7-1-1986

Steady state stress distribution and force transmissibility of a rotating disk of variable thickness

John Ole Hansen

STEADY STATE STRESS DISTRIBUTION AND FORCE TRANSMISSIBILITY OF A ROTATING DISK OF VARIABLE THICKNESS

BY

JOHN OLE HANSEN

SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

MASTER OF SCIENCE

IN

MECHANICAL ENGINEERING

APPROVED BY:

PROF. Hany Ghoneim
THESIS ADVISOR

PROF. Richard B. Hetnarski

PROF. Joseph S. Torok

PROF. Name Illegible
HEAD OF DEPARTMENT

DEPARTMENT OF MECHANICAL ENGINEERING

COLLEGE OF ENGINEERING

ROCHESTER INSTITUTE OF TECHNOLOGY

ROCHESTER, NEW YORK

JULY, 1986

Revised sample statement for granting or denying permission to reproduce an RIT thesis.

Title of Thesis STEADY STATE STRESS DISTRIBUTION AND FORCE TRANSMISSIBILITY OF A ROTATING DISK OF VARIABLE THICKNESS

I JOHN HANSEN John Hansen 8/4/86 hereby ([✓ grant], ~~deny~~) permission to the Wallace Memorial Library, of R.I.T., to reproduce my thesis in whole or in part. Any reproduction will not be for commercial use or profit.

Or

I ______________________ prefer to be contacted each time a request for reproduction is made. I can be reached at the following address. ______________________

____

Date ______________

ABSTRACT

The governing equations of the transverse vibration of a spinning disk of varying thickness are derived and solved using numerical integration techniques. A clamped-free rotating annular disk driven at the outer edge with sinusoidally varying force is considered for analysis. Representative graphs showing the stress distribution and the frequency dependence of the force transmissibility of the disk are presented. Results obtained in this paper are compared as applicable to results of previous investigations.

ACKNOWLEDGEMENT

The author wishes to express sincere appreciation to Professor H. Ghoneim for his interest and encouragement throughout this research.

The author is also indebted to Mr. Mandius Hansen and especially his wife, Judy, for their understanding and devotion.

TABLE OF CONTENTS

1. INTRODUCTION

Centrally-clamped rotating disks are the basic element of turbines, circular saw blades, grinding wheels, and computer floppy disks. Transverse vibration of these components will cause failure of turbine wheels by wheel-to-housing contact, inaccurate cuts from saw blades and grinding wheels, and memory loss in computer systems.

Several investigators analyzed the problem of transverse vibrations of spinning disks using Bessel's functions [1], Rayleigh-Ritz procedure [2], and finite element techniques [3],[4]. These previous investigations did not include inertia or shear deformation effects in the analysis. Ghosh [5] has formulated the vibration of a rotating circular disk of uniform thickness neglecting the effect of bending stiffness.

The aim of this paper is to reconfirm the results of a recent investigation [6] by reproducing the governing equations, the radial stress, circumferential stress, and force transmissibility relationships as outlined in that publication. Basic assumptions are maintained in the solution of excitation of clamped-free rotating disks in order that direct comparison of results here to those in Irie's paper [6] be possible.

The solution of the disk stress distribution and the steady-state vibration response is determined by numerical integration techniques. Therefore, the solution with this approach is exact to within the accuracy of the numerical computations and is free of the usual uncertainties of approximate methods.

Effect of disk parameters, such as outer-inner radius ratio, inside thickness-inside radius ratio, disk thickness profile, and disk angular speed, is analyzed.

2. THEORY

Consider an annular disk rotating at a constant velocity with the geometry as defined in Figure 1.

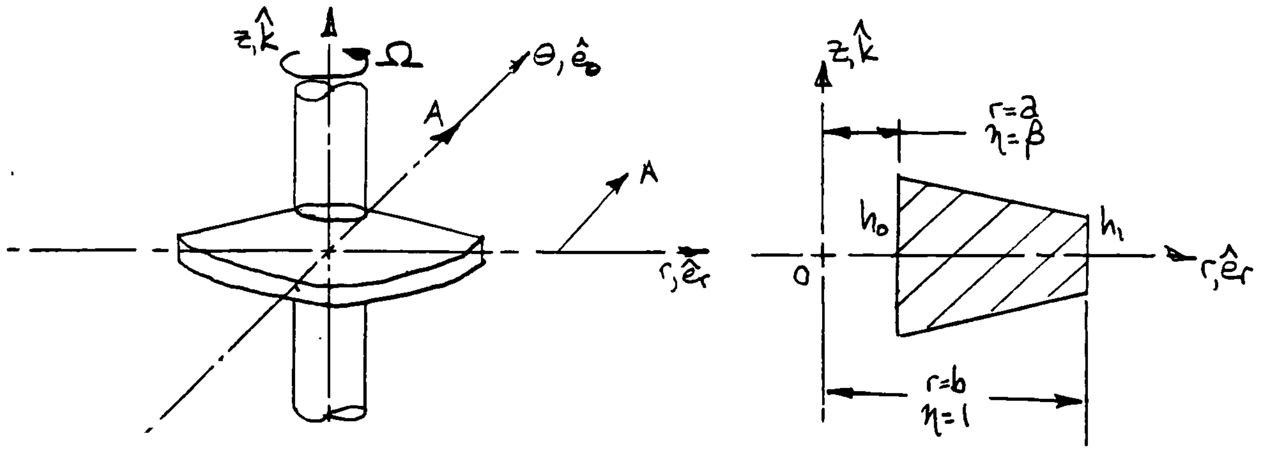


Figure 1

Describing the stress distribution on an elemental segment depicted in Figure 2 is required prior to solving the vibration equations.

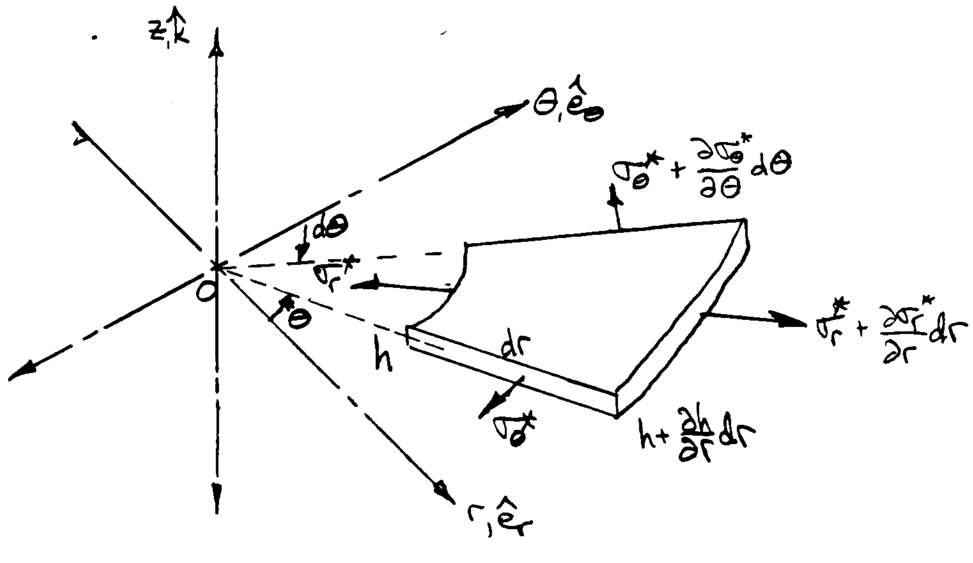


Figure 2

The asterisk denotes dimensional quantities. These parameters will be transformed to dimensionless quantities later in the derivation.

Equating the radial forces in Figure 2 yields

$$-\sigma_r^* r d\theta dh - 2\left(\sigma_\theta^* dr h \frac{d\theta}{2}\right) + \left(\sigma_r^* + \frac{\partial \sigma_r^*}{\partial r} dr\right)(r+dr)\, d\theta \left(h + \frac{\partial h}{\partial r} dr\right) = \rho h r^2 \Omega^2 dr d\theta \quad (1)$$

Simplifying Equation (1) and dividing by the factor ($drd\theta$) gives

$$-\sigma_\theta^* h + \sigma_r^* r \frac{\partial h}{\partial r} + \sigma_r^* h + \sigma_r^* \frac{\partial h}{\partial r} dr + \frac{\partial \sigma_r^*}{\partial r} r h + \frac{\partial \sigma_r^*}{\partial r} r \frac{\partial h}{\partial r} dr + \frac{\partial \sigma_r^*}{\partial r} dr h + \frac{\partial \sigma_r^*}{\partial r} dr \frac{\partial h}{\partial r} dr = \rho h r^2 \Omega^2 \quad (2)$$

The second, third, and fifth terms on the right-hand side of Equation (2) may be rewritten to provide

$$-\sigma_\theta^* h + \frac{d}{dr}(\sigma_r^* r h) + \sigma_r^* \frac{\partial h}{\partial r} dr + \frac{\partial \sigma_r^*}{\partial r} r \frac{\partial h}{\partial r} dr + \frac{\partial \sigma_r^*}{\partial r} dr h + \frac{\partial \sigma_r^*}{\partial r} dr \frac{\partial h}{\partial r} dr = \rho h r^2 \Omega^2 \quad (3)$$

Rewriting Equation (3) and neglecting higher order differential terms gives

$$\frac{d}{dr}(\sigma_r^* r h) - \sigma_\theta^* h - \rho h r^2 \Omega^2 = 0 \quad (4)$$

Expressions of linear strain for small displacements are

$$\epsilon_\theta^* = \frac{u^*}{r} \quad (5)$$

and $$\epsilon_r^* = \frac{\partial u^*}{\partial r} \quad (6)$$

For an isotropic material, stress-strain relationships are

$$\epsilon_r^* = \frac{1}{E}(\sigma_r^* - \nu \sigma_\theta^*) \quad (7)$$

and $$\epsilon_\theta^* = \frac{1}{E}(\sigma_\theta^* - \nu \sigma_r^*) \quad (8)$$

From Equations (7) and (8) the following may be derived

$$\sigma_r^* = \frac{E}{1-\nu^2}\left(\epsilon_r^* + \nu\epsilon_\theta^*\right) \tag{9}$$

and
$$\sigma_\theta^* = \frac{E}{1-\nu^2}\left(\epsilon_\theta^* + \nu\epsilon_r^*\right) \tag{10}$$

Substituting the expressions given in Equations (5) and (6) into Equations (9) and (10) gives

$$\sigma_r^* = \frac{E}{1-\nu^2}\left(\frac{\partial u^*}{\partial r} + \nu\frac{u^*}{r}\right) \tag{11}$$

and
$$\sigma_\theta^* = \frac{E}{1-\nu^2}\left(\frac{u^*}{r} + \nu\frac{\partial u^*}{\partial r}\right) \tag{12}$$

Equation (11) may be written as

$$\frac{\partial u^*}{\partial r} = -\frac{\nu}{r}u^* + \frac{1-\nu^2}{E}\sigma_r^* \tag{13}$$

Introducing the non-dimensionalized expression for radial elongation

$$u = \frac{u^*}{b} \tag{14}$$

into Equation (13) and rearranging yields

$$\frac{\partial u}{\partial(r/b)} = -\frac{\nu}{(r/b)}u + \frac{1-\nu^2}{E}\sigma_r^* \tag{15}$$

The radial and circumferential stresses are nondimensionalized using the following expressions

$$\sigma_r = \frac{b h_o^2}{D_o}\sigma_r^* \tag{16}$$

$$\sigma_\theta = \frac{b h_o^2}{D_o}\sigma_\theta^* \tag{17}$$

where $$D_0 = \frac{E h_o^3}{12(1-\nu^2)} \tag{18}$$

Substituting Equations (16) and (17) into Equation (15) gives

$$\frac{\partial u}{\partial(\frac{r}{b})} = -\frac{\nu}{(\frac{r}{b})}u + \frac{1-\nu^2}{E}\frac{D_o}{b h_o^2}\sigma_r \tag{19}$$

Simplifying gives

$$\frac{\partial u}{\partial(\frac{r}{b})} = -\frac{\nu}{(\frac{r}{b})}u + \frac{1}{12}\frac{h_o}{b}\sigma_r \tag{20}$$

Substituting the nondimensialized linear displacement variable, η, defined as

$$\eta = \frac{r}{b} \tag{21}$$

into Equation (20) yields

$$\frac{\partial u}{\partial \eta} = -\frac{\nu}{\eta}u + \frac{1}{12}\frac{h_o}{b}\sigma_r \tag{22}$$

Solving Equation (8) for σ_θ^* and substituting it in Equation (4) yields

$$\frac{d}{dr}(\sigma_r^* r h) - (E\frac{u^*}{r} + \nu\sigma_r^*)h - \rho h r^2 \Omega^2 = 0 \tag{23}$$

Expanding Equation (23) gives

$$\frac{d\sigma_r^*}{dr} r h + \sigma_r^* h + \sigma_r^* r \frac{dh}{dr} - E\frac{u^*}{r} h - \nu \sigma_r^* h - \rho h r^2 \Omega^2 = 0 \tag{24}$$

Non-dimensionalizing Equation (24) by using Equations (14), (16), (17), and (18) gives

$$\frac{D_0}{bh_0^2}\frac{d\sigma_r}{dr} r h + \frac{D_0}{bh_0^2}\sigma_r h + \frac{D_0}{bh_0^2}\sigma_r r \frac{dh}{dr} - E\frac{bu}{r} h - \frac{\nu D_0}{bh_0^2}\sigma_r h - \rho h r^2 \Omega^2 = 0 \tag{25}$$

Using Equation (21), and after algebraic manipulation, Equation (25) can be written as

$$\frac{d\sigma_r}{d\eta} = 12\frac{b(1-\nu^2)}{h_0}\frac{u}{\eta^2} - \frac{\sigma_r}{\eta}(1-\nu) - \frac{dh}{d\eta}\frac{1}{h}\sigma_r + \frac{b^3 \rho h_0^2 \Omega^2}{D_0}\eta \tag{26}$$

Introducing

$$d = \frac{h}{h_0} \quad , \tag{27}$$

$$\Lambda = \sqrt{\frac{\rho h_0^2 b^3}{D_0}}\,\Omega , \tag{28}$$

and

$$q_0 = \frac{1}{12}\left(\frac{h_0}{b}\right) , \tag{29}$$

Equation (26) becomes

$$\frac{d\sigma_r}{d\eta} = \frac{1}{q_0}\frac{(1-\nu^2)}{\eta^2}u - \frac{(1-\nu)}{\eta}\sigma_r - \frac{dd}{d\eta}\frac{1}{d}\sigma_r + \Lambda^2\eta \tag{30}$$

Equations (22) and (30) combined define the displacement and radial stress distribution of the rotating annular disk. These equations written in matrix form are

$$\frac{d}{d\eta}\begin{Bmatrix} u \\ \sigma_r \end{Bmatrix} = \begin{bmatrix} -\frac{\nu}{\eta} & q_0 \\ \frac{1}{q_0}\frac{(1-\nu^2)}{\eta^2} & -\frac{1-\nu}{\eta} - \frac{1}{d}\frac{dd}{d\eta} \end{bmatrix}\begin{Bmatrix} u \\ \sigma_r \end{Bmatrix} + \begin{bmatrix} 0 \\ \Lambda^2\eta \end{bmatrix} \tag{31}$$

These equations are solved using numerical integration techniques until the criteria defined by the disk boundary conditions are obtained.

For a disk clamped at the inner radius ($r=a, \eta=\beta$) and free at the outer radius ($r=b, \eta=1$), the boundary conditions are

$$\psi^* = \psi = 0 \quad \text{at} \quad r = a \quad (\eta = \beta)$$
$$u^* = u = 0 \quad \text{at} \quad r = a \quad (\eta = \beta)$$
$$\sigma_r^* = \sigma_r = 0 \quad \text{at} \quad r = b \quad (\eta = 1)$$

where

$$\beta = \frac{a}{b} \tag{32}$$

The equations describing the flexural vibrations of the rotating annular disk are found as follows.

Consider the disk element shown previously in Figure 2 affected by radial and transverse forces as defined in Figure 3.

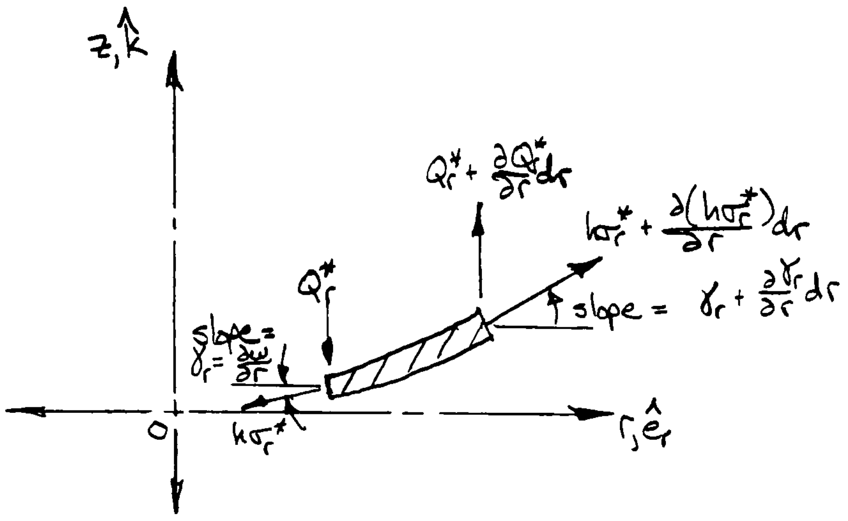

$z, \hat{k}$
$Q_r^* + \frac{\partial Q_r^*}{\partial r}dr$
$h\sigma_r^* + \frac{\partial(h\sigma_r^*)}{\partial r}dr$
Q_r^*
slope = $\gamma_r + \frac{\partial \gamma_r}{\partial r}dr$
slope= $\gamma_r = \frac{\partial w}{\partial r}$
O
$h\sigma_r^*$
$r, \hat{e}_r$

Figure 3

The equation of motion in the z-axis is

$$\left(Q_r^* + \frac{\partial Q_r^*}{\partial r}dr\right)(r+dr)d\theta - Q_r^* r\,d\theta + \left(h\sigma_r^* + \frac{\partial(h\sigma_r^*)}{\partial r}dr\right)(r+dr)d\theta\left(\gamma + \frac{\partial \gamma}{\partial r}dr\right) -$$

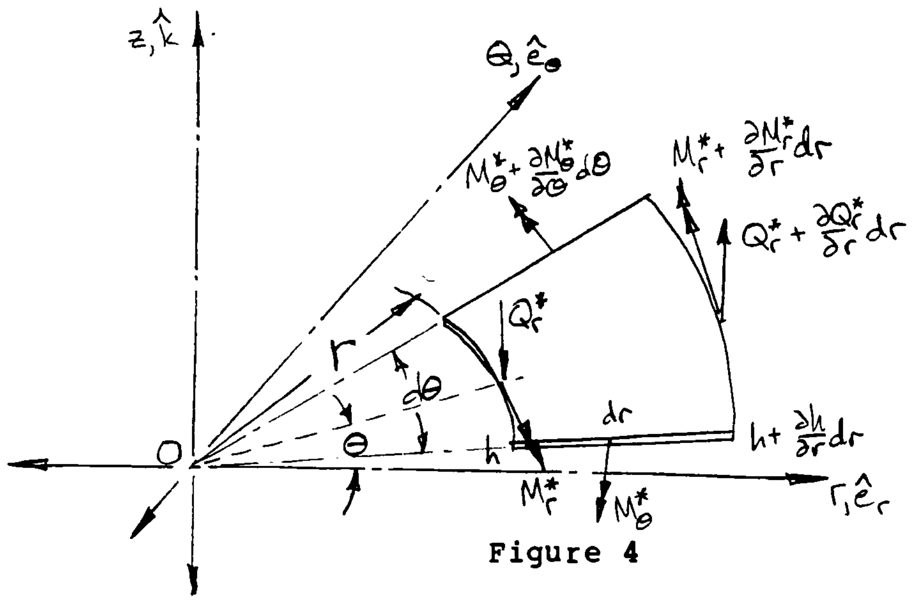
$$h\sigma_r^* r\,d\theta\,\gamma = \rho\left(r + \frac{dr}{2}\right)d\theta\,dr\,h\,\frac{\partial^2 W^*}{\partial t^2} \qquad (33)$$

After simplification and division by the factor (rdrdΘ), and incorporating the viscous damping term C_1, Equation (33) becomes

$$\frac{\partial Q_r^*}{\partial r} + \frac{Q_r^*}{r} + \frac{1}{r}\frac{\partial}{\partial r}\left(h\sigma_r^* r\,\frac{\partial W^*}{\partial r}\right) = \rho h\frac{\partial^2 W^*}{\partial t^2} + C_1\frac{\partial W^*}{\partial t} \qquad (34)$$

This paper does not consider the effect of viscous damping to the rotating disk steady-state response. Consideration of viscous terms would provide the relationship of vibration frequency effects to disk rotational speed.

Consider the freebody the same general disk element under the influence of circumferential and radial moments and transverse forces as defined in Figure 4.


$z, \hat{k}$
$\theta, \hat{e}_\theta$
$M_\theta^* + \frac{\partial M_\theta^*}{\partial \theta} d\theta$
$M_r^* + \frac{\partial M_r^*}{\partial r} dr$
$Q_r^* + \frac{\partial Q_r^*}{\partial r} dr$
Q_r^*
r
$d\theta$
θ
dr
h
$h + \frac{\partial h}{\partial r} dr$
O
M_r^*
M_θ^*
$r, \hat{e}_r$
Figure 4

Note that the disk element has a polar moment of inertia defined as

$$J = \frac{1}{12} \rho h^3 dr\, r d\theta \tag{35}$$

Summing moments in the 'θ' direction gives

$$\left(M_r^* + \frac{\partial M_r^*}{\partial r} dr\right)(r+dr)d\theta - M_r^* r d\theta - Q_r^* r dr d\theta - \frac{\partial Q_r^*}{\partial r} dr (r+dr)^2 d\theta -$$

$$\left(M_\theta^* + \frac{\partial M_\theta^*}{\partial \theta} d\theta\right) dr \frac{d\theta}{2} - M_\theta^* dr \frac{d\theta}{2} = \frac{1}{12} \rho h^3 r dr d\theta \frac{\partial^2 \varphi_r}{\partial t^2} \tag{36}$$

Simplifying Equation (36) and dividing by the term ($rdrd\theta$), and incorporating the rotational viscous damping term, gives

$$\frac{\partial M_r^*}{\partial r} + \frac{M_r^*}{r} - \frac{M_\theta^*}{r} - Q_r^* = \frac{1}{12} \rho h^3 \frac{\partial^2 \varphi_r^*}{\partial t^2} + C_2 \frac{\partial \varphi_r^*}{\partial t} \tag{37}$$

The radial component of the moment, the tangential component of moment, and the shearing force are determined, respectively, [7]

$$M_r^* = \bar{D}\left(\frac{\partial \psi_r^*}{\partial r} + \nu \frac{\psi_r^*}{r}\right) \tag{38}$$

$$M_\theta^* = \bar{D}\left(\nu \frac{\partial \psi_r^*}{\partial r} + \frac{\psi_r^*}{r}\right) \tag{39}$$

$$Q_r^* = K\bar{G}h\left(\psi_r^* + \frac{\partial W^*}{\partial r}\right) \tag{40}$$

where

$$K = \frac{\pi^2}{12}$$

The flexural rigidity, Young's modulus, and the shear modulus of an internally damped disk is assumed to be a complex expression and respectively equal to

$$\bar{D} = \frac{\bar{E}h^3}{12(1-\nu^2)} \tag{41}$$

$$\bar{E} = E(1+j\delta_E) \tag{42}$$

and

$$\bar{G} = G(1+j\delta_G) \tag{43}$$

The steady-state equations for bending moment, shearing force, slope, and vertical deflection when the disk is acted upon by an external sinusoidal force

$$F^* = \frac{D_o}{b^2} F e^{j\omega t} \tag{44}$$

are, respectively,

$$M_r^* = \frac{D_o}{b} M_r e^{j\omega t} \tag{45}$$

$$M_\theta^* = \frac{D_o}{b} M_\theta e^{j\omega t} \tag{46}$$

$$Q_{r*} = \frac{D_o}{b^2} Q_r e^{j\omega t}$$

$$= \tag{47}$$

$$\psi_r^* = \psi_r e^{j\omega t} \tag{48}$$

$$W^* = bW e^{j\omega t} \tag{49}$$

where

$$\lambda = \sqrt{\frac{\rho h_0^2 b^3}{D_0}}\,\omega \tag{50}$$

and

$$\tau = \sqrt{\frac{D_0}{\rho h_0^2 b^3}}\,t \tag{51}$$

Equations (34), (37), (38), (39), and (40) are used to develop the matrix differential equation, with M_θ^* eliminated, expressed as

$$\frac{d}{d\eta}\{Z(\eta)\} = [U(\eta)]\{Z(\eta)\} \tag{52}$$

where

$$\{Z(\eta)\} = \{M_r \; Q_r \; \psi_r \; W\}^T \tag{53}$$

and the elements of the coefficient matrix $[U(\eta)]$ are

$$U_{11} = -\frac{1-\nu}{\eta} \tag{54}$$

$$U_{12} = 1 \tag{55}$$

$$U_{13} = \frac{1-\nu^2}{\eta^2}d^3 + j\delta_E\frac{1-\nu^2}{\eta^2}d^3 - q_0\lambda^2 d^3 - 2j\delta_2\lambda \tag{56}$$

$$U_{14} = 0 \tag{57}$$

$$U_{21} = \frac{\sigma_r(1+j\delta_G)}{12q_0(1+j\delta_E)(1+j\delta_G+k_0\sigma_r)d^2} \tag{58}$$

$$U_{22} = -\frac{1}{1+j\delta_G+k_o\sigma_r}\left\{\frac{1+j\delta_G}{\eta}+\frac{k_o\sigma_r}{\eta}+\frac{dh}{d\eta}\frac{k_o\sigma_r}{h}+k_o\frac{d\sigma_r}{d\eta}\right\} \tag{59}$$

$$U_{23} = \frac{d(1+j\delta_G)}{12q_o(1+j\delta_G+k_o\sigma_r)}\left\{\frac{\sigma_r}{\eta}(1-\nu)+\frac{1}{d}\frac{dd}{d\eta}\sigma_r+\frac{d\sigma_r}{d\eta}\right\} \tag{60}$$

$$U_{24} = -\frac{(\lambda^2 d+2j\lambda\mathcal{J}_1)(1+j\delta_G)}{1+j\delta_G+k_o\sigma_r} \tag{61}$$

$$U_{31} = \frac{1-j\delta_E}{d^3(1+\delta_E^2)} \tag{62}$$

$$U_{32} = 0 \tag{63}$$

$$U_{33} = -\frac{\nu}{\eta} \tag{64}$$

$$U_{34} = 0 \tag{65}$$

$$U_{41} = 0 \tag{66}$$

$$U_{42} = \frac{k_o h_o}{db(1+j\delta_G)} \tag{67}$$

$$U_{43} = -1 \tag{68}$$

$$U_{44} = 0 \tag{69}$$

where

$$\mathcal{J}_1 = \frac{b^2 C_1}{2\sqrt{\rho h_o D_o}} \quad , \tag{70}$$

$$\mathcal{J}_2 = \frac{C_2}{2\sqrt{\rho h_o D_o}} \quad , \tag{71}$$

and

$$k_0 = \frac{2q_0}{K(1-\nu)} \qquad (72)$$

The solution of Equation (52) is accomplished by the transfer matrix approach. (Refer to Appendix V for a discussion of the transfer matrix method.)

The vector $\{Z(\eta)\}$ is written as

$$\{Z(\eta)\} = [T(\eta)]\{Z(\beta)\} \qquad (73)$$

where $[T(\eta)]$ is the transfer matrix.

Substituting Equation (73) into Equation (52) yields

$$\frac{d}{d\eta}[T(\eta)] = [U(\eta)][T(\eta)] \qquad (74)$$

To better facilitate the numerical analysis of the complex Equation (74), Equation (74) is rewritten to separate the real and imaginary components resulting in the equation

$$\frac{d}{d\eta}\begin{bmatrix} T_R(\eta) \\ T_I(\eta) \end{bmatrix} = \begin{bmatrix} U_R(\eta) & U_I(\eta) \\ -U_I(\eta) & U_R(\eta) \end{bmatrix}\begin{bmatrix} T_R(\eta) \\ T_I(\eta) \end{bmatrix} \qquad (75)$$

The values of T_R and T_I are obtained using Runge-Kutta numerical integration technique over the range $[\beta, \eta]$.

The initial condition of a free-clamped annular disk is

$$[T_R(\beta)] = [1] \qquad (76)$$

and

$$[T_I(\beta)] = [0] \qquad (77)$$

The boundary conditions are determined to be

$$\psi_r = 0 \quad \text{at} \quad \eta = \beta \qquad (78)$$

$$W = 0 \quad \text{at} \quad \eta = \beta \qquad (79)$$

$$M_r = 0 \quad \text{at} \quad \eta = 1 \tag{80}$$

$$Q_r = F \quad \text{at} \quad \eta = 1 \tag{81}$$

Substituting the above values into Equation (73) gives

$$\begin{Bmatrix} 0 \\ F \\ \psi_r \\ W \end{Bmatrix}_{(1)} = \begin{bmatrix} T_{11} & T_{12} & 0 & 0 \\ T_{21} & T_{22} & 0 & 0 \\ T_{31} & T_{32} & 0 & 0 \\ T_{41} & T_{42} & 0 & 0 \end{bmatrix}_{(1)} \begin{Bmatrix} M_r \\ Q_r \\ 0 \\ 0 \end{Bmatrix}_{(\beta)} \tag{82}$$

The complete solution of Equation (82) is obtained by first determining M_r and Q_r at $\eta = \beta$ from

$$\begin{Bmatrix} Mr \\ Qr \end{Bmatrix}_{(\beta)} = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix}^{-1}_{(1)} \begin{Bmatrix} 0 \\ F \end{Bmatrix}_{(1)} \tag{83}$$

and then ψ_r and W at $\eta = 1$ from

$$\begin{Bmatrix} \psi_r \\ W \end{Bmatrix}_{(1)} = \begin{bmatrix} T_{31} & T_{32} \\ T_{41} & T_{42} \end{bmatrix}_{(1)} \begin{Bmatrix} M_r \\ Q_r \end{Bmatrix}_{(\beta)} \tag{84}$$

The steady-state response of the disk in terms of radial bending moment, radial shear, radial slope, and deflection are given by Equations (73), (83), and (84).

The force transmissibility of the disk at $\eta = \beta$ is determined by summing moments resulting from the input force applied at the disk outer radius and the shear force at the disk center and is given by the following

$$T_F = \left| \frac{\beta Q_r(\beta)}{F} \right| . \tag{85}$$

3. ANALYSIS

An algorithm to numerically solve both the stress distribution given in Equation (31) and the resulting force transmissibility-frequency relationship of Equation (85) was programmed to run on a 16-bit, 8088 processor personal computer. The program, listed in Appendix III, is written in Pascal to take advantage of the high-level language, of the ability of utilizing a 8087 coprocessor, and of the greater precision in real algebraic operations.

A sensitivity analysis was performed on the following selected parameters: the disk outside thickness-inside thickness ratio (h_1/h_0), the disk profile (linear, exponential, and hyperbolic), and the disk inside thickness-inside radius ratio (h_0/a).

Correlation to T. Irie's results are provided when allowable. As knowledge of actual numeric values of many of the parameters used in the calculations are not known, comparison of the magnitudes of radial stress, axial stress, and force transmissibility or of the critical frequencies of is not possible.

Comparison with theoretically-determined force transmissibility profiles or stress distributions is not attempted due to the nonlinearity of the governing equations to be solved.

Analysis is performed on a free-clamped annular, rotating disk. Values of δ_E and δ_G have been assumed to be equal to each other, constant at all frequencies, and have been assigned the values of 0.01 as experimentally proposed [8] and of 0.1. The rotating disk also has been assumed to be undamped (i.e., the parameters C_1 and C_2 equal to zero). The function utilized for a linearly-varying annular disk is

$$h = h_0 \{1 - (1 - \frac{h_1}{h_0})(\frac{r - a}{b - a})\}. \qquad (86)$$

To determine the thicknesses for an exponentially-varying annular disk the equation is

$$h = h_0(h_1/h_0)^{(r - a)/(b - a)}. \qquad (87)$$

The radius-thickness relationship of a hyperbolically-varying annular disk is

$$h = h_0(r/a)^{-\log_\beta(h_1/h_0)}. \qquad (88)$$

4. RESULTS/CONCLUSIONS

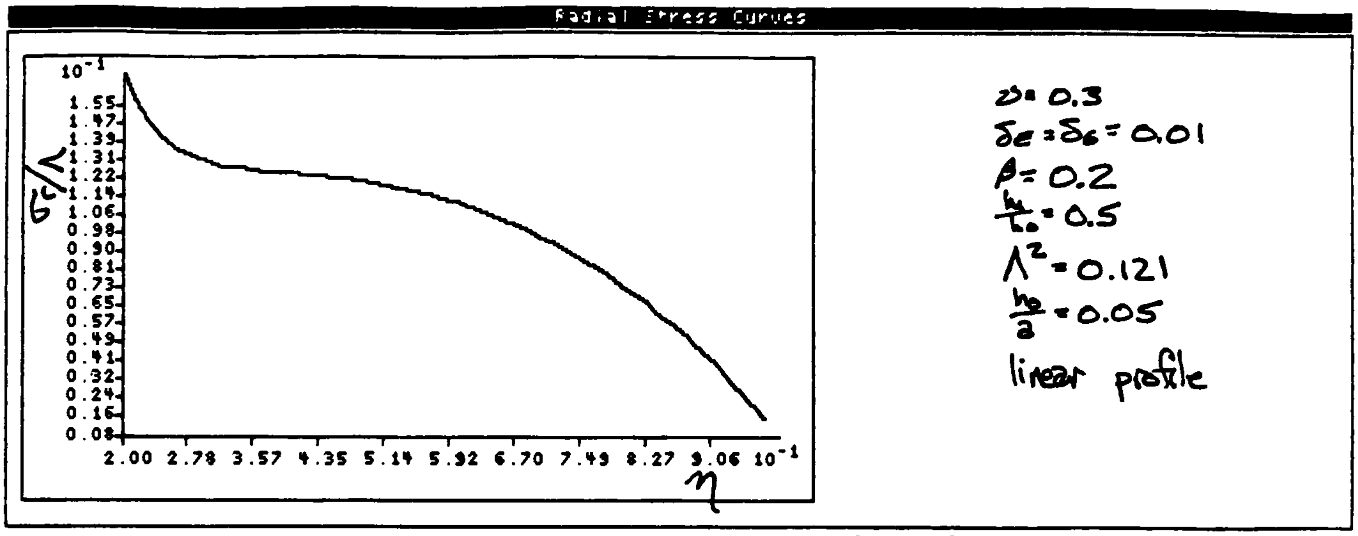
Stress distribution and force transmissibility results for disks possessing different angular velocities, inside thickness-inside radius ratios, thickness ratios, inside thickness-inside radius ratios, and profiles are displayed in Figures 6 - 13.

Figures 5a and 5b compared to Figures 6a and 6b indicate that both the radial stresses and the circumferential stresses increase with increased disk angular velocity ( ). This is accountable to the resulting increased disk angular momentum. The first three critical frequencies in Figures 5c and 6c remain essentially constant with increased disk angular velocity. However, the magnitude of force transmissibility greatly varies with identically increased disk angular velocity.

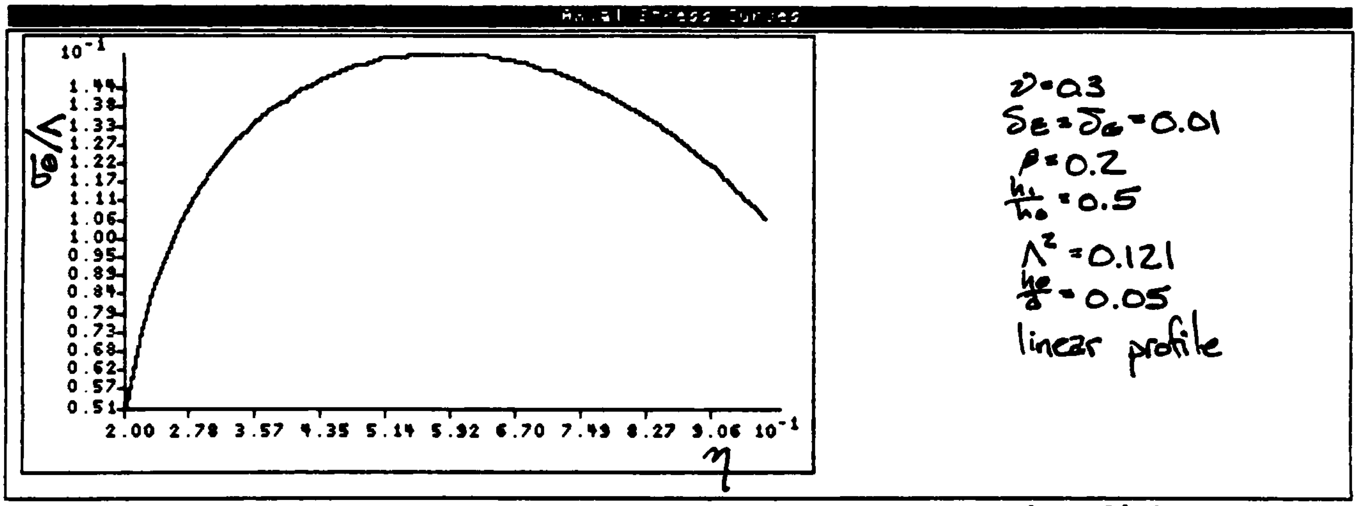
No change is apparent in either the radial or the circumferential stress as the inside thickness-to-inside radius ratio is altered as evidenced in Figures 6a and 6b versus Figures 7a and 7b. The magnitude of force transmissibility is changed considerably and the critical frequency locations of the force transmissibility peaks shift higher as this thickness-radius ratio decreases (Figures 6c and 7c).

Radial and axial stress profiles decrease with a decrease in outer radius-to-inside radius ratio (Figures 9 - 11). This corresponds with the results obtained in Irie's paper. In addition, the maximum circumferential stress value shifts toward the outer disk edge with a decrease of the h_i/h_o ratio. Figures 9c, 10c, and 11c also indicate an inverse relationship between force transmissibility and this radius ratio, and between the critical frequency values of peak force transmissibility and the radius ratio.

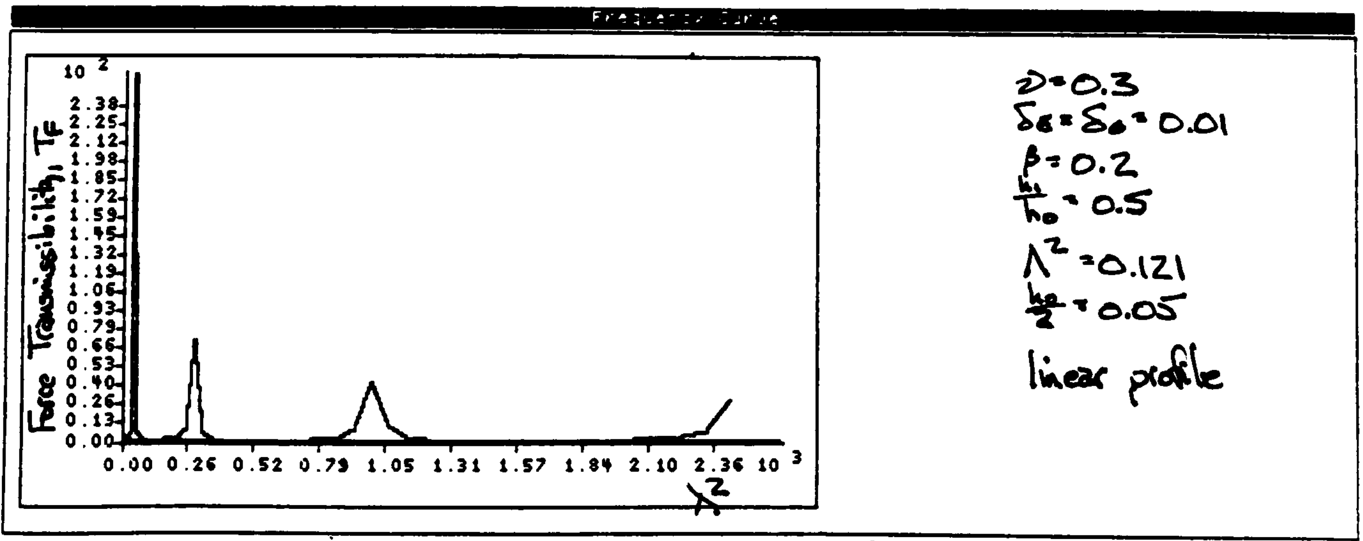
Varying the disk profile alters the radial stress, circumferential stress, and force transimissibility profiles as seen in Figures 11, 12, and 13. A disk of linearly varying thickness will possess the maximum radial stress value, while disks with hyperbolically varying thickness have the minimum stress values. Negligble effect on force transmissibility values is observed for clamped-free disks of linearly, exponentially, and hyperbolically varying thicknesses. These relationships of stress and force transmissibility profiles verifies the results in Irie's document.



a. Radial stress distribution of a rotating disk.

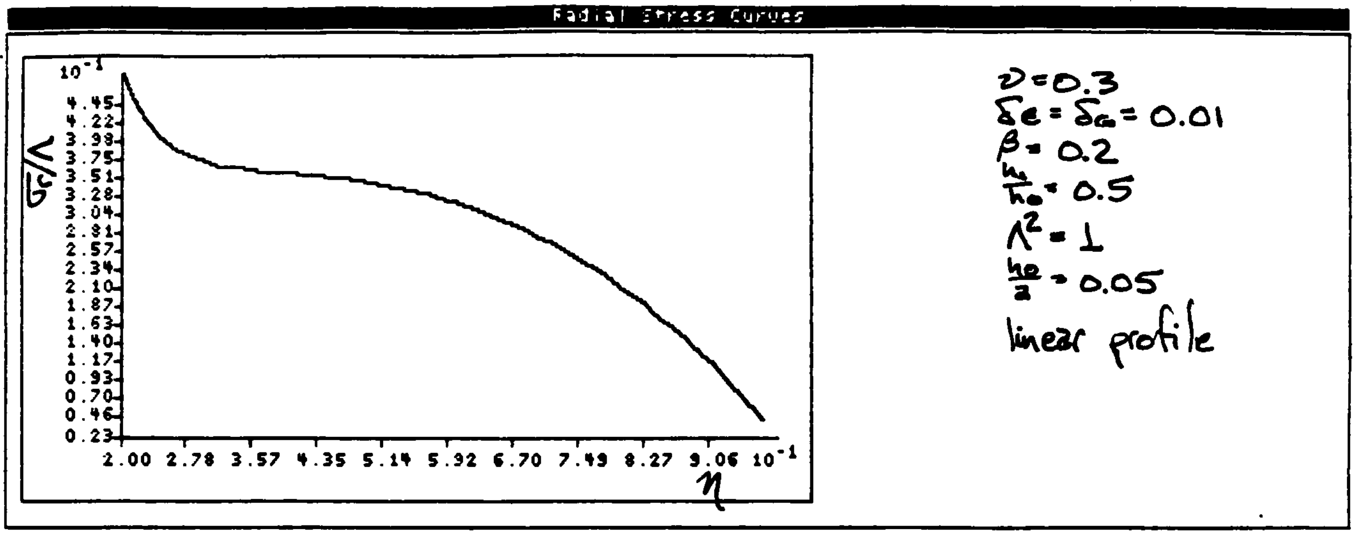


b. Circumferential stress distribution of a rotating disk.

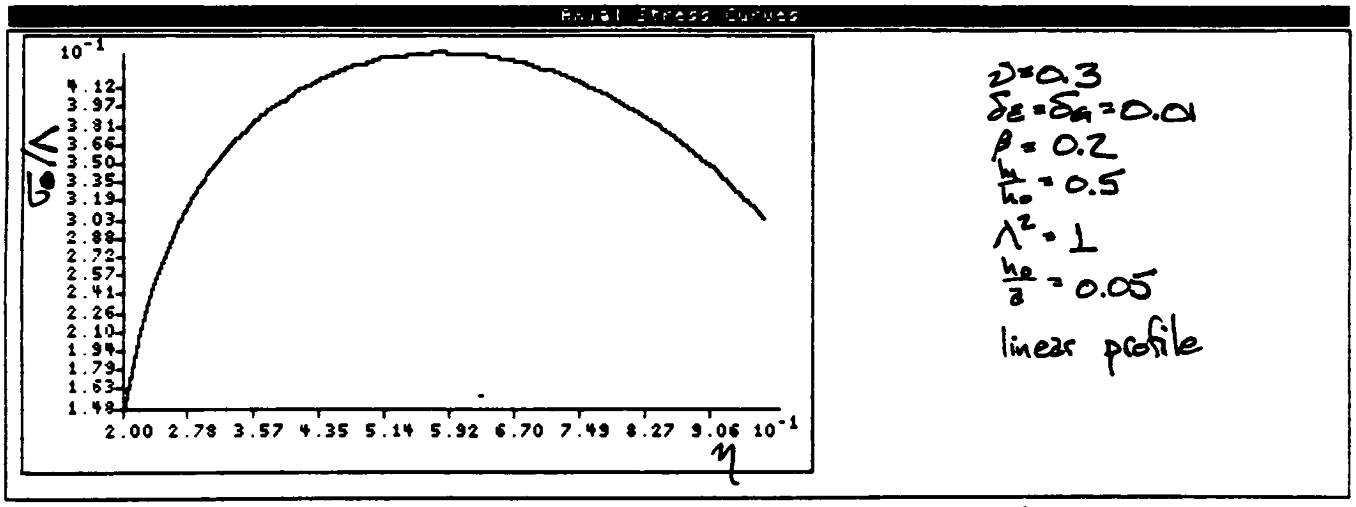


c. Steady-state response of a rotating disk.

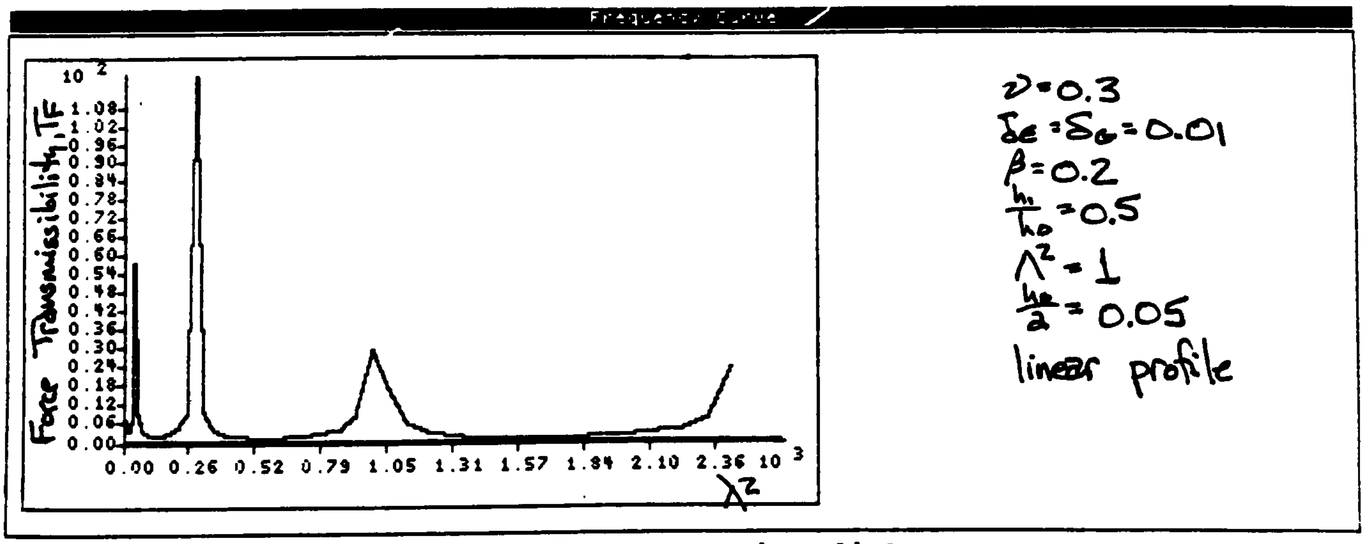
Figure 5.



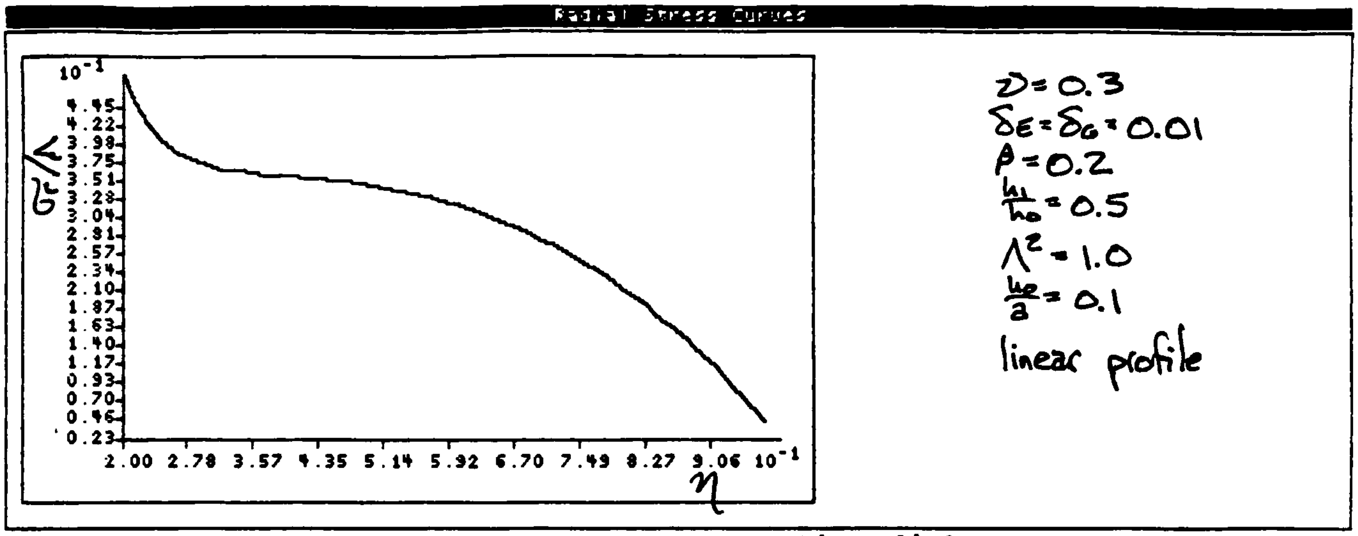
a. Radial stress distribution of a rotating disk.



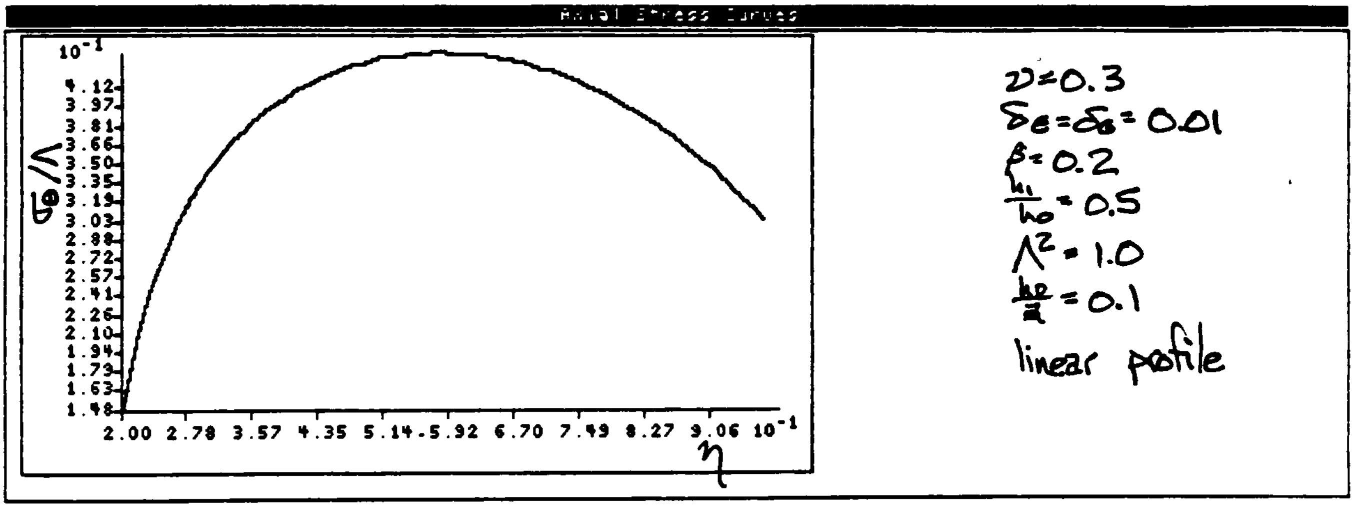
b. Circumferential stress distribution of a rotating disk.



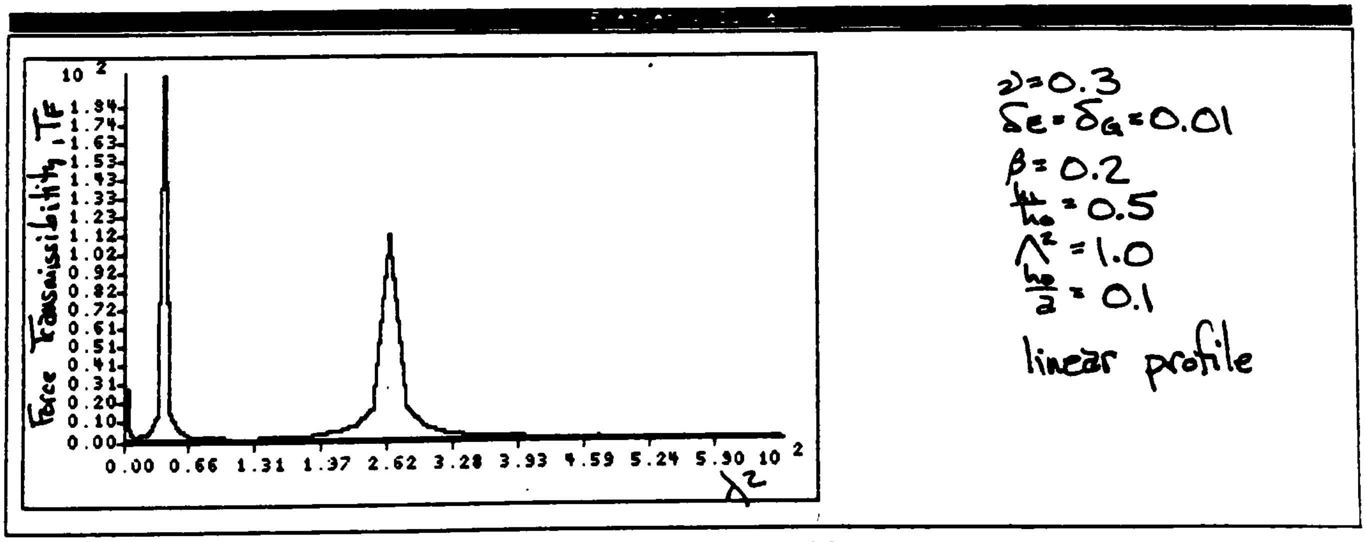
c. Steady-state response of a rotating disk.



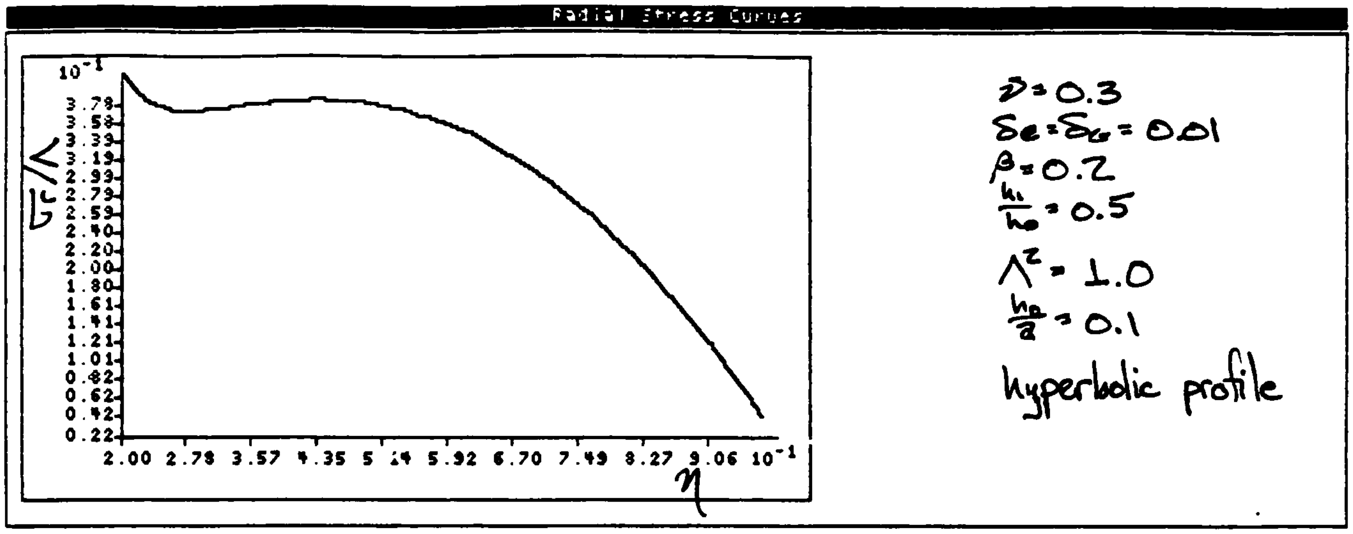
a. Radial stress distribution of a rotating disk.



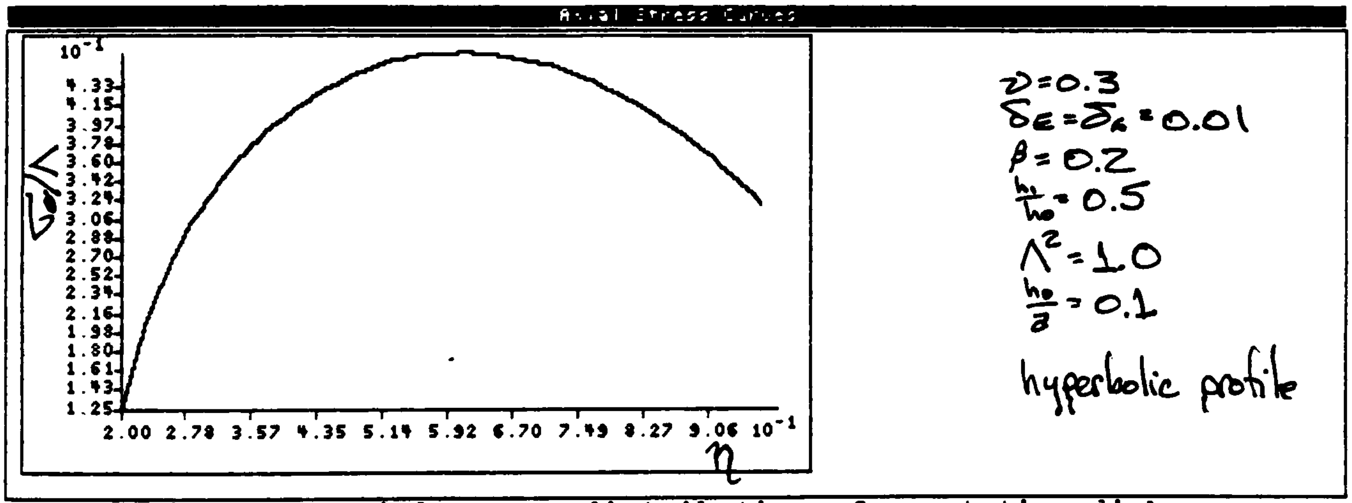
b. Circumferential stress distribution of a rotating disk.



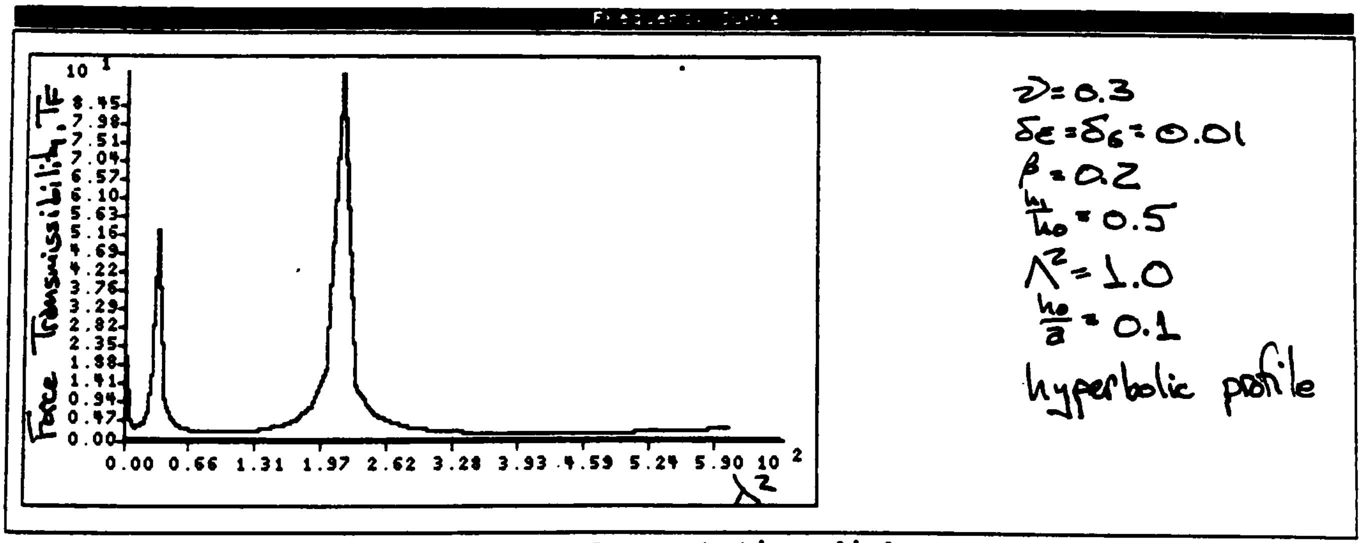
c. Steady-state response of a rotating disk.



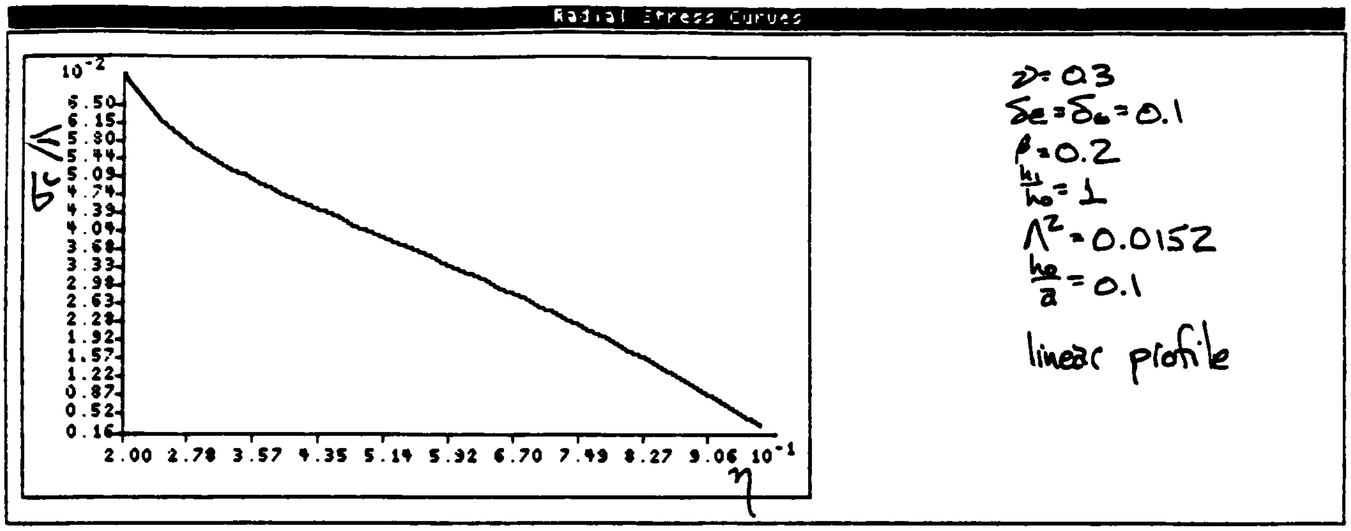
a. Radial stress distribution of a rotating disk.



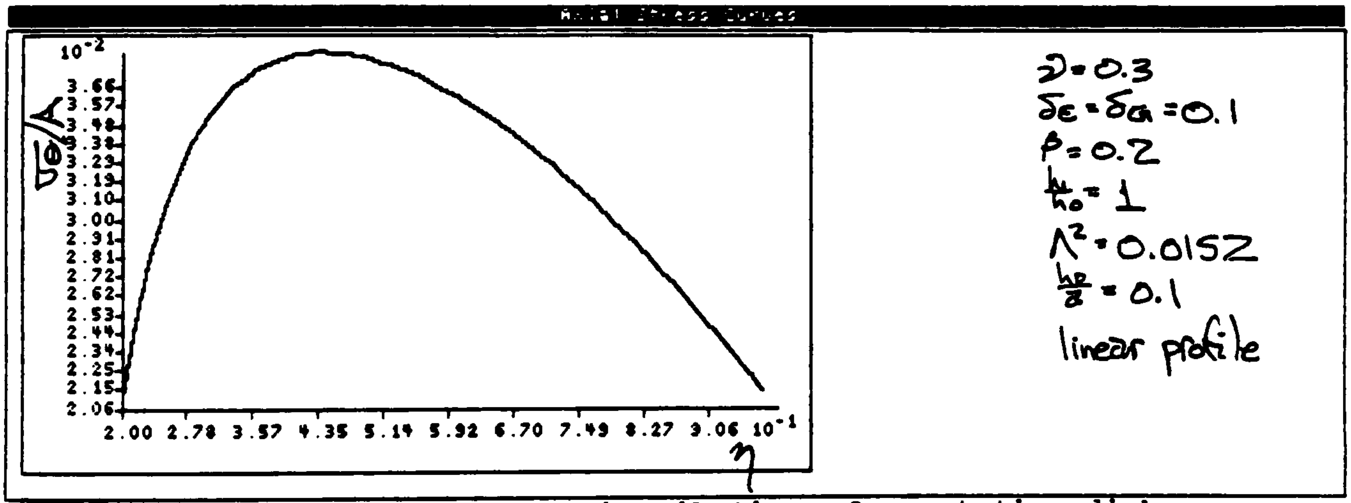
b. Circumferential stress distribution of a rotating disk.



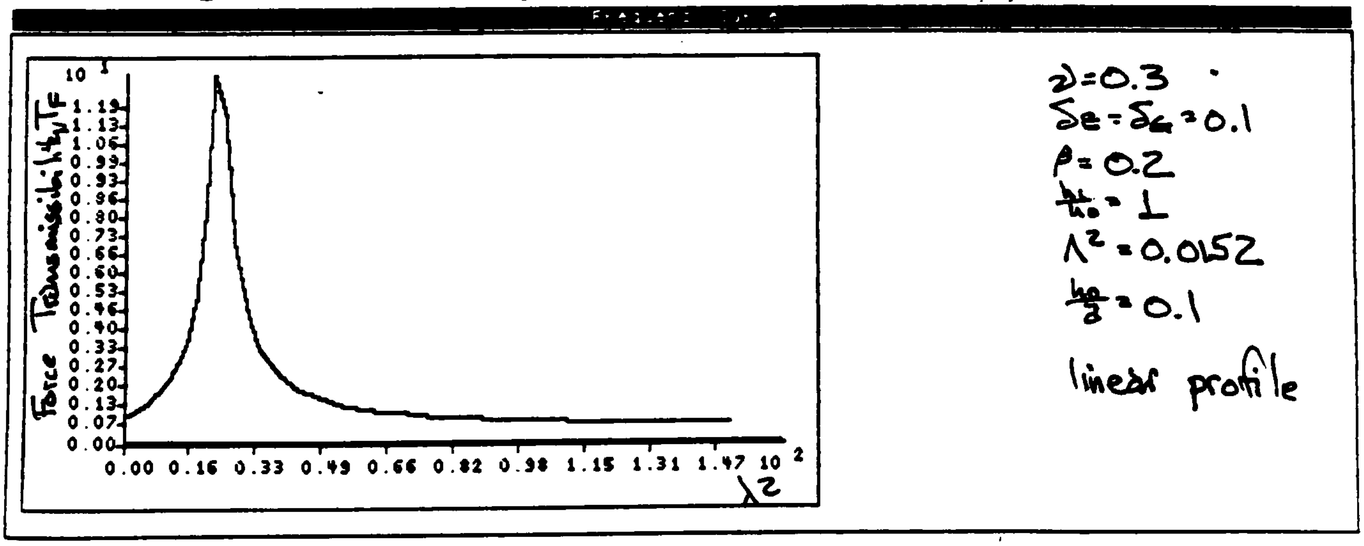
c. Steady-state response of a rotating disk.



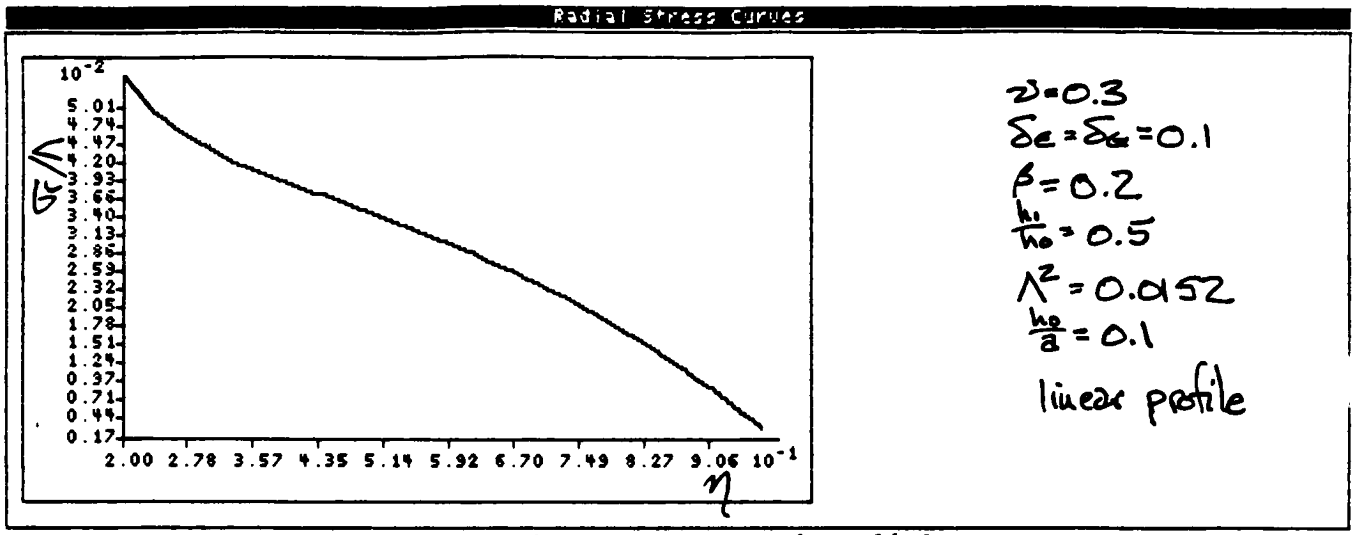
a. Radial stress distribution of a rotating disk.



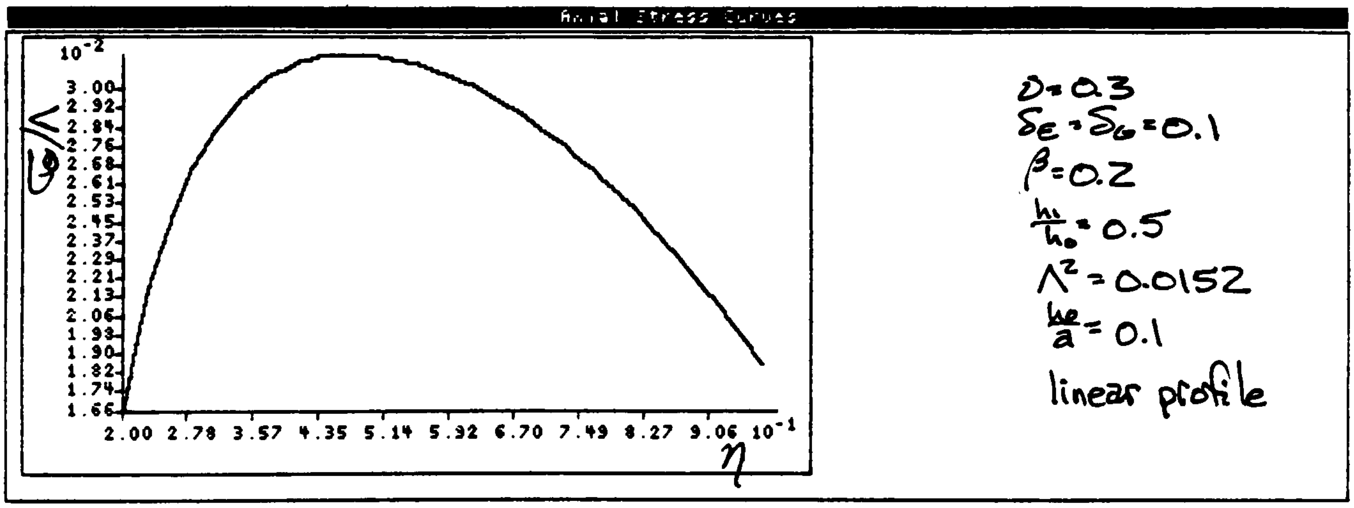
b. Circumferential stress distribution of a rotating disk.



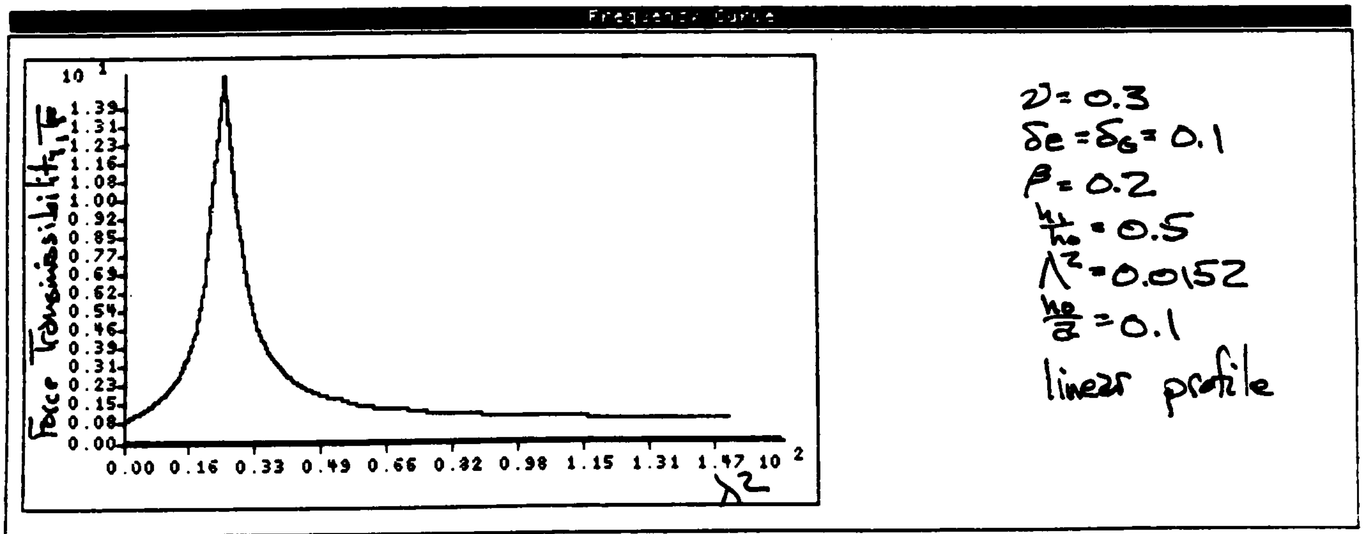
c. Steady-state response of a rotating disk.



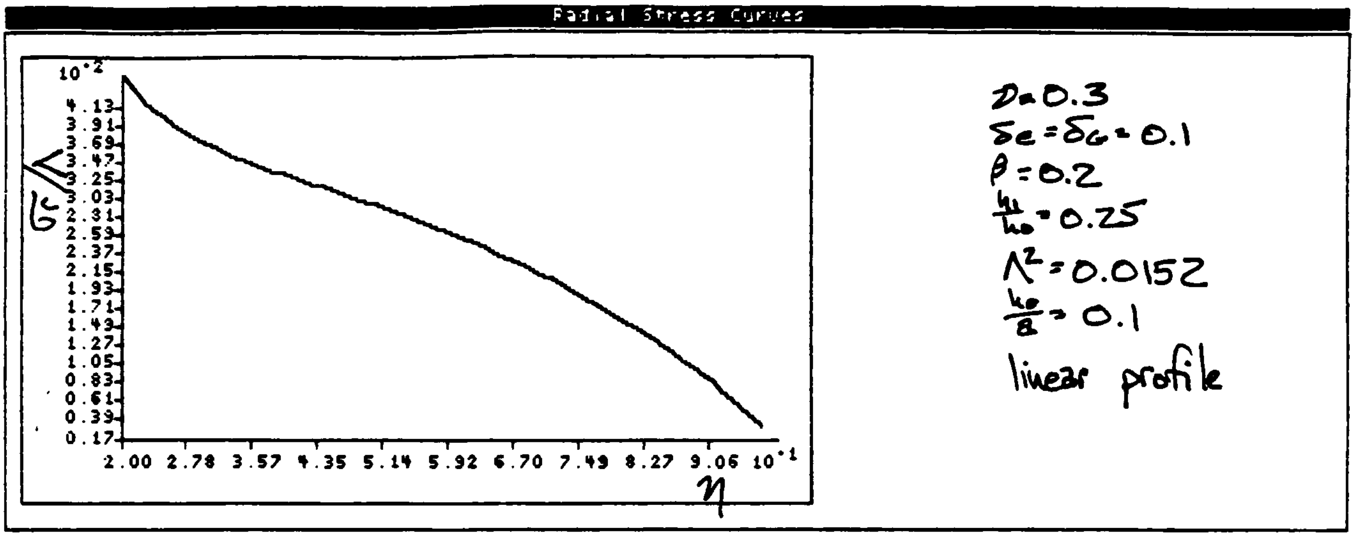
a. Radial stress distribution of a rotating disk.



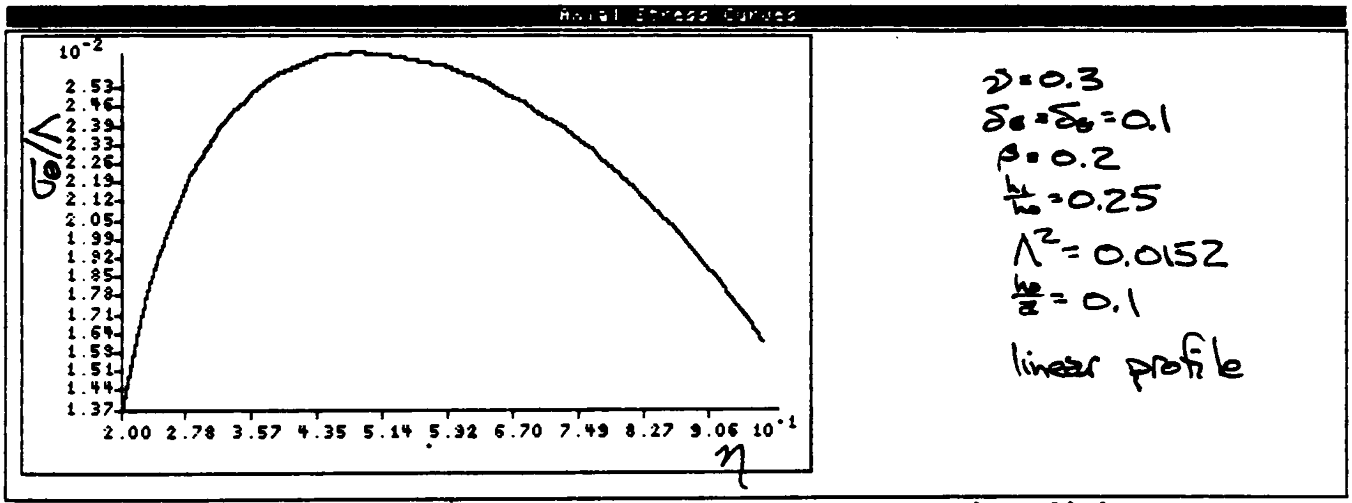
b. Circumferential stress distribution of a rotating disk.



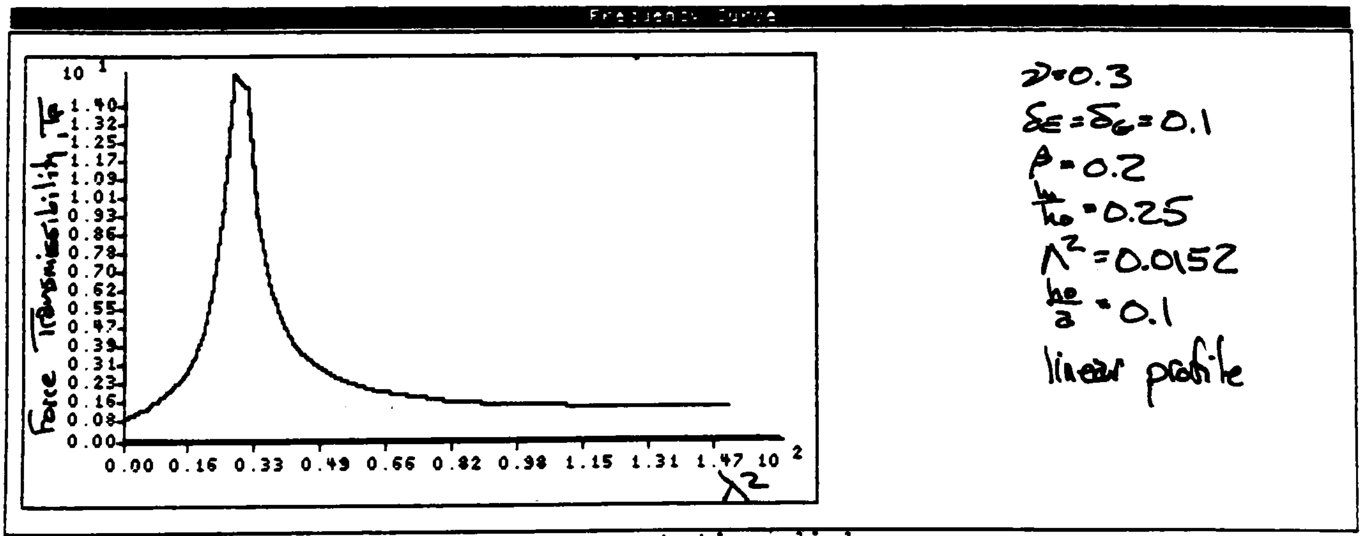
c. Steady-state response of a rotating disk.



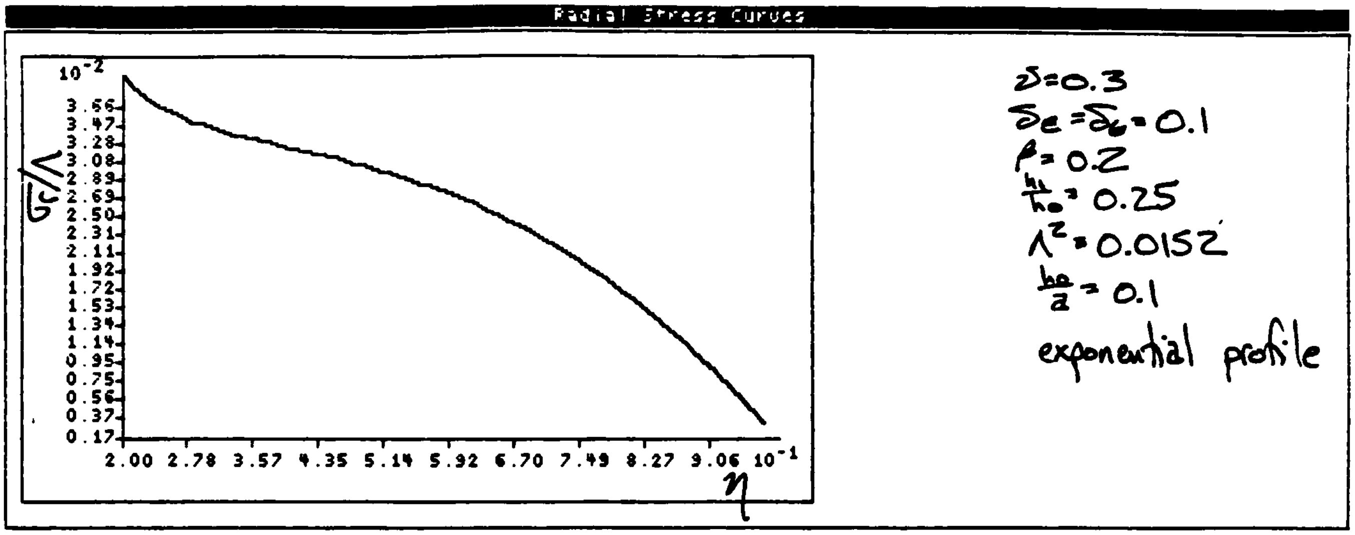
a. Radial stress distribution of a rotating disk.



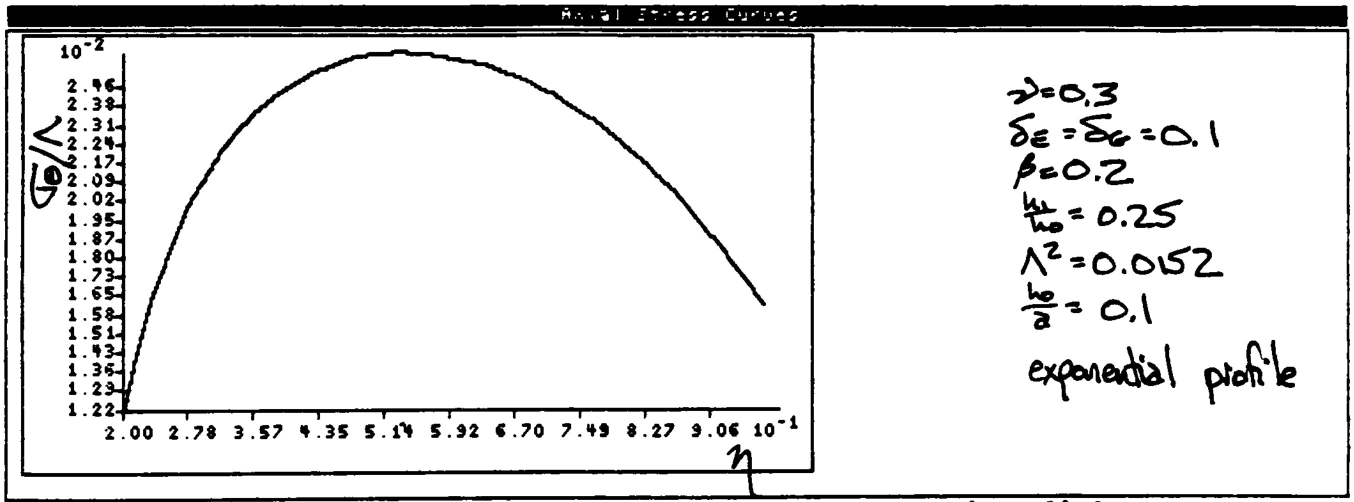
b. Circumferential stress distribution of a rotating disk.



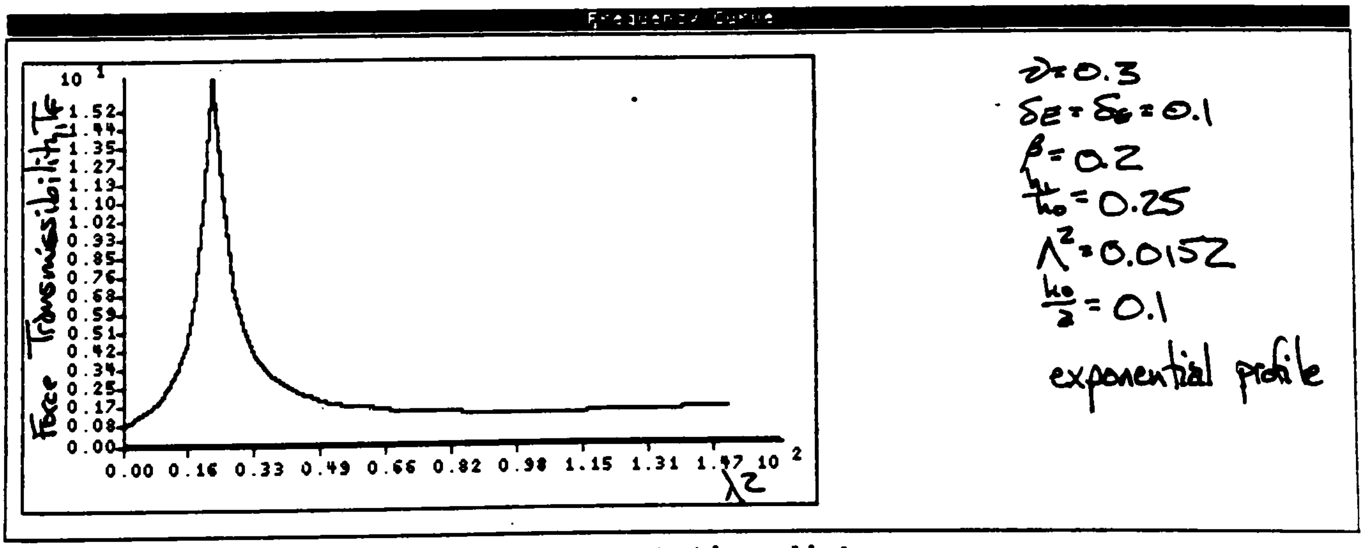
c. Steady-state response of a rotating disk.



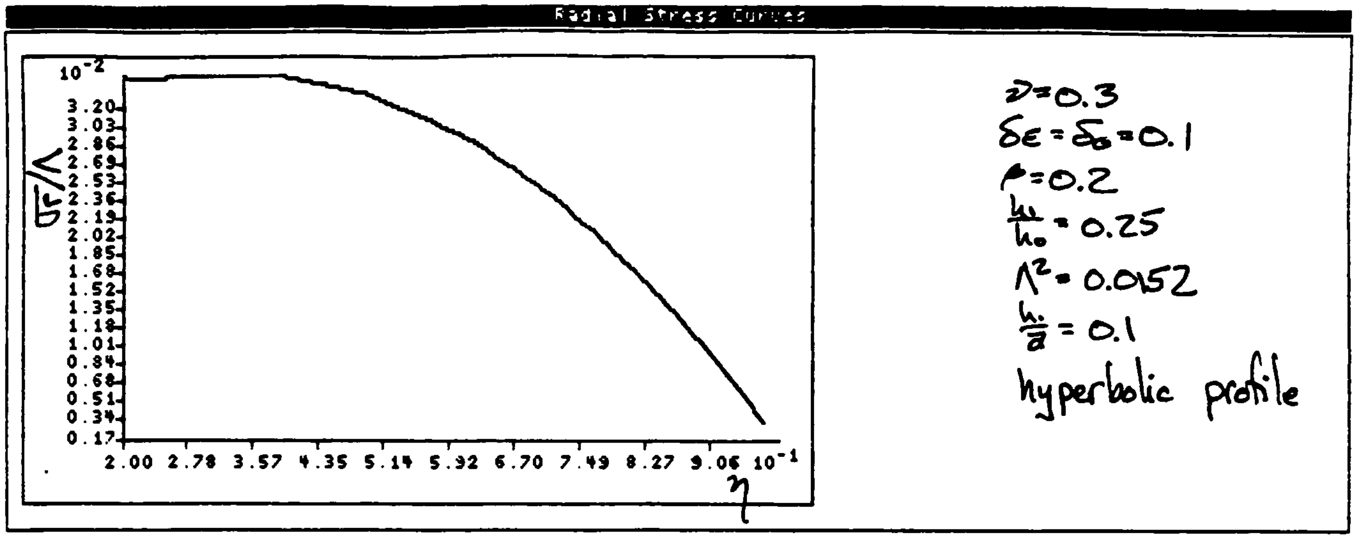
a. Radial stress distribution of a rotating disk.



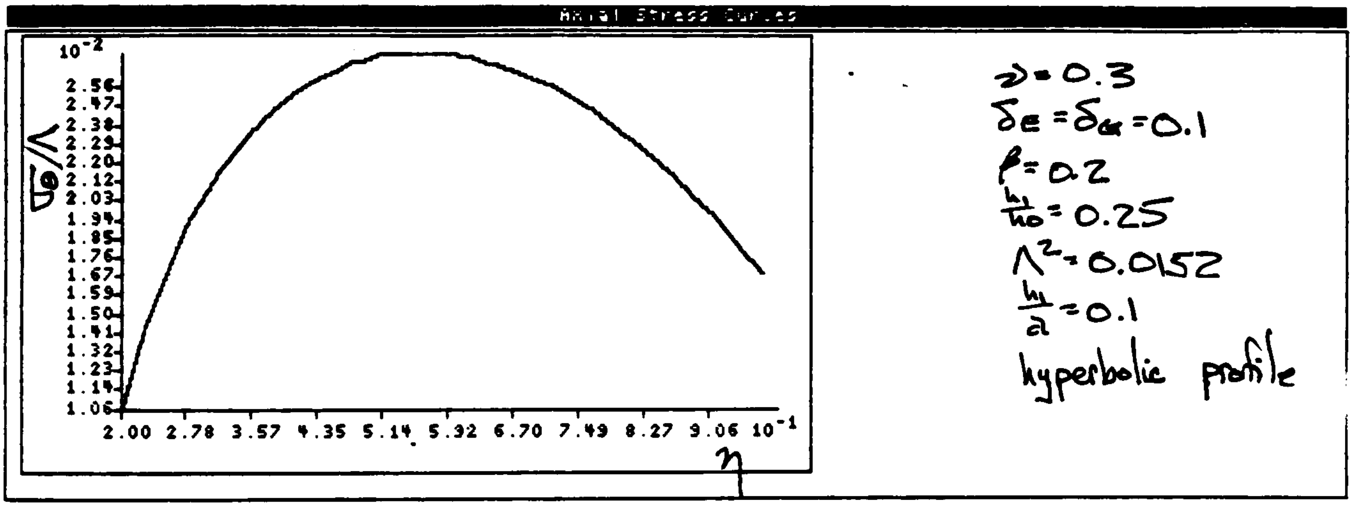
b. Circumferential stress distribution of a rotating disk.



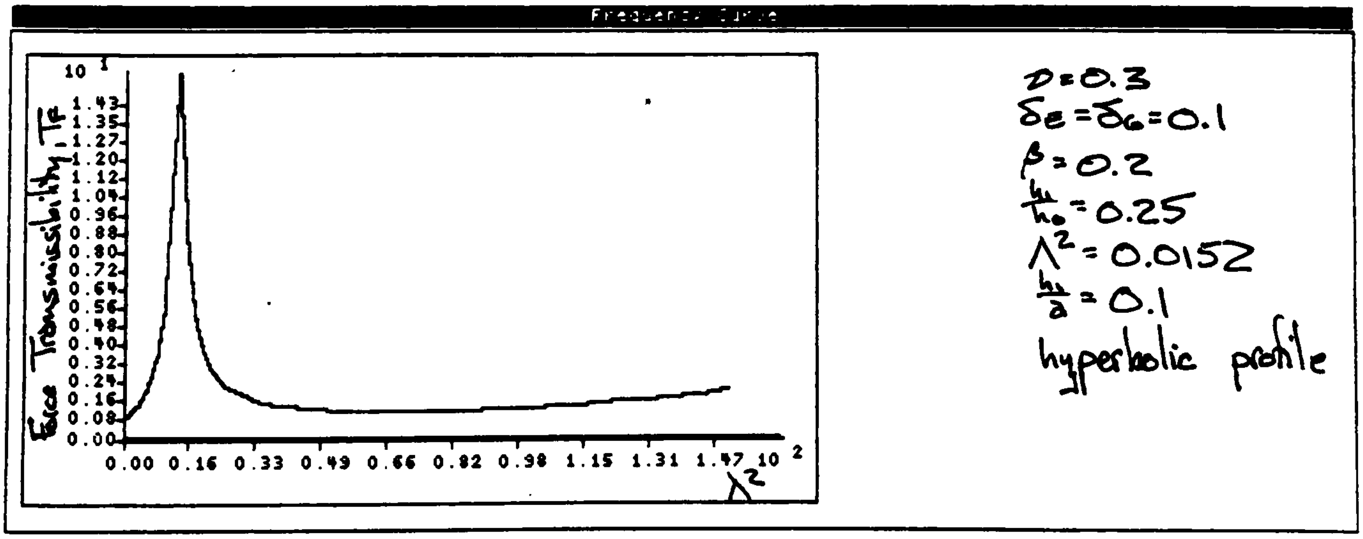
c. Steady-state response of a rotating disk.



a. Radial stress distribution of a rotating disk.



b. Circumferential stress distribution of a rotating disk.



c. Steady-state response of a rotating disk.

5. SUMMARY

The governing equations of steady-state stress and force transmissibility of a clamped-free rotating angular disk are derived. Deviations with Irie's paper are noted in Appendix III.

Effects resulting from varying selected disk parameters are analyzed with relatively good comparison with previously published results obtained.

Further investigation with other disk profiles, laminar disks, damped disks, and alternately loaded disks is recommended to optimize the disk configuration allowing for a disk design having minimum stress and force transmissibility profiles over a range of externally-applied loadings.
Extensive investigation of varying radius-radius ratio, radius-thickness ratios, and thickness-thickness ratio on radial stress, circumferential stress and the force transmissibility profile is also recommended.

References

1. Snowden, J. C., "Forced Vibration of Internally Damped Circular and Annular Plates with Clamped Boundaries," Journal of the Acoustical Society of America, Vol. 50, 1971, pp. 846-858.

2. Mote, C. D., "Free Vibration of Initially Stressed Circular Disks," Journal of Engineering for Industry, May 1965, pp. 258-264.

3. Kennedy,W., and Gorman, D., "Vibration Analysis of Variable Thickness Disks Subjected to Centrifugal and Thermal Stresses," Journal of Sound and Vibration, Vol. 53, 1977,pp. 83-101.

4. Pardoen, G., "Vibration and Buckling Analysisof Axisymmetric Polar Orthotropic Circular Plates," Computers and Structures, Vol. 4, 1973, pp. 951-960.

5. Ghosh, N. C., "The Vibration of Rotating Aelotropic Elastic Disk," Indian Journal of Physics, Vol. 45, 1971, pp. 262-267.

6. Irie, T., G. Yamada, and S. Aomura, "The Steady-State Response of a Rotating Damped Disk of Variable Thickness," Journal of Applied Mechanics, Vol. 47, 1980, pp. 896-900.

7. Timoshenko, S., Theory of Plates and Shells, 1959, 2nd ed., McGraw-Hill Book Company, pp. 51-52.

8. Kerlin, R. L. and Snowden, J. C., "Driving Point Impedances of Cantilever Beams, Comparison of Measurement and Theory," Journal of the Acoustical Society of America, Vol. 47,1970, pp. 220-228.

9. Burden, R. L., Numerical Analysis, 1981, 2nd ed., Prindle, Weber, & Schmidt, pp. 200-244.

10. Livesley, R. K., Matrix Methods of Structural Analysis," 1975, 2nd ed., Pergamon Press, pp. 165-187.

Appendix I. Notation

a	Disk inner edge radius
b	Disk outer edge radius
C_1	Translational viscous damping coefficient
C_2	Rotational viscous damping coefficient
D	Flexural rigidity
e	Naperian constant, 2.781828...
E	Young's modulus
F	Externally applied force
G	Shear modulus
h	Thickness
h_i	Outside thickness
h_o	Inside thickness
I	Imaginery component
j	Complex constant, $\sqrt{-1}$
K	Shear coefficient, $\pi^2/12$
M_r	Radial bending moment
M_θ	Circumferential bending moment
Q_r	Shearing force
r	Radius
R	Real component
t	Time
T_F	Force transmissibilty
u	Radial displacement
W	Transverse deflection
ρ	Mass per unit volume
Θ	Angular co-ordinate

ν	Poisson's ratio
σ_r	Radial stress
σ_θ	Circumferential stress
Ω	Disk angular velocity
ψ_r	Disk slope
δ_E	Young's constant imaginary-real ratio
δ_G	Shear modulus imaginary-real ratio
λ	Frequency
ϵ	Strain

Appendix II. Deviations from T. Irie's Paper

In separately deriving the governing equations, instances where the author's equation deviated from T. Irie's paper.

This section will detail these occurrences writing first the version of the equation published in T. Irie's paper and then follwed by the equation as derived by the author.

q_0, Dimensionless quantity

T. Irie's paper: $q_0 = \frac{1}{12}\left(\frac{h_0}{b}\right)^2$ (89)

Author's paper: $q_0 = \frac{1}{12}\left(\frac{h_0}{b}\right)$ (29)

Λ, Angular velocity

T. Irie's paper: $\Lambda = \sqrt{\frac{\rho h_0 b^4}{D_0}}\,\Omega$ (90)

Author's paper: $\Lambda = \sqrt{\frac{\rho h_0^2 b^3}{D_0}}\,\Omega$ (28)

λ, Frequency

T. Irie's paper: $\lambda = \sqrt{\frac{\rho h_0 b^4}{D_0}}\,\omega$ (91)

Author's paper: $\lambda = \sqrt{\frac{\rho h_0^2 b^3}{D_0}}\,\omega$ (50)

τ, Time constant

T. Irie's paper: $\tau = \sqrt{\frac{D_0}{\rho h_0 b^4}}\,t$ (92)

Author's paper: $\tau = \sqrt{\frac{D_0}{\rho h_0^2 b^3}}\,t$ (51)

U_{21}, Coefficient matrix element

T. Irie's paper: $$U_{21} = \frac{\sigma_r(1+j\delta_G)}{(1+j\delta_E)(1+j\delta_G+k_o\sigma_r)d^2} \tag{93}$$

Author's paper: $$U_{21} = \frac{\sigma_r(1+j\delta_G)}{12q_o(1+j\delta_E)(1+j\delta_G+k_o\sigma_r)d^2} \tag{58}$$

U_{22}, Coefficient matrix element

T. Irie's paper: $$U_{22} = -\frac{1}{1+j\delta_G+k_o\sigma_r}\left(k_o\frac{d\sigma_r}{d\eta}+k_o\frac{\sigma_r}{\eta}+\frac{1+j\delta_G}{\eta}\right) \tag{94}$$

Author's paper: $$U_{22} = -\frac{1}{1+j\delta_G+k_o\sigma_r}\left\{\frac{1+j\delta_G}{\eta}+\frac{k_o\sigma_r}{\eta}+\frac{dh}{d\eta}\frac{k_o\sigma_r}{h}+k_o\frac{d\sigma_r}{d\eta}\right\} \tag{59}$$

U_{23}, Coefficient matrix element

T. Irie's paper: $$U_{23} = \frac{(1+j\delta_G)d}{1+j\delta_G+k_o\sigma_r}\left\{\frac{d\sigma_r}{d\eta}+\left(\frac{1-\nu}{\eta}+\frac{dd}{d\eta}\frac{1}{d}\right)\sigma_r\right\} \tag{95}$$

Author's paper: $$U_{23} = \frac{(1+j\delta_G)d}{12q_o(1+j\delta_G+k_o\sigma_r)}\left\{\left(\frac{1-\nu}{\eta}+\frac{1}{d}\frac{dd}{d\eta}\right)\sigma_r+\frac{d\sigma_r}{d\eta}\right\} \tag{60}$$

Appendix III. Computer Program

The following is the listing of the Pascal computer program used to :

i) solve the radial stress and axial stress distribution of a rotating annular disk,

ii) solve the force transmissibilty-frequency relationship of the rotating disk, and

iii) plot the graph of force transmissibility versus frequency.

```
Program DiskVibration;

{$I typedef.sys}
{$I graphix.sys}
{$I kernel.sys}
{$I windows.sys}
{$I findwrld.hgh}
{$I axis.hgh}
{$I polygon.hgh}

Const
   Pi : Real = 3.1415926536E+00;
   Freqnum  = 50;
   Numstep  = 50;
           (* Must be positive,even number     *)
   Tol1 : Real = 1.0E-10;
             (* Tol for differ. between two      *)
             (*  successive Sigmar[0] values     *)
   Tol2 : Real = 1.0E-11;
             (* Tol for endpoint Sigmar[Numstep] *)
   Tol3 : Real = 1.0E-10;
             (* Tol for zero-checking parameters *)
   Kk : Real = 8.22467033E-01;
             (* Stiffness coefficient            *)
   Maxiter : Integer = 170;
             (* Max iter. for solving stress-    *)
             (*  raddisp equations               *)

Type
   Complex = Record re,im : Real
              End;
   Mat_U = Array[1..4,1..4] of Complex;
   Mat_T = Array[1..2,1..2] of Complex;
   Mat_Tri = Array[1..4,1..4] of Complex;
   Mat_Uri = Array[1..8,1..8] of Real;
   Mat_Arr = Array[1..8,1..4] of Real;
   Mat_Var = Array[1..2,1..1] of Complex;
   Mat_D = Array[0..Numstep] of Real;
   Mat_Q = Array[0..Numstep] of Real;
   Mat_Freq = Array[0..Freqnum] of Real;

Var
   Diskprofile : Integer;
   Radius,D : Mat_D;
   Lambda,W,Zeta1,Zeta2 : Real;
   C1,C2 : Real;
   K0 : Real;
   Rho : Real;
   A,B,H0,H1 : Real;
   E,Delte,G,Deltg,Nu,Mu : Real;
   Omega,Force : Real;
   Beta,Q0,D0,Pyr : Real;
```

```
    I,J,I2,J2,I3,J3,I4,J4,I5,J5,I6,J6,I7,J7 : Integer;
    Sigmar : Mat_D;
    Rstep : Real;
    K1A,K2A,K3A,K4A,K1B,K2B,K3B,K4B : Real;
    U,T : Mat_U;
    Tri : Mat_Tri;
    Coeff21,Coeff22,Coeff23 : Real;
    DSigDEta,Temp : Real;
    Raddisp : Mat_D;
    Uri : Mat_Uri;
    Tr : Array[1..4,1..4] of Real;
    Ti : Array[1..4,1..4] of Real;
    TriTemp : Mat_Uri;
    Karray,Qarray,Rarray,Yarray : Mat_Arr;
    Radi,Sig : Real;
    Mr,Qr : Complex;
    Chir,Wl : Complex;
    Iter : Integer;
    Dhj,Dh : Real;
    DiskRad,HH : Real;
    SStore,SNumstep1,SNumstep2 : Real;
    STemp : Real;
    DUDR,Sigmatheta : Mat_D;
    T1,T2,T3 : Mat_T;
    T4,T6,T5 : Mat_Var;
    Det : Complex;
    Imped,Transmis : Real;
    Integ : Real;
    Stoppgrm : Boolean;
    SigrPyr : Mat_Q;
    Eta,SigThetPyr : Mat_Q;
    Temp1,Temp2 : Real;
    Rdisp,HJ,HJ_1,Radj,Radj_1 : Real;
    Eta1,Rad_1 : Real;
    STempL,StempH : Real;
    U1,U2,U3,U4 : Real;
    DataType : Integer;
    Lamsqr : Mat_Freq;
    Trnsms : Mat_Freq;

Procedure MatMult (Var T3 : Mat_T; Var T4,T6 : Mat_Var);

(*     This procedure will multiply a complex
                 2x2 matrix (T3)    *)
(*     and a complex 2x1 matrix (T4) and
                 store the result in    *)
(*     a complex 2x1 matrix (T6).   *)

  Begin

    T6[1,1].re := T3[1,1].re*T4[1,1].re -
                  T3[1,1].im*T4[1,1].im +
                  T3[1,2].re*T4[2,1].re -
                  T3[1,2].im*T4[2,1].im;
```

```
     T6[1,1].im := T3[1,1].re*T4[1,1].im +
                   T3[1,1].im*T4[1,1].re +
                   T3[1,2].re*T4[2,1].im +
                   T3[1,2].im*T4[2,1].re;


     T6[2,1].re := T3[2,1].re*T4[1,1].re -
                   T3[2,1].im*T4[1,1].im +
                   T3[2,2].re*T4[2,1].re -
                   T3[2,2].im*T4[2,1].im;


     T6[2,1].im := T3[2,1].re*T4[1,1].im +
                   T3[2,1].im*T4[1,1].re +
                   T3[2,2].re*T4[2,1].im +
                   T3[2,2].im*T4[2,1].re;


  End;

Procedure TMatInv (Var T1,T3 : Mat_T;Stoppgrm : Boolean);

Var
   Determ : Real;

(*     This procedure will determine the inverse           *)
(*     of a complex 2x2 matrix.  The original matrix       *)
(*     is T1 and the inverse matrix is returned as         *)
(*     T3.                                                 *)
(*                                                         *)
(*     Det.re    =  Real part of determinant                *)
(*     Det.im    =  Imaginary part of determinant           *)
(*     Determ  =  Determinant of T1 matrix                 *)

Begin
   Det.re := T1[1,1].re*T1[2,2].re -
             T1[1,1].im*T1[2,2].im -
             T1[1,2].re*T1[2,1].re +
             T1[1,2].im*T1[2,1].im;


   Det.im := T1[1,1].re*T1[2,2].im +
             T1[1,1].im*T1[2,2].re -
             T1[1,2].re*T1[2,1].im -
             T1[1,2].im*T1[2,1].re;


  Determ := (sqr(Det.re) + sqr(Det.im));


  If Abs(Determ) < Tol3 then
     Begin
```

```
        Writeln('Matrix determinant = +/- ',sqrt(Determ));
        Writeln(    ' which is less than ',Tol3);

        Writeln('The matrix is considered to be singular');
        Stoppgrm := True;
     End;


  T3[1,1].re := (T1[2,2].re*Det.re-T1[2,2].im*(-
Det.im))/Determ;

  T3[1,1].im := (T1[2,2].im*Det.re+T1[2,2].re*(-
Det.im))/Determ;

  T3[1,2].re := ((-T1[1,2].re)*Det.re+T1[1,2].im*(-
Det.im))/Determ;

  T3[1,2].im := ((-T1[1,2].im)*Det.re-T1[1,2].re*(-
Det.im))/Determ;

  T3[2,1].re := ((-T1[2,1].re)*Det.re+
                    T1[2,1].im*(-Det.im))/Determ;

  T3[2,1].im := ((-T1[2,1].im)*Det.re-
                    T1[2,1].re*(Det.im))/Determ;

  T3[2,2].re := (T1[1,1].re*Det.re-
                    T1[1,1].im*(-Det.im))/Determ;

  T3[2,2].im := (T1[1,1].im*Det.re+
                    T1[1,1].re*(-Det.im))/Determ;


End;


Procedure Integrate (Var D : Mat_D; Var Eta : Mat_Q; Numstep
: Integer;Var     Rstep,Integ : Real);

Var
  Alt,Jin   : Integer;
  S : Real;

Begin

  S := D[0]*Eta[0] + D[Numstep]*Eta[Numstep];
  Alt := 4;

For Jin := 1 to Numstep-1 do

  Begin

     S := S + Alt*D[Jin]*Eta[Jin];
     Alt := 6 - Alt;
  End;
  Integ := Rstep*S/3.0;
End;

  Function H(DiskRad : Real;Diskprofile : Integer) : Real;
  Begin                                   (* H *)
     If Diskprofile = 1 then
        Begin
           H := H0*(1.-((1.-(H1/H0))*((DiskRad-A)/(B-A))));
        End;
     If Diskprofile = 2 then
        Begin
           H := H0*EXP(((DiskRad-A)/(B-A))*ln(H1/H0));
        End;
     If Diskprofile = 3 then
        Begin
           H := H0*Exp(- ((ln(H1/H0))/ln(Beta))*
                           (ln(DiskRad/A)));
        End;
     If Diskprofile < 1 then
        Begin
           Writeln('Error : Diskprofile = ',Diskprofile);
        End;
     If Diskprofile > 3 then
        Begin
           Writeln('Error : Diskprofile = ',Diskprofile);
        End;
  End;
```

```
     Function RKEq1(Rdisp,Radi,Sig : Real) : Real;

     Begin

        RKEq1 := -(Nu*Rdisp/(Radi/B)) + (Q0*Sig);

     End;                            (* Function RKEq1 *)

     Function RKEq2(Radi,Rad_1,Rdisp,Sig,HJ,HJ_1 : Real) :
Real;
     Label 33;
     Begin
        If Radi = Rad_1 then
           Begin
              RKEq2 := ( ((1.0-
                    sqr(Nu))*Rdisp)/(Q0*sqr(Radi/B)) ) -
                         ( ((1.0-Nu)*Sig)/(Radi/B)  ) -
                         ( (sqr(Pyr))*(Radi/B) );
              Goto 33;
           End;
        RKEq2 := ( ((1.0-sqr(Nu))*Rdisp)/
                 (Q0*sqr(Radi/B)) ) -
                 ( ((1.0-Nu)*Sig)/(Radi/B)  ) -
                 ( (B/HJ)*Sig*((HJ-HJ_1)/(Radi-Rad_1)) ) -
                 ( (sqr(Pyr))*(Radi/B) );

 33 :  Begin  End;

    End;                  (*  Function   RKEq2   *)

  Procedure  Data1;
    Begin

         (*   Input Geomdata;  *)

     A := 4.000;                         (* Inner Radius *)
     B := 20.0000;                        (* Outer Radius *)
     H0 := 0.04;                    (* Thickness at radius A *)
     H1 := 0.02;                    (* Thickness at radius B *)
     Diskprofile := 3;                 (* Diskprofile =
                                                1, linear

                                            2, exponential

                                           3, hyperbolic *)
         (*   Input Matldata; *)

     E := 30.0000E+10;                 (* Young's Modulus--Real
                                                    Part *)
    Delte := 0.010000;                (* Ratio Imag:Real part
                                         of
                                          Young's Modulus at
                                          any frequency *)
    G := 10.0000E+10;                 (* Shear modulus--real
                                          part *)
    Deltg := 0.010000;                (* Ratio Imag:Real part
                                               of
                                          Shear Modulus at
                                         any frequency *)
    Rho := 0.3333333;                 (* Material mass
                                         density*)
    Nu := 0.300;                      (* Poisson's ratio *)

         (*   Input Systemdata; *)

    Omega := 1000.00;                    (* Disk
                            rotational speed *)
    Force := 200.0;                    (* Transverse force
                                         applied > 0.0 *)

    C1 := 0.0;
    C2 := 0.0;

  End;


 Function Eq(Var Uri : Mat_Uri; Var Yarray : Mat_Arr;
                                 Var I5,J5 : Integer) : Real;

Var
   Iin : Integer;
   TEq : Real;

     Begin

     TEq := 0.0;

     For Iin := 1 to 8 do

        Begin

        TEq :=  TEq + Uri[I5,Iin]*Yarray[Iin,J5];

        End;

        Eq := TEq;

     End;                                 (* Eq *)
```

```
Procedure Plotfreq;

Var
   Dx,Dy,iq,m,lines,scale : integer;
   X1,Y1,X2,Y2 : integer;
   aa,ab,ac : Plotarray;
   Temp : real;

Begin
   DefineWindow(1,0,0,XMaxGlb,YMaxGlb);
   DefineWindow(2,trunc(XMaxGlb/40),trunc(YMaxGlb/10),
                  trunc(XMaxGlb*6/10),
                  trunc(YMaxGlb*19/20));
   DefineWorld(1,0,4000,4000,0);

   DefineHeader(1,'Frequency Curve');
   SetHeaderOn;
   DrawBorder;

   (*  Fill data arrays  *)

   For Iq := 0 to Freqnum-1 do
   Begin

      aa[iq+1,1] := Lamsqr[iq];
      aa[iq+1,2] := Trnsms[iq];

   End;

   FindWorld(2,aa,Freqnum,1,1);

   with World[2] do
   Begin
      Temp := Y1;
      Y1 := Y2;
      Y2 := Temp;
   End;

   SelectWorld(2);
   SelectWindow(2);
   DrawBorder;

   dx := 9;
   dy := 9;
   X1 := 2;
   Y1 := 0;
   X2 := 0;
   Y2 := 10;
   lines := 0;
   scale := 0;

   SetLineStyle(0);
   DrawAxis(dx,dy,x1,Y1,x2,Y2,lines,scale,false);
   DrawPolygon(aa,1,-(Freqnum-2),0,1,0);
```

```
       ResetAxis;

      SelectWorld(1);
      SelectWindow(1);

   End;



(*   Procedure Stressolve;  *)
     Label 5;
     Label 10;
     Label 20;
     Label 30;
     Label 25;
     Label 35;
     Label 39;
     Label 45;
     Label 55;
     Label 330;

     Begin

     Stoppgrm := False;

     Datal;
     DataType := 1;

 55:  Begin  End;

     Writeln(Lst,' Pi = ',Pi,' Numstep = ',Numstep);
     Writeln(Lst,' Toll = ',Toll,' Kk = ',Kk);
     Writeln(Lst,' A = ',A,' B = ',B);
     Writeln(Lst,' H0 = ',H0,' H1 = ',H1);
     Writeln(Lst,' Diskprofile = ',Diskprofile);
     Writeln(Lst,' E = ',E,' Delte = ',Delte);
     Writeln(Lst,' G = ',G,' Deltg = ',Deltg);
     Writeln(Lst,' Rho = ',Rho,' Nu = ',Nu);
     Writeln(Lst,' Omega = ',Omega,' Force = ',Force);

(*   Procedure Dimensionless; *)

     Beta := A/B;
     Q0 := (H0/B)/12.0;
     D0 := (E*H0*H0*H0)/(12.0*(1.0-sqr(Nu)));
     Pyr := Sqrt((Rho*H0*H0*B*B*B)/D0)*Omega;

     Writeln(Lst,' Beta = ',Beta,' Q0 = ',Q0);
     Writeln(Lst,' D0 = ',D0,' Pyr = ',Pyr);

     Rstep := (B-A)/Numstep;            (* Stepsize *)

     Writeln(Lst,' Rstep = ',Rstep);
```

```
  For I := 0 to Numstep do
     Begin

        Raddisp[I] := 0.0;

        Radius[I] := A + I*Rstep;

        Eta[I] := Radius[I]/B;

        If Abs(Eta[I]) < Tol3 then

           Begin

              Writeln(' Eta[ ',I,'] is less than ',Tol3);
              Writeln(' The algorithm will not divide by
              Writeln('   this small a number.');
              Goto 20;
           End;

        D[I] := H(Radius[I],Diskprofile)/H0;

        If Abs(D[I]) < Tol3 then

           Begin

              Writeln(' D[ ',I,'] is less than ',Tol3);
              Writeln(' The algorithm will not divide by ');
              Writeln('    this small a number.');
              Goto 20;

           End;

        Writeln(' Radius= ',Radius[I],
                    ' U= ',Raddisp[I],' H= ',
                     H(Radius[I],Diskprofile));
     End;

     Iter := 0;

     Sigmar[0] := 1.0;

     STempL := 0.0;

     STempH := Sigmar[0];

                 (*   Start Runge-Kutta Procedure     *)

5:  Begin    End;
```

```
For J := 1 to Numstep do
Begin

      (* Get four estimates of deltas *)


      K1A := Rstep*(RKEq1(Raddisp[J-1],
          Radius[J-1],Sigmar[J-1]));
      U1 := Raddisp[J-1] + K1A;

      K2A := Rstep*(RKEq1(Raddisp[J-1]+K1A/2.0,
            Radius[J-1]+Rstep/2.0,
            Sigmar[J-1]));
      U2 := Raddisp[J-1] + K2A;

      K3A := Rstep*(RKEq1(Raddisp[J-1]+K2A/2.0,
            Radius[J-1]+Rstep/2.0,
            Sigmar[J-1]));
       U3 := Raddisp[J-1] + K3A;

      K4A := Rstep*(RKEq1(Raddisp[J-1]+K3A,
            Radius[J-1]+Rstep,
            Sigmar[J-1]));
      U4 := Raddisp[J-1] + K4A;


   Temp1 := (K1A + 2.0*K2A + 2.0*K3A + K4A)/6.0;

   Raddisp[J] := Raddisp[J-1] + Temp1;



      Dh := H(Radius[J-1],Diskprofile)/H0;

      K1B := Rstep*(RKEq2(Radius[J-1],Radius[J-1],U1,
                Sigmar[J-1],H0*Dh,H0*D[J-1]));

      Dh := H(Radius[J-1]+Rstep/2.0,Diskprofile)/H0;


      K2B := Rstep*(RKEq2(Radius[J-1]+
            Rstep/2.0,Radius[J-1],U2,
                Sigmar[J-1]+K1B/2.0,
                 H0*Dh,H0*D[J-1]));

      Dh := H(Radius[J-1]+Rstep/2.0,Diskprofile)/H0;


      K3B := Rstep*(RKEq2(Radius[J-1]+
                 Rstep/2.0,Radius[J-1],U3,
                Sigmar[J-1]+K2B/2.0,
                 H0*Dh,H0*D[J-1]));

      Dh := H(Radius[J-1]+Rstep,Diskprofile)/H0;
      K4B := Rstep*(RKEq2(Radius[J-1]+
                    Rstep,Radius[J-1],U4,
                   Sigmar[J-1]+K3B,H0*Dh,H0*D[J-1]));


      (* Compute the x at the end of
                    the interval from
         a weighted average
                    of the four estimates.     *)


   Temp2 := (K1B + 2.0*K2B + 2.0*K3B + K4B)/6.0;

   Sigmar[J] := Sigmar[J-1] + Temp2;


   End;
         (*      End Runge-Kutta Sequence    *)


   SNumstep1 := Sigmar[Numstep];

   If (SNumstep1 < Tol2) and (SNumstep1 > 0.0)  then

      Begin

         Goto 25;

      End;

   If SNumstep1 < 0.0 then

      Begin

         STempL := Sigmar[0];
         Sigmar[0] := 2.0*Abs(Sigmar[0]);
         STempH := Sigmar[0];
         Goto 5;

      End;

10:  Begin      End;

   Sigmar[0] := STempH - (STempH - STempL)/2.0;

       (*   Start Runge-Kutta Procedure     *)

35:  Begin   End;
```

```
For J := 1 to Numstep do

Begin


   (* Get four estimates of deltas *)



   K1A := Rstep*(RKEq1(Raddisp[J-1],
            Radius[J-1],Sigmar[J-1]));
   U1 := Raddisp[J-1] + K1A;

   K2A := Rstep*(RKEq1(Raddisp[J-1]
            +K1A/2.0,Radius[J-1]+Rstep/2.0,
                         Sigmar[J-1]));
   U2 := Raddisp[J-1] + K2A;

   K3A := Rstep*(RKEq1(Raddisp[J-1]
            +K2A/2.0,Radius[J-1]+
            Rstep/2.0,Sigmar[J-1]));
   U3 := Raddisp[J-1] + K3A;

   K4A := Rstep*(RKEq1(Raddisp[J-1]+K3A,
            Radius[J-1]+Rstep,
            Sigmar[J-1]));
   U4 := Raddisp[J-1] + K4A;


Templ := (K1A + 2.0*K2A + 2.0*K3A + K4A)/6.0;

Raddisp[J] := Raddisp[J-1] + Templ;


   Dh := H(Radius[J-1],Diskprofile)/H0;

   K1B := Rstep*(RKEq2(Radius[J-1],Radius[J-1],U1,
                Sigmar[J-1],H0*Dh,H0*D[J-1]));

   Dh := H(Radius[J-1]+Rstep/2.0,Diskprofile)/H0;


   K2B := Rstep*(RKEq2(Radius[J-1]+
            Rstep/2.0,Radius[J-1],U2,
            Sigmar[J-1]+K1B/2.0,H0*Dh,H0*D[J-1]));

   Dh := H(Radius[J-1]+Rstep/2.0,Diskprofile)/H0;
```

```
      K3B := Rstep*(RKEq2(Radius[J-1]+
                Rstep/2.0,Radius[J-1],U3,
                Sigmar[J-1]+K2B/2.0,H0*Dh,H0*D[J-1]));

      Dh := H(Radius[J-1]+Rstep,Diskprofile)/H0;


      K4B := Rstep*(RKEq2(Radius[J-1]+
                Rstep,Radius[J-1],U4,
                    Sigmar[J-1]+K3B,H0*Dh,H0*D[J-1]));

  (* Compute the x at the end of the interval from
     a weighted average of the four estimates.     *)


   Temp2 := (K1B + 2.0*K2B + 2.0*K3B + K4B)/6.0;

   Sigmar[J] := Sigmar[J-1] + Temp2;


   End;
         (*      End Runge-Kutta Sequence   *)

   SNumstep2 := Sigmar[Numstep];

   If (SNumstep2 < Tol2) and (SNumstep2 > 0.0) then
      Begin

         Goto 25;

      End;

   If (Sigmar[0] - STempL)/2.0 < Tol1 then

      Begin

         Goto 25;

      End;

   Iter := Iter + 1;

If Iter > Maxiter then

Begin

   Writeln( 'Zero stress not found within ',
                Iter,' iterations');
   Goto 20;

End;
```

```
       If SNumstep1*SNumstep2 > 0.0 then

       Begin
          STempH :=Sigmar[0];
          SNumstep1 := SNumstep2;
          Goto 10;

       End;

       STempL := Sigmar[0];
       Sigmar[0] := Sigmar[0] + (STempH - STempL)/2.0;
       Goto 35;


25:    For I := 0 to Numstep do
       Begin

          DUDR[I] := -(Nu*Raddisp[I]/Radius[I]) +
                         (Q0*Sigmar[I]/B);

       End;

      For I := 0 to Numstep do
         Begin

            Sigmatheta[I] := ((12.0*B*B/H0)*((Nu*DUDR[I]) +
                                  (Raddisp[I]/Radius[I])));

        End;


      For I := 0 to Numstep do

         Begin

            SigrPyr[I] := Sigmar[I]/Pyr;
            SigThetPyr[I] := Sigmatheta[I]/Pyr;

         End;

 (* Procedure Vibsolve; *)

 (* Procedure Umatrix; *)


    Integrate(D,Eta,Numstep,Rstep,Integ);


    T4[1,1].re := 0.0;
    T4[1,1].im := 0.0;
    T4[2,1].re := 0.0;
    T4[2,1].im := 0.0;
```

```
Uri[1,5] := 0.0;
Uri[5,1] := 0.0;

Uri[1,2] := 1.0;
Uri[5,6] := 1.0;

Uri[1,6] := 0.0;
Uri[5,2] := 0.0;

Uri[1,4] := 0.0;
Uri[5,8] := 0.0;

Uri[1,8] := 0.0;
Uri[5,4] := 0.0;

Uri[3,2] := 0.0;
Uri[7,6] := 0.0;

Uri[3,6] := 0.0;
Uri[7,2] := 0.0;

Uri[3,7] := 0.0;
Uri[7,3] := 0.0;

Uri[3,4] := 0.0;
Uri[7,8] := 0.0;

Uri[3,8] := 0.0;
Uri[7,4] := 0.0;

Uri[4,1] := 0.0;
Uri[8,5] := 0.0;

Uri[4,5] := 0.0;
Uri[8,1] := 0.0;

Uri[4,3] := -1.0;
Uri[8,7] := -1.0;

Uri[4,7] := 0.0;
Uri[8,3] := 0.0;

Uri[4,4] := 0.0;
Uri[8,8] := 0.0;

Uri[4,8] := 0.0;
Uri[8,4] := 0.0;

                          (* Umatrix *)
```

```
   For I := 1 to Freqnum do

Begin

   W := 1500.0*I;

   Lambda := sqrt(Rho*H0*sqr(B*B)/D0)*W;

   Zeta1 := sqr(B)*C1/(2.0*sqrt(Rho*H0*D0));

   Zeta2 := C2/(2.0*sqrt(Rho*H0*D0));

   K0 := 2.0*Q0/(Kk*(1.0-Nu));

   For I4 := 1 to 8 do

   For J4 := 1 to 4 do

   Begin

      Yarray[I4,J4] := 0.0;
      Qarray[I4,J4] := 0.0;

   End;


   Yarray[1,1] := 1.0;

   Yarray[2,2] := 1.0;

   Yarray[3,3] := 1.0;

   Yarray[4,4] := 1.0;

   For J := 0 to Numstep do

   Begin

      If J = 0 then
         Begin

       DSigDEta := ( ((1.0-sqr(Nu))*
                        Raddisp[J])/(Q0*
                        sqr(Radius[J]/B)) ) -
                         ( ((1.0-Nu)*Sigmar[J])/
                        (Radius[J]/B)   ) -
                         ( (sqr(Pyr))*(Radius[J]/B) );
            Goto 330;

         End;
         DSigDEta := ( ((1.0-sqr(Nu))*
                         Raddisp[J])/(Q0*
                         sqr(Radius[J]/B)) ) -
                       ( ((1.0-Nu)*Sigmar[J])
                         /(Radius[J]/B)  ) -
                          ( (1.0/D[J])*Sigmar[J]*
                         ((D[J]-D[J-1])/
                         (Eta[J]-Eta[J-1])) ) -
                       ( (sqr(Pyr))*(Radius[J]/B) );


330:  Begin  End;

      Uri[1,1] := -(1.0 - Nu)/Eta[J];
      Uri[5,5] := Uri[1,1];

     Uri[1,3] := ( (D[J]*sqr(D[J])*
                    (1.0-sqr(Nu))/sqr(Eta[J])) ) -
                    ( (Q0*sqr(Lambda)*D[J]*sqr(D[J])) );

     Uri[5,7] := Uri[1,3];

     Uri[1,7] := ( D[J]*sqr(D[J])*
                    (1.0-sqr(Nu))*Delte/sqr(Eta[J]) ) -
                       ( 2.0*Zeta2*Lambda );

     Uri[5,3] := -Uri[1,7];


     Coeff21 := (Sigmar[J])/((1.0+sqr(Delte))*
                    (sqr(1.0+K0*Sigmar[J]) +
                    sqr(Deltg))*sqr(D[J]));


     Uri[2,1] := Coeff21*(1.0 + K0*Sigmar[J] +
                     sqr(Deltg) +
                      Deltg*Delte*K0*Sigmar[J]);

     Uri[6,5] := Uri[2,1];


     Uri[2,5] := Coeff21*( -Delte -
                    Delte*K0*Sigmar[J] +
                    Deltg*K0*Sigmar[J] - sqr(Deltg)*Delte);

     Uri[6,1] := -Uri[2,5];

     Coeff22 := -1.0/(sqr(1.0+K0*Sigmar[J])+sqr(Deltg));

     Uri[2,2] := Coeff22*( ((1.0 +
                    K0*Sigmar[J])*( (K0*DSigDEta) +
                    (K0*Sigmar[J]/Eta[J]) +
                    (1.0/Eta[J]) )) +
                    (sqr(Deltg)/(Eta[J])) );
```

```
Uri[6,6] := Uri[2,2];

Uri[2,6] := Coeff22*( (Deltg*K0*
             ( DSigDeta - (Sigmar[J]/Eta[J]))) +
              (Deltg*(1.0+K0*Sigmar[J])/Eta[J])
              - (Deltg/Eta[J]) );

Uri[6,2] := -Uri[2,6];

Coeff23 := D[J]*(-Coeff22)*( DSigDEta +
             ((1.0-Nu)*Sigmar[J]/Eta[J]) +
             ((D[J]-D[J-1])/(Eta[J]-
             Eta[J-1]))*Sigmar[J]/D[J]);


Uri[2,3] := Coeff23*(1.0+K0*Sigmar[J]+sqr(Deltg));

Uri[6,7] := Uri[2,3];

Uri[2,7] := Coeff23*K0*Sigmar[J]*Deltg;

Uri[6,3] := -Uri[2,7];

Uri[2,4] := (-Coeff22)*( (2.0*Lambda*
                 Zetal*Deltg) +
                (2.0*Deltg*Lambda*
                 Zetal*K0*Sigmar[J]) -
                 (D[J]*sqr(Lambda)) -
                 (D[J]*K0*Sigmar[J]*
                 sqr(Lambda)) -
                 (2.0*Lambda*Zetal*Deltg) -
                 (D[J]*sqr(Lambda*Deltg))   );

Uri[6,8] := Uri[2,4];

Uri[2,8] := (-Coeff22)*( (D[J]*Deltg*sqr(Lambda)) -
                      (2.0*Lambda*Zetal) -
                  (2.0*Zetal*Lambda*K0*Sigmar[J]) -
                      (Deltg*D[J]*sqr(Lambda)) -
                   (Deltg*D[J]*K0*Sigmar[J]*
                     sqr(Lambda)) -
                  (2.0*Lambda*Zetal*sqr(Deltg)) );

Uri[6,4] := -Uri[2,8];

Uri[3,1] := 1.0/(D[J]*sqr(D[J])*(1.0+sqr(Delte)));

Uri[7,5] := Uri[3,1];

Uri[3,5] := -Uri[3,1]*Delte;

Uri[7,1] := -Uri[3,5];
```

```
Uri[3,3] := -Nu/Eta[J];

Uri[7,7] := Uri[3,3];

Uri[4,2] := K0/(D[J]*(1.0+sqr(Deltg)));

Uri[8,6] := Uri[4,2];

Uri[4,6] := -Deltg*Uri[4,2];

Uri[8,2] := -Uri[4,6];

                            (* Umatrix *)

For I5 := 1 to 8 do

For J5 := 1 to 4 do

Begin

   Karray[I5,J5] := Rstep*Eq(Uri,Yarray,I5,J5);

   Rarray[I5,J5] := 0.5*Karray[I5,J5] - Qarray[I5,J5];

   Yarray[I5,J5] := Yarray[I5,J5] + Rarray[I5,J5];

   Qarray[I5,J5] := Qarray[I5,J5] + 3.0*Rarray[I5,J5] -

                     0.5*Karray[I5,J5];

   Karray[I5,J5] := Rstep*Eq(Uri,Yarray,I5,J5);

   Rarray[I5,J5] := (1.0 - sqrt(0.5))*
                    (Karray[I5,J5] - Qarray[I5,J5]);

   Yarray[I5,J5] := Yarray[I5,J5] + Rarray[I5,J5];

   Qarray[I5,J5] := Qarray[I5,J5] + 3.0*Rarray[I5,J5]-

                     (1.0 - sqrt(0.5))*Karray[I5,J5];

   Karray[I5,J5] := Rstep*Eq(Uri,Yarray,I5,J5);

   Rarray[I5,J5] := (1.0 + sqrt(0.5))*

                    (Karray[I5,J5] - Qarray[I5,J5]);

   Yarray[I5,J5] := Yarray[I5,J5] + Rarray[I5,J5];

   Qarray[I5,J5] := Qarray[I5,J5] + 3.0*Rarray[I5,J5] -

                     (1.0 + sqrt(0.5))*Karray[I5,J5];

   Karray[I5,J5] := Rstep*Eq(Uri,Yarray,I5,J5);
```

```
         Rarray[I5,J5] := (1.0/6.0)*
                     (Karray[I5,J5] - 2.0*Qarray[I5,J5]);
         Yarray[I5,J5] := Yarray[I5,J5] + Rarray[I5,J5];
         Qarray[I5,J5] := Qarray[I5,J5] + 3.0*Rarray[I5,J5] -
                                0.5*Karray[I5,J5];
         End;
      End;
(*    Procedure Mrsolve    *)
      For I6 := 1 to 4 do
      For J6 := 1 to 4 do
      Begin
         Tri[I6,J6].re := Yarray[I6,J6];
         Tri[I6,J6].im := Yarray[I6+4,J6];
      End;
      For I7 := 1 to 2 do
     For J7 := 1 to 2 do
      Begin
         T1[I7,J7].re := Tri[I7,J7].re;
         T1[I7,J7].im := Tri[I7,J7].im;
         T2[I7,J7].re := Tri[I7+2,J7].re;
         T2[I7,J7].im := Tri[I7+2,J7].im;
      End;
(*    Procedure Tinvert    *)
      TMatInv(T1,T3,Stoppgrm);
      If Stoppgrm = True then
         Begin
            Goto 20;
```

```
   End;

T4[2,1].re := Force;

MatMult(T3,T4,T6);

Matmult(T2,T6,T5);

Mr.re := T6[1,1].re;

Writeln(' Mr.re=',Mr.re);

Mr.im := T6[1,1].im;

Writeln(' Mr.im=',Mr.im);

Qr.re := T6[2,1].re;

Writeln(' Qr.re=',Qr.re);

Qr.im := T6[2,1].im;

Writeln(' Qr.im=',Qr.im);

Chir.re := T5[1,1].re;

Writeln(' Chir.re=',Chir.re);

Chir.im := T5[1,1].im;

Writeln(' Chir.im=',Chir.im);

W1.re := T5[2,1].re;

Writeln(' W1.re=',W1.re);

W1.im := T5[2,1].im;

Writeln(' W1.im=',W1.im);

Imped := Force/((sqr(Lambda))*
               (sqrt(sqr(T5[2,1].re) +
               sqr(T5[2,1].im)))*(Integ));

Writeln(' Impedance = ',Imped);

Lamsqr[I] := sqr(Lambda);
Trnsms[I] := Abs((Beta)*(sqrt(sqr(Qr.re) +
                    sqr(Qr.im)))/Force);

Writeln(' Force Transmissibility = ',Trnsms[I]);

End;
```

```
        Initgraphic;
        ClearScreen;
        Plotfreq;
        Hardcopy(False,6);
        repeat until Keypressed;
        Leavegraphic;

20:  End.
```

Appendix IV. Summary of Runge-Kutta-Gill Method

The Runge-Kutta-Gill method is a numerical integration technique where use of previously-determined function values is not required in intermediate calculations. Thus, to arrive at a value y_n knowledge of y_{n-1}, y_{n-2}, ... is not necessary.

References to mathematical processes which are of this type are provided in [9]. A common for starting an integration is the Runge-Kutta (fourth-order) process. The error in each step of this process is of the order h^5, where h is the length of each interval.

Runge-Kutta's fourth-order process is based on the following general theory:
Consider a first-order differential equation

$$\frac{dy}{dx} = f(x,y) \tag{96}$$

with the initial condition

$$y = Y \text{ at } x = X \tag{97}$$

To obtain the value of y corresponding to the value $x = X + h$, the latter x-value is substituted in Equation (96) to obtain the value of dy/dx at the beginning of the interval. This slope value is used to determine the first approximation to the y-value at $x = X + h$. This new co-ordinate value is expressed as $(X+h, Y+k_0)$, where

$$k_0 = hf(x,y) \tag{98}$$

Advancing a fraction m of the interval h from X and substituting this new x-co-ordinate in Equation (96) results in the second approximation to the desired co-ordinate, namely

$$(X + mh,\ Y + k_1) \tag{99}$$

where

$$k_1 = hf(X + mh, Y + mk_0) \tag{100}$$

Combining the estimates k_0 and k_1 provides a third estimate of co-ordinate,

$$(X + nh,\ Y + k_2) \tag{101}$$

where

$$k_2 = [n-r]k_0 + rk_1 \tag{102}$$

This process is repeated with k_0, k_1, and k_2 to yield a fourth co-ordinate

$$(X + ph,\ Y + k_3) \tag{103}$$

where

$$k_3 = [p-s-t]k_0 + sk_1 + tk_2 \tag{104}$$

The incremental y-value corresponding to the interval h added to the x co-ordinate is calculated using the following expression:

$$y = y(X+h) - y(X) = ak_0 + bk_1 + ck_2 + dk_3 \tag{105}$$

where

$$a + b + c + d = 1. \tag{106}$$

By appropriately selecting the coefficients for a, b, c, and d, the resulting accuracy in terms of h5 may be adjusted.

By extending this technique to systems of equations, and after optimizing the coefficients used to calculate the final y-value thereby increasing accuracy, the following mathematical iterative process is used to determine successive y-values for the i-th equation:

$$k_{i0} = hf_i(y_{00}, y_{10}, \ldots) \tag{107}$$

$$r_{i1} = .5k_{i0} - q_{i0} \tag{108}$$

$$y_{i1} = y_{i0} + r_{i1} \tag{109}$$

$$q_{i1} = q_{10} + 3r_{i1} - .5k_{i0} \tag{110}$$

$$k_{i1} = hf_i(y_{01}, y_{11}, \ldots) \tag{111}$$

$$r_{i2} = [1 - \sqrt{.5}](k_{i1} - q_{i1}) \tag{112}$$

$$y_{i2} = y_{i1} + r_{i2} \tag{113}$$

$$q_{i2} = q_{i1} + 3r_{i2} - [1-\sqrt{.5}]k_{i1} \tag{114}$$

$$k_{i2} = hf_i(y_{02}, y_{12}, \ldots) \tag{115}$$

$$r_{i3} = [1 + \sqrt{.5}](k_{i2} - q_{i2}) \tag{116}$$

$$y_{i3} = y_{i2} + r_{i3} \tag{117}$$

$$q_{i3} = q_{i2} + 3r_{i3} - [1 + \sqrt{.5}]k_{i2} \tag{118}$$

$$k_{i3} = hf_i(y_{03}, y_{13}, \ldots) \tag{119}$$

$$r_{i4} = \frac{1}{6}(k_{i3} - 2q_{i3}) \tag{120}$$

$$y_{i4} = y_{i3} + r_{i4} \tag{121}$$

$$q_{i4} = q_{i3} + 3r_{i4} - .5k_{i3} \tag{122}$$

The last quantity q_{i4} is introduced to retain accuracy and becomes q_{i0} in the following iteration.

Appendix V. Summary of Transfer Matrix Method

The transfer matrix method [10] is an approach that "transfers" the behavior parameters across a joint (a point transfer matrix) or from one end of a system to the other (global transfer matrix). The global transfer matrix analysis is an extension of point transfer matrix analysis.

Use of the transfer matrix method requires that relationships that give the parametric state at one end of the element in terms of the parametric state at the opposite end.

Consider the element shown in Figure 14, whose endpoints are designated as i and i+1. The state of force and displacement at an endpoint is expressed by the "state" vector

$$\begin{Bmatrix} F \\ \Delta \end{Bmatrix}_i = \begin{Bmatrix} F_y \\ M_z \\ v_y \\ \theta_z \end{Bmatrix}_i . \qquad (123)$$

The expression relating the state vector at i+1 to the state vector at i is given by

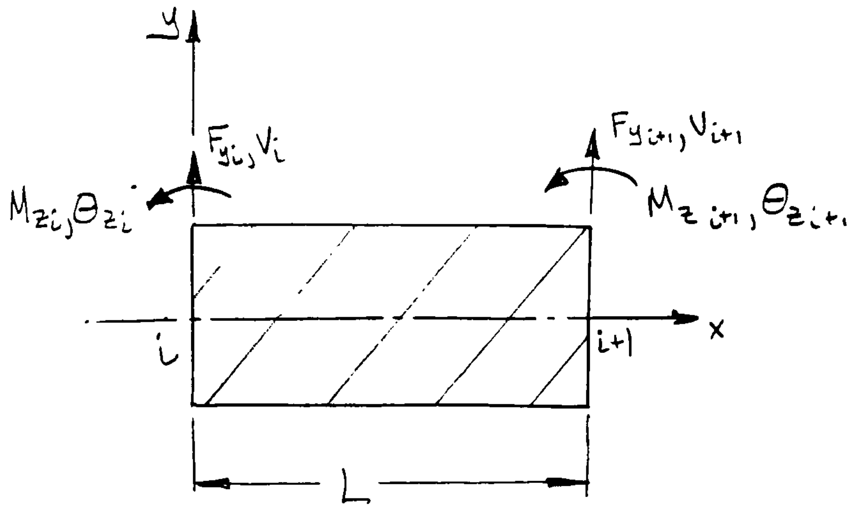
$$\begin{Bmatrix} F_{i+1} \\ \Delta_{i+1} \end{Bmatrix} = \left[\Omega \right] \begin{Bmatrix} F_i \\ \Delta_i \end{Bmatrix} \qquad (124)$$

In this instance, the transfer matrix $[\Omega]$ is a mixed form of the force-displacement relationships for the element.

For this example, the state of the force and displacement at location i+1 can be determined assuming that the initial state (that is, the force and displacement conditions at location i) is known by soving the following transfer matrix equation

$$\begin{Bmatrix} F_y & i+1 \\ M_z & i+1 \\ V & i+1 \\ \Theta_z & i+1 \end{Bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ L & -1 & 0 & 0 \\ L^3/6EI & -L^2/ & 1 & 0 \\ L^2/2EI & -L/EI & 0 & 1 \end{bmatrix} \begin{Bmatrix} F_y & i \\ M_z & i \\ V & i \\ \Theta_z & i \end{Bmatrix} \qquad (125)$$

Successive use of the point transfer matrix method by starting at one system endpoint and continuing to the other system endpoint results in the global transfer matrix.


y
F_{y_i}, V_i
M_{z_i}, Θ_{z_i}
$F_{y_{i+1}}, V_{i+1}$
$M_{z_{i+1}}, \Theta_{z_{i+1}}$
i
i+1
x
L