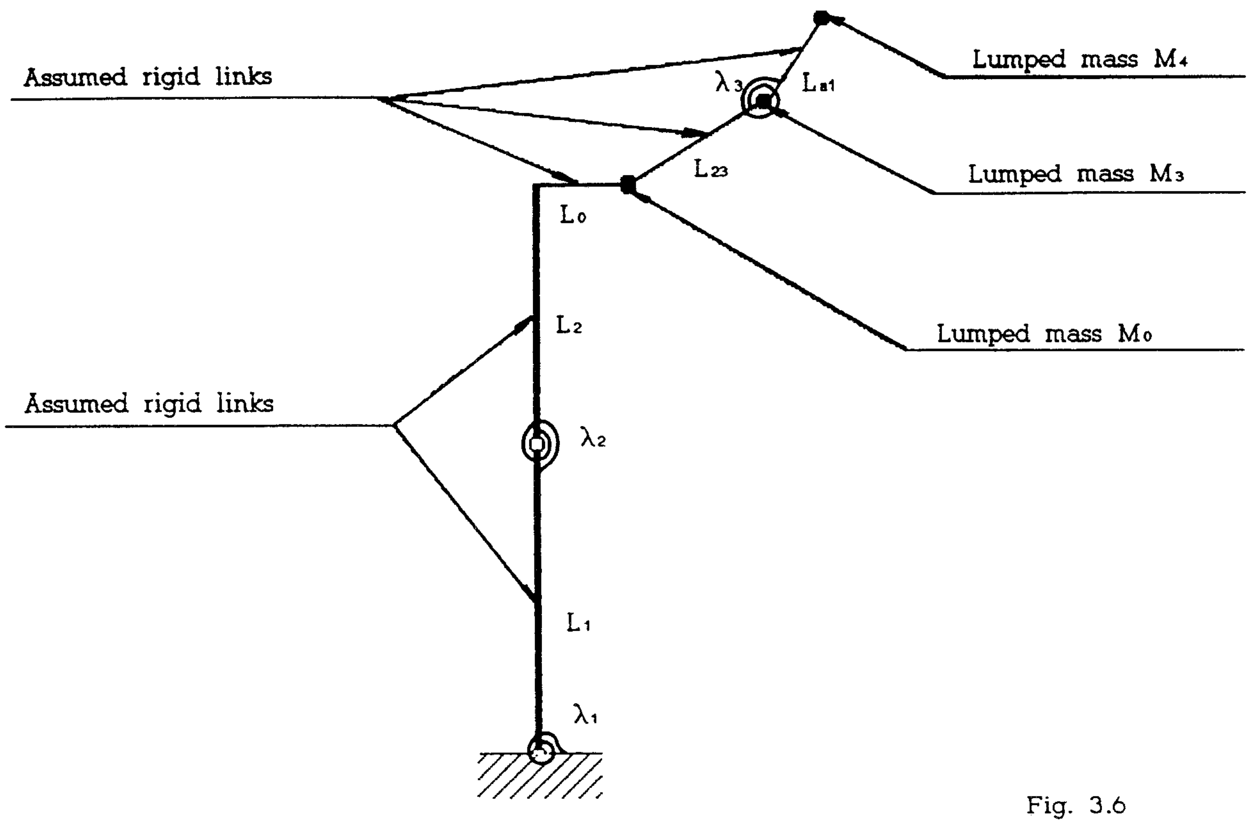


Fig. 3.6

T_1, T_2, T_3 are the kinetic energies of the astro-mast dynamic model, which we have determined in section 2 (see eqs. (2.25.1) - (2.25.2)) :

$$(T_1 + T_2 + T_3)/mL^3 = mL\left[R^3/6(R+1)^3 + R^2/2(R+1)^3 + (M_o/mL)(R^2/2(R+1)^2)\right]\dot{\Theta}_1^2 \quad (3.11)$$
$$+ mL^3\left[R^3/6(R+1)^3 + (M_o/mL)(1+(L_o(R+1)/L)^2)\right]\dot{\Theta}_2^2$$
$$+ mL^3\left[R/2(R+1)^3 + (M_o/mL)(R/(R+1)^2)\right]\dot{\Theta}_1\dot{\Theta}_2$$

The kinetic energy of the antenna dynamic model is the sum of the kinetic energies of the lumped masses M_3, M_4. The kinetic energies of the lumped masses M_3 and M_4 are:

$$T_4 = 1/2\, M_3 V_{3r}^2 \quad (3.12)$$
$$T_5 = 1/2\, M_4 V_{4r}^2 \quad (3.13)$$

We introduce eqs. (3.2.b), (3.4.b) and eqs. (3.11) - (3.13) into eq. (3.10) to obtain the total kinetic energy of the complete dynamic model. As in section 2 we use the factor $1/mL^3$ to normalize the kinetic energy.

Total kinetic energy of the astro-mast antenna dynamic model:

$$
\begin{aligned}
T/mL^3 = \; & C_1\dot{\Theta}_1^2 + \\
& + \{C_2 + C_3\sin(\Theta_2+\Theta_3) + C_4\sin(\Theta_2+\Theta_3+\Theta_4)\}\,\dot{\Theta}_2^2 + \\
& + C_5\,\dot{\Theta}_3^2 + \\
& + C_6\,\dot{\Theta}_4^2 + \\
& + \{C_7 + C_8\sin(\Theta_2+\Theta_3) + C_9\sin(\Theta_2+\Theta_3+\Theta_4)\}\,\dot{\Theta}_1\dot{\Theta}_2 + \\
& + \{C_{10}\sin(\Theta_2+\Theta_3) + C_{11}\sin(\Theta_2+\Theta_3+\Theta_4)\}\,\dot{\Theta}_1\dot{\Theta}_3 + \\
& + \{C_{12}\sin(\Theta_2+\Theta_3+\Theta_4)\}\,\dot{\Theta}_1\dot{\Theta}_4 + \\
& + \{C_{13} + C_{14}\sin(\Theta_2+\Theta_3) + C_{15}\sin(\Theta_2+\Theta_3+\Theta_4)\}\,\dot{\Theta}_2\dot{\Theta}_3 + \\
& + \{C_{16} + C_{17}\sin(\Theta_2+\Theta_3+\Theta_4)\}\,\dot{\Theta}_2\dot{\Theta}_4 + \\
& + C_{18}\,\dot{\Theta}_3\dot{\Theta}_4
\end{aligned}
\tag{3.14}
$$

C_1, C_2, . . . , C_{18} are real constants, which are defined by geometry and mass parameters of the model (See section 2, definition of dynamic model) :

L =	total length of astro-mast. [m]
L_1 =	length of first link of mast. [m]
L_2 =	length of second link of mast. [m]
L_0 =	length of horizontal link to gimbal mass M_0. [m]
L_{23} =	length of first link of antenna system. [m]
L_{a1} =	length of second link of antenna system. [m]
R =	dimensionless length ratio L_1/L_2. [−]

m =	mass / length of links L_1, L_2 . [kg/m]
M_0 =	lumped mass of gimbal. [kg]
M_3 =	first lumped mass of antenna system. [kg]
M_4 =	second lumped mass of antenna system. [kg]

$$C_1 = R^3/6(R+1)^3 + R^2/2(R+1)^3 + M_0R^2/2mL(R+1)^2 + M_3L_1^2/2mL^3 + M_4L_1^2/2mL^3 \tag{3.15.1}$$

$$C_2 = R^3/6(R+1)^3 + (M_0/mL)\{1+((L_0/L)(R+1))^2\} + M_3(L_2^2+L_{23}^2)/2mL^3 + M_4(L_2^2+L_{23}^2+L_{a1}^2+2L_{23}L_{a1})/2mL^3 \tag{3.15.2}$$

$$C_3 = M_3 L_2 L_{23} / mL^3 \tag{3.15.3}$$

$$C_4 = M_4(L_2 L_{23} + L_2 L_{a1}) / mL^3 \tag{3.15.4}$$

$$C_5 = M_3 L_{23}^2 / 2mL^3 + M_4(L_{23}^2 + L_{a1}^2 + 2 L_{23} L_{a1}) / 2mL^3 \tag{3.15.5}$$

$$C_6 = M_4 L_{a1}^2 / 2mL^3 \tag{3.15.6}$$

$$C_7 = R / 2(R+1)^3 + M_0 R / mL(R+1)^2 + M_3 L_1 L_2 / mL^3 + M_4 L_1 L_2 / mL^3 \tag{3.15.7}$$

$$C_8 = M_3 L_1 L_{23} / mL^3 \tag{3.15.8}$$

$$C_9 = M_4(L_1 L_{23} + L_1 L_{a1}) / mL^3 \tag{3.15.9}$$

$$C_{10} = M_3 L_1 L_{23} / mL^3 \tag{3.15.10}$$

$$C_{11} = M_4(L_1 L_{23} + L_1 L_{a1}) / mL^3 \tag{3.15.11}$$

$$C_{12} = M_4 L_1 L_{a1} / mL^3 \tag{3.15.12}$$

$$C_{13} = M_3 L_{23} / mL^3 + M_4(L_{23}^2 + L_{a1}^2 + 2 L_{23} L_{a1}) / mL^3 \tag{3.15.13}$$

$$C_{14} = M_3 L_2 L_{23} / mL^3 \tag{3.15.14}$$

$$C_{15} = M_4(L_2 L_{23} + L_2 L_{a1}) / mL^3 \tag{3.15.15}$$

$$C_{16} = M_4(L_{a1}^2 + L_{23} L_{a1}) / mL^3 \tag{3.15.16}$$

$$C_{17} = M_4 L_2 L_{a1} / mL^3 \tag{3.15.17}$$

$$C_{18} = C_{16} \tag{3.15.18}$$

Total strain energy of dynamic model:

The total strain energy is the sum of all strain energies, provided by 3 torsional springs. The torsional spring stiffness constants are λ_1, λ_2, λ_4 [Nm/rad]. Similar to the kinetic energy, we also normalize the strain energy, using the factor $1/mL^3$. Hence we use the normalized torsional spring stiffness constants:

$$\Lambda_i = \lambda_i/mL^3 \qquad i = 1, 2, 4 \qquad [1/sec^2] \tag{3.16}$$

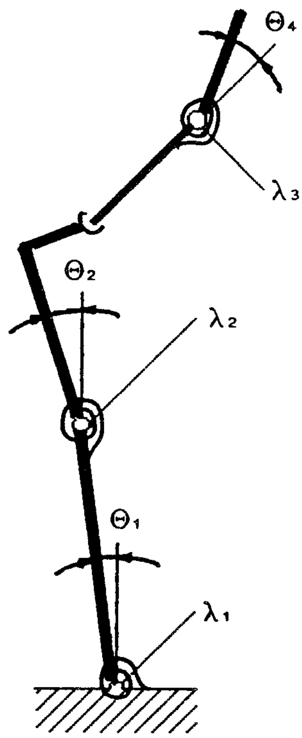


Fig. 3.7

Total strain energy:

$$V = V_1 + V_2 + V_3 \tag{3.17}$$

$$V/mL^3 = 1/2\ \Lambda_1\Theta_1^2 + 1/2\ \Lambda_2(\Theta_2 - \Theta_1)^2 + 1/2\ \Lambda_4\Theta_4^2 \tag{3.18}$$

The generalized exciting forces

We have to use generalized exciting forces for the Lagrange approach. The generalized exciting forces are defined as follows:

1. Give each generalized coordinate (Θ_1, Θ_2, Θ_3, Θ_4) an independent virtual displacement $\delta\Theta_i$.

2. Determine the associated virtual work δW_e, done by the the external forces F_i.

3. Derive the virtual work with respect to the generalized coordinates, to obtain the generalized exciting force:

$$Q_i = \partial(\delta W_e) / \partial(\delta\Theta_i) \qquad i = 1, 2, 3, 4 \qquad (3.19)$$

The external exciting force is the torque M_0 of the servo motor, applied at the gimbal pivot. Due to Newtons law: actio = reactio , we have to apply a positive external torque $+M_0$ at the link L_{23} and a negative external torque $-M_0$ at the link L_2.

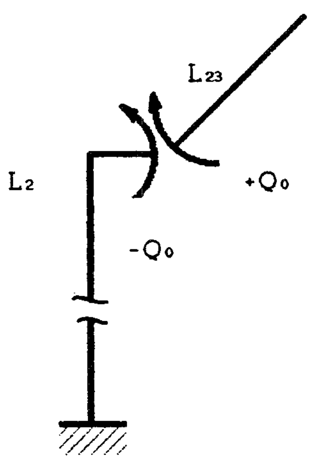


fig. 3.8

The virtual work, done by the external torques $+M_0$ and $-M_0$, is :

$$\delta W_e = -M_0\,\delta\Theta_2 + M_0\,\delta\Theta_3 \qquad (3.20)$$

We obtain the generalized exciting forces by derivating the virtual work:

$$
\begin{aligned}
Q_1 &= \partial(\delta W_e)/\partial(\delta\Theta_1) = 0 \\
Q_2 &= \partial(\delta W_e)/\partial(\delta\Theta_2) = -M_0 \\
Q_3 &= \partial(\delta W_e)/\partial(\delta\Theta_3) = +M_0 \\
Q_4 &= \partial(\delta W_e)/\partial(\delta\Theta_4) = 0
\end{aligned}
\qquad (3.21)
$$

The generalized damping forces

Since passive damping is unknown, we neglect it in the analysis, counting active damping to dissipative vibrational energy. Any passive damping present, will then be a bonus in the system in increasing the decay rate of the vibration. Also modal damping cannot be used in the persent formulation and actual values are unknown.

3.3 DIFFERENTIAL EQUATION OF MOTION

We apply the Lagrange differential equation to the 4 degree of freedom system.
The Lagrange equations for nonlinear systems are:

$$d/dt(\partial T/\partial\dot{\Theta}_i) - \partial T/\partial\Theta_i + \partial V/\partial\Theta_i + \partial F/\partial\dot{\Theta}_i = Qi \qquad i = 1, 2, ..n$$

Where:

$\boldsymbol{\Theta} = (\Theta_1, \Theta_2, \ldots, \Theta_n)^T$ = Generalized coordinates = Lagrangian coordinates.

$T = T(\boldsymbol{\Theta}, \dot{\boldsymbol{\Theta}})$ = Total kinetic energy.

$V = V(\boldsymbol{\Theta})$ = Total strain energy.

$F = F(\boldsymbol{\Theta}, \dot{\boldsymbol{\Theta}})$ = Dissipative function for passive structural damping.

$Q_i = Q_i(t)$ = Generalized exciting forces.

Since we neglect structural damping, the dissipative function F is equal to zero. The exciting forces are given by eq. (3.21). The differential equation system for the astro-mast antenna dynamic model is :

$$\begin{aligned}
d/dt(\partial T/\partial \dot{\Theta}_1) &- \partial T/\partial \Theta_1 + \partial V/\partial \Theta_1 = 0 \\
d/dt(\partial T/\partial \dot{\Theta}_2) &- \partial T/\partial \Theta_2 + \partial V/\partial \Theta_2 = -M_0 \\
d/dt(\partial T/\partial \dot{\Theta}_3) &- \partial T/\partial \Theta_3 + \partial V/\partial \Theta_3 = +M_0 \\
d/dt(\partial T/\partial \dot{\Theta}_4) &- \partial T/\partial \Theta_4 + \partial V/\partial \Theta_4 = 0
\end{aligned} \qquad (3.23)$$

For convenience, we introduce the following abbreviations in eqs. (3.14) and (3.18):

$$\Phi = \Theta_2 + \Theta_3 \qquad \dot{\Phi} = \dot{\Theta}_2 + \dot{\Theta}_3 \qquad (3.24)$$

$$\Psi = \Theta_2 + \Theta_3 + \Theta_3 \qquad \dot{\Psi} = \dot{\Theta}_2 + \dot{\Theta}_3 + \dot{\Theta}_4 \qquad (3.25)$$

We differentiate the total kinetic energy T with respect to the angular velocities $\dot{\Theta}_i$ $i = 1, 2, 3, 4$:

$$\partial T/\partial \dot{\Theta}_1 = 2C_1\dot{\Theta}_1 + (C_7 + C_8 \sin\Phi + C_9 \sin\Psi)\dot{\Theta}_2 + (C_{10}\sin\Phi + C_{11}\sin\Psi) + C_{12}\sin\Psi\,\dot{\Theta}_4 \qquad (3.26.1)$$

$$\partial T/\partial \dot{\Theta}_2 = 2(C_2 + C_3\sin\Phi + C_4\sin\Psi)\dot{\Theta}_2 + (C_7 + C_8\sin\Phi + C_9\sin\Psi)\dot{\Theta}_1 + (C_{13} + C_{14}\sin\Phi + C_9\sin\Psi)\dot{\Theta}_3 + (C_{16} + C_{17}\sin\Psi)\dot{\Theta}_4 \qquad (3.26.2)$$

$$\partial T/\partial \dot{\Theta}_3 = (C_{10}\sin\Phi + C_{11}\sin\Psi)\dot{\Theta}_1 + (C_{13} + C_{14}\sin\Phi + C_{15}\sin\Psi)\dot{\Theta}_2 + 2C_5\dot{\Theta}_3 + C_{12}\dot{\Theta}_4 \qquad (3.26.3)$$

$$\partial T/\partial \dot{\Theta}_4 = C_{12}\sin\Phi\,\dot{\Theta}_1 + (C_{16} + C_{17}\sin\Psi)\dot{\Theta}_2 + C_{18}\dot{\Theta}_3 + 2C_6\dot{\Theta}_4 \qquad (3.26.4)$$

And differentiate the partial derivatives $(\partial T/\partial \dot{\Theta}_i)$ with respect to the time:

$$\begin{aligned}
d/dt\,(\partial T/\partial \dot{\Theta}_1) = {} & 2C_1\ddot{\Theta}_1 + (C_7 + C_8\sin\Phi + C_9\sin\Psi)\ddot{\Theta}_2 + \\
& + (C_{10}\sin\Phi + C_{11}\sin\Psi)\ddot{\Theta}_3 + C_{12}\sin\Psi\,\ddot{\Theta}_4 + \\
& + \dot{\Phi}\cos\Phi\,(C_8\dot{\Theta}_2 + C_{10}\dot{\Theta}_3) + \dot{\Psi}\cos\Psi\,(C_9\dot{\Theta}_2 + C_{11}\dot{\Theta}_3 + C_{12}\dot{\Theta}_4)
\end{aligned} \qquad (3.27.1)$$

$$d/dt\,(\partial T/\partial\dot{\Theta}_2) = (C_7 + C_8\sin\Phi + C_9\sin\Psi)\ddot{\Theta}_1 + \\ + 2(C_2 + C_3\sin\Phi + C_4\sin\Psi)\ddot{\Theta}_2 + (C_{13} + C_{14}\sin\Phi + C_{15}\sin\Psi)\ddot{\Theta}_3 + \\ + (C_{16} + C_{17}\sin\Psi)\ddot{\Theta}_4 + \dot{\Phi}\cos\Phi(2C_3\dot{\Theta}_2 + C_8\dot{\Theta}_1 + C_{14}\dot{\Theta}_3) + \\ + \dot{\Psi}\cos\Psi(C_9\dot{\Theta}_1 + 2C_4\dot{\Theta}_2 + C_{15}\dot{\Theta}_3 + C_{17}\dot{\Theta}_4) \tag{3.27.2}$$

$$d/dt\,(\partial T/\partial\dot{\Theta}_3) = (C_{10}\sin\Phi + C_{11}\sin\Psi)\ddot{\Theta}_1 + (C_{13} + C_{14}\sin\Phi + C_{15}\sin\Psi)\ddot{\Theta}_2 + \\ + 2C_5\ddot{\Theta}_3 + C_{18}\ddot{\Theta}_4 + \dot{\Phi}\cos\Phi(C_{10}\dot{\Theta}_1 + C_{14}\dot{\Theta}_2) + \\ + \dot{\Psi}\cos\Psi(C_{11}\dot{\Theta}_1 + C_{15}\dot{\Theta}_2) \tag{3.27.3}$$

$$d/dt\,(\partial T/\partial\dot{\Theta}_4) = C_{12}\sin\Psi\ddot{\Theta}_1 + (C_{16} + C_{17}\sin\Psi)\ddot{\Theta}_2 + C_{18}\dot{\Theta}_3 + 2C_6\ddot{\Theta}_4 + \\ + \dot{\Psi}\cos\Psi(C_{12}\dot{\Theta}_1 + C_{17}\dot{\Theta}_2) \tag{3.27.4}$$

The partial derivatives of the kinetic energy with respect to the generalized angles are:

$$\partial T/\partial\Theta_1 = 0 \tag{3.28.1}$$

$$\partial T/\partial\Theta_2 = (C_3\dot{\Theta}_2^2 + C_8\dot{\Theta}_1\dot{\Theta}_2 + C_{10}\dot{\Theta}_1\dot{\Theta}_3 + C_{14}\dot{\Theta}_2\dot{\Theta}_3)\cos\Phi + \\ + (C_4\dot{\Theta}_2^2 + C_9\dot{\Theta}_1\dot{\Theta}_2 + C_{11}\dot{\Theta}_1\dot{\Theta}_3 + C_{12}\dot{\Theta}_1\dot{\Theta}_4 + C_{15}\dot{\Theta}_2\dot{\Theta}_3 + C_{17}\dot{\Theta}_2\dot{\Theta}_4)\cos\Psi \tag{3.28.2}$$

$$\partial T/\partial\Theta_3 = (C_3\dot{\Theta}_2^2 + C_8\dot{\Theta}_1\dot{\Theta}_2 + C_{10}\dot{\Theta}_1\dot{\Theta}_3 + C_{14}\dot{\Theta}_2\dot{\Theta}_3)\cos\Psi \\ + (C_4\dot{\Theta}_2^2 + C_9\dot{\Theta}_1\dot{\Theta}_2 + C_{11}\dot{\Theta}_1\dot{\Theta}_3 + C_{12}\dot{\Theta}_1\dot{\Theta}_4 + C_{15}\dot{\Theta}_2\dot{\Theta}_3 + C_{17}\dot{\Theta}_2\dot{\Theta}_4)\cos\Psi \tag{3.28.3}$$

$$\partial T/\partial\Theta_4 = (C_4\dot{\Theta}_2^2 + C_9\dot{\Theta}_1\dot{\Theta}_2 + C_{11}\dot{\Theta}_1\dot{\Theta}_3 + C_{12}\dot{\Theta}_1\dot{\Theta}_4 + C_{15}\dot{\Theta}_2\dot{\Theta}_3 + C_{17}\dot{\Theta}_2\dot{\Theta}_4)\cos\Psi \tag{3.28.4}$$

The partial derivatives of the strain energy with respect to the generalized angles are:

$$\partial V/\partial\Theta_1 = \Lambda_1\Theta_1 - \Lambda_2(\Theta_2 - \Theta_1) \tag{3.29.1}$$

$$\partial V/\partial\Theta_2 = \Lambda_2(\Theta_2 - \Theta_1) \tag{3.29.2}$$

$$\partial V/\partial\Theta_3 = 0 \tag{3.29.3}$$

$$\partial V/\partial\Theta_4 = \Lambda_4\Theta_4 \tag{3.29.4}$$

We introduce eqs. (3.24.) - (3.29.) into eqs. (3.23) to obtain the differential equation of motion. The common notation for the differential equation of motion is :

$$\mathbf{M}\ddot{\boldsymbol{\Theta}} + \mathbf{C}\dot{\boldsymbol{\Theta}} + \mathbf{K}\boldsymbol{\Theta} = \mathbf{Q} \qquad (3.30)$$

where:

M = Constant mass matrix.

C = Constant quadratic structural damping matrix.

K = Constant quadratic stiffness matrix.

Q = Vector of generalized exciting forces: $\mathbf{Q} = \begin{bmatrix} 0 & -M_0 & M_0 & 0 \end{bmatrix}^T$

Θ = Vector of generalized coordinates.

Unfortunately, the particular problem yields a nonlinear differential equation system and the elements of the vector $\boldsymbol{\Theta}$ are arguments of trigonometric functions. Also, is the mass matrix time varying: $\mathbf{M} = \mathbf{M_0} + \mathbf{M}(t)$. Thus, we use the state space form of the differential equation of motion.

First, we can set the damping matrix **C** equal to zero, since we do not consider passive structural damping. Hence Eq. (3.30) gets:

$$\mathbf{M}\ddot{\boldsymbol{\Theta}} + \mathbf{K}\boldsymbol{\Theta} = \mathbf{Q} \qquad (3.31)$$

We define a new (4 x 1) vector:

$$\mathbf{p} = \mathbf{K}\boldsymbol{\Theta} \qquad (3.32)$$

Which yields the state space form of the differential equations:

$$\boxed{\ddot{\boldsymbol{\Theta}} = \mathbf{M}^{-1}(-\mathbf{p} + \mathbf{Q})} \qquad (3.33)$$

The right side of equation (3.33) is a (4 x 1) vector, where **M** is time varying mass matrix. Introducing the partial derivatives (3.24) - (3.29) into the system of differential equations (3.23), the vector $\ddot{\boldsymbol{\Theta}}$ can be separated, which yields the mass matrix **M** and to the vector **p**.

Introducing the partial derivatives (3.24) - (3.29) into the Lagrange equation (3.23), yields a system of 4 coupled, nonlinear, second order, differential equations:

$$2C_1\ddot{\Theta}_1 + (C_7 + C_8 + C_9\sin\Psi)\ddot{\Theta}_2 + (C_{10}\sin\Phi + C_{11}\sin\Psi)\ddot{\Theta}_3 + C_{12}\sin\Psi\,\ddot{\Theta}_4 =$$
$$\dot{\Phi}\cos\Phi\,(C_8\dot{\Theta}_2 + C_{10}\dot{\Theta}_3) + \dot{\Psi}\cos\Psi\,(C_9\dot{\Theta}_2 + C_{11}\dot{\Theta}_3 + C_{12}\dot{\Theta}_4) + \Lambda_1\Theta_1 - \Lambda_2(\Theta_2 - \Theta_1)$$

$$(C_7 + C_8\sin\Phi + C_9\sin\Psi)\ddot{\Theta}_1 + 2(C_2 + C_3\sin\Phi + C_4\sin\Psi)\ddot{\Theta}_2 + (C_{13} + C_{14}\sin\Phi + C_{15}\sin\Psi)\ddot{\Theta}_3 +$$
$$+ (C_{16} + C_{17}\sin\Psi)\ddot{\Theta}_4 =$$
$$\dot{\Phi}\cos\Phi(2C_3\dot{\Theta}_2 + C_8\dot{\Theta}_1 + C_{14}\dot{\Theta}_3) + \dot{\Psi}\cos\Psi(2C_4\dot{\Theta}_2 + C_9\dot{\Theta}_1 + C_{15}\dot{\Theta}_3 + C_{17}\dot{\Theta}_4) +$$
$$+ \cos\Phi(C_3\dot{\Theta}_2^2 + C_8\dot{\Theta}_1\dot{\Theta}_2 + C_{10}\dot{\Theta}_1\dot{\Theta}_3 + C_{14}\dot{\Theta}_2\dot{\Theta}_3) +$$
$$- \cos\Psi(C_4\dot{\Theta}_2^2 + C_9\dot{\Theta}_1\dot{\Theta}_2 + C_{11}\dot{\Theta}_1\dot{\Theta}_3 + C_{12}\dot{\Theta}_1\dot{\Theta}_4 + C_{15}\dot{\Theta}_2\dot{\Theta}_3 + C_{17}\dot{\Theta}_2\dot{\Theta}_4) +$$
$$+ \Lambda_2(\Theta_2 - \Theta_1) - M_0$$

$$(C_{10}\sin\Phi + C_{11}\sin\Psi)\ddot{\Theta}_1 + (C_{13} + C_{14}\sin\Phi + C_{15}\sin\Psi)\ddot{\Theta}_2 + 2C_5\ddot{\Theta}_3 + C_{18}\ddot{\Theta}_4 =$$
$$\dot{\Phi}\cos\Phi(C_{10}\dot{\Theta}_1 + C_{14}\dot{\Theta}_2) - \dot{\Psi}\cos\Psi(C_{11}\dot{\Theta}_1 + C_{15}\dot{\Theta}_2) +$$
$$-\cos\Phi(C_3\dot{\Theta}_2^2 + C_8\dot{\Theta}_1\dot{\Theta}_2 + C_{10}\dot{\Theta}_1\dot{\Theta}_3 + C_{14}\dot{\Theta}_2\dot{\Theta}_3) +$$
$$-\cos\Psi(C_4\dot{\Theta}_2^2 + C_9\dot{\Theta}_1\dot{\Theta}_2 + C_{11}\dot{\Theta}_1\dot{\Theta}_3 + C_{12}\dot{\Theta}_1\dot{\Theta}_4 + C_{15}\dot{\Theta}_2\dot{\Theta}_3 + C_{17}\dot{\Theta}_2\dot{\Theta}_4) + M_0$$

$$C_{12}\sin\Psi\,\ddot{\Theta}_1 + (C_{16} + C_{17}\sin\Psi)\ddot{\Theta}_2 + 2C_{18}\ddot{\Theta}_3 + 2C_6\ddot{\Theta}_4 =$$
$$-\dot{\Psi}\cos\Psi\,(C_{12}\dot{\Theta}_1 + C_{17}\dot{\Theta}_2) - \cos\Psi(C_4\dot{\Theta}_2^2 + C_9\dot{\Theta}_1\dot{\Theta}_2 + C_{11}\dot{\Theta}_1\dot{\Theta}_3 + C_{12}\dot{\Theta}_1\dot{\Theta}_4 + C_{15}\dot{\Theta}_2\dot{\Theta}_3\ C_{17}\dot{\Theta}_2\dot{\Theta}_4) +$$
$$+ \Lambda_4\Theta_4$$

The left side oft the previous equation system can be written in matrix notation, where **M** denotes the time varying and symmetric mass matrix:

$$\mathbf{M} = \begin{bmatrix} 2C_1 & C_7 + C_8 \sin\Phi + C_9 \sin\Psi & C_{10} \sin\Phi + C_{11} \sin\Psi & C_{12} \sin\Psi \\ C_7 + C_8 \sin\Phi + C_9 \sin\Psi & 2(C_2 + C_3 \sin\Phi + C_4 \sin\Psi) & C_{13} + C_{14} \sin\Phi + C_{15} \sin\Psi & C_{16} + C_{17} \sin\Psi \\ C_{10} \sin\Phi + C_{11} \sin\Psi & C_{13} + C_{14} \sin\Phi + C_{15} \sin\Psi & 2C_5 & C_{18} \\ C_{12} \sin\Psi & C_{16} + C_{17} \sin\Psi & C_{18} & 2C_6 \end{bmatrix} \tag{3.34}$$

The elements of vector **p** of eq. (3.33) and the exciting force vector **Q** are written on the right side of the previous differential equation system.

$$\mathbf{p} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix} \qquad (3.35)$$

$$p_1 = \dot{\Phi} \cos \Phi \,(C_8\dot{\Theta}_2 + C_{10}\dot{\Theta}_3) + \dot{\Psi} \cos \Psi \,(C_9\dot{\Theta}_2 + C_{11}\dot{\Theta}_3 + C_{12}\dot{\Theta}_4) + \Lambda_1\Theta_1 - \Lambda_2(\Theta_2 - \Theta_1) \qquad (3.35.1)$$

$$\begin{aligned} p_2 = {} & \dot{\Phi} \cos \Phi\,(2C_3\dot{\Theta}_2 + C_8\dot{\Theta}_1 + C_{14}\dot{\Theta}_3) + \\ & + \dot{\Psi} \cos \Psi\,(2C_4\dot{\Theta}_2 + C_9\dot{\Theta}_1 + C_{15}\dot{\Theta}_3 + C_{17}\dot{\Theta}_4) + \\ & + \cos \Phi\,(C_3\dot{\Theta}_2^2 + C_8\dot{\Theta}_1\dot{\Theta}_2 + C_{10}\dot{\Theta}_1\dot{\Theta}_3 + C_{14}\dot{\Theta}_2\dot{\Theta}_3) + \\ & - \cos \Psi\,(C_4\dot{\Theta}_2^2 + C_9\dot{\Theta}_1\dot{\Theta}_2 + C_{11}\dot{\Theta}_1\dot{\Theta}_3 + C_{12}\dot{\Theta}_1\dot{\Theta}_4 + C_{15}\dot{\Theta}_2\dot{\Theta}_3 + C_{17}\dot{\Theta}_2\dot{\Theta}_4) + \\ & + \Lambda_2(\Theta_2 - \Theta_1) \end{aligned} \qquad (3.35.2)$$

$$\begin{aligned} p_3 = {} & \dot{\Phi} \cos \Phi\,(C_{10}\dot{\Theta}_1 + C_{14}\dot{\Theta}_2) - \dot{\Psi} \cos \Psi\,(C_{11}\dot{\Theta}_1 + C_{15}\dot{\Theta}_2) + \\ & - \cos \Phi\,(C_3\dot{\Theta}_2^2 + C_8\dot{\Theta}_1\dot{\Theta}_2 + C_{10}\dot{\Theta}_1\dot{\Theta}_3 + C_{14}\dot{\Theta}_2\dot{\Theta}_3) + \\ & - \cos \Psi\,(C_4\dot{\Theta}_2^2 + C_9\dot{\Theta}_1\dot{\Theta}_2 + C_{11}\dot{\Theta}_1\dot{\Theta}_3 + C_{12}\dot{\Theta}_1\dot{\Theta}_4 + C_{15}\dot{\Theta}_2\dot{\Theta}_3 + C_{17}\dot{\Theta}_2\dot{\Theta}_4) \end{aligned} \qquad (3.35.3)$$

$$\begin{aligned} p_4 = {} & -\dot{\Psi} \cos \Psi\,(C_{12}\dot{\Theta}_1 + C_{17}\dot{\Theta}_2) + \\ & - \cos \Psi\,(C_4\dot{\Theta}_2^2 + C_9\dot{\Theta}_1\dot{\Theta}_2 + C_{11}\dot{\Theta}_1\dot{\Theta}_3 + C_{12}\dot{\Theta}_1\dot{\Theta}_4 + C_{15}\dot{\Theta}_2\dot{\Theta}_3\ C_{17}\dot{\Theta}_2\dot{\Theta}_4) + \\ & + \Lambda_4\Theta_4 \end{aligned} \qquad (3.35.4)$$

Exciting force vector:

$$\mathbf{Q} = \begin{bmatrix} 0 \\ -M_0(t) \\ +M_0(t) \\ 0 \end{bmatrix} \qquad (3.36)$$

3.4 SOLVING THE DIFFERENTIAL EQUATION OF MOTION

The differential equation of motion is given by eq. (3.33), where the right side denotes $\ddot{\boldsymbol{\Theta}} = \left[\ddot{\Theta}_1 \;\; \ddot{\Theta}_2 \;\; \ddot{\Theta}_3 \;\; \ddot{\Theta}_4\right]^T$ and the left side is a (4 x 1) vector which is a nonlinear function of $\boldsymbol{\Theta}$, $\dot{\boldsymbol{\Theta}}$ and time :

$$\ddot{\boldsymbol{\Theta}} = \mathbf{M}^{-1}(-\mathbf{p} + \mathbf{Q}) \qquad (3.33)$$

Now, we solve this system numerically, to obtain the time history of the generalized coordinates $\Theta_1(t)$, $\Theta_2(t)$, $\Theta_3(t)$, $\Theta_4(t)$, when applying a torque M (t) at the gimbal.
For convenience, we introduce a new (4 x 1) vector **b**, which we define as:

$$\mathbf{b} = \mathbf{M}^{-1}(-\mathbf{p} + \mathbf{q}) \qquad (3.37)$$

Hence we can write the differential equation system as:

$$\ddot{\boldsymbol{\Theta}} = \mathbf{b} \qquad (3.38)$$

Which are 4 coupled differential equations of the second order. We do not determine the vector **b** analytically, because the effort is too high and it would yield to vast equations. Since we know that **b** is a function of $\boldsymbol{\Theta}$, $\dot{\boldsymbol{\Theta}}$, t , we can write equation (3.38) as:

$$\begin{aligned}
\ddot{\Theta}_1 &= b_1(\Theta_1, \Theta_2, \Theta_3, \Theta_4, \dot{\Theta}_1, \dot{\Theta}_2, \dot{\Theta}_3, \dot{\Theta}_4, t) \\
\ddot{\Theta}_2 &= b_2(\Theta_1, \Theta_2, \Theta_3, \Theta_4, \dot{\Theta}_1, \dot{\Theta}_2, \dot{\Theta}_3, \dot{\Theta}_4, t) \\
\ddot{\Theta}_3 &= b_3(\Theta_1, \Theta_2, \Theta_3, \Theta_4, \dot{\Theta}_1, \dot{\Theta}_2, \dot{\Theta}_3, \dot{\Theta}_4, t) \\
\ddot{\Theta}_4 &= b_4(\Theta_1, \Theta_2, \Theta_3, \Theta_4, \dot{\Theta}_1, \dot{\Theta}_2, \dot{\Theta}_3, \dot{\Theta}_4, t)
\end{aligned} \qquad (3.39)$$

We will use a numerical method to solve this system, because a direct analytical solution is not feasible. First, we reduce the second order system to a first order system. Therefore, we define new variables:

$$\begin{aligned} Y_1(t) &= \Theta_1(t) & \quad Y_5(t) &= \dot{\Theta}_1(t) \\ Y_2(t) &= \Theta_2(t) & \quad Y_6(t) &= \dot{\Theta}_2(t) \\ Y_3(t) &= \Theta_3(t) & \quad Y_7(t) &= \dot{\Theta}_3(t) \\ Y_4(t) &= \Theta_4(t) & \quad Y_8(t) &= \dot{\Theta}_4(t) \end{aligned} \qquad (3.40)$$

Using these variables, the 4 equations / second order system, becomes a 8 equations / first order system:

$$\begin{aligned} \dot{Y}_1 &= Y_5 \\ \dot{Y}_2 &= Y_6 \\ \dot{Y}_3 &= Y_7 \\ \dot{Y}_4 &= Y_8 \\ \dot{Y}_5 &= b_1(Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_8, t) \\ \dot{Y}_6 &= b_2(Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_8, t) \\ \dot{Y}_7 &= b_3(Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_8, t) \\ \dot{Y}_8 &= b_4(Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_8, t) \end{aligned} \qquad (3.41)$$

This first order equation system describes a typical initial value problem:

$$\dot{\mathbf{Y}} = \text{Fct.}(\mathbf{Y}, t)$$

We can use common standard software, to obtain a numerical solution. For this thesis, there is used a Runge Kutta iteration, which is taken from the mathematical software library of the University Aachen in Germany. The Runge Kutta subroutine is written in Fortran 77 and the source code is printed in the appendix. In addition to this Runge Kutta subroutine, we have to develop further subroutines and a main program which suits our specific requirements. Especially we need a program, which calculates the time history of the generalized angles Θ_i. This requires a numerical solution of the differential equation of motion for continuing descrete time values within given time boundaries In addition, we are also interested in the total vibrational energy of the system. We define the tasks, which our program has to fulfill:

1. interactive input: 1.a Initial values of angular displacements and angular velocities.

   1.b Time boundaries (t_{start}, t_{end}) and number of K discrete time values, for which the initial value problem has to be solved.

   1.c Demanded accuracy and maximum permissible number N of iteration steps.

2. Determine discrete time values.

3. Calculate numerical solutions of the initial value problem for all discrete time values.

4. Calculate the total vibrational energy for all discrete time values.

5. Write results on file. Use format, which is readable by a graphical post-processing program.

The program is named "IVP" (**I**nitial **V**alue **P**roblem), and all subroutines are written in Fortran 77. The source code is printed in the appendix. The structure of "IVP", can be seen on the simplified floating chart (fig. 3.11)

Brief description of the program "IVP"

The program IVP has an interactive input by keyboard:

$\mathbf{Y}_{initial}$ = Initial value vector for angular displacements and angular velocities (8 values).

t_{start}, t_{end} = Time values for start and end.

K = Number of required discrete solutions within the time boundaries.

ε_{max} = Limit for the absolute error of the iteration. The iteration terminates, if the absolute error is less then ε_{max}.

N = Number of iteration steps, after which the iteration terminates, even if the demanded accuracy is not succeeded.

The Runge Kutta iteration subroutine calculates discrete numerical solutions of the differential equation.

Input: Initial value vector $\mathbf{Y}_{initial}$, initial time value $t_{initial}$, and aimed time value t_{aim} .

Output: Aimed vector $\mathbf{Y}_{aim}$ for aimed time value.

All input values are provided by the main program. The iteration terminates, either if the demanded accuracy is succeeded, or if the maximum number of iteration steps is succeeded. Since the mass matrix is a function of the variables, it has to be calculated and inverted for each iteration step.

An additional subroutine calculates the total vibrational energy for each discrete time value. The total energy TE is the sum of the total kinetic and the total potential energy:

$$TE = T + V$$

The total energies are given by eqs. (3.14) and (3.18). To obtain the total vibrational energy, we have to deduct the total energy, which the system would have, if it would be infinite stiff. This energy is simply the the product of constant torque M_0 and pitch angle α (eq. (3.1)).

$$TE_{rigid} = M_0 \alpha$$

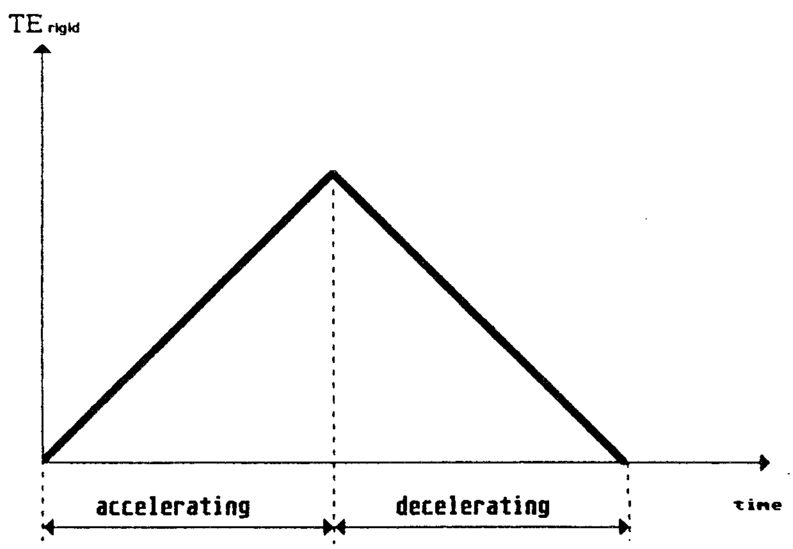


Fig. 3.10

Thus we obtain the total vibrational energy:

$$TE' = T + V - M_0 \alpha$$

On run time, all results are stored in Matrix **A**:

0	t_{start}	$\Theta_1(t_{start})$	$\Theta_2(t_{start})$	...	$\dot{\Theta}_1(t_{start})$	$\dot{\Theta}_2(t_{start})$	$TE'(t_{start})$
1	t_1	$\Theta_1(t_1)$	$\Theta_2(t_1)$	...	$\dot{\Theta}_1(t_1)$	$\dot{\Theta}_2(t_1)$	$TE'(t_1)$
2	t_2	$\Theta_1(t_2)$	$\Theta_2(t_2)$	...	$\dot{\Theta}_1(t_2)$	$\dot{\Theta}_2(t_2)$	$TE'(t_2)$
.							
.							
.							
.							
K	t_{end}	$\Theta_1(t_{end})$	$\Theta_2(t_{end})$	...	$\dot{\Theta}_1(t_{end})$	$\dot{\Theta}_2(t_{end})$	$TE'(t_{end})$

Finally, this Matrix will be stored on hard disk and is readable by a graphical post-processing program.

FLOATING CHART OF THE PROGRAM IVP

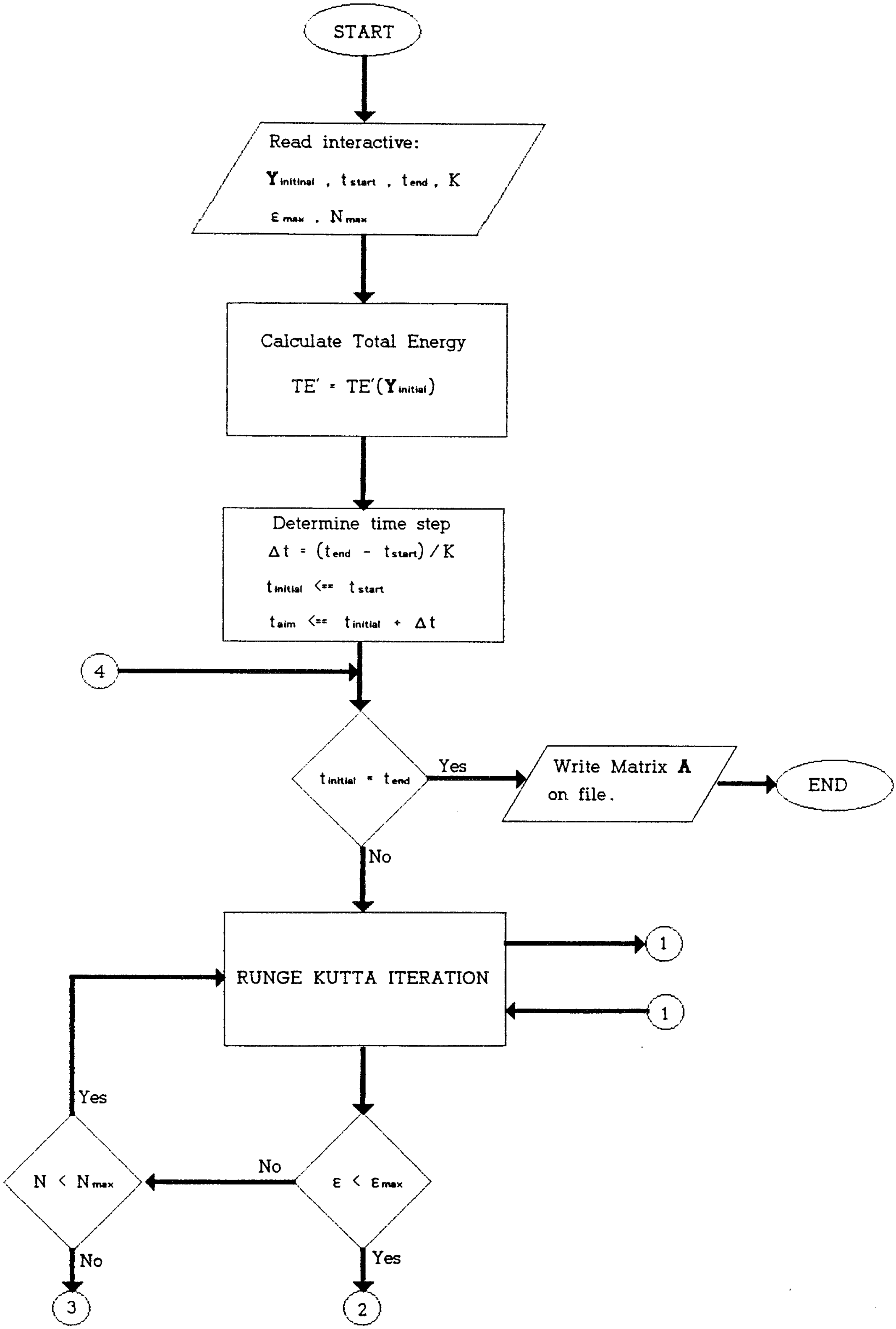


Fig. 3.11

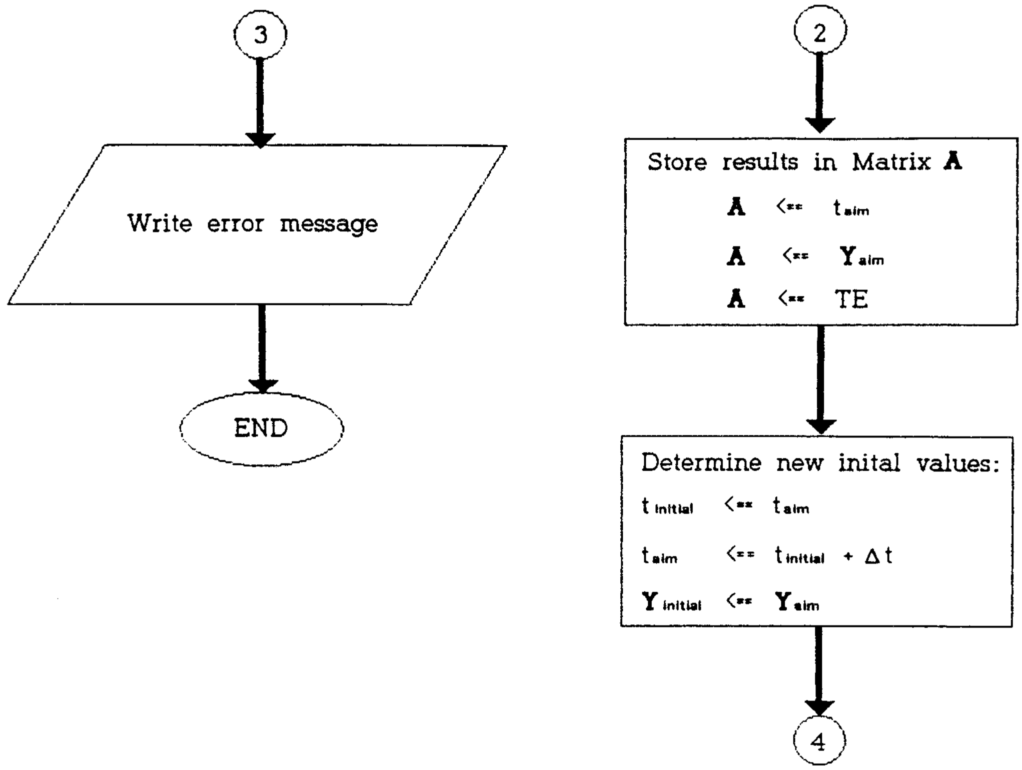

3
Write error message
END
2
Store results in Matrix A
A <== t aim
A <== Y aim
A <== TE
Determine new inital values:
t initial <== t aim
t aim <== t initial + Δt
Y initial <== Y aim
4

Fig. 3.11

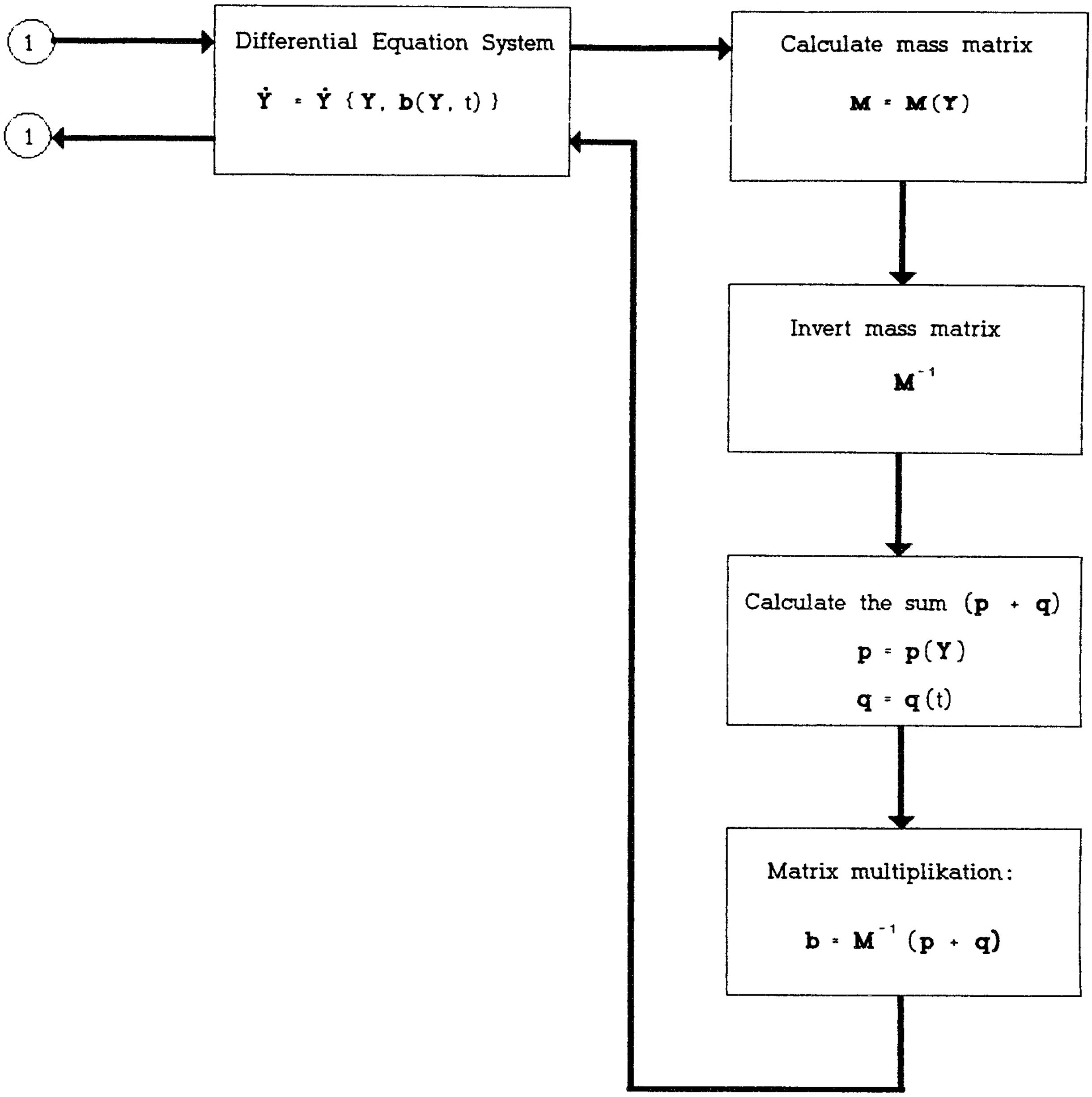

1
1
Differential Equation System
$\dot{\mathbf{Y}} = \dot{\mathbf{Y}}\{\mathbf{Y}, \mathbf{b}(\mathbf{Y}, t)\}$
Calculate mass matrix
$\mathbf{M} = \mathbf{M}(\mathbf{Y})$
Invert mass matrix
$\mathbf{M}^{-1}$
Calculate the sum $(\mathbf{p} + \mathbf{q})$
$\mathbf{p} = \mathbf{p}(\mathbf{Y})$
$\mathbf{q} = \mathbf{q}(t)$
Matrix multiplikation:
$\mathbf{b} = \mathbf{M}^{-1}(\mathbf{p} + \mathbf{q})$

Fig. 3.11

NUMERICAL RESULTS

We use the program "IVP" to calculate the time history of the angular displacements for a typical pitch maneuver. The program "IVP" is valid for arbitrary pitch maneuvers and arbitrary initial values. As an example we investigate a pitch maneuver from $\alpha_0 = 0°$ to $\alpha_2 = 90°$. We assume no vibrations initially. To achieve this maneuver, we apply a constant torque $M_0 = (+/-)$ 30 Nm at the gimbal. The maneuver is divided in constant acceleration and constant deceleration. If we neglect friction, both parts of the maneuver take the same time. Thus acceleration goes from $\alpha_0 = 0°$ to $\alpha_1 = 45°$ and decelaration goes from $\alpha_1 = 45°$ to $\alpha_2 = 90°$.

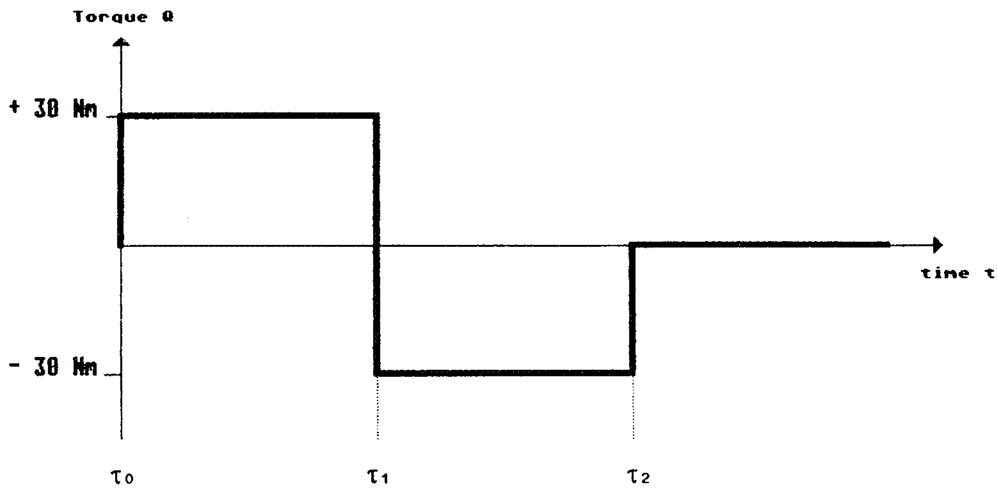


fig. 3.12

We determine the time for constant acceleration from $\alpha_0 = 0$ to α_1.
If $\dot{\alpha}_0 = 0$, we can write:

$$(\tau_1 - \tau_0) = \sqrt{2\,\alpha_1/\ddot{\alpha}} \tag{3.45}$$

The constant angular acceleration $\ddot{\alpha}$ is determined by the constant torque M_0 of the servo motor and the moment of inertia I of the antenna dish:

$$\ddot{\alpha} = M_0/I \tag{3.46}$$

Thus eq. (3.45) gets:

$$(\tau_1 - \tau_0) = \sqrt{2\,I\,\alpha_1/M_0} \tag{3.47}$$

The moment of inertia of the antenna dish is $I = 102\,143.4\ kgm^2$. Applying equation (3.47), we obtain the results:

Pitch maneuver:	time:
$\alpha_1 - \alpha_0 = 45°$	$\tau_1 - \tau_0 = 73.135$ sec
$\alpha_2 - \alpha_1 = 45°$	$\tau_2 - \tau_1 = 73.135$ sec
$\alpha_2 - \alpha_0 = 90°$	$\tau_2 - \tau_0 = 142.263$ sec

We are interested in the time history of the displacements for the complete pitch maneuver plus the first 70 seconds after the satellite dish has reached its desired position. Since the torque $M = M(t)$ is a step function, we have to divide the calculations in 3 parts and link the results together afterwards. The final values of the previous parts are the initial values of the following parts. Only the motor torque changes.

INITIAL AND FINAL VALUES

time:	τ_0	τ_1	τ_2
Θ_1 [rad]	0	-1.555 E-4	2.053 E-4
Θ_2 [rad]	0	-2.811 E-4	4.311 E-4
Θ_3 [rad]	0	0.7862	1.5690
Θ_4 [rad]	0	-6.458 E-4	9.252 E-4
$\dot{\Theta}_1$ [rad/sec]	0	-1.199 E-5	- 6.438 E-6
$\dot{\Theta}_2$ [rad/sec]	0	-3.428 E-5	-1.157 E-4
$\dot{\Theta}_3$ [rad/sec]	0	1.973 E-2	-6.173 E-3
$\dot{\Theta}_4$ [rad/sec]	0	2.696 E-3	9.496 E-3
t [sec]	0	73.135	146.263
TE [Nm]	0	5.127 E-2	1.526 E-1

For all calculations the maximum absolute error was set to ε_{max} = 1E-6 and it was used an Atari ST computer. (Motorola 68000 processor, 16 bit, 8Mhz). The running time for calculating 300 discrete solution vectors was approximately 40 minutes. The results are shown in the diagrams:

Diagram 3.1	Angular displacement $\Theta_1(t)$
Diagram 3.2	Angular displacement $\Theta_2(t)$
Diagram 3.3	Angular displacement $\Theta_3(t)$
Diagram 3.4	Angular velocity $\dot{\Theta}_3(t)$
Diagram 3.5	Angular displacement $\Theta 4(t)$
Diagram 3.6	Total energy TE(t)

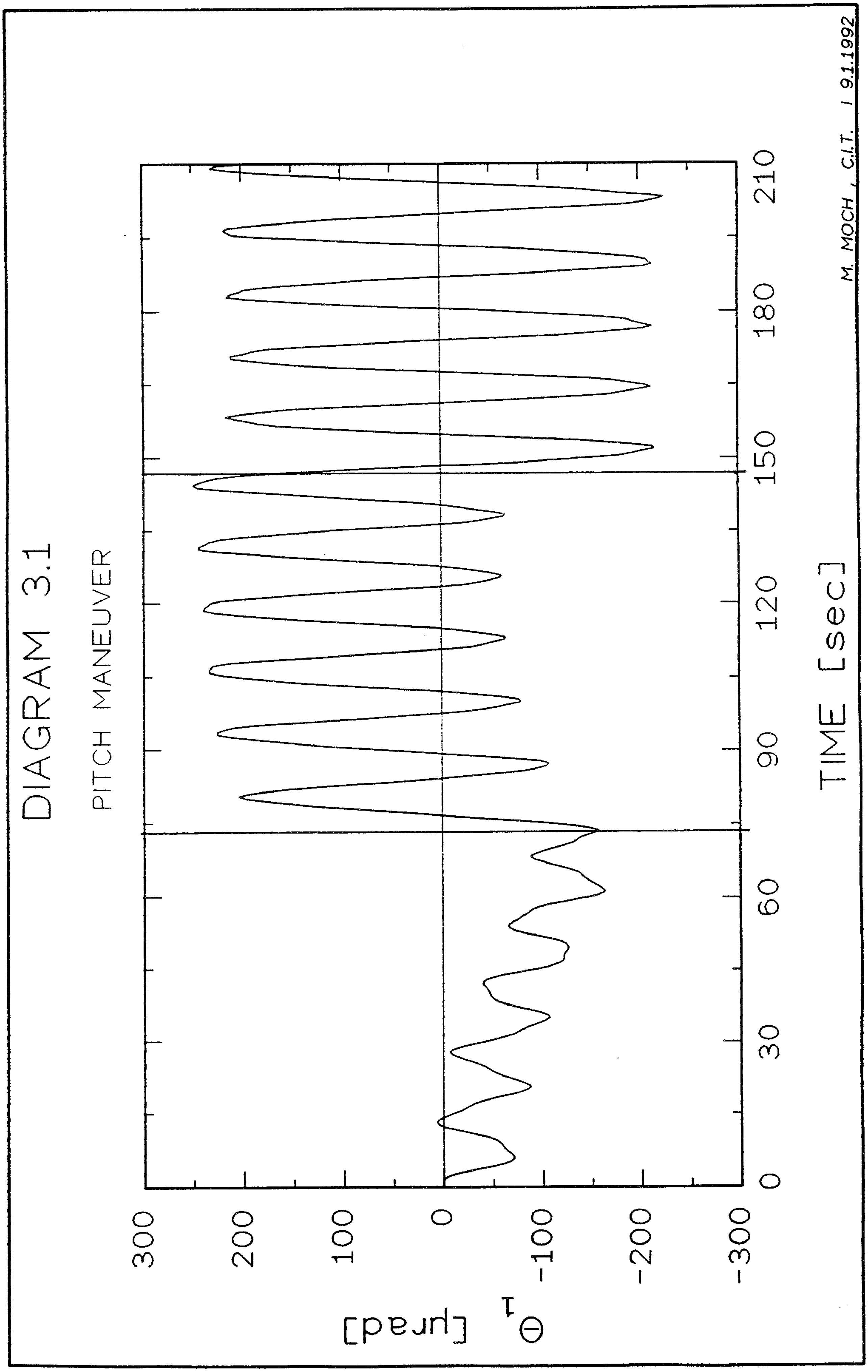


DIAGRAM 3.1
PITCH MANEUVER
Θ1 [μrad]
300
200
100
0
-100
-200
-300
0
30
60
90
120
150
180
210
TIME [sec]
M. MOCH , C.I.T. I 9.1.1992

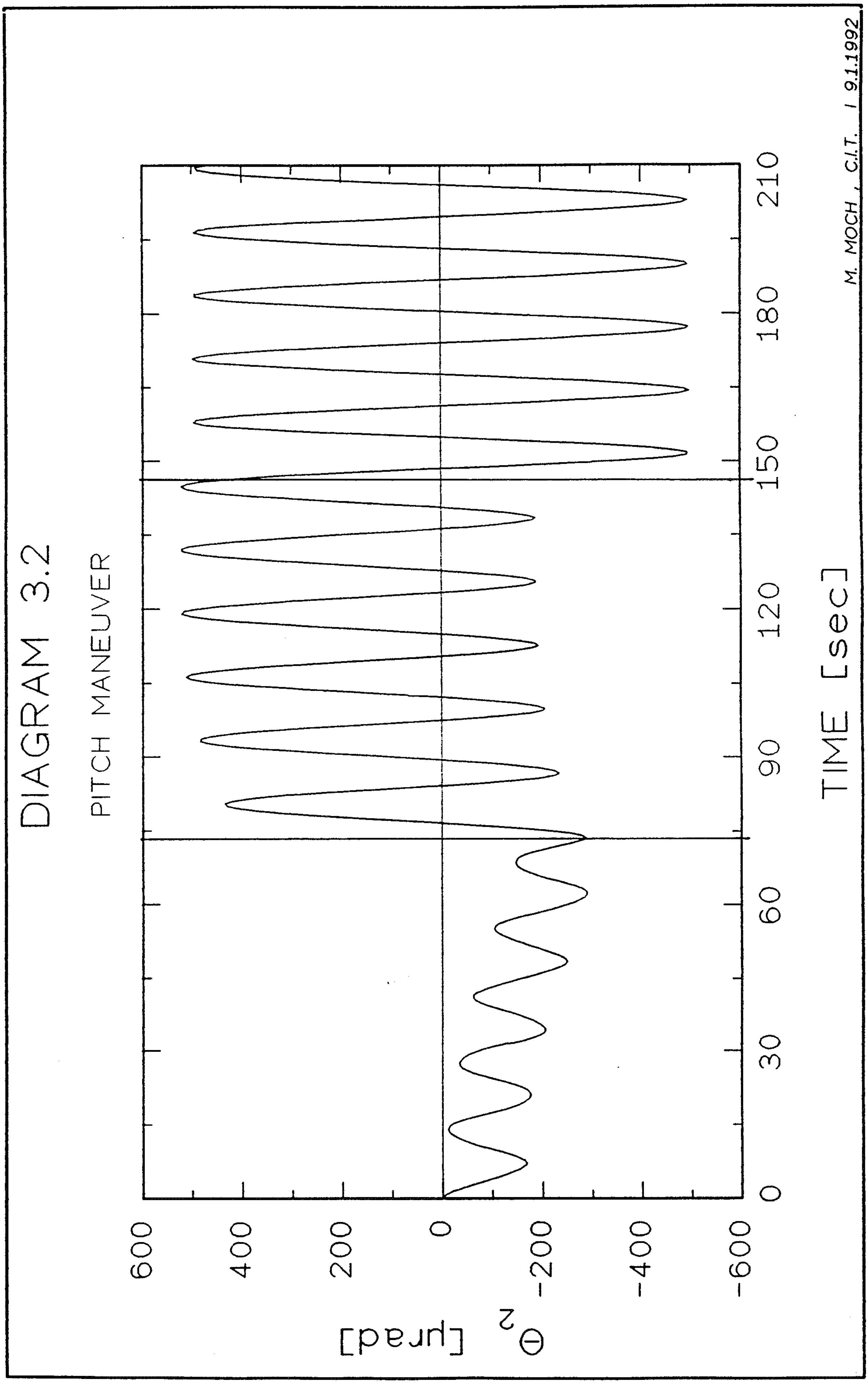


DIAGRAM 3.2
PITCH MANEUVER
TIME [sec]
0
30
60
90
120
150
180
210
600
400
200
0
-200
-400
-600
Θ2 [µrad]
M. MOCH , C.I.T. / 9.1.1992

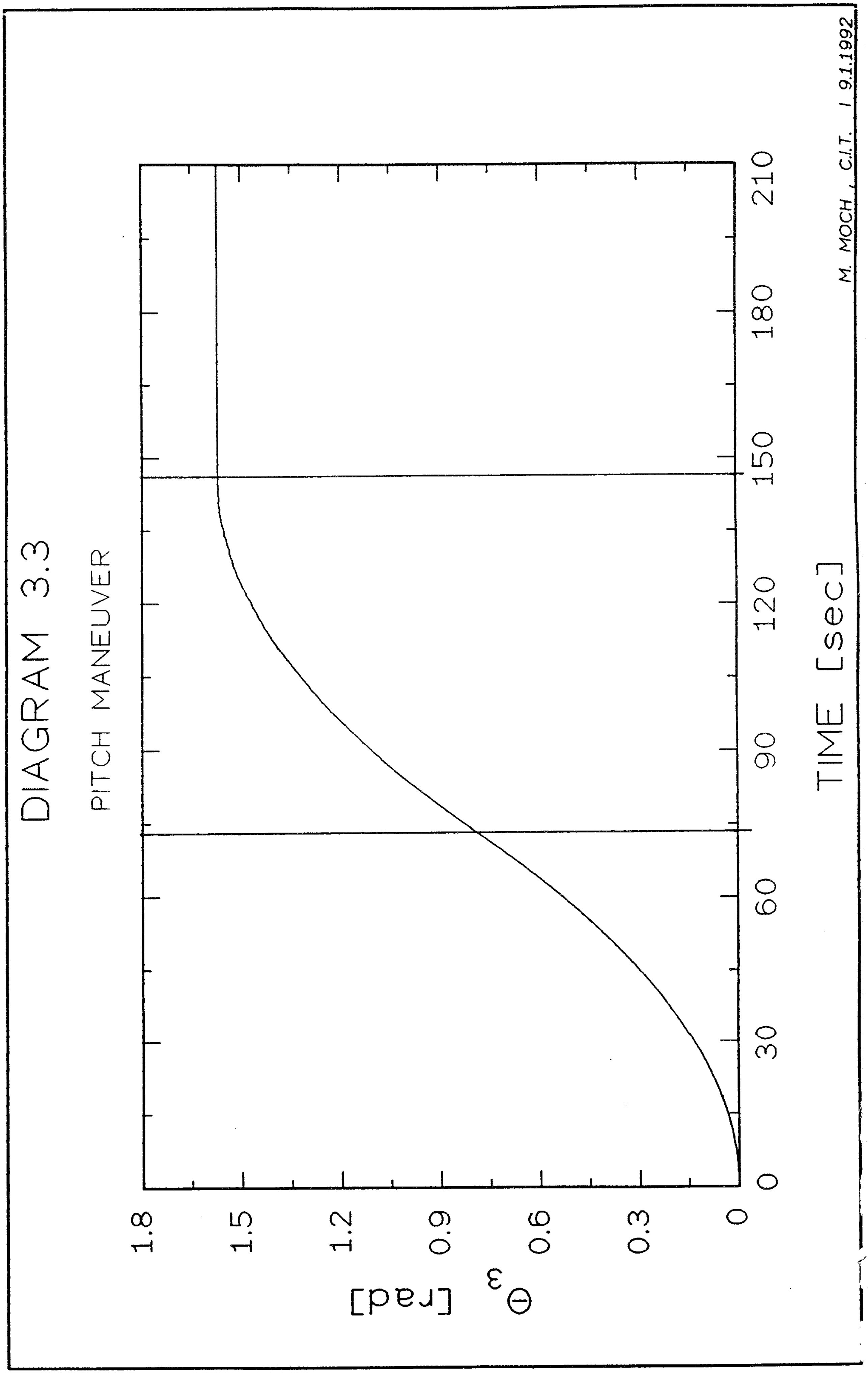


DIAGRAM 3.3
PITCH MANEUVER
TIME [sec]
0
30
60
90
120
150
180
210
Θ_3 [rad]
0
0.3
0.6
0.9
1.2
1.5
1.8
M. MOCH , C.I.T. / 9.1.1992

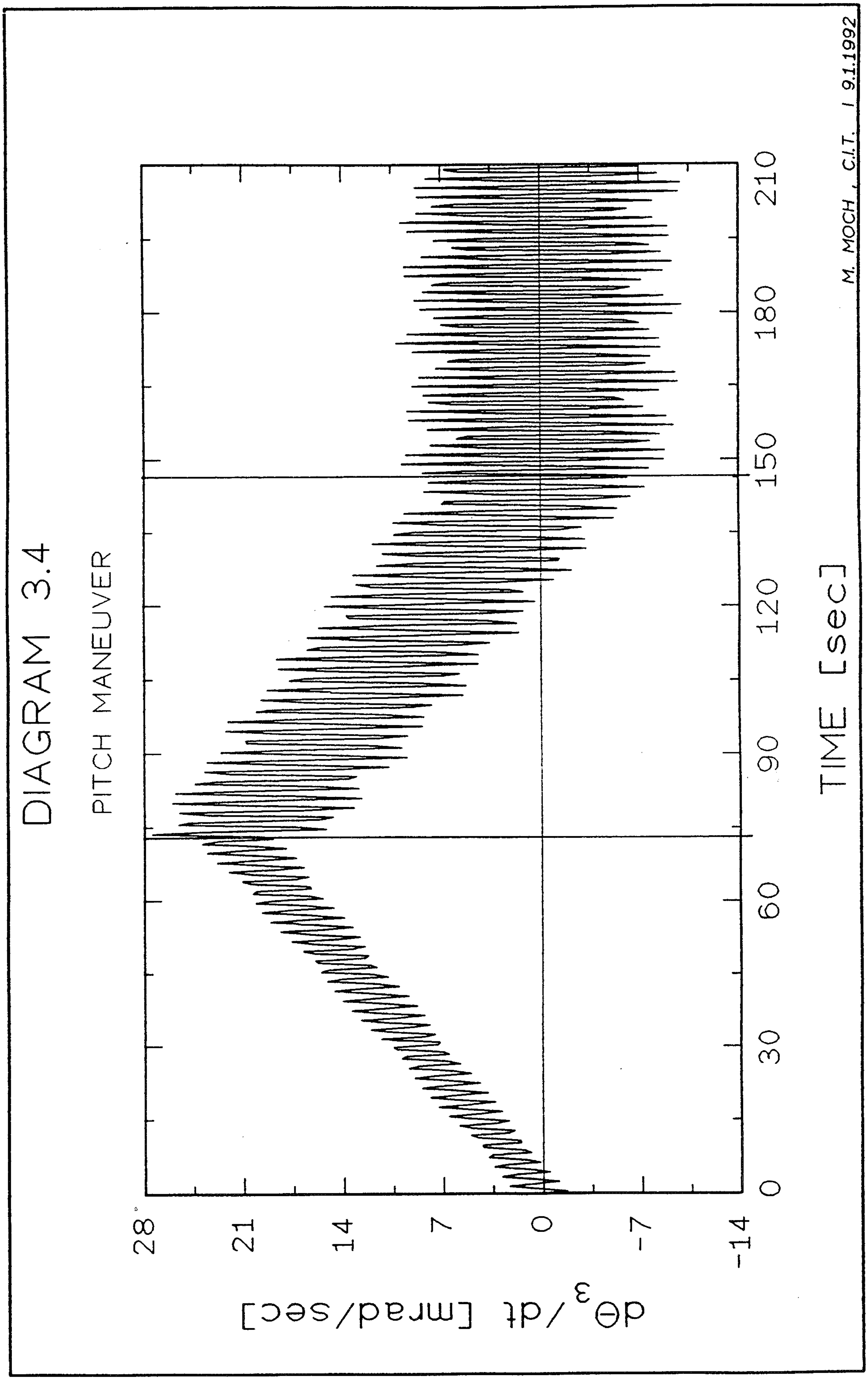


DIAGRAM 3.4
PITCH MANEUVER
dΘ3/dt [mrad/sec]
28
21
14
7
0
-7
-14
0
30
60
90
120
150
180
210
TIME [sec]
M. MOCH , C.I.T. I 9.1.1992

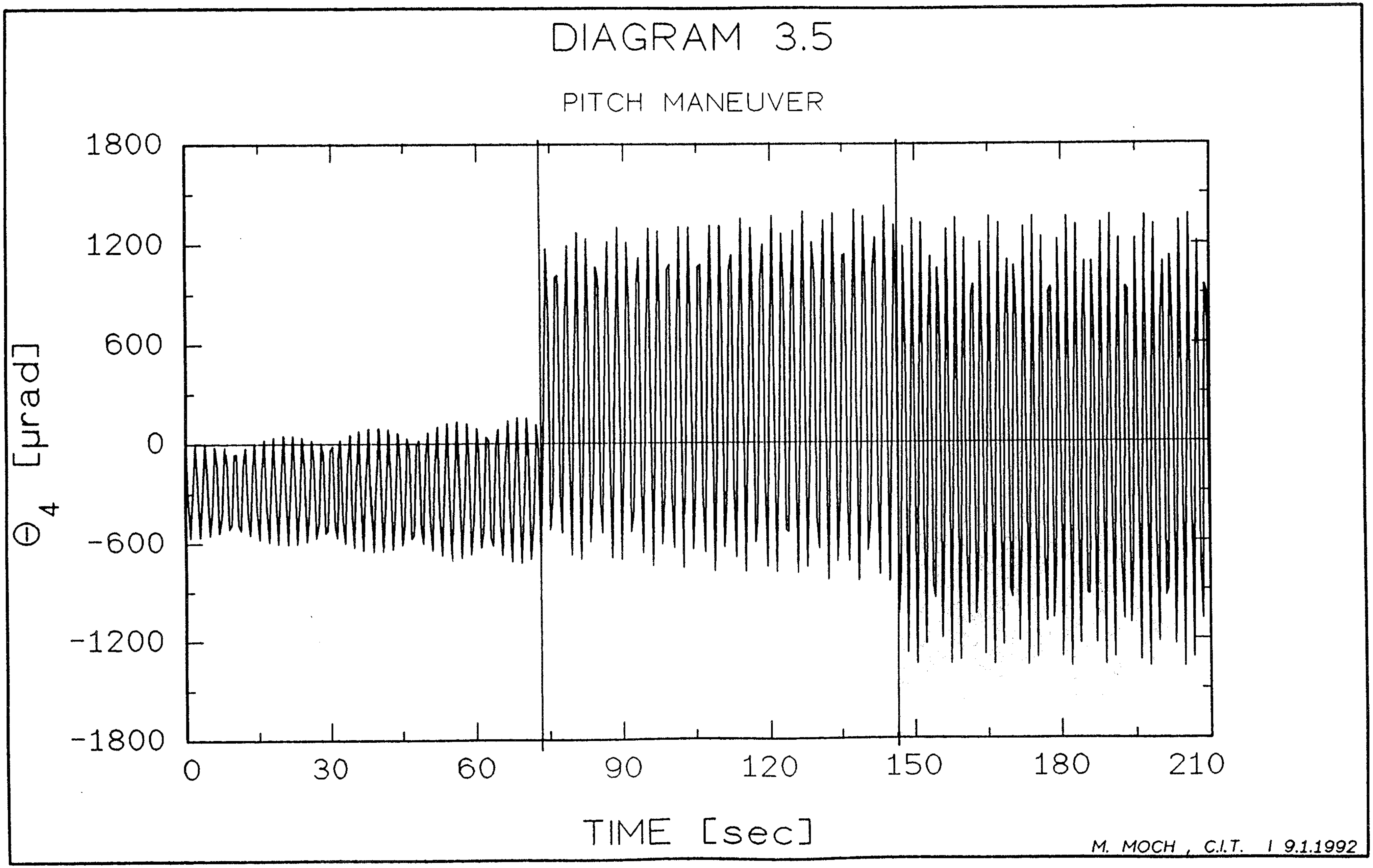
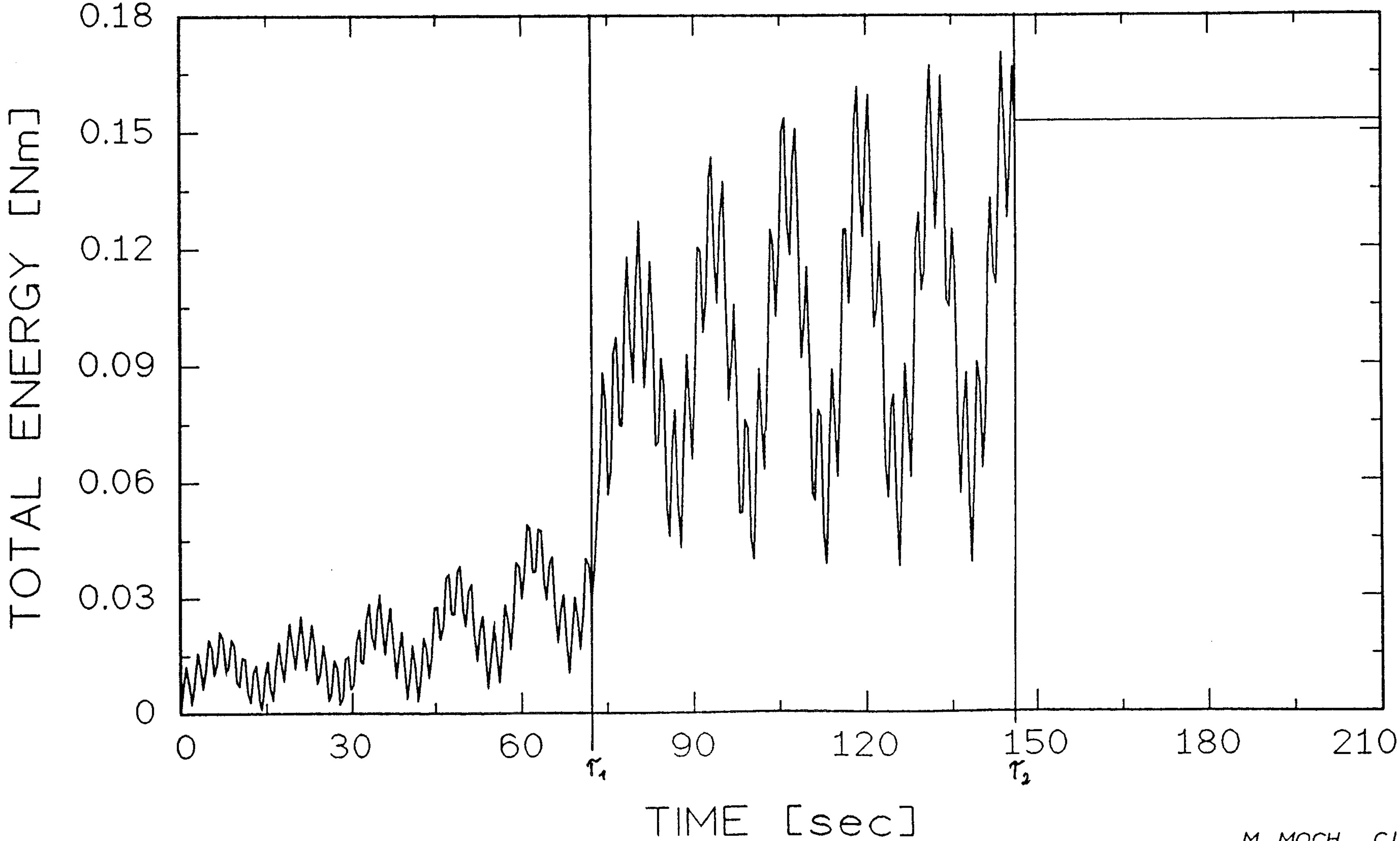


DIAGRAM 3.5
PITCH MANEUVER
Θ4 [μrad]
1800
1200
600
0
-600
-1200
-1800
0
30
60
90
120
150
180
210
TIME [sec]
M. MOCH , C.I.T. / 9.1.1992

DIAGRAM 3.6

PITCH MANEUVER



M. MOCH , C.I.T. I 9.1.1992

3.6 VERIFICATION OF THE RESULTS

Since the results are obtained by numerical iteration we have to verify them. For this purpose, we will apply following verification opportunities:

A. Compare the magnitude of dynamic displacements with (theoretical) static displacements.

B. Compare the periods of vibration with the natural frequencies, obtained by eigenvalue analysis.

C. Calculate the total energy as an analytical function of the results. For no damping and no external forces, the total energy must remain constant.

D. Compare the generalized angle $\Theta_3(t)$ with the pitch angle $\alpha(t)$, eq. (3.1) . Except vibrational displacements, both must be equal.

We must not use thes verifications as an evidence for correct results. But we can use them as an indication, whether the numerical results make sense or not.

Static displacements of the astro mast:

To obtain the static displacements, we consider the mast dynamic model and the antenna dynamic model as separated systems with external forces.

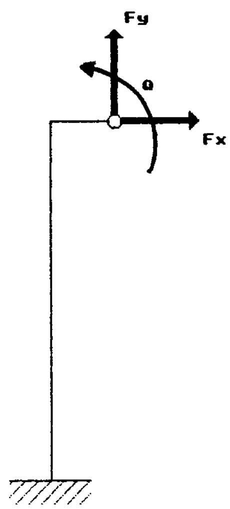


Fig. 3.22: Astro-mast model with external forces at the gimbal.

We consider the 2 dimensional mast model. In general we apply a horizontal and a vertical force, and a bending moment at the gimbal. The forces and the moments are caused by the pitch maneuver and we want to describe them as functions of time:

$$F_x = F_x(t)$$
$$F_y = F_y(t)$$
$$M = M(t)$$

The pitch maneuver is prescribed by the pitch angle $\alpha(t)$:

$$\begin{aligned}
\alpha(t) &= 1/2\,\ddot{\alpha}\,t^2 & 0 <= t <= \tau_1 \\
\alpha(t) &= \ddot{\alpha}(\tau_1^2 - 2\tau_1 t + 1/2 t^2) & \tau_1 < t <= \tau_2 \\
\\
\dot{\alpha}(t) &= \ddot{\alpha}\,t & 0 <= t <= \tau_1 \\
\dot{\alpha}(t) &= \ddot{\alpha}(-2\tau_1 + t) & \tau_1 < t <= \tau_2 \\
\\
\ddot{\alpha}(t) &= +M_0/I & 0 <= t <= \tau_1 \\
\ddot{\alpha}(t) &= -M_0/I & \tau_1 < t <= \tau_2
\end{aligned} \tag{3.48}$$

The forces at the gimbal are the sum of mass forces of the dish:

1. centrifugal force, when the satelite dish rotates. $|\dot{\alpha}| > 0$
2. Transversal force, when accelerating or decelerating the satellite dish. $|\ddot{\alpha}| > 0$

We determine the centrifugal force as a function of the angular velocity $\dot{\alpha}$:

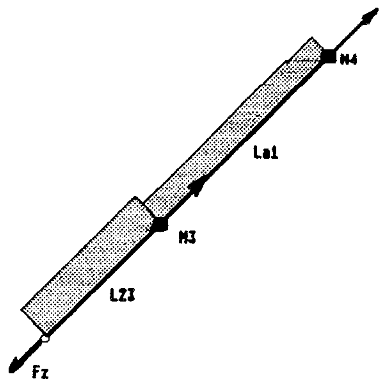


Fig. 3.23

$$F_z = \dot{\alpha}^2 \{L_{23}M_3 + (L_{23} + L_{a1})M_4\} \tag{3.49}$$

And the transversal force as a function of the angular accelleration $\ddot{\alpha}$:

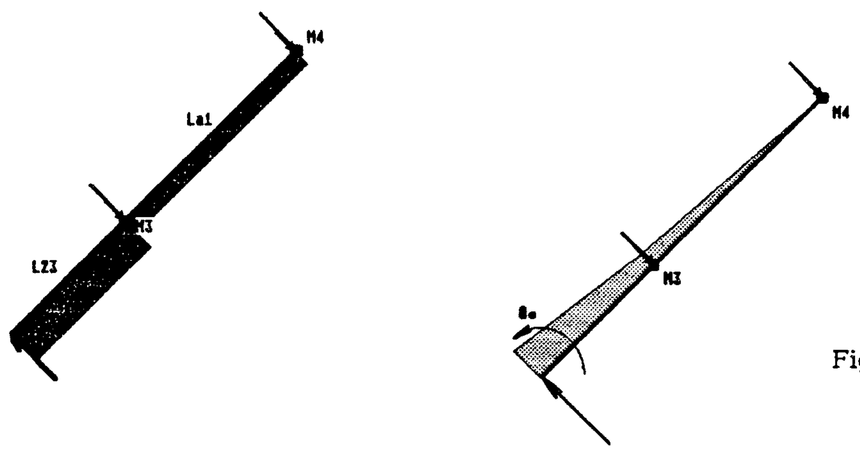


Fig. 3.24

$$F_R = \ddot{\alpha} \{L_{23}M_3 + (L_{23} + L_{a1})M_4\} \tag{3.50}$$

To obtain the horizontal and vertical gimbal forces, we use the coordinate transformation:

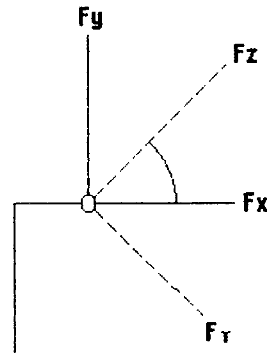


Fig. 3.25

$$F_x = F_z \cos\alpha + F_R \sin\alpha$$
$$F_y = F_z \sin\alpha - F_R \cos\alpha \qquad (3.51)$$

To obtain the gimbal forces as a function of time, we eliminate α, $\dot{\alpha}$, $\ddot{\alpha}$ in eqs. (3.49) and (3.50) with eq. (3.48) and introduce them into eq. (3.51) :

For $0 <= t <= \tau_1$:

$$F_x(t) = \{(M_o/I)\,t\}^2 \{L_{23}M_3 + (L_{23} + L_{a1})M_4\} \cos(M_o/2I\ t^2) + \qquad (3.52)$$
$$+ \; M_o/I \{L_{23}M_3 + (L_{23} + L_{a1})M_4\} \sin(M_o/2I\ t^2)$$

$$F_y(t) = \{(M_o/I)\,t\}^2 \{L_{23}M_3 + (L_{23} + L_{a1})M_4\} \sin(M_o/2I\ t^2) \qquad (3.53)$$
$$+ \; M_o/I \{L_{23}M_3 + (L_{23} + L_{a1})M_4\} \cos(M_o/2I\ t^2)$$

$$M(t) = -M_o \qquad (3.54)$$

For $\tau_1 < t <= \tau_2$:

$$F_x(t) = \{M_o/I\,(2\tau_1 - t)\}^2 \{L_{23}M_3 + (L_{23} + L_{a1})M_4\} \cos(M_o/I(-\tau_1^2 + 2\tau_1 t - 1/2t^2)) + \qquad (3.52)$$
$$- \; M_o/I \{L_{23}M_3 + (L_{23} + L_{a1})M_4\} \sin(M_o/I(-\tau + 2\tau_1 t - 1/2t^2))$$

$$F_y(t) = \{M_o/I\,(2\tau_1 - t)\}^2 \{L_{23}M_3 + (L_{23} + L_{a1})M_4\} \sin(M_o/I(-\tau_1^2 + 2\tau_1 t - 1/2t^2)) \qquad (3.53)$$
$$- \; M_o/I \{L_{23}M_3 + (L_{23} + L_{a1})M_4\} \cos(M_o/I(-\tau_1^2 + 2\tau_1 t - 1/2t^2))$$

$$M(t) = +M_o \qquad (3.54)$$

$t > \tau_2$:

$$F_x(t) = 0 \tag{3.52}$$

$$F_y(t) = 0 \tag{3.53}$$

$$M(t) = 0 \tag{3.54}$$

We apply eqs. (3.52) - (3.54) on the pitch maneuver:

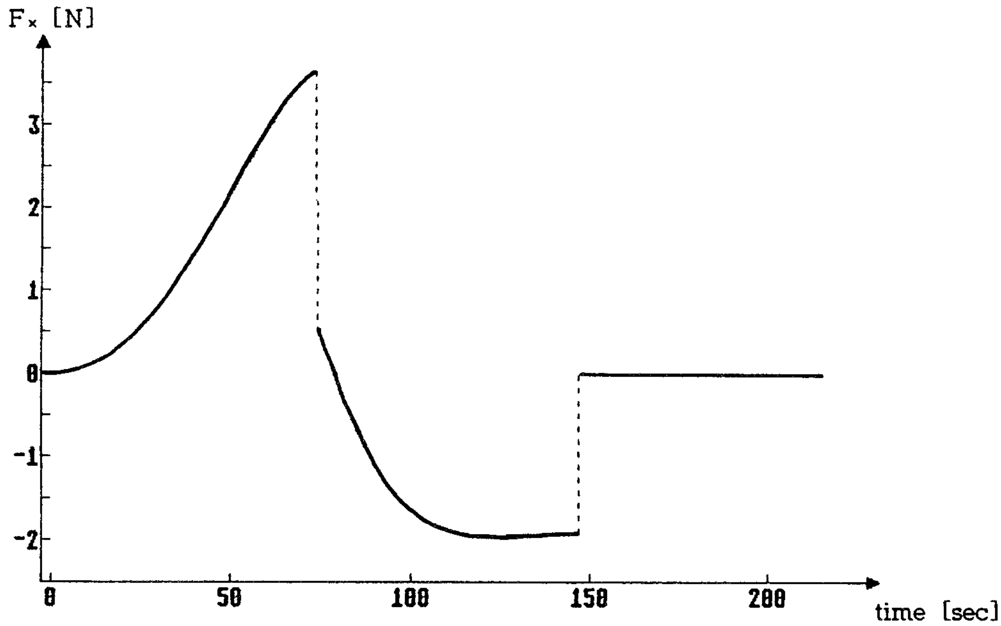


Fig. 3.26: horizontal gimbal force F_x

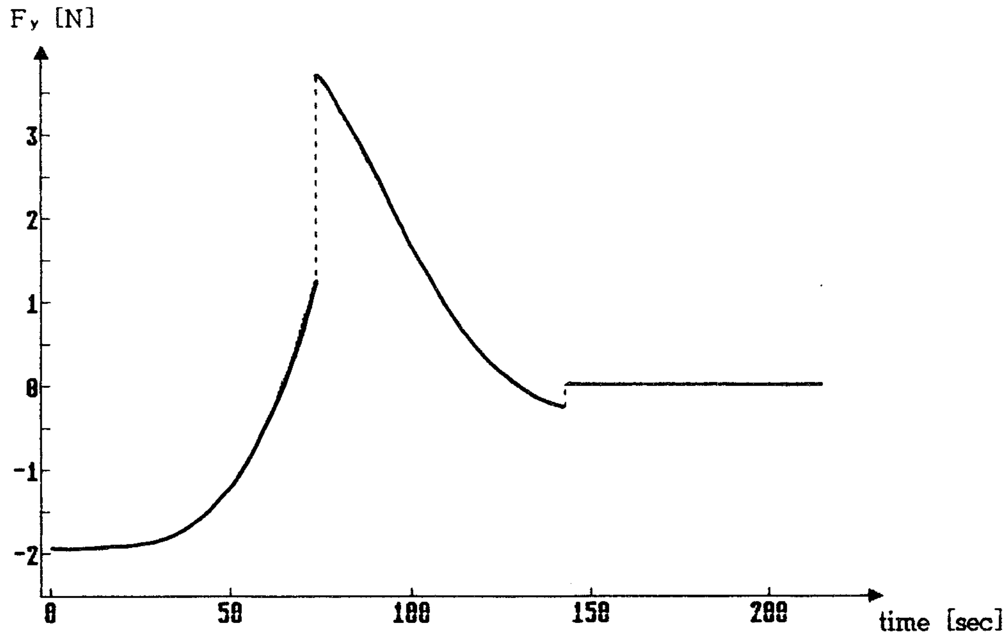


Fig. 3.27: Vertical gimbal force F_y

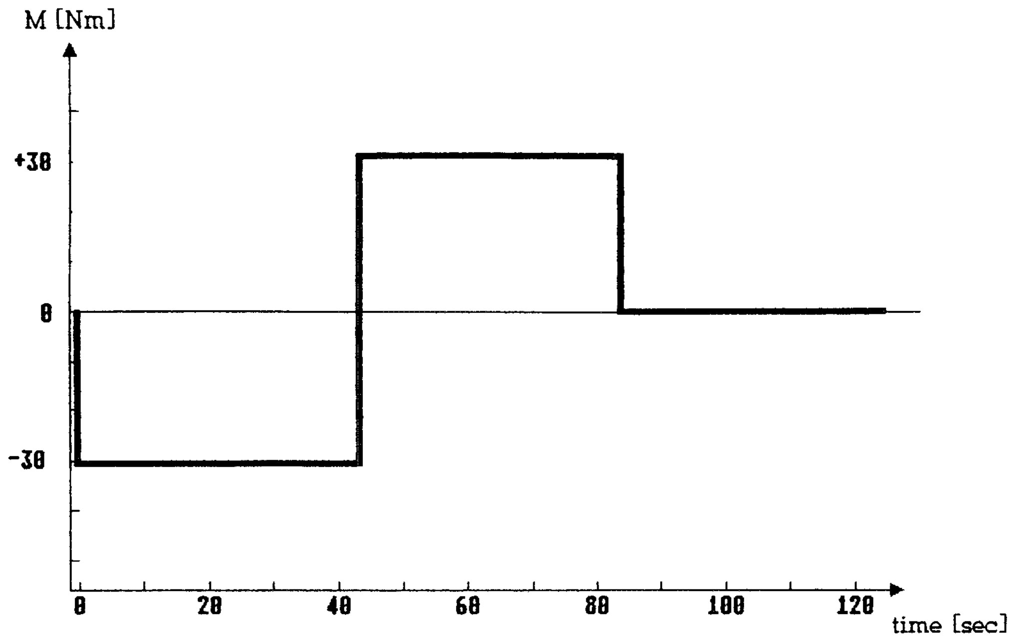


Fig. 3.28: Torque M at the gimbal

We determine the torques M_1, M_2 at the torsion springs λ_1, λ_2:

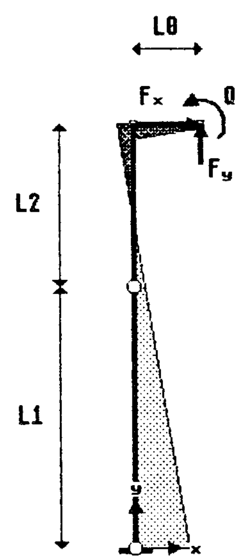


Fig. 3.29

$$M_1 = -F_x(L_1 + L_2) + F_yL_0 + M \tag{3.55}$$

$$M_2 = -F_xL_2 + F_yL_0 + M \tag{3.56}$$

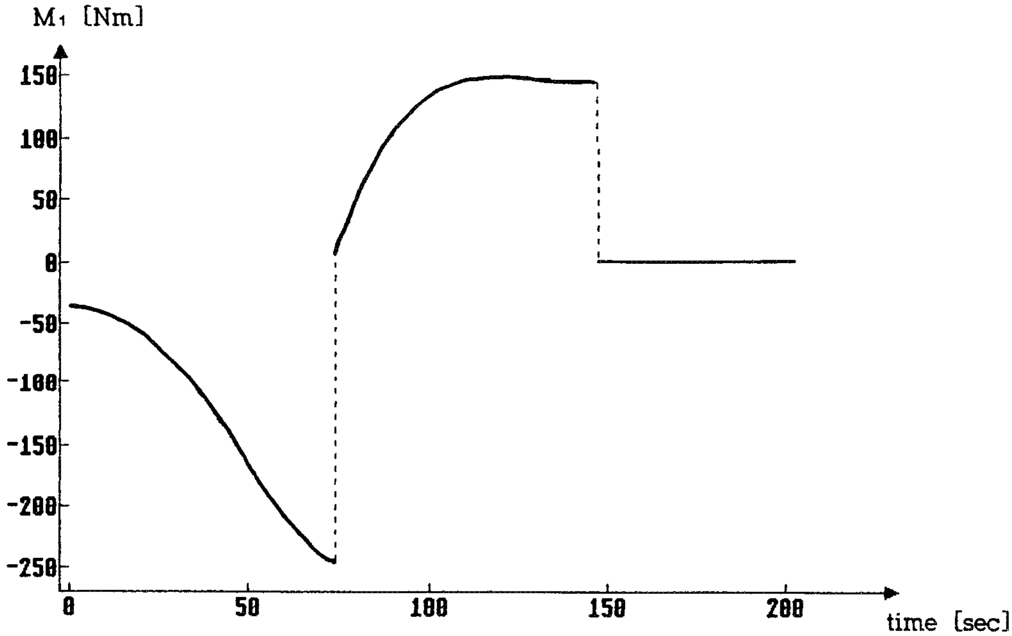


Fig. 3.30: Torque M_1 at torsion spring λ_1

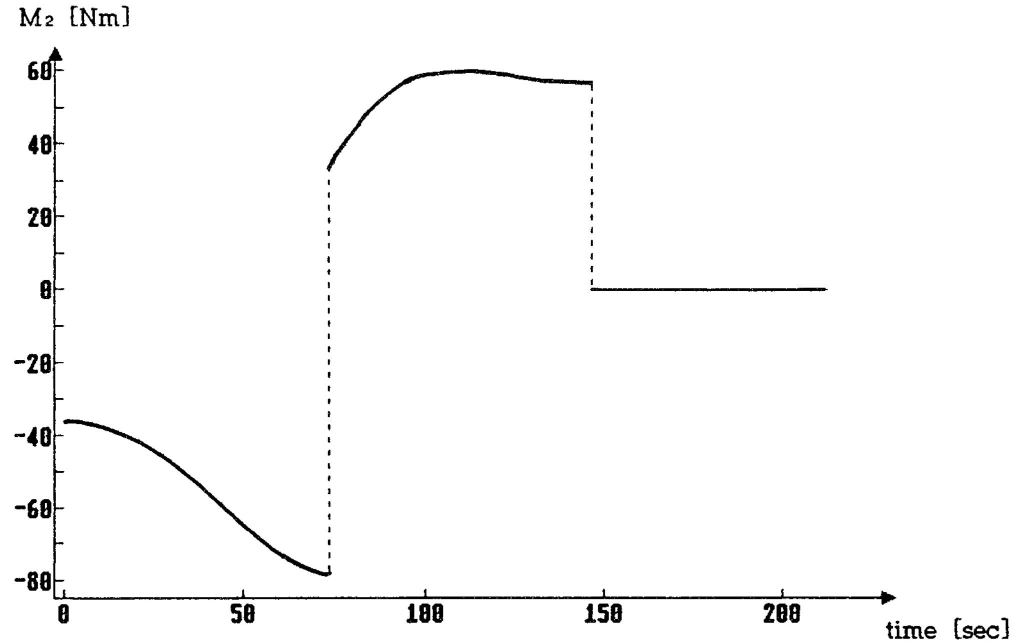


Fig. 3.31: Torque M_2 at the torsion spring λ_2

It is remarkable, that the bending moment at the base of the antenna mast is about 8 times greater than the torque of the server motor. Thus we see, that the displacements are mainly caused by centrifugal forces and reaction forces of the pitch maneuver. We obtain the static displacement as:

$$\Theta_1 = M_1/\lambda_1 \tag{3.57}$$

$$\Theta_2 = M_1/\lambda_1 + M_2/\lambda_2 \tag{3.58}$$

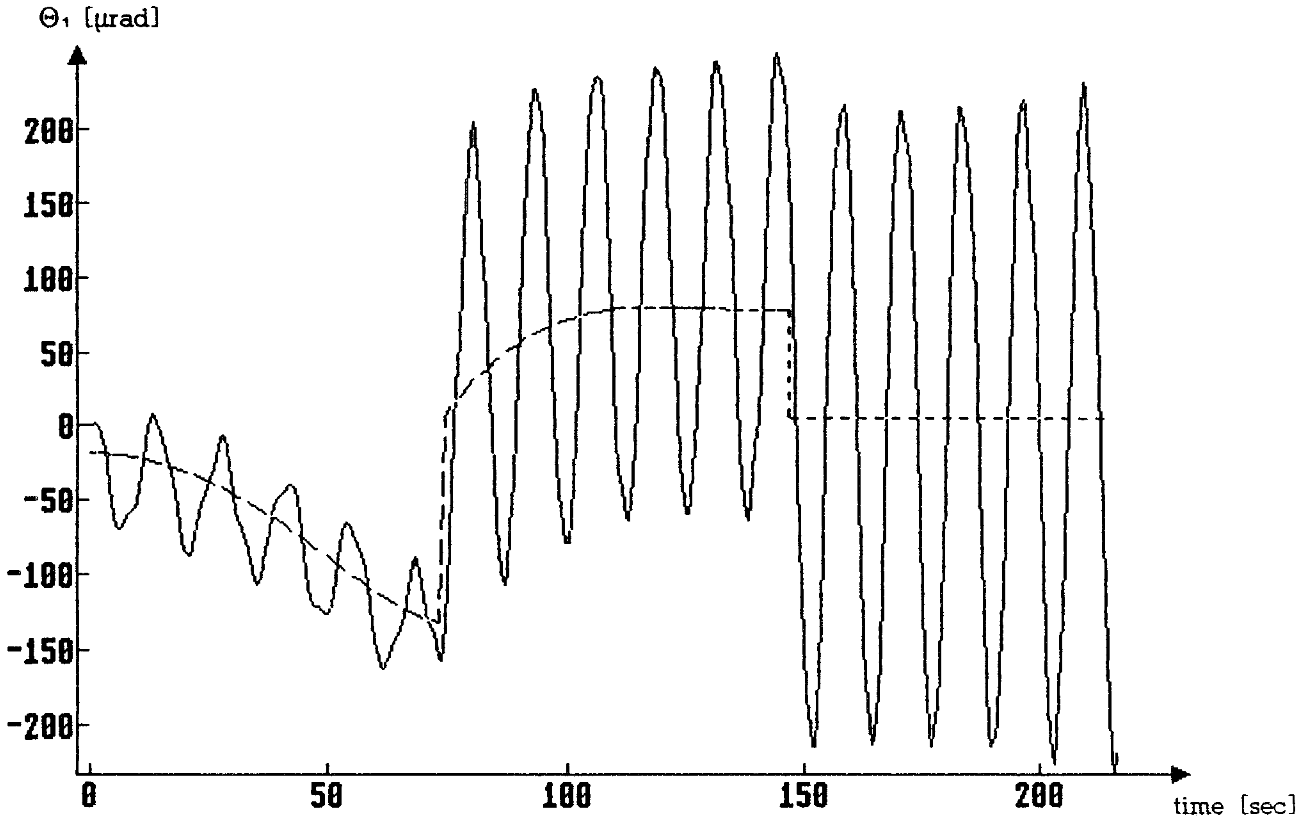


Fig. 3.32: ——— dynamic displacement of Θ_1

– – – – static displacement of Θ_1

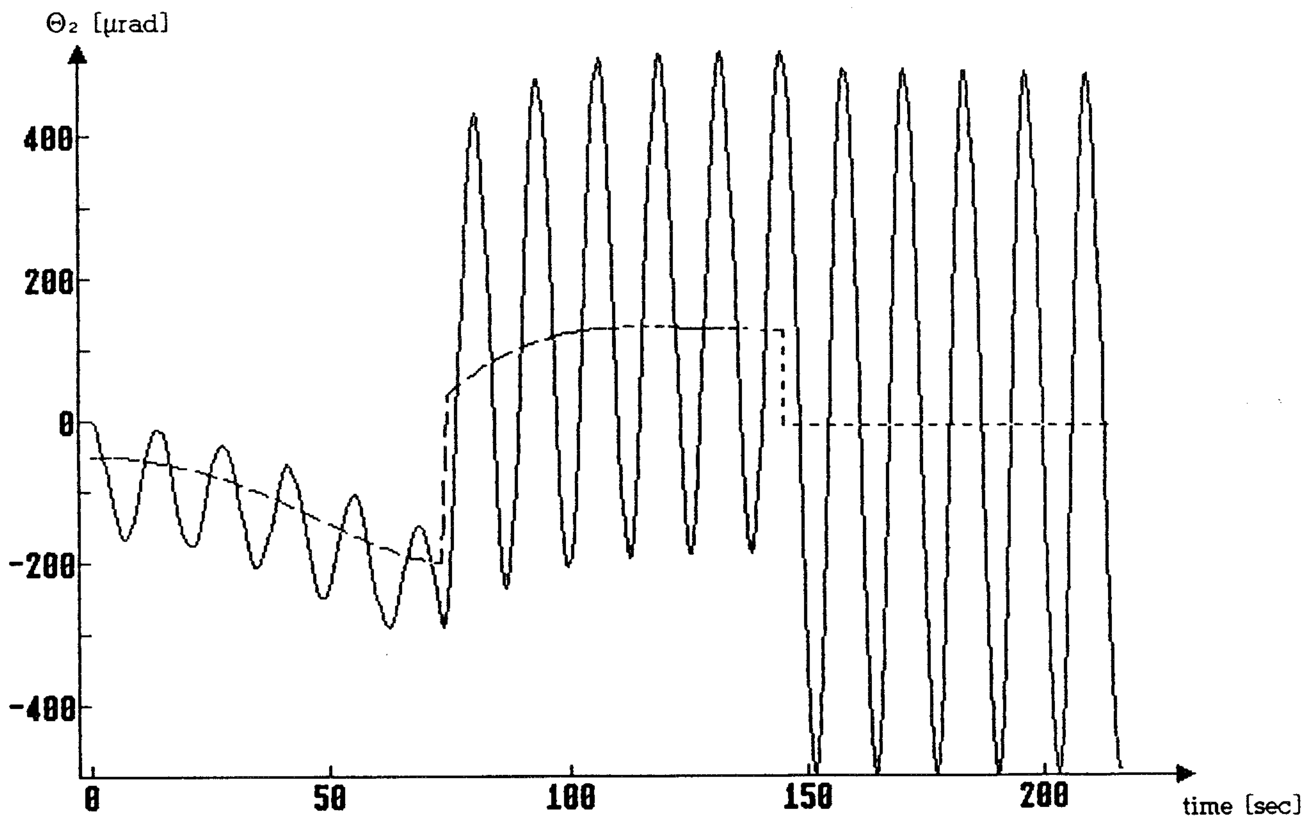


Fig. 3.33: ——— dynamic displacement of Θ_2

– – – – static displacement of Θ_2

Figs. 3.32 and 3.31 confirm the numerical results! We recognize also an advantage of the Langragian approach: We obtained the dynamic displacements without concerning about any internal forces.

Now we want to calculate the static displacement of Θ_4, which is a linear function of the torque M_4 at the torsion spring λ_4:

$$\Theta_{4,static} = M_4 / \lambda_4 \tag{3.59}$$

For constant acceleration $\ddot{\alpha}$, we determine the torque M_4 to:

$$M_4 = \ddot{\alpha}(L_{a1} + L_{23}) M_4 L_{a1} \tag{3.60}$$

Intruducing eq. (3.60) into eq. (3.59) and eliminating $\ddot{\alpha}$ by eq. (3.48) yields to the static displacement of Θ_4, as a function of time:

$$\begin{aligned} \Theta_{4,static}(t) &= M_0(L_{a1} + L_{23}) M_4 L_{a1} / I \lambda_4 && 0 <= t <= \tau_1 \\ \Theta_{4,static}(t) &= -M_0(L_{a1} + L_{23}) M_4 L_{a1} / I \lambda_4 && \tau_1 < t <= \tau_2 \\ \Theta_{4,static}(t) &= 0 && t > \tau_2 \end{aligned} \tag{3.61}$$

Applying eq.(3.61) to the particular pitch maneuver, we calculate $\Theta_{4,static}$ = +/- 179.4 μrad. We compare the static displacement with the dynamic displacement, which confirms the numerical results:

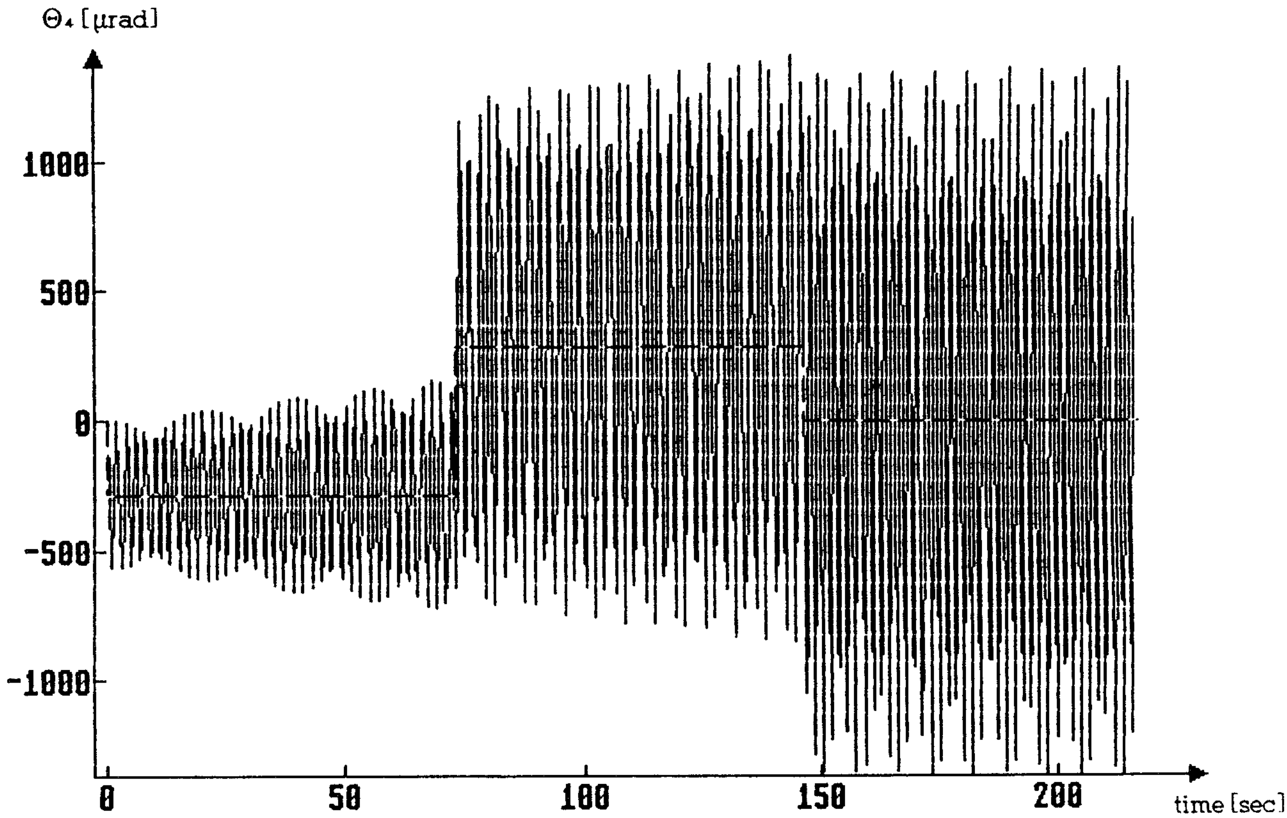


Fig. 3.34: ———— dynamic displacement Θ_4

- - - - - - - - - - static displacement $\Theta_{4,static}$

Total energy:

The total energy is an analytical function of angular displacements and angular velocities (eq. (3.42)). In the specific case of no structural damping and no external forces, the total energy must remain constant. We can use this fact to verify the numerical results for $t > \tau_2$. We define a relative error:

$$\varepsilon_{rel} = \text{Maximum} \left(TE_i / TE_1 \right) \qquad i = 1, 2, \ldots N \qquad (3.62)$$

TE_i are discrete values of the total energy of the system for $\tau_2 < t <= \tau_2 + 42$ sec.
TE_1 is the total energy of the system at $t = \tau_2$. (τ_2 : end of the pitch maneuver).
N is the number of calculated discrete values.

In the first iteration, the numerical results had a relative error of $\varepsilon_{rel} = 3\%$. Increasing the accuracy of the iteration and using double precision variables, lowererd ε_{rel} to 1%. We see that the error is a result of numerical inaccuracy and increasing the accuracy yields an improved approximation to theoretical values.

It is interesting, that we can improve the accuracy significantly, when we put a torsion spring at the gimbal, after the dish has reached its desired position. In practical, this can be realized by blocking the gymbal, after the pitch maneuver has finished. Even a small torsion spring constant at the gymbal, lowers the the relative error to $\varepsilon_{rel} < 1E-6$. This is an excellent confirmation of the numerical result! Obviously, it leads to a poor convergence of the iteration, if there is no constraint at the gimbal, which keeps the satellite dish in the desired position.

Pitch angle:

Another easy opportunity to verify the numerical results, is to compare the time histories of the generalized angle Θ_3 and the pitch angle α, where $\alpha(t)$ is a analytical quadratic function eq. (3.1) . Except small vibration amplitudes, both angles must be equal.

3.7 EVALUATION OF RESULTS

Magnitudes of displacements:

We want to verify, if our assumption of small angles, and therefore the use of linearized equations is valid. The maximum amplitude of the small angles Θ_1, Θ_2, Θ_4 is approximately 400 µrad = 0.023°, which indicates that the of linearized equations are a very good approximation. To get an idea of the displacements, we calculate the maximum horizontal displacement of the gymbal:

$$Y = L_1\Theta_1 + L_2\Theta_2$$

$$Y_{max} = 2.2 \text{ cm}$$

Which is very little, compared to the mast length of 60.6 m .

Large angle Θ_3, without any constraints:

At the hinge of Θ_3 is no torsion spring, which constraints the angle Θ_3 to a determined position. Thus we can see 2 effects:

1. $(\Theta_3 - \alpha)$ is relatively large, compared to Θ_1, Θ_2, Θ_4.

2. After the pitch maneuver has terminated, Θ_3 does not remain constant.
   The reason for this behaviour is, that it is not possible to deccelerate absolut precisly to zero angular velocity $\dot{\alpha}$. Since we neglected friction, the motion $\dot{\alpha}$ continues.

Total vibrational displacement of satellite dish:

For our applications it is interesting to know the total vibrational displacement of the antenna dish. This is simply the sum of the vibrational angles:

$$\Theta_2 + (\Theta_3 - \alpha) + \Theta_4$$

The amplitude is approximately 2E-3 rad = 0.115° .

Total vibrational energy:

The total vibrational energy has a maximum value of aproximately 0.13 Nm.
This value looks quite small. It is an effect of the very small vibrational displacements.

Effects of the step function of servo motor torque:

If we consider the vibrational displacements, we recognize a significant increase, after the servo motor torque is switched from acceleration to deceleration. Unfortunately this maneuver feeds maximum energy into the system, because both directions of the motor torque and the angular velocities $\dot{\Theta}_1, \dot{\Theta}_2$, are the same. For this reason it might be useful to use a shaped function for the motor torque.

A more interesting alternative would be, to switch off the servo motor after accelerating and to switch it on for deceleration, after a half period of the first vibrational mode. In that case, motor torque and angular velocities would have opposite directions. This means, vibrational energy would be deductet from the system.

4. OPEN LOOP EIGENVALUE ANALYSIS

4.1 INTRODUCTION:

During the pitch maneuver the equations of motion are nonlinear, the system is open loop, and eigenvalues do not exist due to the nonlinear and time varying nature of the system. In this section we examine the natural frequencies and mode shapes at the end of the maneuver for small perturbations in order to obtain qualitative information on the system dynamics.

In particular, we will investigate the characteristics of vibrations for constant pitch angle $\alpha = 90°$, which is the final position after the pitch maneuver of section (3) has finished. For constant $\alpha = 90°$, the equations of motion can be linearized as a very good approximation.

We will apply eigenvalue analysis on the system. Results of eigenvalues are displayed in the complex plane and mode shapes are obtained from the eigenvectors.

It is little effort to investigate particular modifications of the system. We will compare 3 different configurations:

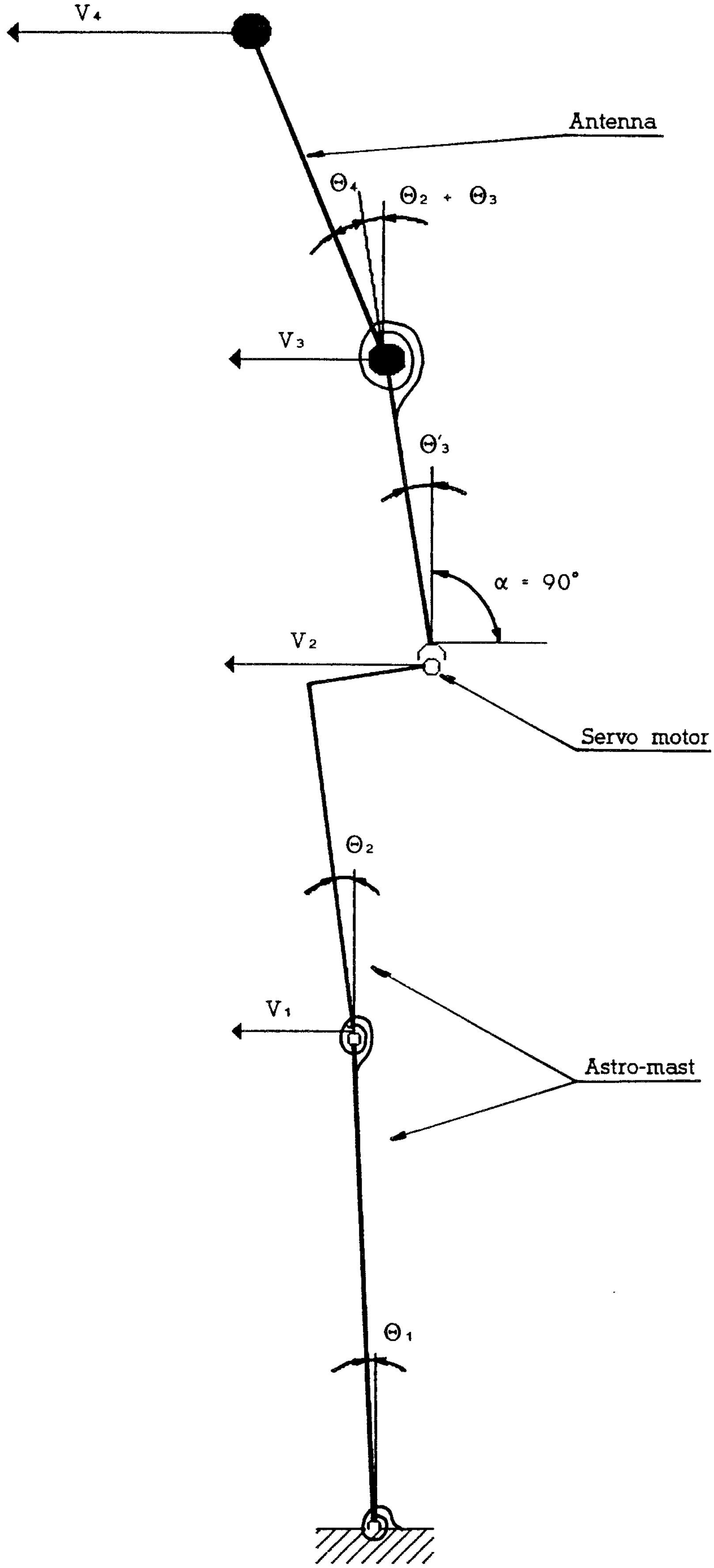
1. No constraint at the gimbal hinge.
2. Gimbal hinge blocked.
3. Servo motor on short circuit.

The first case simulates an uncontrolled system, where the satellite dish can rotate freely.

The second case means, that the gimbal hinge is blocked by a brake, after the desired pitch angle is achieved.

The third case simulates open loop damping by servo motor, which means that the servo motor runs on short circuit and dissipates vibrational system energy. In section (5) , we consider closed loop damping. Open loop damping might be used as emergency run, if the closed loop electronics have a male function.

4.2 LINEARIZED EQUATION OF MOTION



Dynamic model in the final position, after the pitch maneuver has finished.

Total kinetic energy:

The total kinetic energy for arbitrary pitch angles was found to be

$$
\begin{aligned}
T/mL^3 = \; & C_1 \dot{\Theta}_1^2 \; + \\
& + \{ C_2 + C_3 \sin(\Theta_2 + \Theta_3) + C_4 \sin(\Theta_2 + \Theta_3 + \Theta_4) \} \dot{\Theta}_2^2 \; + \\
& + C_5 \dot{\Theta}_3^2 \; + \\
& + C_6 \dot{\Theta}_4^2 \; + \\
& + \{ C_7 + C_8 \sin(\Theta_2 + \Theta_3) + C_9 \sin(\Theta_2 + \Theta_3 + \Theta_4) \} \dot{\Theta}_1 \dot{\Theta}_2 \; + \\
& + \{ C_{10} \sin(\Theta_2 + \Theta_3) + C_{11} \sin(\Theta_2 + \Theta_3 + \Theta_4) \} \dot{\Theta}_1 \dot{\Theta}_3 \; + \\
& + \{ C_{12} \sin(\Theta_2 + \Theta_3 + \Theta_4) \} \dot{\Theta}_1 \dot{\Theta}_4 \; + \\
& + \{ C_{13} + C_{14} \sin(\Theta_2 + \Theta_3) + C_{15} \sin(\Theta_2 + \Theta_3 + \Theta_4) \} \dot{\Theta}_2 \dot{\Theta}_3 \; + \\
& + \{ C_{16} + C_{17} \sin(\Theta_2 + \Theta_3 + \Theta_4) \} \dot{\Theta}_2 \dot{\Theta}_4 \; + \\
& + C_{18} \dot{\Theta}_3 \dot{\Theta}_4
\end{aligned}
\tag{3.14}
$$

To linearize this equation, we replace the generalized angle Θ_3:

$$\Theta_3 = \Theta'_3 + \alpha \tag{4.1}$$

Where Θ'_3 is the new generalized coordinate, which denotes small angular displacements caused by structural vibrations, and α denotes the constant pitch angle.

$$\sin(\Theta_2 + \Theta_3) = \sin(\Theta_2 + \Theta'_3 + \alpha) \tag{4.2}$$

$$\sin(\Theta_2 + \Theta_3 + \Theta_4) = \sin(\Theta_2 + \Theta'_3 + \Theta_4 + \alpha) \tag{4.3}$$

where we consider a final position $\alpha = 90°$.

Typically values for vibrational displacements are: Θ_2 , Θ'_3 , $\Theta_4 \approx$ 1000 μrad = 0.075°. Thus Θ_2, Θ'_3, $\Theta_4 \ll \alpha$, and we can neglect the vibrational displacements in the arguments of the sine functions, which results in:

$$\sin(\Theta_2 + \Theta'_3 + \alpha) \approx 1 \tag{4.4}$$

for $\alpha = 90°$ and $\Theta_2, \Theta'_3, \Theta_4 \ll \alpha$

$$\sin(\Theta_2 + \Theta'_3 + \Theta_4 + \alpha) \approx 1 \tag{4.5}$$

This simplifies the equation of motion significantly.

The total kinetic energy for constant pitch angle $\alpha = 90^\circ$ can be written as:

$$
\begin{aligned}
T/mL^3 \approx\ & C_1\dot{\Theta}_1^2 + \\
& + (C_2 + C_3 + C_4)\,\dot{\Theta}_2^2 + \\
& + C_5\,\dot{\Theta}_3'^2 + \\
& + C_6\,\dot{\Theta}_4^2 + \\
& + (C_7 + C_8 + C_9)\,\dot{\Theta}_1\dot{\Theta}_2 + \\
& + (C_{10} + C_{11})\,\dot{\Theta}_1\dot{\Theta}_3' + \\
& + C_{12}\,\dot{\Theta}_1\dot{\Theta}_4 + \\
& + (C_{13} + C_{14} + C_{15})\,\dot{\Theta}_2\dot{\Theta}_3' + \\
& + (C_{16} + C_{17})\,\dot{\Theta}_2\dot{\Theta}_4 + \\
& + C_{18}\,\dot{\Theta}_3'\dot{\Theta}_4
\end{aligned}
\qquad (4.6)
$$

Where the constants C_i are denoted by eqs. (3.15.1) - (3.15.18).

Strain energy:

The total strain energy is denoted by eq. (3.18). We also want to investigate the characteristics of the system, if the gimbal hinge is blocked. This can be modelled by a torsion spring λ_3 at the hinge. Hence, the total strain energy is:

$$V/mL^3 = 1/2\left[\Lambda_1\Theta_1^2 + \Lambda_2(\Theta_2 - \Theta_1)^2 + \Lambda_3\Theta_3'^2 + \Lambda_4\Theta_4^2\right] \qquad (4.7)$$

Where:

$\Lambda_3 = 0$: Original system.

$\Lambda_3 > 0$: Blocked gimbal hinge.

Generalized exciting forces:

After the pitching maneuver has finished, we do not consider any exciting forces.

Generalized damping forces:

We obtain a simple open loop damping, if the servo motor works on short circuit. Vibrational system energy will be transferred into electrical energy, which will be converted into heat. The motor torque can be assumed to be proportional to the angular velocity $\dot{\Theta}'_3$, which implies viscous damping. The corresponding dissipative function is:

$$F = 1/2\, D\, \dot{\Theta}'^{2}_3 \tag{4.8}$$

Where D is a real constant. $D = 0$ means no damping and $D > 0$ means positive damping. Thus we obtain the generalized damping force:

$$\partial F/\partial \dot{\Theta}'_3 = D\dot{\Theta}'_3 \tag{4.9}$$

Partial derivatives:

Analogous to the third section, we evaluate the partial derivatives of the total kinetic energy and the total strain energy.

The partial derivatives of the total kinetic energy are:

$$d/dt\,(\partial T/\partial \dot{\Theta}_1) = 2C_1\ddot{\Theta}_1 + (C_7 + C_8 + C_9)\ddot{\Theta}_2 + (C_{10} + C_{11})\ddot{\Theta}'_3 + C_{12}\ddot{\Theta}_4 \tag{4.10.1}$$

$$d/dt\,(\partial T/\partial \dot{\Theta}_2) = (C_7 + C_8 + C_9)\ddot{\Theta}_1 + 2(C_2 + C_3 + C_4)\ddot{\Theta}_2 + (C_{13} + C_{14} + C_{15})\ddot{\Theta}'_3 + (C_{16} + C_{17})\ddot{\Theta}_4 \tag{4.10.2}$$

$$d/dt\,(\partial T/\partial \dot{\Theta}'_3) = (C_{10} + C_{11})\ddot{\Theta}_1 + (C_{13} + C_{14} + C_{15})\ddot{\Theta}_2 + 2C_5\ddot{\Theta}'_3 + C_{18}\ddot{\Theta}_4 \tag{4.10.3}$$

$$d/dt\,(\partial T/\partial \dot{\Theta}_4) = C_{12}\ddot{\Theta}_1 + (C_{16} + C_{17})\ddot{\Theta}_2 + C_{18}\ddot{\Theta}'_3 + 2C_6\ddot{\Theta}_4 \tag{4.10.4}$$

$$\partial T/\partial \Theta_1 = \partial T/\partial \Theta_2 = \partial T/\partial \Theta'_3 = \partial T/\partial \Theta_4 = 0 \tag{4.11}$$

The partial derivatives of the total strain energy are:

$$\partial V/\partial\Theta_1 = (\Lambda_1 + \Lambda_2)\Theta_1 - \Lambda_2\Theta_2 \qquad (4.12.1)$$

$$\partial V/\partial\Theta_2 = -\Lambda_2\Theta_1 + \Lambda_2\Theta_2 \qquad (4.12.2)$$

$$\partial V/\partial\Theta'_3 = \Lambda_3\Theta'_3 \qquad (4.12.3)$$

$$\partial V/\partial\Theta_4 = \Lambda_4\Theta_4 \qquad (4.12.4)$$

System of differential equations:

The system of Lagrange differential equations for linea conservative problems is denoted by:

$$d/dt(\partial T/\partial\dot{\Theta}_i) + \partial V/\partial\Theta_i + \partial F/\partial\dot{\Theta}_i = 0 \qquad i = 1, 2, \ldots 4 \qquad (4.13)$$

Or in matrix notation:

$$\mathbf{M}\ddot{\boldsymbol{\Theta}} + \mathbf{C}\dot{\boldsymbol{\Theta}} + \mathbf{K}\boldsymbol{\Theta} = \mathbf{0} \qquad (4.14)$$

Where **M**, **C**, **K** are constant 4 x 4 system matrices.

Introducing the partial derivatives (4.9) - (4.12) into eqs. (4.13) and ordering to the configuration vectors, leads to the constant system matrices.

MASS MATRIX :

$$\mathbf{M} = \begin{bmatrix} 2C_1 & C_7 + C_8 + C_9 & C_{10} + C_{11} & C_{12} \\ & 2(C_2 + C_3 + C_4) & C_{13} + C_{14} + C_{15} & C_{16} + C_{17} \\ \text{symmetric} & & 2C_5 & C_{18} \\ & & & 2C_6 \end{bmatrix} \tag{4.15}$$

STIFFNESS MATRIX:

$$\mathbf{K} = \begin{bmatrix} \Lambda_1 + \Lambda_2 & -\Lambda_2 & 0 & 0 \\ & \Lambda_2 & 0 & 0 \\ \text{symmetric} & & \Lambda_3 & 0 \\ & & & \Lambda_4 \end{bmatrix} \tag{4.16}$$

STRUCTURAL DAMPING MATRIX:

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ & 0 & 0 & 0 \\ \text{symmetric} & & D & 0 \\ & & & 0 \end{bmatrix} \tag{4.17}$$

4.3 EIGENVALUE ANALYSIS:

We transform the system of second order differential equations (4.14) into a system of first order differential equations. First, we extend the system of second order differential equations:

$$\begin{aligned} \mathbf{M}\dot{\boldsymbol{\theta}} - \mathbf{M}\dot{\boldsymbol{\theta}} &= 0 \\ \mathbf{M}\ddot{\boldsymbol{\theta}} + \mathbf{C}\dot{\boldsymbol{\theta}} + \mathbf{K}\boldsymbol{\theta} &= 0 \end{aligned} \qquad (4.18)$$

The reader can verify easily, that equation system (4.18) is equal to:

$$\begin{bmatrix} \mathbf{M} & 0 \\ 0 & \mathbf{M} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{\theta}} \\ \ddot{\boldsymbol{\theta}} \end{bmatrix} = \begin{bmatrix} 0 & \mathbf{M} \\ -\mathbf{K} & -\mathbf{C} \end{bmatrix} \begin{bmatrix} \boldsymbol{\theta} \\ \dot{\boldsymbol{\theta}} \end{bmatrix} \qquad (4.19)$$

or:

$$\mathbf{A}\,\dot{\mathbf{z}} = \mathbf{B}\,\mathbf{z}$$

Where we replace the vectors and matrices by new variables **A**, **B**, **z** :

$$\mathbf{A}\dot{\mathbf{z}} = \mathbf{B}\mathbf{z} \qquad (4.19)$$

Eq.(4.19) is a system of first order differential equations.

z is the new (8 x 1) configuration vector and **A**,**B** are the corresponding (8 x 8) system matrices:

$$\mathbf{z} = \begin{bmatrix} \boldsymbol{\theta} \\ \dot{\boldsymbol{\theta}} \end{bmatrix} \qquad \mathbf{A} = \begin{bmatrix} \mathbf{M} & 0 \\ 0 & \mathbf{M} \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} 0 & \mathbf{M} \\ -\mathbf{K} & -\mathbf{C} \end{bmatrix} \qquad (4.20)$$

Applying the Euler approach, a general solution of this system of first order differential equations is:

$$\mathbf{z}(t) = \mathbf{c}\,e^{\lambda t} \qquad (4.21.1)$$

$$\begin{aligned} \dot{\mathbf{z}}(t) &= \lambda \mathbf{c}\,e^{\lambda t} \\ &= \lambda \mathbf{z}(t) \end{aligned} \qquad (4.21.2)$$

Where λ is a complex eigenvalue, and $\mathbf{c}$ is a (8 x 1) vector of real constants. Introducing eq. (4.21.2) into the system of differential equations:

$$\lambda \mathbf{A z} = \mathbf{B z} \tag{4.22}$$

Yields to the system of charcteristic equations of the particular eigenvalue problem:

$$(\mathbf{A}^{-1}\mathbf{B} - \lambda_K \mathbf{E})\, \mathbf{z} = \mathbf{0} \tag{4.23}$$

Since we seek nontrivial solutions of eq.(4.23), the coefficient determinat must be equal to zero. Thus, the eigenvalues λ_k are the roots of:

$$\det[\mathbf{A}^{-1}\mathbf{B} - \lambda_K \mathbf{E}] = \mathbf{0} \tag{4.24}$$

Where $\mathbf{E}$ = (8 x 8) unity matrix, and k = 1, 2, . . . 8 .
The corresponding eigenvectors $\mathbf{\Phi}_K$ are defined by:

$$(\mathbf{A}^{-1}\mathbf{B} - \lambda_k \mathbf{E})\mathbf{\Phi}_K = \mathbf{0} \tag{4.25}$$

Physical interpretation of eigenvalues and eigenvectors:

A solution of the first order differential equation is:

$$\mathbf{z} = \mathbf{c}\, e^{\lambda t} \tag{4.21.1}$$

We separate the real and the imaginary part of the eigenvalue:

$$\lambda = \sigma + i\omega$$

$$\mathbf{z} = \mathbf{c}\,(e^{\sigma t}\, e^{i\omega t})$$

$$= \mathbf{c} e^{\sigma t}(\cos \omega t + i \sin \omega t)$$

$$= \mathbf{c} e^{\sigma t} \cos(\omega t - \varphi) \tag{4.26}$$

From eq.(4.26), we see:

The imaginary part ω_K of the eigenvalue λ_K is equal to the (damped) natural frequency of the k-th mode in rad/sec.

The real part σ_K of the eigenvalue λ_K describes the damping of the k-th mode. Thus we have:

$\sigma = 0$: constant amplitudes = undamped vibration.
$\sigma < 0$: decreasing amplitudes = damped vibration.
$\sigma > 0$: increasing amplitudes = unstable system.

The eigenvector $\mathbf{\Phi}_K$ is a solution of the system of differential equations, corresponding to the eigenvalue λ_k. Since the (8x1) configuration vector is:

$$\mathbf{z} = \left[\mathbf{\Theta}^T \ \dot{\mathbf{\Theta}}^T \right]^T$$

The first 4 elements of the eigenvector denote the displacements Θ_i, and the last 4 elements denote the corresponding velocities $\dot{\Theta}_i$.

The k-th eigenvector describes the displacements and velocities of the k-th mode. All linear combinations of the eigenvectors are also a solution of the system of differential equations (modal analysis).

4.4 NUMERICAL RESULTS

4.4.1 Time histories of displacements

The time histories of the generalized coordinates Θ_i were studied for the first 5 minutes, after the pitching maneuver of section 3 has finished. We use the program "IVP-C", which is derived from the program "IVP", solving an initial value problem. Since the system matrices are constant, "IVP-C" evaluates and inverts the system matrices only once. This speeds up the computational time significantly. (In case of the pitching maneuver, it was necessary to evaluate the system matrices for each iteration step, because the geometry was continously changing during the maneuver).

In investigating the principal effects of slight modifications of the system, we consider 3 cases:

Case 1: No constraint at the gimbal hinge. The system is equal to the system of section 3.

Case 2: Gimbal hinge blocked by a brake, or accidential blocking, which is simulated by a torsion spring λ_3 at the gimbal hinge. As an example, we choose λ_3 equal to λ_4:

$$\lambda_3 = \lambda_4 = 7.1442E\ 4\ \text{Nm/rad}$$

Case 3: Servo motor on short circuit. This case also simulates friction at the gimbal hinge, caused by the gearing of the servo motor. As an example, we consider a maximum gimbal torque of 2.5 Nm, which requires a damping constant:

$$D = 2.0E\ 3\ \text{Nm/rad sec}$$

CASE 1 : $\lambda_3 = 0$, damping $= 0$

Diagram 4.1 : $\Theta_1(t)$

Diagram 4.2 : $\Theta_2(t)$

Diagram 4.3 : $\Theta'_3(t)$

Diagram 4.4 : $\Theta_4(t)$

CASE 2 : $\lambda_3 = \lambda_4$, damping $= 0$

Diagram 4.5 : $\Theta_1(t)$

Diagram 4.6 : $\Theta_2(t)$

Diagram 4.7 : $\Theta'_3(t)$

Diagram 4.8 : $\Theta_4(t)$

CASE 3: $\lambda_3 = 0$, damping > 0

Diagram 4.9 : $\Theta_1(t)$

Diagram 4.10 : $\Theta_2(t)$

Diagram 4.11 : $\Theta'_3(t)$

Diagram 4.12 : $\Theta_4(t)$

Diagram 4.13 : Total system energy.

Diagram 4.14 : Gymbal torque.

Evalution of results:

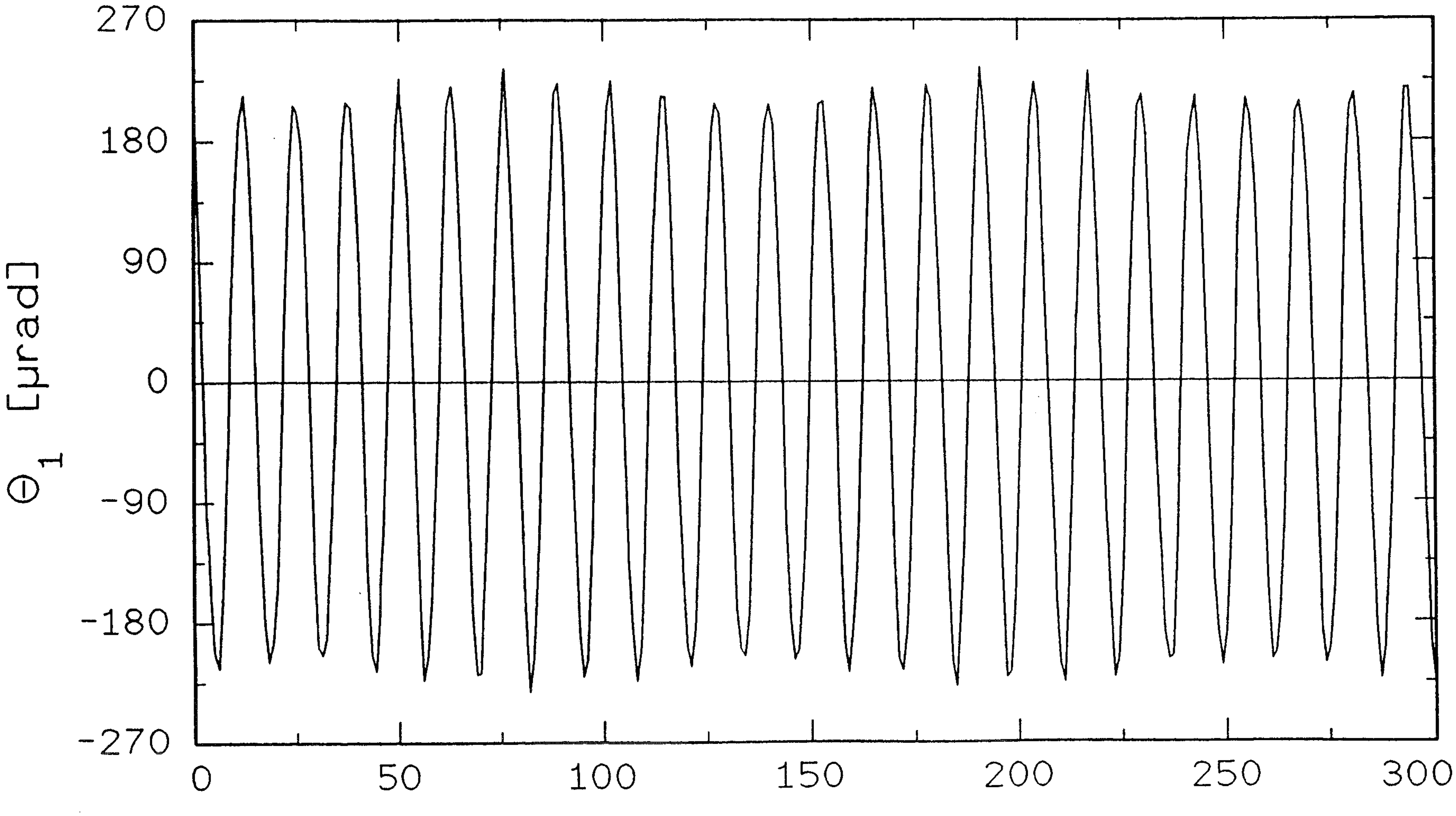
Case 1: The missing constraint at the gimbal hinge causes a free drift of the pitch angle.

Case 2: The blocked gimbal hinge has a great influence to the vibrational characteristics of the system.

Case 3: Friction at the gimbal hinge, damps out higher frequencies more effectively than lower frequencies. Because of the missing constraint, the pitch angle converges to a position, which is not necessarily the desired position.

DIAGRAM 4.1

$\lambda_3 = 0$, damping = 0



M. MOCH , C.I.T. / 5.1.1992

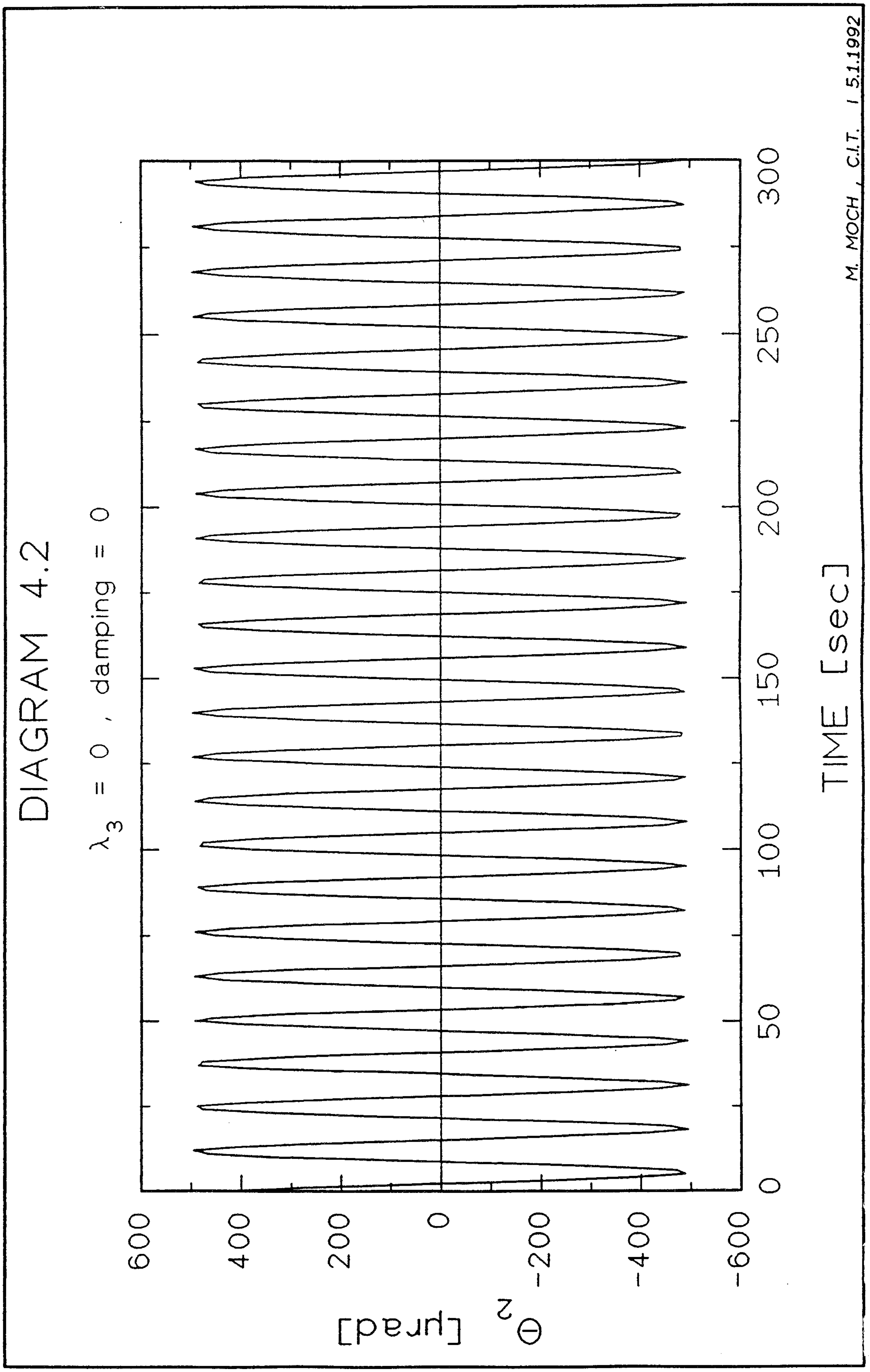


DIAGRAM 4.2
λ_3 = 0 , damping = 0
600
400
200
0
-200
-400
-600
Θ_2 [μrad]
0
50
100
150
200
250
300
TIME [sec]
M. MOCH , C.I.T. 15.1.1992

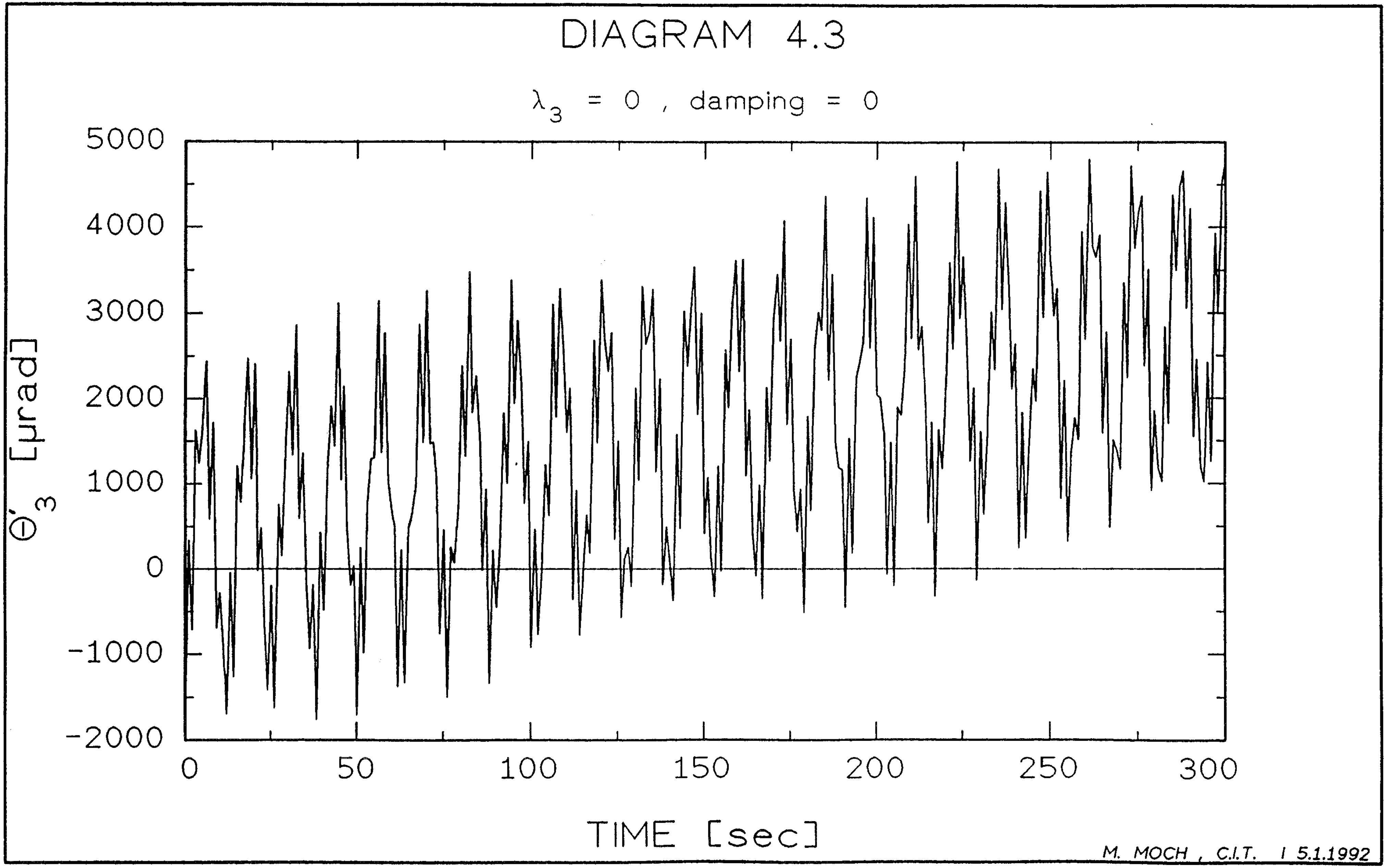


DIAGRAM 4.3
λ3 = 0 , damping = 0
5000
4000
3000
2000
1000
0
−1000
−2000
Θ'3 [μrad]
0
50
100
150
200
250
300
TIME [sec]
M. MOCH , C.I.T. / 5.1.1992

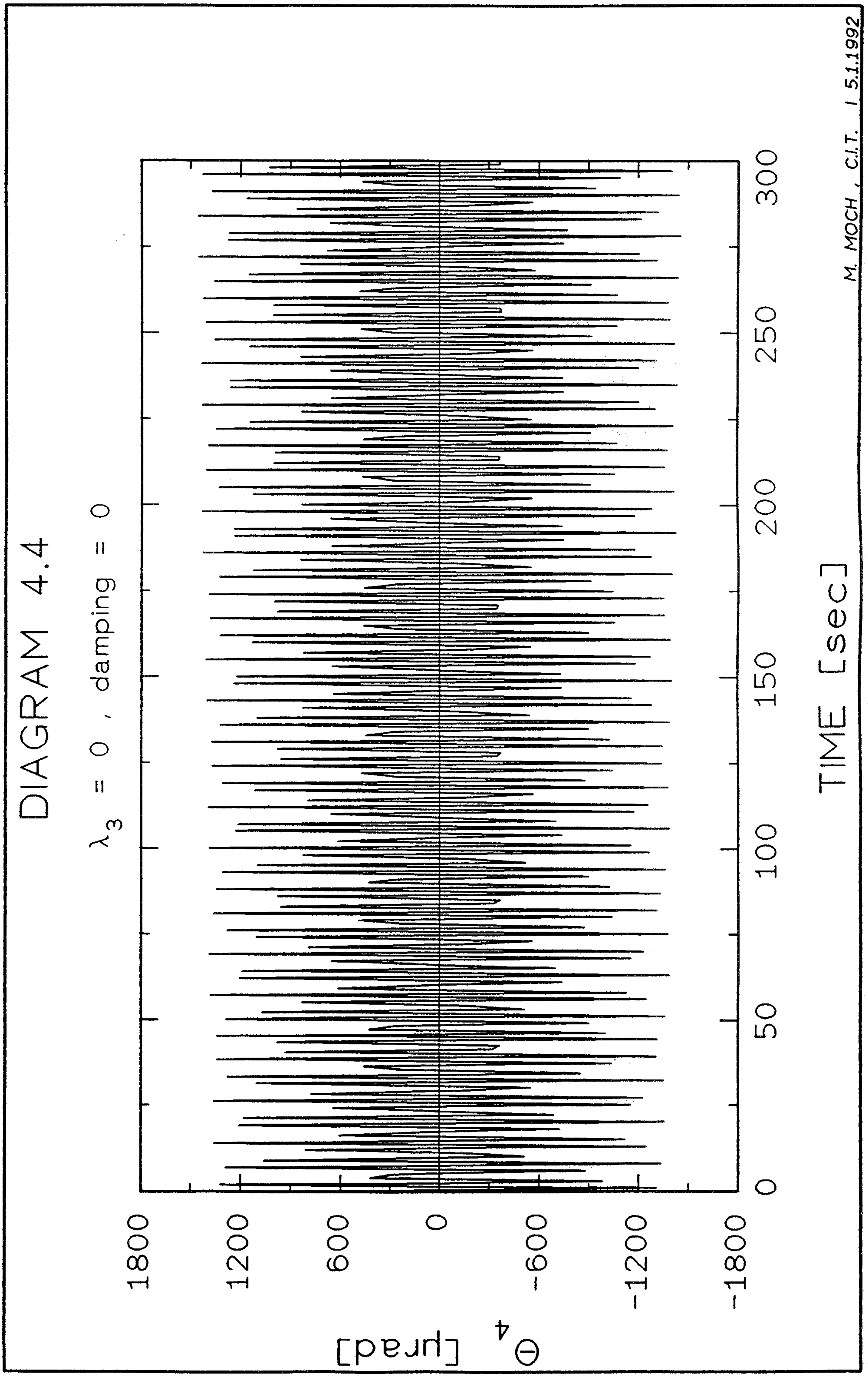


DIAGRAM 4.4
λ_3 = 0 , damping = 0
TIME [sec]
Θ_4 [μrad]
1800
1200
600
0
-600
-1200
-1800
0
50
100
150
200
250
300
M. MOCH , C.I.T. / 5.1.1992

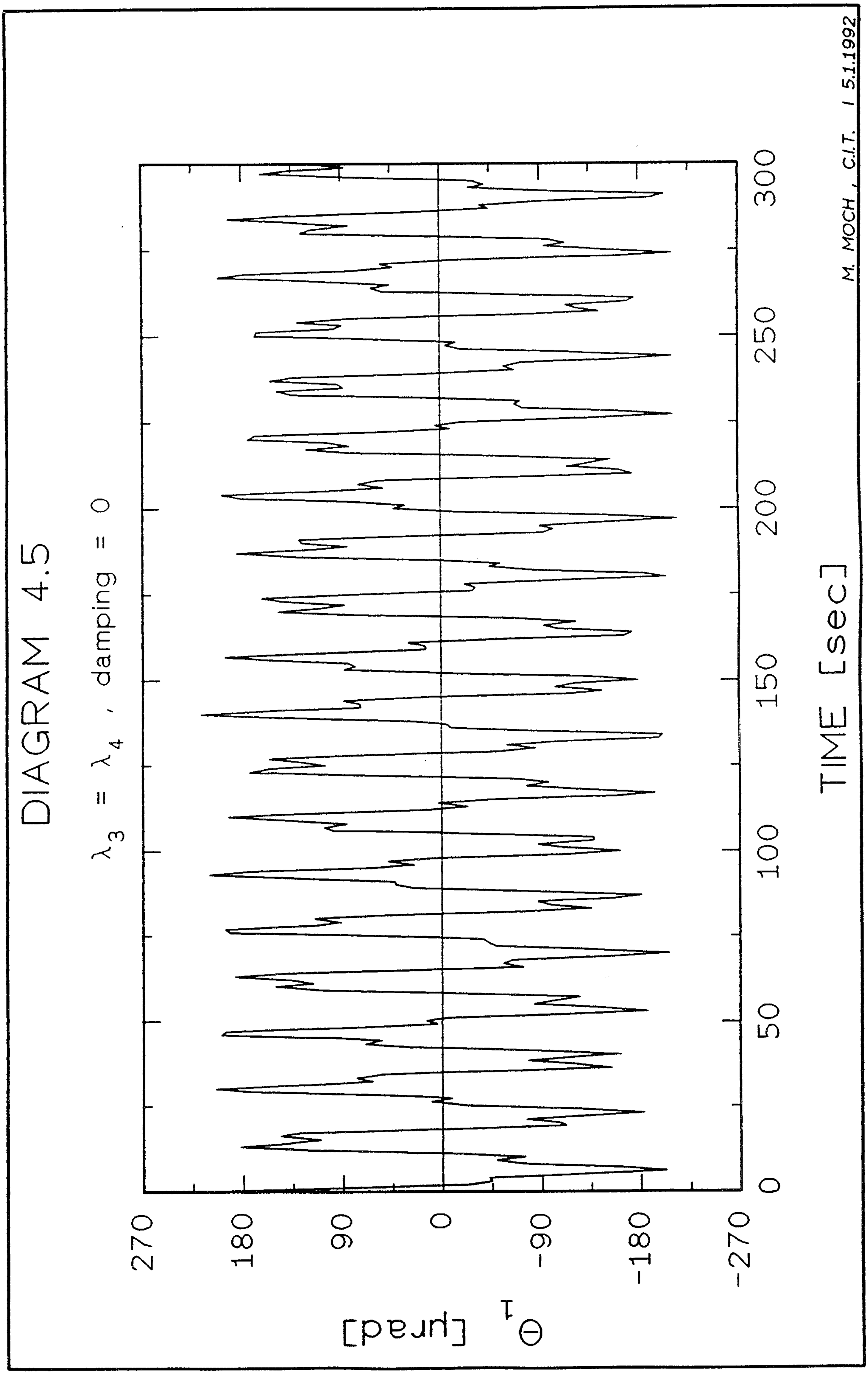


DIAGRAM 4.5
λ3 = λ4 , damping = 0
TIME [sec]
Θ1 [µrad]
0
50
100
150
200
250
300
270
180
90
0
-90
-180
-270
M. MOCH , C.I.T. 1 5.1.1992

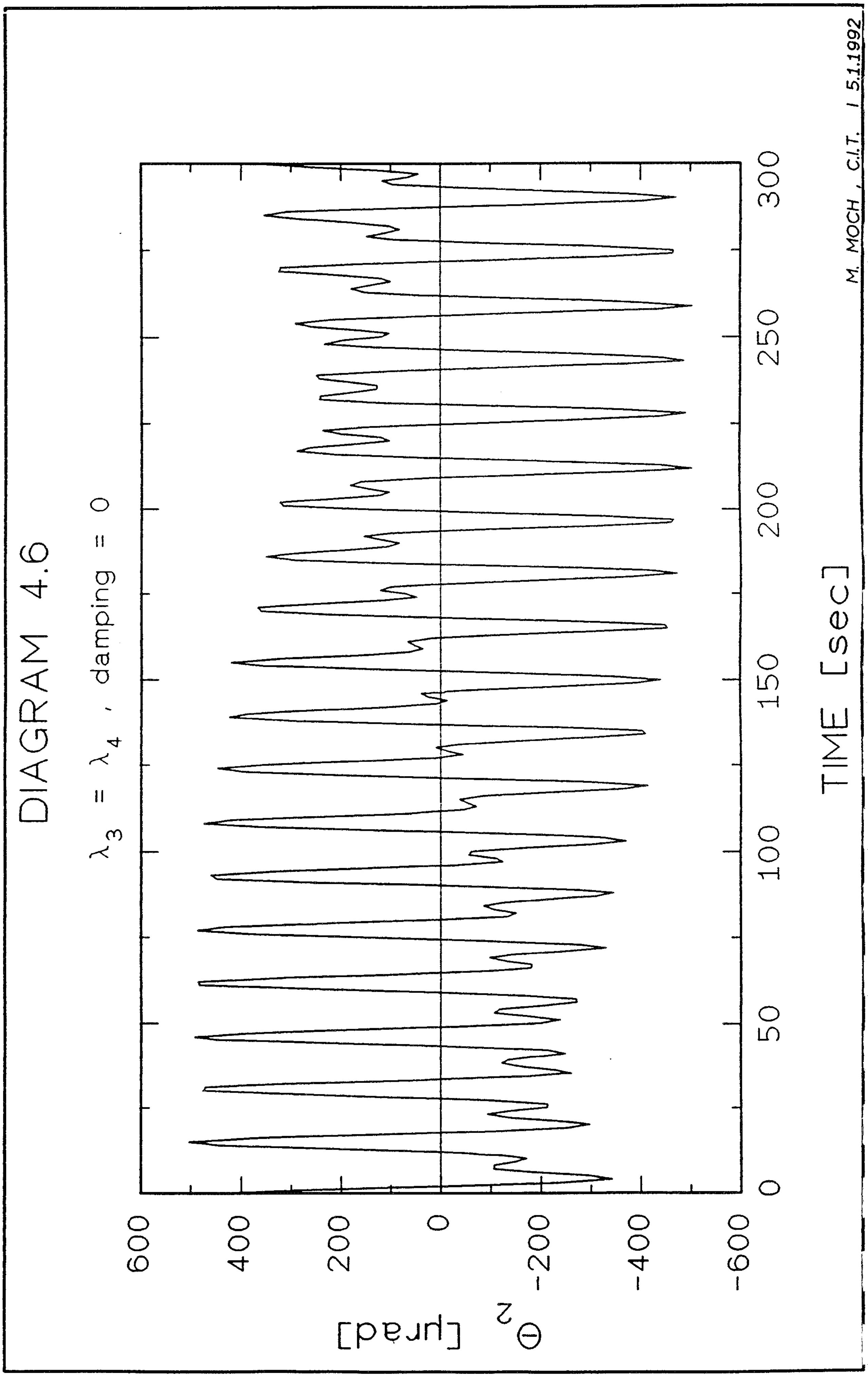


DIAGRAM 4.6
$\lambda_3 = \lambda_4$, damping = 0
TIME [sec]
Θ_2 [µrad]
600
400
200
0
-200
-400
-600
0
50
100
150
200
250
300
M. MOCH , C.I.T. / 5.1.1992

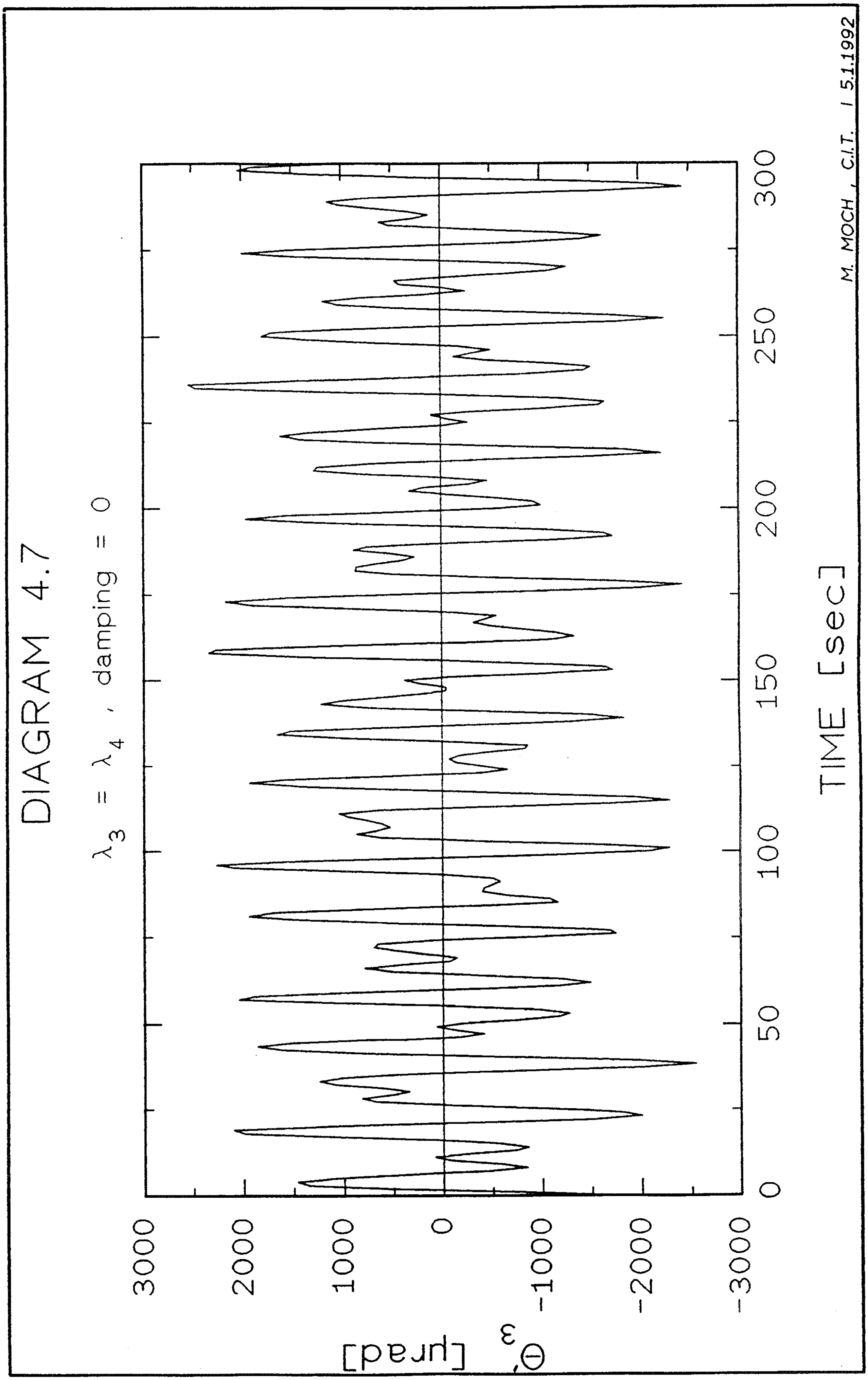
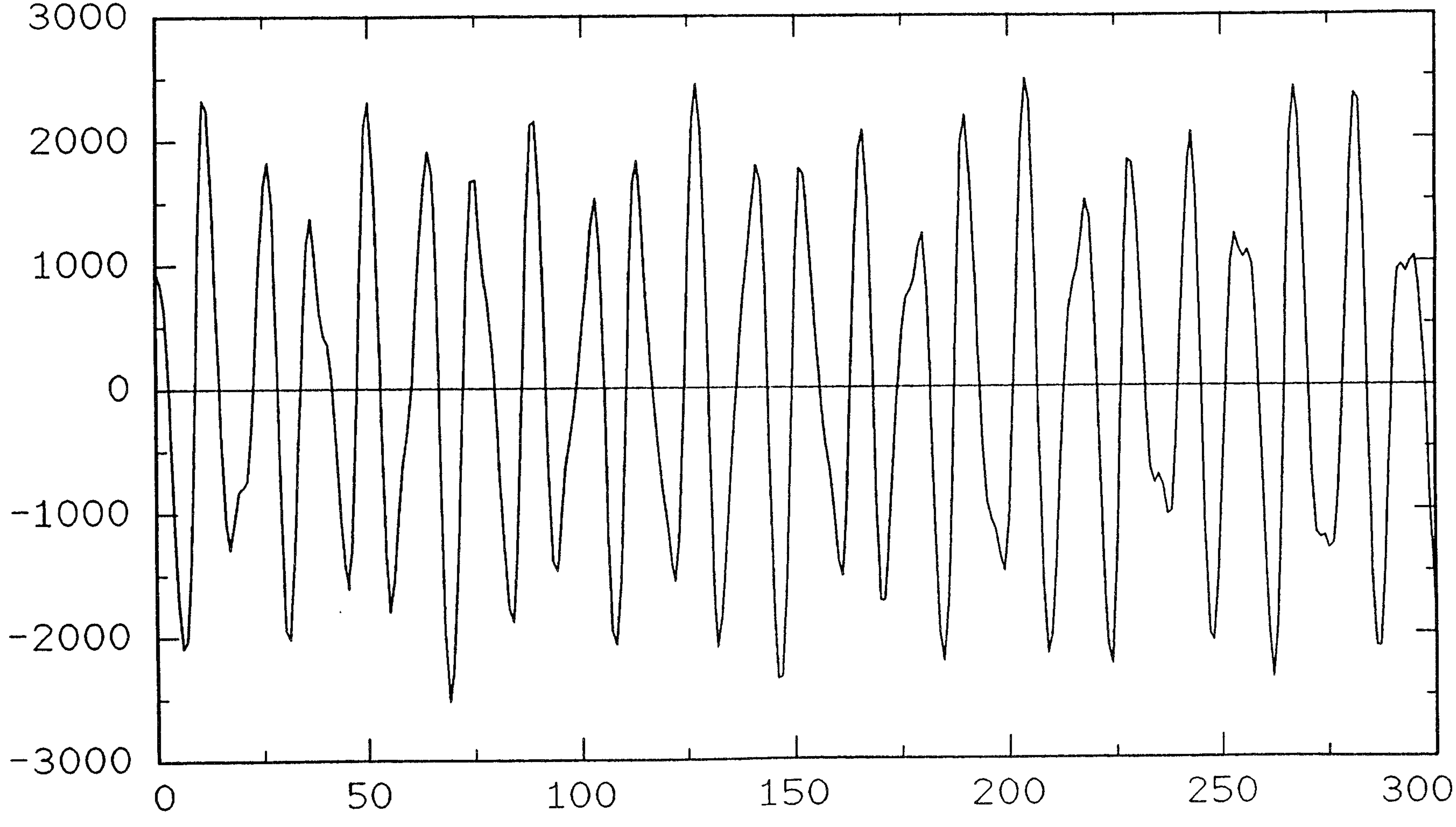


DIAGRAM 4.7
$\lambda_3 = \lambda_4$, damping = 0
Θ'_3 [µrad]
3000
2000
1000
0
-1000
-2000
-3000
0
50
100
150
200
250
300
TIME [sec]
M. MOCH , C.I.T. / 5.1.1992

DIAGRAM 4.8

$\lambda_3 = \lambda_4$, damping = 0



M. MOCH , C.I.T. I 5.1.1992

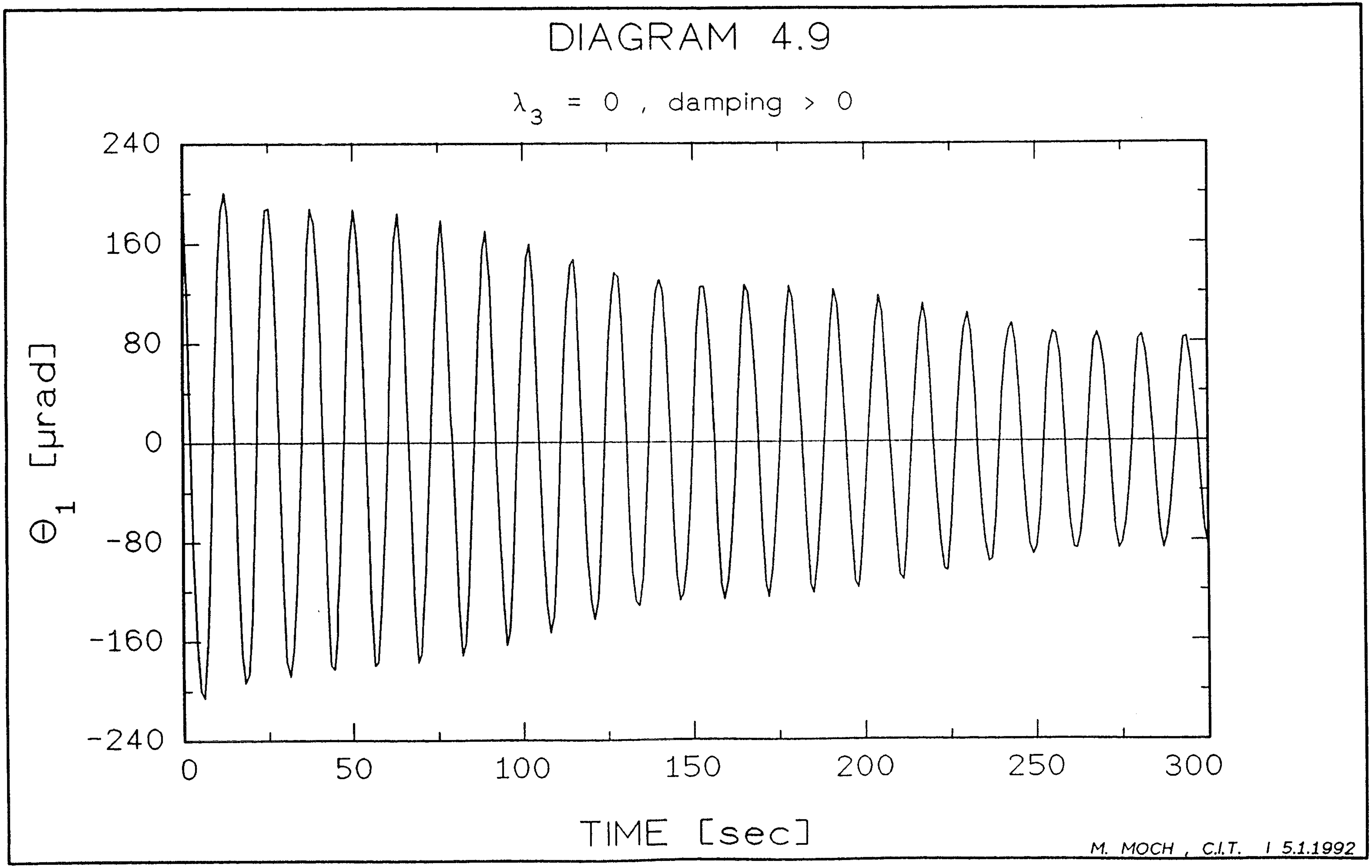


DIAGRAM 4.9
λ3 = 0 , damping > 0
240
160
80
0
-80
-160
-240
Θ1 [μrad]
0
50
100
150
200
250
300
TIME [sec]
M. MOCH , C.I.T. I 5.1.1992

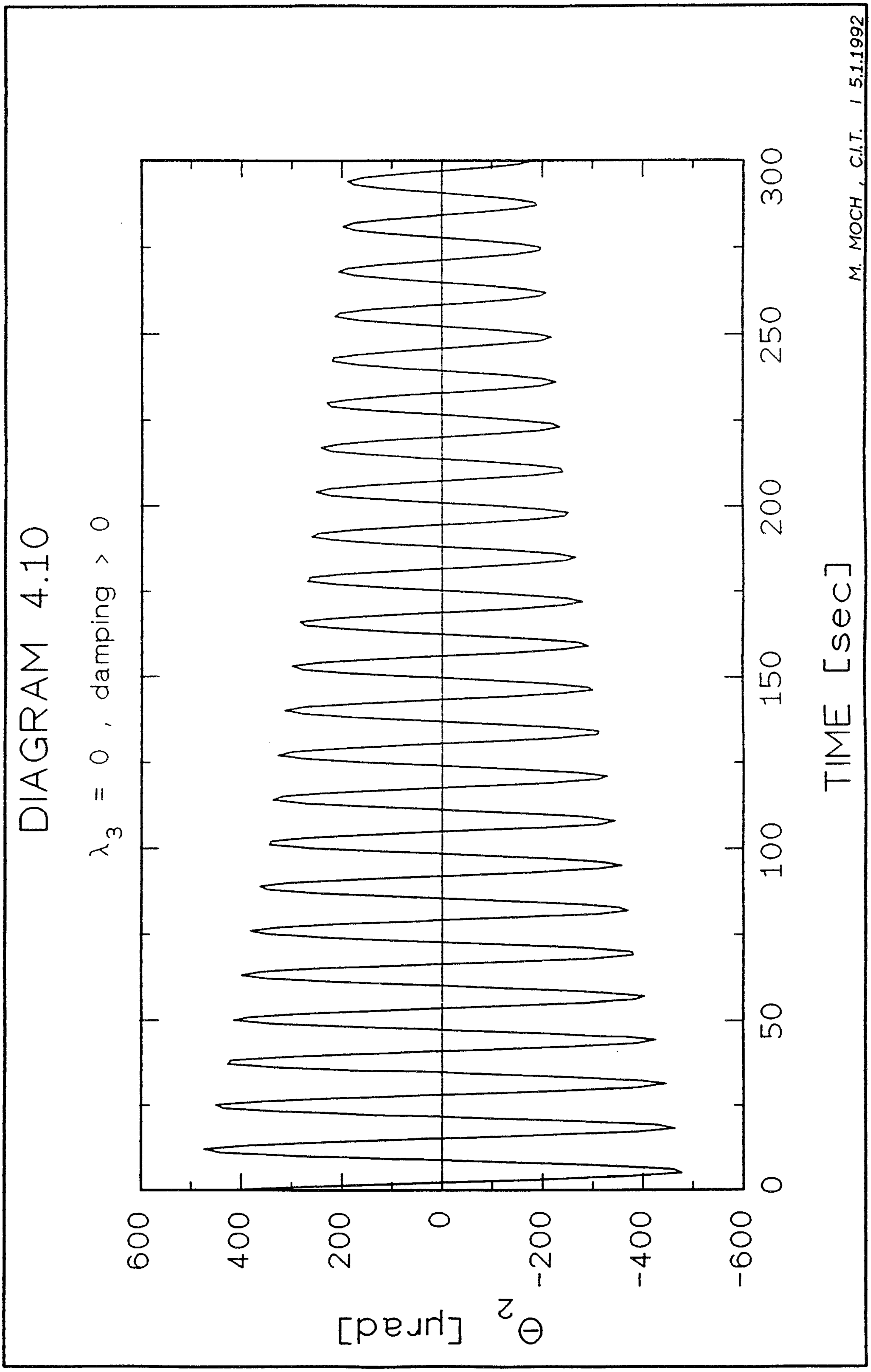


DIAGRAM 4.10
λ3 = 0 , damping > 0
Θ2 [μrad]
600
400
200
0
-200
-400
-600
0
50
100
150
200
250
300
TIME [sec]
M. MOCH , C.I.T. I 5.1.1992

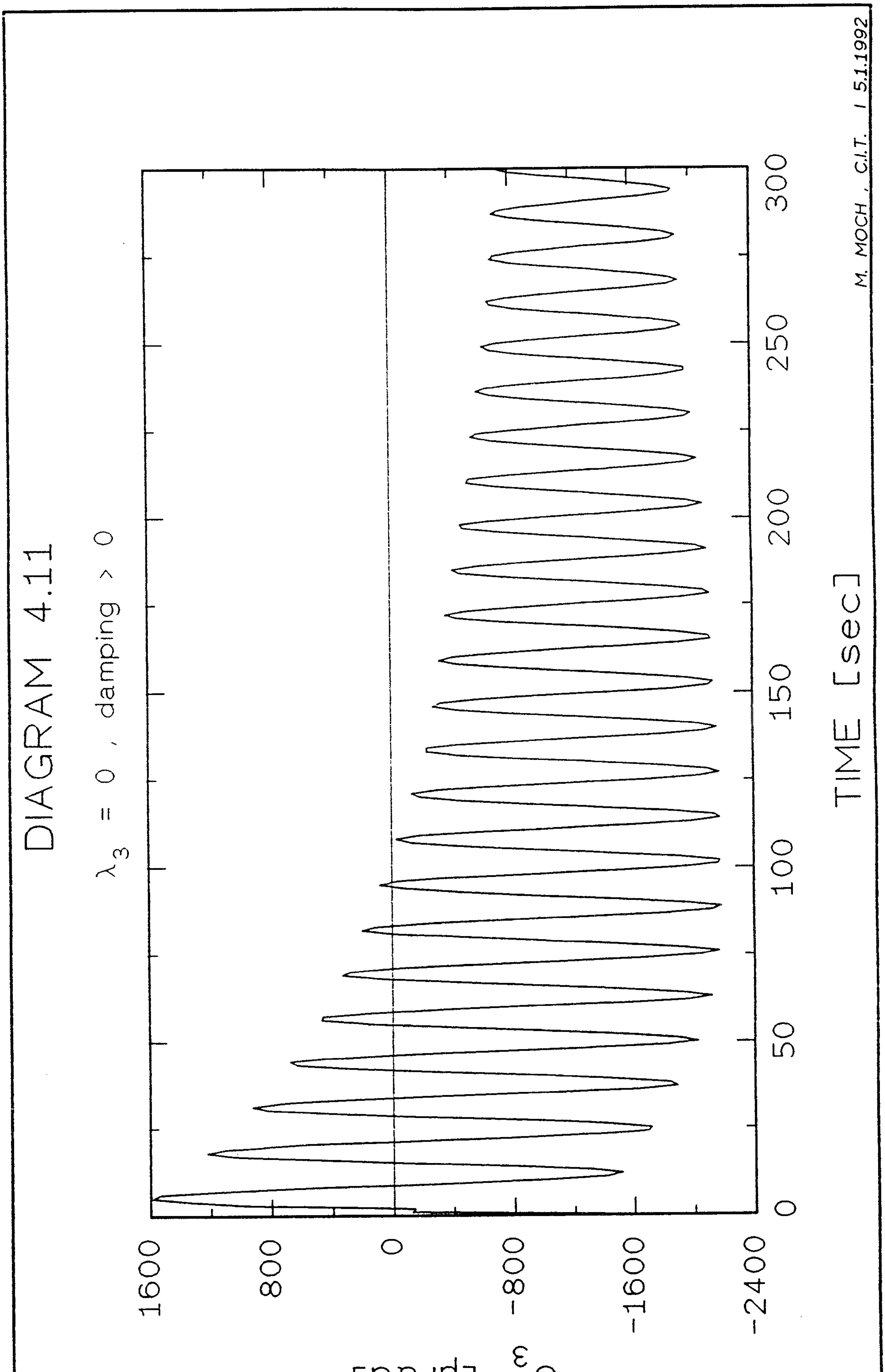


DIAGRAM 4.11
λ_3 = 0 , damping > 0
1600
800
0
-800
-1600
-2400
Θ_3 [µrad]
0
50
100
150
200
250
300
TIME [sec]
M. MOCH , C.I.T. / 5.1.1992

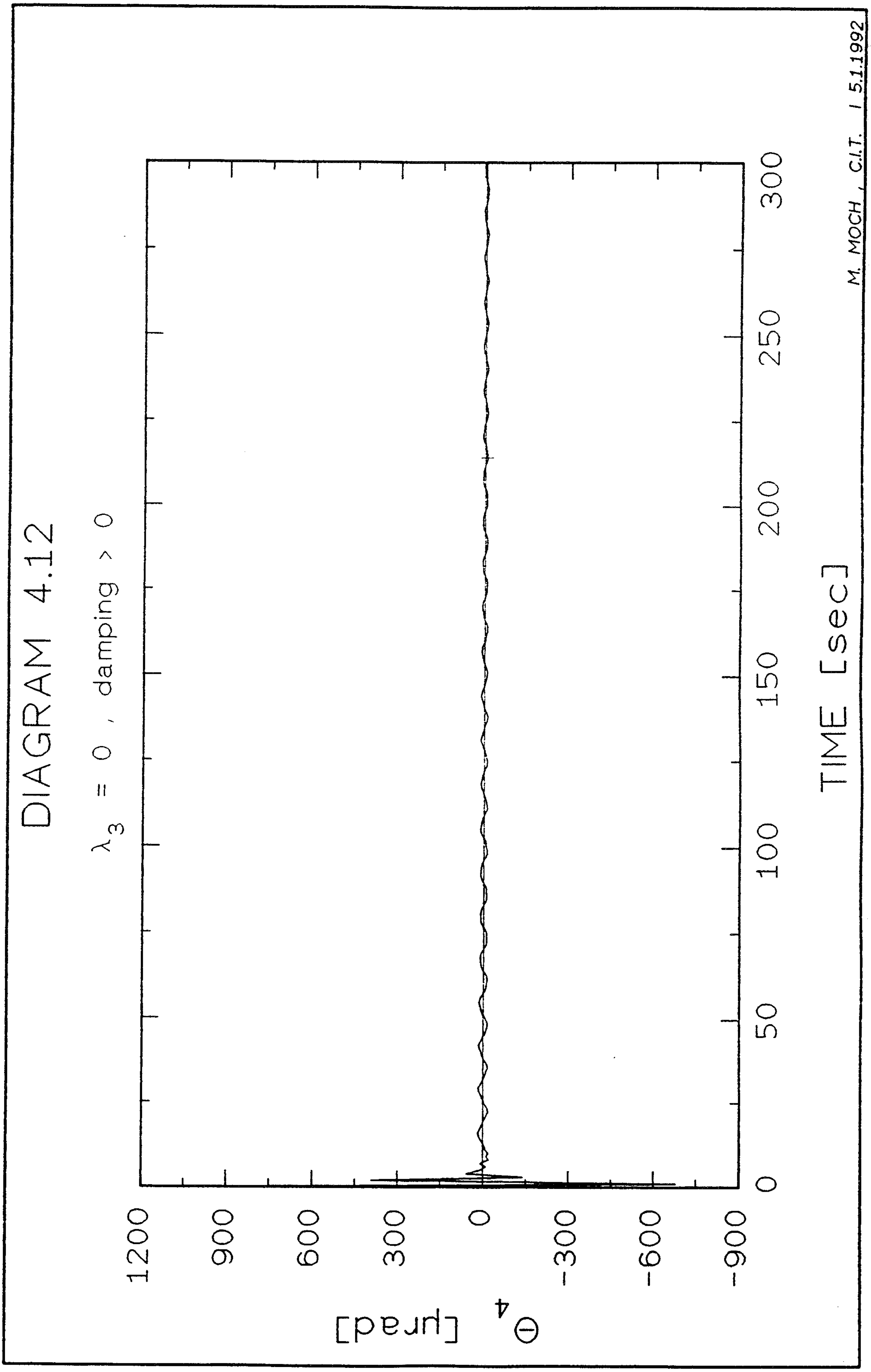


DIAGRAM 4.12
λ_3 = 0 , damping > 0
Θ_4 [µrad]
1200
900
600
300
0
-300
-600
-900
0
50
100
150
200
250
300
TIME [sec]
M. MOCH , C.I.T. 15.1.1992

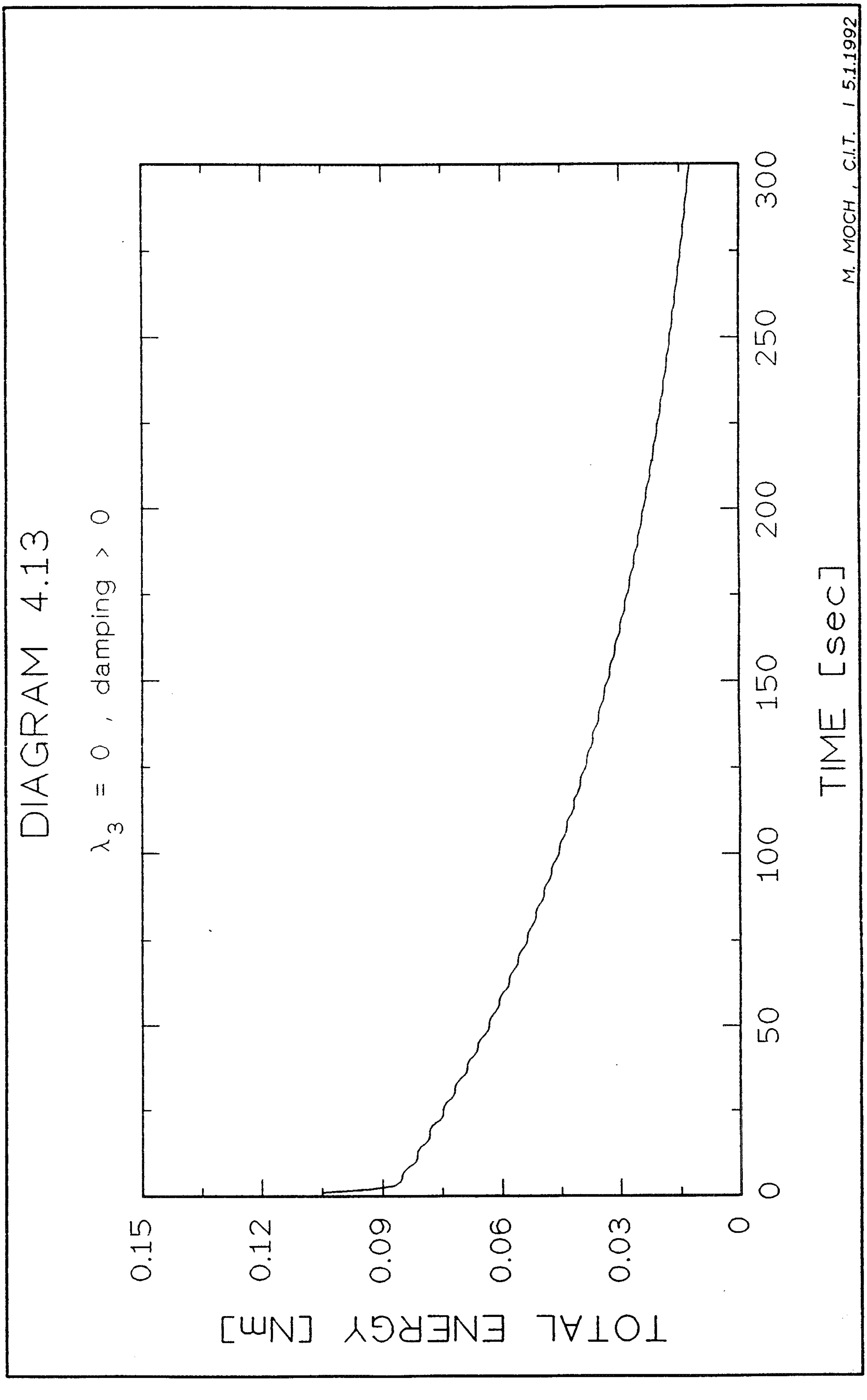
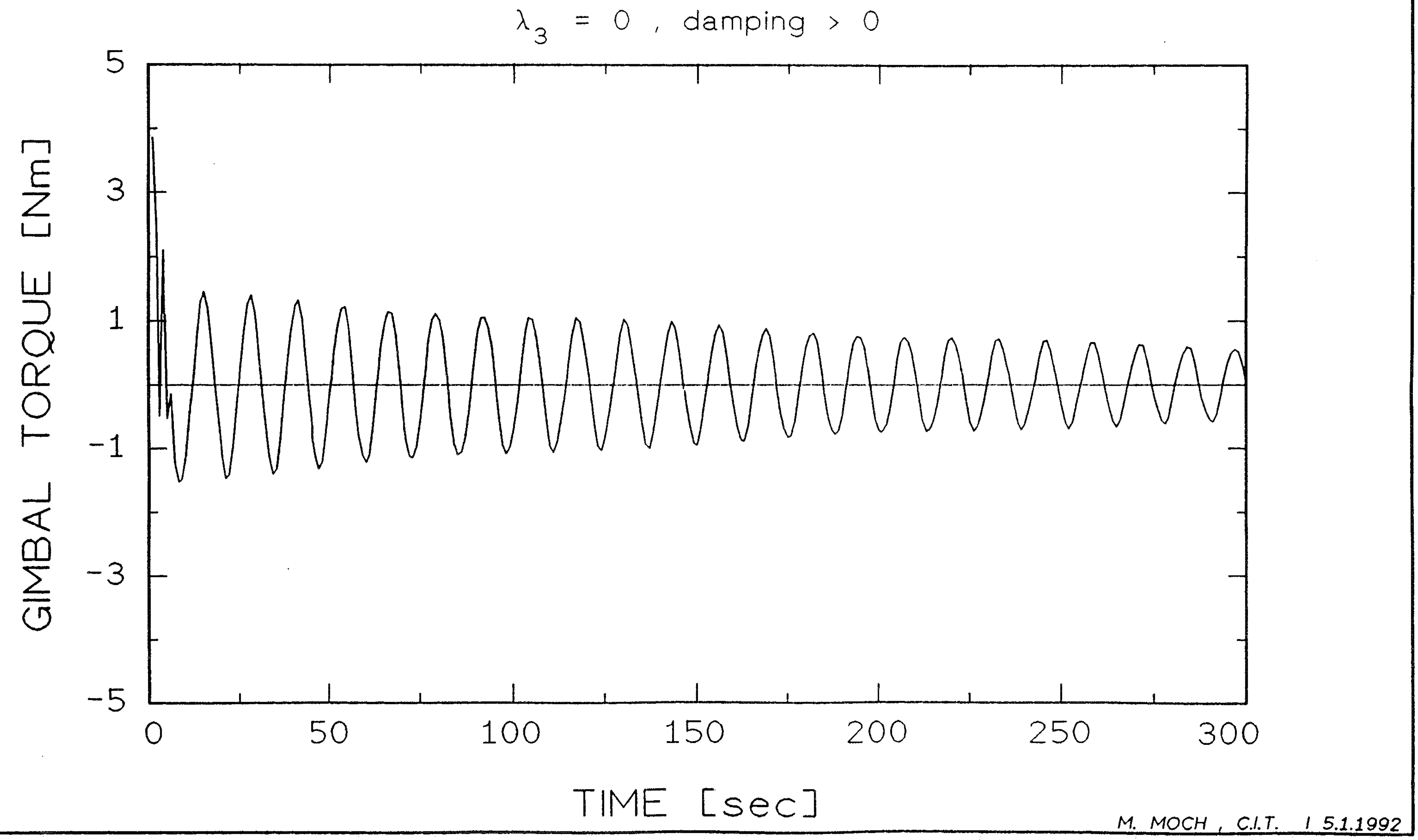


DIAGRAM 4.13
λ_3 = 0 , damping > 0
TOTAL ENERGY [Nm]
0.15
0.12
0.09
0.06
0.03
0
0
50
100
150
200
250
300
TIME [sec]
M. MOCH , C.I.T. I 5.1.1992

DIAGRAM 4.14


λ3 = 0 , damping > 0
GIMBAL TORQUE [Nm]
5
3
1
-1
-3
-5
0
50
100
150
200
250
300
TIME [sec]
M. MOCH , C.I.T. I 5.1.1992

4.4.2 Eigenvalues and eigenvectors

Eigenvalues and eigenvectors are evaluated by the program "EIGEN", using the system matrices (4.15) - (4.17).

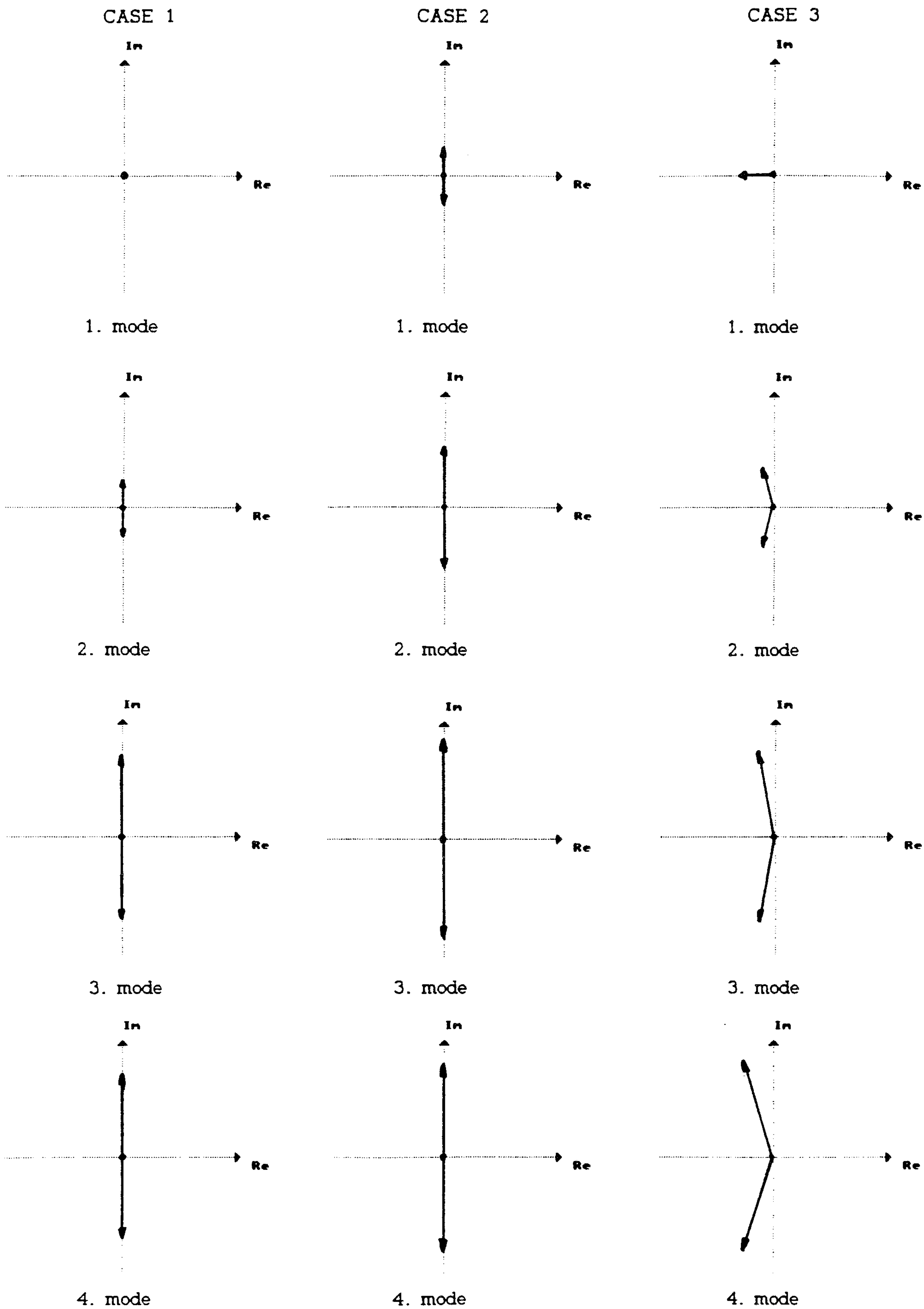
For each configuration (except case 3, first mode), we obtain pairs of complex conjugate eigenvalues. Each pair of eigenvalues describes one mode. Thus, the first 4 modes of the system are described. The magnitudes of the imaginary parts are equal to the natural frequency of the k-th mode. It is characteristic, that in case of $\lambda_3 = 0$, the natural frequency of the first mode is equal to zero, which means that this mode is a rigid body mode. Since in the case of $\lambda_3 > 0$ the system gets more stiff, the natural frequencies are shifted towards higher frequencies. For the undamped configurations, the real parts of the eigenvalues are equal to zero, which means constant amplitudes for all oscillations.

In case of the damped configuration, all real parts of existing modes are naturally negative. The natural frequencies are slightly smaller, as for the undamped configuration. Like in the first case, the first natural frequency does not exist and the real parts of the eigenvalues of the first mode have no physical meaning except a rigid body mode.

COMPLEX EIGENVALUES

	Case 1 $\lambda_3 = 0$, damping = 0	Case 2 $\lambda_3 = \lambda_4$, damping = 0	Case 3 $\lambda_3 = 0$, damping > 0
1. MODE	0.000E 0 + i 0.000E 0 0.000E 0 + i 0.000E 0	0.000E 0 + i 0.398 E 0 0.000E 0 - i 0.398 E 0	0.000E 0 + i 0.000E 0 -0.589E-1 + i 0.000E 0
2. MODE	0.000E 0 + i 0.490E 0 0.000E 0 - i 0.490E 0	0.000E 0 + i 0.812 E 0 0.000E 0 - i 0.812E 0	-0.748E-2 + i 0.490E 0 -0.748E-2 - i 0.490E 0
3. MODE	0.000E 0 + i 1.417E 0 0.000E 0 - i 1.417E 0	0.000E 0 + i 1.480E 0 0.000E 0 - i 1.480E 0	-0.105E-1 + i 1.417E 0 -0.105E-1 - i 1.417E 0
4. MODE	0.000E 0 + i 9.936E 0 0.000E 0 - i 9.936E 0	0.000E 0 + i 1.207E 1 0.000E 0 - i 1.207E 1	-0.622E 0 + i 9.913E 0 -0.622E 0 - i 9.913E 0

EIGENVALUES IN COMPLEX PLANE



NATURAL FREQUENCIES

	Case 1 $\lambda_3 = 0$, damping = 0		Case 2 $\lambda_3 = \lambda_4$, damping = 0		Case 3 $\lambda_3 = 0$, damping > 0	
	ω_k [rad/sec]	ω_k [1/sec]	ω_k [rad/sec]	ω_k [1/sec]	ω_k [rad/sec]	ω_k [1/sec]
1. MODE	0.0000	0.0000	0.3979	0.0633	0.0000	0.0000
2. MODE	0.4903	0.0780	0.8120	0.1292	0.4902	0.0780
3. MODE	1.4172	0.2256	1.4803	0.2356	1.4172	0.2256
4. MODE	9.9362	1.5814	12.075	1.9218	9.9126	1.5776

Eigenvectors:

For each mode, we evaluate the corresponding eigenvector, which shows us the mode shape. The first 4 elements of the eigenvector denote the displacements and the last 4 elements denote the velocities.

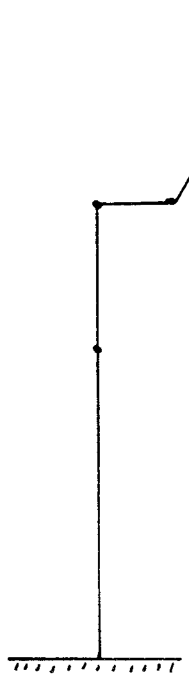
We see, that in case of no damping, the phase lag between the displacements is equal to zero and the phase lag between the displacements and the velocities is 90°.

In case of damped vibration, the phase lag between the displacements is non zero. Thus the mode shape can not be depicted, like for the undamped cases.

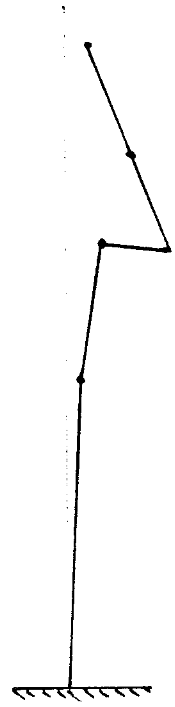
COMPLEX EIGENVECTORS

	Case 1 $\lambda_3 = 0$, damping = 0	Case 2 $\lambda_3 = \lambda_4$, damping = 0	Case 3 $\lambda_3 = 0$, damping > 0
1. MODE	0.000E 0 + i 0.000E 0	0.492E 0 + i 0.000E 0	0.000E 0 + i 0.707E-5
displacements	0.000E 0 + i 0.000E 0	1.000E 0 + i 0.000E 0	0.000E 0 + i 0.321E-5
	1.000E 0 + i 0.000E 0	0.498E 0 + i 0.000E 0	1.000E 0 + i 0.985E-3
	0.000E 0 + i 0.000E 0	0.208E 0 + i 0.000E 0	0.000E 0 - i 0.266E-2
	0.000E 0 + i 0.000E 0	0.000E 0 + i 0.212E 0	0.000E 0 + i 0.171E-2
	0.000E 0 + i 0.000E 0	0.000E 0 + i 0.431E 0	0.000E 0 + i 0.951E-4
velocities	0.000E 0 + i 0.000E 0	0.000E 0 + i 0.214E 0	0.000E 0 - i 0.436E 0
	0.000E 0 + i 0.000E 0	0.000E 0 + i 0.899E 0	0.000E 0 + i 1.000E 0
2. MODE	0.925E-1 + i 0.000E 0	0.000E 0 - i 0.155E 0	0.116E-1 - i 0.476E-3
displacements	-0.210E 0 + i 0. 000E 0	0.000E 0 + i 0.152E 0	0.230E-1 - i 0.727E-3
	1.000E 0 + i 0.000E 0	0.000E 0 - i 0.886E 0	0.000E 0 + i 1.000E 0
	0.107E 0 + i 0.000E 0	0.000E 0 - i 0.374E 0	0.578E-2 - i 0.705E 0
	0.000E 0 - i 0.455E-1	0.175E 0 + i 0.000E 0	-0.454E-1 + i 0.280E-4
	0.000E 0 - i 0.103E 0	0.171E 0 + i 0.000E 0	-0.103E 0 + i 0.428E-4
velocities	0.000E 0 - i 0.492E 0	1.000E 0 + i 0.000E 0	0.491E 0 - i 0.589E-1
	0.000E 0 - i 0.524E-3	0.422E 0 + i 0.000E 0	0.435E 0 + i 0.415E-4
3. MODE	0.000E 0 + i 0.159E 0	0.000E 0 + i 0.625E-1	0.159E 0 - i 0.923E 0
displacements	0.000E 0 - i 0.439E-1	0.000E 0 + i 0.214E-2	-0.439E-1 - i 0.210E 0
	0.000E 0 - i 0.699E 0	0.000E 0 - i 0.522E 0	-0.700E 0 + i 1.000E 0
	0.000E 0 - i 0.124E-1	0.000E 0 - i 0.236E 0	-0.124E-1 + i 0.975E 3
	-0.228E 0 + i 0.000E 0	0.120E 0 + i 0.000E 0	0.927E-2 - i 0.500E-2
	0.628E-1 + i 0.000E 0	-0.410E-2 + i 0.000E 0	-0.538E-2 - i 0.972E-2
velocities	1.000E 0 + i 0.000E 0	1.000E 0 + i 0.000E 0	0.000E 0 - i 0.748E-2
	0.177E-1 + i 0.000E 0	0.451E 0 + i 0.000E 0	0.169E 0 - i 0.284E-2
4. MODE	0.000E 0 - i 0.112E-3	0.000E 0 - i 0.123E-3	-0.112E-3 + i 0.766E-2
displacements	0.000E 0 - i 0.610E-5	0.000E 0 - i 0.119E-4	-0.636E-5 - i 0.408E-2
	0.000E 0 + i 0.285E-1	0.000E 0 + i 0.265E-1	0.285E-1 - i 0.514E-2
	0.000E 0 - i 0.654E-1	0.000E 0 - i 0.599E-1	0.654E-1 + i 0.117E-1
	0.171E-2 + i 0.000E 0	0.206E-2 - i 0.000E 0	0.178E-3 - i 0.228E 0
	0.933E-4 + i 0.000E 0	0.198E-3 + i 0.000E 0	0.529E-4 + i 0.627E-1
velocities	-0.436E 0 + i 0.000E 0	-0.442E 0 + i 0.000E 0	-0.272E 0 + i 1.000E 0
	1.000E 0 + i 0.000E 0	1.000E 0 + i 0.000E 0	0.000E 0 + i 0.176E-1

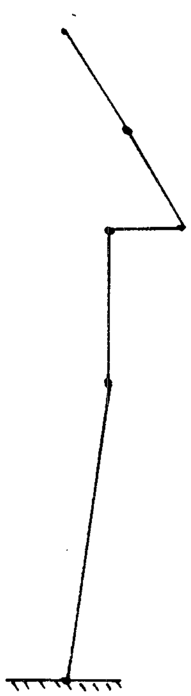
MODE SHAPES . CASE 1 : λ_3 = 0 . damping = 0



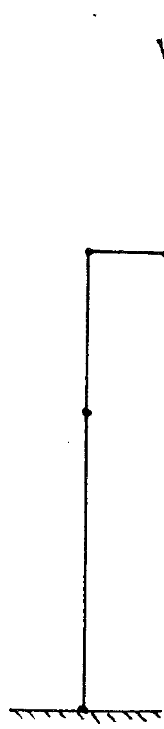
1. MODE ω_1 = 0



2. MODE ω_2 = 0.078 Hz

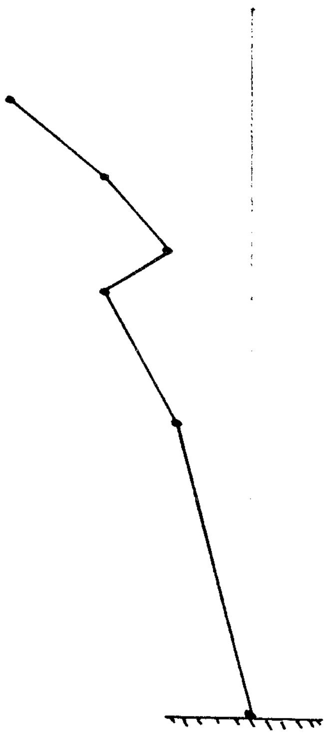


3. MODE ω_3 = 0.226 Hz

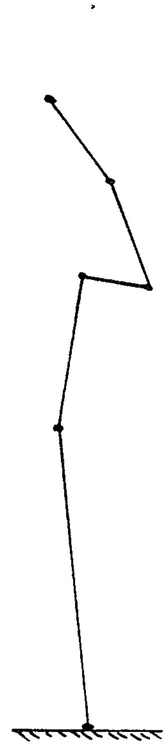


4. MODE ω_4 = 1.581 Hz

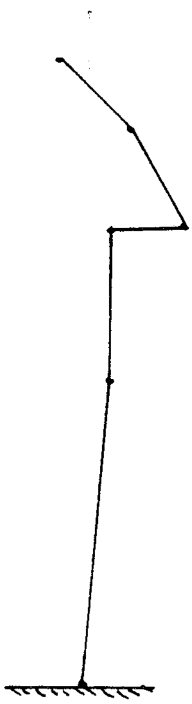
MODE SHAPES . CASE 2 : $\lambda_3 = \lambda_4$. damping = 0



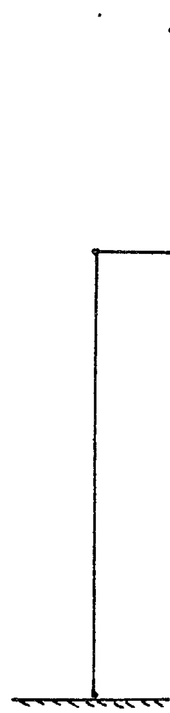
1. MODE ω_1 = 0.063



2. MODE ω_2 = 0.129 Hz



3. MODE ω_3 = 0.236 Hz



4. MODE ω_4 = 1.922 Hz

CLOSED LOOP FEEDBACK CONTROL

6.1 INTRODUCTION

After the pitching maneuver has finished, residual oscillations caused by the maneuver must be damped out as quickly as possible. To fulfil this demand, active damping by the gimbal servo motor is used. Therefore we consider closed loop feedback control, feeding back the systems displacements and velocities. A transfer function determines the corresponding servo motor torque from measured displacements and velocities. Finding the transfer function can be solved in terms of eigenvalue optimization, for which we apply an approach of Bodden and Junkins (ref. 2).

5.2 EIGENVALUE OPTIMIZATION ALGORITHMS

Parameterization of the controlled system's eigenvalues and eigenvectors:

We consider the system of differential equations for closed loop feedback control:

$$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{C}\dot{\mathbf{x}} + \mathbf{K}\mathbf{x} = \mathbf{B}\mathbf{u} \tag{5.1}$$

$\mathbf{M}$ = n x n symmetric positiv definite mass matrix.

$\mathbf{C}$ = n x n symmetric positive semidefinite structural damping matrix.

$\mathbf{K}$ = n x n symmetric semidefinite stiffness matrix.

$\mathbf{B}$ = n x m control influence matrix.

$\mathbf{x}$ = n x 1 configuration vector.

$\mathbf{u}$ = m x 1 control vector.

For linear output feedback control, the local position, velocity and acceleration measurements are denoted by:

$$\mathbf{y}_1 = \mathbf{H}_1\mathbf{x} \qquad \mathbf{y}_2 = \mathbf{H}_2\dot{\mathbf{x}} \qquad \mathbf{y}_3 = \mathbf{H}_3\ddot{\mathbf{x}} \tag{5.2}$$

which represent the linear relationship of the locally measured position $\mathbf{y}_1$, velocity $\mathbf{y}_2$, and acceleration $\mathbf{y}_3$, where:

$\mathbf{y}_1$ = m_1 x 1 vector	$\mathbf{H}_1$ = m_1 x n matrix
$\mathbf{y}_2$ = m_2 x 1 vector	$\mathbf{H}_2$ = m_2 x n matrix
$\mathbf{y}_3$ = m_3 x 1 vector	$\mathbf{H}_3$ = m_3 x n matrix

We seek the constant gain matrices $\mathbf{G}_1$, $\mathbf{G}_2$, and $\mathbf{G}_3$ so that:

$$\begin{aligned}\mathbf{u} &= -(\mathbf{G}_1\mathbf{y}_1 + \mathbf{G}_2\mathbf{y}_2 + \mathbf{G}_3\mathbf{y}_2) \\ &= -\mathbf{G}_1\mathbf{H}_1\mathbf{x} - \mathbf{G}_2\mathbf{H}_2\dot{\mathbf{x}} - \mathbf{G}_3\mathbf{H}_3\ddot{\mathbf{x}}\end{aligned} \tag{5.3}$$

Introducing eq. (5.3) into eq. (5.1), the system of differential equations for closed loop feedback control can be written as:

$$\overline{\mathbf{M}}\ddot{\mathbf{x}} + \overline{\mathbf{C}}\dot{\mathbf{x}} + \overline{\mathbf{K}}\mathbf{x} = 0 \qquad (5.4)$$

Where the closed loop system matrices are defined as:

$$\overline{\mathbf{M}} = \mathbf{M} + \mathbf{B}\mathbf{G}_3\mathbf{H}_3 \qquad \overline{\mathbf{C}} = \mathbf{C} + \mathbf{B}\mathbf{G}_2\mathbf{H}_2 \qquad \overline{\mathbf{K}} = \mathbf{K} + \mathbf{B}\mathbf{G}_1\mathbf{H}_1 \qquad (5.5)$$

We introduce the notations:

$$\begin{array}{llll} \mathbf{M} = \mathbf{M}(\mathbf{a}) & \mathbf{C} = \mathbf{C}(\mathbf{a}) & \mathbf{K} = \mathbf{K}(\mathbf{a}) & \\ \mathbf{B} = \mathbf{B}(\mathbf{c}) & \mathbf{H}_i = \mathbf{H}_i(\mathbf{b}) & \mathbf{G}_i = \mathbf{G}_i(\mathbf{g}) & i = 1, 2, 3 \end{array} \qquad (5.6)$$

Where **a** is a vector of the geometric snd structural parameters of the model, **b** is a vector of sensor location parameters, **c** is a vector of the actuator type and location parameters, and **g** is a vector of the control gains. We define the N x 1 global structural and control parameter vector as:

$$\mathbf{p}^T = (\mathbf{a}^T\mathbf{b}^T\mathbf{c}^T\mathbf{g}^T) \qquad (5.7)$$

It is apparent from eq. (5.5) that:

$$\overline{\mathbf{M}} = \overline{\mathbf{M}}(\mathbf{p}) \qquad \overline{\mathbf{C}} = \overline{\mathbf{C}}(\mathbf{p}) \qquad \overline{\mathbf{K}} = \overline{\mathbf{K}}(\mathbf{p}) \qquad (5.8)$$

We consider the first-order state-space differential equation, which is equivalent to the second-order closed-loop system of eq. (5.4) :

$$\mathbf{A}\dot{\mathbf{z}} = \mathbf{B}\mathbf{z} \qquad (5.9)$$

Where:

$$\mathbf{z} = \begin{bmatrix} \mathbf{x} \\ \dot{\mathbf{x}} \end{bmatrix} \qquad \mathbf{A} = \begin{bmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \overline{\mathbf{M}} \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} \mathbf{0} & \mathbf{M} \\ -\overline{\mathbf{K}} & -\overline{\mathbf{C}} \end{bmatrix} \qquad (5.10)$$

It is evident that:

$$\mathbf{A} = \mathbf{A}(\mathbf{p}) \qquad \mathbf{B} = \mathbf{B}(\mathbf{p}) \qquad (5.11)$$

The left and the right eigenvector problems are:

$$\text{left:} \quad \lambda_i \mathbf{A}^T \mathbf{\Psi}_i = \mathbf{B}^T \mathbf{\Psi}_i \qquad \text{right:} \quad \lambda_i \mathbf{A} \mathbf{\Phi}_i = \mathbf{B} \mathbf{\Psi}_i \qquad (5.12.L), \quad (5.12.R)$$

Where the conventional normalizations for the eigenvectors are adopted as:

$$\mathbf{\Psi}_j^T \mathbf{A} \mathbf{\Phi}_i = \delta_{ij} \qquad \mathbf{\Psi}_j^T \mathbf{B} \mathbf{\Phi}_i = \lambda_i \delta_{ij} \qquad (5.13.1)\ ,\ (5.13.2)$$

Since $\mathbf{A} = \mathbf{A}(\mathbf{p})$ and $\mathbf{B} = \mathbf{B}(\mathbf{p})$ it seems natural to consider the eigenvalues $\{\lambda_1, \ldots \lambda_{2n}\}$ and the eigenvectors $\{\mathbf{\Phi}_1, \mathbf{\Psi}_1, \ldots \mathbf{\Phi}_{2n}, \mathbf{\Psi}_{2n}\}$ to be functions of the parameter vector $\mathbf{p}$:

$$\lambda_i = \lambda_i(\mathbf{p}) \qquad \mathbf{\Phi}_i = \mathbf{\Phi}_i(\mathbf{p}) \qquad \mathbf{\Psi}_i = \mathbf{\Psi}_i(\mathbf{p}) \qquad (5.14)$$

Except for occasional singular events (e.g. multiple eigenvalues, eigenvalues = 0) the nonlinear functional dependence of eqs. (5.14) can be assumed as continuous.

Minimum modification strategy for structural/controller design iterations:

Consider a constrained optimization problem wherein we seek the optimal value of the parameter vector $\mathbf{p}$ that extremizes some performance measure:

$$J = J\{\lambda_1(\mathbf{p}), \ldots \lambda_{2n}(\mathbf{p}), \mathbf{\Phi}_1(\mathbf{p}), \ldots \mathbf{\Phi}_{2n}(\mathbf{p}), \mathbf{p}\} \tag{5.15}$$

Subject to the satisfaction of the N_α equality constraints:

$$\alpha_j\{\lambda_1(\mathbf{p}), \ldots \lambda_{2n}(\mathbf{p}), \mathbf{\Phi}_1(\mathbf{p}), \ldots \mathbf{\Phi}_{2n}(\mathbf{p}), \mathbf{p}\} = 0 \qquad j = 1, 2, \ldots N_\alpha \tag{5.16}$$

And N_β inequality constraints:

$$\beta_{jL} \le \beta_j\{\lambda_1(\mathbf{p}), \ldots \lambda_{2n}(\mathbf{p}), \mathbf{\Phi}_1(\mathbf{p}), \ldots \mathbf{\Phi}_{2n}(\mathbf{p}), \mathbf{p}\} \le \beta_{ju} \qquad j = 1, 2, \ldots N_\beta \tag{5.17}$$

β_{jL} = lower limit $\qquad$ β_{ju} = upper limit

Thus, the performance measure and constraints are defined in terms of the eigensolution, but we also admit explicit dependence upon $\mathbf{p}$ to include, for example, structure and control system criteria. Eqs. (5.15) - (5.17) define a nonlinear programming problem. One iteration strategy confines local attention to only the locally violated inequality constraints and all of the functions of eqs. (5.15) & (5.16). Specifically, we can seek the smallest correction vector $\Delta\mathbf{p}$ that achieves specified increments of ΔJ, $\Delta\alpha_j$, $\Delta\beta_j$ for a subset of the functions of eqs. (5.15) - (5.17). Linearizing these equations about a typical value p_i results in:

$$\Delta\boldsymbol{\gamma} = \left[\left.\frac{\partial\boldsymbol{\gamma}}{\partial\mathbf{p}}\right|_{p_i}\right]\Delta\mathbf{p} \tag{5.18}$$

Where $\boldsymbol{\gamma}$ is the object vector:

$$\boldsymbol{\gamma} = (J^T, \alpha_j, \beta_j)^T \tag{5.19}$$

The quadratic correction norm for $\Delta \mathbf{p}$ is defined as:

$$N = 1/2\, \Delta \mathbf{p}^T \mathbf{W}\, \Delta \mathbf{p} \tag{5.20}$$

Where $\mathbf{W}$ is a suitable weight matrix to highlight certain terms. Since eq. (5.18) constitutes typically, a small number of equations for a large number of unknowns, we expect an infinity of exact $\Delta \mathbf{p}$ solutions; some criterion must be introduced to select a particular solution. Motivated by the desire to satisfy as nearly as possible, the implicit local linearity assumption, we seek a "small $\Delta \mathbf{p}$" solution. To mimimize the correction norm of $\Delta \mathbf{p}$ subject to the linearized constrained equation(5.18), we multiply eq. (5.18) with the unknown Lagrange multiplier vector $\mathbf{\Lambda}$ and add eq. (5.20) to form the Homiltonian:

$$\mathcal{L}(\Delta \mathbf{p}) = 1/2\, \Delta \mathbf{p}^T \mathbf{W}\, \Delta \mathbf{p} + \mathbf{\Lambda} \left[\Delta \gamma - \left[\frac{\partial \gamma}{\partial \mathbf{p}} \bigg|_{p_i} \right] \Delta \mathbf{p} \right] \tag{5.21}$$

We seek the mimimum of eq. (5.21). Differentiating with respect to $\Delta \mathbf{p}$ results in:

$$(\partial \mathcal{L} / \Delta \mathbf{p}) = (\partial \mathcal{L} / \Delta \mathbf{p}^T) = \mathbf{W}\, \Delta \mathbf{p} - \mathbf{\Lambda} \left[\frac{\partial \gamma}{\partial \mathbf{p}} \bigg|_{p_i} \right] = 0 \tag{5.22}$$

or:

$$\mathbf{W}\, \Delta \mathbf{p} - \left[\frac{\partial \gamma}{\partial \mathbf{p}} \bigg|_{p_i} \right]^T \mathbf{\Lambda} = 0$$

hence:

$$\Delta \mathbf{p} = \mathbf{W}^{-1} \left[\frac{\partial \gamma}{\partial \mathbf{p}} \bigg|_{p_i} \right]^T \mathbf{\Lambda} \tag{5.23}$$

We introduce eq. (5.18) into eq. (5.23):

$$\left[\frac{\partial \gamma}{\partial \mathbf{p}} \bigg|_{p_i} \right] \mathbf{W}^{-1} \left[\frac{\partial \gamma}{\partial \mathbf{p}} \bigg|_{p_i} \right]^T \mathbf{\Lambda} = \Delta \gamma \tag{5.24}$$

Eq. (5.24) is a linear, inhomogenous equation system, to determine the unknown Lagrange multiplier vector $\mathbf{\Lambda}$. The number of unknowns is equal to the number of equations. The parameter vector $\Delta \mathbf{p}$ is subsequently determined by eq.(5.23).

Partial derivatives of closed loop eigenvalues with respect to structural and control parameters:

For the linearized constrained equation (5.18), we have to determine the sensitivity matrix:

$$\left[\left. \frac{\partial \boldsymbol{\gamma}}{\partial \mathbf{p}} \right|_{p_L} \right]$$

The object vector $\boldsymbol{\gamma}$ is a function of the eigenvalue vector $\boldsymbol{\lambda}$, which is a function of the parameter vector $\mathbf{p}$:

$$\boldsymbol{\gamma} = \boldsymbol{\gamma}\left[\boldsymbol{\lambda}(\mathbf{p}) \right]$$

Differentiating with respect to $\mathbf{p}$ results in (chain rule):

$$(\partial \boldsymbol{\gamma} / \partial \mathbf{p}) = (\partial \boldsymbol{\gamma} / \partial \boldsymbol{\lambda}) \, (\partial \boldsymbol{\lambda} / \partial \mathbf{p}) \tag{5.25}$$

The derivatives $(\partial \boldsymbol{\gamma} / \partial \boldsymbol{\lambda})$ depend on the constrained functions of the specific problem, but the derivatives $(\partial \boldsymbol{\lambda} / \partial \mathbf{p})$ can be solved generally. We consider the partial derivative $(\partial \lambda_K / \partial p_L)$ for two typical elements λ_K and p_L.
We use the definitions for the left and right eigenvalue problems:

$$\lambda_i \mathbf{A}^T \boldsymbol{\Psi}_i = \mathbf{B}^T \boldsymbol{\Psi}_i \qquad \lambda_i \mathbf{A} \boldsymbol{\Phi}_i = \mathbf{B} \boldsymbol{\Phi}_i \qquad (5.12.L) \, , \, (5.12.R)$$

Where the conventional normalizations for the eigenvectors are:

$$\lambda_i \boldsymbol{\Psi}_j^T \mathbf{A} \, \boldsymbol{\Phi}_i = \lambda_i \delta_{ij} \qquad \boldsymbol{\Psi}_j^T \mathbf{B} \, \boldsymbol{\Phi} = \lambda_i \delta_{ij} \qquad (5.13.1) \, , \, (5.13.2)$$

Equating eq. (5.13.1) to eq. (5.13.2) results in:

$$\lambda_i \boldsymbol{\Psi}_j^T \mathbf{A} = \boldsymbol{\Psi}_j^T \mathbf{B} \tag{5.13}$$

To obtain the partial derivatives $(\partial \lambda_K/\partial p_L)$, we differentiate eq. (5.12.R) with respect to a typical element p_L and premultiply the result with Ψ_j^T :

$$\Psi_j^T\Big[(\partial\lambda_i/\partial p_L)\mathbf{A}\,\Phi_i + \lambda_i[(\partial\mathbf{A}/\partial p_L)\Phi_i + \underbrace{\mathbf{A}(\partial\Phi_i/\partial p_L)}_{(1)}]\Big] = \Psi_j^T\Big[(\partial\mathbf{B}/\partial p_L)\Phi_i + \underbrace{\mathbf{B}(\partial\Phi_i/\partial p_L)}_{(2)}\Big] \quad (5.26)$$

To eliminate the terms (1) & (2), we postmultiply eq. (5.13) with $(\partial\Phi_i/\partial p_L)$:

$$\lambda_i\Psi_i^T\mathbf{A}(\partial\Phi_i/\partial p_L) = \Psi_i^T\mathbf{B}(\partial\Phi_i/\partial p_L) \quad (5.27)$$

And substitute term 1 of eq. (5.26) with the left hand side of eq. (5.27). We then see, both terms 1&2 are equal and therefore cancel out, leaving the following equation:

$$\Psi_j^T\Big[\underbrace{(\partial\lambda_i/\partial p_L)\mathbf{A}\,\Phi_i}_{(3)} + \lambda_i(\partial\mathbf{A}/\partial p_L)\Phi_i\Big] = \Psi_j^T(\partial\mathbf{B}/\partial p_L)\Phi_i \quad (5.26)$$

To eliminate term (3) of eq. (5.26), we multiply eq. (5.13.1) by $(\partial\lambda_i/\partial p_L)$ to give:

$$\Psi_j^T\mathbf{A}\,\Phi_i(\partial\lambda_i/\partial p_L) = \delta_{ij}(\partial\lambda_i/\partial p_L) \quad (5.28)$$

And substitute term 3 with the right hand side of eq. (5.28) to give :

$$\delta_{ij}(\partial\lambda_i/\partial p_L) + \Psi_j^T\lambda_i(\partial\mathbf{A}/\partial p_L)\Phi_i = \Psi_j^T(\partial\mathbf{B}/\partial p_L)\Phi_i \quad (5.26)$$

For $i = j = k$, eq. (5.26) gives:

$$(\partial\lambda_K/\partial p_L) = \Psi_K^T\Big[(\partial\mathbf{B}/\partial p_L) - \lambda_K(\partial\mathbf{A}/\partial p_L)\Big]\Phi_K \quad (5.29)$$

Having solved for the closed loop eigenvalues λ_k and the left and the right hand eigenvectors, eq. (5.29) yields the first order eigenvalue λ_K sensitivities with respect to p_L.

5.3 EVALUATING THE CONTROL PARAMETERS FOR OPTIMAL ACTIVE DAMPING

We intend to damp out the structural vibrations of the antenna system, using the d.c. servo motor. Consider linear displacement and velocity closed loop feedback, the motor torque Q(t) is a linear function of the displacements (Θ_1 , Θ_2 , Θ_3 , Θ_4) and the velocities ($\dot{\Theta}_1$, $\dot{\Theta}_2$, $\dot{\Theta}_3$, $\dot{\Theta}_4$):

$$Q(t) = Q(\boldsymbol{\Theta}, \dot{\boldsymbol{\Theta}}) = \mathbf{g} \begin{bmatrix} \boldsymbol{\Theta} \\ \dot{\boldsymbol{\Theta}} \end{bmatrix} \tag{5.30}$$

Where $\mathbf{g}$ is the constant 8 x 1 control parameter or gain vector described earlier. We will apply the eigenvalue optimization algorithms, to evaluate the constants g_i of the gain vector. As input parameters, we choose desired damping ratios for the first 4 modes.

The dimensionless damping ratio β_K, for the k-th mode, can be determined as a function of the complex system eigenvalues:

$$\beta_K = \beta_K [\mathrm{Re}(\lambda_K), \ \mathrm{Im}(\lambda_K)] \tag{5.31}$$

The damping ratio on its own, does not specify a particular eigenvalue. Thus, we need the damped natural frequency ω as another parameter:

$$\omega_K = \mathrm{Im}(\lambda_K) \tag{5.32}$$

ω can be evaluated as a function of the undamped natural frequency ω_0 and the damping ratio:

$$\omega_K^2 = \omega_{0K}^2 \, (1 - \beta_K^2) \tag{5.33}$$

The system parameters above define an (8 x 1) object vector $\boldsymbol{\gamma}$, which contains the damped natural frequencies and damping ratios for the first 4 modes:

$$\boldsymbol{\gamma} = \left[\omega_1 , \omega_2 , \omega_3 , \omega_4 , \beta_1 , \beta_2 , \beta_3 , \beta_4 \right]^T \tag{5.34}$$

Parameterization of the closed loop feedback differential equation:

The system of differential equations for the closed loop feedback is denoted by eq. (5.1):

$$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{C}\dot{\mathbf{x}} + \mathbf{K}\mathbf{x} = \mathbf{B}\mathbf{u} \qquad (5.1)$$

We consider linear displacement and velocity feedback control:

$$\mathbf{y}_1 = \mathbf{H}_1\boldsymbol{\Theta} \qquad \mathbf{y}_2 = \mathbf{H}_2\dot{\boldsymbol{\Theta}} \qquad (5.2)$$

$\mathbf{y}_i$ = m_i x 1 feedback vector

$\mathbf{H}_i$ = m_i x 4 feedback matrix

$\boldsymbol{\Theta}$ = 4 x 1 configuration vector $\begin{bmatrix}\Theta_1 & \Theta_2 & \Theta_3 & \Theta_4\end{bmatrix}^T$

(Angular displacements of the mast/antenna system).

We choose the feedback matrices as unity matrices:

$$\mathbf{H}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{H}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (5.35)$$

Introducing (5.35) into (5.2) leads to the feedback vectors:

$$\mathbf{y}_1 = \begin{bmatrix} \Theta_1 \\ \Theta_2 \\ \Theta_3 \\ \Theta_4 \end{bmatrix} \qquad \mathbf{y}_2 = \begin{bmatrix} \dot{\Theta}_1 \\ \dot{\Theta}_2 \\ \dot{\Theta}_3 \\ \dot{\Theta}_4 \end{bmatrix} \qquad (5.36)$$

The 8 x 1 control gain vector **g** consists of the unknowns. The first 4 coefficients are associated with displacement feedback, and the last 4 coefficients are associated with velocity feedback, thus:

$$\mathbf{g} = \big[\underbrace{g_1 \quad g_2 \quad g_3 \quad g_4}_{\boldsymbol{\Theta}} \quad \underbrace{g_5 \quad g_6 \quad g_7 \quad g_8}_{\dot{\boldsymbol{\Theta}}}\big]^T \qquad (5.37)$$

Since the servo motor torque is applied only to two adjacent links, the gain matrices for displacement feedback ($\mathbf{G}_1$) and velocity feedback ($\mathbf{G}_2$) must be (2 x 4) matrices:

$$\mathbf{G}_1 = \begin{bmatrix} g_1 & g_2 & g_3 & g_4 \\ g_1 & g_2 & g_3 & g_4 \end{bmatrix} \qquad \mathbf{G}_2 = \begin{bmatrix} g_5 & g_6 & g_7 & g_8 \\ g_5 & g_6 & g_7 & g_8 \end{bmatrix} \tag{5.38}$$

The control vector is denoted by eq. (5.3):

$$\mathbf{u} = - \left[\mathbf{G}_1\mathbf{y}_1 + \mathbf{G}_2\mathbf{y}_2 + \mathbf{G}_3\mathbf{y}_3 \right] \tag{5.3}$$

Inroducing eqs. (5.36)&(5.38) into eq.(5.3) yields the (2 x 1) control vector:

$$\mathbf{u} = \begin{bmatrix} g_1\Theta_1 + g_2\Theta_2 + g_3\Theta_3 + g_4\Theta_4 + g_5\dot{\Theta}_1 + g_6\dot{\Theta}_2 + g_7\dot{\Theta}_3 + g_8\dot{\Theta}_4 \\ g_1\Theta_1 + g_2\Theta_2 + g_3\Theta_3 + g_4\Theta_4 + g_5\dot{\Theta}_1 + g_6\dot{\Theta}_2 + g_7\dot{\Theta}_3 + g_8\dot{\Theta}_4 \end{bmatrix} \tag{5.39}$$

The control influence matrix describes the application of the control torque to the system:

$$\mathbf{B} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & -1 \\ 0 & 0 \end{bmatrix} \tag{5.40}$$

We then obtain the right hand side of eq. (5.1) which is a (4 x 1) control vector:

$$\mathbf{Bu} = \begin{bmatrix} 0 \\ g_1\Theta_1 + g_2\Theta_2 + g_3\Theta_3 + g_4\Theta_4 + g_5\dot{\Theta}_1 + g_6\dot{\Theta}_2 + g_7\dot{\Theta}_3 + g_8\dot{\Theta}_4 \\ - g_1\Theta_1 - g_2\Theta_2 - g_3\Theta_3 - g_4\Theta_4 - g_5\dot{\Theta}_1 - g_6\dot{\Theta}_2 - g_7\dot{\Theta}_3 - g_8\dot{\Theta}_4 \\ 0 \end{bmatrix} \tag{5.41}$$

The matrices of conservative system are:

Mass matrix $$\mathbf{M} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ & m_{22} & m_{23} & m_{24} \\ \text{sym.} & & m_{33} & m_{34} \\ & & & m_{44} \end{bmatrix} \tag{5.42}$$

Struct. damping matrix (assumed zero) $$\mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.43}$$

Stiffness matrix $$\mathbf{K} = \begin{bmatrix} k_{11} & k_{12} & 0 & 0 \\ & k_{22} & 0 & 0 \\ \text{sym.} & & 0 & 0 \\ & & & k_{44} \end{bmatrix} \tag{5.44}$$

The closed loop system matrices are denoted by eq. (5.5):

$$\overline{\mathbf{M}} = \mathbf{M} \qquad \overline{\mathbf{C}} = \mathbf{C} + \mathbf{B}\mathbf{G}_2\mathbf{H}_2 \qquad \overline{\mathbf{K}} = \mathbf{K} + \mathbf{B}\mathbf{G}_1\mathbf{H}_1 \tag{5.5}$$

Introducing eqs. (5.35), (5.38), (5.40) and (5.42) - (5.44) into eq. (5.5) yields the closed loop system matrices:

Closed loop system matrices:

$$\overline{\mathbf{M}} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ & m_{22} & m_{23} & m_{24} \\ \text{sym.} & & m_{33} & m_{34} \\ & & & m_{44} \end{bmatrix} \tag{5.45}$$

$$\overline{\mathbf{C}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -g_5 & -g_6 & -g_7 & -g_8 \\ g_5 & g_6 & g_7 & g_8 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.46}$$

$$\overline{\mathbf{K}} = \begin{bmatrix} k_{11} & k_{12} & 0 & 0 \\ k_{12}-g_1 & k_{22}-g_2 & -g_3 & -g_4 \\ g_1 & g_2 & g_3 & g_4 \\ 0 & 0 & 0 & k_{44} \end{bmatrix} \tag{5.47}$$

The first order state space differential equation is denoted by eqs. (5.9) & (5.10):

$$\mathbf{A}\dot{\mathbf{z}} = \mathbf{B}\mathbf{z} \tag{5.9}$$

$$\mathbf{z} = \begin{bmatrix} \boldsymbol{\theta} \\ \dot{\boldsymbol{\theta}} \end{bmatrix} \qquad \mathbf{A} = \begin{bmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \overline{\mathbf{M}} \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} \mathbf{0} & \mathbf{M} \\ -\overline{\mathbf{K}} & -\overline{\mathbf{C}} \end{bmatrix} \tag{5.10}$$

Introducing eqs. (5.42) and (5.45) - (5.47) into eq.(5.10) yields the (8 x 8) system matrices of the first order differential equation system (5.9):

System matrices of first order differential equation system:

$$\mathbf{A} = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} & 0 & 0 & 0 & 0 \\ m_{21} & m_{22} & m_{23} & m_{24} & 0 & 0 & 0 & 0 \\ m_{31} & m_{32} & m_{33} & m_{34} & 0 & 0 & 0 & 0 \\ m_{41} & m_{42} & m_{43} & m_{44} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & m_{11} & m_{12} & m_{13} & m_{14} \\ 0 & 0 & 0 & 0 & m_{21} & m_{22} & m_{23} & m_{24} \\ 0 & 0 & 0 & 0 & m_{31} & m_{32} & m_{33} & m_{34} \\ 0 & 0 & 0 & 0 & m_{41} & m_{42} & m_{43} & m_{44} \end{bmatrix} \qquad (5.48)$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 & m_{11} & m_{12} & m_{13} & m_{14} \\ 0 & 0 & 0 & 0 & m_{21} & m_{22} & m_{23} & m_{24} \\ 0 & 0 & 0 & 0 & m_{31} & m_{32} & m_{33} & m_{34} \\ 0 & 0 & 0 & 0 & m_{41} & m_{42} & m_{43} & m_{44} \\ -k_{11} & -k_{12} & 0 & 0 & 0 & 0 & 0 & 0 \\ -k_{12}+g_1 & -k_{22}+g_2 & g_3 & g_4 & g_5 & g_6 & g_7 & g_8 \\ -g_1 & -g_2 & -g_3 & -g_4 & -g_5 & -g_6 & -g_7 & -g_8 \\ 0 & 0 & 0 & -k_{44} & 0 & 0 & 0 & 0 \end{bmatrix} \qquad (5.49)$$

The corresponding state vector is denoted by:

$$\mathbf{z} = \begin{bmatrix} \Theta_1 \\ \Theta_2 \\ \Theta_3 \\ \Theta_4 \\ \dot{\Theta}_1 \\ \dot{\Theta}_2 \\ \dot{\Theta}_3 \\ \dot{\Theta}_4 \end{bmatrix} \qquad (5.50)$$

Eigenvalue and eigenvector problem:

The specific eigenvalue problem of the first order differential equation system (eq.(5.9)) is:

$$\det[\mathbf{A}^{-1}\mathbf{B} - \lambda_K \mathbf{E}] = 0 \qquad (5.51)$$

We obtain 4 complex and 4 complex conjugate eigenvalues (= 8 eigenvalues):

$$\lambda_K = \mathrm{Re}(\lambda_K) \; (+/-) \, \mathrm{Im}(\lambda_K) \qquad k = 1, 2, \ldots 4 \qquad (5.52)$$

Concerning the eigenvalue optimization, the complex conjugate eigenvalues provide no further information. Thus we consider only the eigenvalues with positive imaginary part, which can be labeled according to the ordering:

$$\mathrm{Im}(\lambda_1) < \mathrm{Im}(\lambda_2) < \mathrm{Im}(\lambda_3) < \mathrm{Im}(\lambda_4) \qquad (5.53)$$

The corresponding left and right eigenvectors are obtained from:

$$\lambda_K \mathbf{A}^T \mathbf{\Psi}_K = \mathbf{B}^T \mathbf{\Psi}_K \qquad \lambda_K \mathbf{A}\, \mathbf{\Phi}_K = \mathbf{B}\, \mathbf{\Phi}_K \qquad (5.12.L)\ ,\ (5.12.R)$$

We use the conventional normalizations for the eigenvectors:

$$\mathbf{\Psi}_j^T \mathbf{A}\, \mathbf{\Phi}_i = \delta_{ij} \qquad \mathbf{\Psi}_j^T \mathbf{B}\, \mathbf{\Phi}_i = \lambda_i \delta_{ij} \qquad (5.13.1)\ ,\ (5.13.2)$$

Partial derivatives of the object vector γ with respect to gain vector g:

The sensitivity matrix is denoted by the 8 x 8 Jacobi matrix:

$$\left[\frac{\partial \boldsymbol{\gamma}}{\partial \mathbf{g}}\bigg|_{g_L}\right] = \begin{bmatrix} \frac{\partial \omega_1}{\partial g_1} & \frac{\partial \omega_1}{\partial g_2} & \frac{\partial \omega_1}{\partial g_3} & \frac{\partial \omega_1}{\partial g_4} & \frac{\partial \omega_1}{\partial g_5} & \frac{\partial \omega_1}{\partial g_6} & \frac{\partial \omega_1}{\partial g_7} & \frac{\partial \omega_1}{\partial g_8} \\ \frac{\partial \omega_2}{\partial g_1} & \frac{\partial \omega_2}{\partial g_2} & \frac{\partial \omega_2}{\partial g_3} & \frac{\partial \omega_2}{\partial g_4} & \frac{\partial \omega_2}{\partial g_5} & \frac{\partial \omega_2}{\partial g_6} & \frac{\partial \omega_2}{\partial g_7} & \frac{\partial \omega_2}{\partial g_8} \\ \frac{\partial \omega_3}{\partial g_1} & \frac{\partial \omega_3}{\partial g_2} & \frac{\partial \omega_3}{\partial g_3} & \frac{\partial \omega_3}{\partial g_4} & \frac{\partial \omega_3}{\partial g_5} & \frac{\partial \omega_3}{\partial g_6} & \frac{\partial \omega_3}{\partial g_7} & \frac{\partial \omega_3}{\partial g_8} \\ \frac{\partial \omega_4}{\partial g_1} & \frac{\partial \omega_4}{\partial g_2} & \frac{\partial \omega_4}{\partial g_3} & \frac{\partial \omega_4}{\partial g_4} & \frac{\partial \omega_4}{\partial g_5} & \frac{\partial \omega_4}{\partial g_6} & \frac{\partial \omega_4}{\partial g_7} & \frac{\partial \omega_4}{\partial g_8} \\ \frac{\partial \beta_1}{\partial g_1} & \frac{\partial \beta_1}{\partial g_2} & \frac{\partial \beta_1}{\partial g_3} & \frac{\partial \beta_1}{\partial g_4} & \frac{\partial \beta_1}{\partial g_5} & \frac{\partial \beta_1}{\partial g_6} & \frac{\partial \beta_1}{\partial g_7} & \frac{\partial \beta_1}{\partial g_8} \\ \frac{\partial \beta_2}{\partial g_1} & \frac{\partial \beta_2}{\partial g_2} & \frac{\partial \beta_2}{\partial g_3} & \frac{\partial \beta_2}{\partial g_4} & \frac{\partial \beta_2}{\partial g_5} & \frac{\partial \beta_2}{\partial g_6} & \frac{\partial \beta_2}{\partial g_7} & \frac{\partial \beta_2}{\partial g_8} \\ \frac{\partial \beta_3}{\partial g_1} & \frac{\partial \beta_3}{\partial g_2} & \frac{\partial \beta_3}{\partial g_3} & \frac{\partial \beta_3}{\partial g_4} & \frac{\partial \beta_3}{\partial g_5} & \frac{\partial \beta_3}{\partial g_6} & \frac{\partial \beta_3}{\partial g_7} & \frac{\partial \beta_3}{\partial g_8} \\ \frac{\partial \beta_4}{\partial g_1} & \frac{\partial \beta_4}{\partial g_2} & \frac{\partial \beta_4}{\partial g_3} & \frac{\partial \beta_4}{\partial g_4} & \frac{\partial \beta_4}{\partial g_5} & \frac{\partial \beta_4}{\partial g_6} & \frac{\partial \beta_4}{\partial g_7} & \frac{\partial \beta_4}{\partial g_8} \end{bmatrix} \qquad (5.54)$$

The damped natural frequencies ω_K and the corresponding damping ratios β_K in eq.(5.54) are obtained from the eigenvalues λ_K (see eq. (5.33),(5.34)) :

$$\omega_K = \mathrm{Im}(\lambda_K) \qquad (5.55)$$

$$k = 1, 2, \ldots 4$$

$$\beta_k = -\mathrm{Re}(\lambda_K) \,/\, [\,(\mathrm{Re}(\lambda_K))^2 + (\mathrm{Im}(\lambda_K))^2]^{1/2} \qquad (5.56)$$

Since $\lambda_K = \lambda_K(\mathbf{g})$, the partial derivatives of the damped natural frequencies with respect to a typical element g_L are:

$$\partial \omega_K / \partial g_L = \mathrm{Im}(\partial \lambda_K / \partial g_L) \qquad K = 1, 2, \ldots 4 \quad L = 1, 2, \ldots 8 \tag{5.57}$$

The partial derivatives of the damping ratio β in eq.(5.54) with respect to a typical element g_L are found from eq. (5.54):

$$(\partial \beta_K / \partial g_L) = [Re_K (Re_K Re'_{KL} + Im_K Im'_{KL}) - Re'_{KL}(Re_K^2 + Im_K^2)] / (Re_K^2 + Im_K^2)^{3/2} \tag{5.58}$$

$$K = 1, 2, \ldots 4 \qquad L = 1, 2, \ldots 8$$

Where :

$Re_K = Re(\lambda_K)$ $\qquad Im_K = Im(\lambda_K)$

$Re'_{KL} = Re(\partial \lambda_K / \partial g_L)$ $\qquad Im'_{KL} = Im(\partial \lambda_K / \partial g_L)$

The partial derivatives of eigenvalues are denoted by eq.(5.29):

$$(\partial \lambda_K / \partial p_L) = \mathbf{\Psi}_K^T \left[(\partial \mathbf{B} / \partial p_L) - \lambda_K (\partial \mathbf{A} / \partial p_L) \right] \mathbf{\Phi}_K \tag{5.29}$$

Since the partial derivatives of Matrix $\mathbf{A}$ (eq.(5.48)) are equal to zero :

$$\partial \mathbf{A} / \partial p_L = \mathbf{0} \qquad L = 1, 2, \ldots 8 \tag{5.59}$$

We can simplify eq. (5.29) to:

$$(\partial \lambda_K / \partial p_L) = \mathbf{\Psi}_K^T (\partial \mathbf{B} / \partial p_L) \mathbf{\Phi}_K \tag{5.60}$$

Where the partial derivatives of the matrix **B** (eq.(5.49)) are constant matrices:

$$\partial \mathbf{B}/\partial p_L = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \delta_{1L} & \delta_{2L} & \delta_{3L} & \delta_{4L} & \delta_{5L} & \delta_{6L} & \delta_{7L} & \delta_{8L} \\ -\delta_{1L} & -\delta_{2L} & -\delta_{3L} & -\delta_{4L} & -\delta_{5L} & -\delta_{6L} & -\delta_{7L} & -\delta_{8L} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad (5.61)$$

$L = 1, 2, \ldots 8 \qquad \delta_{ij} =$ Kronecker delta

Iteration procedure :

For each iteration step, we have to solve the linear, real, inhomogenous equation system, to obtain the 8 x 1 Lagrange multiplier vector $\mathbf{\Lambda}$ (eq. (5.24)):

$$\left[\frac{\partial \boldsymbol{\gamma}}{\partial \mathbf{g}}\bigg|_{g_L}\right] \mathbf{W}^{-1} \left[\frac{\partial \boldsymbol{\gamma}}{\partial \mathbf{g}}\bigg|_{g_L}\right]^T \mathbf{\Lambda} = \Delta\boldsymbol{\gamma} \qquad (5.24)$$

We choose the weigting matrix **W** as the 8 x 8 unity Matrix:

$$\mathbf{W} = \mathbf{E} \qquad (5.60)$$

Thus we can write eq. (5.24):

$$\left[\frac{\partial \boldsymbol{\gamma}}{\partial \mathbf{g}}\bigg|_{g_L}\right] \left[\frac{\partial \boldsymbol{\gamma}}{\partial \mathbf{g}}\bigg|_{g_L}\right]^T \mathbf{\Lambda} = \Delta\boldsymbol{\gamma} \qquad (5.61)$$

Where $\Delta\boldsymbol{\gamma}$ is a "small change" of the object vector, and $\mathbf{\Lambda}$ is obtained by solving eq.(5.24) The corresponding change of the gain vector is obtained from eq. (5.23):

$$\Delta\mathbf{g} = \left[\frac{\partial \boldsymbol{\gamma}}{\partial \mathbf{g}}\bigg|_{g_L}\right]^T \mathbf{\Lambda} \qquad (5.23)$$

Finally, we evaluate the new gain vector, for the next iteration step:

$$\mathbf{g}_{new} = \mathbf{g}_{old} + \Delta\mathbf{g} \tag{5.63}$$

From eq. (5.61), we see, that the solution for a desired object vector $\boldsymbol{\gamma}_{object}$ is obtained in terms of small changes $\Delta\boldsymbol{\gamma}$. Thus, we need a sequence of local object vectors $\boldsymbol{\gamma}_i$, leading from an initial object vector $\boldsymbol{\gamma}_0$ to the final object vector $\boldsymbol{\gamma}_n = \boldsymbol{\gamma}_{object}$. Where:

$$\Delta\boldsymbol{\gamma} = \boldsymbol{\gamma}_{i+1} - \boldsymbol{\gamma}_i \tag{5.64}$$

The sequence of local object vectors can be denoted by descrete values of a step parameter α:

$$\boldsymbol{\gamma}_i = \boldsymbol{\gamma}_i(\alpha) \qquad 0 < \alpha \leq 1$$

As an example, one can choose the step parameter $\alpha = 0.1, 0.2, \ldots 1$.

We set the damping ratios of the initial object vector equal to zero. Thus the damping ratios for the k-th mode are:

$$\beta_{K,i} = \alpha \, \beta_{k,object} \qquad 0 < \alpha \leq 1 \tag{5.65}$$

And the corresponding damped natural frequencies are denoted by:

$$\omega_{K,i} = \mathrm{Im}(\lambda_{K,o})\,(1 - \beta_{K,i}^2)^{1/2} \tag{5.66}$$

Where $\lambda_{K,o}$ is the eigenvalue of the k-th mode of the undamped system.

5.4 PROGRAM FOR EIGENVALUE OPTIMIZATION:

Program "EIGOP" applies the eigenvalue optimization approach, which is described above. It evaluates the corresponding gain vector for given damping ratios for the first 4 modes (object vector).

It is the task of the user, to choose a reasonable number of intermediate steps. Too few steps might cause no convergence, and too many steps takes a long calculating time. Typically, the number of intermediate steps is between 10 and 30.

The user has the opportunity to set an initial gain vector. It would speed up the iteration process, if a gain vector is known, which approximately satisfies the desired object vector.

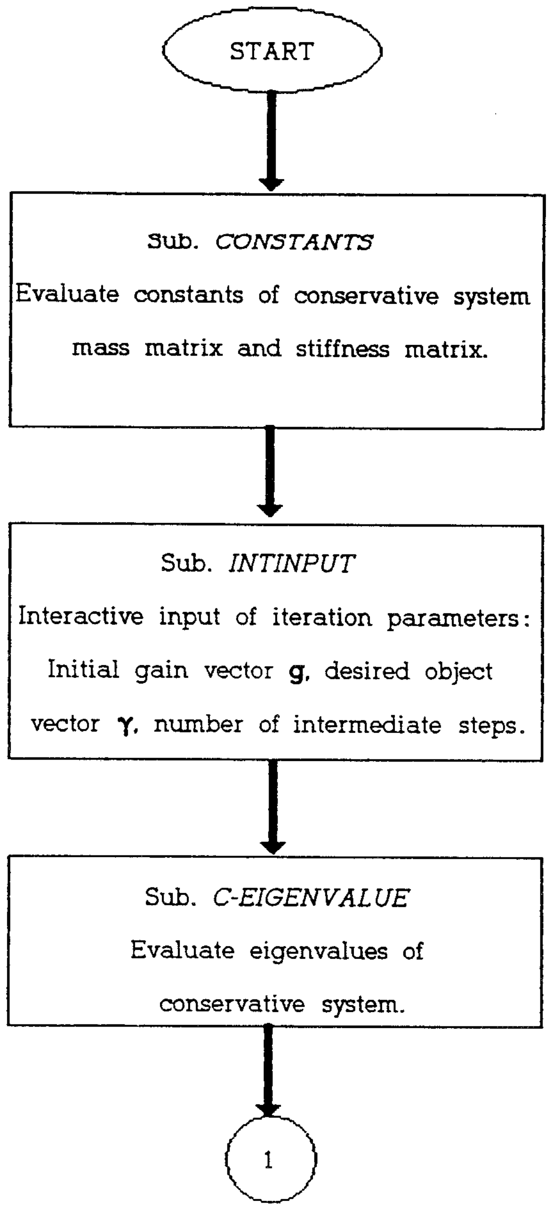
Caution:

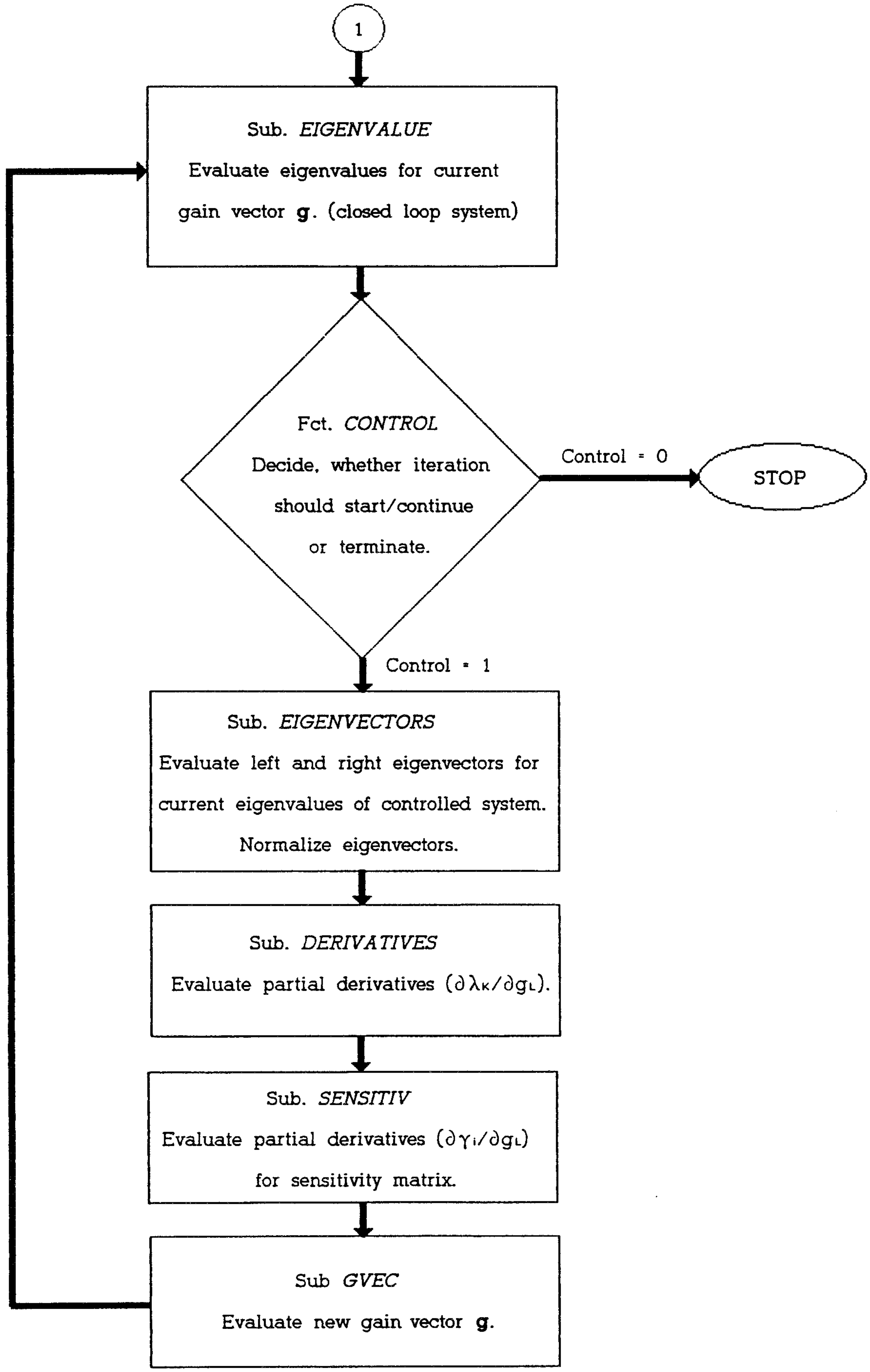
To set an initial gain vector which is equal to zero, causes a singularity. (Reason: The missing constraint at gymbal hinge of the uncontrolled system). Thus, it is necessary to choose at least:

$$\mathbf{g}_{initial} = \left[0 \; , \; 0 \; , \; 0.001 \; , \; 0 \; , \; 0 \; , \; 0 \; , \; 0 \; , \; 0\right]^T \qquad (5.67)$$

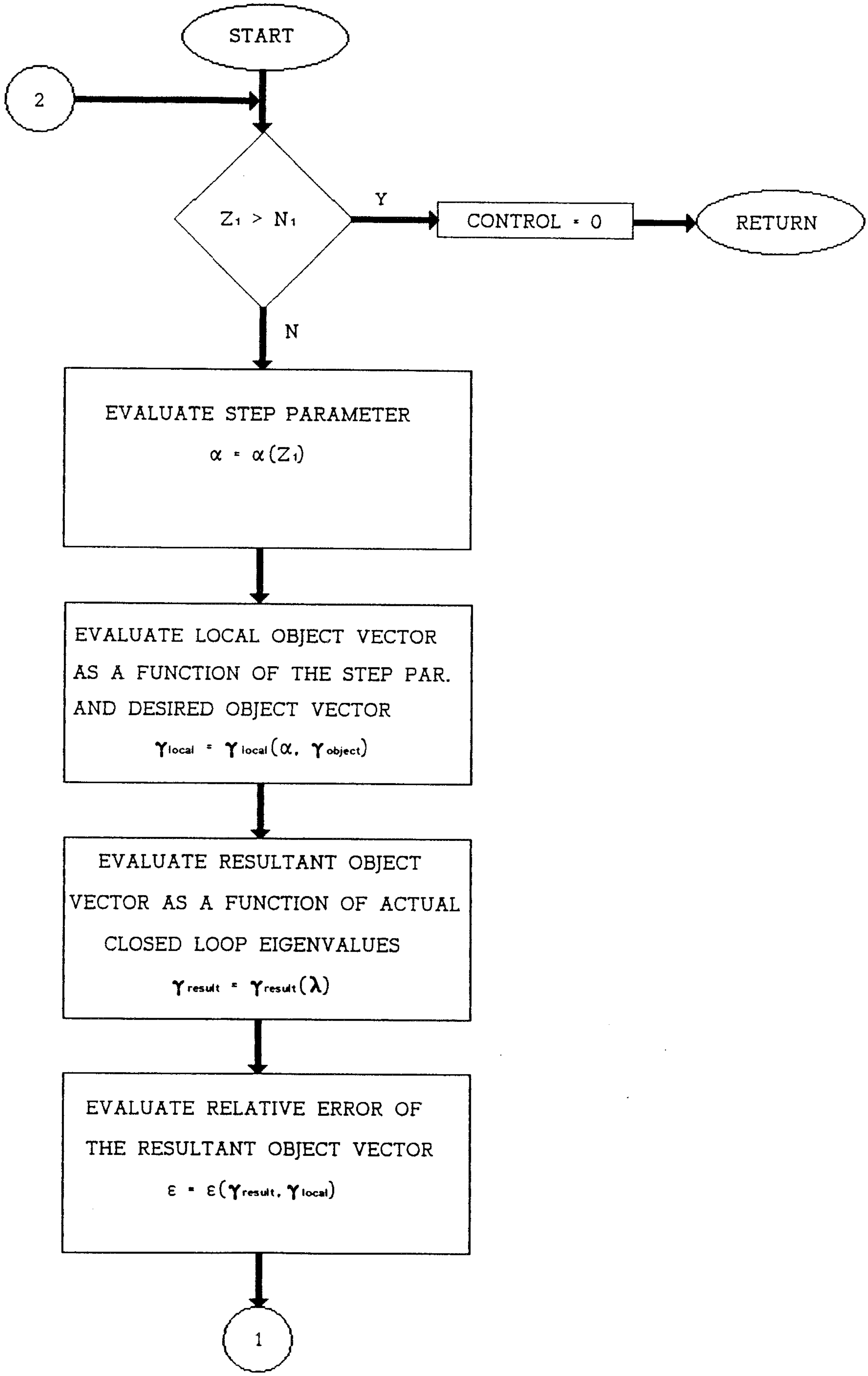
The program is structured for high flexibility. Thus it might not only be used for this particular problem, but also used as a basic program for different eigenvalue optimization applications.

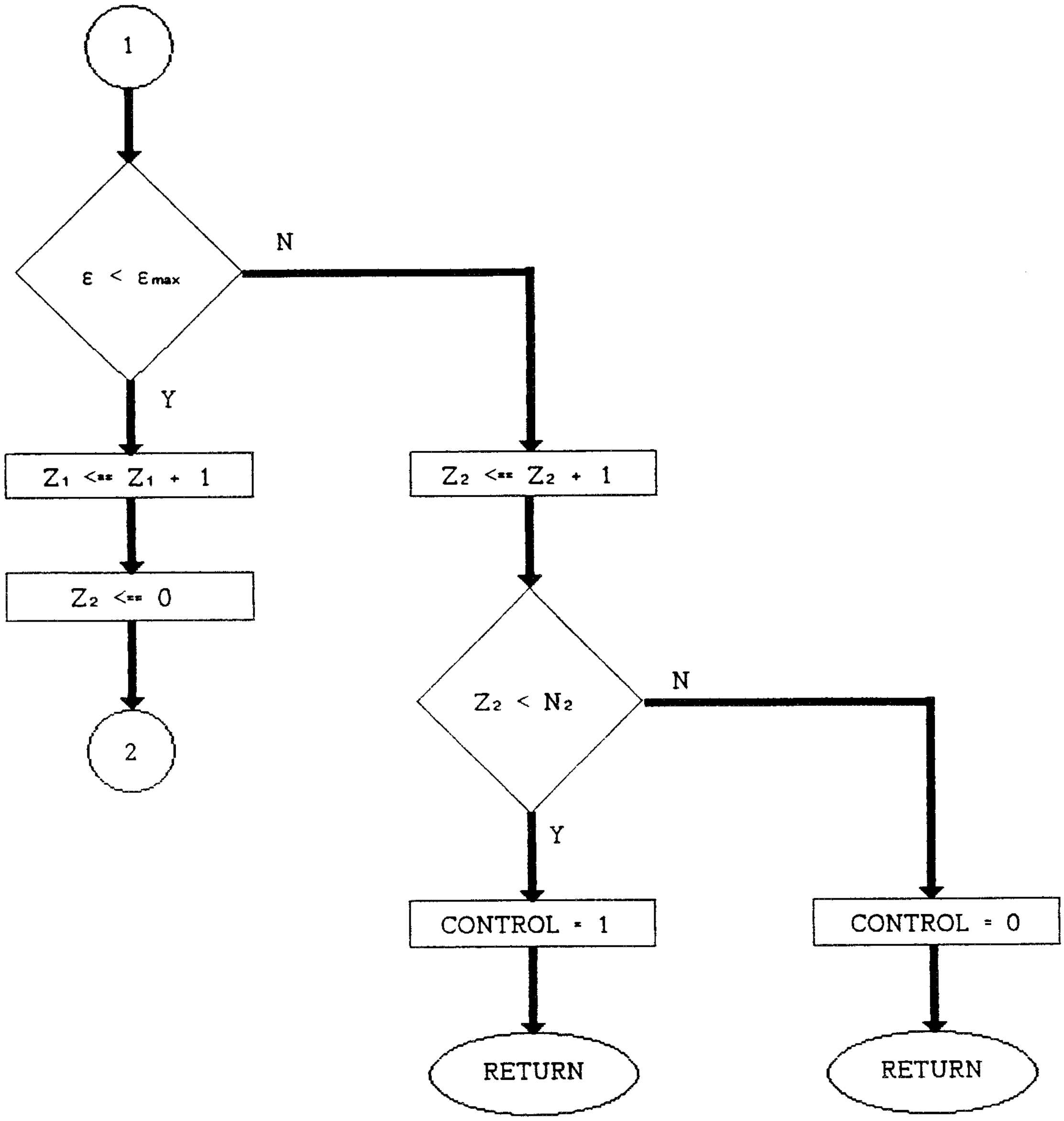
Floating chart of subroutines and functions of the program "EIGOP"




1
Sub. EIGENVALUE
Evaluate eigenvalues for current gain vector g. (closed loop system)
Fct. CONTROL
Decide, whether iteration should start/continue or terminate.
Control = 0
STOP
Control = 1
Sub. EIGENVECTORS
Evaluate left and right eigenvectors for current eigenvalues of controlled system. Normalize eigenvectors.
Sub. DERIVATIVES
Evaluate partial derivatives (∂λK/∂gL).
Sub. SENSITIV
Evaluate partial derivatives (∂γi/∂gL) for sensitivity matrix.
Sub GVEC
Evaluate new gain vector g.

Floating chart of the function CONTROL :





The function "CONTROL" steers the iteration process.

The variable Z_1 is needed to evaluate the local object vector. If the resultant object vector is equal to the local object vector (within the demanded accuracy), Z_1 will be increased by 1 and the succeeding local object vector will be calculated. N_1 is the number of intermediate steps, and is given by the user. If Z_1 exceeds N_1, the desired object vector is evaluated, and the iteration terminates succesfully.
The variable Z_2 counts the number of iteration steps for each local object vector. If N_2 is exceeded, then the process terminates without having gained the solution. (No convergence).

Control = 0 Stop the iteration.

Control = 1 Start/continue the iteration, seeking the solution for the local object vector.

5.5 NUMERICAL RESULTS

We use the program 'EIGOP' to evaluate the corresponding gain vector for desired closed loop damping via server motor.

The motor torque = control torque is denoted by eq. (5.30), which is a linear function of displacements and velocities and the corresponding elements of the gain vector:

$$Q = \mathbf{g} \begin{bmatrix} \boldsymbol{\Theta} \\ \dot{\boldsymbol{\Theta}} \end{bmatrix} \qquad (5.30)$$

$$= g_1\Theta_1 + g_2\Theta_2 + g_3\Theta_3 + g_4\Theta_4 + g_5\dot{\Theta}_1 + g_6\dot{\Theta}_2 + g_7\dot{\Theta}_3 + g_8\dot{\Theta}_8$$

In general we are allowed to demand arbitrary damping ratios. The damping is limited by the maximum motor torque, which is +/- 30Nm. We demand 10% damping for each of the first 4 modes:

$$\beta_1 = \beta_2 = \beta_3 = \beta_4 = 10\% = \beta$$

Thus the desired object vector is:

$$\boldsymbol{\gamma}_{object} = \begin{bmatrix} \omega_1 & \omega_2 & \omega_3 & \omega_4 & 0.1 & 0.1 & 0.1 & 0.1 \end{bmatrix}^T$$

The corresponding damped natural frequencies are evaluated by the program. We choose an initial gain vector (see eq. (5.67)):

$$\mathbf{g}_{initial} = \begin{bmatrix} 0 & 0 & 0.002 & 0 & 0 & 0.003 & 0.003 & 0 \end{bmatrix}$$

And we choose 10 intermediate steps:

$$N_1 = 10$$

The convergence of the iteration is excellent. Only 3 iterations are needed for each intermediate step.

We obtain the final gain vector for 10% damping:

$$\mathbf{g} = \begin{bmatrix} -9.0716\ E-3 \\ 3.6000\ E-3 \\ 2.0000\ E-3 \\ -2.3008\ E-4 \\ -6.4060\ E-2 \\ 1.9892\ E-2 \\ 2.7535\ E-3 \\ 1.5554\ E-3 \end{bmatrix}$$

The time history of the angular displacements, the control torque and the total system energy are plotted in the diagrams for the first 120 seconds. At $t = 0$, the system has just finished the open loop slewing maneuver from chapter 5.4, and it is switched to the closed loop feedback damping maneuver.

Diagr. 5.1	Angular displacement $\Theta_1(t)$, $\beta = 10\%$
Diagr. 5.2	Angular displacement $\Theta_2(t)$, $\beta = 10\%$
Diagr. 5.3	Angular displacement $\Theta_3(t)$, $\beta = 10\%$
Diagr. 5.4	Angular displacement $\Theta_4(t)$, $\beta = 10\%$
Diagr. 5.5	Total sytem energy, $\beta = 10\%$
Diagr. 5.6	Control torque of servo motor, $\beta = 10\%$

We consider the amplitudes of the absolute angular displacement (diagr.5.5) in the beginning of the damping maneuver and after 2 minutes:

$$\hat{\Theta}\,(t = 6\,\text{sec}) = 2350\,\mu\text{rad}$$
$$\hat{\Theta}\,(t = 118\,\text{sec}) = 24\,\mu\text{rad}$$

After 2 minutes, the magnitude of structural vibrations has declined to approximately 1% of the initial value.

The maximum magnitude of the control torque is 23 Nm, which is within the limits.

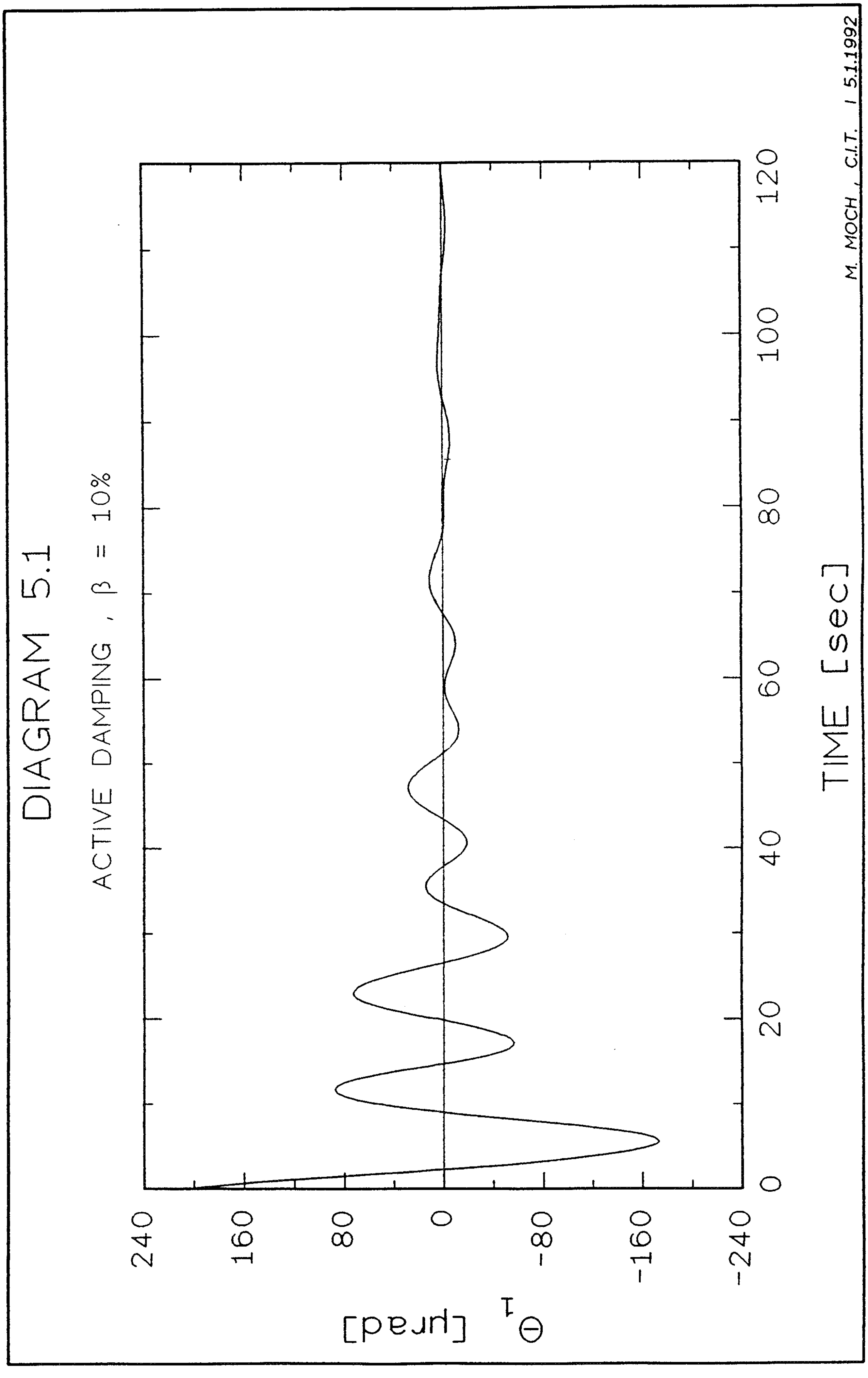


DIAGRAM 5.1
ACTIVE DAMPING , β = 10%
Θ_1 [μrad]
240
160
80
0
-80
-160
-240
0
20
40
60
80
100
120
TIME [sec]
M. MOCH , C.I.T. / 5.1.1992

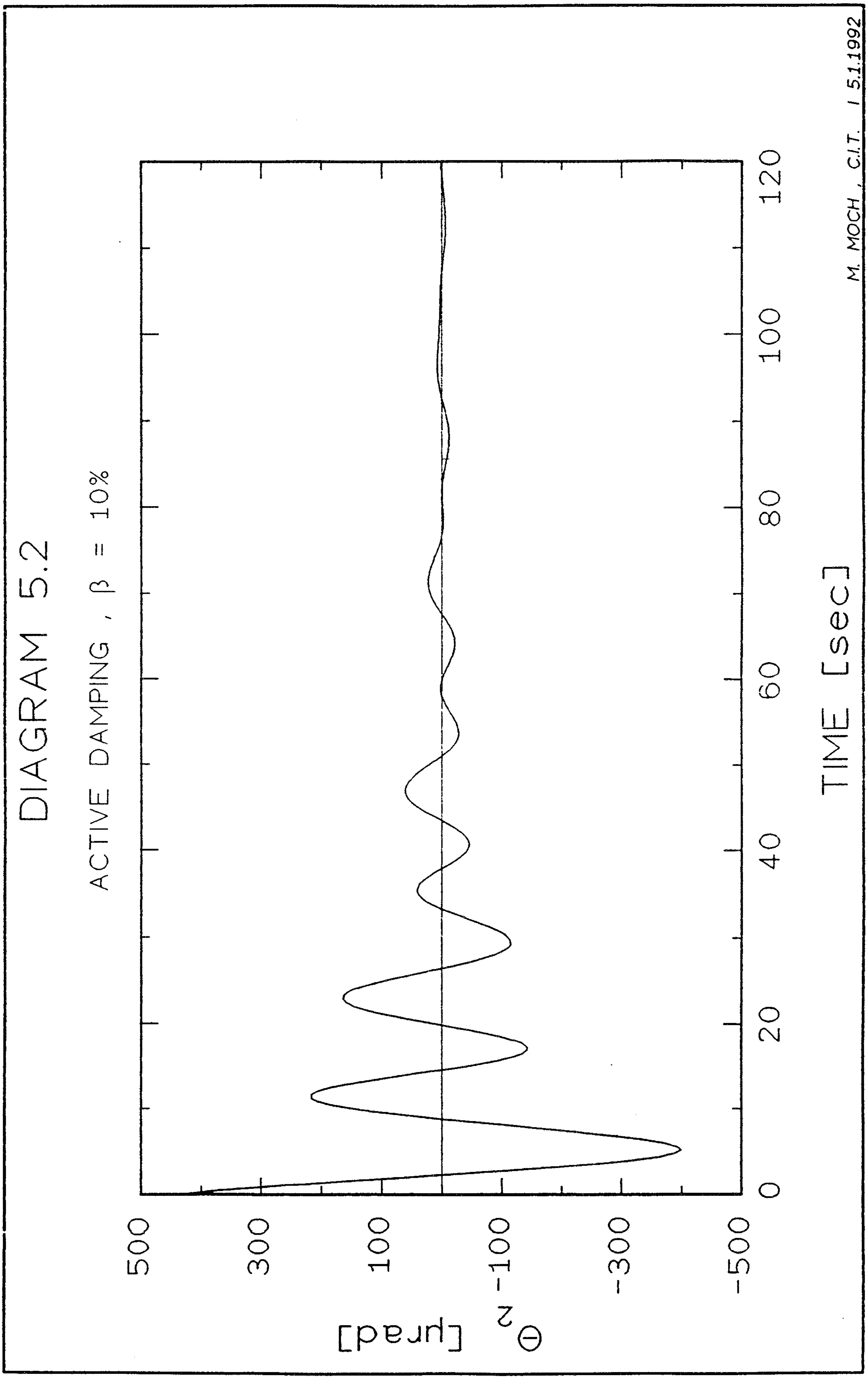


DIAGRAM 5.2
ACTIVE DAMPING , β = 10%
500
300
100
-100
-300
-500
Θ2 [μrad]
0
20
40
60
80
100
120
TIME [sec]
M. MOCH , C.I.T. I 5.1.1992

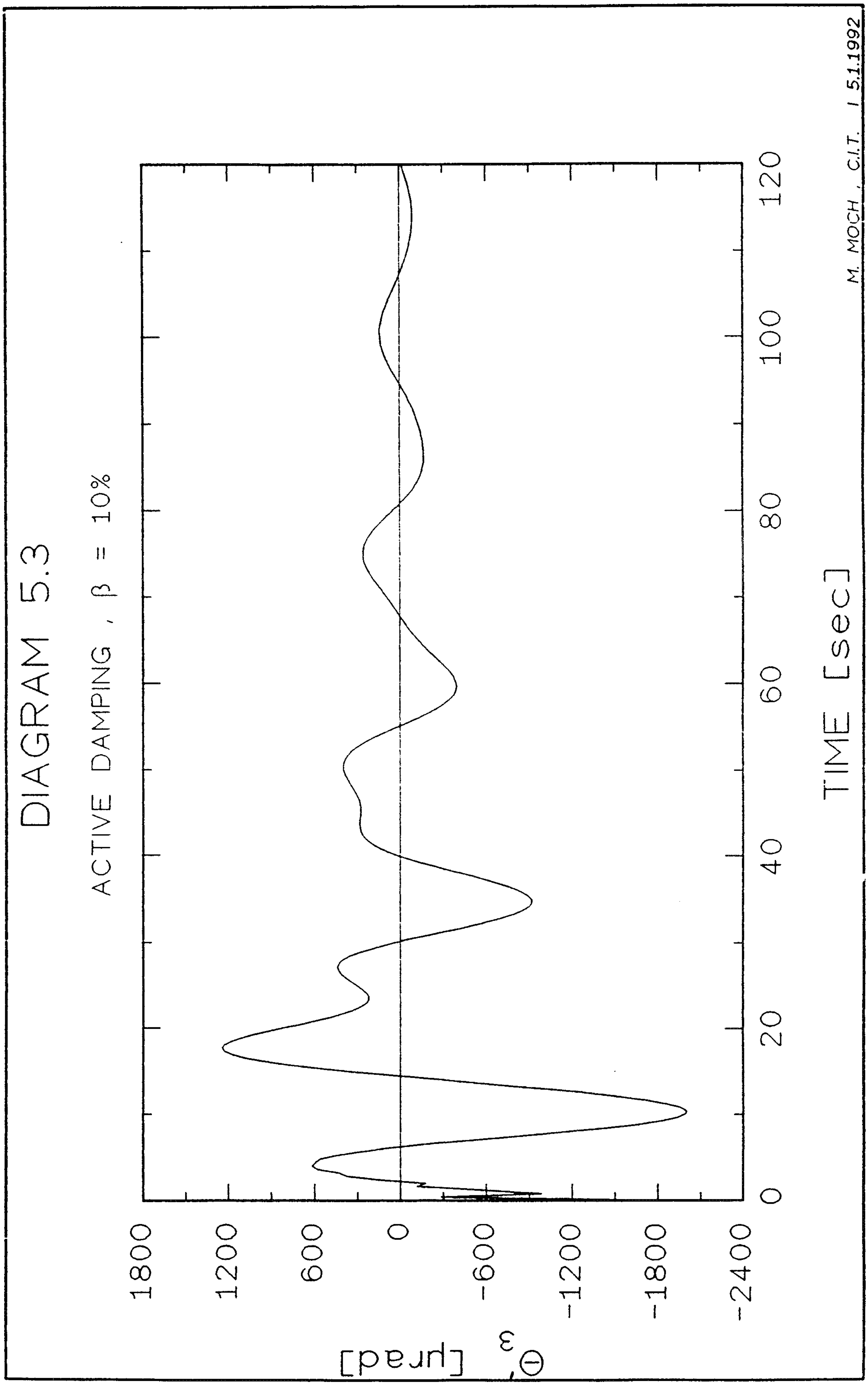


DIAGRAM 5.3
ACTIVE DAMPING , β = 10%
1800
1200
600
0
-600
-1200
-1800
-2400
Θ3 [μrad]
0
20
40
60
80
100
120
TIME [sec]
M. MOCH , C.I.T. 15.1.1992

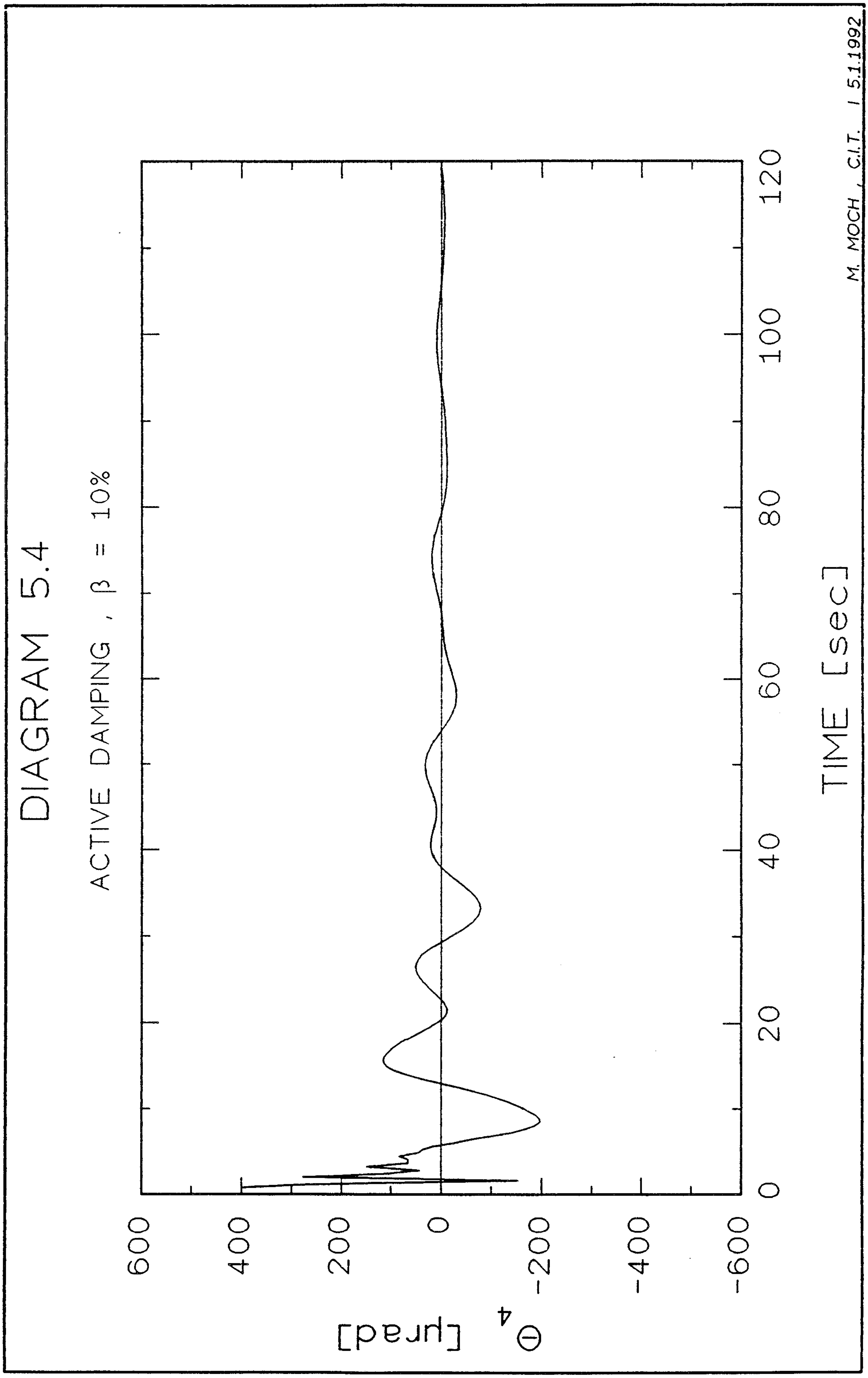


DIAGRAM 5.4
ACTIVE DAMPING , β = 10%
600
400
200
0
-200
-400
-600
Θ4 [μrad]
0
20
40
60
80
100
120
TIME [sec]
M. MOCH , C.I.T. / 5.1.1992

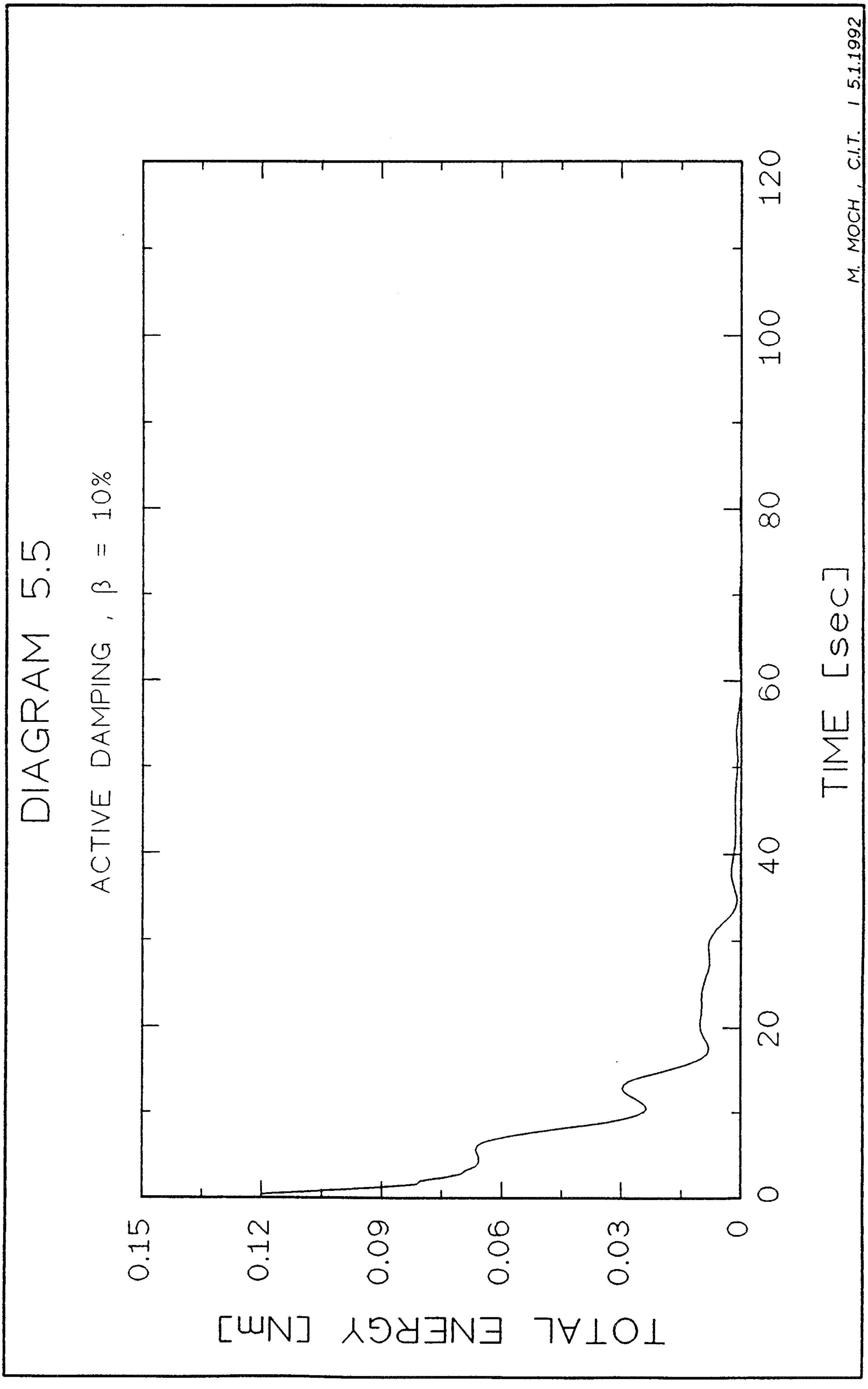


DIAGRAM 5.5
ACTIVE DAMPING , β = 10%
TOTAL ENERGY [Nm]
0.15
0.12
0.09
0.06
0.03
0
0
20
40
60
80
100
120
TIME [sec]
M. MOCH , C.I.T. / 5.1.1992

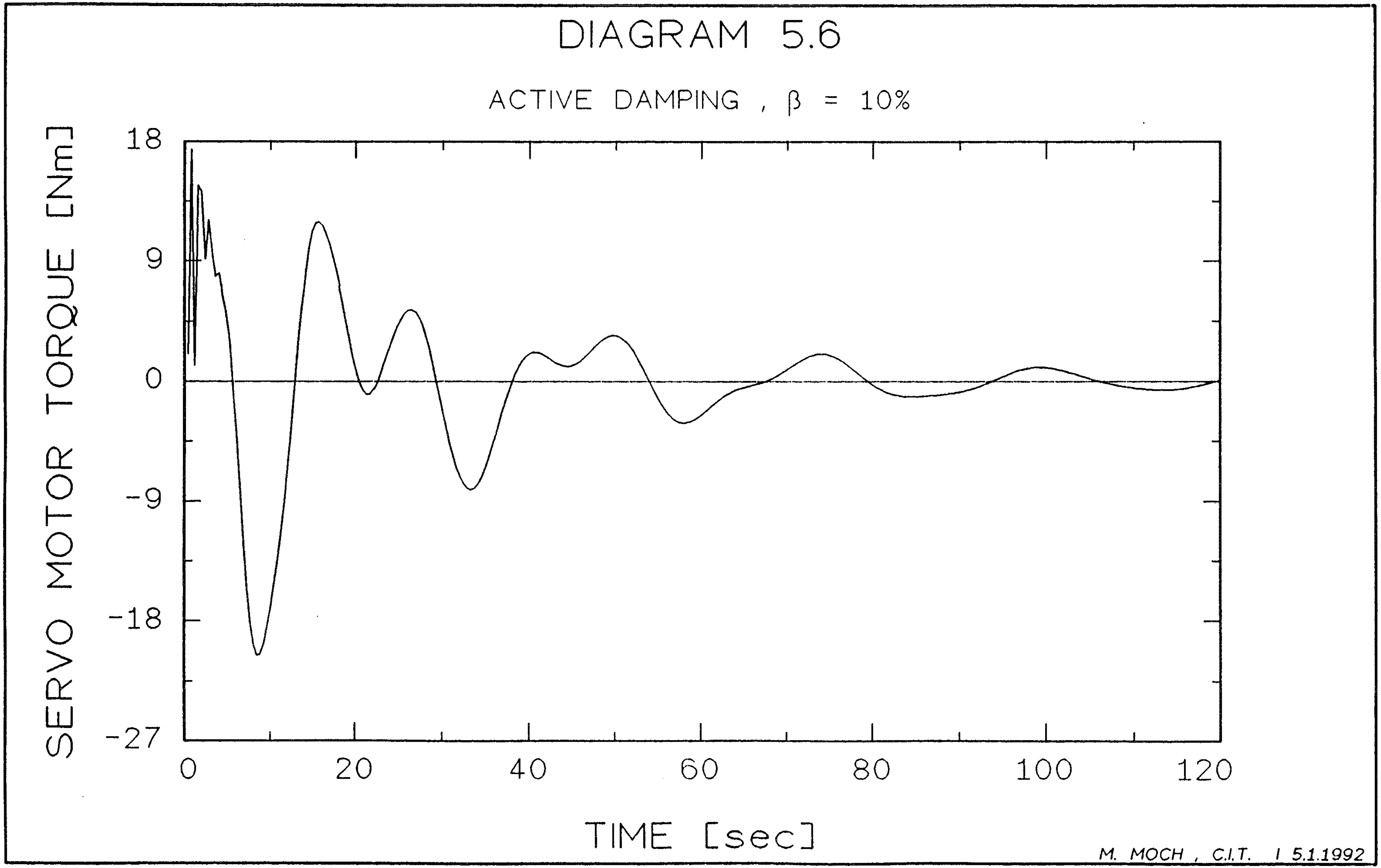


DIAGRAM 5.6
ACTIVE DAMPING , β = 10%
SERVO MOTOR TORQUE [Nm]
18
9
0
-9
-18
-27
0
20
40
60
80
100
120
TIME [sec]
M. MOCH , C.I.T. / 5.1.1992

6. CONCLUSIONS

The Lagrangian approach has been used to model a complex flexible antenna/astro-mast system as a simpler alternative to that employed in ref. (2) where the finite element model was used to formulate the system mass and stiffness matrices. In order to keep the open and closed loop dynamic models to an reasonable size an reduced model having four degrees of freedom was selected. This implied that the higher modes of vibrational response were neglected. The aim of the research was to investigate the possibility of using the d. c. servo motor in a closed loop system so as to damp out the oscillations produced by the step function motor torque required for acceleration and deceleration. It is considered to be a justifyable simplification in restricting the model to four degrees of freedom since the main aim was not to model the very small amplitude oscillations in the higher modes but to obtain a simple model for the purpose of designing an effective feedback control system which would damp out the residual oscillations at the end of the maneuver in a specific time.

The Lagrangian method was applied yielding a set of non-linear equations having time varying mass matrix arising from the large angle Θ_3. The reduced dynamic model is of practical use in verifying the control system design and obtaining the maximum dynamic response during the maneuver. The maximum bending moment in the astro-mast is also of interest to the mast designer and is easily obtained using the stiffness of the torsion spring at the mast/shuttle interface and the angle Θ_1.

During the maneuver the equations of motion were highly nonlinear and hence no true modes of vibration of the system are present, however if the angle Θ_3 changes slowly due to the small servo motor torque, then for small increments of time the system will be linear. Thus the system natural frequencies and mode shapes during the maneuver phase are of general interest to the system designer, even though modal analysis is not used.

It has has been demonstrated that by using state feedback in a closed loop system the d. c. servo motor, whith was responsible for producing the system response, is also able to effectively damp out the residual oscillations in about two minutes without overloading the servo motor. Thus a complex system can be controlled and its vibration level quickly reduced to zero using a single actuator provided the state of the system can be measured at appropriate locations to obtain Θ_i , with $i = 1, 2, 3, 4$.

The main aim of the closed loop analysis was to obtain effective damping in the four degrees of freedom. A study of the various methods of pole placement was made and it was found that the eigenvalue optimization method of D. S. Bodden and J. L. Junkins ref. (2) was a straight forward and computationally efficient method of the determining system feedback gains to provide specific damping. The interesting thing is the verification that application of a control torque at a single location is easily able to damp out all the modes of oscillation effectively.

The pitch maneuver was performed as an open loop operation because of the occurence of motor saturation due to the high torques present, see ref. (4) and ref. (2). Also it is not important to control the system during the maneuver but only at the termination of the maneuver. Experiments on antenna maneuvers in space are time consuming and effective damping of the systems oscillations are essential to the accurate pointing of the antenna and the overall reduction in the duration of a particular test series.

In order to simply the analysis it was assumed that the rotational inertia of the shuttle was sufficenty great that rotation of the shuttle could be ignored. subsequent elementary calculation of the shuttle rotation due to the torque of the astro-mast proved that this assumption was completely justified.

APPENDIX A: REFERENCES

[1] NASA report CR 172095, 1988 , Control of flexible structures 2 (COFS-2) . The Charles Stark Draper Laboratory, Inc.

[2] D. S. Bodden, J. L. Junkins , Eigenvalue optimization algorithms for structure/ controller design iterations , 1985 , Journal of guidance and control Vol. 8 No. 6 Nov-Dec 1985.

[3] College of Aeronautics Cranfield , Dynamics of flexible space structures , 1990 .

[4] S. Battaia, F. Magne , Maneuvering dynamics of shuttle based flexible mast , 1991 , Thesis at College of Aeronautics Cranfield.

[5] Meirovitch L. , Introduction to dynamics and control , 1985.

[6] Barnett S. , Matrices in control theory , 1971.

[7] Collar A. R. , Matrices and engineering dynamics , 1987.

[8] Newland D. E. , Mechanical vibrations analysis and computation , 1989.

Rotation of the shuttle, caused by vibrations of the antenna system:

This section shows that the assumption of zero shuttle rotation is justified.
Equilibrium of torque at the mast base:

$$I_s \ddot{\Theta}_s = -\lambda_1 \hat{\Theta}_1 \sin \omega t$$

where:

I_s = moment of inertia of shuttle

Θ_s = rotary displacement of shuttle

$\hat{\Theta}_1$ = amplitude of displacement

Integrating yields the rotary displacement of the shuttle:

$$I_s \Theta_s = (\lambda_1 \hat{\Theta}_1 / \omega^2) \sin \omega t$$

The amplitude of displacement is obtained for $\sin \omega t = 1$:

$$\hat{\Theta}_s = \lambda_1 \hat{\Theta}_1 / I_s \omega^2$$

where:

$$\omega = 2\pi / T$$

From diagram 3.1, we read for $t > \tau_2$: $T = 10.8\,\text{sec}$ and $\hat{\Theta}_1 = 212\,\mu\text{rad}$.
The moment of inertia of the shuttle is $I_{yy} = 8.9134\,E\,6\ \text{kgm}^2$. Thus we obtain the amplitude of rotary displacement of the shuttle:

$$\hat{\Theta}_s = 131\,\mu\text{rad} = 0.0075°$$

which shows that the maximum rotation of the shuttle is extremely small.
The maximum bending moment at the mast base is approximately 400 Nm. This obtained from $M = \lambda_1 \theta_1 (\max)$.

APPENDIX C : OBTAINING FEEDBACK SIGNALS

In section (5) we require feedback signals:

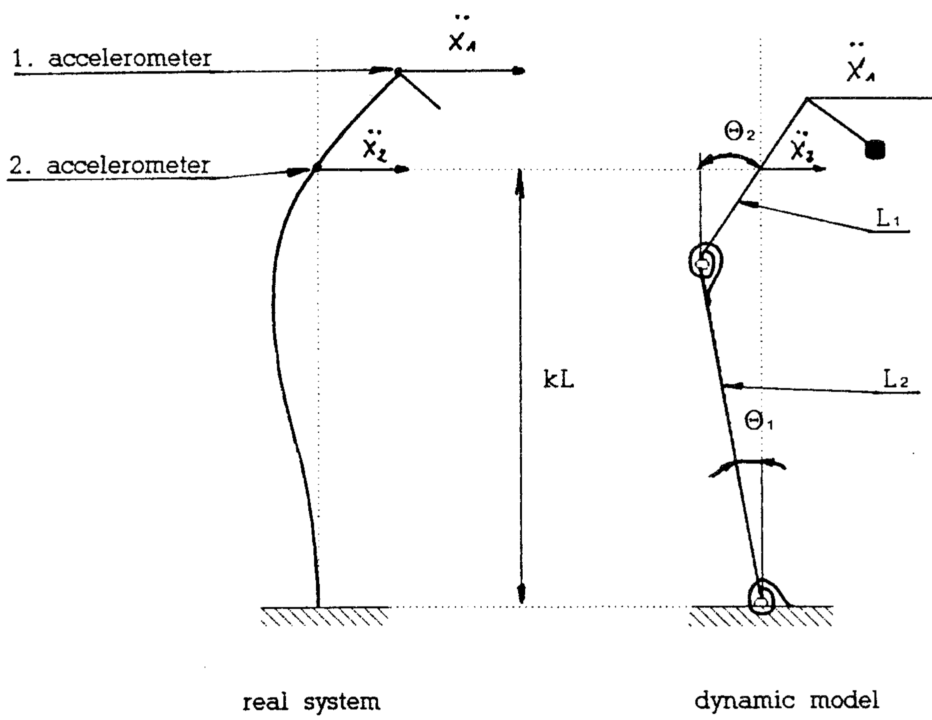
$$\boldsymbol{\Theta} = \begin{bmatrix} \Theta_1 & \Theta_2 & \Theta_3 & \Theta_4 \end{bmatrix}^T$$

The displacements of the Lagrangian coordinates of the dynamic model are proportional to the angular accelerations:

$$\ddot{\boldsymbol{\Theta}} = \begin{bmatrix} \ddot{\Theta}_1 & \ddot{\Theta}_2 & \ddot{\Theta}_3 & \ddot{\Theta}_4 \end{bmatrix}^T$$

Accelerations can be measured by accelerometers, mounted on the astro-mast. To obtain the corresponding values of $\ddot{\boldsymbol{\Theta}}$, we have to transform the signals, obtained from the real system.

As an example we determine $\ddot{\Theta}_1$ and $\ddot{\Theta}_2$ of the mast dynamic model. Therefore we place two accelerometers on the astro-mast. We choose places, which are equally defined for the real system and for the model. The first accelerometer is placed of course at the top of the mast, and the second one is placed at the crossover point of the second mode shape, which is equal for the real system and for the dynamic model (see section 2).



We neglect vertical motion. Thus the acceleration at the top is:

$$\ddot{x}_1 = (d^2/dt^2)(\Theta_1 L_1 + \Theta_2 L_2)$$

$$\ddot{x}_1 = \ddot{\Theta}_1 L_1 + \ddot{\Theta}_2 L_2$$

The corresponding acceleration at the crossover point is:

$$\ddot{x}_2 = \ddot{\Theta}_1 L_1 + \ddot{\Theta}_2 (kL - L_1)$$

$$L = L_1 + L_2$$

Where $\ddot{x}_1$ and $\ddot{x}_2$ are measured from the real system by filtering out the higher modes. $\ddot{\Theta}_1$ and $\dot{\Theta}_2$ are obtained by solving the system of linear equations:

$$\begin{bmatrix} L_1 & L_2 \\ L_1 & kL - L_1 \end{bmatrix} \cdot \begin{bmatrix} \ddot{\Theta}_1 \\ \ddot{\Theta}_2 \end{bmatrix} = \begin{bmatrix} \ddot{x}_1 \\ \ddot{x}_2 \end{bmatrix}$$

APPENDIX D : SOURCE CODE OF PROGRAMS

This appendix contains the source code of the programs, which were developped for this thesis. All programs are written in FORTRAN 77, but input/output routines may be different, since they depend on the machine.

Subroutines taken from the mathematical software library University Aachen/Germany, which are needed for the programs above:

PROGRAM IVP
****

COMPUTES DISPLACEMENTS AND VELOCITIES FOR OPEN LOOP PITCHING MANEUVER.
REQUIRES SUBROUTINES FOR INITIALVALUE PROBLEMS AND INVERTING OF REAL
MATRICES.

```
      PROGRAM IVP
C     ***********

C     DECLARE PROGRAM SPECIFIC VARIABLES:
C     ----------------------------------
C     N:      ANZAHL DER GLEICHUNGEN
C     M:      ANZAHL DER STÜTZSTELLEN DER LÖSUNGSFUNKTION

      EXTERNAL DGL
      INTEGER N,M,INDEX,IFMAX,IFANZ,IFEHL,K
      REAL XA,XE,XK,HK,XENDE,EPSABS,EPSREL
      PARAMETER (N=8,M=300)
      REAL YK(1:N),A(1:N+2,0:M)
      CHARACTER NAME*7


C     DECLARE PROBLEM SPECIFIC VARIABLES:
C     -----------------------------------
      REAL LM1,LM2,LM3,LM4,TO,ALPHA
      COMMON/CONST/C1,C2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,
     1            C14,C15,C16,C17,C18,C19,LM1,LM2,LM3,LM4,TO

C     CALL SUBROUTINES:
C     -----------------
      CALL EINGABE(N,XK,HKM,YK,INDEX,IFMAX,EPSABS,EPSREL,XA,XE,K,NAME)
      HK=HKM
      CALL FACT(N,XK,YK)
      CALL TE(M,N,A,XK,YK)
      CALL UEBERGABE(N,M,XK,YK,A,HK,IFANZ)

1     IF (ABS(XK/XE-1).LT.0.0001) THEN
        CALL SPEICHERN(K,M,N,A,NAME)
        GOTO 2
      ELSE  IF (IFEHL.NE.0) THEN
        CALL FEHLER (IFEHL)
        CALL SPEICHERN(K,M,N,A,NAME)
        GOTO 2
      ELSE
        CALL ZIELWERT(XENDE,XA,XE,K)
        HK=HKM
        CALL AWP(XK, HK, YK, N, DGL, XENDE, EPSABS, EPSREL,
     1                 INDEX, IFMAX, IFANZ, IFEHL)
```

```
        CALL TE(M,N,A,XK,YK)
        CALL UEBERGABE(N,M,XK,YK,A,HK,IFANZ)
      ENDIF
      GOTO 1

2     CONTINUE
      END



      SUBROUTINE FACT(N,XK,YK)
C     ************************
C     COMPUTE PROBLEM SPECIFC CONSTANTS

      INTEGER N
      REAL XK
      REAL YK(1:N)
      REAL LL,L0,L1,L2,L23,LA1,MM,M0,M3,M4,RR,LM1,LM2,LM3,LM4,TO,ALPHA
      COMMON/CONST/C1,C2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,
     1              C14,C15,C16,C17,C18,C19,LM1,LM2,LM3,LM4,TO

C     DEFINE CONSTANTS:
C     -----------------
      LL=60.693
      L0=3.15
      L1=46.575
      L2=14.118
      L23=5.22
      LA1=11.432
      MM=15.931
      M0=274.58
      M3=59.453
      M4=362.522
      RR=3.237

      LM1=1.867E6
      LM2=1.150E6
C      LM3=0.03E6
      LM3=0.
      LM4=0.071442E6
      TO=0.

      LM1=LM1/(MM*LL**3)
      LM2=LM2/(MM*LL**3)
      LM3=LM3/(MM*LL**3)
      LM4=LM4/(MM*LL**3)
      TO=TO/(MM*LL**3)
```

```
C     COMPUTE CONSTANTS:
C     ------------------
      C1=RR**3/(6*(RR+1)**3)+RR**2/(2*(RR+1)**3)+(M0*RR**2)/
     1 (MM*LL*2*(RR+1)**2)+(M3*L1**2)/(2*MM*LL**3)+(M4*L1**2)/
     2 (2*MM*LL**3)
      C2=RR**3/(6*(RR+1)**3)+(M0/(MM*LL))*(1+(L0*(RR+1)/LL)**2)
     2 +(M3/(2*MM*LL**3))*(L2**2+L23**2)+(M4/(2*MM*LL**3))
     3 *(L2**2+L23**2+LA1**2+2*L23*LA1)
      C3=(M3*L2*L23)/(MM*LL**3)
      C4=(M4*(L2*L23+L2*LA1))/(MM*LL**3)
      C5=(M3*L23**2)/(2*MM*LL**3)+(M4*(L23**2+LA1**2+2*L23*LA1))/
     1 (2*MM*LL**3)
      C6=(M4*LA1**2)/(2*MM*LL**3)
      C7=R/(2*(RR+1)**3)+(M0*R)/(MM*LL*(RR+1)**2)+(M3*L1*L2)/(MM*LL**3)
     1 +(M4*L1*L2)/(MM*LL**3)
      C8=(M3*L1*L23)/(MM*LL**3)
      C9=(M4*(L1*L23+L1*LA1))/(MM*LL**3)
      C10=(M3*L1*L23)/(MM*LL**3)
      C11=(M4*(L1*L23+L1*LA1))/(MM*LL**3)
      C12=(M4*L1*LA1)/(MM*LL**3)
      C13=(M3*L23)/(MM*LL**3)+(M4*(L23**2+LA1**2+2*L23*LA1))/(MM*LL**3)
      C14=(M3*L2*L23)/(MM*LL**3)
      C15=(M4*(L2*L23+L2*LA1))/(MM*LL**3)
      C16=(M4*(LA1**2+L23*LA1))/(MM*LL**3)
      C17=(M4*L2*LA1)/(MM*LL**3)
      C18=(M4*(LA1**2+L23*LA1))/(MM*LL**3)

      C19=(TO*MM*LL**3)/(L23**2*M3+(L23+LA1)**2*M4)

      END


      SUBROUTINE EINGABE(N,XK,HK,YK,INDEX,IFMAX,EPSABS,EPSREL,XA,XE,
     1                   K,NAME)
C     ***************************************************************
C     INTERACTIVE INPUT

C     XK:     START VALUE OF TIME VARIABLE
C     HK:     PACE OF TIME VARIABLE FOR ITERATION
C     XENDE:  FINAL VALUE OF TIME VARIABLE
C     EPSABS: ABSOLUTE ERROR
C     EPSREL: RELATIVE ERROR
C     INDEX:  CHOICE OF ITERATION FORMULA     0 : RUNGE KUTTA
C                                            <> 0 : FORMULA OF ENGLAND
C     IFMAX:  MAXIMUM NUMBER OF ITERATION STEPS
C     YK:     SOLUTION VECTOR

      INTEGER N,K,INDEX,IFMAX
      REAL XA,XE,XK,HK,EPSABS,EPSREL
```

```
      REAL YK(1:N)
      CHARACTER NAME*7

      EPSABS=0.0001
      EPSREL=0
      INDEX=1
      IFMAX=5000

      WRITE(*,1000)
      WRITE(*,1001)
      READ(*,*) XA,XE,K

C      WRITE(*,1002)
C      READ(*,*) YK

C     INITIAL SOLUTION VECTOR :
C     -------------------------
      YK(1)=-155.481E-6
      YK(2)=-281.0657E-6
      YK(3)=1.56904
      YK(4)=-645.777E-6
      YK(5)=-11.9926E-6
      YK(6)=-34.2818E-6
      YK(7)=19731.44E-6
      YK(8)=2696.293E-6

      WRITE(*,1003)
      READ(*,*) HK
      WRITE(*,1004)
      READ(*,1005) NAME

      XK=XA

1000  FORMAT(' INTERVAL AND NUMBER OF STEPS OF TIME VARIABLE')
1001  FORMAT(' XK, XEND, N : ')
1002  FORMAT(' INITIAL SOLUTION VECTOR YK(8 X 1): ')
1003  FORMAT(' PACE OF TIME VARIABLE HK : ')
1004  FORMAT(' NAME OF FILE FOR STORING RESULTS (MAX 7 CHARACT.): ')
1005  FORMAT(A)

      END

      SUBROUTINE ZIELWERT(XENDE,XA,XE,K)
C     **********************************
C     EVALUATE NEW TIME VALUE

      INTEGER K,I
      REAL XA,XE,XENDE
      SAVE I
```

```
      I=I+1
      XENDE=XA+FLOAT(I)*(XE-XA)/FLOAT(K)

      END


      SUBROUTINE UEBERGABE(N,M,XK,YK,A,HK,IFANZ)
C     ********************************************
C     STORE DISCRETE RESULTS IN MATRIX

      INTEGER N,M,I,IFANZ
      REAL XK,HK
      REAL YK(1:N),A(1:N+2,0:M)
      SAVE I

      A(1,I)=XK
      DO 1,J=1,N,1
        A(J+1,I)=1E6*YK(J)
1     CONTINUE
      WRITE(*,*) I,IFANZ,A(1,I),A(4,I),A(10,I)
      I=I+1

      END


      SUBROUTINE DGL(X,Y,N,F)
C     *************************
C     DETERMINE FIRST ORDER STATE SPACE DIFFERENTIAL EQUATION.

      INTEGER N
      REAL X
      REAL Y(1:N),F(1:N)

      INTEGER IFEHL,FLAG
      REAL S1,S2
      REAL MM(1:4,1:4),MI(1:4,1:4),B(1:4)
      INTEGER MX(1:4),MY(1:4)

      REAL LM1,LM2,LM3,LM4,TO,ALPHA
      REAL SH,SS,CH,CS,DH,DS
      COMMON/CONST/C1,C2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,
     1             C14,C15,C16,C17,C18,C19,LM1,LM2,LM3,LM4,TO

C     ACCELERATION:
C     -------------
C      ALPHA=0.5*C19*X**2
C      ALPHA_=C19*X
```

```
C       DECELERATION:
C       --------------
C        ALPHA0=0.785398
C        ALPHA_0=2.14801E-2
C        TIME0=73.135
C        ALPHA=ALPHA0+ALPHA_0*(X-TIME0)+0.5*C19*(X-TIME0)**2
C        ALPHA_=ALPHA_0+C19*(X-TIME0)

C       CONSTANT PITCH ANGLE:
C       ---------------------
        ALPHA=1.57079633
        ALPHA_=0

        SH=SIN(Y(2)+Y(3))
        SS=SIN(Y(2)+Y(3)+Y(4))
        CH=COS(Y(2)+Y(3))
        CS=COS(Y(2)+Y(3)+Y(4))
        DH=Y(6)+Y(7)
        DS=Y(6)+Y(7)+Y(8)

C       MASS MATRIX:
C       ------------
        MM(1,1)=2*C1
        MM(1,2)=C7+C8*SH+C9*SS
        MM(1,3)=C10*SH+C11*SS
        MM(1,4)=C12*SS
        MM(2,1)=MM(1,2)
        MM(2,2)=2*(C2+C3*SH+C4*SS)
        MM(2,3)=C13+C14*SH+C15*SS
        MM(2,4)=C16+C17*SS
        MM(3,1)=MM(1,3)
        MM(3,2)=MM(2,3)
        MM(3,3)=2*C5
        MM(3,4)=C18
        MM(4,1)=MM(1,4)
        MM(4,2)=MM(2,4)
        MM(4,3)=MM(3,4)
        MM(4,4)=2*C6


C       RIGHT HAND SIDE OF DIFFERENTIAL EQUATION:
C       -----------------------------------------
        B(1)=-DH*CH*(C8*Y(6)+C10*Y(7))-DS*CS*(C9*Y(6)+C11*Y(7)+C12*Y(8))
     1       -LM1*Y(1)-LM2*(Y(1)-Y(2))
        B(2)=-DH*CH*(C8*Y(5)+2*C3*Y(6)+C14*Y(7))
     2       -DS*CS*(C9*Y(5)+2*C4*Y(6)+C15*Y(7)+C17*Y(8))
     3       +CH*(C3*Y(6)**2+C8*Y(5)*Y(6)+C10*Y(5)*Y(7)+C14*Y(6)*Y(7))
     4       +CS*(C4*Y(6)**2+C9*Y(5)*Y(6)+C11*Y(5)*Y(7)+C15*Y(6)*Y(7)
     5       +C17*Y(6)*Y(8)+C12*Y(5)*Y(8))-LM2*(Y(2)-Y(1))-TO
```

```
      B(3)=-DH*CH*(C10*Y(5)+C14*Y(6))-DS*CS*(C11*Y(5)+C15*Y(6))
     1      +CH*(C3*Y(6)**2+C8*Y(5)*Y(6)+C10*Y(5)*Y(7)+C14*Y(6)*Y(7))
     2      +CS*(C4*Y(6)**2+C9*Y(5)*Y(6)+C11*Y(5)*Y(7)+C12*Y(5)*Y(8)
     3      +C15*Y(6)*Y(7)+C17*Y(6)*Y(8))-LM3*(Y(3)-ALPHA)+TO
      B(4)=-DS*CS*(C12*Y(5)+C17*Y(6))
     1      +CS*(C4*Y(6)**2+C9*Y(5)*Y(6)+C11*Y(5)*Y(7)+C12*Y(5)*Y(8)
     2      +C15*Y(6)*Y(7)+C17*Y(6)*Y(8))-LM4*Y(4)


C     INVERT MASS MATRIX:
C     -------------------
      CALL PIVOT(MM,4,4,MI,S1,S2,IFEHL,MX,MY,WERT)

      IF (IFEHL.EQ.2) THEN
        WRITE(*,*)' MASSEN-MATRIX SINGULÄR'
        STOP
      ENDIF
      IF (S1.GT.0.001.OR.S2.GT.0.001) THEN
        WRITE(*,*)' INVERTIERUNG ZU UNGENAU'
        SUMME1=0
        SUMME2=0
        DO 1,I=1,4
          DO 2,J=1,4
            SUMME1=SUMME1+ABS(MM(I,J))
            SUMME2=SUMME2+ABS(MI(I,J))
2         CONTINUE
1       CONTINUE
        WRITE(*,*)' S1,S2,SUMME MM, SUMME MI: '
        WRITE(*,*)S1,S2,SUMME1,SUMME2
        STOP
      ENDIF

C     FIRST ORDER DIFFERENTIAL EQUATION IN STATE SPACE FORM:
C     ------------------------------------------------------
      F(1)=Y(5)
      F(2)=Y(6)
      F(3)=Y(7)
      F(4)=Y(8)
      F(5)=MI(1,1)*B(1)+MI(1,2)*B(2)+MI(1,3)*B(3)+MI(1,4)*B(4)
      F(6)=MI(2,1)*B(1)+MI(2,2)*B(2)+MI(2,3)*B(3)+MI(2,4)*B(4)
      F(7)=MI(3,1)*B(1)+MI(3,2)*B(2)+MI(3,3)*B(3)+MI(3,4)*B(4)
      F(8)=MI(4,1)*B(1)+MI(4,2)*B(2)+MI(4,3)*B(3)+MI(4,4)*B(4)

      END
```

```
      SUBROUTINE TE(M,N,A,X,YK)
C     *************************
C     COMPUTE TOTAL ENERGY:
C     ---------------------
      INTEGER M,N
      REAL MM,LL
      REAL A(1:N+2,0:M),YK(1:N)
      REAL LM1,LM2,LM3,LM4,TO,ALPHA,ALPHA_
      COMMON/CONST/C1,C2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,
     1              C14,C15,C16,C17,C18,C19,LM1,LM2,LM3,LM4,TO

      SAVE I
      MM=15.931
      LL=60.693

C     ACCELERATION:
C     -------------
C      ALPHA=0.5*C19*X**2
C      ALPHA_=C19*X

C     DECELERATION:
C     -------------
      ALPHA0=0.785398
      ALPHA_0=2.14801E-2
      TIME0=73.135
      ALPHA=ALPHA0+ALPHA_0*(X-TIME0)+0.5*C19*(X-TIME0)**2
      ALPHA_=ALPHA_0+C19*(X-TIME0)

C     CONSTANT PITCH ANGLE:
C     ---------------------
      ALPHA=1.57079633
      ALPHA_=0


      Y1=YK(1)
      Y2=YK(2)
      Y3=YK(3)
      Y4=YK(4)
      Y5=YK(5)
      Y6=YK(6)
      Y7=YK(7)-ALPHA_
      Y8=YK(8)


      SH1=SIN(Y2+Y3)
      SH2=SIN(Y2+Y3+Y4)
```

```
      A(10,I)=(C1*Y5**2+(C2+C3*SH1+C4*SH2)*Y6**2+C5*Y7**2+C6*Y8**2
     1          +(C7+C8*SH1+C9*SH2)*Y5*Y6+(C10*SH1+C11*SH2)*Y5*Y7
     2          +C12*SH2*Y5*Y8+(C13+C14*SH1+C15*SH2)*Y6*Y7
     3          +(C16+C17*SH2)*Y6*Y8+C18*Y7*Y8
     4          +0.5*(LM1*Y1**2+LM2*(Y2-Y1)**2+LM3*(Y3-ALPHA)**2
     5          +LM4*Y4**2))*MM*LL**3

      WRITE(*,1000) A(10,I)

      I=I+1

1000  FORMAT(' TOTAL ENERGY = ',E10.3)
      END




      SUBROUTINE SPEICHERN(K,M,N,A,NAME)
C     **********************************
C     STORE RESULTS ON FILE.

      INTEGER K,M,N
      REAL A(1:N+2,0:M)
      CHARACTER ORDNER*23,NAME*7,NAMEXT*8,DATEI*35,TEXT1*7,
     1          TEXT*19,EXT*1


      ORDNER='F:ÜPLOTFITÜPLOTFIT.DATÜ'
      TEXT1='X-TEXT:'

      DO 20,J=2,N+2,1
        EXT=CHAR(J+63)
        NAMEXT=EXT//NAME
        DATEI=ORDNER//NAMEXT
        TEXT=TEXT1//NAMEXT
        OPEN(UNIT=1,FILE=DATEI,STATUS='UNKNOWN')
        WRITE(1,1004) TEXT
        WRITE(1,1001)
        DO 10,I=0,K,1
          WRITE(1,*) A(1,I),A(J,I)
10      CONTINUE
        WRITE(1,1002)
        CLOSE (UNIT=1)
20    CONTINUE

1001  FORMAT('DATA: X,Y')
1002  FORMAT('END')
1004  FORMAT(A19)

      END
```

```
      SUBROUTINE FEHLER(IFEHL)
C     ************************
C     ERROR INDICATOR.

      INTEGER IFEHL

      IF (IFEHL.EQ.0) THEN
        WRITE(*,*)' VERFAHREN ERFOLGREICH'
      ELSE IF (IFEHL.EQ.1) THEN
        WRITE(*,*)' FEHLERSCHRANKEN ZU KLEIN'
      ELSE IF (IFEHL.EQ.2) THEN
        WRITE(*,*)' ANFANGSWERT <= ZIELWERT:  XENDE <= XK'
      ELSE IF (IFEHL.EQ.3) THEN
        WRITE(*,*)' SCHRITTWEITE HK <= 0'
      ELSE IF (IFEHL.EQ.4) THEN
        WRITE(*,*)' ANZAHL DER GLEICHUNGEN GRÖßER ALS 20'
      ELSE IF (IFEHL.EQ.5) THEN
        WRITE(*,*)' ANZAHL DER ZULÄSSIGEN FUNKTIONSAUSWERTUNGEN NICHT'
        WRITE(*,*)' AUSREICHEND FÜR GEFORDERTE GENAUIGKEIT'
      ENDIF

      END
```

PROGRAM IVPC
****

COMPUTES DISPLACEMENTS AND VELOCITIES FOR FIXED PITCH ANGLE.
OPEN LOOP OR CLOSED LOOP FEEDBACK. REQUIRES SUBROUTINES FOR
INITIAL VALUE PROBLEMS AND INVERTING OF REAL MATRICES.

```
      PROGRAM IVPC
C     ************

C     PROGRAM SPECIFIC VARIABLES:
C     ---------------------------
C     N:     ANZAHL DER GLEICHUNGEN
C     M:     ANZAHL DER STÜTZSTELLEN DER LÖSUNGSFUNKTION

      EXTERNAL DGL
      INTEGER N,M,INDEX,IFMAX,IFANZ,IFEHL,K
      REAL XA,XE,XK,HK,XENDE,EPSABS,EPSREL
      PARAMETER (N=8,M=400)
      REAL YK(1:N),A(1:N+3,0:M)
      CHARACTER NAME*7


C     PROBLEM SPECIFIC VARIABLES:
C     -----------------------------
      REAL MIK(4,4),MID(4,4),CONST(8)
      REAL C1,C2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,C14,C15,C16,
     1      C17,C18,LM1,LM2,LM3,LM4,DF
      COMMON/CONST/MIK,MID,C1,C2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,
     1               C14,C15,C16,C17,C18,LM1,LM2,LM3,LM4,DF

C     CALL OF SUBROUTINES:
C     --------------------
      CALL EINGABE(N,XK,HKM,YK,INDEX,IFMAX,EPSABS,EPSREL,XA,XE,K,NAME)

      HK=HKM
      CALL CONSTANTS(N/2,CONST)
      CALL TE(M,N,A,XK,YK)
      CALL TORQUE(M,N,A,YK,CONST)
      CALL UEBERGABE(N,M,XK,YK,A,HK,IFANZ)

1     IF (ABS(XK/XE-1).LT.0.0001) THEN
        CALL SPEICHERN(K,M,N,A,NAME)
        GOTO 2
      ELSE  IF (IFEHL.NE.0) THEN
        CALL FEHLER (IFEHL)
        CALL SPEICHERN(K,M,N,A,NAME)
        GOTO 2
```

```
      ELSE
        CALL ZIELWERT(XENDE,XA,XE,K)
        HK=HKM
        CALL AWP(XK, HK, YK, N, DGL, XENDE, EPSABS, EPSREL,
     1                   INDEX, IFMAX, IFANZ, IFEHL)
        CALL TE(M,N,A,XK,YK)
        CALL TORQUE(M,N,A,YK,CONST)
        CALL UEBERGABE(N,M,XK,YK,A,HK,IFANZ)
      ENDIF
      GOTO 1

2     CONTINUE
1000  FORMAT(A)
      END


      SUBROUTINE CONSTANTS(DIM,CONST)
C     ===============================
C     COMPUTE CLOSED LOOP SYSTEM MATRICES.

      INTEGER DIM,IFEHL,FLAG,MX(4),MY(4)
      REAL S1,S2,FACT
      REAL MM(4,4),MI(4,4),K(4,4),D(4,4),F1(4,4),F2(4,4),CONST(8)
      REAL K_(4,4),D_(4,4),MIK(4,4),MID(4,4)

      REAL LL,L0,L1,L2,L23,LA1,MA,M0,M3,M4,RR,LM1,LM2,LM3,LM4,DF
      REAL C1,C2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,C14,C15,C16,C17
     1    C18
      CHARACTER NAME*8
      COMMON/CONST/MIK,MID,C1,C2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,
     1              C14,C15,C16,C17,C18,LM1,LM2,LM3,LM4,DF


C     DEFINE PROBLEM SPECIFIC CONSTANTS:
C     ---------------------------------
      LL=60.693
      L0=3.15
      L1=46.575
      L2=14.118
      L23=5.22
      LA1=11.432
      MA=15.931
      M0=274.58
      M3=59.453
      M4=362.522
      RR=3.237
```

```
      LM1=1.867E6
      LM2=1.150E6
      LM3=0.
      LM4=0.071442E6
C      DF=2E3

      LM1=LM1/(MA*LL**3)
      LM2=LM2/(MA*LL**3)
      LM3=LM3/(MA*LL**3)
      LM4=LM4/(MA*LL**3)
      DF=DF/(MA*LL**3)


C     COMPUTE PROBLEM SPECIFIC CONSTANTS:
C     ------------------------------------
      C1=RR**3/(6*(RR+1)**3)+RR**2/(2*(RR+1)**3)+(M0*RR**2)/
     1 (MA*LL*2*(RR+1)**2)+(M3*L1**2)/(2*MA*LL**3)+(M4*L1**2)/
     2 (2*MA*LL**3)
      C2=RR**3/(6*(RR+1)**3)+(M0/(MA*LL))*(1+(L0*(RR+1)/LL)**2)
     2 +(M3/(2*MA*LL**3))*(L2**2+L23**2)+(M4/(2*MA*LL**3))
     3 *(L2**2+L23**2+LA1**2+2*L23*LA1)
      C3=(M3*L2*L23)/(MA*LL**3)
      C4=(M4*(L2*L23+L2*LA1))/(MA*LL**3)
      C5=(M3*L23**2)/(2*MA*LL**3)+(M4*(L23**2+LA1**2+2*L23*LA1))/
     1 (2*MA*LL**3)
      C6=(M4*LA1**2)/(2*MA*LL**3)
      C7=R/(2*(RR+1)**3)+(M0*R)/(MA*LL*(RR+1)**2)+(M3*L1*L2)/(MA*LL**3)
     1 +(M4*L1*L2)/(MA*LL**3)
      C8=(M3*L1*L23)/(MA*LL**3)
      C9=(M4*(L1*L23+L1*LA1))/(MA*LL**3)
      C10=(M3*L1*L23)/(MA*LL**3)
      C11=(M4*(L1*L23+L1*LA1))/(MA*LL**3)
      C12=(M4*L1*LA1)/(MA*LL**3)
      C13=(M3*L23)/(MA*LL**3)+(M4*(L23**2+LA1**2+2*L23*LA1))/(MA*LL**3)
      C14=(M3*L2*L23)/(MA*LL**3)
      C15=(M4*(L2*L23+L2*LA1))/(MA*LL**3)
      C16=(M4*(LA1**2+L23*LA1))/(MA*LL**3)
      C17=(M4*L2*LA1)/(MA*LL**3)
      C18=(M4*(LA1**2+L23*LA1))/(MA*LL**3)
```

```
C     MASS MATRIX:
C     ------------
      MM(1,1)=2*C1
      MM(1,2)=C7+C8+C9
      MM(1,3)=C10+C11
      MM(1,4)=C12
      MM(2,1)=MM(1,2)
      MM(2,2)=2*(C2+C3+C4)
      MM(2,3)=C13+C14+C15
      MM(2,4)=C16+C17
      MM(3,1)=MM(1,3)
      MM(3,2)=MM(2,3)
      MM(3,3)=2*C5
      MM(3,4)=C18
      MM(4,1)=MM(1,4)
      MM(4,2)=MM(2,4)
      MM(4,3)=MM(3,4)
      MM(4,4)=2*C6


C     STIFFNESS MATRIX:
C     -----------------
      K(1,1)=LM1+LM2
      K(1,2)=-LM2
      K(1,3)=0
      K(1,4)=0
      K(2,1)=K(1,2)
      K(2,2)=LM2
      K(2,3)=0
      K(2,4)=0
      K(3,1)=K(1,3)
      K(3,2)=K(2,3)
      K(3,3)=LM3
      K(3,4)=0
      K(4,1)=K(1,4)
      K(4,2)=K(2,4)
      K(4,3)=K(3,4)
      K(4,4)=LM4
```

```
C     DAMPING MATRIX:
C     ---------------
      D(1,1)=0
      D(1,2)=0
      D(1,3)=0
      D(1,4)=0
      D(2,1)=D(1,2)
      D(2,2)=0
      D(2,3)=0
      D(2,4)=0
      D(3,1)=D(1,3)
      D(3,2)=D(2,3)
      D(3,3)=DF
      D(3,4)=0
      D(4,1)=D(1,4)
      D(4,2)=D(2,4)
      D(4,3)=D(3,4)
      D(4,4)=0


C     INPUT OF PARAMETER VECTOR:
C     --------------------------
      FACT=1

      DO 1,N=1,4
        DO 2,M=1,4
          F1(N,M)=0
          F2(N,M)=0
2       CONTINUE
1     CONTINUE


C      PARAMETER VECTOR FOR ACTIVE DAMPING:
C      ------------------------------------
      WRITE(*,*)' DATEINAME FÜR PARAMETERVEKTOR: '
      READ(*,1000) NAME
      OPEN(1,FILE='F:ÜPLOTFITÜAWP.DATÜ'//NAME)
      READ(1,*)CONST
      CLOSE(1)

      DO 5,N=1,4
5     CONST(N)=FACT*CONST(N)
```

```
C     STIFFNESS:
C     ---------
      F1(2,1)=CONST(1)
      F1(2,2)=CONST(2)
      F1(2,3)=CONST(3)
      F1(2,4)=CONST(4)
      F1(3,1)=-CONST(1)
      F1(3,2)=-CONST(2)
      F1(3,3)=-CONST(3)
      F1(3,4)=-CONST(4)

C     DAMPING:
C     -------
      F2(2,1)=CONST(5)
      F2(2,2)=CONST(6)
      F2(2,3)=CONST(7)
      F2(2,4)=CONST(8)
      F2(3,1)=-CONST(5)
      F2(3,2)=-CONST(6)
      F2(3,3)=-CONST(7)
      F2(3,4)=-CONST(8)


C     COMPUTE INVERSE MASS MATRIX:
C     ----------------------------
      CALL PIVOT(MM,DIM,DIM,MI,S1,S2,IFEHL,MX,MY,WERT)

      IF (IFEHL.EQ.2) THEN
        WRITE(*,*)' MASSEN-MATRIX SINGULÄR'
        STOP
      ENDIF
      IF (S1.GT.0.001.OR.S2.GT.0.001) THEN
        WRITE(*,*)' INVERTIERUNG ZU UNGENAU'
        SUMME1=0
        SUMME2=0
        DO 3,I=1,4
          DO 4,J=1,4
            SUMME1=SUMME1+ABS(MM(I,J))
            SUMME2=SUMME2+ABS(MI(I,J))
4         CONTINUE
3       CONTINUE
        WRITE(*,*)' S1,S2,SUMME MM, SUMME MI: '
        WRITE(*,*)S1,S2,SUMME1,SUMME2
        STOP
      ENDIF
```

```
C     COMPUTE CLOSED LOOP SYSTEM MATRICES:
C     -----------------------------------
      DO 24,N=1,DIM
        DO 25,M=1,DIM
          K_(N,M)=K(N,M)-F1(N,M)
          D_(N,M)=D(N,M)-F2(N,M)
          MIK(N,M)=0
          MID(N,M)=0
25      CONTINUE
24    CONTINUE


C     PRODUCT OF INVERSE MASS MATRIX * STIFFNESS MATRIX:
C     -------------------------------------------------
      DO 10,N=1,DIM
        DO 11 M=1,DIM
          DO 12,I=1,DIM
            MIK(N,M)=MIK(N,M)+MI(N,I)*K_(I,M)
12        CONTINUE
11      CONTINUE
10    CONTINUE


C     PRODUCT OF INVERSE MASS MATRIX * DAMPING MATRIX:
C     -----------------------------------------------
      DO 17,N=1,DIM
        DO 18 M=1,DIM
          DO 19,I=1,DIM
            MID(N,M)=MID(N,M)+MI(N,I)*D_(I,M)
19        CONTINUE
18      CONTINUE
17    CONTINUE

1000  FORMAT (A)

      END



      SUBROUTINE EINGABE(N,XK,HK,YK,INDEX,IFMAX,EPSABS,EPSREL,XA,XE,
     1                   K,NAME)
C     *****************************************************************
C     INTERACTIVE INPUT

C     XK:     START VALUE OF TIME VARIABLE
C     HK:     PACE OF TIME VARIABLE FOR ITERATION
C     XENDE:  FINAL VALUE OF TIME VARIABLE
C     EPSABS: ABSOLUTE ERROR
C     EPSREL: RELATIVE ERROR
```

```
C      INDEX:  CHOICE OF ITERATION FORMULA     0 : RUNGE KUTTA
C                                             <> 0 : FORMULA OF ENGLAND
C      IFMAX:  MAXIMUM NUMBER OF ITERATION STEPS
C      YK:      SOLUTION VECTOR

       INTEGER N,K,INDEX,IFMAX
       REAL XA,XE,XK,HK,EPSABS,EPSREL
       REAL YK(1:N)
       CHARACTER NAME*7

       EPSABS=0.0001
       EPSREL=0
       INDEX=1
       IFMAX=5000

       WRITE(*,1000)
       WRITE(*,1001)
       READ(*,*) XA,XE,K

C       WRITE(*,1002)
C       READ(*,*) YK

C      INITIAL SOLUTION VECTOR :
C      -------------------------
       YK(1)=-155.481E-6
       YK(2)=-281.0657E-6
       YK(3)=1.56904
       YK(4)=-645.777E-6
       YK(5)=-11.9926E-6
       YK(6)=-34.2818E-6
       YK(7)=19731.44E-6
       YK(8)=2696.293E-6

       WRITE(*,1003)
       READ(*,*) HK
       WRITE(*,1004)
       READ(*,1005) NAME

       XK=XA

1000  FORMAT(' INTERVAL AND NUMBER OF STEPS OF TIME VARIABLE')
1001  FORMAT(' XK, XEND, N : ')
1002  FORMAT(' INITIAL SOLUTION VECTOR YK(8 X 1): ')
1003  FORMAT(' PACE OF TIME VARIABLE HK : ')
1004  FORMAT(' NAME OF FILE FOR STORING RESULTS (MAX 7 CHARACT.): ')
1005  FORMAT(A)

       END
```

```
      SUBROUTINE ZIELWERT(XENDE,XA,XE,K)
C     **********************************
C     EVALUATE NEW TIME VALUE

      INTEGER K,I
      REAL XA,XE,XENDE
      SAVE I

      I=I+1
      XENDE=XA+FLOAT(I)*(XE-XA)/FLOAT(K)

      END


      SUBROUTINE UEBERGABE(N,M,XK,YK,A,HK,IFANZ)
C     *******************************************
C     STORE DISCRETE RESULTS IN MATRIX

      INTEGER N,M,I,IFANZ
      REAL XK,HK
      REAL YK(1:N),A(1:N+2,0:M)
      SAVE I

      A(1,I)=XK
      DO 1,J=1,N,1
        A(J+1,I)=1E6*YK(J)
1     CONTINUE
      WRITE(*,*) I,IFANZ,A(1,I),A(4,I),A(10,I)
      I=I+1

      END



      SUBROUTINE DGL(X,Y,N,F)
C     ************************
C     DETERMINE FIRST ORDER STATE SPACE DIFFERENTIAL EQUATION SYSTEM.

      INTEGER N
      REAL X
      REAL Y(1:N),F(1:N)

      REAL MIK(4,4),MID(4,4)

      REAL C1,C2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,C14,C15,C16,
     1     C18,LM1,LM2,LM3,LM4,DF,K1,K2
      COMMON/CONST/MIK,MID,C1,C2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,
     1               C14,C15,C16,C17,C18,LM1,LM2,LM3,LM4,DF
```

```
      F(1)=Y(5)
      F(2)=Y(6)
      F(3)=Y(7)
      F(4)=Y(8)

      DO 1,I=5,8
1     F(I)=0

      DO 2,I=1,4
      DO 2,J=1,4
2     F(I+4)=F(I+4)-MIK(I,J)*Y(J)-MID(I,J)*Y(J+4)

      END



      SUBROUTINE TE(M,N,A,X,YK)
C     *************************
C     COMPUTE TOTAL ENERGY

      INTEGER M,N
      REAL MA,LL
      REAL A(1:N+3,0:M),YK(1:N),MIK(4,4),MID(4,4)
      REAL C1,C2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,C14,C15,C16,
     1      C18,LM1,LM2,LM3,LM4,DF
      COMMON/CONST/MIK,MID,C1,C2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,
     1              C14,C15,C16,C17,C18,LM1,LM2,LM3,LM4,DF

      SAVE I
      MA=15.931
      LL=60.693


      Y1=YK(1)
      Y2=YK(2)
      Y3=YK(3)
      Y4=YK(4)
      Y5=YK(5)
      Y6=YK(6)
      Y7=YK(7)
      Y8=YK(8)

      SH1=1.
      SH2=1.

      A(10,I)=(C1*Y5**2+(C2+C3*SH1+C4*SH2)*Y6**2+C5*Y7**2+C6*Y8**2
     1         +(C7+C8*SH1+C9*SH2)*Y5*Y6+(C10*SH1+C11*SH2)*Y5*Y7
     2         +C12*SH2*Y5*Y8+(C13+C14*SH1+C15*SH2)*Y6*Y7
     3         +(C16+C17*SH2)*Y6*Y8+C18*Y7*Y8
     4         +0.5*(LM1*Y1**2+LM2*(Y2-Y1)**2+LM3*Y3**2
     5         +LM4*Y4**2))*MA*LL**3
```

```
      I=I+1

      END


      SUBROUTINE TORQUE(M,N,A,YK,CONST)
C     ********************************
C     COMPUTE CONTROL TORQUE

      INTEGER M,N
      REAL MA,LL
      REAL A(1:N+3,0:M),YK(1:N),CONST(8)
      SAVE I
      MA=15.931
      LL=60.693

C     ACTIVE DAMPING:
C     ---------------
      DO 1,J=1,8
1     A(11,I)=A(11,I)+CONST(J)*YK(J)
      A(11,I)=A(11,I)*MA*LL**3


C     PASSIVE DAMPING:
C     ----------------
C      DF=2E3
C      A(11,I)=DF*YK(7)


C     INDEX:
C     ------
      I=I+1

      END


      SUBROUTINE SPEICHERN(K,M,N,A,NAME)
C     **********************************
C     STORE RESULTS ON FILE.

      INTEGER K,M,N
      REAL A(1:N+2,0:M)
      CHARACTER ORDNER*23,NAME*7,NAMEXT*8,DATEI*35,TEXT1*7,
     1          TEXT*19,EXT*1


      ORDNER='F:ÜPLOTFITÜPLOTFIT.DATÜ'
      TEXT1='X-TEXT:'
```

```
      DO 20,J=2,N+2,1
        EXT=CHAR(J+63)
        NAMEXT=EXT//NAME
        DATEI=ORDNER//NAMEXT
        TEXT=TEXT1//NAMEXT
        OPEN(UNIT=1,FILE=DATEI,STATUS='UNKNOWN')
        WRITE(1,1004) TEXT
        WRITE(1,1001)
        DO 10,I=0,K,1
          WRITE(1,*) A(1,I),A(J,I)
10      CONTINUE
        WRITE(1,1002)
        CLOSE (UNIT=1)
20    CONTINUE

1001  FORMAT('DATA: X,Y')
1002  FORMAT('END')
1004  FORMAT(A19)

      END


      SUBROUTINE FEHLER(IFEHL)
C     ************************
C     ERROR INDICATOR.

      INTEGER IFEHL

      IF (IFEHL.EQ.0) THEN
        WRITE(*,*)' VERFAHREN ERFOLGREICH'
      ELSE IF (IFEHL.EQ.1) THEN
        WRITE(*,*)' FEHLERSCHRANKEN ZU KLEIN'
      ELSE IF (IFEHL.EQ.2) THEN
        WRITE(*,*)' ANFANGSWERT <= ZIELWERT:  XENDE <= XK'
      ELSE IF (IFEHL.EQ.3) THEN
        WRITE(*,*)' SCHRITTWEITE HK <= 0'
      ELSE IF (IFEHL.EQ.4) THEN
        WRITE(*,*)' ANZAHL DER GLEICHUNGEN GRÖßER ALS 20'
      ELSE IF (IFEHL.EQ.5) THEN
        WRITE(*,*)' ANZAHL DER ZULÄSSIGEN FUNKTIONSAUSWERTUNGEN NICHT'
        WRITE(*,*)' AUSREICHEND FÜR GEFORDERTE GENAUIGKEIT'
      ENDIF

      END
```

PROGRAM EIGEN
****

COMPUTES EIGENVALUES AND EIGENVECTORS OF OPEN LOOP OR CLOSED LOOP SYSTEM. REQUIRES SUBROUTINES FOR EIGENVALUES/EIGENVECTORS AND INVERTING REAL MATRICES.

```
      PROGRAM EIGEN
C     *************

C     DEFINE VARIABLES:
C     -----------------
      PARAMETER (N=4,M=8)
      INTEGER FEHLER
      REAL A(M,M),B(M,M)
      DOUBLE PRECISION WERTR(M,1),WERTI(M,1),EIVEC(M,M)

C     CALL SUBROUTINES:
C     -----------------
      CALL CONSTANTS(A,B)
      CALL EIGENVALUES(A,B,FEHLER,WERTR,WERTI,EIVEC)
      CALL OUTPUT(FEHLER,WERTR,WERTI,EIVEC)

      END



      SUBROUTINE CONSTANTS(A,B)
C     ========================
C     EVALUATE 8 x 8 SYSTEM MATRICES A,B.


C     DEFINE VARIABLES:
C     -----------------
      PARAMETER (N=4,M=8)
      INTEGER IFEHL,Z1
      REAL S1,S2
      INTEGER MX(M),MY(M)
      REAL A(M,M),B(M,M),MM(N,N),K(N,N),CC(N,N)

      REAL LL,L0,L1,L2,L23,LA1,MA,M0,M3,M4,RR,LM1,LM2,LM3,LM4
      REAL C1,C2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,C14,C15,C16,C17
     1     C18
```

```
C     CONSTANTS, CONCERNING THE SPECIFIC PROBLEM:
C     ------------------------------------------
      LL=60.693
      L0=3.15
      L1=46.575
      L2=14.118
      L23=5.22
      LA1=11.432
      MA=15.931
      M0=274.58
      M3=59.453
      M4=362.522
      RR=3.237

      LM1=1.867E6
      LM2=1.150E6
C      LM3=0.071442E6
      LM4=0.071442E6

      DM=2E3


      LM1=LM1/(MA*LL**3)
      LM2=LM2/(MA*LL**3)
      LM3=LM3/(MA*LL**3)
      LM4=LM4/(MA*LL**3)
      DM=DM/(MA*LL**3)


C     EVALUATE CONSTANTS:
C     -------------------
      C1=RR**3/(6*(RR+1)**3)+RR**2/(2*(RR+1)**3)+(M0*RR**2)/
     1  (MA*LL*2*(RR+1)**2)+(M3*L1**2)/(2*MA*LL**3)+(M4*L1**2)/
     2  (2*MA*LL**3)
      C2=RR**3/(6*(RR+1)**3)+(M0/(MA*LL))*(1+(L0*(RR+1)/LL)**2)
     2  +(M3/(2*MA*LL**3))*(L2**2+L23**2)+(M4/(2*MA*LL**3))
     3  *(L2**2+L23**2+LA1**2+2*L23*LA1)
      C3=(M3*L2*L23)/(MA*LL**3)
      C4=(M4*(L2*L23+L2*LA1))/(MA*LL**3)
      C5=(M3*L23**2)/(2*MA*LL**3)+(M4*(L23**2+LA1**2+2*L23*LA1))/
     1  (2*MA*LL**3)
      C6=(M4*LA1**2)/(2*MA*LL**3)
      C7=R/(2*(RR+1)**3)+(M0*R)/(MA*LL*(RR+1)**2)+(M3*L1*L2)/(MA*LL**3)
     1  +(M4*L1*L2)/(MA*LL**3)
      C8=(M3*L1*L23)/(MA*LL**3)
      C9=(M4*(L1*L23+L1*LA1))/(MA*LL**3)
      C10=(M3*L1*L23)/(MA*LL**3)
      C11=(M4*(L1*L23+L1*LA1))/(MA*LL**3)
      C12=(M4*L1*LA1)/(MA*LL**3)
```

```
      C13=(M3*L23)/(MA*LL**3)+(M4*(L23**2+LA1**2+2*L23*LA1))/(MA*LL**3)
      C14=(M3*L2*L23)/(MA*LL**3)
      C15=(M4*(L2*L23+L2*LA1))/(MA*LL**3)
      C16=(M4*(LA1**2+L23*LA1))/(MA*LL**3)
      C17=(M4*L2*LA1)/(MA*LL**3)
      C18=(M4*(LA1**2+L23*LA1))/(MA*LL**3)


C     MASS MATRIX:
C     ------------
      MM(1,1)=2*C1
      MM(1,2)=C7+C8+C9
      MM(1,3)=C10+C11
      MM(1,4)=C12
      MM(2,1)=MM(1,2)
      MM(2,2)=2*(C2+C3+C4)
      MM(2,3)=C13+C14+C15
      MM(2,4)=C16+C17
      MM(3,1)=MM(1,3)
      MM(3,2)=MM(2,3)
      MM(3,3)=2*C5
      MM(3,4)=C18
      MM(4,1)=MM(1,4)
      MM(4,2)=MM(2,4)
      MM(4,3)=MM(3,4)
      MM(4,4)=2*C6


C     STIFFNESS MATRIX:
C     -----------------
      K(1,1)=LM1+LM2
      K(1,2)=-LM2
      K(1,3)=0
      K(1,4)=0
      K(2,1)=K(1,2)
      K(2,2)=LM2
      K(2,3)=0
      K(2,4)=0
      K(3,1)=K(1,3)
      K(3,2)=K(2,3)
      K(3,3)=LM3
      K(3,4)=0
      K(4,1)=K(1,4)
      K(4,2)=K(2,4)
      K(4,3)=K(3,4)
      K(4,4)=LM4
```

```
C     STRUCTURAL DAMPING MATRIX:
C     --------------------------
      CC(1,1)=0
      CC(1,2)=0
      CC(1,3)=0
      CC(1,4)=0
      CC(2,1)=0
      CC(2,2)=0
      CC(2,3)=0
      CC(2,4)=0
      CC(3,1)=0
      CC(3,2)=0
      CC(3,3)=DM
      CC(3,4)=0
      CC(4,1)=0
      CC(4,2)=0
      CC(4,3)=0
      CC(4,4)=0


C     8 * 8 SYSTEM MATRICES A*(dz/dt)=B*z:
C     -----------------------------------
      DO 1,I=1,N
        DO 2,J=1,N
          A(I,J)=MM(I,J)
          A(I+4,J+4)=MM(I,J)
          B(I,J+4)=MM(I,J)
          B(I+4,J)=-K(I,J)
          B(I+4,J+4)=-CC(I,J)
2       CONTINUE
1     CONTINUE


      END


      SUBROUTINE EIGENVALUES (A,B,FEHLER,WERTR,WERTI,EIVEC)
C     =====================================================
      PARAMETER(N=4,M=8)
      INTEGER MX(M),MY(M)
      INTEGER IFEHL,EIGEN,BASIS,LOW,HIGH,FEHLER,CNT(M,1)
      REAL A(M,M),AI(M,M),B(M,M)
      DOUBLE PRECISION AIB(M,M),SKAL(M,1),EIVEC(M,M),
     1                 WERTR(M,1),WERTI(M,1)

      BASIS=16
```

```
C     INVERT MATRIX A:
C     ----------------
      CALL PIVOT(A,M,M,AI,S1,S2,IFEHL,MX,MY,WERT)

      IF (IFEHL.EQ.2) THEN
        WRITE(*,*)' MASSEN-MATRIX SINGULÄR'
        STOP
      ENDIF
      IF (S1.GT.0.001.OR.S2.GT.0.001) THEN
        WRITE(*,*)' INVERTIERUNG ZU UNGENAU'
        WRITE(*,*)S1,S2,SUMME1,SUMME2
        STOP
      ENDIF



C     CHARACTERISTIK MATRIX:
C     ----------------------
      DO 1,I=1,M
      DO 1,J=1,M
1     AIB(I,J)=0

      DO 2,I=1,M
      DO 2,J=1,M
      DO 2,K=1,M
2     AIB(I,J)=AIB(I,J)+DBLE(AI(I,K)*B(K,J))



C     CALL PROGRAM FOR EVALUATING EIGENVALUES OF CHARACT. MATRIX:
C     -----------------------------------------------------------
      FEHLER=EIGEN
     1        (BASIS,M,M,AIB,SKAL,EIVEC,WERTR,WERTI,CNT,LOW,HIGH)


      END


      SUBROUTINE OUTPUT(FEHLER,WERTR,WERTI,EIVEC)
C     ==========================================
      PARAMETER(M=8)
      INTEGER FEHLER
      DOUBLE PRECISION WERTR(M,1),WERTI(M,1),EIVEC(M,M)
      CHARACTER NAME*8,DATEI*34

C      WRITE(*,*)' DATEINAME EINGEBEN: '
C      READ(*,1008) NAME
C      DATEI='C:ÜFORTRANÜUSERÜEIGEN.DATÜ'//NAME
C      OPEN(UNIT=1,FILE=DATEI,STATUS='NEW')
      OPEN(1,FILE='PRN:')
```

```
      WRITE(1,1008) NAME
      IF (FEHLER.EQ.0) THEN
        WRITE(1,1003)
      ELSE IF (FEHLER.EQ.401) THEN
        WRITE(1,1004)
      ELSE IF (FEHLER.EQ.402) THEN
        WRITE(1,1005)
      ELSE IF (FEHLER.EQ.403) THEN
        WRITE(1,1006)
      ENDIF


      WRITE(1,1000)
      DO 11,I=1,M
11    WRITE(1,1007)WERTR(I,1),WERTI(I,1)

      WRITE(1,1002)
      DO 10,K=1,7,2
      WRITE(1,*)
      DO 10,J=1,M
10    WRITE(1,1007)(EIVEC(J,I),I=K,K+1)

      CLOSE(1)

1000  FORMAT(/,' EIGENVALUES REAL AND IMAGINARY PART:')
1002  FORMAT(/,' EIGENVECTORS REAL AND IMAGINARY PART:')
1003  FORMAT(' PROCEDURE SUCCESFUL')
1004  FORMAT(' ORDNUNG N DER MATRIX KLEINER 1')
1005  FORMAT(' EINGABEMATRIX IST NULLMATRIX')
1006  FORMAT(' MAXIMALE SCHRITTZAHL ÜBERSCHRITTEN')
1007  FORMAT(2E14.6)
1008  FORMAT(A)

      END
```

PROGRAM EIGOP
****

COMPUTES CONTROL GAIN VECTOR FOR CLOSED LOOP VELOCITY AND DISPLACEMENT FEEDBACK. REQUIRES SUBROUTINES FOR EIGENVALUES/EIGENVECTORS, INVERTING OF REAL MATRICES AND REAL LINEAR EQUATION SYSTEMS.

```
      PROGRAM EIGOP
C     *************

C     DEFINE VARIABLES:
C     -----------------
      PARAMETER (N=4,M=8)
      REAL A(M,M),AI(M,M),B0(M,M),B(M,M),BA(M,M,M),BP(M,M)
      REAL P(M),DG(M),GOBJ(M),G(M)
      COMPLEX EW(N),EW0(N),PHI(M,N),PSI(M,N),LP(N,M)
      INTEGER Z1,CONTROL,CON
      CHARACTER NAME*8

C     CALL SUBROUTINES:
C     -----------------
      CALL CONSTANTS(A,AI,B0,BA,Z1)
      CALL INTINPUT(P,GOBJ,N1)
      CALL EIGENVALUES0(AI,B0,B,P,EW0)
1     CALL EIGENVALUES(AI,B0,B,P,EW)


      IF (CONTROL(EW,EW0,GOBJ,G,DG,N1,Z1).EQ.1) THEN
        CALL EIGENVECTORS (A,B,EW,PHI,PSI)
        CALL DERIVATIVES(PHI,PSI,BA,LP)
        CALL SENSITIV(EW,LP,BP)
        CALL PAVEC(P,DG,BP)
        GOTO 1
      ENDIF


      WRITE(*,*)' NAME FOR PARAMETER VECTOR ?'
      READ(*,1002) NAME
      OPEN(1,FILE='F:ÜPLOTFITÜAWP.DATÜ'//NAME)
      WRITE(1,*) P
      CLOSE(1)


      OPEN(2,FILE='PRN:')
      WRITE(2,1000)
      WRITE(2,*)G
      WRITE(2,*)
      WRITE(2,1001)
```

```
      WRITE(2,*) P
      CLOSE(2)

1000  FORMAT(/,' CALCULATED OBJECTVETOR:')
1001  FORMAT(/,' PARAMETER VECTOR:')
1002  FORMAT(A)


      END



      SUBROUTINE EIGENVALUES0 (AI,B0,B,P,EW0)
C     ======================================
C     EVALUATE EIGENVALUES OF UNDAMPED SYSTEM.

      PARAMETER(N=4,M=8)
      REAL AI(M,M),B0(M,M),B(M,M),P(M),P0(M)
      COMPLEX EW0(N)


C     PARAMETER VECTOR:

      P0(1)=P(1)
      P0(2)=P(2)
      P0(3)=P(3)
      P0(4)=P(4)
      P0(5)=0
      P0(6)=0
      P0(7)=0
      P0(8)=0

      CALL EIGENVALUES (AI,B0,B,P0,EW0)


      END



      INTEGER FUNCTION CONTROL(EW,EW0,GOBJ,G,DG,N1,Z1)
C     ================================================
      PARAMETER (N=4,M=8)
      INTEGER Z1,Z2
      REAL AG(M),G(M),GOBJ(M),DG(M)
      COMPLEX EW(N),EW0(N)
      SAVE Z2
      N2=10
```

```
C     TERMINATE ITERATION:
C     --------------------
1     IF (Z1.GT.N1) THEN
        CONTROL=0
        RETURN
      ENDIF


C     EVALUATE STEP PARAMETER AND LOCAL OBJECT VECTOR:
C     ------------------------------------------------
      ALPHA=FLOAT(Z1)/FLOAT(N1)
      DO 2,I=5,M
2     AG(I)=ALPHA*GOBJ(I)
      DO 6,K=1,N
6     AG(K)=AIMAG(EW0(K))*(1-AG(K+4)**2)**0.5


C     EVALUATE RESULTANT OBJECT VECTOR:
C     ---------------------------------
      DO 3,K=1,N
      G(K)=AIMAG(EW(K))
      G(K+4)=-REAL(EW(K))/((REAL(EW(K)))**2+(AIMAG(EW(K)))**2)**0.5
3     CONTINUE


C     EVALUATE ABSOLUTE ERROR:
C     ------------------------
      DO 4,K=1,M
4     DG(K)=AG(K)-G(K)


C     EVALUATE RELATIVE ERROR:
C     ------------------------
      SUMZ=0
      SUMN=0
      DO 5,K=1,M
      SUMZ=SUMZ+ABS(DG(K))
      SUMN=SUMN+ABS(AG(K))
5     CONTINUE
      ERR=SUMZ/SUMN

      WRITE(*,*)
      WRITE(*,*)' LOCAL OBJECT VECTOR'
      WRITE(*,*) Z1,AG
      WRITE(*,*)' RESULTANT OBJECTVECTOR:'
      WRITE(*,*) G
```

```
C     DECIDE WHETHER ITERATION SHOULD CONTINUE OR STOP:
C     -------------------------------------------------
      IF (ERR.LT.1E-5) THEN
        Z1=Z1+1
        Z2=0
        GOTO 1
      ELSE
        Z2=Z2+1
        IF (Z2.LT.N2) THEN
          CONTROL=1
          RETURN
        ELSE
          CONTROL=0
          RETURN
        ENDIF
      ENDIF


      END


      SUBROUTINE CONSTANTS(A,AI,B0,BA,Z1)
C     ===================================

C     DEFINE VARIABLES:
C     -----------------
      PARAMETER (N=4,M=8)
      INTEGER IFEHL,Z1
      REAL S1,S2
      INTEGER MX(M),MY(M)
      REAL A(M,M),AI(M,M),B0(M,M),BA(M,M,M),MM(N,N),K(N,N)

      REAL LL,L0,L1,L2,L23,LA1,MA,M0,M3,M4,RR,LM1,LM2,LM4,G0
      REAL C1,C2,C3,C4,C5,C6,C7,C8,C9,C10,C11,C12,C13,C14,C15,C16,C17
     1     C18

C     CONSTANTS, CONCERNING THE SPECIFIC PROBLEM:
C     -------------------------------------------
      LL=60.693
      L0=3.15
      L1=46.575
      L2=14.118
      L23=5.22
      LA1=11.432
      MA=15.931
      M0=274.58
      M3=59.453
      M4=362.522
      RR=3.237
```

```
      LM1=1.867E6
      LM2=1.150E6
      LM4=0.071442E6


      LM1=LM1/(MA*LL**3)
      LM2=LM2/(MA*LL**3)
      LM4=LM4/(MA*LL**3)


C     EVALUATE CONSTANTS:
C     -------------------
      C1=RR**3/(6*(RR+1)**3)+RR**2/(2*(RR+1)**3)+(M0*RR**2)/
     1 (MA*LL*2*(RR+1)**2)+(M3*L1**2)/(2*MA*LL**3)+(M4*L1**2)/
     2 (2*MA*LL**3)
      C2=RR**3/(6*(RR+1)**3)+(M0/(MA*LL))*(1+(L0*(RR+1)/LL)**2)
     2 +(M3/(2*MA*LL**3))*(L2**2+L23**2)+(M4/(2*MA*LL**3))
     3 *(L2**2+L23**2+LA1**2+2*L23*LA1)
      C3=(M3*L2*L23)/(MA*LL**3)
      C4=(M4*(L2*L23+L2*LA1))/(MA*LL**3)
      C5=(M3*L23**2)/(2*MA*LL**3)+(M4*(L23**2+LA1**2+2*L23*LA1))/
     1 (2*MA*LL**3)
      C6=(M4*LA1**2)/(2*MA*LL**3)
      C7=R/(2*(RR+1)**3)+(M0*R)/(MA*LL*(RR+1)**2)+(M3*L1*L2)/(MA*LL**3)
     1 +(M4*L1*L2)/(MA*LL**3)
      C8=(M3*L1*L23)/(MA*LL**3)
      C9=(M4*(L1*L23+L1*LA1))/(MA*LL**3)
      C10=(M3*L1*L23)/(MA*LL**3)
      C11=(M4*(L1*L23+L1*LA1))/(MA*LL**3)
      C12=(M4*L1*LA1)/(MA*LL**3)
      C13=(M3*L23)/(MA*LL**3)+(M4*(L23**2+LA1**2+2*L23*LA1))/(MA*LL**3)
      C14=(M3*L2*L23)/(MA*LL**3)
      C15=(M4*(L2*L23+L2*LA1))/(MA*LL**3)
      C16=(M4*(LA1**2+L23*LA1))/(MA*LL**3)
      C17=(M4*L2*LA1)/(MA*LL**3)
      C18=(M4*(LA1**2+L23*LA1))/(MA*LL**3)


C     MASS MATRIX:
C     ------------
      MM(1,1)=2*C1
      MM(1,2)=C7+C8+C9
      MM(1,3)=C10+C11
      MM(1,4)=C12
      MM(2,1)=MM(1,2)
      MM(2,2)=2*(C2+C3+C4)
      MM(2,3)=C13+C14+C15
      MM(2,4)=C16+C17
      MM(3,1)=MM(1,3)
```

```
      MM(3,2)=MM(2,3)
      MM(3,3)=2*C5
      MM(3,4)=C18
      MM(4,1)=MM(1,4)
      MM(4,2)=MM(2,4)
      MM(4,3)=MM(3,4)
      MM(4,4)=2*C6


C     STIFFNESS MATRIX:
C     -----------------
      K(1,1)=LM1+LM2
      K(1,2)=-LM2
      K(1,3)=0
      K(1,4)=0
      K(2,1)=K(1,2)
      K(2,2)=LM2
      K(2,3)=0
      K(2,4)=0
      K(3,1)=K(1,3)
      K(3,2)=K(2,3)
      K(3,3)=0
      K(3,4)=0
      K(4,1)=K(1,4)
      K(4,2)=K(2,4)
      K(4,3)=K(3,4)
      K(4,4)=LM4


C     8 * 8 SYSTEM MATRICES:
C     ----------------------
      DO 1,I=1,N
        DO 2,J=1,N
          A(I,J)=MM(I,J)
          A(I+4,J+4)=MM(I,J)
          B0(I,J+4)=MM(I,J)
          B0(I+4,J)=-K(I,J)
2       CONTINUE
1     CONTINUE


C     INVERT MATRIX A:
C     ----------------
      CALL PIVOT(A,M,M,AI,S1,S2,IFEHL,MX,MY,WERT)

      IF (IFEHL.EQ.2) THEN
        WRITE(*,*)' MASSEN-MATRIX SINGULÄR'
        STOP
      ENDIF
```

```
      IF (S1.GT.0.001.OR.S2.GT.0.001) THEN
        WRITE(*,*)' INVERTIERUNG ZU UNGENAU'
        WRITE(*,*)S1,S2,SUMME1,SUMME2
        STOP
      ENDIF



C     PARTIELL DERIVATIVES OF MATRIX B:
C     ================================
      DO 3,I=1,M
      BA(I,6,I)=1
      BA(I,7,I)=-1
3     CONTINUE



C     COUNTER FOR FUNCTION CONTROL:
C     ----------------------------
      Z1=1

      END

      SUBROUTINE INTINPUT(P,GOBJ,N1)
C     =============================

C     VARIABLE DEFINIEREN:
C     --------------------
      PARAMETER (N=4,M=8)
      REAL P(M),GOBJ(M)


C     INTERACTIVE INPUT:
C     ------------------
C      WRITE(*,1002)
C      READ(*,*) P
      P(1)=0
      P(2)=0
      P(3)=.002
      P(4)=0
      P(5)=0
      P(6)=.003
      P(7)=.003
      P(8)=0

      WRITE(*,1000)
      READ(*,*) GOBJ(5),GOBJ(6),GOBJ(7),GOBJ(8)
      WRITE(*,1001)
      READ(*,*) N1
```

```
1002  FORMAT(' PARAMETER VEKTOR (8 VALUES) ?',/)
1000  FORMAT
     1(' OBJECTVECTOR BETA1,BETA2,BETA3,BETA4 ?',/)
1001  FORMAT(' NUMBER OF INTERMEDIATE STEPS ? : ')

      END


      SUBROUTINE EIGENVALUES (AI,B0,B,P,EW)
C     ====================================
      PARAMETER(N=4,M=8)
      INTEGER EIGEN,BASIS,LOW,HIGH,FEHLER,CNT(M,1)
      REAL AI(M,M),B0(M,M),B(M,M),P(M),MAX
      DOUBLE PRECISION AIB(M,M),SKAL(M,1),EIVEC(M,M),
     1                 WERTR(M,1),WERTI(M,1)
      COMPLEX EW(N)

      BASIS=16



C     ADD CURRENT PARAMETER VECTOR TO SYSTEM MATRIX B:
C     ------------------------------------------------
      DO 8,I=1,M
      DO 8,J=1,M
8     B(I,J)=B0(I,J)

      DO 5,I=1,M
      B(6,I)=B0(6,I)+P(I)
      B(7,I)=B0(7,I)-P(I)
5     CONTINUE



C     CHARACTERISTIK MATRIX:
C     ----------------------
      DO 1,I=1,M
      DO 1,J=1,M
1     AIB(I,J)=0

      DO 2,I=1,M
      DO 2,J=1,M
      DO 2,K=1,M
2     AIB(I,J)=AIB(I,J)+DBLE(AI(I,K)*B(K,J))


C     CALL PROGRAM FOR EVALUATING EIGENVALUES OF CHARACT. MATRIX:
C     -----------------------------------------------------------
```

```
      FEHLER=EIGEN
     1       (BASIS,M,M,AIB,SKAL,EIVEC,WERTR,WERTI,CNT,LOW,HIGH)

      IF (FEHLER.NE.0) THEN
      WRITE(*,*)' FEHLER BEI BESTIMMUNG DER EIGENWERTE'
      WRITE(*,*) FEHLER
      ENDIF


C     ORDER COMPLEX EIGENVALUES:
C     --------------------------
      DO 7,K=N,1,-1
      MAX=WERTI(1,1)
      IMAX=1
      DO 6,I=2,M
      IF (WERTI(I,1).GT.MAX) THEN
      MAX=WERTI(I,1)
      IMAX=I
      ENDIF
6     CONTINUE
      EW(K)=CMPLX(WERTR(IMAX,1),WERTI(IMAX,1))
      WERTI(IMAX,1)=0
7     CONTINUE

      IF (AIMAG(EW(1)).LE.0) THEN
      WRITE(*,*)' EIGENVALUE = 0'
      STOP
      ELSE
      WRITE(*,*)' EIGENVALUES O/K'
      ENDIF


      END



      SUBROUTINE EIGENVECTORS(A,B,EW,PHI,PSI)
C     =======================================
      PARAMETER (N=4,M=8)
      REAL A(M,M),B(M,M)
      COMPLEX EW(N),C(M,M),EV(M),PHI(M,N),PSI(M,N)


C     RIGHT EIGENVECTORS:
C     --------------------
      DO 1,K=1,N
```

```
C     EIGENVALUE MATRIX:
C     ------------------
      DO 2,I=1,M
      DO 2,J=1,M
      C(I,J)=EW(K)*CMPLX(A(I,J))
      C(I,J)=C(I,J)-CMPLX(B(I,J))
2     CONTINUE

C     CALL GAUSS ELIMINATION:
C     -----------------------
      CALL GAUSS(C,EV)

C     EIGENVECTOR MATRIX:
C     -------------------
      DO 3,I=1,M
3     PHI(I,K)=EV(I)

1     CONTINUE



C     LEFT EIGENVECTORS:
C     ------------------
      DO 4,K=1,N

C     EIGENVALUE MATRIX:
C     ------------------
      DO 5,I=1,M
      DO 5,J=1,M
      C(I,J)=EW(K)*CMPLX(A(J,I))
      C(I,J)=C(I,J)-CMPLX(B(J,I))
5     CONTINUE

C     CALL GAUSS ELIMINATION:
C     -----------------------
      CALL GAUSS(C,EV)

C     EIGENVECTOR MATRIX:
C     -------------------
      DO 6,I=1,M
6     PSI(I,K)=EV(I)

4     CONTINUE


C     NORMALIZE EIGENVECTORS:
C     -----------------------
      CALL EIGNORM(A,B,PHI,PSI)
```

```
C       VERIFY EIGENVECTORS:
C       --------------------
        CALL EIGVEKCHECK(A,B,EW,PHI,PSI)

        END


        SUBROUTINE EIGNORM(A,B,PHI,PSI)
C       ==================================
C       NORMALIZE EIGENVECTOR PHI

        PARAMETER (N=4,M=8)
        REAL A(M,M),B(M,M)
        COMPLEX PHI(M,N),PSI(M,N),PSIA(M,N),PSIAPHI(N,N),C(N)


C       MATRIX MULTIPLICATION:
C       ----------------------
        DO 2,J=1,M
        DO 2,K=1,N
2       PSIA(J,K)=(0,0)

        DO 1,K=1,N
        DO 1,J=1,M
        DO 1,I=1,M
1       PSIA(J,K)=PSIA(J,K)+PSI(I,K)*CMPLX(A(I,J))

        DO 4,K=1,N
        DO 4,L=1,N
4       PSIAPHI(K,L)=(0,0)


        DO 3,K=1,N
        DO 3,L=1,N
        DO 3,I=1,M
3       PSIAPHI(K,L)=PSIAPHI(K,L)+PSIA(I,K)*PHI(I,L)


C       EVALUATE CONSTANTS FOR EIGENVECTOR NORMALIZATION:
C       -------------------------------------------------
        DO 9,I=1,N
9       C(I)=(1,0)/PSIAPHI(I,I)

C       NORMALIZE PHI EIGENVECTORS:
C       ---------------------------
        DO 10,J=1,N
        DO 10,I=1,M
10      PHI(I,J)=PHI(I,J)*C(J)

        END
```

```
      SUBROUTINE EIGVEKCHECK(A,B,EW,PHI,PSI)
C     ======================================
C     VERIFY NORMALIZED LEFT AND RIGHT EIGENVECTORS
C     EIGENVECTORS MUST BE ORTHONORMAL

      PARAMETER (N=4,M=8)
      REAL A(M,M),B(M,M)
      COMPLEX EW(N),PHI(M,N),PSI(M,N),PSIA(M,N),PSIB(M,N)
      COMPLEX PSIAPHI(N,N),PSIBPHI(N,N),C(N)

C     MATRIX MULTIPLICATIONS:

C     MATRIX A:
C     ---------
      DO 2,J=1,M
      DO 2,K=1,N
2     PSIA(J,K)=(0,0)

      DO 1,K=1,N
      DO 1,J=1,M
      DO 1,I=1,M
1     PSIA(J,K)=PSIA(J,K)+PSI(I,K)*CMPLX(A(I,J))

      DO 4,K=1,N
      DO 4,L=1,N
4     PSIAPHI(K,L)=(0,0)


      DO 3,K=1,N
      DO 3,L=1,N
      DO 3,I=1,M
3     PSIAPHI(K,L)=PSIAPHI(K,L)+PSIA(I,K)*PHI(I,L)


C     MATRIX B:
C     ---------
      DO 5,J=1,M
      DO 5,K=1,N
5     PSIB(J,K)=(0,0)

      DO 6,K=1,N
      DO 6,J=1,M
      DO 6,I=1,M
6     PSIB(J,K)=PSIB(J,K)+PSI(I,K)*CMPLX(B(I,J))
      DO 7,K=1,N
      DO 7,L=1,N
7     PSIBPHI(K,L)=(0,0)
```

```
      DO 8,K=1,N
      DO 8,L=1,N
      DO 8,I=1,M
8     PSIBPHI(K,L)=PSIBPHI(K,L)+PSIB(I,K)*PHI(I,L)


C     ALL ELEMENTS EXCEPT DIAGONAL ELEMENTS MUST BE EQUAL ZERO:
C     --------------------------------------------------------
      SUM1=0
      DO 9,I=1,N
      DO 9,J=1,N
      IF (I.NE.J) THEN
      SUM1=SUM1+ABS(PSIAPHI(I,J))
      SUM1=SUM1+ABS(PSIBPHI(I,J))
      ENDIF
9     CONTINUE


C     VERIFY DIAGONAL ELEMENTS:
C     -------------------------
      SUM2=0
      SUM3=0
      DO 10,I=1,N
      SUM2=SUM2+(ABS(PSIAPHI(I,I))-1)
      SUM3=SUM3+ABS(PSIBPHI(I,I)-EW(I))
10    CONTINUE

      WRITE(*,*)' CHECK OF EIGENVECTORS, NUMBERS MUST BE EQUAL ZERO:'
      WRITE(*,*) SUM1,SUM2,SUM3

      END


      SUBROUTINE DERIVATIVES(PHI,PSI,BA,LP)
C     =====================================
      PARAMETER (N=4,M=8)
      REAL BA(M,M,M)
      COMPLEX PHI(M,N),PSI(M,N),LP(N,M),PSIB(M)


C     EVALUATE PARTIAL DERIVATIVES:
C     -----------------------------

      DO 4,I=1,N
      DO 4,J=1,M
4     LP(I,J)=(0,0)

      DO 2,K=1,N
      DO 2,L=1,M
      DO 3,I=1,M
3     PSIB(I)=(0,0)
```

```
      DO 1,J=1,M
      DO 1,I=1,M
1     PSIB(J)=PSIB(J)+PSI(I,K)*CMPLX(BA(L,I,J))
      DO 2,I=1,M
2     LP(K,L)=LP(K,L)+PSIB(I)*PHI(I,K)

      END

      SUBROUTINE SENSITIV(EW,LP,BP)
C     ==============================
C     EVALUATE SENSITY MATRIX

      PARAMETER (N=4,M=8)
      REAL BP(M,M)
      COMPLEX EW(N),LP(N,M)

      DO 2,K=1,N
      DO 2,L=1,M
2     BP(K,L)=AIMAG(LP(K,L))


      DO 1,K=1,N
      DO 1,L=1,M
      BP(K+4,L)=(REAL(EW(K))
     1    *(REAL(EW(K))*REAL(LP(K,L))+AIMAG(EW(K))*AIMAG(LP(K,L)))
     2     -REAL(LP(K,L))*((REAL(EW(K)))**2+(AIMAG(EW(K)))**2))
     3     /((REAL(EW(K)))**2+(AIMAG(EW(K)))**2)**1.5
1     CONTINUE

      END


      SUBROUTINE PAVEC(P,DG,BP)
C     ========================
C     EVALUATE NEW PARAMETER VECTOR

      PARAMETER (N=4,M=8)
      INTEGER IPIVOT(M)
      REAL P(M),DG(M),BP(M,M),DP(M),BPT(M,M),BBP(M,M),LV(M),D(M)


C     TRANSPOND MATRIX BP:
C     --------------------
      DO 4,I=1,M
      DO 4,J=1,M
4     BPT(I,J)=BP(J,I)
```

```
C     EVALUATE COEFFICIENT MATRIX FOR LAGRANGE VECTOR:
C     ---------------------------------------------
      DO 3,K=1,M
      DO 3,L=1,M
3     BBP(K,L)=0

      DO 1,K=1,M
      DO 1,L=1,M
      DO 2,I=1,M
2     BBP(K,L)=BBP(K,L)+BP(K,I)*BPT(I,L)
1     CONTINUE


C     SOLVE REAL, INHOMOGENOUS EQUATION SYSTEM:
C     ---------------------------------------
      CALL FGAUSS(M,BBP,M,DG,LV,MARKE,D,IPIVOT)
      IF (MARKE.EQ.0) THEN
        WRITE(*,*)' SUB PAVEC, COEFFICIENT MATRIX SINGULAR'
        STOP
      ENDIF


C     CHANGE OF PARAMETER VECTOR:
C     ---------------------------
      DO 5,I=1,M
5     DP(I)=0

      DO 7,J=1,M
      DO 7,I=1,M
7     DP(J)=DP(J)+BPT(J,I)*LV(I)


C     NEW PARAMETER VECTOR:
C     --------------------
      DO 6,I=1,M
6     P(I)=P(I)+DP(I)


      END

      SUBROUTINE GAUSS(A,EV)
C     ======================
C     GAUSS ELIMINATION FOR LINEAR, COMPLEX, HOMOGENOUS EQUATION
C     SYSTEM TO EVALUATE EIGENVALUES

      PARAMETER (N=4,M=8)
      INTEGER INDEX(M)
      COMPLEX A(M,M),B(M,M),EV(M)
```

```
      CALL GAUSS_A(A,B,INDEX)
      CALL GAUSS_B(B,INDEX,EV)
      CALL GAUSS_C(A,EV,SUM)

      END


      SUBROUTINE GAUSS_A(A,B,INDEX)
C     ================================
C     FORWARD ELIMINATION, EVALUATE TRIANGULAR COEFFICIENT MATRIX

      PARAMETER (N=4,M=8)
      INTEGER INDEX(M),PIVSPA,IPUF,ZAEHLER
      REAL PIVEL
      COMPLEX A(M,M),B(M,M),PUF,FACT


C     GIVE VARIABLES TO NEW MATRIX B:
C     -------------------------------
      DO 12,I=1,M
      DO 12,J=1,M
12    B(I,J)=A(I,J)


C     INDEX VECTOR:
C     -------------
      DO 5,I=1,M
      INDEX(I)=I
5     CONTINUE
      ZAEHLER=0


C     NORMAILZE ROW VECTOR:
C     ---------------------
      DO 8,I=1,M
        PIVEL=ABS(B(I,1))
        DO 9,J=2,M
          IF (ABS(B(I,J)).GT.PIVEL) THEN
            PIVEL=ABS(B(I,J))
          ENDIF
9       CONTINUE
        IF (PIVEL.GT.1E-6) THEN
        DO 11,J=1,M
11      B(I,J)=B(I,J)/PIVEL
        ENDIF
8     CONTINUE


      DO 2,I=1,M-1
```

```
C     DETERMINE PIVOT COLUMN:
C     -----------------------
7     PIVEL=ABS(B(I,I))
      PIVSPA=I
      DO 3,L=I+1,M
        IF (ABS(B(I,L)).GT.PIVEL) THEN
          PIVEL=ABS(B(I,L))
          PIVSPA=L
        ENDIF
3     CONTINUE


C     SWAP ROWS, IF PIVOT ELEMENT = 0 :
C     ---------------------------------
      IF (PIVEL.LT.1E-6) THEN
        ZAEHLER=ZAEHLER+1
        IF (I+ZAEHLER.GT.M) GOTO 10
        DO 6,L=1,M
          PUF=B(I,L)
          B(I,L)=B(I+ZAEHLER,L)
          B(I+ZAEHLER,L)=PUF
6       CONTINUE
      GOTO 7
      ENDIF
      ZAEHLER=0


C     SWAP COLUMNS, IF ACTUAL COLLUMN IS NOT PIVOT COLUMN:
C     ----------------------------------------------------
      IF (PIVSPA.NE.I) THEN
        DO 4,L=1,M
          PUF=B(L,I)
          B(L,I)=B(L,PIVSPA)
          B(L,PIVSPA)=PUF
4       CONTINUE
        IPUF=INDEX(I)
        INDEX(I)=INDEX(PIVSPA)
        INDEX(PIVSPA)=IPUF
      ENDIF


C     NORMALIZE ROW VECTOR:
C     ---------------------
C      PIVEL=ABS(B(I,I))
C      DO 13,J=I,M
C13   B(I,J)=B(I,J)/PIVEL
```

```
C     GAUSS ELIMINATION:
C     ------------------
      DO 2,K=I+1,M
        FACT=-B(K,I)/B(I,I)
        DO 1,J=I,M
1       B(K,J)=B(K,J)+FACT*B(I,J)
2     CONTINUE

10    CONTINUE

      END



      SUBROUTINE GAUSS_B(B,INDEX,X)
C     =============================
C     BACKWARD ELIMINATION OF COMPLEX, HOMOGENOUS EQUATION SYSTEM

      PARAMETER (N=4,M=8)
      INTEGER INDEX(M)
      COMPLEX B(M,M),X(M),Y(M)


C     STOP, IF COEFFIZIENT MATRIX IS REGULAR:
C     ---------------------------------------
      IF (ABS(B(M,M)).GT.1E-4) THEN
        WRITE(*,*)' KOEFFIZIENTENMATRIX REGULÄR, NUR TRIVIALE LÖSUNG'
        WRITE(*,*) ABS(B(M,M))
C        STOP
      ENDIF


C     BACKWARD ELIMINATION:
C     ---------------------
      DO 3,I=1,M
3     Y(I)=(0,0)

      DO 1,I=M,1,-1
      IF (ABS(B(I,I)).LT.1E-4) THEN
        Y(I)=(1,0)
        GOTO 1
      ENDIF
      DO 2,J=I+1,M
2     Y(I)=Y(I)-B(I,J)*Y(J)
      Y(I)=Y(I)/B(I,I)
1     CONTINUE
```

```
C       COLUMN INDEX:
C       -------------
        DO 4,I=1,M
4       X(INDEX(I))=Y(I)

        END


        SUBROUTINE GAUSS_C(A,X,SUM)
C       ==========================
C       CHECK RESUTS

        PARAMETER (N=4,M=8)
        REAL SUM
        COMPLEX A(M,M),X(M),C(M)


C       VERIFY SOLUTION VECTOR:
C       -----------------------
        DO 1,I=1,M
1       C(I)=(0,0)

        DO 2,I=1,M
        DO 2,J=1,M
2       C(I)=C(I)+A(I,J)*X(J)

        SUM=0
        DO 3,I=1,M
3       SUM=SUM+ABS(C(I))

C        WRITE(*,1000) SUM
C1000   FORMAT(' VERIFICATION OF GAUSS ELIMINATION,
C      1            SUM MUST BE ZERO : ',E10.3)

        END
```

PROGRAM G_FORCES
****

COMPUTES FORCES AND BENDING MOMENTS FOR RIGID SYSTEM, CAUSED BY THE PITCHING MANEUVER.

```
      PROGRAM G.FORCES
C     *****************
C     THIS PROGRAM EVALUATES FORCES AND BENDING MOMENTS FOR
C     A RIGID SYSTEM, CAUSED BY THE PITCHING MANEUVER. RESULTS
C     ARE STORED ON FILE.


C     DECLARE PROGRAM SPECIFIC VARIABLES :
C     ------------------------------------
      INTEGER N,I
      PARAMETER (N=100)
      REAL AX(0:N),AY(0:N)
      CHARACTER NAME*12

C     DECLARE PROBLEM SPECIFIC VARIABLES:
C     ------------------------------------
      REAL A__,TAU1,K,L0,L1,L2,M0,LM1,LM2
      COMMON/CONST/A__,TAU1,K,L0,L1,L2,M0,LM1,LM2
      TAU1=73.135
      L0=3.15
      L1=46.368
      L2=14.325
      L23=5.22
      LA1=11.432
      M3=59.453
      M4=362.522
      M0=30
      LM1=1.867
      LM2=1.15
      A__=M0/(L23**2*M3+(L23+LA1)**2*M4)
      K=L23*M3+(L23+LA1)*M4

C     TIME INTERVAL {XMIN,XMAX}:
C     --------------------------
      XMIN=0
      XMAX=146.263

C     CALL OF SUBROUTINES:
C     --------------------
      WRITE(*,1000)
      READ(*,1001)NAME
      CALL CALC(N,XMIN,XMAX,AX,AY)
      CALL SPEICHERN(N,AX,AY,NAME)
```

```
1000  FORMAT(' DATEINAME EINGEBEN (MAX. 12 ZEICHEN) ')
1001  FORMAT(A)
       END

       REAL FUNCTION FCT(X)
C      ===================
       REAL X
       REAL A,A_,A__,TAU1,K,L0,L1,L2,M0,M1,M2,LM1,LM2
       COMMON /CONST/A__,TAU1,K,L0,L1,L2,M0,LM1,LM2

C      ZENTRIFUGAL AND TRANSVERSAL FORCE:
C      ---------------------------------
       If (X.LE.TAU1) THEN
         A=(A__*X**2)/2
         A_=A__*X
         FZ=A_**2*K
         FR=A__*K
       ELSE
         A=A__*(-TAU1**2+2*TAU1*X-(X**2)/2)
         A_=A__*(2*TAU1-X)
         FZ=A_**2*K
         FR=-A__*K
       ENDIF

C      VERTICAL AND HORIZONTAL FORCE:
C      ------------------------------
       FX=FZ*COS(A)+FR*SIN(A)
       FY=FZ*SIN(A)-FR*COS(A)
C       FCT=FX
C       FCT=FY

C      BENDING MOMENTS AT HINGE 1 & 2 :
C      --------------------------------
       IF (X.LE.TAU1) THEN
         M1=-FX*(L1+L2)+FY*L0-M0
         M2=-FX*L2+FY*L0-M0
       ELSE
         M1=-FX*(L1+L2)+FY*L0+M0
         M2=-FX*L2+FY*L0+M0
       ENDIF
C       FCT=M1
C       FCT=M2

C      DISPLACEMENTS:
C      --------------
C       FCT=M1/LM1
       FCT=M1/LM1+M2/LM2

       RETURN
       END
```

```
      SUBROUTINE CALC(N,XMIN,XMAX,AX,AY)
C     ==================================
      INTEGER N,I
      REAL X,Y,FCT,XMIN,XMAX,AX(0:N),AY(0:N)
      DO 10,I=0,N,1
        X=XMIN+((XMAX-XMIN)/FLOAT(N))*I
        Y=FCT(X)
        WRITE(*,*) X,Y
        AX(I)=X
        AY(I)=Y
10    CONTINUE
      RETURN
      END


      SUBROUTINE SPEICHERN(N,AX,AY,NAME)
C     ==================================
      INTEGER N,I
      REAL AX(0:N),AY(0:N)
      CHARACTER ORDNER*23,NAME*12,DATEI*35,TEXT1*7,TEXT*19


      ORDNER='F:ÜPLOTFITÜPLOTFIT.DATÜ'
      TEXT1='X-TEXT:'
      DATEI=ORDNER//NAME
      TEXT=TEXT1//NAME

      OPEN(UNIT=1,FILE=DATEI,STATUS='UNKNOWN')
      WRITE(1,1004) TEXT
      WRITE(1,1001)
      DO 10,I=0,N,1
        WRITE(1,*) AX(I),AY(I)
10    CONTINUE
      WRITE(1,1002)
      CLOSE (UNIT=1)

1001  FORMAT('DATA: X,Y')
1002  FORMAT('END')
1004  FORMAT(A19)

      RETURN
      END
```

SUBROUTINES FOR COMPUTING THE INVERSE OF A REAL MATRIX
****
TAKEN FROM THE MATHEMATICAL SOFTWARE LIBRARY UNIVERSITY AACHEN.

```
      SUBROUTINE PIVOT (A,IA,N,B,S1,S2,IFEHL,MX,MY,WERT)
C     ==================================================

C*****************************************************************
C
C   THIS SUBROUTINE EVALUATES THE INVERSE OF A REAL QUADRATIC
C   N * N MATRIX.
C
C  INPUT PARAMETER:
C  ================
C  A:       2-DIM(IA,N) ARRAY, CONTENTS THE MATRIX WHICH SHOULD
C           BE INVERTED.
C  IA:      LEADING DIMENSION OF A, AS IN CALLING PROGRAM
C           DECLARED.
C  N:       ORDER OF MATRIX A.
C
C  OUTPUT PARAMETER:
C  =================
C  B:       2-DIM(IA,N) ARRAY, CONTENTS THE INVERTED MATRIX OF A
C  S1,S2:  VERIFICATION OF RESULT. S1 = SUM  OF THE AMOUNTS
C           OF THE DIAGONAL ELEMENTS OF THE MATRIX A*B-E
C           (E= UNITY MATRIX), S2= SUM OF THE AMOUNTS OF ALL
C           OTHER ELEMENTS. A*B-E IS NAUGHT MATRIX,THEORETICALLY
C  IFEHL   ERROR PARAMETER
C           IFEHL=1, INVERSE MATRIX OF A FOUND
C           IFEHL=2, MATRIX A IS SINGULAR
C
C
C  OTHERS :
C  ========
C  MX,MY : (N) 1-DIM INTEGER ARRAY
C
C
C  QUELLEN : 1. AUSTAUSCHVERFAHREN IN ABSCHNITT 3.6
C             2. FORTRAN IV - PROGRAMM AUSTAU VON TH. TOLXDORFF
C                IN DER 4. AUFLAGE DIESER FORMELSAMMLUNG.
C
C*****************************************************************
C
C  AUTOR: GISELA ENGELN-MUELLGES, FORTRAN 77, 01.02.1987
C
C*****************************************************************
C
```

```
C  DEKLARATIONEN.
C
      REAL A(IA,N),B(IA,N)
      INTEGER MX(N),MY(N)
C
C  BERECHNUNG DES MASCHINENRUNDUNGSFEHLERS FMACHN.
C
      FMACHN=1.
    5 FMACHN=0.5*FMACHN
      IF(1..LT.1.+FMACHN) GOTO 5
      FMACHN=2.*FMACHN
C
C  UMSPEICHERN DER MATRIX A AUF B.
C
      DO 10 I=1,N
        DO 10 J=1,N
          B(I,J)=A(I,J)
   10 CONTINUE
C
C  VORBESETZEN DER PIVOTVEKTORN MX UND MY MIT NULL.
C
      DO 20 I=1,N
        MX(I)=0
        MY(I)=0
   20 CONTINUE
C
C  BESTIMMUNG DES PIVOTELEMENTES.
C
      DO 30 I=1,N
        PIVO=0.
        DO 40 IX=1,N
          IF(MX(IX).EQ.0) THEN
            DO 50 IY=1,N
              IF(MY(IY).EQ.0) THEN
                IF(ABS(B(IX,IY)).GT.ABS(PIVO)) THEN
                  PIVO=B(IX,IY)
                  NX=IX
                  NY=IY
                ENDIF
              ENDIF
   50       CONTINUE
          ENDIF
   40   CONTINUE
C
C  FALLS DAS PIVOTELEMENT NULL IST, IST DIE MATRIX SINGULAER.
C
        IF(ABS(PIVO).LT.4.*FMACHN) THEN
          WERT=PIVO
          IFEHL=2
          RETURN
        ENDIF
```

```
C
C  MERKEN DER INDIZES DES PIVOTELEMENTES.
C
        MX(NX)=NY
        MY(NY)=NX
C
C  BERECHNUNG DER MATRIXELEMENTE GEMAESS DER RECHENREGELN FUER
C  EINEN AUSTAUSCHSCHRITT.
C
        HILF=1./PIVO
        DO 60 J=1,N
          IF(J.NE.NX) THEN
            FAKTOR=B(J,NY)*HILF
            DO 70 K=1,N
              B(J,K)=B(J,K)-B(NX,K)*FAKTOR
              B(J,NY)=-FAKTOR
   70       CONTINUE
          ENDIF
   60   CONTINUE
        DO 80 K=1,N
          B(NX,K)=B(NX,K)*HILF
          B(NX,NY)=HILF
   80   CONTINUE
   30 CONTINUE
C
C  ZEILEN- UND SPALTENVERTAUSCHUNGEN RUECKGAENGIG MACHEN.
C
      DO 90 I=1,N-1
        DO 100 M=I,N
          IF(MX(M).EQ.I) GOTO 110
  100   CONTINUE
  110   J=M
        IF(J.NE.I) THEN
          DO 120 K=1,N
            H=B(I,K)
            B(I,K)=B(J,K)
            B(J,K)=H
  120     CONTINUE
          MX(J)=MX(I)
          MX(I)=I
        ENDIF
        DO 130 M=I,N
          IF(MY(M).EQ.I) GOTO 140
  130   CONTINUE
  140   J=M
        IF(J.NE.I) THEN
          DO 150 K=1,N
            H=B(K,I)
            B(K,I)=B(K,J)
            B(K,J)=H
  150     CONTINUE
```

```
          MY(J)=MY(I)
          MY(I)=I
        ENDIF
   90 CONTINUE
C
C  BILDUNG DER DIFFERENZMATRIX S=(A*B-E), E=EINHEITSMATRIX.
C  BILDUNG DER SUMME S1 DER BETRAEGE DER DIAGONALELEMENTE VON S
C  UND DER SUMME S2 DER BETRAEGE ALLER UEBRIGEN GLIEDER.
C  THEORETISCH MUESSTEN S1 UND S2 NULL SEIN.
C
      S1=0.
      S2=0.
      DO 160 I=1,N
        DO 160 J=1,N
          H=0.
          DO 170 K=1,N
            H=H+A(I,K)*B(K,J)
  170     CONTINUE
          IF(I.EQ.J) THEN
            S1=S1+ABS(H-1.)
          ELSE
            S2=S2+ABS(H)
          ENDIF
  160 CONTINUE
      IFEHL=1
      RETURN
      END
```

SUBROUTINES FOR COMPUTING EIGENVALIES AND EIGENVECTORS OF REAL MATRICES
****
TAKEN FROM THE MATHEMATICAL SOFTWARE LIBRARY UNIVERSITY AACHEN.

```
      INTEGER FUNCTION EIGEN
     *    (BASIS,ND,N,MAT,SKAL,EIVEC,WERTR,WERTI,CNT,LOW,HIGH)
C
C*****************************************************************
C
C     BRIEF DESCRIPTION OF PROGRAM
C     ============================
C     SUBROUTINE TAKEN FROM MATHEMATICAL SODTWARE LIBRARY
C     UNIVERSITY OF AACHEN/GERMANY.
C
C     THIS FUNCTION SUBROUTINE TYP INTEGER EVALUATES
C     EIGENVALUES AND EIGENVECTORS OF AN ARBITRARY REAL
C     MATRIX MAT.
C     THE EIGENVALUES ARE STORED IN THE ARRAYS WERTR(1:N)
C     (REAL PART) AND WERTI(1:N) (IMAGINARY PART). THE EIGEN-
C     VECTORS ARE STORED IN THE ARRAY EIVEC(1:N,1:N).
C
C     INPUT PARAMETERS:
C     =================
C     N:             ORDER OF MATRIX MAT
C     ND:            LEADING DIMENSION OF ARRAYS MAT AND
C                    EIVEC, AS IN CALLING PROGRAM DECALRED.
C     MAT:           (N,N)-ARRAY, TYPE DOUBLE PRECISION, CONTAINS
C                    THE MATRIX, WHOSE EIGENVALUES AND EIGENV-
C                    VECTORS SHOULD BE CACULATED.
C     BASIS:         BASIS OF FLOATING POINT OF THE USED
C                    MACHINE. (MOSTLY 2 OR 16)
C
C     OUTPUT PARAMETERS:
C     ==================
C     MAT:           UPPER PART OF (N,N) ARRAY CONTAINS THE
C                    EIGENVECTORS OF THE QUASI TRIANGULAR MATRIX
C                    WHICH IS EVALUATED FROM QR-PROCEDURE.
C     WERTR,WERTI: TWO (N,1) ARRAYS, TYPE DOUBLE PRECISION,
C                    REAL AND IMAGINARY PART OF EIGENVALUES.
C     EIVEC:         (N,N) ARRAY, TYPE DOUBLE PRECISION,
C                    NORMAIZED EIGENVECTORS OF INITIAL MATRIX.
C                    IF THE EIGENVALUE I IS REAL, THEN IS
C                    THE COLUMN I THE CORRESPONDING EIGEN-
C                    VECTOR. IF THE EIGENVALUES I AND I+1 ARE
C                    COMPLEX AND COMPLEX CONJUGATE, THEN CONTAINS
C                    THE COLUMN I THE REAL PART AND THE COLUMN
C                    I+1 THE IMAGINARY PART OF THE EIGENVECTOR,
C                    WHICH  BELONGS TO THE EIGENVALUE WITH
C                    THE POSITIVE IMAGINARY PART.
```

```
C     CNT:          (N,1) ARRAY, TYPE INTEGER, CONTAINS
C                    THE NUMBER OF ITERATION STEPS FOR EACH
C                    EIGENVALUE.
C     SKAL:         (N,1) ARRAY, TYPE DOUBLE PRECISION,
C                    CONTAINS SCALE FACTORS.
C
C     LOCAL VARIABLES:
C     ================
C     ONE,TWO,HALF: FLOATING POINT CONSTANTS.
C     EPS:           ACCURACY OF USED MACHINE.
C
C     NEEDED SUBROUTINES:
C     ===================
C     BALAN:  BALANCING OF GIVEN MATRIX IN 1-NORM
C     ELMHES: HESSENBERG TRANSFORMATION
C     ELMTRA: EVALUATES TRANSFORMED MATRIX.
C     HQR2:   EVALUATES EIGENVALUES AND EIGENVECTORS OF UPPER
C              HESSENBERG MATRIX.
C     BALBAK: TRANSFORMS EIGENVECTORS OF THE BALANCED MATRIX
C              TO THE EIGENVECTORS OF THE ORIGINAL MATRIX.
C     NORMAL: NORMALZES EIENVECTORS, MAXIMUM NORM.
C     SWAP:   SWAPS TWO VARIABLES.
C     COMDIV: DIVISION OF COMPLEX VARIABLES.
C     COMABS: MAGNITUDE OF A COMPLEX VARIABLE.
C
C     BACK VALUES:
C     ============
C     0:       NO ERROR.
C     401:     ORDER OF MATRIX MAT IS LESS THAN 1
C     402:     MAT IS NAUGHT MATRIX
C     403:     MAXIMUM NUMBER OF ITERATION STEPS IS EXCEEDED
C               WHITHOUT GAINING THE SOLUTION.
C
C*****************************************************************
C
C     AUTOR:               JUERGEN DIETEL, RWTH AACHEN
C     DATUM:               FR 10. 4. 1987
C     QUELLEN:
C       A) MARTIN, R. S. UND WILKINSON, J. H.: SIMILARITY
C          REDUCTION OF GENERAL MATRICES TO HESSENBERG FORM.
C          NUM. MATH. 12, 349-368 (1969)
C       B) PARLETT, B. N. UND REINSCH, C.: BALANCING A MATRIX
C          FOR CALCULATION OF EIGENVALUES AND EIGENVECTORS.
C          NUM. MATH. 13, 293-304 (1969)
C       C) PETERS, G. UND WILKINSON, J. H.: EIGENVECTORS OF
C          REAL AND COMPLEX MATRICES BY LR AND QR
C          TRIANGULARIZATIONS. NUM. MATH. 16, 181-204 (1970)
C     PROGRAMMIERSPRACHE: FORTRAN 77
C
C*****************************************************************
```

```
C
      INTEGER BASIS,ND,N,CNT(N),LOW,HIGH
      DOUBLE PRECISION MAT(ND,N),SKAL(N),EIVEC(ND,N),WERTR(N),
     *                 WERTI(N)
      DOUBLE PRECISION ONE,TWO,HALF
      PARAMETER (ONE = 1.0D0,TWO = 2.0D0,HALF = 0.5D0)
      INTEGER RES,BALAN,ELMHES,ELMTRA,HQR2,BALBAK,NORMAL
      DOUBLE PRECISION EPS,TEMP
C
C     BERECHNUNG DER MASCHINENGENAUIGKEIT EPS (D. H. DER KLEINSTEN
C     POSITIVEN MASCHINENZAHL, FUER DIE AUF DEM RECHNER GILT:
C     1 + EPS > 1):
C
      TEMP = TWO
      EPS = ONE
   10 IF (ONE .LT. TEMP) THEN
         EPS = EPS * HALF
         TEMP = ONE + EPS
         GOTO 10
         ENDIF
      EPS = TWO * EPS
      RES = BALAN(ND,N,MAT,SKAL,LOW,HIGH,BASIS)
         IF (RES .NE. 0) THEN
            EIGEN = RES + 100
            RETURN
         ENDIF
      RES = ELMHES(ND,N,LOW,HIGH,MAT,CNT)
         IF (RES .NE. 0) THEN
            EIGEN = RES + 200
            RETURN
         ENDIF
      RES = ELMTRA(ND,N,LOW,HIGH,MAT,CNT,EIVEC)
         IF (RES .NE. 0) THEN
            EIGEN = RES + 300
            RETURN
         ENDIF
      RES = HQR2(ND,N,LOW,HIGH,MAT,WERTR,WERTI,EIVEC,CNT,EPS)
         IF (RES .NE. 0) THEN
            EIGEN = RES + 400
            RETURN
         ENDIF
      RES = BALBAK(ND,N,LOW,HIGH,SKAL,EIVEC)
         IF (RES .NE. 0) THEN
            EIGEN = RES + 500
            RETURN
         ENDIF
      RES = NORMAL(ND,N,EIVEC,WERTI)
         IF (RES .NE. 0) THEN
            EIGEN = RES + 600
            RETURN
```

```
         ENDIF
      EIGEN = 0
      RETURN
      END


      INTEGER FUNCTION BALAN (ND,N,MAT,SKAL,LOW,HIGH,BASIS)
C     ==================================================
C     SUBROUTINE FOR FUNCTION EIGEN

      INTEGER ND,N,LOW,HIGH,BASIS
      DOUBLE PRECISION SKAL(N),MAT(ND,N)
      DOUBLE PRECISION ZERO,ONE,PT95
      PARAMETER (ZERO = 0.0D0,ONE = 1.0D0,PT95 = 0.95D0)
      INTEGER I,J,K,L,B2
      DOUBLE PRECISION R,C,F,G,S
C
C     DIE NORM VON MAT(1:N,1:N) REDUZIEREN DURCH EXAKTE AEHNLICH-
C     KEITSTRANSFORMATIONEN, DIE IN SKAL(1:N) ABGESPEICHERT WERDEN
C
      B2 = BASIS*BASIS
      L = 1
      K = N
C
C     NACH ZEILEN MIT EINEM ISOLIERTEN EIGENWERT SUCHEN UND SIE
C     NACH UNTEN SCHIEBEN
C
   10 DO 50 J=K,1,-1
         R = ZERO
         DO 20 I=1,K
   20       IF (I .NE. J) R = R+ABS(MAT(J,I))
         IF (R .EQ. ZERO) THEN
            SKAL(K) = J
            IF (J .NE. K) THEN
               DO 30 I=1,K
   30             CALL SWAP(MAT(I,J),MAT(I,K))
               DO 40 I=L,N
   40             CALL SWAP(MAT(J,I),MAT(K,I))
            ENDIF
            K = K-1
            GOTO 10
         ENDIF
   50    CONTINUE
C
C     NACH SPALTEN MIT EINEM ISOLIERTEN EIGENWERT SUCHEN UND SIE
C     NACH LINKS SCHIEBEN
C
   60 DO 100 J=L,K
         C = ZERO
         DO 70 I=L,K
   70       IF (I .NE. J) C = C+ABS(MAT(I,J))
```

```
        IF (C .EQ. ZERO) THEN
           SKAL(L) = J
           IF (J .NE. L) THEN
              DO 80 I=1,K
 80              CALL SWAP(MAT(I,J),MAT(I,L))
              DO 90 I=L,N
 90              CALL SWAP(MAT(J,I),MAT(L,I))
           ENDIF
           L = L+1
           GOTO 60
        ENDIF
 100    CONTINUE
C
C     NUN DIE TEILMATRIX IN DEN ZEILEN L BIS K AUSBALANCIEREN
C
      LOW = L
      HIGH = K
      DO 110 I=L,K
 110     SKAL(I) = ONE
 120  DO 180 I=L,K
         C = ZERO
         R = ZERO
         DO 130 J=L,K
            IF (J .NE. I) THEN
               C = C+ABS(MAT(J,I))
               R = R+ABS(MAT(I,J))
            ENDIF
 130        CONTINUE
         G = R/BASIS
         F = ONE
         S = C+R
 140     IF (C .LT. G) THEN
            F = F*BASIS
            C = C*B2
            GOTO 140
         ENDIF
         G = R*BASIS
 150     IF (C .GE. G) THEN
            F = F/BASIS
            C = C/B2
            GOTO 150
         ENDIF
         IF ((C+R)/F .LT. PT95*S) THEN
            G = ONE/F
            SKAL(I) = SKAL(I)*F
            DO 160 J=L,N
 160           MAT(I,J) = MAT(I,J)*G
            DO 170 J=1,K
 170           MAT(J,I) = MAT(J,I)*F
            GOTO 120
```

```
         ENDIF
  180    CONTINUE
      BALAN = 0
      END

      INTEGER FUNCTION BALBAK(ND,N,LOW,HIGH,SKAL,EIVEC)
C     ================================================
C     SUBROUTINE FOR FUNCTION EIGEN

      INTEGER ND,N,LOW,HIGH
      DOUBLE PRECISION SKAL(N),EIVEC(ND,N)
      INTEGER I,J,K
      DOUBLE PRECISION S
C
      DO 20 I=LOW,HIGH
         S = SKAL(I)
C
C        LINKSEIGENVEKTOREN WERDEN ZURUECKTRANSFORMIERT, INDEM MAN
C        DIE VORIGE ANWEISUNG ERSETZT DURCH: 'S = 1.0D0/SKAL(I)'
C
         DO 10 J=1,N
   10       EIVEC(I,J) = EIVEC(I,J)*S
   20    CONTINUE
      DO 40 I=LOW-1,1,-1
         K=SKAL(I)
         DO 30 J=1,N
   30       CALL SWAP(EIVEC(I,J),EIVEC(K,J))
   40    CONTINUE
      DO 60 I=HIGH+1,N
         K=SKAL(I)
         DO 50 J=1,N
   50       CALL SWAP(EIVEC(I,J),EIVEC(K,J))
   60    CONTINUE
      BALBAK = 0
      END

      INTEGER FUNCTION ELMHES(ND,N,LOW,HIGH,MAT,PERM)
C     ==============================================

C     FUNCTION FOR FUNCTION EIGEN

      INTEGER N,LOW,HIGH,PERM(N)
      DOUBLE PRECISION MAT(ND,N)
      DOUBLE PRECISION ZERO,ONE
      PARAMETER (ZERO = 0.0D0,ONE = 1.0D0)
      INTEGER I,J,M
      DOUBLE PRECISION X,Y
```

```
C
      DO 70 M=LOW+1,HIGH-1
         I = M
         X = ZERO
         DO 10 J=M,HIGH
            IF (ABS(MAT(J,M-1)) .GT. ABS(X)) THEN
               X = MAT(J,M-1)
               I=J
            ENDIF
   10       CONTINUE
         PERM(M) = I
         IF (I .NE. M) THEN
C
C           ZEILEN UND SPALTEN VON MAT VERTAUSCHEN
C
            DO 20 J=M-1,N
   20          CALL SWAP (MAT(I,J),MAT(M,J))
            DO 30 J=1,HIGH
   30          CALL SWAP (MAT(J,I),MAT(J,M))
         ENDIF
         IF (X .NE. ZERO) THEN
            DO 60 I=M+1,HIGH
               Y = MAT(I,M-1)
               IF (Y .NE. ZERO) THEN
                  Y = Y/X
                  MAT(I,M-1) = Y
                  DO 40 J=M,N
   40                MAT(I,J) = MAT(I,J)-Y*MAT(M,J)
                  DO 50 J=1,HIGH
   50                MAT(J,M) = MAT(J,M)+Y*MAT(J,I)
               ENDIF
   60       CONTINUE
         ENDIF
   70 CONTINUE
      ELMHES = 0
      END

      INTEGER FUNCTION ELMTRA(ND,N,LOW,HIGH,MAT,PERM,H)
C     ===============================================
C     FUNCTION FOR FUNCTION EIGEN

      INTEGER ND,N,LOW,HIGH,PERM(N)
      DOUBLE PRECISION MAT(ND,N),H(ND,N)
      DOUBLE PRECISION ZERO,ONE
      PARAMETER (ZERO = 0.0D0,ONE = 1.0D0)
      INTEGER I,J,K
      DO 20 I=1,N
         DO 10 J=1,N
   10       H(I,J) = ZERO
         H(I,I) = ONE
   20 CONTINUE
```

```
C
      DO 50 I=HIGH-1,LOW+1,-1
         J=PERM(I)
         DO 30 K=I+1,HIGH
   30       H(K,I) = MAT(K,I-1)
         IF (I .NE. J) THEN
            DO 40 K=I,HIGH
               H(I,K)=H(J,K)
               H(J,K)=ZERO
   40          CONTINUE
            H(J,I) = ONE
         ENDIF
   50    CONTINUE
      ELMTRA = 0
      END

      INTEGER FUNCTION HQR2 (ND,N,LOW,HIGH,H,WERTR,WERTI,EIVEC,CNT,EPS)
C     ================================================================

C     FUNCTION FOR FUNCTION EIGEN

      INTEGER ND,N,LOW,HIGH,CNT(N)
      DOUBLE PRECISION H(ND,N),EIVEC(ND,N),WERTR(N),WERTI(N),EPS
      DOUBLE PRECISION ZERO,ONE,TWO,PT75,PT4375
      INTEGER MAXSTP
      PARAMETER (ZERO = 0.0D0,ONE = 1.0D0,TWO = 2.0D0,
     *           PT75 = 0.75D0,PT4375 = 0.4375D0,MAXSTP = 100)
      INTEGER I,J,K,L,M,NA,EN,ITER
      DOUBLE PRECISION P,Q,R,S,T,W,X,Y,Z,NORM,RA,SA,VR,VI
C
C     FEHLER 1: DIE PARAMETER N, LOW ODER HIGH HABEN UNERLAUBTE
C     WERTE:
C
      IF (N .LT. 1 .OR. LOW .LT. 1 .OR. HIGH .GT. N) THEN
         HQR2 = 1
         RETURN
      ENDIF
C
C     VORBESETZUNG FUER DIE BEI DER AUSBALANCIERUNG GEFUNDENEN
C     ISOLIERTEN EIGENWERTE:
C
      DO 10 I=1,N
         IF (I .LT. LOW .OR. I .GT. HIGH) THEN
            WERTR(I) = H(I,I)
            WERTI(I) = ZERO
            CNT(I) = 0
         ELSE
            CNT(I) = -1
         ENDIF
   10    CONTINUE
```

```
C
      EN = HIGH
      T = ZERO
   15    IF (EN .LT. LOW) GOTO 333
         ITER = 0
         NA = EN-1
C
C           NACH EINEM EINZELNEN KLEINEN SUBDIAGONALELEMENT
C           SUCHEN:
C
   20       DO 30 L=EN,LOW+1,-1
   30          IF (ABS(H(L,L-1)) .LE. EPS*
     *               (ABS(H(L-1,L-1))+ABS(H(L,L)))) GOTO 40
   40       X = H(EN,EN)
            IF (L .EQ. EN) THEN
C
C              EINE WURZEL GEFUNDEN:
C
               WERTR(EN) = X + T
               H(EN,EN) = WERTR(EN)
               WERTI(EN) = ZERO
               CNT(EN) = ITER
               EN = NA
               GOTO 15
            ENDIF
C
            Y = H(NA,NA)
            W = H(EN,NA)*H(NA,EN)
            IF (L .EQ. NA) THEN
C
C              ZWEI WURZELN GEFUNDEN:
C
               P = (Y-X)/TWO
               Q = P*P+W
               Z = SQRT(ABS(Q))
               H(EN,EN) = X+T
               X = H(EN,EN)
               H(NA,NA) = Y+T
               CNT(EN) = -ITER
               CNT(NA) = ITER
               IF (Q .GE. ZERO) THEN
C
C                 EIN REELLES PAAR GEFUNDEN:
C
                  IF (P .LT. ZERO) Z = -Z
                  Z = P+Z
                  WERTR(NA) = X+Z
                  WERTR(EN) = X-W/Z
                  WERTI(NA) = ZERO
                  WERTI(EN) = ZERO
```

```
                X = H(EN,NA)
                R = SQRT(X*X+Z*Z)
                P = X/R
                Q = Z/R
C
C               ZEILENMODIFIKATION:
C
                DO 50 J=NA,N
                   Z = H(NA,J)
                   H(NA,J) = Q*Z+P*H(EN,J)
                   H(EN,J) = Q*H(EN,J)-P*Z
   50           CONTINUE
C
C               SPALTENMODIFIKATION:
C
                DO 60 I=1,EN
                   Z = H(I,NA)
                   H(I,NA) = Q*Z+P*H(I,EN)
                   H(I,EN) = Q*H(I,EN)-P*Z
   60           CONTINUE
C
C               AKKUMULATION:
C
                DO 70 I=LOW,HIGH
                   Z = EIVEC(I,NA)
                   EIVEC(I,NA) = Q*Z+P*EIVEC(I,EN)
                   EIVEC(I,EN) = Q*EIVEC(I,EN)-P*Z
   70           CONTINUE
             ELSE
C
C               KOMPLEXES PAAR:
C
                WERTR(NA) = X+P
                WERTR(EN) = WERTR(NA)
                WERTI(NA) = Z
                WERTI(EN) = -Z
             ENDIF
             EN = EN-2
             GOTO 15
          ENDIF
C
          IF (ITER .EQ. MAXSTP) THEN
C
C            FEHLER 3: MAXIMALE SCHRITTZAHL UEBERSCHRITTEN:
C
             CNT(EN) = MAXSTP+1
             HQR2 = 3
             RETURN
          ENDIF
          IF (MOD(ITER,10) .EQ. 0 .AND. ITER .NE. 0) THEN
```

```
C
C           EINEN UNGEWOEHNLICHEN SHIFT DURCHFUEHREN:
C
            T = T+X
            DO 80 I=LOW,EN
   80          H(I,I) = H(I,I)-X
            S = ABS(H(EN,NA))+ABS(H(NA,EN-2))
            X = PT75*S
            Y = X
            W = -PT4375*S*S
         ENDIF
         ITER = ITER+1
C
C        NACH ZWEI AUFEINANDERFOLGENDEN KLEINEN
C        SUBDIAGONALELEMENTEN SUCHEN:
C
         DO 90 M=EN-2,L,-1
            Z = H(M,M)
            R = X-Z
            S = Y-Z
            P = (R*S-W)/H(M+1,M)+H(M,M+1)
            Q = H(M+1,M+1)-Z-R-S
            R = H(M+2,M+1)
            S = ABS(P)+ABS(Q)+ABS(R)
            P = P/S
            Q = Q/S
            R = R/S
            IF (M .EQ. L) GOTO 100
            IF (ABS(H(M,M-1))*(ABS(Q)+ABS(R)) .LE. EPS*ABS(P)*
     *         (ABS(H(M-1,M-1))+ABS(Z)+ABS(H(M+1,M+1))))
     *         GOTO 100
   90       CONTINUE
  100    DO 110 I=M+2,EN
  110       H(I,I-2) = ZERO
         DO 120 I=M+3,EN
  120       H(I,I-3) = ZERO
C
C        EIN DOPPELTER QR-SCHRITT, DER DIE ZEILEN L BIS EN UND
C        DIE SPALTEN M BIS EN DES GANZEN FELDES BETRIFFT:
C
         DO 200 K=M,NA
            IF (K .NE. M) THEN
               P = H(K,K-1)
               Q = H(K+1,K-1)
               IF (K .NE. NA) THEN
                  R = H(K+2,K-1)
               ELSE
                  R = ZERO
               ENDIF
               X = ABS(P)+ABS(Q)+ABS(R)
               IF (X .EQ. ZERO) GOTO 200
```

```
            P = P/X
            Q = Q/X
            R = R/X
         ENDIF
         S = SQRT(P*P+Q*Q+R*R)
         IF (P .LT. ZERO) S = -S
         IF (K .NE. M) THEN
            H(K,K-1) = -S*X
         ELSEIF (L .NE. M) THEN
            H(K,K-1) = -H(K,K-1)
         ENDIF
         P = P+S
         X = P/S
         Y = Q/S
         Z = R/S
         Q = Q/P
         R = R/P
C
C        ZEILENMODIFIKATION:
C
         DO 130 J=K,N
            P = H(K,J)+Q*H(K+1,J)
            IF (K .NE. NA) THEN
               P = P+R*H(K+2,J)
               H(K+2,J) = H(K+2,J)-P*Z
            ENDIF
            H(K+1,J) = H(K+1,J)-P*Y
            H(K,J) = H(K,J)-P*X
  130       CONTINUE
         J = MIN(K+3,EN)
C
C        SPALTENMODIFIKATION:
C
         DO 140 I=1,J
            P = X*H(I,K)+Y*H(I,K+1)
            IF (K .NE. NA) THEN
               P = P+Z*H(I,K+2)
               H(I,K+2) = H(I,K+2)-P*R
            ENDIF
            H(I,K+1) = H(I,K+1)-P*Q
            H(I,K) = H(I,K)-P
  140       CONTINUE
C
C        TRANSFORMATIONEN AKKUMULIEREN:
C
         DO 150 I=LOW,HIGH
            P = X*EIVEC(I,K)+Y*EIVEC(I,K+1)
            IF (K .NE. NA) THEN
               P = P+Z*EIVEC(I,K+2)
               EIVEC(I,K+2) = EIVEC(I,K+2)-P*R
            ENDIF
```

```
                EIVEC(I,K+1) = EIVEC(I,K+1)-P*Q
                EIVEC(I,K) = EIVEC(I,K)-P
  150           CONTINUE
  200         CONTINUE
            GOTO 20
C
C
C     ALLE WURZELN GEFUNDEN, NUN WIRD RUECKTRANSFORMIERT:
C
C     1-NORM VON H BESTIMMEN:
C
  333 NORM = ZERO
      K=1
      DO 201 I=1,N
         DO 101 J=K,N
  101       NORM = NORM+ABS(H(I,J))
  201    K = I
      IF (NORM .EQ. ZERO) THEN
C        FEHLER 2: 1-NORM VON H IST GLEICH 0:
         HQR2 = 2
         RETURN
      ENDIF
C
C     RUECKTRANSFORMATION:
C
      DO 207 EN=N,1,-1
         P = WERTR(EN)
         Q = WERTI(EN)
         NA = EN - 1
         IF (Q .EQ. ZERO) THEN
C
C           REELLER VEKTOR:
C
            M = EN
            H(EN,EN) = ONE
            DO 63 I=NA,1,-1
               W = H(I,I)-P
               R = H(I,EN)
               DO 38 J=M,NA
   38             R = R+H(I,J)*H(J,EN)
               IF (WERTI(I) .LT. ZERO) THEN
                  Z = W
                  S = R
               ELSE
                  M = I
                  IF (WERTI(I) .EQ. ZERO) THEN
                     IF (W .NE. ZERO) THEN
                        H(I,EN) = -R/W
```

```
                     ELSE
                        H(I,EN) = -R/(EPS*NORM)
                     ENDIF
                  ELSE
C
                     X = H(I,I+1)
                     Y = H(I+1,I)
                     Q = (WERTR(I)-P)*(WERTR(I) - P)+
                         WERTI(I)*WERTI(I)
                     T = (X*S-Z*R)/Q
                     H(I,EN) = T
                     IF (ABS(X) .GT. ABS(Z)) THEN
                        H(I+1,EN) = (-R-W*T)/X
                     ELSE
                        H(I+1,EN) = (-S-Y*T)/Z
                     ENDIF
                  ENDIF
               ENDIF
   63          CONTINUE
            ELSEIF (Q .LT. ZERO) THEN
C
C              KOMPLEXER VEKTOR, DER ZU LAMBDA = P - I * Q GEHOERT:
C
               M = NA
               IF (ABS(H(EN,NA)) .GT. ABS(H(NA,EN))) THEN
                  H(NA,NA) = -(H(EN,EN)-P)/H(EN,NA)
                  H(NA,EN) = -Q/H(EN,NA)
               ELSE
                  CALL COMDIV(-H(NA,EN),ZERO,H(NA,NA)-P,Q,
     *                        H(NA,NA),H(NA,EN))
               ENDIF
               H(EN,NA) = ONE
               H(EN,EN) = ZERO
               DO 190 I=NA-1,1,-1
                  W = H(I,I)-P
                  RA = H(I,EN)
                  SA = ZERO
                  DO 75 J=M,NA
                     RA = RA+H(I,J)*H(J,NA)
                     SA = SA+H(I,J)*H(J,EN)
   75                CONTINUE
                  IF (WERTI(I) .LT. ZERO) THEN
                     Z = W
                     R = RA
                     S = SA
                  ELSE
                     M = I
                     IF (WERTI(I) .EQ. ZERO) THEN
                         CALL COMDIV(-RA,-SA,W,Q,H(I,NA),H(I,EN))
                     ELSE
```

```
C
C                     LOESE DIE KOMPLEXEN GLEICHUNGEN:
C
                      X = H(I,I+1)
                      Y = H(I+1,I)
                      VR = (WERTR(I)-P)*(WERTR(I)-P)+
     *                     WERTI(I)*WERTI(I)-Q*Q
                      VI = TWO*Q*(WERTR(I)-P)
                      IF (VR .EQ. ZERO .AND. VI .EQ. ZERO) VR =
     *                   EPS*NORM*
     *                   (ABS(W)+ABS(Q)+ABS(X)+ABS(Y)+ABS(Z))
                      CALL COMDIV(X*R-Z*RA+Q*SA,X*S-Z*SA-Q*RA,
     *                            VR,VI,H(I,NA),H(I,EN))
                      IF (ABS(X) .GT. ABS(Z)+ABS(Q)) THEN
                         H(I+1,NA) = (-RA-W*H(I,NA)+Q*H(I,EN))/X
                         H(I+1,EN) = (-SA-W*H(I,EN)-Q*H(I,NA))/X
                      ELSE
                        CALL COMDIV(-R-Y*H(I,NA),-S-Y*H(I,EN),Z,Q,
     *                            H(I+1,NA),H(I+1,EN))
                      ENDIF
                   ENDIF
                ENDIF
  190           CONTINUE
            ENDIF
  207   CONTINUE
C
C     ZU DEN ISOLIERTEN WURZELN GEHOERIGE VEKTOREN:
C
      DO 230 I=1,N
         IF (I .LT. LOW .OR. I .GT. HIGH) THEN
            DO 220 J=I+1,N
  220          EIVEC(I,J) = H(I,J)
         ENDIF
  230 CONTINUE
C
C     MIT DER TRANSFORMATIONSMATRIX MULTIPLIZIEREN, UM DIE
C     VEKTOREN DER URSPRUENGLICHEN VOLLEN MATRIX ZU ERHALTEN:
C
      DO 300 J=N,LOW,-1
         IF (J .LE. HIGH) THEN
            M = J
         ELSE
            M = HIGH
         ENDIF
         L = J-1
         IF (WERTI(J) .LT. ZERO) THEN
            DO 330 I=LOW,HIGH
               Y = ZERO
               Z = ZERO
```

```
              DO 320 K=LOW,M
                 Y = Y+EIVEC(I,K)*H(K,L)
                 Z = Z+EIVEC(I,K)*H(K,J)
 320             CONTINUE
              EIVEC(I,L) = Y
              EIVEC(I,J) = Z
 330          CONTINUE
        ELSE
           IF (WERTI(J) .EQ. ZERO) THEN
              DO 350 I=LOW,HIGH
                 Z = ZERO
                 DO 340 K=LOW,M
 340                Z =Z+EIVEC(I,K)*H(K,J)
 350             EIVEC(I,J) = Z
           ENDIF
        ENDIF
 300    CONTINUE
C
C     RUECKGABE VON 0: KEIN FEHLER:
C
      HQR2 = 0
      END

      SUBROUTINE COMDIV (AR,AI,BR,BI,RESR,RESI)
C     ========================================
C     SUBROUTINE FOR FUNCTION EIGEN

      DOUBLE PRECISION AR,AI,BR,BI,RESR,RESI
      DOUBLE PRECISION ZERO
      PARAMETER (ZERO = 0.0D0)
      DOUBLE PRECISION TEMP1,TEMP2,TEMP3
C
      IF (BR .EQ. ZERO .AND. BI .EQ. ZERO) THEN
         RESR = ZERO
         RESI = ZERO
         RETURN
      ENDIF
      IF (ABS(BR) .GT. ABS(BI)) THEN
         TEMP1 = BI/BR
         TEMP2 = TEMP1*BI+BR
         TEMP3 = (AR+TEMP1*AI)/TEMP2
         RESI = (AI-TEMP1*AR)/TEMP2
         RESR = TEMP3
      ELSE
         TEMP1 = BR/BI
         TEMP2 = TEMP1*BR+BI
         TEMP3 = (TEMP1*AR+AI)/TEMP2
         RESI = (TEMP1*AI-AR)/TEMP2
         RESR = TEMP3
      ENDIF
      END
```

```
      DOUBLE PRECISION FUNCTION COMABS(AR,AI)
C     =====================================
C     FUNCTION FOR FUNCTION EIGEN

C
      DOUBLE PRECISION AR,AI
      DOUBLE PRECISION ZERO,ONE
      PARAMETER (ZERO = 0.0D0,ONE = 1.0D0)
      DOUBLE PRECISION TEMP1,TEMP2
      TEMP1 = ABS(AR)
      TEMP2 = ABS(AI)
      IF (AR .EQ. ZERO .OR. AI .EQ. ZERO) THEN
         COMABS = ZERO
         RETURN
      ENDIF
      IF (TEMP2 .GT. TEMP1) CALL SWAP (TEMP1,TEMP2)
      IF (TEMP2 .EQ. ZERO) THEN
         COMABS = TEMP1
      ELSE
         COMABS = TEMP1*SQRT(ONE+(TEMP2/TEMP1)**2)
      ENDIF
      END

      INTEGER FUNCTION NORMAL (ND,N,V,WI)
C     ==================================
C     FUNCTION FOR FUNCTION EIGEN

      INTEGER ND,N
      DOUBLE PRECISION V(ND,N),WI(N)
      DOUBLE PRECISION ZERO,ONE
      PARAMETER (ZERO = 0.0D0,ONE = 1.0D0)
      INTEGER I,J
      DOUBLE PRECISION MAXI,TR,TI,COMABS
C
      J = 1
   10 IF (J .GT. N) GOTO 80
         IF (WI(J) .EQ. ZERO) THEN
            MAXI = V(1,J)
            DO 15 I=2,N
   15          IF (ABS(V(I,J)) .GT. ABS(MAXI)) MAXI = V(I,J)
            IF (MAXI .NE. ZERO) THEN
               MAXI = ONE/MAXI
               DO 20 I=1,N
   20             V(I,J) = V(I,J)*MAXI
            ENDIF
            J = J+1
         ELSE
            TR = V(1,J)
            TI = V(1,J+1)
```

```
          DO 30 I=2,N
             IF (COMABS(V(I,J),V(I,J+1)) .GT. COMABS(TR,TI))
     *       THEN
                TR = V(I,J)
                TI = V(I,J+1)
             ENDIF
 30          CONTINUE
          IF (TR .NE. ZERO .OR. TI .NE. ZERO) THEN
             DO 40 I=1,N
 40             CALL COMDIV (V(I,J),V(I,J+1),TR,TI,
     *                       V(I,J),V(I,J+1))
          ENDIF
          J = J+2
       ENDIF
       GOTO 10
 80 NORMAL = 0
    END

    SUBROUTINE SWAP(X,Y)
C   ====================
C   SUBROUTINE FOR FUNCTION EIGEN

    DOUBLE PRECISION X,Y,TEMP
    TEMP = X
    X = Y
    Y = TEMP
    END
```

SUBROUTINES FOR INITIAL VALUE PROBLEMS OF FIRST ORDER DIFFERENTIAL EQ.
****
TAKEN FROM THE MATHEMATICAL SOFTWARE LIBRARY UNIVERSITY AACHEN.

```
C AWP      ANFANGSWERTPROBLEM-LOESER (AUCH FUER SYSTEME)                    85
      SUBROUTINE  AWP(XK, HK, YK, N, DGL, XENDE, EPSABS, EPSREL,
     1        INDEX, IFMAX, IFANZ, IFEHL)
C
C*******************************************************************
C
C  DIESES PROGRAMM BERECHNET, AUSGEHEND VON DER NAEHERUNG YK
C  FUER DIE LOESUNG  Y  DES SYSTEMS GEWOEHNLICHER DIFFERENTIAL-
C  GLEICHUNGEN 1.ORDNUNG
C                         Y' = F(X,Y)
C  IM PUNKT XK, EINE NAEHERUNG  FUER DIE LOESUNG  Y  IM PUNKT
C  XENDE. DABEI WIRD INTERN MIT SCHRITTWEITENSTEUERUNG SO GE-
C  RECHNET, DASS DER FEHLER DER BERECHNETEN NAEHERUNG ABSOLUT
C  ODER RELATIV IN DER GROESSENORDNUNG DER VORGEGEBENEN FEHLER-
C  SCHRANKEN  EPSABS  UND  EPSREL  LIEGT.
C  ( FUER DIE VARIABLENTYPEN DER UEBERGABEPARAMETER GELTEN DIE
C    STANDARD-FORTRAN-KONVENTIONEN. )
C
C
C  EINGABEPARAMETER:
C  ================
C  XK     - AUSGANGSPUNKT DER UNABHAENGIGEN VARIABLEN  X
C  HK     - PROGNOSTIZIERTE SCHRITTWEITE FUER DEN NAECHSTEN
C           SCHRITT
C  YK     - 1-DIM. FELD (1:N) ;WERT FUER DIE LOESUNG DER DIFFE-
C           RENTIALGLEICHUNG AN DER STELLE  XK
C  N      - ANZAHL DER DIFFERENTIALGLEICHUNGEN ( 1 <= N <= 20 )
C  DGL    - RECHTE SEITE DER DIFFERENTIALGLEICHUNG, DIE ALS
C           UNTERPROGRAMM DER FORM
C              S U B R O U T I N E   D G L  (X, Y, N, F)
C           ( BEGINNEND MIT:  REAL  Y(N), F(N) ) ZUR VERFUEGUNG
C           GESTELLT SEIN MUSS, WOBEI  F  DER WERT DER RECHTEN
C           SEITE DER DIFFERENTIALGLEICHUNG AN DER STELLE  (X,Y)
C           IST  ( DGL  MUSS IM RUFENDEN PROGRAMM IN EINER
C           EXTERNAL-ANWEISUNG AUFGEFUEHRT SEIN ! )
C  XENDE - STELLE, AN DER DIE LOESUNG GEWUENSCHT WIRD; XENDE
C           DARF NICHT KLEINER ALS  XK  GEWAEHLT WERDEN
C EPSABS - FEHLERSCHRANKE FUER DIE ABSOLUTE GENAUIGKEIT DER ZU
C           BERECHNENDEN LOESUNG. ES MUSS  EPSABS >= 0 SEIN; FUER
C           EPSABS = 0 WIRD NUR DIE RELATIVE GENAUIGKEIT BEACHTET
C EPSREL - FEHLERSCHRANKE FUER DIE RELATIVE GENAUIGKEIT DER ZU
C           BERECHNENDEN LOESUNG. ES MUSS  EPSREL >= 0 SEIN; FUER
C           EPSREL = 0 WIRD NUR DIE ABSOLUTE GENAUIGKEIT BEACHTET
```

```
C  INDEX - WAHL DER ZU BENUTZENDEN EINBETTUNGSFORMEL MIT
C          SCHRITTWEITENSTEUERUNG:
C              = 0: RUNGE-KUTTA-VERFAHREN  2./3. ORDNUNG
C            SONST: FORMEL VON ENGLAND  4./5. ORDNUNG
C  IFMAX - OBERE SCHRANKE FUER DIE ANZAHL DER ZULAESSIGEN
C          FUNKTIONSAUSWERTUNGEN DER RECHTEN SEITE  F  DER DIF-
C          FERENTIALGLEICHUNG
C
C
C  AUSGABEPARAMETER:
C  =================
C  XK    - STELLE, DIE BEI DER INTEGRATION ZULETZT ERREICHT WUR-
C          DE. IM FALL  IFEHL = 0  IST NORMALERWEISE  XK = XENDE
C  HK    - ZULETZT VERWENDETE LOKALE SCHRITTWEITE (SIE SOLLTE
C          FUER DEN NAECHSTEN SCHRITT UNVERAENDERT GELASSEN
C          WERDEN!)
C  YK    - NAEHERUNGSWERT DER LOESUNG AN DER NEUEN STELLE XK
C  IFANZ - ANZAHL TATSAECHLICH BENOETIGTER FUNKTIONSAUSWERTUNGEN
C  IFEHL - FEHLERPARAMETER:
C          = 0: ALLES O.K.
C          = 1: BEIDE FEHLERSCHRANKEN  EPS...  ZU KLEIN
C                                  (RELATIV ZUR RECHENGENAUIGKEIT)
C          = 2: XENDE <= XK    (INNERHALB DER RECHENGENAUIGKEIT)
C          = 3: SCHRITTWEITE  HK <= 0
C                                  (RELATIV ZUR RECHENGENAUIGKEIT)
C          = 4: N > 20  ODER  N <= 0
C          = 5: IFANZ > IFMAX:  DIE ANZAHL DER ZULAESSIGEN FUNK-
C               TIONSAUSWERTUNGEN REICHT NICHT AUS, EINE GEEIG-
C               NETE NAEHERUNGSLOESUNG MIT DER GEFORDERTEN GE-
C               NAUIGKEIT ZU BESTIMMEN; XK  UND  HK  ENTHALTEN
C               DIE AKTUELLEN WERTE BEIM ABBRUCH
C
C
C  BENOETIGTE UNTERPROGRAMME:
C  ==========================
C  REAL FUNCTION NORM
C  SUBROUTINE  RUKU23
C  SUBROUTINE  ENGL45
C
C
C  RECHNERABHAENGIGE LOKALE VARIABLE:
C  ----------------------------------
C  MASCHINENGENAUIGKEIT EPSLON, EPS1, EPS2
C
C*******************************************************************
C
C  AUTOR :   KLAUS NIEDERDRENK , FTN5 , 1.11.1985
C
C*******************************************************************
```

```
C
      REAL YK(N)
      REAL Y(20), YT(20), Y00(20), NORM
      LOGICAL IEND, LEPSI
      EXTERNAL DGL
      SAVE EPSLON, EPS1, EPS2, LEPSI
      DATA  LEPSI / .TRUE. / , Y00 / 20 * 0.0 /
C
C** EPSLON IST DIE MASCHINENGENAUIGKEIT DER BENUTZTEN ANLAGE
C** (DH. DIE KLEINSTE POSITIVE MASCHINENZAHL, DIE  1 + EPSLON > 1
C** ERFUELLT). EPS1 FAENGT EINE MOEGLICHERWEISE ZU KLEINE SCHRITT-
C** WEITE HK AM INTERVALLENDE AB, EPS2 DIENT BEI ABFRAGEN AUF NULL
C
      IF ( LEPSI ) THEN
        EPSLON = 1.0
  10    EPSLON = 0.5 * EPSLON
        IF ( 1.0 + EPSLON .NE. 1.0) GOTO 10
        EPSLON = 2.0 * EPSLON
        EPS1 = EPSLON ** 0.75
        EPS2 = 100.0 * EPSLON
        LEPSI = .FALSE.
      ENDIF
C
C** VORBESETZEN LOKALER GROESSEN
C
      VZ = SIGN (1.0, XENDE)
      XEND = (1.0 - VZ * EPS2) * XENDE
      IFEHL = 0
      IFANZ = 0
      IEND = .FALSE.
C
C** PLAUSIBILITAETSKONTROLLE DER EINGABEPARAMETER
C
      YMAX = NORM(YK, Y00, N)
      IF (EPSABS .LE. EPS2*YMAX .AND. EPSREL .LE. EPS2) THEN
        IFEHL = 1
      ELSE IF (XEND .LT. XK) THEN
        IFEHL = 2
      ELSE IF (HK .LT. EPS2 * ABS(XK)) THEN
        IFEHL = 3
      ELSE IF (N .LE. 0 .OR. N .GT. 20) THEN
        IFEHL = 4
      ENDIF
      IF (IFEHL .NE. 0)  R E T U R N
C
```

```
C
C*****  S T E U E R U N G S A L G O R I T H M U S  *****
C
C
        IF (XK+HK .GT. XEND) THEN
          HK = XENDE - XK
          HHILF = HK
          IEND = .TRUE.
        ENDIF
C
C** INTEGRATION AUF DEM INTERVALL *[*XK, XENDE*]* IN ANGEMES-
C** SENEN SCHRITTEN
C
   50 CONTINUE
C
C** AUFRUF DES GEWAEHLTEN EINSCHRITTVERFAHRENS
C
      IF (INDEX .EQ. 0) THEN
         CALL RUKU23 (XK, HK, YK, N, DGL, Y, YT)
         IFANZ=IFANZ+3
      ELSE
         CALL ENGL45 (XK, HK, YK, N, DGL, Y, YT)
         IFANZ=IFANZ+6
      ENDIF
C
      DIFF = NORM(Y, YT, N)
C
      IF (DIFF .LT. EPS2) THEN
         S = 2.0
      ELSE
         YMAX = NORM(YT, Y00, N)
         S = SQRT(HK * (EPSABS + EPSREL*YMAX) / DIFF)
         IF (INDEX .NE. 0)   S = SQRT(S)
      ENDIF
C
      IF (S .GT. 1.0) THEN
C
C** DER DURCHGEFUEHRTE SCHRITT MIT HK WIRD AKZEPTIERT
C
        DO 60 I = 1, N
          YK(I) = YT(I)
   60   CONTINUE
C
        XK = XK + HK
C
C** FALLS DIE INTEGRATIONSGRENZE XENDE ERREICHT WURDE ODER
C** SCHON MEHR ALS DIE ZULAESSIGEN FUNKTIONSAUSWERTUNGEN
C** GEMACHT WURDEN : RUECKSPRUNG
C
```

```
   70    IF (IEND) THEN
           HK = HHILF
           R E T U R N
         ELSE IF (IFANZ .GT. IFMAX) THEN
           IFEHL = 5
           R E T U R N
         ENDIF
C
C** DIE SCHRITTWEITE FUER DEN NAECHSTEN SCHRITT WIRD ANGEMESSEN,
C** MAXIMAL UM DEN FAKTOR  2 , VERGROESSERT
C
         HK = HK * MIN(2.0, 0.98*S)
C
         IF ((XK+HK) .GE. XEND) THEN
           HHILF = HK
           HK = XENDE - XK
           IEND = .TRUE.
C
C** FALLS MAN SCHON SEHR NAHE BEI XENDE ANGELANGT IST : RUECKSPRUNG
C
           IF (HK .LT. EPS1 * ABS(XENDE)) GOTO 70
         ENDIF
      ELSE
C
C** DER LETZTE SCHRITT WIRD NICHT AKZEPTIERT, DIE SCHRITTWEITE HK
C** WIRD VOR WIEDERHOLUNG DIESES SCHRITTES ANGEMESSEN VERKLEINERT,
C** JEDOCH HOECHSTENS HALBIERT
C
         HK = HK * MAX(0.5, 0.98*S)
         IEND = .FALSE.
      ENDIF
C
      GOTO 50
C
      END



      REAL FUNCTION NORM (F1, F2, N)
C
C*********************************************************************
C
C  DIESES PROGRAMM BESTIMMT DIE MAXIMUMNORM DER DIFFERENZ
C   F1 - F2  DER VEKTOREN  F1  UND  F2  DER LAENGE  N .
C
C*********************************************************************
C
      REAL F1(N), F2(N)
      NORM=0.0
```

```
      DO 10 I = 1, N
        NORM = MAX (NORM, ABS(F1(I)-F2(I)))
   10 CONTINUE
      R E T U R N
      END


      SUBROUTINE RUKU23 (X, H, Y, N, DGL, Y2, Y3)
C
C*****************************************************************
C
C  DIESES PROGRAMM BERECHNET, AUSGEHEND VON DER NAEHERUNG  Y
C  AN DER STELLE  X , UEBER EINE RUNGE-KUTTA-EINBETTUNGSFORMEL
C  NAEHERUNGEN  2. UND  3. ORDNUNG   Y2  UND  Y3  AN DER STELLE
C   X + H  DES UEBER   D G L   ZUR VERFUEGUNG GESTELLTEN DIFFE-
C  RENTIALGLEICHUNGSSYSTEMS  1. ORDNUNG
C                    Y' = F(X,Y)
C  VON  N  GEWOEHNLICHEN DIFFERENTIALGLEICHUNGEN  1. ORDNUNG.
C
C
C  EINGABEPARAMETER:
C  =================
C  X      - AUSGANGSPUNKT DER UNABHAENGIGEN VARIABLEN  X
C  H      - SCHRITTWEITE
C  Y      - 1-DIM. FELD (1:N) ; WERT FUER DIE LOESUNG DER DIFFE-
C           RENTIALGLEICHUNG AN DER STELLE  X
C  N      - ANZAHL DER DIFFERENTIALGLEICHUNGEN  ( 1 <= N <= 20 )
C  DGL    - RECHTE SEITE DER DIFFERENTIALGLEICHUNG, DIE ALS
C           UNTERPROGRAMM DER FORM
C              S U B R O U T I N E   D G L  (X, Y, N, F)
C           ( BEGINNEND MIT:  REAL  Y(N), F(N) ) ZUR VERFUEGUNG
C           GESTELLT SEIN MUSS, WOBEI  F  DER WERT DER RECHTEN
C           SEITE DER DIFFERENTIALGLEICHUNG AN DER STELLE  (X,Y)
C           IST  ( DGL  MUSS IM RUFENDEN PROGRAMM IN EINER
C           EXTERNAL-ANWEISUNG AUFGEFUEHRT SEIN ! )
C
C
C  AUSGABEPARAMETER:
C  =================
C  Y2     - 1-DIM. FELD (1:N) ; NAEHERUNG  2. ORDNUNG FUER DIE
C           LOESUNG DER DIFFERENTIALGLEICHUNG AN DER STELLE  X+H
C  Y3     - 1-DIM. FELD (1:N) ; NAEHERUNG  3. ORDNUNG FUER DIE
C           LOESUNG DER DIFFERENTIALGLEICHUNG AN DER STELLE  X+H
C
C*****************************************************************
C
      REAL Y(N), Y2(N), Y3(N)
      REAL K1(20), K2(20), K3(20), YHILF(20)
C
```

```
      CALL DGL (X, Y, N, K1)
      DO 10 I = 1, N
        YHILF(I)=Y(I)+H*K1(I)
   10 CONTINUE
      CALL DGL (X+H, YHILF, N, K2)
      DO 20 I = 1, N
        YHILF(I)=Y(I)+0.25*H*(K1(I)+K2(I))
   20 CONTINUE
      CALL DGL (X+0.5*H, YHILF, N, K3)
C
      DO 100 I = 1, N
        Y2(I)=Y(I)+0.5*H*(K1(I)+K2(I))
        Y3(I)=Y(I)+H/6.*(K1(I)+K2(I)+4.*K3(I))
  100 CONTINUE
      R E T U R N
      END



      SUBROUTINE ENGL45 (X, H, Y, N, DGL, Y4, Y5)
C
C*******************************************************************
C
C  DIESES PROGRAMM BERECHNET, AUSGEHEND VON DER NAEHERUNG  Y
C  AN DER STELLE  X , UEBER DIE EINBETTUNGSFORMEL VON ENGLAND
C  NAEHERUNGEN  4. UND  5. ORDNUNG   Y4  UND  Y5  AN DER STELLE
C   X + H  DES UEBER   D G L   ZUR VERFUEGUNG GESTELLTEN DIFFE-
C  RENTIALGLEICHUNGSSYSTEMS  1. ORDNUNG
C                     Y' = F(X,Y)
C  VON  N  GEWOEHNLICHEN DIFFERENTIALGLEICHUNGEN  1. ORDNUNG.
C
C
C  EINGABEPARAMETER:
C  =================
C  X      - AUSGANGSPUNKT DER UNABHAENGIGEN VARIABLEN  X
C  H      - SCHRITTWEITE
C  Y      - 1-DIM. FELD (1:N) ; WERT FUER DIE LOESUNG DER DIFFE-
C           RENTIALGLEICHUNG AN DER STELLE  X
C  N      - ANZAHL DER DIFFERENTIALGLEICHUNGEN  ( 1 <= N <= 20 )
C  DGL    - RECHTE SEITE DER DIFFERENTIALGLEICHUNG, DIE ALS
C           UNTERPROGRAMM DER FORM
C              S U B R O U T I N E   D G L  (X, Y, N, F)
C           ( BEGINNEND MIT:  REAL  Y(N), F(N) ) ZUR VERFUEGUNG
C           GESTELLT SEIN MUSS, WOBEI  F  DER WERT DER RECHTEN
C           SEITE DER DIFFERENTIALGLEICHUNG AN DER STELLE  (X,Y)
C           IST  ( DGL  MUSS IM RUFENDEN PROGRAMM IN EINER
C           EXTERNAL-ANWEISUNG AUFGEFUEHRT SEIN ! )
C
```

```
C
C  AUSGABEPARAMETER:
C  ================
C  Y4     - 1-DIM. FELD (1:N) ; NAEHERUNG  4. ORDNUNG FUER DIE
C           LOESUNG DER DIFFERENTIALGLEICHUNG AN DER STELLE  X+H
C  Y5     - 1-DIM. FELD (1:N) ; NAEHERUNG  5. ORDNUNG FUER DIE
C           LOESUNG DER DIFFERENTIALGLEICHUNG AN DER STELLE  X+H
C
C*****************************************************************
C
      REAL Y(N), Y4(N), Y5(N)
      REAL K1(20),K2(20),K3(20),K4(20),K5(20),K6(20),YHILF(20)
C
      CALL DGL (X, Y, N, K1)
      DO 10  I = 1, N
        YHILF(I)=Y(I)+0.5*H*K1(I)
   10 CONTINUE
      CALL DGL (X+0.5*H, YHILF, N, K2)
C
      DO 20  I = 1, N
        YHILF(I)=Y(I)+0.25*H*(K1(I)+K2(I))
   20 CONTINUE
      CALL DGL (X+0.5*H, YHILF, N, K3)
C
      DO 30  I = 1, N
        YHILF(I)=Y(I)+H*(-K2(I)+2.*K3(I))
   30 CONTINUE
      CALL DGL (X+H, YHILF, N, K4)
C
      DO 40  I = 1, N
        YHILF(I)=Y(I)+H/27.*(7.*K1(I)+10.*K2(I)+K4(I))
   40 CONTINUE
      CALL DGL (X+2./3.*H, YHILF, N, K5)
C
      DO 50  I = 1, N
        YHILF(I)=Y(I)+0.0016*H*(28.*K1(I)-125.*K2(I)+546.*K3(I)
     #                          +54.*K4(I)-378.*K5(I))
   50 CONTINUE
      CALL DGL (X+0.2*H, YHILF, N, K6)
C
      DO 100  I = 1, N
        Y4(I)=Y(I)+H/6.*(K1(I)+4.*K3(I)+K4(I))
        Y5(I)=Y(I)+
     #        H/336.*(14.*K1(I)+35.*K4(I)+162.*K5(I)+125.*K6(I))
  100 CONTINUE
      R E T U R N
      END
```

SUBROUTINES FOR SOLVING REAL LINEAR EQUATION SYSTEMS.
****
TAKEN FROM THE MATHEMATICAL SOFTWARE LIBRARY UNIVERSITY AACHEN.

```
      SUBROUTINE FGAUSS(N,A,IA,B,X,MARKE,D,IPIVOT)
C
C*****************************************************************
C
C     SOLVING OF A REAL LINEAR EQUATION SYSTEM
C             A * X = B
C     GAUSS ELIMINATION PROCEDURE, SEEKING PIVOT ELEMENT
C
C
C     INPUT PARAMETER:
C     ================
C     N:   ORDER OF COEFFICIENT MATRIX
C     IA:  LEADING DIMENSION OF A, AS IN CALLING PROGRAM
C          DECLARED.
C     A:   2-DIM ARRAY (IA,N)
C     B:   1-DIM ARRAY (N), RIGHT SIDE OF EQUATION SYSTEM
C
C
C     OUTPUT PARAMETER:
C     =================
C     A:      2-DIM ARRAY(IA,N), CONTAINS DECOMPOSED ORIGINAL
C             MATRIX A INTO C*B
C     X:      1-DIM ARRAY (N), CONTAINS SOLUTION OF EQUATION SYST.
C     MARKE: =1  EVEN NUMBER OF SWAPS OF COLUMNS.
C            =-1 ODD NUMBER OF SWAPS OF COLUMNS.
C            =0  INPUT MATRIX SINGULAR
C     D       1-DIM ARRAY (N), D(K) = MAX(ABS(A(K,J))
C             K=1, . .N    J=1, .. N
C     IIVOT  1-DIM INTEGER ARRAY(N) CONTAINS COLUMN INDEX
C
C     NEEDED SUBROUTINES:
C     ===================
C     FGAUSZ
C     FNACIT
C     FGAUSL
C     FKONDH
C*****************************************************************
C
C     AUTOR :    RICHARD REUTER , FTN5 , 10.2.1983
C
C*****************************************************************
C
      REAL A(IA,N),B(N),X(N),D(N)
      INTEGER IPIVOT(N)
```

```
C
C     ZERLEGUNG
C
      CALL FGAUSZ(N,A,IA,IPIVOT,MARKE,D)
C
C     LOESUNG; VORWAERTS-, RUECKWAERTSELIMINATION
C
      IF(MARKE.EQ.0) GOTO 10
        CALL FGAUSL(N,A,IA,IPIVOT,B,X)
   10 CONTINUE
      RETURN
      END



      SUBROUTINE FGAUSZ(N,A,IA,IPIVOT,MARKE,D)
C     ========================================
C     SUBROUTINE FOR SUBROUTINE FGAUSS
C
      REAL A(IA,N),D(N)
      INTEGER IPIVOT(N)
      MARKE=1
C
C     INITIALISIERUNG: PERMUTATIONSVEK., SKALIERUNGSFAKTOREN
C
      DO 30 I=1,N
         IPIVOT(I)=I
         ZMAX     =0.
         DO 10 J=1,N
            ZMAX=AMAX1(ZMAX,ABS(A(I,J)))
   10    CONTINUE
         IF(ZMAX.NE.0.) GOTO 20
            MARKE=0
            ZMAX =1.
   20    CONTINUE
         D(I)=ZMAX
   30 CONTINUE
      IF(N.LE.1) RETURN
C
C     FAKTORISIERUNG
C
      N1=N-1
      DO 120 K=1,N1
C
C       PIVOTZEILE BESTIMMEN
C
        PIVOT=ABS(A(K,K))/D(K)
        IPVT =K
        KP1  =K+1
        DO 50 I=KP1,N
           HILF=ABS(A(I,K))/D(I)
```

```
         IF(HILF.LE.PIVOT) GOTO 40
            PIVOT=HILF
            IPVT =I
   40    CONTINUE
   50 CONTINUE
      IF(PIVOT.NE.0.) GOTO 60
         MARKE=0
         GOTO 110
   60 CONTINUE
         IF(IPVT.EQ.K) GOTO 80
C
C           VERTAUSCHE K-TE MIT IPVT-TER ZEILE
C
            MARKE= -MARKE
            I            =IPIVOT(IPVT)
            IPIVOT(IPVT)=IPIVOT(K)
            IPIVOT(K)   =I
            HILF   = D(IPVT)
            D(IPVT)= D(K)
            D(K)   = HILF
            DO 70 J=1,N
               HILF      = A(IPVT,J)
               A(IPVT,J)= A(K,J)
               A(K,J)    = HILF
   70       CONTINUE
   80    CONTINUE
C
C        ELIMINATIONSSCHRITT
C
         DO 100 I=KP1,N
            A(I,K)=A(I,K)/A(K,K)
            FAK   =A(I,K)
            DO 90 J=KP1,N
               A(I,J)=A(I,J)-FAK*A(K,J)
   90       CONTINUE
  100    CONTINUE
  110  CONTINUE
  120 CONTINUE
      IF(A(N,N).EQ.0.) MARKE =0
      RETURN
      END
```

```
      SUBROUTINE FGAUSL(N,A,IA,IPIVOT,B,X)
C     ===================================
C     SUBROUTINE FOR SUBROUTINE FGAUSS

      REAL A(IA,N),B(N),X(N)
      INTEGER IPIVOT(N)
      IF(N.GT.1) GOTO 10
         X(1)=B(1)/A(1,1)
         RETURN
   10 CONTINUE
C
C     VORWAERTSELIMINATION (ZEILENVERTAUSCHUNGEN BERUECKSICHTIGEN)
C
      IPVT=IPIVOT(1)
      X(1)=B(IPVT)
      DO 30 I=2,N
         SUM=0.
         I1 =I-1
         DO 20 J=1,I1
            SUM=SUM+A(I,J)*X(J)
   20    CONTINUE
         IPVT=IPIVOT(I)
         X(I)=B(IPVT)-SUM
   30 CONTINUE
C
C     RUECKWAERTSELIMINATION
C
      X(N)=X(N)/A(N,N)
      DO 50 I=2,N
         IR  =N+1-I
         IRP1=IR+1
         SUM=0.
         DO 40 J=IRP1,N
            SUM=SUM+A(IR,J)*X(J)
   40    CONTINUE
         X(IR)=(X(IR)-SUM)/A(IR,IR)
   50 CONTINUE
      RETURN
      END


      SUBROUTINE FNACIT(N,A0,A,IA,IPIVOT,
     1                  B,X,EPS,ITMAX,KRIT,Z,R,RS)
C     ==========================================
C     SUBROUTINE FOR SUBROUTINE FGAUSS

      DIMENSION A0(IA,N),A(IA,N),B(N),X(N),Z(N),RS(N)
      INTEGER IPIVOT(N)
      DOUBLE PRECISION R(N)
      IT=0
```

```
      ZMA=1.E38
      KRIT=0
   40 CONTINUE
            DO 10 I=1,N
                  R(I)=B(I)
                  DO 20 K=1,N
                        R(I)=R(I)-DBLE(A0(I,K))*DBLE(X(K))
   20             CONTINUE
                  RS(I)=R(I)
   10       CONTINUE
            CALL FGAUSL(N,A,IA,IPIVOT,RS,Z)
            ZM=0.
            DO 30 I=1,N
                  X(I)=X(I)+Z(I)
                  ZM=AMAX1(ZM,ABS(Z(I)))
   30       CONTINUE
            IT=IT+1
      IF(ZM.GE.EPS) GOTO 50
             KRIT=1
             RETURN
   50 IF(IT.GE.ITMAX) GOTO 60
             ZMA=ZM
             GOTO 40
C     WENN ALSO IT.GE.ITMAX :
   60        KRIT=2
             IF(ZM.GT.ZMA) KRIT=3
             RETURN
      END
      SUBROUTINE FKONDH(N,A0,A,IA,MARKE,HKOND)
C     ========================================
C     SUBROUTINE FOR SUBROUTINE FGAUSS

      DIMENSION A0(IA,N),A(IA,N)
      HKOND=0.
      IF(MARKE.EQ.0) RETURN
      HKOND=1.
      DO 10 I=1,N
             ZLNORM=0.
             DO 20 K=1,N
                   ZLNORM=ZLNORM+A0(I,K)*A0(I,K)
   20        CONTINUE
             HKOND=HKOND*A(I,I)/SQRT(ZLNORM)
   10 CONTINUE
      HKOND=ABS(HKOND)
      RETURN
      END
```