Flutter Evaluation and Control of an Airfoil Solved in the Laplace Domain

Alfred Jeffrey

Mechanical Engineering, McGill University, Montreal

1 May, 1991

A Thesis Submitted to the Faculty of Graduate Studies and Research in partial fulfilment of the requirements for the degree of Master's in Engineering

Abstract

Aeroelasticity is concerned with the interaction of aerodynamic forces and the resulting structural deformations for a structure in an airflow. Flutter, which is one example of an aeroelastic phenomena, may result in the catastrophic failure of an aircraft. The traditional methods of predicting the flutter velocity, namely the $U - g$ and $p - k$, have been used with great success. However, in recent years, new methods have been suggested which approximate the unsteady aerodynamic forces and moments by rational functions. The resulting equations are then solved in the Laplace domain. The goals of this work are twofold. Firstly, to solve the equations of motion for an airfoil, in both incompressible and transonic flow, in the Laplace domain using rational functions to approximate the unsteady aerodynamics. Secondly, to implement active control for the airfoil in incompressible flow, with the aim of increasing the critical flutter speed. The solution of the aeroelastic equations of motion in the Laplace domain proved to be a powerful tool in the analysis of flutter.

Sommaire

L'aéroélasticité traite de l'interaction des forces aérodynamiques et des déformations structurelles résultantes d'un corps dans un écoulement d'air. Le flottement, qui est un exemple de phénomène aéroélastique, peut provoquer d'importants dommages à un avion. Les méthodes traditionnelles, $U-g$ et $p-k$, qui sont capables de prédire la vitesse critique de flottement, ont été utilisées avec grand succès. Cependant, récemment, de nouvelles méthodes ont vu le jour, estimant les forces et moments instationnaires par des fonctions rationnelles. Les équations résultantes sont résolues dans le domaine de Laplace. Cette thèse a deux objectifs. Tout d'abord, résoudre les équations du mouvement pour un profil en écoulement incompressible et en écoulement transsonique dans le domaine de Laplace, en utilisant ces fonctions rationnelles. Ensuite, inclure un contrôle actif du profil en écoulement incompressible, dans le but d'augmenter le vitesse critique de flottement. On a démoutié aussi que la solution des équations aéroélastiques dans le domaine de Laplace représente un outil puissant pour l'analyse du flottement.

Acknowledgements

The author wishes to express his sincere gratitude to Professor S. J. Price for his guidance during the course of this work. His valuable suggestions, friendly discussions and the time spent in reviewing the manuscript are well acknowledged.

The work was made possible through the financial assistance of a Natural Sciences and Engineering Research Council (NSERC) post-graduate scholarship.

Acknowledgement is also given to the Institute of Aerospace Research (formerly National Aeronautical Establishment) of Canada who partially financed this research under contract number 055SS 31946-8-0014.

Contents

List of Figures

List of Tables

List of Appendices

Nomenclature

symbol	
a	non-dimensional distance measured from midchord to elastic axis (positive towards trailing edge)
$a + ib$	eigenvalue of form $\beta + i\omega$, a and b give damping and frequency, respectively
a_T	control of damping input to system by flap
a_m	numerator coefficients Pade approximate
$[A_j]$	linear coefficients of approximating rational function
b	semi-chord of airfoil
b_j	nonlinear lag terms of approximating rational function
b_l	denominator coefficients Pade approximate
c	chord
$c + id$	eigenvalue of form $\beta + i\omega$, c and d give damping and frequency, respectively
c_β	non-dimensional distance measured from the midchord to the flap hinge line (positive towards trailing edge)
$C(k)$	exact Theodorsen function $F + iG$
$\bar{C}(k)$	approximate Theodorsen function $\bar{F} + i\bar{G}$
$[C]$	damping matrix (structural and/or aerodynamic)
C_{l_h}	lift coefficient due to plunge
C_{l_α}	lift coefficient due to pitch
C_{m_h}	moment coefficient due to plunge

C_{m_α}	moment coefficient due to pitch
$c.g.$	center of gravity
$D(p)$	denominator of Pade approximant
$e.a.$	elastic axis
$err(ik)$	error function
F	real part
$F_o, \ldots, F_m$	coefficients of a m^{th} order polynomial
g	structural damping coefficient
G	imaginary part
h	plunge displacement
i	complex variable $\sqrt{-1}$
I	wing mass moment of inertia about center of gravity
I_α	wing mass moment of inertia about elastic axis
I_β	flap mass moment of inertia about flap hinge line
$[I]$	identity matrix
k	reduced frequency wb/U
k_c	reduced frequency wc/U
k_{cr}	flutter reduced frequency wb/U
K_h	spring constant in bending at elastic axis
K_α	spring constant in pitching at elastic axis
K_β	spring constant in pitching at flap hinge line
$[K]$	stiffness matrix (structural and/or aerodynamic)
L	lift force
m	mass of airfoil per unit span

M	free stream Mach number
M_α	moment about elastic axis
M_β	moment about flap hinge line
$[M]$	inertia matrix (structural and/or aerodynamic)
$M(ik)$	normalizing function
$[N]$	approximation matrix, a function of reduced frequency
$N(p)$	numerator of Pade approximant
p	complex eigenvalue $\beta + i\omega$
P	parameter gives nature of rational function
PD	proportional-derivative control
PID	proportional-integral-derivative control
q	dynamic pressure $1/2\rho U^2$
q_c	control generalized coordinate parameter, i.e. β
$Q(ik)$	tabular data of unsteady aerodynamics
$Q(ik)$	approximate data of unsteady aerodynamics
Q_h	total unsteady, nonconservative forces
Q_α	total unsteady, nonconservative moments
R	order-of-fit
r_α	radius of gyration about elastic axis
r_β	radius of gyration about flap hinge line
s	Laplace variable, $i\omega$
t	time variable
$[T]$	transfer function (1×2 matrix)
U	free stream velocity

U^*	non-dimensional velocity U/bw_α
x	general Laplace solution multiplier (flow regime: incompressible or transonic)
x_α	non-dimensional distance measured from elastic axis to center of mass
α	pitch angle
α_o	complex constant pitch angle
β	flap angle
ζ	damping ratio $-\beta/\sqrt{\beta^2 + w^2}$ ($p - k$ and Laplace methods)
γ	damping ratio $-g/2$ ($U - g$ method)
ζ_h	viscous damping for plunging motion
ζ_α	viscous damping for pitching motion
λ	flutter eigenvalue
λ_g	$\mu(1 + ig)w_\alpha^2 b^2/U^2$
μ	airfoil-air mass ratio $m/\pi\rho b^2$
ξ	non-dimensional displacement h/b
ξ_o	complex constant non-dimensional displacement h/b
ρ	air density
w	oscillation frequency
w_h	uncoupled bending frequency
w_α	uncoupled torsional frequency
w_β	uncoupled torsional frequency
w_r	reference frequency, normally taken as no control flutter frequency
$[\,]_s$	structural quantity
$[\,]_c$	control quantity

1 Introduction

1.1 Motivation

Aeroelasticity is concerned with the interaction of aerodynamic forces and the resulting structural deformations for a structure in an airflow. Flutter, which is one example of an aeroelastic phenomena, is a flow induced vibration that may result in the catastrophic failure of an aircraft. The traditional methods of predicting the flutter velocity, namely the $U - g$ and $p - k$ methods, have been used with great success. However, in recent years, new methods have been suggested which approximate the unsteady aerodynamic forces and moments by rational functions. The resulting equations are then solved in the Laplace domain. This allows for a more efficient solution to the flutter problem, and also, allows for a direct extension of the aeroelastic equations for the implementation of active control.

The goals of this work are twofold. Firstly, to solve the equations of motion for an airfoil in both incompressible and transonic flow in the Laplace domain using rational functions to approximate the unsteady aerodynamics. These results will be compared with $U - g$ and $p - k$ generated results. Secondly, to implement active control for the airfoil in incompressible flow, with the aim of increasing the critical flutter speed.

1.2 Flutter Definition

In the field of aeroelasticity (Fung 1955; Ashley, Bisplinghoff and Halfman 1955; Ashley and Bisplinghoff 1962), no topic has received more attention than that of flutter. Flutter is a violent self-induced aerodynamic vibration that may affect airplane wings, tail surfaces and control tabs. The following example will give a more physical representation of flutter. Consider a cantilever airfoil, with no sweepback and no aileron, rigidly attached to the side wall of a wind tunnel. The wing is then confined to exhibit motion only in the bending and pitching directions. With the air flow turned off, the airfoil is given an initial small amplitude disturbance, and then allowed to freely oscillate. The amplitude of the oscillation will decay until it is totally damped out. When the air flow is turned on, this original oscillation will decay much quicker. With subsequent increases in air speed the system will show improved decay characteristics; however, there exists a specific speed where the total system damping will begin to decrease. The air speed where the wing shows constant amplitude motion is referred to as the critical flutter speed. At speeds slightly greater, a small disturbance force will trigger violent oscillations, that in most cases lead to immediate catastrophic failure of the airfoil.

The dramatic and destructive consequences of flutter have gained the interest of much of the aeronautical community. One industry representative went so far as to state that : " The flutter problem is now generally accepted as a problem of primary concern in the design of current aircraft structures. Stiffness criteria based on flutter requirements are, in many instances, the critical design criteria." (Head 1958). In recent years, airplane design has resulted in the emergence of lighter, more flexible aircraft, with an increase in maximum speed, resulting in an increase in the likelihood of flutter occurring. The transonic regime tends to be the most critical with respect to flutter; however, there are still many other flutter problems that require attention.

Aerodynamic structures are, by definition, continuous, having an infinite number of degrees-of-freedom. However, most theoretical analysis of such structures discretize the structure into a finite number of degrees-of-freedom, using, for example, either the finite element method or the assumed mode method. To obtain accurate estimates of the flutter velocity it is necessary to consider a large number of modes in the flutter analysis. However, it has been found that only two degrees-of-freedom, namely bending and pitching, are required to give a good physical representation of flutter. It is the coupling between these two modes that produces flutter. A rigid airfoil allowed only to have bending motion does not flutter. In general, a rigid airfoil with only a torsional degree-of-freedom will not flutter. It is the coupling of the two degrees-of-freedom that allows flutter to possibly exist. Past experience has shown that near coincidence of the bending and torsion frequencies is one of the main factors in the cause of flutter, and that the character of the torsional mode plays a more fundamental part in its occurrence.

The only source of external energy causing the flutter is that of the airstream. Hence the airstream must be the key component in the instability of the airfoil. In fact, a complete analysis of the interaction of the structural forces coupled with the aerodynamic forces is required to determine when the system instability will occur. The aerodynamic forces increase with velocity. However, the stiffness in the bending and torsional directions is independent of the air speed. Any damping that is introduced to the system comes from the aerodynamic forces; the structural damping is often so negligible that it is omitted in most aeroelastic analysis. Hence, there exists a critical speed at which the aerodynamic damping is insufficient to give system stability, and flutter occurs.

1.3 Unsteady Aerodynamics

The interaction of an airstream with an airfoil produces a behaviour that is very complicated. The calculation of the resulting aerodynamic forces presents, possibly, the major difficulty in the analysis of flutter. The analysis of flutter has seen a gradual progression in the complexity of the aerodynamic forces, this has considered the aerodynamics initially as steady, then quasi-steady, and finally the true unsteady motion of the aerodynamics has been considered.

When an airflow reacts with an airfoil, disturbances are introduced that result in the flow following the contour of the airfoil. According to thin airfoil theory (Fung 1955), the airfoil can be represented by a continuous distribution of vorticity. The strength and configuration of this vortex sheet produces the aerodynamic properties of the flow for the desired airfoil. However, this is not quite correct, not only does there exist the bound vortices on the airfoil, but there also exist free vortices in the wake of the airfoil. These free vortices are being produced continuously by the oscillating airfoil, shed at the trailing edge of the wing and carried downstream by the flow. These free vortices also produce vertical components of velocity on the wing, thus altering the aerodynamic properties of the wing. Steady flow analysis completely neglects the frequency of oscillation of the airfoil, and hence the free vortices produced by this oscillation.

Quasi-steady flow analysis realizes that vortices are being produced in the wake, but assumes they have no effect. This is true if these free vortices are swept quickly downstream. A major simplification of applying a quasi-steady analysis is that the aerodynamics can be considered independent of the airfoil's frequency of oscillation, which is generally not the case. However, if this assumption is made, it eliminates a time consuming iterative process. The strength of this theory is that it gives qualitative information about the system, without an extreme computational burden; but caution must be taken, since the method has made many simplifying assumptions.

Unsteady flow analysis accounts for the free vortices in the wake. The aerodynamics are now a function of the frequency of oscillation, $k = wb/U$ (or $k_c = wc/U$), where w is frequency, $c = 2b$ is the chord and U is the free stream velocity. Thus, for a given velocity, the frequency of oscillation is required, before the aerodynamic forces can be determined. This results in an iterative procedure. If the frequency of oscillation is small, steady or quasi-steady flow analysis may suffice, but for the most part this is not the case and an unsteady flow analysis must be used. Due to the interaction of the flow with the airfoil geometry, lag effects result in the flow; the circulation around the airfoil is not developed instantaneously due to a change in incidence, but there is a phase lag between the motion and the resulting aerodynamic forces. These lag effects can lead to an amplification of the small oscillatory motions.

The main difficultly in solving the aeroelastic equations of motion rests with the calculation of the unsteady aerodynamics. The unsteady aerodynamics are sometimes written as Q_h and Q_α, where $Q_h = -L$ and $Q_\alpha = M_\alpha$ are the total unsteady, nonconservative forces and moments acting about the elastic axis, respectively.

The quantities Q_h and Q_α (Lee 1984) can be expressed in this form

$$Q_h = -qc(C_{l_h}\frac{\xi}{2} + C_{l_\alpha}\alpha) \; , \tag{1}$$

$$Q_\alpha = qc^2(C_{m_h}\frac{\xi}{2} + C_{m_\alpha}\alpha) \; , \tag{2}$$

where, C_{l_h}, C_{l_α}, C_{m_h} and C_{m_α} are the nondimensional derivatives of the lift and moment coefficients, with respect to plunging, h, and pitching, α, respectively. The dynamic pressure, q, is a function of velocity, and given by

$$q = \frac{1}{2}\rho V^2 \; , \tag{3}$$

where ρ is the air density.

The determination of the unsteady aerodynamics is not an easy task. Theodorsen has determined analytically the unsteady aerodynamics for incompressible flow on a two-dimensional airfoil (Theodorsen 1935; Garrick and Theodorsen 1941). The derivation of these equations is based on potential flow theory and the Kutta condition at the trailing edge, and assumes inviscid, irrotational flow. All derivations assume small amplitude oscillations, and thus, this allows for the linearization of the equations. The unsteady aerodynamics, Q_h and Q_α, are written as a function of the Theodorsen Function, $C(k)$, and are expressed as:

$$Q_h = -L = -\pi\rho b^2\left(\ddot{h} + U\dot{\alpha} - ba\ddot{\alpha}\right) -$$

$$2\pi\rho U b C(k)\left(\dot{h} + U\alpha + b(0.5 - a)\dot{\alpha}\right) \; , \tag{4}$$

and

$$Q_\alpha = M_\alpha = \pi\rho b^2\left(ba\ddot{h} - Ub(0.5 - a)\dot{\alpha} - b^2(.125 + a^2)\ddot{\alpha}\right) +$$

$$2\pi\rho U b^2(a + 0.5)C(k)\left(\dot{h} + U\alpha + b(0.5 - a)\dot{\alpha}\right) \; . \tag{5}$$

The Theodorsen Function $C(k)$, composed of a real part F and an imaginary part G, is a function of reduced frequency, k,

$$C(k) = F + iG \,, \tag{6}$$

$$k = \frac{wb}{U} \,, \tag{7}$$

where w is frequency, b is a characteristic length, the semi-chord, and U is the velocity. Note that the equations are both a function of velocity and frequency. This will result in an iterative type solution even for a fixed velocity.

For the transonic regime, readily solvable equations do not exist. This is due to the nonlinearity and irregularity of the associated unsteady transonic aerodynamics. The analysis of transonic flow started with linear potential theory in the 1950's and has progressed to the use of the nonlinear small disturbance equations, nonlinear Euler and the Navier-Stokes equations. Several methods, such as the relaxation methods (STRANS2 and UTRANS2) developed by Traci, Albano and Farr in 1974, and the indirect method (LTRAN2) developed by Ballhaus and Goorjian in 1977, can be used to determine the unsteady aerodynamic coefficients C_{l_h}, C_{l_α}, C_{m_h} and C_{m_α} as a function of reduced frequency, k, for transonic flow. LTRAN2 was based on the transonic small disturbance equations, and utilized a time integration (indicial) finite difference method. However, it gave accurate results only for low values of reduced frequency. Rizzetta and Chin furthered the work of Ballhaus and Goorjian in 1979, by integrating the LTRAN2 program and the structural equations of motion. Since then, the work by Houwink and van der Vooren, and later by Couston and Angelini, have taken the original code and produced much improvement in the accuracy and range of results for increased reduced frequencies.

It is now routine to calculate the aerodynamic derivatives, C_{l_h}, C_{l_α}, C_{m_h} and C_{m_α}

as a function of reduced frequency, k, to give the unsteady aerodynamics required for the aeroelastic equations of motion. The unsteady aerodynamics are calculated by performing computer simulations of pure bending and pure twisting in small disturbance flows. These equations cannot be linearized because the flow behaviour is so irregular in the transonic regime. Finding these unsteady aerodynamics for this regime is important since this regime tends to be the most critical for the flutter of wings. Lee (1984) gives the results for the transonic unsteady aerodynamic derivatives as a function of reduced frequency, k, for $M = 0.80$, 0.85 and 0.875.

1.4 Flutter Solution

Prior to 1938, it was thought that the solution to the flutter problem could be found by flight testing alone. It was supposed that flutter could be determined by noting a reduction in the system damping with increase in velocity. Unfortunately, in February of that year, during a carefully planned flight test, an aircraft went down killing all the crew and scientists. Such testing had a number of shortcomings that affected reliability and safety (Weissenburger and Zimmerman 1964). In particular, it was shown that flutter can occur very quickly and without much warning. Thus, it was realized that to determine the onset of flutter, a theoretical analysis must be used at the initial design stage and for the modification of the existing aircraft, in conjunction with flight testing still being the ultimate test of when flutter will occur.

New theoretical techniques consider an eigenvalue analysis of the flutter equations of motion. A two-degree-of-freedom rigid airfoil is represented as a flat plate, which is flexibly mounted and undergoing bending and twisting (torsion) motion (see Figure 1).

Bending (plunge) is denoted by h, positive downward, and the twisting (pitch) about the elastic axis is denoted by α, positive in the clockwise direction. The elastic axis is defined as the point in which a load force would produce only pure bending (no twist). The aeroelastic equations of motion, which are a function of inertia forces, elastic forces and aerodynamic forces (Fung 1955; Ashley and Bisplinghoff 1962), can be derived by summing forces and moments, about the elastic axis:

$$m\ddot{h} + m(x_\alpha b)\ddot{\alpha} = -K_h h - L \ , \tag{8}$$

$$(I + m(x_\alpha b)^2)\ddot{\alpha} + m(x_\alpha b)\ddot{h} = -K_\alpha \alpha + M_\alpha \ . \tag{9}$$

where m is the mass per unit span, I is the wing mass moment of inertia about the centre of gravity, $c = 2b$ is the chord length, $x_\alpha b$ is the distance from the centre of gravity to the elastic axis, ab is the distance measured from the midchord to elastic axis, both positive towards the trailing edge, K_h is the spring constant in bending, K_α is the spring constant in twisting, and L and M_α are the lift force and moment acting about the elastic axis, respectively.

The calculated roots of the system aeroelastic equations give an indication of stability. This form of analysis gives an alternative indication of the onset of flutter. There exists two well established techniques for determining flutter; in particular for dealing with the iterative nature of the solution, namely the $p-k$ (British) and $U-g$ (American) methods, see for example Lee (1984).

In the $U-g$ method, harmonic oscillatory motion is assumed, such that the generalized coordinates, ξ and α, can be written in the following form:

$$\xi = \xi_o e^{\,i\omega t}$$

$$\alpha = \alpha_o e^{\,iwt} \,, \tag{10}$$

where ξ_o and α_o are complex constants, the absolute values of which represent the amplitudes, and their arguments give phase angles. A theoretical structural damping component, g, is then introduced into the equations of motion, in order to produce steady harmonic system motion. The rational behind the $U - g$ method is that the system will require just a sufficient amount of this fictitious hysteretic structural damping, g, to maintain steady harmonic motion. If the system requires the input of negative damping $(-g)$ to make it harmonic, it is stable, likewise if it requires positive damping $(+g)$ to make it harmonic, it is unstable. An expression for the damping ratio (Lee 1984) can be written as

$$\gamma = -\frac{g}{2} \,. \tag{11}$$

Hence, a positive value of damping ratio indicates a stable system.

In the $p - k$ method, the generalized coordinates, ξ and α, take on a more general form

$$\xi = \xi_c e^{\,pt}$$

$$\alpha = \alpha_o e^{\,pt} \tag{12}$$

where ξ_o and α_o are complex constants, and $p = \beta + iw$, where β gives an indication of damping and w gives frequency. The $p - k$ method solves for the critical flutter velocity by utilizing an iteration procedure of frequency for each eigenvalue. The damping ratio (Lee 1984), a function of β and w, can be given as

$$\zeta = -\frac{\beta}{\sqrt{\beta^2 + w^2}} \,. \tag{13}$$

Flutter occurs when there is insufficient aerodynamic damping to keep the system stable. Hence, a negative value for the system damping ratio would indicate an instability.

Both methods agree at the critical instability point. but it has been found that the American approach can largely over estimate the magnitude of the relative damping ratio for values of subcritical velocity (Jackson and Lawrence 1968). These methods have been used to great success in flutter analysis, but they tend to be time consuming iterative processes. In recent years, the focus of flutter analysis has turned away from the time domain analysis to the frequency domain or Laplace domain.

The Laplace domain employs rational function approximations of the unsteady aerodynamic forces. When the unsteady aerodynamics are approximated this eliminates much of the iterative work that characterizes the previous methods. The aerodynamic forces are usually found in tabular form for simple harmonic motion at discrete values of reduced frequency. The tabular data for the unsteady aerodynamics has been, for sometime, readily available for the incompressible regime, in the form of the Theodorsen Function, $C(k)$. It is not until recently, that advancements in transonic numerical methods for computing unsteady aerodynamic forces in the transonic regime, have made it possible to consider this regime for further aeroelastic analysis. Transonic codes can now routinely calculate the required unsteady aerodynamic coefficients C_{l_h}, C_{l_α}, C_{m_h} and C_{m_α} at various Mach numbers.

The idea of using rational functions to approximate unsteady aerodynamics is not a new concept. Jones (1940) is credited with first using rational Laplace transform functions to approximate unsteady aerodynamics. He considered an approximation of the Theodorsen Function, C(k), of this form,

$$C(k) = 1 - \frac{0.165ik}{ik + 0.0455} - \frac{0.335ik}{ik + 0.32} \, . \tag{14}$$

The importance of such approximations were not fully realized until its versatility in the analysis of active control systems was discovered. However, for the proper analysis of the aeroelastic equations of motion, the governing rational functions have to be somewhat more robust, than that first introduced by Jones. A rational function is required that gives good approximation to the unsteady aerodynamics as a function of reduced frequency, with a compatible amount of complexity. The approximation of the incompressible unsteady aerodynamics, namely the Theodorsen Function, $C(k)$, as a function of reduced frequency, k, may not pose much of a difficulty; however, problems may be encountered in the transonic regime, where the unsteady transonic aerodynamic coefficients C_{l_h}, C_{l_α}, C_{m_h} and C_{m_α} vary in a very irregular manner with reduced frequency. Thus, in order for the Laplace method to be viable, the approximations of these tabular unsteady aerodynamics can not be considered lightly.

A function $\bar{Q}(ik)$ is required that approximates $Q(ik)$, as a function of reduced frequency, k, and gives good results over the entire specified range . The tabular data $Q(ik)$ of the unste..dy aerodynamic forces may be written in terms of real and imaginary parts as a function of reduced frequency,

$$Q(ik) = F + iG \ , \tag{15}$$

where F and G are the real and imaginary parts, respectively.

Many rational functions can be used to approximate the unsteady aerodynamics (Poirel 1988). The Nasa-Langley rational function has been shown to be fairly robust in its ability to approximate given data (Adams and Tiffany 1988), and is given by

$$\bar{Q}(p) = A_0 + A_1 p + \cdots + A_{\bar{P}} p^{\bar{P}} + \sum_{l=1}^{R} A_{l+\bar{P}} \frac{p}{p + b_l} \ , \tag{16}$$

where $p = ik$. The coefficients A_0, A_1, ... and b_1, b_2, ... are chosen to ensure minimum approximation error to the given tabular data. This is the form of the approximation employed by Tiffany and Adams (1988), and the same method that is used in this thesis. The associated error distribution seems to be less, and it has a built in robustness that lets it consider a larger initial frequency range. The physical interpretation of the equation is that the first three terms stem from quasi-steady influences, and the other terms give the unsteady effects. The lag terms b_j give an approximation to the time delays (lag in the development of the circulation about the airfoil) inherent in unsteady aerodynamics.

In the past, these lag terms were not optimized, but arbitrarily selected from the range of reduced frequencies for which tabular data of the unsteady aerodynamic forces were available (Abel 1979). Present methods, that optimize these nonlinear lag terms, give better results with a decreased order in the approximating equation, and thus a solution of less complexity and increased efficiency. Optimization of the rational function coefficient to ensure minimum approximation error was performed by utilizing a least square method to determine the *linear coefficients* and a sequential simplex method to determine the nonlinear lag terms. The method employed to solve for the optimal nonlinear lag terms is a sequential simplex method (Mead and Nelder 1965; Nelson and Olsson 1975) developed by Nelder and Mead in 1965. This method is simple, robust and requires no derivatives, and hence lends itself quite well to finding the minimum of a non-linear objective function of more than one independent variable. The validity of the Laplace solution depends greatly on how well the unsteady aerodynamics can be approximated.

The advantage of approximating the unsteady aerodynamics is that it allows the aeroelastic equations of motion to be written in the Laplace domain. The equations, in this form, can be solved quite easily by readily available computer subroutines, that utilize matrix algebra and eigenvalue analysis. Abel (1979) has demonstrated how the aeroelastic equations of motion in the Laplace domain can be solved. This requires the expansion of the equations to obtain a m^{th} order polynomial of the following form

$$[F_m]\, s^m + [F_{m-1}]\, s^{m-1} + [F_{m-2}]\, s^{m-2} + \cdots + [F_1]\, s + [F_0] = 0 \ , \tag{17}$$

where the coefficients $[F_m]$, $[F_{m-1}]$, $[F_{m-2}]$ $\cdots$ $[F_0]$ are functions of dynamic pressure (velocity).

This can be easily placed in a typical eigenvalue problem of the given form

$$s\{x\} = [A]\{x\} \ , \tag{18}$$

to give a series of $m \times n$ first order equations, where

$$\{x\} = \{\ s^{m-1}\ ,\ s^{m-2}\ \cdots\ s^1\ ,\ s^0\ \} \ . \tag{19}$$

These equations can be solved quite easily by linear matrix techniques.

1.5 Flutter-Suppression Systems

Flutter-suppression systems have received increased attention in the last two decades (Kass and Thompson 1971; Guruswamy, Olsen, Striz and Yang 1980; Karpel 1981). They have been found to be an effective way of increasing the flutter speed. There are two types of flutter-suppression systems: passive and active flutter control. Passive control includes increasing the structural stiffness and/or mass balancing. Proper mass balancing has a direct effect on the system inertia coupling, which is often a predominant factor in the occurrence of flutter. Active control uses a control surface which is deflected in response to the wing motion, resulting in a change in the aerodynamic forces on the wing. The first application of such a system in flight was on the B-52 in the mid 1970's. Much work has been done in wind tunnels and flight tests to show the feasibility of implementing active control (Nissim and Abel, 1978; Abel 1979; Messina and O'Connell, 1979). Increases in critical flutter speed of 20-25 % have been obtained through these methods.

The problem associated with applying an active control device to an airfoil hinges on developing a set of aeroelastic equations, where the form of the unsteady aerodynamics and the control law are compatible. This can be achieved by approximating the unsteady aerodynamics by a rational function, and thus allows for both the unsteady aerodynamics and the control law to be written in the Laplace domain.

The aeroelastic equations for a three-degree-of-freedom rigid airfoil including the incorporation of active control, a flap, (see Figure 10) are given by numerous references (Fung 1955; Ashley and Bisplinghoff 1962). The flap is deflected by an angle β, in response to the wing motion, a function of the plunge, h/b, and pitch, α, with the aim of causing an increase in the flutter speed. Control laws of the following form have been considered (Nissim and Abel 1978),

$$\{ \beta \} = [T] \left\{ \begin{array}{c} h/b \\ \alpha \end{array} \right\} , \tag{20}$$

where $[T]$ is a transfer function matrix of size 1×2.

The transfer function used was a function of h/b and α and their first derivatives,

$$[T] = [\, t_{11} \ , \ t_{12} \,] + \frac{a_T}{w_R} s [\, t_{11}^* \ , \ t_{12}^* \,] \ . \tag{21}$$

This is referred to as a *damping type transfer function* by Nissim and Abel (1978), where a_T is a control of the amount of damping introduced by the control surface (flap), and w_R is a reference frequency, normally taken as the no-control flutter frequency. This transfer function can be considered as a type of proportional-derivative (PD) control.

In the late 1970's, considerable work was done on flutter control systems to determine realistic transfer function parameters (Nissim and Abel 1978; Nissim 1977; Nissim 1971). Nissim and Abel (1978) used a theory based on the aerodynamic energy concept to determine realistic transfer function parameter values. The aerodynamic energy concept considers the work done by the aerodynamic forces on the wing per cycle of oscillation, referred to as P. Hence, the transfer function $[T]$ was determined to give a stable system, or a negative value of work P. Nissim and Abel determined the optimal parameters for the transfer function for an airfoil of specific geometric configuration. A simple transformation was also derived that allowed for changes in airfoil geometry.

With the introduction of the control law into the aeroelastic equations of motion, the equations can be expanded, as before, to obtain a m^{th} order polynomial of the following form

$$[F_m]\, s^m + [F_{m-1}]\, s^{m-1} + [F_{m-2}]\, s^{m-2} + \cdots + [F_1]\, s + [F_0] = 0 \; , \tag{22}$$

where the coefficients $[F_m]$, $[F_{m-1}]$, $[F_{m-2}]$ $\cdots$ $[F_0]$ are functions of dynamic pressure (velocity). The solution follows the same methodology as shown earlier.

1.6 Thesis Overview

This study considers the analysis of the aeroelastic equations of motion of a rigid airfoil, flexibly mounted, undergoing bending and twisting motion, as it pertains to flutter. Initially, the formulation of the equations of motion for a two-degree-of-freedom system are considered in chapter 2. Chapter 3 discusses the unsteady aerodynamics and their associated difficulties for both the incompressible and transonic regimes. An overview of eigenvalue analysis methods, techniques that are commonly used for all forms of analytical flutter calculation, is described in detail in chapter 4. In chapter 5 the traditional solutions to the flutter problem, namely the $p - k$ and $U - g$, are derived for the general case of incompressible flow.

In chapter 6 the Laplace method is introduced. This method requires the approximation of the unsteady forces and moments by a rational function. The rational function employed for this thesis is of the Nasa-Langley form. It requires the use of a two stage function optimization (Adams and Tiffany 1988). A least square method is used to determine the linear coefficients and a sequential simplex method (Mead and Nelder 1965) to determine the nonlinear lag terms. A general Laplace solution is derived to consider not only incompressible flow, but also that of transonic flow.

Chapter 7 is devoted mainly to the presentation and discussion of the results for the approximation of the unsteady aerodynamics, for both the incompressible and transonic regimes. The following chapter, chapter 8, is devoted to a comparison of the traditional methods, namely the $U - g$ and $p - k$, with the Laplace method. Comparisons are done in the transonic regime by using previous $U - g$ and $p - k$ results given by Lee (1984).

In Chapter 9, the incompressible equations of motion for a three-degree-of-freedom airfoil are formulated to consider control. Active control is considered with the aim of obtaining an increase in the critical flutter speed. The values suggested by Abel and Nissim, who employed the Aerodynamic Energy Concept (Abel and Nissim 1978), are used as a starting point to find the optimal transfer function parameters for the control law employed. The simplex method is then used to determine if these are indeed the optimal values. Chapter 10 is devoted to a discussion of the consequences of implementing active control for the incompressible flow regime.

The final chapter, chapter 11, summarizes the conclusions.

The Appendices contain additional notes, tables and figures, as well as, complete computer program listings ($U - g$, $p - k$ and Laplace methods, rational function optimization, incompressible control using Laplace method).

2 Equations of Motion 2DOF

Consider a two-degree-of-freedom rigid airfoil, represented as a flat plate, flexibly mounted undergoing bending and twisting (torsion) motion (see Figure 1). Bending (plunge) is denoted by h, positive downward, and the twisting (pitch) about the elastic axis is denoted by α, positive in the clockwise direction. The elastic axis is defined as the point in which a load force would produce only pure bending (no twist). The aeroelastic equations of motion, which are a function of inertia forces, elastic forces and aerodynamic forces (Fung 1955; Ashley and Bisplinghoff 1962), can be derived by summing forces and moments, about the elastic axis:

$$m\ddot{h} + m(x_\alpha b)\ddot{\alpha} = -K_h h - L \ , \tag{23}$$

$$(I + m(x_\alpha b)^2)\ddot{\alpha} + m(x_\alpha b)\ddot{h} = -K_\alpha \alpha + M_\alpha \ . \tag{24}$$

where m is the mass per unit span, I is the wing mass moment of inertia about the centre of gravity, $c = 2b$ is the chord length, $x_\alpha b$ is the distance from the centre of gravity to the elastic axis, ab is the distance measured from the midchord to elastic axis, both positive towards the trailing edge, K_h is the spring constant in bending, K_α is the spring constant in twisting, and L and M_α are the lift force and moment acting about the elastic axis, respectively.

Past experience has shown that structural damping can often be assumed to be negligible for wing flutter analysis, and thus, it is not included.

Introducing a non-dimensional generalized displacement coordinate

$$\xi = \frac{h}{b} \tag{25}$$

and making the following substitutions

$$K_h = w_h^2 m \ , \tag{26}$$

$$K_\alpha = w_\alpha^2 I_\alpha \ , \tag{27}$$

where

$$I_\alpha = J + m(x_\alpha b)^2 \ , \tag{28}$$

gives

$$mb\ddot{\xi} + m(x_\alpha b)\ddot{\alpha} = -w_h^2 mb\xi - L \ , \tag{29}$$

$$I_\alpha \ddot{\alpha} + m(x_\alpha b^2)\ddot{\xi} = -w_\alpha^2 I_\alpha \alpha + M_\alpha \ . \tag{30}$$

The above two equations may be simplified by dividing through by mb and mb^2, respectively to give

$$\ddot{\xi} + x_\alpha \ddot{\alpha} + w_h^2 \xi = \frac{-L}{mb} \ , \tag{31}$$

$$\frac{I_\alpha}{mb^2}\ddot{\alpha} + x_\alpha \ddot{\xi} + w_\alpha^2 \frac{I_\alpha}{mb^2}\alpha = \frac{M_\alpha}{mb^2} \ . \tag{32}$$

Finally, using $I_\alpha = r_\alpha^2 mb^2$, where r_α is the radius of gyration about the elastic axis, the complete aeroelastic equations become

$$\ddot{\xi} + x_\alpha \ddot{\alpha} + w_h^2 \xi = \frac{-L}{mb} \ , \tag{33}$$

$$r_\alpha^2 \ddot{\alpha} + x_\alpha \ddot{\xi} + w_\alpha^2 r_\alpha^2 \alpha = \frac{M_\alpha}{mb^2} \ . \tag{34}$$

3 Unsteady Aerodynamics

The calculation of the unsteady aerodynamics forces presents, possibly, the major difficulty in the analysis of flutter. The unsteady aerodynamics are sometimes written as Q_h and Q_α, where $Q_h = -L$ and $Q_\alpha = M_\alpha$ are the total unsteady, nonconservative forces and moments acting about the elastic axis, respectively.

The quantities Q_h and Q_α can be expressed in this form

$$Q_h = -qc(C_{l_h}\frac{\xi}{2} + C_{l_\alpha}\alpha) \ , \tag{35}$$

$$Q_\alpha = qc^2(C_{m_h}\frac{\xi}{2} + C_{m_\alpha}\alpha) \ , \tag{36}$$

where, C_{l_h}, C_{l_α}, C_{m_h} and C_{m_α} are the derivatives of the lift and moment coefficients, with respect to h, plunging, and α, pitching, respectively. The dynamic pressure, q, is a function of velocity, and given by

$$q = \frac{1}{2}\rho V^2 \ . \tag{37}$$

The solution to the flutter equations of motion is complicated by the nature of the unsteady aerodynamics. They tend to vary in a very nonlinear manner with reduced frequency, which results in an iterative type solution.

3.1 Incompressible Two-Dimensional Aerodynamics

In the incompressible regime, Q_h and Q_α take on exact forms, expressed as a function of the Theodorsen Function $C(k)$ (Theodorsen 1935; Garrick and Theodorsen 1941),

$$Q_h = -L = -\pi\rho b^2\left(\ddot{h} + U\dot{\alpha} - ba\ddot{\alpha}\right) -$$

$$2\pi\rho UbC(k)\left(\dot{h} + U\alpha + b(0.5 - a)\dot{\alpha}\right) \ . \tag{38}$$

and

$$Q_\alpha = M_\alpha = \pi\rho b^2\left(ba\ddot{h} - Ub(0.5 - a)\dot{\alpha} - b^2(.125 + a^2)\ddot{\alpha}\right) +$$

$$2\pi\rho Ub^2(a + 0.5)C(k)\left(\dot{h} + U\alpha + b(0.5 - a)\dot{\alpha}\right) \ . \tag{39}$$

The Theodorsen Function, $C(k)$, (see Table 1), composed of a real part F and an imaginary part G, is a function of reduced frequency, k,

$$C(k) = F + iG \tag{40}$$

$$k = \frac{wb}{U} \tag{41}$$

where w is frequency, b is a characteristic length, the semi-chord, and U is the velocity. Note that the equations are a function of both velocity and frequency. This will result in an iterative type solution even for a fixed velocity.

The derivation of these equations is based on potential flow theory and the Kutta condition at the trailing edge, and assumes incompressible, inviscid and irrotational flow. Furthermore, all derivations assume small amplitude oscillations for h and α, so the flow remains potential and unseparated. This allows for the linearization of the equations.

Theodorsen Function C(k)

k	F	G
10.00	0.5006	-0.0124
6.000	0.5017	-0.0206
4.000	0.5037	-0.0305
3.000	0.5063	-0.0400
2.000	0.5129	-0.0577
1.500	0.5210	-0.0736
1.000	0.5394	-0.1003
0.800	0.5541	-0.1165
0.500	0.5979	-0.1507
0.400	0.6250	-0.1650
0.300	0.6650	-0.1793
0.200	0.7276	-0.1886
0.100	0.8320	-0.1723
0.050	0.9090	-0.1305
0.025	0.9545	-0.0872
0.010	0.9824	-0.0482
0.000	1.0000	-0.0000

Table 1 Theodorsen Function $C(k) = F(k) + iG(k)$, where $k = wb / U$

3.2 Transonic Two-Dimensional Aerodynamics

As yet, no such readily solvable equations for Q_h and Q_α exist for the transonic regime. Due to the associated nonlinearities and irregularity in this regime, it has been very difficult to model the unsteady aerodynamics. However, the computation of transonic unsteady aerodynamics has seen much development in the last decade.

Before the development of transonic numerical codes, aeroelastic analysis in this regime was considered difficult, if not impossible. Fortunately, much progress has been made in this field. The numerical methods available consider the uncoupled motion of an airfoil pitching and plunging in small disturbance flow. It also assumes the flow is inviscid and irrotational. The equations cannot be linearized, because the behaviour of flow in the transonic regime is so nonlinear. The codes now available allow for the routine calculation of the required unsteady aerodynamics, C_{l_h}, C_{l_α}, C_{m_h} and C_{m_α} (Lee 1984). The aerodynamic derivatives for a $NACA64A006$ airfoil at $M = 0.85$ are shown graphically in Figures 2a to 2d. Each aerodynamic derivative is composed of a real and an imaginary part, where the real part represents the total forces and moments in phase with the airfoil motion, and the imaginary part represents those ninety degrees out of phase with the motion. Refinements in existing codes have allowed for more time efficient calculations, as well as an increase in the range of reduced frequency, k, and for different values of M number.

Note:

The techniques presently available to generate the unsteady aerodynamics, give the forces and moments only for purely oscillatory motion at discrete values of reduced frequency (tabular data). In order to obtain solutions for decaying and growing motions (not simply harmonic, $i\omega$), the concept of analytic continuation is used. These results are extended off the imaginary axis, by iteration (or by using analytic approximating functions) of the given tabular unsteady aerodynamics. This procedure is not completely valid, but because flutter occurs along the imaginary axis of the complex plane, the approximation is sufficient.

4 Eigenvalue Analysis

The numerical techniques that are used in this study to solve the aeroelastic equations of motion rely on the equations being placed in an eigenvalue format. The procedure that each method utilizes to obtain this form is slightly different, but the end result is the same.

If a standard $2DOF$ system is considered in eigenvalue form, the solution will generate two complex conjugate pairs (four eigenvalues) of the following form

$$\lambda_1 = a \pm ib \; ,$$

$$\lambda_2 = c \pm id \; . \tag{42}$$

The roots determine the stability of the system at a given value of dynamic pressure (velocity). The real part of the roots, a and c, give an indication of the damping, β, and the imaginary part, b and d, give frequency, w. Hence, the roots having negative imaginary parts are not considered (a system can not have a negative frequency). The damping ratio, a function of β and w, is given as

$$\zeta = -\frac{\beta}{\sqrt{\beta^2 + w^2}} \; . \tag{43}$$

Flutter occurs when there is insufficient aerodynamic damping to keep the system stable. Hence, a positive real part, a or c, indicates an instability (this gives a negative value for the system damping ratio). It is difficult to determine which root will produce the instability, therefore a complete tracking of both roots is required.

Flutter analysis is commonly considered by plotting both the system frequency and the damping ratio versus dynamic pressure or velocity. Commonly a non-dimensional value of velocity is used, given by

$$\bar{U} = \frac{U}{b\omega_\alpha} . \tag{44}$$

For example, see Figure 3, (Abel 1979) where the dynamic instability occurs at a dynamic pressure of about 5.0. Another predominant characteristic of flutter is the coalescence of the two frequencies. It is not a complete coalescence, but the two frequencies become very close to one another at the onset of flutter. This can be seen from the graph of frequency versus dynamic pressure (see Figure 3).

5 Flutter Analysis : Traditional Methods

5.1 $U - g$ Method

In the $U - g$ method, harmonic oscillatory motion is assumed such that the generalized coordinates, ξ and α, can be written in the following form:

$$\xi = \xi_o e^{\,iwt} \;,$$

$$\alpha = \alpha_o e^{\,iwt} \;, \tag{45}$$

where ξ_o and α_o are complex constants, the absolute values of which represent the amplitudes, and their arguments give phase angles.

A structural damping coefficient g is introduced into (33) and (34) by multiplying the third term (stiffness) of the two equations by the factor $(1 + ig)$, a hysteretic damping. A sufficient amount of this structural damping, g, will be required to produce steady harmonic motion. For the case of incompressible aerodynamics, the appropriate equations for Q_h and Q_α, (38) and (39), respectively, are then substituted in to the equations, to give:

$$\ddot{\xi} + x_\alpha \ddot{\alpha} + (1 + ig)w_h^2 \xi = \frac{Q_h}{mb}$$

$$= -\frac{\pi \rho b}{m}\left(b\ddot{\xi} + U\dot{\alpha} - ba\ddot{\alpha}\right) - \frac{2\pi\rho U}{m} C(k)\left(b\dot{\xi} + U\alpha + b(0.5 - a)\dot{\alpha}\right) \;, \tag{46}$$

$$x_\alpha \ddot{\xi} + r_\alpha^2 \ddot{\alpha} + (1 + ig) r_\alpha^2 w_\alpha^2 \alpha = \frac{Q_\alpha}{mb^2}$$

$$= \frac{\pi \rho}{m}\left(b^2 a\ddot{\xi} - Ub(0.5 - a)\dot{\alpha} - b^2(0.125 + a^2)\ddot{\alpha}\right) + \frac{2\pi\rho U}{m}(a{+}0.5)C(k)\left(b\dot{\xi} + U\alpha + b(0.5 - a)\dot{\alpha}\right) \;. \tag{47}$$

Multiplying through by $\mu = m/\pi\rho b^2$ (a non-dimensional airfoil mass) and $c^2/4U^2$ and making the following substitutions

$$\xi = \xi_o e^{iwt} \;\; and \;\; \alpha = \alpha_o e^{iwt} \;,$$

$$\lambda = iw \;,$$

$$k_c = \frac{wc}{U} = 2k \;,$$

$$\lambda_g = \mu(1+ig)\frac{w_\alpha^2 b^2}{U^2} \;, \tag{48}$$

the preceding equations can be expressed as

$$\left(\frac{1}{4}\mu k_c^2 - \lambda_g\left(\frac{w_h}{w_\alpha}\right)^2 + \frac{1}{4}k_c^2 - ik_cC(k)\right)\;\xi_o +$$

and

$$\left(\frac{1}{4}x_\alpha\mu k_c^2 - i\frac{1}{2}k_c - \frac{a}{4}k_c^2 - 2C(k) - i(\frac{1}{2}-a)k_cC(k)\right)\;\alpha_o = 0 \;, \tag{49}$$

$$\left(\frac{1}{4}x_\alpha\mu k_c^2 - \frac{1}{4}ak_c^2 + i(a+\frac{1}{2})C(k)k_c\right)\;\xi_o +$$

$$\left(\frac{1}{4}r_\alpha^2\mu k_c^2 - \lambda_g r_\alpha^2 - \frac{i}{2}(\frac{1}{2}-a)k_c + \frac{1}{4}(\frac{1}{8}+a^2)k_c^2 + 2(a+\frac{1}{2})C(k) + i(a+\frac{1}{2})(\frac{1}{2}-a)C(k)k_c\right)\;\alpha_o = 0\,. \tag{50}$$

The above equations can be placed in matrix form as

$$[A]\{x\} = \lambda_g\,[B]\,\{x\} \;, \tag{51}$$

where $[A]$, $[B]$ are 2×2 matrices and

$$\{x\} = \{\xi_o \;,\; \alpha_o\}^T \;.$$

This typical eigenvalue problem can be solved to obtain λ_g for different values of reduced frequency, k, namely

$$\lambda_{g_1} = \bar{a} + i\bar{b} \; .$$

$$\lambda_{g_2} = \bar{c} + i\bar{d} \; . \tag{52}$$

Note that

$$\begin{aligned} \lambda_{g_{1,2}} &= \mu(1+ig)\frac{w_\alpha^2 b^2}{U^2} \\ &= \mu\frac{w_\alpha^2 b^2}{U^2} + i\mu g\frac{w_\alpha^2 b^2}{U^2} \; . \end{aligned} \tag{53}$$

The only unknowns in these equations are g, the structural damping coefficient, and U, the free stream velocity. Considering λ_{g_1}, U_1 and g_1 can be solved by the following equations

$$\lambda_{g_1} = \bar{a} + i\bar{b} \; .$$

Thus

$$U_1 = \sqrt{\frac{\mu w_\alpha^2 b^2}{\bar{a}}} = w_\alpha b\sqrt{\frac{\mu}{\bar{a}}} \; , \tag{54}$$

$$g_1 = \frac{U_1^2\bar{b}}{w_\alpha^2 b^2\mu} = \frac{\bar{b}}{\bar{a}} \; , \tag{55}$$

and the frequency,

$$w_1 = \frac{U_1 k}{b} \; . \tag{56}$$

An expression for the damping ratio (Lee 1984) can be written as

$$\gamma_1 = -\frac{g_1}{2} \; . \tag{57}$$

A similar analysis for λ_{g_2}, will give U_2, g_2, w_2 and γ_2. Thus, each value of reduced frequency, k, will give two velocities and their respective associated frequency and damping ratio.

The rational behind the $U-g$ method is that the system will require just a sufficient amount of this fictitious hysteretic structural damping, g, to maintain steady harmonic motion. If the system requires the input of negative damping ($-g$) to make it harmonic, it is stable; likewise if it requires positive damping ($+g$) to make it harmonic, it is unstable. If the system response is harmonic, hence $g = 0$, this indicates a stability boundary.

5.2 $p-k$ Method

In the $p-k$ method the generalized coordinates, ξ and α, take on a more general form

$$\xi = \xi_o e^{\,pt}$$

$$\alpha = \alpha_o e^{\,pt} \tag{58}$$

where ξ_o and α_o are complex constants, and $p = \beta + iw$, where β gives an indication of damping and w gives the frequency.

Substituting the appropriate equations for Q_h and Q_α in (33) and (34), again, for the particular case of incompressible aerodynamics, gives

$$\ddot{\xi} + x_\alpha \ddot{\alpha} + w_h^2 \xi = \frac{Q_h}{mb}$$

$$= -\frac{\pi \rho b}{m}\left(b\ddot{\xi} + U\dot{\alpha} - ba\ddot{\alpha}\right) - \frac{2\pi\rho U}{m}C(k)\left(b\dot{\xi} + U\alpha + b(0.5-a)\dot{\alpha}\right)\ , \tag{59}$$

$$x_\alpha \ddot{\xi} + r_\alpha^2 \ddot{\alpha} + r_\alpha^2 w_\alpha^2 \alpha = \frac{Q_\alpha}{mb^2}$$

$$= \frac{\pi\rho}{m}\left(b^2 a\ddot{\xi} - Ub(0.5-a)\dot{\alpha} - b^2(0.125+a^2)\ddot{\alpha}\right) + \frac{2\pi\rho U}{m}(a+0.5)C(k)\left(b\dot{\xi} + U\alpha + b(0.5-a)\dot{\alpha}\right)\ . \tag{60}$$

Grouping common powers of ξ and α, and using the substitution $\mu = m/\pi\rho b^2$, the above equations can be rewritten as

$$\ddot{\xi}(1+\frac{1}{\mu})+\dot{\xi}(\frac{2C(k)U}{b\mu})+\xi(w_h^2)+\ddot{\alpha}(x_\alpha-\frac{a}{\mu})+$$

$$\dot{\alpha}\left(\frac{U}{b\mu}+2C(k)(\frac{1}{2}-a)\frac{U}{b\mu}\right)+\alpha(\frac{2C(k)U^2}{b^2\mu})=0\ , \tag{61}$$

and

$$\ddot{\xi}(x_\alpha-\frac{a}{\mu})+\dot{\xi}\left(-2(a+\frac{1}{2})C(k)\frac{U}{b\mu}\right)+\ddot{\alpha}\left(r_\alpha^2+\frac{\frac{1}{8}+a^2}{\mu}\right)+$$

$$\dot{\alpha}\left(\frac{U}{b\mu}(\frac{1}{2}-a)-2(a+\frac{1}{2})(\frac{1}{2}-a)C(k)\frac{U}{b\mu}\right)+\alpha\left(r_\alpha^2w_\alpha^2-2(a+\frac{1}{2})C(k)\frac{U^2}{b^2\mu}\right)=0\ . \tag{62}$$

These two equations can then be expressed in matrix form as

$$[M]\left\{\begin{array}{c}\ddot{\xi}\\ \ddot{\alpha}\end{array}\right\}+[C]\left\{\begin{array}{c}\dot{\xi}\\ \dot{\alpha}\end{array}\right\}+[K]\left\{\begin{array}{c}\xi\\ \alpha\end{array}\right\}=0\ , \tag{63}$$

where $[M]$, $[C]$ and $[K]$ are the structural and aerodynamic mass matrix, damping matrix and stiffness matrix, respectively.

A solution to this system of equations is done by considering an eigenvalue analysis of the equations rewritten in the following form

$$\begin{bmatrix} M & C\\ 0 & I\end{bmatrix}\left\{\begin{array}{c}\ddot{z}\\ \dot{z}\end{array}\right\}+\begin{bmatrix} 0 & K\\ -I & 0\end{bmatrix}\left\{\begin{array}{c}\dot{z}\\ z\end{array}\right\}=0\ , \tag{64}$$

where

$$I=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$$

and

$$z=\{\ \xi\ \ \alpha\ \}^T\ .$$

Equation (64) can be written as

$$[D]\{\dot{p}\} + [E]\{p\} = 0 \ , \tag{65}$$

where

$$p = \{\dot{z}\ z\}^T \ .$$

and $[D]$ and $[E]$ are 4×4 matrices, that are a function of the Theodorsen Function, $C(k)$, and velocity, U. By letting $p = p_o e^{\lambda t}$, the resulting eigenvalue problem will give a solution of two complex conjugate pairs,

$$\lambda_{1,2} = \bar{a} \pm i\bar{b} \ ,$$

$$\lambda_{3,4} = \bar{c} \pm i\bar{d} \ , \tag{66}$$

where $\bar{a}$ and $\bar{c}$ are damping, and $\bar{b}$ and $\bar{d}$ represent frequency, w.

The solution to this problem is an iterative one. Initially, a specific velocity is chosen, U, and a reduced frequency, k, guessed. This then gives a frequency, w, by the equation

$$k = \frac{wb}{U} \ .$$

The choice of k, will give $C(k)$.

When solving the eigenvalue problem, only the positive imaginary parts of the complex conjugate pairs are considered. The roots having negative imaginary parts are omitted from the analysis; a system cannot have a negative frequency. Considering only one of the roots, say $\lambda = \bar{a} + i\bar{b}$, the frequency $\bar{b}$ is compared to w. If they are equal, a solution has been found, if not, it is required to calculate a new reduced frequency by

$$k = \frac{\bar{b}b}{U} \tag{67}$$

and the process is continued till convergence occurs. This same process is repeated for the other root. The idea is to create a table of velocity and its two associated solutions. The roots determine the stability, or instability, of the system at a given value of dynamic pressure (velocity). The real parts of the roots, $\bar{a}$ and $\bar{c}$, give an indication of damping, β, and the imaginary parts, $\bar{b}$ and $\bar{d}$, give frequency, w. The damping ratio (Lee 1984), a function of β and w, can be given as

$$\zeta = -\frac{\beta}{\sqrt{\beta^2 + w^2}} \ . \tag{68}$$

Flutter occurs when a negative value for the system damping is obtained.

6 Laplace Solution

More efficient algorithms for determining flutter speeds can be found by solving the equations of motion in the Laplace domain. The main problem in this approach is that of finding a suitable rational function approximation to a given tabular data set of the unsteady aerodynamics as a function of reduced frequency. This eliminates the costly iterative process, that is characteristic of the $U-g$ and $p-k$ methods.

The tabular data for the unsteady aerodynamics has been, for sometime, readily available for incompressible flow, in the form of the Theodorsen Function, $C(k)$. It is not until recently, that advancements in transonic numerical methods for computing unsteady aerodynamic forces in the transonic regime, have made it possible to consider this regime for further aeroelastic analysis. Transonic codes can now routinely calculate the required unsteady aerodynamic coefficients C_{l_h}, C_{l_α}, C_{m_h} and C_{m_α} at various Mach numbers.

6.1 Rational Function Approximation

The tabular data $Q(ik)$ for the unsteady aerodynamic forces may be written in terms of its real and imaginary parts as a function of reduced frequency.

$$Q(ik) = F + iG \ , \tag{69}$$

where F and G are the real and imaginary parts, respectively.

A function $\bar{Q}(ik)$ is required that approximates $Q(ik)$, as a function of reduced frequency, k, and gives good results over the entire specified range . There are many possible methods available that use rational functions to approximate unsteady aerodynamic given in tabular form. One method employed by Poirel (1988) uses a Pade approximant in the following form

$$\bar{Q}(p) = \frac{N(p)}{D(p)} = \frac{\sum_{m=0}^{M} a_m(p)^m}{\sum_{l=1}^{L} b_l(p)^l} , \tag{70}$$

where

$$p = ik .$$

The Pade approximant (Baker and Gammel 1970; Vepa 1976) is simply a fraction consisting of two polynomials of order M and L, where the coefficients a_m and b_l are chosen to give minimum error to the given unsteady aerodynamic tabular data. It leads to useful results, but accuracy in the range of interest depends upon a fairly complicated weighting scenario. This weighting at the point of interest tends to result in more error at the extremes of the desired frequency range. This method is accurate for a specified decreased overall reduced frequency range, but the approximation is not as robust when one is concerned with a larger overall range. Another method used to approximate given tabular data is that given by Nasa-Langley, and is often referred to as Roger's Approximation (Roger 1977)

$$\bar{Q}(p) = A_0 + A_1 p + \cdots + A_{\bar{P}} p^{\bar{P}} + \sum_{l=1}^{R} A_{l+\bar{P}} \frac{p}{p + b_l} . \tag{71}$$

The value of $\bar{P}$ depends on the nature of the function, and R is the order-of-fit of the approximant. The coefficients A_0, A_1, ... and b_1, b_2, ... are chosen to ensure minimum approximation error to the given tabular data. With the given parameters $\bar{P}$ and R made equivalent to 2 and 4, respectively, this gives

$$\bar{Q}(p) = A_0 + A_1 p + A_2 p^2 + A_3 \frac{p}{p + b_1} + A_4 \frac{p}{p + b_2} + A_5 \frac{p}{p + b_3} + A_6 \frac{p}{p + b_4} . \tag{72}$$

This is the form of the approximation employed by Tiffany and Adams (1988), and the method that is used in this thesis. The associated error distribution seems to be less, and it has a built in robustness that lets it consider a large frequency range. The physical interpretation of the equation is that the first three terms stem from quasi-steady influences, and the other terms give the unsteady effects. The lag terms b_j give an approximation to the time delays (lag in the development of the circulation about the airfoil) inherent in unsteady aerodynamics.

In the past, these lag terms were not optimized, but arbitrarily selected from the range of reduced frequencies for which tabular data of the unsteady aerodynamic forces were available (Abel 1979). Present methods, that optimize these nonlinear lag terms, give better results with a decreased order in the approximating equation, and thus a solution of less complexity and increased efficiency.

6.2 Constant Optimization

A method is required that can optimally choose the coefficients of a governing rational function, to give the best possible approximation of the unsteady aerodynamics. This optimization must keep the computational burden to a minimum , have the ability to distinguish between global minimum and local minimum, as well as converge to the same solution from varying starting values. The method must also require no derivatives, since the objective function is of considerable complexity. The result is a two stage function optimization. Optimization of the minimization of the objective function (error function) was performed by utilizing a least square method to determine the *linear coefficients* and a sequential simplex method to determine the nonlinear lag terms.

Linear Coefficients : $A_0, A_1, \ldots A_6$

The linear coefficients are determined by utilizing a least square method, based on the minimization of a given error function. Many different forms of error functions are investigated (Appendix A) to determine the one that led to the best overall approximation of the unsteady aerodynamics. They consider slight modifications in error calculation between real and approximate data and incorporate various forms of normalizing techniques. The one that seemed to give the best results (Adams and Tiffany 1988) is

$$err(ik) = \sum \frac{| \bar{Q}(ik) - Q(ik) |^2}{M(ik)} , \tag{73}$$

where

$$M(ik) = max\left\{1, | Q(ik) |^2\right\} .$$

The quantities $\bar{Q}(ik)$ and $Q(ik)$ represent the rational function approximation and the true tabular data at a given reduced frequency, respectively. The term $M(ik)$ is used to normalize the aerodynamic data, such that certain points do not receive larger than normal weighting. Note, the error function is defined as the total normalized sum for all the given values of the unsteady aerodynamic tabular data.

In order to obtain a minimum error, the derivative of the coefficients A_j, $j = 0$ *to* 6, with respect to the error function is taken

$$\frac{\partial err(ik)}{\partial (A_j)} = 0 . \tag{74}$$

The result is a set of linear algebraic equations (see Appendix B), that can be written in the following form,

$$[A]\{x\} = [B] , \tag{75}$$

where

$$
[A] = \sum \frac{1}{M(ik)} \begin{bmatrix}
1 & 0 & -k^2 & B_1 & \cdots & B_4 \\
0 & k^2 & 0 & B_1 b_1 & \cdots & B_4 b_4 \\
-k^2 & 0 & k^4 & -k^2 B_1 & \cdots & -k^2 B_4 \\
B_1 & B_1 b_1 & -k^2 B_1 & B_1 B_1 (1 + \frac{b_1 b_1}{k^2}) & \cdots & B_1 B_4 (1 + \frac{b_1 b_4}{k^2}) \\
B_2 & B_2 b_2 & -k^2 B_2 & B_2 B_1 (1 + \frac{b_2 b_1}{k^2}) & \cdots & B_2 B_4 (1 + \frac{b_2 b_4}{k^2}) \\
B_3 & B_3 b_3 & -k^2 B_3 & B_3 B_1 (1 + \frac{b_3 b_1}{k^2}) & \cdots & B_3 B_4 (1 + \frac{b_3 b_4}{k^2}) \\
B_4 & B_4 b_4 & -k^2 B_4 & B_4 B_1 (1 + \frac{b_4 b_1}{k^2}) & \cdots & B_4 B_4 (1 + \frac{b_4 b_4}{k^2})
\end{bmatrix}
$$

$$
[B] = \sum \frac{1}{M(ik)} \left[F \; , \;\; kG \; , \;\; -k^2 F \; , \;\; F B_1 + \frac{G B_1 b_1}{k} \; , \;\; \cdots \; , \;\; F B_4 + \frac{G B_4 b_4}{k} \right]^T ,
$$

$$
\{x\} = \{ \; A_0 \; , \; A_1 \; , \; A_2 \; , \; \cdots \; , \; A_6 \; \} \; ,
$$

and

$$
B_j = \frac{k^2}{k^2 + b_j^2} \; , \; j = 1 \;\; to \; 4 \; .
$$

These equations can be solved quite easily by readily available computer subroutines, for matrix manipulation and eigenvalue analysis.

The objective error function is calculated by considering the approximate function, $Q(ik)$, and the actual tabular data, $Q(ik)$. Thus, given a set of lag terms, b_l, ($l = 1$ *to* 4), the coefficients A_j, ($j = 0$ *to* 6), can be found. The values for the *linear coefficients* obtained will result in a minimum error, but it will not be an optimal. There exists a particular set of values for the nonlinear lag terms that will result in an overall optimized minimal error function; this must be found as follows.

Nonlinear Coefficients : b_1, b_2, b_3, b_4

The method employed to solve for the optimal nonlinear lag terms is a sequential simplex method (Mead and Nelder 1965; Nelson and Olsson 1975) developed by Nelder and Mead in 1965. This method is simple, robust and requires no derivatives, and hence lends itself quite well to finding the minimum of a nonlinear objective function of more than one independent variable (see Appendix C). The only assumption made is that the function does have a minimum and that the surface is continuous. Its only downfall is that it is not very efficient in the number of function evaluations that it requires. This method requires the function evaluation at $(n+1)$ vertices for a function of n variables. The largest function value is then replaced by a new point that is decided by reflection, extension or contraction onto the space. This process continues until convergence occurs. The simplex procedure has the ability to adapt itself to any given contour, it does not get easily fooled by a possible dead end. The beauty of the method is that it approaches the function minimum by moving away from the higher function values (see Appendix C for more details).

This method is quite self-contained (FORTRAN code: Olson 1974; O'Neil 1971), requiring the user to supply only the objective (error) function and the initial start and step values. This method also has the ability to consider self imposed boundary conditions. The sequential simplex procedure is a direct method that requires no derivatives of the objective function. In this study, since the objective function is quite complicated, it would be very difficult to supply such derivatives.

Powell's method may also be considered as an attractive alternative to the simplex method. This is a direction set method that produces N mutually conjugate directions to decide on its path of solution. Problems with large computational burdens involving a large number of function evaluations will surely be faster using Powell's method, but the simplex adds an added robustness that seems to be lacking in Powell's method (Fletcher and Powell 1963; Flannery, Press, Teukolsky and Vetterling 1986). Powell's method has a tendency to give incorrect values if initial guess values result in a simplex that covers more than one valley.

With any method, there is still a chance of false convergence. The results obtained can be checked by choosing different starting points and repeating the process, or/and by letting the process continue after the initial convergence to see if it is indeed a minimum.

Boundary Conditions

The rational function used to approximate any given tabular set of aerodynamic data must obey several important boundary conditions. In order to be solved analytically, none of the lag terms can be equivalent to one another. If two lag terms are equal, the resulting equation $\tilde{Q}(ik)$ could be reduced to a six-term equation versus the original seven-term equation. The lag terms must also be greater than zero (non-negative) to ensure system stability (Poirel 1988). This forces the poles of the resulting function to be in the left half-plane of the Laplace domain. If poles are found to be in the right half-plane, this implies an unstable governing function. The lag terms usually take on values in the range of reduced frequencies over which tabular data is available. Note, that in previous work, before these terms were optimized, they were chosen arbitrarily over this given reduced frequency range. Boundary conditions can be handled quite easily in the computer algorithm by replacing an incorrect (or undesirable) value with a new (allowable) one, i.e. a negative number, with, say, 0.00001.

Constraints

It is often required to place constraints on the approximating function. For example, constraints could be imposed at the reduced frequency k=0.0, forcing the approximating rational function to agree exactly with the tabular data at this point. This constraint results in a better fit at k=0.0 (an exact fit), but such a condition leads to an overall poorer fit. It may also be desirable to place constraints at a reduced frequency close to the expected flutter frequency. However, any imposed constraints lead to a loss in the degrees-of-freedom for the least-squares solution. In order to obtain an accurate fit to the given tabular data with imposed constraints, it is necessary to choose an adequate order of the approximating function. An increase in the order of the function (increase in number of lag terms) will definitely give better results, but with an increase in computational time.

An analysis was considered to determine the optimal number of lag terms required to give the desired accuracy with a compatible amount of computational time. This was done by looking at the approximation error obtained by the rational function, and by comparing flutter results obtained from the Laplace method to those obtained from the $U - g$ and $p - k$ analysis; both for the incompressible and transonic regime. If necessary, constraints are applied to the aerodynamic derivatives at a reduced frequency of $k = 0.0$, in hope of achieving improved results.

6.3 Aeroelastic Equations in the Laplace Domain

In this section the equations of motion for both incompressible and transonic flow will be transformed into a general Laplace domain solution (see Table 2) of the following form

$$([M]\,s^2 + [C]\,s + [K] + x\,[N])\tilde{z}(s) = 0 \ , \tag{76}$$

where $[M]$, $[C]$ and $[K]$ are the mass matrix, damping matrix and stiffness matrix, respectively. The matrix $[N]$ consists of all the terms that are a function of reduced frequency, k, and is approximated by the Nasa-Langley rational function. The constant value x is simply a factor-multiplier that depends on the flow regime, incompressible or transonic. The term $\tilde{z}$ is the laplace operator of z where

$$z = \{\ \xi \quad \alpha\ \}^T \ .$$

The approximation matrix $[N]$ given by

$$[N] = \left[\tilde{Q}(p)\right] = [A_0] + [A_1]\,p + [A_2]\,p^2 +$$

$$[A_3]\,\frac{p}{p+b_1} + [A_4]\,\frac{p}{p+b_2} + [A_5]\,\frac{p}{p+b_3} + [A_6]\,\frac{p}{p+b_4} \ . \tag{77}$$

can be easily written in the Laplace domain by using the expression for reduced frequency

$$k = \frac{wb}{U} \ ,$$

along with

$$s = iw \ \ and \ \ p = ik \ ,$$

to give

$$p = \frac{c}{2U}s = \frac{b}{U}s \ . \tag{78}$$

The approximate rational function in Laplace form is then

$$\left[\bar{Q}(s)\right] = [A_0] + \frac{b}{U}[A_1]\, s + (\frac{b}{U})^2 [A_2]\, s^2 +$$

$$[A_3]\frac{s}{s+\frac{U}{b}b_1} + [A_4]\frac{s}{s+\frac{U}{b}b_2} + [A_5]\frac{s}{s+\frac{U}{b}b_3} + [A_6]\frac{s}{s+\frac{U}{b}b_4}\ . \tag{79}$$

6.3.1 Incompressible Aerodynamics

The general equations of motion of a two degree-of-freedom airfoil (Eq. 33,34) are given by

$$\ddot{\xi} + x_\alpha\ddot{\alpha} + w_h^2\xi = \frac{Q_h}{mb}\ ,$$

$$r_\alpha^2\ddot{\alpha} + x_\alpha\ddot{\xi} + w_\alpha^2 r_\alpha^2\alpha = \frac{Q_\alpha}{mb^2}\ .$$

For the particular case of incompressible aerodynamics these can be written as

$$\ddot{\xi}(1+\frac{1}{\mu}) + \dot{\xi}(\frac{2C(k)U}{b\mu}) + \xi(w_h^2) +$$

$$\ddot{\alpha}(x_\alpha - \frac{a}{\mu}) + \dot{\alpha}\left(\frac{U}{b\mu} + 2C(k)(\frac{1}{2}-a)\frac{U}{b\mu}\right) + \alpha(\frac{2C(k)U^2}{b^2\mu}) = 0\ , \tag{80}$$

and

$$\ddot{\xi}(x_\alpha - \frac{a}{\mu}) + \dot{\xi}\left(-2(a+\frac{1}{2})C(k)\frac{U}{b\mu}\right) + \ddot{\alpha}(r_\alpha^2 + \frac{\frac{1}{8}+a^2}{\mu}) +$$

$$\dot{\alpha}\left(\frac{U}{b\mu}(\frac{1}{2}-a) - 2(a+\frac{1}{2})(\frac{1}{2}-a)C(k)\frac{U}{b\mu}\right) + \alpha\left(r_\alpha^2 w_\alpha^2 - 2(a+\frac{1}{2})C(k)\frac{U^2}{b^2\mu}\right) = 0\ . \tag{81}$$

Rewriting these equations, by taking all terms that are a function of $C(k)$, to the right hand side gives

$$\ddot{\xi}(1+\frac{1}{\mu}) + \xi(w_h^2) + \ddot{\alpha}(x_\alpha - \frac{a}{\mu}) + \dot{\alpha}(\frac{U}{b\mu}) \ =$$

$$\dot{\xi}(-\frac{2C(k)U}{b\mu}) + \dot{\alpha}\left(-2C(k)(\frac{1}{2}-a)\frac{U}{b\mu}\right) + \alpha(\frac{-2C(k)U^2}{b^2\mu})\ , \tag{82}$$

and

$$\ddot{\xi}(x_\alpha - \frac{a}{\mu}) + \ddot{\alpha}(r_\alpha^2 + \frac{\frac{1}{8}+a^2}{\mu}) + \dot{\alpha}\left(\frac{U}{b\mu}(\frac{1}{2}-a)\right) + \alpha(r_\alpha^2 w_\alpha^2) =$$

$$\dot{\xi}\left(2(a+\frac{1}{2})C(k)\frac{U}{b\mu}\right) + \dot{\alpha}\left(2(a+\frac{1}{2})(\frac{1}{2}-a)C(k)\frac{U}{b\mu}\right) + \alpha\left(2(a+\frac{1}{2})C(k)\frac{U^2}{b^2\mu}\right) . \quad (83)$$

The right hand side of both these equations is approximated by the Nasa-Langley expression (Eq. 79) in matrix form.

The Laplace transform of Equations (82) and (83), allowing for the initial conditions

$$\xi(0) = 0 \qquad \alpha(0) = 0$$

$$\dot{\xi}(0) = 0 \qquad \dot{\alpha}(0) = 0 \ , \quad (84)$$

gives

$$([M]\,s^2 + [C]\,s + [K] + [D]\,s + [E])\tilde{z}(s) = 0 \ . \quad (85)$$

where $\tilde{z}$ is the Laplace operator of z where

$$z = \{\ \xi \quad \alpha\ \}^T \ ,$$

and

$$[D] = 2\left(\frac{1}{\mu}(\frac{U}{b})\right)C(k)\begin{bmatrix} 1 & (\frac{1}{2}-a) \\ -(a+\frac{1}{2}) & -(a+\frac{1}{2})(\frac{1}{2}-a) \end{bmatrix} , \quad (86)$$

$$[E] = 2\left(\frac{1}{\mu}(\frac{U}{b})^2\right)C(k)\begin{bmatrix} 0 & 1 \\ 0 & -(a+\frac{1}{2}) \end{bmatrix} , \quad (87)$$

An approximation is required for $[D]\,s + [E]$.

$$\bar{Q}(s) = [D]\,s + [E] \quad (88)$$

By using the following substitution

$$s = \frac{U}{b} p = \frac{U}{b} ik$$

the approximation can be written as a single matrix $\left[\bar{N}\right]$, that is a function of $C(k)$ and $ikC(k)$.

$$\left[\bar{N}\right] = [D] \frac{U}{b} ik + [E] \ .$$

where

$$\left[\bar{N}\right] = 2 \left(\frac{1}{\mu} (\frac{U}{b})^2 \right) C(k) \begin{bmatrix} ik & (\frac{1}{2} - a)ik + 1 \\ -(a + \frac{1}{2})ik & -(a + \frac{1}{2})(\frac{1}{2} - a)ik - (a + \frac{1}{2}) \end{bmatrix} \ . \qquad (89)$$

or

$$\left[\bar{N}\right] = x \left[N\right] \ . \qquad (90)$$

where

$$x = 2 \left(\frac{1}{\mu} (\frac{U}{b})^2 \right) \ ,$$

and

$$[N] = C(k) \begin{bmatrix} ik & (\frac{1}{2} - a)ik + 1 \\ -(a + \frac{1}{2})ik & -(a + \frac{1}{2})(\frac{1}{2} - a)ik - (a + \frac{1}{2}) \end{bmatrix} \ . \qquad (91)$$

Thus, the Laplace transform of the equations of motion in the general form (Eq. 76), are given by

$$([M]\, s^2 + [C]\, s + [K] + x\, [N]) \tilde{z}(s) = 0 \ .$$

6.3.2 Transonic Aerodynamics

For the particular case of transonic aerodynamics, equations (33) and (34), may conveniently be written as

$$\ddot{\xi} + x_\alpha \ddot{\alpha} + w_h^2 \xi = -\frac{qc}{mb}(C_{l_h}\frac{\xi}{2} + C_{l_\alpha}\alpha) \ ,$$

$$r_\alpha^2 \ddot{\alpha} + x_\alpha \ddot{\xi} + w_\alpha^2 r_\alpha^2 \alpha = \frac{qc^2}{mb^2}(C_{m_h}\frac{\xi}{2} + C_{m_\alpha}\alpha) \ .$$

Allowing for the same initial conditions as used for the incompressible aerodynamic example, the transonic equations of motion (see Table 2) can be written in the Laplace domain as

$$([M]\, s^2 + [C]\, s + [K] + x\,[N])\tilde{z}(s) = 0 \ .$$

where

$$x = \frac{1}{\pi}\left(\frac{1}{\mu}(\frac{U}{b})^2\right) \ ,$$

and

$$[N] = \begin{bmatrix} C_{l_h}/2 & C_{l_\alpha} \\ -C_{m_h} & -2C_{m_\alpha} \end{bmatrix} \ . \tag{92}$$

Laplace General Solution

$$([M]\, s^2 + [C]\, s + [K] + x\, [N])\bar{z}(s) = 0$$

Incompressible	Transonic
$[M]$ $\begin{bmatrix} 1+\frac{1}{\mu} & x_\alpha - \frac{a}{\mu} \\ x_\alpha - \frac{a}{\mu} & r_\alpha^2 + (\frac{1}{8}+a^2)/\mu \end{bmatrix}$	$[M]$ $\begin{bmatrix} 1 & x_\alpha \\ x_\alpha & r_\alpha^2 \end{bmatrix}$
$[C]_{aerodynamic}$ $\begin{bmatrix} 0 & \frac{1}{\mu}\frac{U}{b} \\ 0 & (\frac{1}{2}-a)\frac{1}{\mu}\frac{U}{b} \end{bmatrix}$	$[C]_{aerodynamic}$ $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$
$[C]_{structural}$ $\begin{bmatrix} 2\zeta_h w_h & 0 \\ 0 & 2r_\alpha^2\zeta_\alpha w_\alpha \end{bmatrix}$ $\zeta_h, \zeta_\alpha \approx 0.01$	$[C]_{structural}$ $\begin{bmatrix} 2\zeta_h w_h & 0 \\ 0 & 2r_\alpha^2\zeta_\alpha w_\alpha \end{bmatrix}$ $\zeta_h, \zeta_\alpha \approx 0.01$

note: $[C] = [C]_{aerodynamic} + [C]_{structural}$

Table 2 (continued)

Incompressible	Transonic
$[K]$ $\begin{bmatrix} w_h^2 & 0 \\ 0 & r_\alpha^2 w_\alpha^2 \end{bmatrix}$	$[K]$ $\begin{bmatrix} w_h^2 & 0 \\ 0 & r_\alpha^2 w_\alpha^2 \end{bmatrix}$
x $2\left(\frac{1}{\mu}(\frac{U}{b})^2\right)$	x $\frac{1}{\pi}\left(\frac{1}{\mu}(\frac{U}{b})^2\right)$
$[N]$ $C(k)\begin{bmatrix} ik & (\frac{1}{2}-a)ik+1 \\ -(a+\frac{1}{2})ik & (a^2-\frac{1}{4})ik-(a+\frac{1}{2}) \end{bmatrix}$ *approximation of* $C(k)$, $C(k)ik$	$[N]$ $\begin{bmatrix} C_{l_h}/2 & C_{l_\alpha} \\ -C_{m_h} & -2C_{m_\alpha} \end{bmatrix}$ *approximation of* C_{l_h}, C_{l_α}, C_{m_h}, C_{m_α}

Table 2 Laplace Solution : Matrices in Laplace Form for Incompressible and Transonic Flow

6.4 Solution of the Aeroelastic Equations in the Laplace Domain

The solution to the aeroelastic equations of motion in the Laplace domain follows the work done previously by Abel (1979). Consider the general form of the aeroelastic equations of motion

$$([M]\,s^2 + [C]\,s + [K] + x\,[N])z(s) = 0 \; .$$

where

$$[N] = [A_0] + \frac{b}{U}\,[A_1]\,s + (\frac{b}{U})^2\,[A_2]\,s^2 +$$

$$[A_3]\,\frac{s}{s+\frac{U}{b}b_1} + [A_4]\,\frac{s}{s+\frac{U}{b}b_2} + [A_5]\,\frac{s}{s+\frac{U}{b}b_3} + [A_6]\,\frac{s}{s+\frac{U}{b}b_4} \; .$$

This gives

$$([M]\,s^2 + [C]\,s + [K] + x\{[A_0] + \frac{b}{U}\,[A_1]\,s + (\frac{b}{U})^2\,[A_2]\,s^2 +$$

$$[A_3]\,\frac{s}{s+\frac{U}{b}b_1} + [A_4]\,\frac{s}{s+\frac{U}{b}b_2} + [A_5]\,\frac{s}{s+\frac{U}{b}b_3} + [A_6]\,\frac{s}{s+\frac{U}{b}b_4}\})z(s) = 0 \; . \qquad (93)$$

Equation (93) is then eliminated of its denominator terms to obtain the following polynomial

$$(B_0\left[\left[\bar{A}\right]s^2 + \left[\bar{B}\right]s + \left[\bar{C}\right]\right] + B_1\left[\bar{D}\right] + B_2\left[\bar{E}\right] + B_3\left[\bar{F}\right] + B_4\left[\bar{G}\right])z(s) = 0 \; . \qquad (94)$$

where

$$B_0 = (s + \frac{U}{b}b_1)(s + \frac{U}{b}b_2)(s + \frac{U}{b}b_3)(s + \frac{U}{b}b_4) \; ,$$

$$B_1 = s(s + \frac{U}{b}b_2)(s + \frac{U}{b}b_3)(s + \frac{U}{b}b_4) \; ,$$

$$B_2 = s(s + \frac{U}{b}b_1)(s + \frac{U}{b}b_3)(s + \frac{U}{b}b_4) \; ,$$

$$B_3 = s(s + \frac{U}{b} b_1)(s + \frac{U}{b} b_2)(s + \frac{U}{b} b_4) \ .$$

$$B_4 = s(s + \frac{U}{b} b_1)(s + \frac{U}{b} b_2)(s + \frac{U}{b} b_3) \ ,$$

$$\left[\bar{A}\right] = [M] + x(\frac{b}{U})^2 [A_2] \ ,$$

$$\left[\bar{B}\right] = [C] + x\frac{b}{U} [A_1] \ ,$$

$$\left[\bar{C}\right] = [K] + x [A_0] \ ,$$

$$\left[\bar{D}\right] = x [A_3] \ ,$$

$$\left[\bar{E}\right] = x [A_4] \ ,$$

$$\left[\bar{F}\right] = x [A_5] \ ,$$

$$\left[\bar{G}\right] = x [A_6] \ .$$

Finally, this may be written as a 6^{th} order polynomial in s of the following form

$$[F_6]\, s^6 + [F_5]\, s^5 + [F_4]\, s^4 + \cdots + [F_1]\, s + [F_0] = 0 \ . \tag{95}$$

where the coefficients $[F_6]$, $[F_5]$, $[F_4]$, $\cdots$, $[F_0]$ are functions of dynamic pressure (velocity).

This can be easily placed in the form of a typical eigenvalue problem

$$s\{x\} = [A]\{x\} \ , \tag{96}$$

to give a series of $6n$ first order equations, where

$$\{x\} = \{\ s^5,\ s^4,\ s^3,\ s^2,\ s^1,\ s^0\ \} \tag{97}$$

and

$$[A] = \begin{bmatrix} -[F_6]^{-1}[F_5] & -[F_6]^{-1}[F_4] & \cdots & -[F_6]^{-1}[F_1] & -[F_6]^{-1}[F_0] \\ [I] & [0] & \cdots & [0] & [0] \\ [0] & [I] & \cdots & [0] & [0] \\ [0] & [0] & \cdots & [0] & [0] \\ [0] & [0] & \cdots & [I] & [0] \end{bmatrix}$$

Formulating the equations of motion into a typical eigenvalue problem results in a robust efficient method of calculating the critical flutter speeds. The flutter characteristics are found by calculating the complex eigenvalues of $[A]$ at various values of dynamic pressure (velocity) at a specific Mach number. The eigenvalues are the roots of the characteristic equation. The roots of the given problem will be a combination of complex conjugate pairs and real numbers

$$\beta \pm iw ,$$

$$\beta . \tag{98}$$

where w and β are the frequency and damping, respectively. A 6^{th} order degree polynomial will produce 12 roots. Of these, eight roots can be attributed to the dimension of the rational approximating function, and the remaining four roots are due to the aeroelastic modes of the airfoil. For stability, all the real parts of the roots, β, must be negative in sign. A positive real part denotes an instability, flutter. Flutter occurs when at a given velocity the frequency of the two complex conjugates begin to approach one another (coalesce). When one of the real parts, β, changes sign, flutter has occurred.

7 Results : Rational Function Optimization

The validity of the Laplace solution for the aeroelastic equations depends greatly on how well the unsteady aerodynamics can be approximated. It is critical that the approximation be very close to the actual tabular data. Each approximant curve must follow, almost identically, every dip and twist of the original function, and give an adequate representation of both the real and imaginary parts of the aerodynamics - as a function of the reduced frequency, k. If insufficient agreement is obtained the end solution will be erroneous. If an accurate approximation of the unsteady aerodynamics can be found, the solution of the equations in the Laplace domain will eliminate the need for costly iteration, that is characteristic of the traditional $U - g$ and $p - k$ methods.

The unsteady aerodynamics for both the incompressible and transonic regime will be approximated by rational functions. The incompressible aerodynamics, found as a function of the Theodorsen Function, $C(k)$ and $C(k)ik$, should not lead to much difficulty in finding accurate approximating functions, since the aerodynamics vary in a fairly smooth manner with reduced frequency. However, problems may be encountered in the transonic regime, where the unsteady transonic aerodynamic coefficients C_{l_h}, C_{l_α}, C_{m_h} and C_{m_α} are very irregular, demonstrating much twisting as reduced frequency is changed. At a transonic Mach number of $M = 0.85$, the aerodynamics are very irregular and should prove the most difficult to approximate. Thus, in order for the Laplace method to be viable, the approximations of these tabular unsteady aerodynamics cannot be considered lightly.

In order for the overall solution of the aeroelastic equations of motion to be robust, the unsteady aerodynamics should cover an adequate reduced frequency range. The governing criteria for such an analysis is that the largest frequency generated from the eigenvalue analysis should fall within the given range of frequency for the unsteady aerodynamic data. The tabular data for the unsteady aerodynamics has a lower value reduced frequency of $k = 0.0$, and the upper value can be checked by

$$k_{upper} = \frac{wb}{U} . \tag{99}$$

where b is the semi-chord and w is the largest frequency generated from the eigenvalue analysis at the specific velocity U.

For incompressible aerodynamics, approximations of both the Theodorsen function, $C(k)$, and the Theodorsen function times the reduced frequency, $C(k)/k$ are required. Initially, the optimization procedure, which used a combined least square methodology and a simplex method, considered only the Theodorsen function, $C(k)$. This was undertaken in order to answer three fundamental questions. Firstly, the validity of the approximating function needed to be ascertained. Could the Nasa-Langley rational function lead to a valid approximation of the unsteady aerodynamics, and hence lead to a possible successful solution to the aeroelastic equations? Secondly, it was required to determine how many lag terms were needed to ensure an acceptable amount of error. Lastly, a study of the reduced frequency range was required to determine what range could be approximated successfully.

A valid approximation of the Theodorsen function, $C(k)$, over the reduced frequency range, $0.0 < k < 10.0$, was achieved by utilizing a Nasa-Langley rational function, consisting of four lag terms. The calculated optimized linear terms were given by

$A_0 = 0.99858$, $A_1 = -0.000078$, $A_2 = 0.000012$, $A_3 = -0.040125$,

$A_4 = -0.152297$, $A_5 = -0.22708$, $A_6 = -0.078005$,

and the lag terms as

$b_1 = 0.014919$, $b_2 = 0.080715$, $b_3 = 0.238540$, $b_4 = 0.687273$,

with a calculated error of 8.207×10^{-6}.

It was determined that the approximation error to $C(k)$ decreased quite significantly with the increase in the number of lag terms, as shown in the table below.

Error	*# of lag terms*
0.02025	1
0.0009360	2
0.00008485	3
0.000008207	4

The Nasa-Langley rational function, incorporating four lag terms, gave an excellent approximation to the Theodorsen function over the entire reduced frequency range. This same methodology was then used to approximate $C(k)$ and $C(k)ik$ with great success. It should be noted that in order to simplify the subsequent analysis of the aeroelastic equations, $C(k)$ and $C(k)ik$ were forced to have the same lag terms.

Again, considering the same reduced frequency range, $0.0 < k < 10.0$, the calculated optimized linear terms for $C(k)$ and $C(k)ik$, respectively, were

$A_0 = 0.99828$, $A_1 = -0.000035$, $A_2 = 0.0000047$, $A_3 = -0.011201$,

$A_4 = -0.162893$, $A_5 = -0.22909$, $A_6 = -0.0645937$,

and

$A_0 = -0.00001355$, $A_1 = 0.50037475$, $A_2 = -0.00003629$, $A_3 = 0.00061076$,

$A_4 = 0.01380489$, $A_5 = 0.05848218$, $A_6 = 0.04963449$,

with the lag terms,

$b_1 = 0.015441$, $b_2 = 0.084286$, $b_3 = 0.255951$, $b_4 = 0.765114$,

with a calculated error of 1.178×10^{-5}.

The Nasa-Langley rational function approximation with multi-level constraints and nonlinear term optimization proved to be quite useful. In the incompressible regime, the approximation error of the unsteady aerodynamics for $C(k)$ and $C(k)ik$ was found to be extremely small. Excellent agreement can be seen by referring to either the optimization table comparison (see Appendix D) or the graphs of $C(k)$ and $C(k)ik$ (see Figures 4a-1 to 4a-4 and Figures 4b-1 to 4b-4). Each comparison of the aerodynamic functions is given as two graphs, the real component and the imaginary component versus the reduced frequency. The choice of identical lag terms for the approximation of $C(k)$ and $C(k)ik$, allows for a more efficient and robust solution of the aeroelastic equations in the Laplace domain. The given approximation to the unsteady aerodynamics gave useful results to the solution of the flutter problem, as will be seen in the next section.

In the transonic regime, approximations of four aerodynamic coefficients, namely,

C_{l_h}, C_{l_α}, C_{m_h} and C_{m_α}, were required. This inevitably resulted in an increased approximation error. To make matters worse, the transonic aerodynamics are very irregular. This resulted in a slight modification to the way in which the unsteady aerodynamics were approximated. A decreased reduced frequency range was considered, and an analysis was undertaken to study the effect of imposed constraints at the reduced frequency, $k = 0.0$.

Initially, an approximation of the unsteady aerodynamics for the transonic regime at $M = 0.85$ was considered and a reduced frequency range, $0.025 \leq k \leq 1.0$ was studied. The optimization of the aerodynamics appeared to be quite good, as can be seen from Figures 5a-1 to 5d-1; however, these approximations resulted in erroneous solutions for the flutter velocity when employed in the Laplace solution. The values of the unsteady aerodynamics at low values of reduced frequency, particularly in the interval of $0.0 \leq k \leq 0.025$, seemed to be of utmost importance in the analysis of the aeroelastic equations. If no consideration is given to the aerodynamics at the low reduced frequency range, the approximation procedure may give values of the aerodynamics at a value of k, that have no physical significance; i.e., a negative value of C_{l_h} or C_{m_h} at $k = 0.0$ when they should equal 0.0, or extraordinary large values for the aerodynamics at $k = 0.0$. These problems are related to the value chosen for the constant A_0 in the rational function approximation. Note that the value of each aerodynamic derivative at the $k = 0.0$ condition is equal the value of A_0 (see Appendix E).

To solve these problems the original reduced frequency range of $0.025 \leq k \leq 1.0$, had to be changed to include the reduced frequency of $k = 0.0$. Hence, a reduced frequency range of $0.0 \leq k \leq 1.0$ was considered, as well as a decreased range of $0.0 \leq k \leq 0.5$.

Imposed constraints were considered at $k = 0.0$ for all four aerodynamic coefficients, and as well as just for those that were zero at $k = 0.0$, namely C_{l_h} and C_{m_h}. All approximations of the unsteady aerodynamics resulted in functions which gave valid solutions to the aeroelastic equations in the Laplace method, as will be seen in the next section. The optimization results for the different reduced frequency ranges, and the imposed constraints are presented in Table 3. These results, for the transonic regime, were not as good as those for the incompressible regime, but were still found to be excellent overall. Note, that the requirements for the approximation of the unsteady transonic aerodynamics were much more demanding than for the incompressible aerodynamics, as can be seen from the calculated error.

Reduced Frequency Range, k	*Constraints at* $k = 0.0$	*Error*
$0.0 \leq k \leq 1.0$	$C_{l_h}, C_{l_\alpha}, C_{m_h}, C_{m_\alpha}$	0.009355
$0.0 \leq k \leq 1.0$	C_{l_h}, C_{m_h}	0.003602
$0.0 \leq k \leq 0.5$	$C_{l_h}, C_{l_\alpha}, C_{m_h}, C_{m_\alpha}$	0.004183
$0.0 \leq k \leq 0.5$	C_{l_h}, C_{m_h}	0.001396

Table 3 Transonic Regime : Comparison of Rational Function Optimization Error for Various Reduced Frequency Ranges and Imposed Constraints, $M = 0.85$.

A reduced frequency range of $0.0 \leq k \leq 0.5$, with imposed constraints at $k = 0.0$ for the unsteady aerodynamics, C_{l_h} and C_{m_h}, at $M = 0.85$, gave the minimum approximation error, and the best solution to the flutter problem, as will be shown in the next section. Excellent agreement can be seen by referring to either the optimization table comparison (see Appendix F) or the graphs of the transonic aerodynamic coefficients (see Figures 5a-2 to 5d-2). Each comparison is presented as two graphs, the real and imaginary components as functions of the reduced frequency. The decreased reduced frequency range resulted in a less robust total test range; thus, in the solution of the aeroelastic equations, some results obtained at low values of velocity were somewhat erratic. Thus a check was required to determine if the instability was totally within the accepted confines of the given unsteady tabular data range.

Applying the same reduced frequency range, $0.0 \leq k \leq 0.5$, with forced constraints at $k = 0.0$ for C_{l_h} and C_{m_h}, the approximations to the unsteady aerodynamics for $M = 0.80$ and $M = 0.875$ were also found to be very good. These results are graphically presented in Figures 6a to 6d and Figures 7a to 7d, respectively.

It should be noted that when approximations were made for the unsteady aerodynamics, a complete global error was considered. Hence, for the incompressible regime, the coefficients for the rational function were calculated simultaneously considering both $C(k)$ and $C(k)ik$. The approximations were found such that they resulted in a minimum total calculated error for the same nonlinear coefficients. The transonic regime is the most interesting, since a global minimization of the four aerodynamic coefficients, C_{l_h}, C_{l_α}, C_{m_h} and C_{m_α} was performed. It should be noted, once more, that this global approximation error approach was considered at $M = 0.85$, where the nonlinearities are the most pronounced, and still gave adequate results.

The Nasa-Langley solution for approximating unsteady aerodynamics gave excellent results for both the incompressible and transonic regime. This method demonstrates comparably better results to previously used methods with less overall computational time and complexity (Poirel 1988).

8 Results : Laplace Method of Solution for the Aeroelastic Equations

The rational functions obtained for incompressible flow were substituted into the two-degree-of-freedom airfoil equations. The corresponding Laplace eigenvalue problem was solved, producing twelve eigenvalues (see Appendix G). Of these, eight were completely real, attributed to the dimension of the rational approximating function, and the remaining four eigenvalues produced two complex conjugate pairs, due to the aeroelastic modes of the airfoil. The $p-k$ solution generated four roots, which formed two complex conjugate pairs. This allowed for the direct comparison of the $p-k$ solution with the Laplace method, when the real roots were omitted. The $U-g$ method produced only two roots, which resembled those of the positive frequency conjugate roots. Various case studies were considered, incorporating the various methods of flutter solution.

For example, each method was used to solve the equations for the following set of parameters; $\mu = 50.0$, $r_\alpha = 0.5$, $a = -0.5$ and $w_h/w_\alpha = 0.2$ (Case Study 1). The results are presented as graphs of frequency ratio, w/w_α, and damping ratio, versus non-dimensional velocity, $\bar{U}$ (see Figures 8a-1 to 8a-3). All the methods resulted in exactly the same value for the critical flutter velocity, $\bar{U} = 4.53$ (see table 4); also, all the methods gave similar values of frequency ratio, w/w_α, over the non-dimensional velocity range, $\bar{U}$. However, the $U-g$ method showed some variance in the damping ratio compared to either the $p-k$ or Laplace methods. A second example was considered, which led to similar results. In this example, ($\mu = 50.0$, $r_\alpha = 0.6$, $a = -0.6$ and $w_h/w_\alpha = 0.4$ (Case Study 2)), all methods produced a flutter velocity, $\bar{U}$, of approximately 5.10. This example, showed similar response behaviour in frequency and damping ratio versus non-dimensional velocity as in the previous one (see Figures 8b-1 to 8b-3). Other examples, showing comparisons of critical flutter velocity, obtained from the various methods of solution are shown in Table 4. It should be noted that the $U-g$ method encountered some numerical difficulty in some of the cases studied, namely case studies 4 and 5. The $p-k$ and the Laplace methods had no such problems.

Case					*Flutter Velocity* $\bar{U}$		
Study	μ	r_α	a	$w_h \,/\, w_\alpha$	$U-g$	$p-k$	*Laplace*
1	50.0	0.5	-0.5	0.2	4.53	4.53	4.53
2	50.0	0.6	-0.6	0.4	5.10	5.10	5.11
3	100.0	0.5	-0.5	0.2	6.24	6.26	6.26
4	75.0	0.4	-0.4	0.3	– ? –	3.68	3.68
5	100.0	0.4	-0.4	0.3	– ? –	4.16	4.16

Table 4 Incompressible Regime : Comparison of critical flutter velocity, $\bar{U}$, using the $U-g$, $p-k$ and *Laplace* methods. The symbol – ? – indicates that it was not possible to determine a flutter velocity for these conditions due to numerical instabilities.

For transonic flow, the aeroelastic equations of motion were also solved utilizing the rational function approximation of the unsteady aerodynamics. The given Laplace eigenvalue solution produced a total of twelve eigenvalues (see Appendix H). Of these, two real roots and three complex conjugate pairs can be attributed to the rational approximating function, and the other two complex conjugate pairs are from the aeroelastic modes of the airfoil. When the aeroelastic eigenvalues were isolated from the other eigenvalues, not always an easy task, comparisons can be made with the $U-g$ and $p-k$ methods.

The traditional methods of solution for the aeroelastic equations were not formulated for transonic flow, but comparisons were made with $U-g$ and $p-k$ data from Lee (1984). The Laplace solution was formulated for transonic flow at a $M = 0.85$, for the parameters: $\mu = 50.0$, $r_\alpha = 0.5$, $a = -0.5$, $\omega_h/\omega_\alpha = 0.2$ and $x_\alpha = 0.25$. This allowed for direct comparison with the $U-g$ and $p-k$ generated results given by Lee (1984).

An initial reduced frequency range of $0.0 \leq k \leq 1.0$, with forced constraints at $k = 0.0$ for all of the aerodynamic derivatives, C_{l_h}, C_{l_α}, C_{m_h} and C_{m_α} was considered. This yielded the following results, $\bar{U} = U/b\omega_\alpha = 3.36$, $\omega/\omega_\alpha = 0.302$ and $k = \omega b/U = 0.090$.

These results did not compare as well as had been hoped with those of Lee (1984), which are $\bar{U} = U/b\omega_\alpha = 3.46$, $\omega/\omega_\alpha = 0.313$ and $k = \omega b/U = 0.090$.

It was concluded that better results could be obtained if the reduced frequency range was decreased to $0.0 \leq k \leq 0.5$. This allowed for a better approximation of the low-range reduced frequency aerodynamics. This region, say from $0.0 < k < 0.1$ seemed to be the most critical in the overall flutter analysis scheme, since the expected critical flutter frequency, $k = 0.090$, fell within this range. As well as considering this decreased reduced frequency range, constraints were placed only on the aerodynamic derivatives, whose magnitude at $k = 0.0$ was 0.0; i.e., C_{l_h} and C_{m_h}. The following results were obtained, $\bar{U} = U/b\omega_\alpha = 3.45$, $\omega/\omega_\alpha = 0.309$ and $k = \omega b/U = 0.090$.

These results agreed favourably with those given by Lee (1984). They are presented graphically in Figures 9a-1 and 9a-2. Figure 9a-1 shows excellent agreement for both the damping and frequency ratio as a function of non-dimensional velocity, $U/b\omega_\alpha$. However, in Figure 9a-2, there is some variance observed in the magnitude of the damping ratio, particularly for values of low non-dimensional velocity. This was due to the imaginary part of the eigenvalue falling out of the considered reduced frequency range, namely, $0.0 \leq k \leq 0.5$, and thus out of the scope of the approximating aerodynamic function. This however, is not so critical; as the root that does go unstable is well behaved, and the other root shows variation from the expected behaviour only at low values of non-dimensional velocity.

For completeness, other results incorporating varying reduced frequency ranges and constraints are included below, in Table 5.

Reduced Frequency Range	*Constraints applied at* $k = 0.0$	U	ω/ω_α	k
$0.0 \leq k \leq 1.0$	$C_{l_h}, C_{l_\alpha}, C_{m_h}, C_{m_\alpha}$	3.36	0.302	0.090
$0.0 \leq k \leq 1.0$	C_{l_h}, C_{m_h}	3.19	0.297	0.093
$0.0 \leq k \leq 0.5$	$C_{l_h}, C_{l_\alpha}, C_{m_h}, C_{m_\alpha}$	3.31	0.302	0.091
$0.0 \leq k \leq 0.5$	C_{l_h}, C_{m_h}	3.45	0.309	0.090

Table 5 Transonic Regime : Comparison of Laplace Solution for Various Reduced Frequency Ranges and Imposed Constraints for $M = 0.85$.

The Laplace Solution yielded the best results, when the approximations of the unsteady aerodynamics considered a reduced frequency range of $0.0 \leq k \leq 0.5$, with forced constraints for C_{l_h} and C_{m_h} at $k = 0.0$. Using these approximations of the unsteady aerodynamics, the Laplace solution was formulated for varying values of non-dimensional airfoil mass, μ, at $M = 0.85$. This allowed for the direct comparison with $U - g$ and $p - k$ results given by Lee (1984) (see Table 6). Excellent agreement was obtained over the entire range of airfoil non-dimensional mass, μ, such that the percentage error between the non-dimensional velocities, $\bar{U}$, obtained using the Laplace method and those given by Lee, was less than 1%.

Further analyses considered the transonic regime for Mach numbers of $M = 0.80$ and $M = 0.875$ at varying values of airfoil non-dimensional mass, μ, (see Table 6). This showed that, although only a small interval range in Mach number was considered, $0.80 < M < 0.875$, the overall behaviour was quite irregular. For example, an analysis of the aeroelastic equations of motion for the Mach numbers of $M = 0.80$, $M = 0.85$ and $M = 0.875$, at an airfoil non-dimensional mass, $\mu = 50.0$, gave critical flutter velocities of $\bar{U} = 3.70$, 3.45 and 6.17, respectively. If the flutter velocities obtained at $M = 0.80$ and $M = 0.875$, had been used to interpolate for the flutter velocity at $M = 0.85$, the actual value of $\bar{U} = 3.45$ would not have been acquired.

μ	$U-g$ / $p-k$ * $\bar{U}$ $M = 0.85$	*Laplace* $\bar{U}$ $M = 0.85$	*Laplace* $\bar{U}$ $M = 0.80$	*Laplace* $\bar{U}$ $M = 0.875$
50.00	3.46	3.45	3.70	6.17
75.00	3.94	3.95	4.33	7.08
100.00	4.33	4.36	4.86	7.54
150.00	4.99	5.01	5.76	7.41
200.00	5.55	5.53	6.53	7.37
250.00	5.99	5.96	7.21	7.35

* reference Lee 1984

Table 6 Transonic Regime : Comparison of critical flutter velocity, $\bar{U}$, for $M = 0.80$, $M = 0.85$ and $M = 0.875$ using the $U - g$, $p - k$ and Laplace methods, with the given constants : $r_\alpha = 0.5$, $a = -0.5$, $\omega_h/\omega_\alpha = 0.2$, $x_\alpha = 0.25$ and $b = 1.0$.

The optimized coefficients of the rational function approximations for $M = 0.80$, $M = 0.85$ and $M = 0.875$ are given in Appendix I. Again, from a comparison of the respective linear and nonlinear coefficient terms, it can be seen how quickly and irregularly the aerodynamics change from one Mach number to another.

The excellent comparison obtained for the flutter condition between the traditional methods and the Laplace method demonstrates the validity of the Laplace method. It also shows the excellent nature of the approximation for the unsteady aerodynamics obtained by utilizing a Nasa-Langley rational function with multi-level constraints and nonlinear term optimization. The main strength of the Laplace method is that the unsteady aerodynamics have only to be approximated once and then can be used in the analysis of many sets of parameters; thus, an iterative type solution is not required. The benefits will soon be realized if many conditions have to be studied. However, with the introduction of this approximating rational function the dimension of the problem is increased considerably, from four to twelve, thus increasing the size of the eigenvalue problem.

9 Active Control for a Two-Dimensional Airfoil in Incompressible Flow

9.1 Equations of Motion Incorporating a Flap Deflection

Consider a three-degree-of-freedom rigid airfoil flexibly mounted undergoing bending and twisting (torsion) motion (see Figure 10). Bending (plunge) is denoted by h, positive downward, the twisting (pitch) about the elastic axis is denoted by α, positive in the clockwise direction, and the flap angle by β, positive in the clockwise direction. The aeroelastic equations of motion (Fung 1955; Ashley and Bisplinghoff 1962), can be derived by summing forces and moments about the elastic axis and moments about the flap hinge line to give

$$m\ddot{h} + m(x_\alpha b)\ddot{\alpha} + m(x_\beta b)\ddot{\beta} = -K_h h - L \ , \tag{100}$$

$$(I + m(x_\alpha b)^2)\ddot{\alpha} + m(x_\alpha b)\ddot{h} + (I_\beta + (c_\beta - a)mb^2 x_\beta)\ddot{\beta} = -K_\alpha \alpha + M_\alpha \ , \tag{101}$$

$$(I_\beta + (c_\beta - a)mb^2 x_\beta)\ddot{\alpha} + m(x_\beta b)\ddot{h} + (I_\beta)\ddot{\beta} = -K_\beta \beta + M_\beta \ , \tag{102}$$

where I_β is the flap mass moment of inertia about the flap hinge line, $x_\beta b$ is the distance from the flap hinge line to the flap centre of mass, $c_\beta b$ is the distance from the midchord to the flap hinge line, ab is the distance from the midchord to the elastic axis (measured positive towards trailing edge), K_β is the torsional spring constant about the flap hinge line, and M_β is the moment acting about the flap hinge line.

Introducing a non-dimensional generalized displacement coordinate, as before,

$$\xi = \frac{h}{b} \tag{103}$$

and making the following substitutions

$$K_h = w_h^2 m \ , \tag{104}$$

$$K_\alpha = w_\alpha^2 I_\alpha \ , \tag{105}$$

$$K_\beta = w_\beta^2 I_\beta \ , \tag{106}$$

$$I_\alpha = r_\alpha^2 m b^2 \ , \tag{107}$$

$$I_\beta = r_\beta^2 m b^2 \ , \tag{108}$$

where r_α and r_β are the radius of gyration about the elastic axis and the flap hinge line, respectively, the complete 3DOF aeroelastic equations of motion become

$$\ddot{\xi} + x_\alpha \ddot{\alpha} + x_\beta \ddot{\beta} + w_h^2 \xi = \frac{-L}{mb} \ , \tag{109}$$

$$r_\alpha^2 \ddot{\alpha} + x_\alpha \ddot{\xi} + (r_\beta^2 + (c_\beta - a) x_\beta) \ddot{\beta} + w_\alpha^2 r_\alpha^2 \alpha = \frac{M_\alpha}{mb^2} \ , \tag{110}$$

$$(r_\beta^2 + (c_\beta - a) x_\beta) \ddot{\alpha} + x_\beta \ddot{\xi} + r_\beta^2 \ddot{\beta} + w_\beta^2 r_\beta^2 \beta = \frac{M_\beta}{mb^2} \ . \tag{111}$$

For incompressible flow, the expressions incorporating a flap deflection for the lift force, pitching moment and the hinge moment are (Theodorsen 1935; Garrick and Theodorsen 1941)

$$L = -\pi \rho b^3 w^2 \left\{ P_w \xi + \left[P_\phi - (\frac{1}{2} + a) P_w \right] \alpha + P_\beta \beta \right\} \ . \tag{112}$$

$$\begin{aligned} M_\alpha = \pi \rho b^4 w^2 \Big\{ & \left[M_w - (\frac{1}{2} + a) P_w \right] \xi + \left[M_\phi - (\frac{1}{2} + a)(P_\phi + M_w) + (\frac{1}{2} + a)^2 P_w \right] \alpha + \\ & \left[M_\beta - (\frac{1}{2} + a) P_\beta \right] \beta \Big\} \ , \end{aligned} \tag{113}$$

and

$$M_\beta = \pi \rho b^4 w^2 \left\{ T_w \xi + \left[T_\phi - (\frac{1}{2} + a) T_w \right] \alpha + T_\beta \beta \right\} \tag{111}$$

The expressions P_w, P_ϕ, P_β, M_w, M_ϕ, M_β, T_w, T_ϕ and T_β are included for reference in Appendix J. They are a function of the Theodorsen function, $C(k)$, velocity, frequency and geometry.

9.2 Flutter Analysis With Active Controls

Flutter-suppression control systems have received increased attention in the last decade (Karpel 1981; Kass and Thompson 1971). They have been found to be an effective way of increasing the critical flutter speed with substantial weight savings. There are two types of flutter-suppression systems that have proved to be quite useful: passive and active flutter control.

Passive Flutter Control

Passive flutter control considers increasing the structural stiffness and/or mass balancing. Proper wing mass distribution can improve the flutter characteristics tremendously. This is so because careful placing of mass has a direct effect on the inertial coupling of the system. Inertial coupling is very often the predominant factor in flutter. Increasing the in structural stiffness (particularly the torsional stiffness) will also result in an increased critical flutter speed. It has been shown that a stiffness increase of factor n will result in an increase in flutter speed of approximately $\sqrt{n}$. However, the addition of stiffness unfortunately means an increase in the total overall aircraft weight.

Active Flutter Control

The type of control that is employed in this study is active flutter-suppression. Active control uses a control surface (flap) which is deflected in response to the wing motion, resulting in a change in the aerodynamic forces and moments on the wing. Such active control has to be very reliable and durable, because the loss of flutter control would probably result in an immediate airfoil failure. Another problem encountered in the use of active control is that of obtaining accurate measurements from the airfoil. Direct measurement is required for the controller to control the control surface via some specified control law. It is also important for the controller to be robust and allow for control at various flight conditions. The benefit of active control is that its implementation will have no significant increase in the airfoil weight.

Writing the equations of motion in the Laplace domain allows for the easy implementation of active control. Consider the equations of motion of a three-degree-of-freedom rigid airfoil incorporating a flap control, β . Neglecting the hinge moment equation, the equations of motion (100) and (101) can be written in matrix form

$$([M]\,s^2 + [C]\,s + [K_{.} + x\,[N])\tilde{z}(s) = 0 \; , \qquad (115)$$

where $[M]$, $[C]$, $[K]$ and $[N]$ are 2×3 matrices, and the term $\tilde{z}$ is the laplace operator of z where

$$z = \{\ \xi \quad \alpha \quad \beta\ \}^T \; .$$

Equation (115) may then be rewritten in the following form

$$([M_s \mid M_c]\,s^2 + [C_s \mid C_c]\,s + [K_s \mid K_c] + x\,[N_s \mid N_c]) \begin{Bmatrix} q \\ q_c \end{Bmatrix} = 0 \; . \qquad (116)$$

where

$$\{q\} = \{\ \xi \quad \alpha\ \}^T\ . \tag{117}$$

$$\{q_c\} = \{\ \beta\ \}\ . \tag{118}$$

The subscripts s and c denote a structural and control quantities, respectively, where $[M_s]$, $[C_s]$ and $[K_s]$ are 2×2 matrices and $[M_c]$, $[C_c]$ and $[K_c]$ are 2×1 matrices.

A control law is considered of the following form

$$\{\ \beta\ \} = [T] \left\{ \begin{array}{c} \xi \\ \\ \alpha \end{array} \right\}\ . \tag{119}$$

where $[T]$ is a transfer-function matrix of size 1×2. This allows the flap angle, β, to be written as a function of the generalized aerodynamic motions, ξ and α.

Applying the control law to (116) gives

$$([M_s]\,s^2 + [C_s]\,s + [K_s] + x\,[N_s])\,\{q\} +$$

$$([M_c]\,s^2 + [C_c]\,s + [K_c] + x\,[N_c])\,[T]\,\{q\} = 0\ . \tag{120}$$

The solution procedure is the same as that employed earlier in section 6.1. Making the following substitutions in (120),

$$[N_s] = [A_0] + \frac{b}{U}\,[A_1]\,s + (\frac{b}{U})^2\,[A_2]\,s^2 +$$

$$[A_3]\,\frac{s}{s+\frac{U}{b}b_1} + [A_4]\,\frac{s}{s+\frac{U}{b}b_2} + [A_5]\,\frac{s}{s+\frac{U}{b}b_3} + [A_6]\,\frac{s}{s+\frac{U}{b}b_4}\ , \tag{121}$$

and

$$[N_c] = [A_{0,c}] + \frac{b}{U}\,[A_{1,c}]\,s + (\frac{b}{U})^2\,[A_{2,c}]\,s^2 +$$

$$[A_{3,c}]\,\frac{s}{s+\frac{U}{b}b_1} + [A_{4,c}]\,\frac{s}{s+\frac{U}{b}b_2} + [A_{5,c}]\,\frac{s}{s+\frac{U}{b}b_3} + [A_{6,c}]\,\frac{s}{s+\frac{U}{b}b_4}\ . \tag{122}$$

gives

$$\begin{aligned}
&([M_s]\,s^2 + [C_s]\,s + [K_s] + x\{[A_0] + \frac{b}{U}\,[A_1]\,s + (\frac{b}{U})^2\,[A_2]\,s^2 + [A_3]\,\frac{s}{s+\frac{U}{b}b_1} + \\
&[A_4]\,\frac{s}{s+\frac{U}{b}b_2} + [A_5]\,\frac{s}{s+\frac{U}{b}b_3} + [A_6]\,\frac{s}{s+\frac{U}{b}b_4}\})\,\{q\} + \\
&([M_c]\,s^2 + [C_c]\,s + [K_c] + x\{[A_{0,c}] + \frac{b}{U}\,[A_{1,c}]\,s + (\frac{b}{U})^2\,[A_{2,c}]\,s^2 + [A_{3,c}]\,\frac{s}{s+\frac{U}{b}b_1} + \\
&[A_{4,c}]\,\frac{s}{s+\frac{U}{b}b_2} + [A_{5,c}]\,\frac{s}{s+\frac{U}{b}b_3} + [A_{6,c}]\,\frac{s}{s+\frac{U}{b}b_4}\})\,[T]\,\{q\} = 0\ .
\end{aligned} \tag{123}$$

The denominators of the above equations are removed to obtain the following polynomial

$$(B_0\left[\left[\bar{A}\right]s^2 + \left[\bar{B}\right]s + \left[\bar{C}\right]\right] + B_1\left[\bar{D}\right] + B_2\left[\bar{E}\right] + B_3\left[\bar{F}\right] + B_4\left[\bar{G}\right])\,\{q\} +$$

$$(B_0\left[\left[\bar{A}_c\right]s^2 + \left[\bar{B}_c\right]s + \left[\bar{C}_c\right]\right] + B_1\left[\bar{D}_c\right] + B_2\left[\bar{E}_c\right] + B_3\left[\bar{F}_c\right] + B_4\left[\bar{G}_c\right])\,[T]\,\{q\} = 0\ ,$$

where

$$\begin{aligned}
B_0 &= (s+\frac{U}{b}b_1)(s+\frac{U}{b}b_2)(s+\frac{U}{b}b_3)(s+\frac{U}{b}b_4)\ , \\
B_1 &= s(s+\frac{U}{b}b_2)(s+\frac{U}{b}b_3)(s+\frac{U}{b}b_4)\ , \\
B_2 &= s(s+\frac{U}{b}b_1)(s+\frac{U}{b}b_3)(s+\frac{U}{b}b_4)\ , \\
B_3 &= s(s+\frac{U}{b}b_1)(s+\frac{U}{b}b_2)(s+\frac{U}{b}b_4)\ , \\
B_4 &= s(s+\frac{U}{b}b_1)(s+\frac{U}{b}b_2)(s+\frac{U}{b}b_3)\ .
\end{aligned}$$

$$\left[\bar{A}\right] = [M] + x(\frac{b}{U})^2\,[A_2]\ ,$$

$$\left[\bar{B}\right] = [C] + x\frac{b}{U}[A_1] \ ,$$

$$\left[\bar{C}\right] = [K] + x[A_0] \ ,$$

$$\left[\bar{D}\right] = x[A_3] \ ,$$

$$\left[\bar{E}\right] = x[A_4] \ ,$$

$$\left[\bar{F}\right] = x[A_5] \ ,$$

$$\left[\bar{G}\right] = x[A_6] \ ,$$

and

$$\left[\bar{A}_c\right] = [M_c] + x(\frac{b}{U})^2[A_{2,c}] \ ,$$

$$\left[\bar{B}_c\right] = [C_c] + x\frac{b}{U}[A_{1,c}] \ ,$$

$$\left[\bar{C}_c\right] = [K_c] + x[A_{0,c}] \ ,$$

$$\left[\bar{D}_c\right] = x[A_{3,c}] \ ,$$

$$\left[\bar{E}_c\right] = x[A_{4,c}] \ ,$$

$$\left[\bar{F}_c\right] = x[A_{5,c}] \ ,$$

$$\left[\bar{G}_c\right] = x[A_{6,c}] \ .$$

This is then expanded to obtain a m^{th} order polynomial in s of the following form

$$[F_m]\,s^m + [F_{m-1}]\,s^{m-1} + [F_{m-2}]\,s^{m-2} + \cdots + [F_1]\,s + [F_0] = 0 \ , \tag{124}$$

where the coefficients $[F_m]$, $[F_{m-1}]$, $[F_{m-2}]$, $\cdots$, $[F_0]$ are functions of dynamic pressure (velocity).

This can be easily placed in a typical eigenvalue problem of the given form

$$s\{x\} = [A]\{x\} \ , \tag{125}$$

to give a series of $m \times n$ first order equations, where

$$\{x\} = \{ \; s^{m-1} \; s^{m-2} \; \cdots \; s^1 \; s^0 \; \} \; . \tag{126}$$

The method of solution continues as shown earlier (see Section 6.4). Thus, for a given transfer function matrix $[T]$, the problem can be analyzed to determine if such a given control law has achieved an increase in the flutter speed.

10 Results : Active Control for Incompressible Flow

The problem associated with applying an active control device to an airfoil, hinges on developing a set of aeroelastic equations, where the form of the unsteady aerodynamics and the control law are compatible. Writing the equations in the Laplace domain is ideally suited to this task. Active control was considered only for the case of incompressible flow. Active control for transonic flow could be considered; however, it would first be necessary to modify the transonic aerodynamics to incorporate a flap.

A control law was considered of the following form

$$\{ \beta \} = [T] \left\{ \begin{array}{c} \xi \\ \\ \alpha \end{array} \right\} . \tag{127}$$

where $[T]$ is the transfer-function matrix of size 1×2. This allows the flap angle, β, to be written as a function of the generalized aerodynamic motions, ξ and α.

The transfer function used was a function of ξ and α and their first derivatives.

$$[T] = [\ t_{11}\ ,\ t_{12}\ ] + \frac{a_T}{\omega_R} s [\ t_{11}^{*}\ ,\ t_{12}^{*}\ ]\ . \tag{128}$$

This is referred to as a *damping type transfer function* by Nissim and Abel (1978), where a_T is a control of the amount of damping introduced by the control surface (flap), and ω_R is a reference frequency, normally taken as the no-control flutter frequency. This transfer function can be considered as a type of proportional-derivative (PD) control.

A proportional-integral-derivative (PID) control was also implemented by adding an integrator term to give a transfer function of the following form

$$[T] = [\ t_{11}\ ,\ t_{12}\ ] + \left(\frac{a_T}{\omega_R}\right)_1 s [\ t_{11}^{*}\ ,\ t_{12}^{*}\ ] + \left(\frac{a_T}{\omega_R}\right)_2 \frac{1}{s} [\ t_{11}^{**}\ ,\ t_{12}^{**}\ ]\ . \tag{129}$$

For better numerical stability in the eigenvalue solution, modified control transfer functions $[T]$ were employed for both PD and PID control (Nissim and Abel 1978).

PD control

$$[T] = [\, t_{11}\,,\; t_{12} \,] + \frac{a_T}{w_R} s \frac{50000}{s + 50000} [\, t_{11}^{*}\,,\; t_{12}^{*} \,] \tag{130}$$

PID control

$$[T] = [\, t_{11}\,,\; t_{12} \,] + \left(\frac{a_T}{w_R}\right)_1 s \frac{50000}{s + 50000} [\, t_{11}^{*}\,,\; t_{12}^{*} \,] + \left(\frac{a_T}{w_R}\right)_2 \frac{1}{s} [\, t_{11}^{**}\,,\; t_{12}^{**} \,] \tag{131}$$

The added term $\frac{50000}{s+50000}$ forces both the structure and the control matrix terms to be a part of the highest order coefficient in s of the polynomial. Without the addition of such a term, the highest order coefficient of the polynomial would be a function of the control matrix terms alone. This results in both very large and very small solutions for the eigenvalues and thus, places undue strain on the numerical accuracy.

Initially the *damping type transfer function* (PD control) was considered for analysis. The values suggested by Abel and Nissim (1978), who had used a theory based on the aerodynamic energy concept, were used as a starting point to determine realistic parameters in the transfer function. A simplex method was then used to determine if these were indeed the optimal values.

The aerodynamic energy concept considers the work done by the aerodynamic forces on the airfoil per cycle of oscillation, referred to as P. Hence, the transfer function $[T]$ was determined to give a stable system, or a negative value of work P. For an airfoil of specific geometric configuration, Nissim and Abel determined the optimal parameters for a PD transfer function control. For an airfoil flap having a length equal to 20% of the chord (c_β = 0.6, see figure 10), and having its displacement measured at 30% chord from the leading edge, (a = −0.4), they found the following values for the transfer function

$$[\, t \,] = [\, t_{11} \,,\, t_{12} \,] = [\, 0.0 \,,\, -1.86 \,] \;,$$

and

$$[\, t^* \,] = [\, t_{11}^* \,,\, t_{12}^* \,] = [\, 1.0 \,,\, 3.20 \,] \;.$$

In the present study, an airfoil flap having a length equal to 20% of the chord ($c_\beta = 0.6$, see figure 10) and having its elastic axis at 25% of the chord from the leading edge (corresponding value of $a = -0.5$) was considered. A transformation was completed (see Appendix K) to account for the difference in position at which the measurements were made, this gave

$$[\, t \,] = [\, t_{11} \,,\, t_{12} \,] = [\, 0.0 \,,\, -1.86 \,] \;,$$

and

$$[\, t^* \,] = [\, t_{11}^* \,,\, t_{12}^* \,] = [\, 1.0 \,,\, 3.60 \,] \;.$$

The optimization of the transfer function parameters performed by the simplex sequential method considered the net affect the active control had on the flutter velocity and not on the overall system damping. It thus determined parameter values that resulted in an increase in the critical flutter velocity. In fact, the overall magnitude of the system damping seemed to decrease over the majority of the velocity range, but still gave a favourable increase in the flutter speed. The optimized transfer function for case studies

1, 2 and 3, utilizing a PD control, was found to be

$$[T] = [\ 0.0\ ,\ -0.3\ ] + \frac{0.0012}{0.518} s \frac{50000}{s + 50000} [\ 4.0\ ,\ 3.60\ ] \tag{4.32}$$

For case studies 4 and 5, the optimized transfer function was calculated as

$$[T] = [\ 0.0\ ,\ -0.09\ ] + \frac{0.0012}{0.548} s \frac{50000}{s + 50000} [\ 1.0\ ,\ 3.60\ ] \tag{4.33}$$

Note, the value of w_h could have been changed to reflect the no control frequency of the case study in question. However, any change in the value of w_R would have resulted in a proportional increase in the value of a_T. What was important was the relative ratio of t_{11}^* to t_{12}^*.

The calculated values for the transfer function agree quite well with the first derivative generalized coordinates given by the aerodynamic energy concept, but some variance was encountered with respect to the coefficient for the coordinate α. It should be noted that the plunging coordinate, ξ, gave very little useful control information for the system. The method given by Nissim and Abel was quite helpful, in that it gave an indication of the expected magnitude and sign of the required control parameters. It should also be noted that the optimized control parameters are not unique, other values could be used which may achieve the same increase in the flutter velocity.

In order for a system to be completely stable, all the eigenvalues calculated from the aeroelastic equations of motion must possess positive damping characteristics. This can be simplified by considering only the eigenvalues with positive frequency.

The addition of active control to the airfoil had varying affects on each of the cases studied. In some cases, the roots registered only minor changes, while in other cases the changes were quite significant compared to the eigenvalue solution with no control. The results obtained are given as a graph of damping ratio versus non-dimensional velocity, $\bar{U}$ (see Figure 11 and Figure 12). Both the no-control and PD-control eigenvalues are plotted versus the non-dimensional velocity, $\bar{U}$. The no-control roots are defined as NO CONTROL ROOT1 and NO CONTROL ROOT2. The root defined as NO CONTROL ROOT1, shows increasing damping with velocity, until at a certain non-dimensional velocity, it begins to show signs of a decrease in damping. The point where the damping is zero is referred to as the margin of instability, and any increase in velocity will result in flutter. The root denoted as NO CONTROL ROOT2, shows a proportional increase in damping with an increase in velocity. The introduction of active control changes the nature of the two roots, ROOT1 and ROOT2. The PD-control roots are denoted by PD CONTROL ROOT1 and PD CONTROL ROOT2.

Two scenarios were investigated. For example, active control was implemented for the parameters; $\mu = 50.0$, $r_\alpha = 0.5$, $a = -0.5$ and $\omega_h/\omega_\alpha = 0.2$ (Case Study 1). The results are given as a graph of damping ratio versus non-dimensional velocity, $\bar{U}$ (see Figure 11). The control root, denoted as PD CONTROL ROOT1, showed similar behaviour to that of the no-control root, NO CONTROL ROOT1, except that the control root began to show a decay in damping at a larger velocity than that of the no-control root. This resulted in an increase in the critical flutter velocity, from $\bar{U} = 4.53$ to $\bar{U} = 4.8$, an effective increase of approximately 6%. However, the implementation of the control law had quite an affect on the other root, defined as PD CONTROL ROOT2. The result was that this root showed unstable behaviour, negative damping, at low velocity. Thus, such an active control device would have to be turned off initially, and then later turned on, once the aircraft has reached a stable velocity. It may be desired to implement the flutter suppression controller at about 80% of the no control flutter speed

A second example, with the parameters: $\mu = 50.0$, $r_\alpha = 0.6$, $a = -0.6$ and ω_h/ω_α 0.4 (Case Study 2), was also considered (see Figure 12). In this example, the root defined as PD CONTROL ROOT2 showed stable damping characteristics over the entire velocity range. However, the root defined as PD CONTROL ROOT1, showed negative damping at low values of velocity. This root goes through three distinct stages of development. The first stage is defined when the root has non-positive damping. The second stage is defined by the root possessing positive damping characteristics, but still has less damping in comparison to its no-control root, NO CONTROL ROOT1. The final stage begins when the PD-control root shows improved damping in comparison to the no-control root. This allowed for an increase in the flutter velocity from $U = 5.1$ to $U = 5.6$. This resulted in a net increase in flutter velocity of approximately 10% (see Figure 12). Additional comparisons of no-control and PD-control for various sets of parameters are included in Table 7.

Case Study	μ	r_α	a	w_h / w_α	Flutter Velocity $\bar{U}$ No Control	PD Control	% Increase
1	50.0	0.5	-0.5	0.2	4.53	4.8	6.0
2	50.0	0.6	-0.6	0.4	5.10	5.6	10.0
3	100.0	0.5	-0.5	0.2	6.24	6.6	5.5
4	75.0	0.4	-0.4	0.3	3.68	3.9	5.5
5	100.0	0.4	-0.4	0.3	4.16	4.4	5.5

Table 7 Incompressible Regime : Comparison of critical flutter velocity, $\bar{U}$, using no control and PD control solved by the *Laplace* method, with the constants : $r_\beta = 0.5$, $x_\beta = 0.25$, $x_\alpha = 0.25$, $c_\beta = 0.6$ and $b = 1.0$.

Although the increase in flutter speed in these case studies can be considered only as fair, it has been shown that control analysis can be implemented quite easily by working in the Laplace domain. In past studies, increases in flutter speeds of 20 to 25% have been achieved

The use of a PID control law resulted in no additional increase (beyond that achieved via PD control) in the critical flutter velocity, $\bar{U}$. Maybe, this was to be expected. The introduction of an integrator term is usually done in the hope of cancelling the steady-state error, in comparison to a known desired control velocity. However, in this context, the object was not to control the velocity to some known required velocity, but to find the greatest possible maximum critical velocity.

11 Conclusions

A flutter analysis was considered for a rigid airfoil, flexibly mounted in the bending and torsional directions. A simple and effective way of determining flutter velocities was developed by solving the equations in the Laplace domain. In this procedure the iteration process encountered in the traditional $U-g$ and $p-k$ methods is eliminated, by using rational functions to approximate the unsteady aerodynamic forces as a function of reduced frequency.

The validity of the Laplace solution depends greatly on how well the unsteady aerodynamics can be approximated. It is critical that the approximation be very close to the actual data. The unsteady aerodynamics for both incompressible and transonic flow were approximated by rational functions of the Nasa-Langley form with great success

$$[N] = [A_0] + \frac{b}{U}[A_1]s + (\frac{b}{U})^2[A_2]s^2 +$$

$$[A_3]\frac{s}{s+\frac{U}{b}b_1} + [A_4]\frac{s}{s+\frac{U}{b}b_2} + [A_5]\frac{s}{s+\frac{U}{b}b_3} + [A_6]\frac{s}{s+\frac{U}{b}b_4}$$

The optimization method employed a least squares method in conjunction with a simplex method. The use of the simplex method to optimize the lag terms showed marked improvement in the approximation in comparison with previous work, where the lag terms were chosen randomly. For incompressible flow, it was necessary to approximate Theodorsen's function, $C(k)$ and $C(k)ik$; the approximation error was found to be insignificant, even for a large reduced frequency range of $0.0 < k < 10.0$. This allowed for a robust evaluation of any aeroelastic problem in the incompressible flow. The Laplace method was found to give values of flutter velocity and flutter frequency identical to those given by the $p-k$ and the $U-g$ methods. For values of subcritical velocity, the Laplace method showed good comparison with frequency and damping obtained via the $p-k$ method

For transonic flow, it was necessary to approximate the four unsteady aerodynamic coefficients C'_{l_h}, C'_{l_α}, C'_{m_h} and C'_{m_α}. A much decreased range of reduced frequency was considered, $0.0 \leq k \leq 0.5$. Also, constraints were imposed on the aerodynamic derivatives whose magnitudes were zero at $k = 0.0$, namely C_{l_h} and C_{m_h}. The results were not as good as those obtained in incompressible flow, but were still found to be excellent overall. The reduced frequency range had little effect on the calculation of the flutter velocities, and fair representations of frequency and damping ratio over most of the range of non-dimensional velocity were obtained. However, some discrepancies were apparent at the lower values of velocity for one of the complex conjugate roots. Here, the frequencies obtained by the eigenvalue solution fell outside of the applicable reduced frequency range, and thus, resulted in some difference in the damping and frequency response to that obtained with the $p-k$ and $V-g$ methods.

Solving the equations in the Laplace domain allows for the efficient implementation of active control to alleviate the flutter instability. Such control was considered with some success for incompressible flow. Although, the increase in the flutter velocity can only be considered as fair (6 to 10%), it was shown that the Laplace method was a viable way of incorporating active control into the aeroelastic equations.

Future work could consider the implementation of other control laws, in the hope of obtaining a more significant increase in flutter velocity. In addition, the control theory could be extended to account for transonic flow. This would require the determination of the unsteady aerodynamics incorporating a flap for transonic flow.

12 References

Abel, I., *An Analytical Technique for Predicting the Characteristics of a Flexible Wing Equipped with an Active Flutter-Suppression System and Comparison with Wind-Tunnel Data*. NASA TP-1367, 1979.

Abel, I. and Nissim, E., *Development and Application of an Optimization Procedure for Flutter Suppression using the Aerodynamic Concept*. NASA TP-1137, 1978.

Adams, W. M. and Tiffany, S. H., *Fitting Aerodynamic Forces in the Laplace Domain: An Application of a Nonlinear Nongradient Technique to Multilevel Constrained Optimization*. NASA TM-86317, 1984.

Adams, W. M. and Tiffany, S. H., *Nonlinear Programming Extensions to Rational Approximation Methods for Unsteady Aerodynamic Forces*. NASA TP-2776, 1988.

Albano, E. D., Farr, J. L. and Traci, R. M., *Perturbation Method for Transonic Flows About Oscillating Airfoils*. AIAA Journal, vol. 14, September 1976, pp. 1258-1265.

Angelini, J. J. and Couston, M., *Numerical Solutions of Nonsteady Two-Dimensional Transonic Flows*. J. Fluids Engineering, vol. 101, September 1979, pp. 341-347.

Ashley, H. and Bisplinghoff, R. L., Principles of Aeroelasticity. John Wiley & Sons, N. Y., 1962.

Ashley, H., Bisplinghoff, R. L. and Halfman, R. L., Aeroelasticity. Addison-Wesley

Publishing Company Inc. Reading, Massachusetts, 1955

Baker, G. A. Jr. and Gammel, J. L., The Pade Approximant in Theoretical Physics Academic Press, N. Y., 1970

Ballhaus, W. F. and Goorjian, P. M., *Implicit Finite-Difference Computations of Unsteady Transonic Flow About Airfoils* AIAA Journal, vol. 15, December 1977, pp. 1728-1735

Bennett, R. M., *Application of Zimmerman Flutter-Margin Criterion to a Wind-Tunnel Model* NASA TM-84545, 1982.

Chin, W. C. and Rizzetta, D. P., *Effect of Frequency in Unsteady Transonic Flow* AIAA Journal, vol. 17, July 1979, pp. 779-781.

Dowell, E. H., *A Simple Method for Converting Frequency- Domain Aerodynamics to the Time Domain* NASA TM-81844, 1980.

Dowell, E. H., *Unsteady Transonic Aerodynamics and Aeroelasticity* Applied Mechanics Reviews, vol. 41, no. 8, August 1988.

Dunn, H. J., *An Analytical Technique for Approximating Unsteady Aerodynamics in the Time Domain* NASA TP-1738, 1980.

Flannery, B. P., Press, W. H., Teukolsky, S. A., Vetterling, W. T. and Robert M., Numerical Recipes : The Art of Scientific Computing. Cambridge University Press, Cambridge, 1986.

Fletcher, R. and Powell, M. J. D., *A Rapidly Convergent Descent Method for Mini-*

mization, The Computer Journal, vol. 6, 1963, p. 163

Fung, Y. C., An Introduction to The Theory of Aeroelasticity, John Wiley & Sons N.Y., 1955

Garrick, I. E. and Theodorsen, T., *Nonstationary Flow about a Wing-Aileron-Tab Combination including Aerodynamic Balance* Report no. 736, Langley Memorial Aeronautical Laboratory Langley Field, Va., 1941

Ghiringhelli, G. L. and Mantegazza, P., *Active Flutter Suppression for a Wing Model* Journal of Aircraft, vol. 27, no. 4, 1989, pp. 334-341

Guruswamy, P., Olsen, J. J., Striz, A. G. and Yang, T. Y., *Flutter Analysis of a NACA 64A006 Airfoil in Small Disturbance Transonic Flow* Journal of Aircraft, vol. 17 no. 4, April 1980, pp. 225-232.

Head, A. L., Jr., *A Philosophy of Design for Flutter* Proceedings of the National Specialists Meetings on Dynamics and Aeroelasticity, Ft. Worth, Texas, November 1958 pp. 59-65, published by the Institute of the Aeronautical Sciences, New York

Houwink, R. and van der Vooren, J., *Improved Version of LTRAN2 for Unsteady Transonic Flow Computations* AIAA Journal, vol. 18, August 1980 pp. 1008-1010

Jackson, P. and Lawrence, A. J., *Comparison of Different Methods of Assessing the Free Oscillating Characteristics of Aeroelastic Systems*, RAE C.P., no. 1084, 1968

Jones, R. T., *The Unsteady Lift of a Wing of Finite Aspect Ratio* NASA Report No. 681, 1940.

Karpel, M., *Design for Active and Passive Flutter Suppression and Gust Alleviation.* NASA CR 3482, 1981.

Karpel, M., *Sensitivity Derivatives of Flutter Characteristics and Stability Margins for Aeroservoelastic Design.* Journal of Aircraft, vol. 27, no. 1, 1989, pp. 368-375.

Kass, G. J. and Thompson, G. O., *Active Flutter Suppression - An Emerging Technology.* Journal of Aircraft, vol. 9, no. 3, 1971, pp. 230-238.

Lee, B. H. K., *A Study of Transonic Flutter of a Two - Dimensional Airfoil Using the U-g and p-k Methods.*, 1984.

Mead, R. and Nelder, J. A., *A Simplex Method for Function Minimization.* Computer J., vol. 7, no. 1, January 1965, pp. 308- 313.

Messina, A. F. and O'Connell, R. F., *Development and Testing of an Active Flutter Margin Augmentation System for a Commercial Transport Airplane.* AIAA, Paper No. 79-0790, April 1979.

Nelson, L. S. and Olsson, D. M., *The Nelder-Mead Simplex Procedure for Function Minimization.* Technometrics, vol. 17, no. 1, February 1975, pp. 45-51.

Nissim, E., *Flutter Suppression using Active Controls based on the Concept of Aerodynamic Energy.* NASA TN D-6199, Langley Research Center, Hampton, Va., 1971.

Nissim, E., *Recent Advances in Aerodynamic Energy Concept for Flutter Suppression and Gust Alleviation using Active Controls.* NASA TN D-8519, September 1977.

Olson, D. M., *A Sequential Simplex Program for Solving Minimization Problems.* J.

Qual. Technol., vol. 6, no. 1, January 1974, pp. 53-57.

O'Neil, R., *Algorithm AS47 - Function Minimization Using a Simplex Procedure*. Applied Statistics, 20, 1971, pp. 338-345.

Poirel, D., *An Investigation of the use of Rational Functions to Approximate Unsteady Aerodynamics for Active Flutter- Suppression Systems*. National Aeronautical Establishment, Ottawa, McGill University, 1988.

Roger, K., L., *Airplane Math Modelling Methods for Active Control Design*. AGARD CP-228, August 1977.

Theodorsen, T., *General Theory of Aerodynamic Instability and the Mechanism of Flutter*. National Advisory Committee for Aeronautics, Report no. 496, 1935.

Vepa, R., *On the Use of Pade Approximants to Represent Unsteady Aerodynamic Loads for Arbitrarily Small Motions of Wings*. AIAA Paper No. 76-17, January 1976.

Vepa, R., *Finite State Modelling of Aeroelastic Systems*. NASA CR-2779, 1977.

Weissenburger, J. T. and Zimmerman, N. H., *Prediction of Flutter Onset Speed Based on Flight Testing at Subcritical Speeds*. Journal of Aircraft, vol. 1, no. 4, July-August 1964, pp. 190-202.

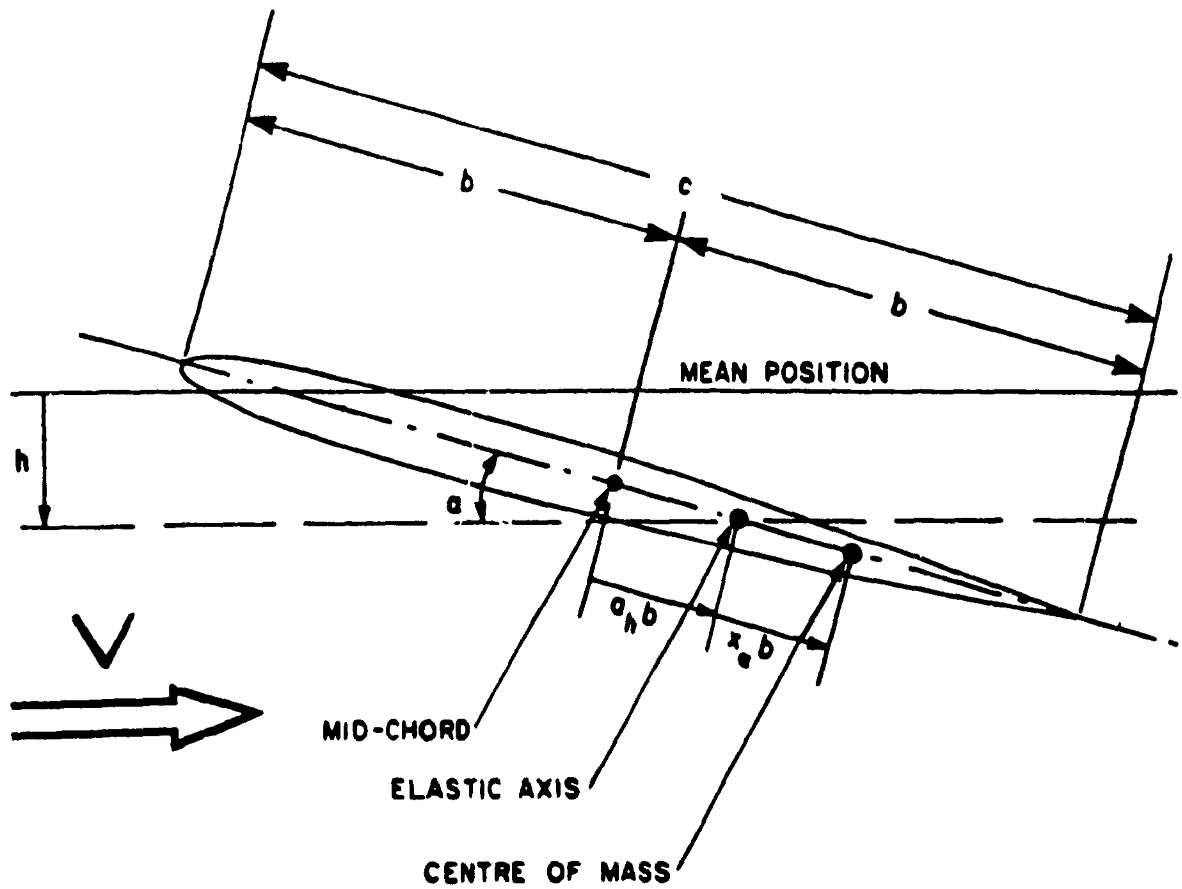


Figure 1 Schematic representation of airfoil with freedom to move in torsional and bending directions (2DOF). (Lee 1984)

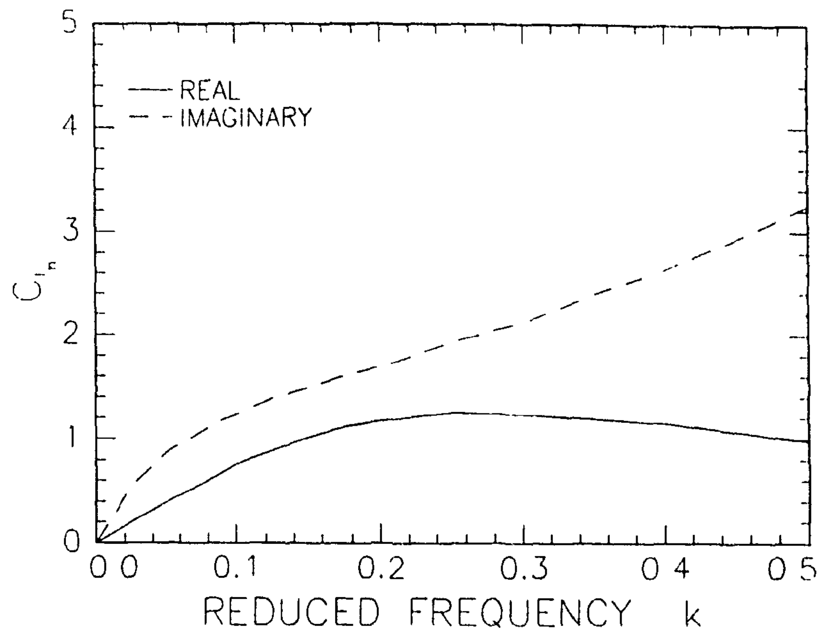


Figure 2a Derivative of Lift Coefficient, C_l, with respect to Plunge, h, versus Reduced Frequency, k, at $M = 0.85$ for a NACA 64A006 Airfoil

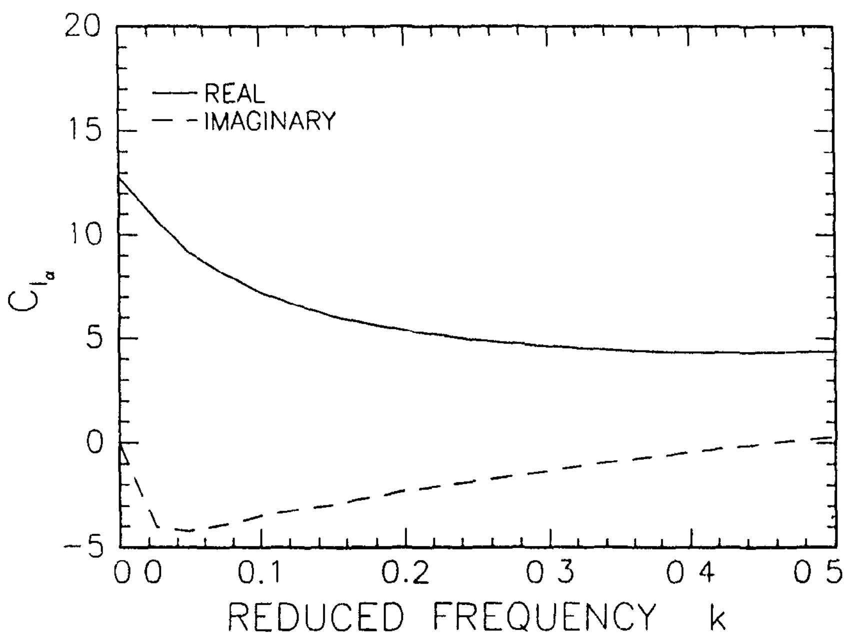


Figure 2b Derivative of Lift Coefficient, C_l, with respect to Pitch, α, versus Reduced Frequency, k, at $M = 0.85$ for a NACA 64A006 Airfoil

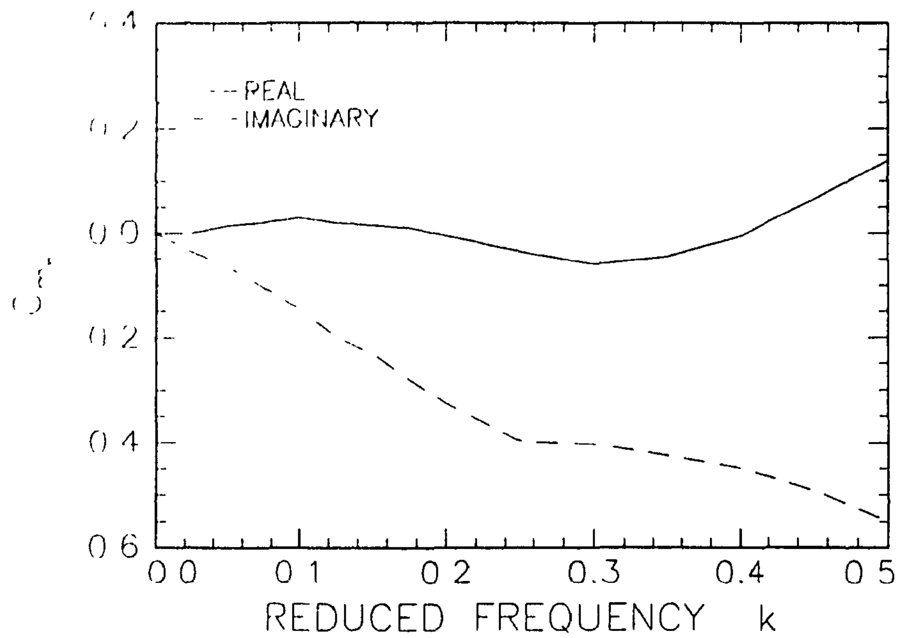


Figure 2c Derivative of Moment Coefficient, C_m, with respect to Plunge, h, versus Reduced Frequency, k, at $M = 0.85$ for a NACA 64A006 Airfoil

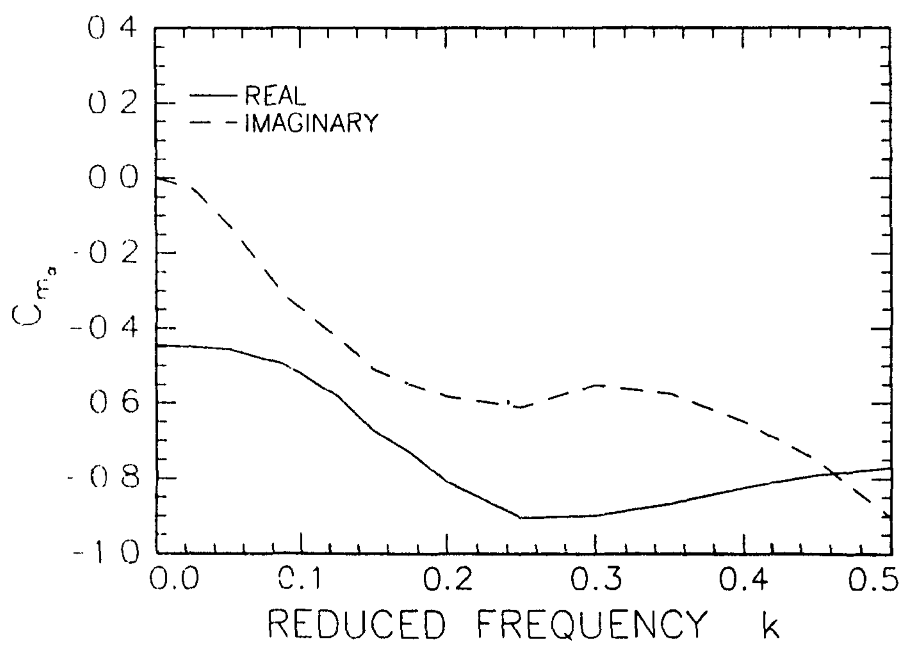


Figure 2d Derivative of Moment Coefficient, C_m, with respect to Pitch, α, versus Reduced Frequency, k, at $M = 0.85$ for a NACA 64A006 Airfoil

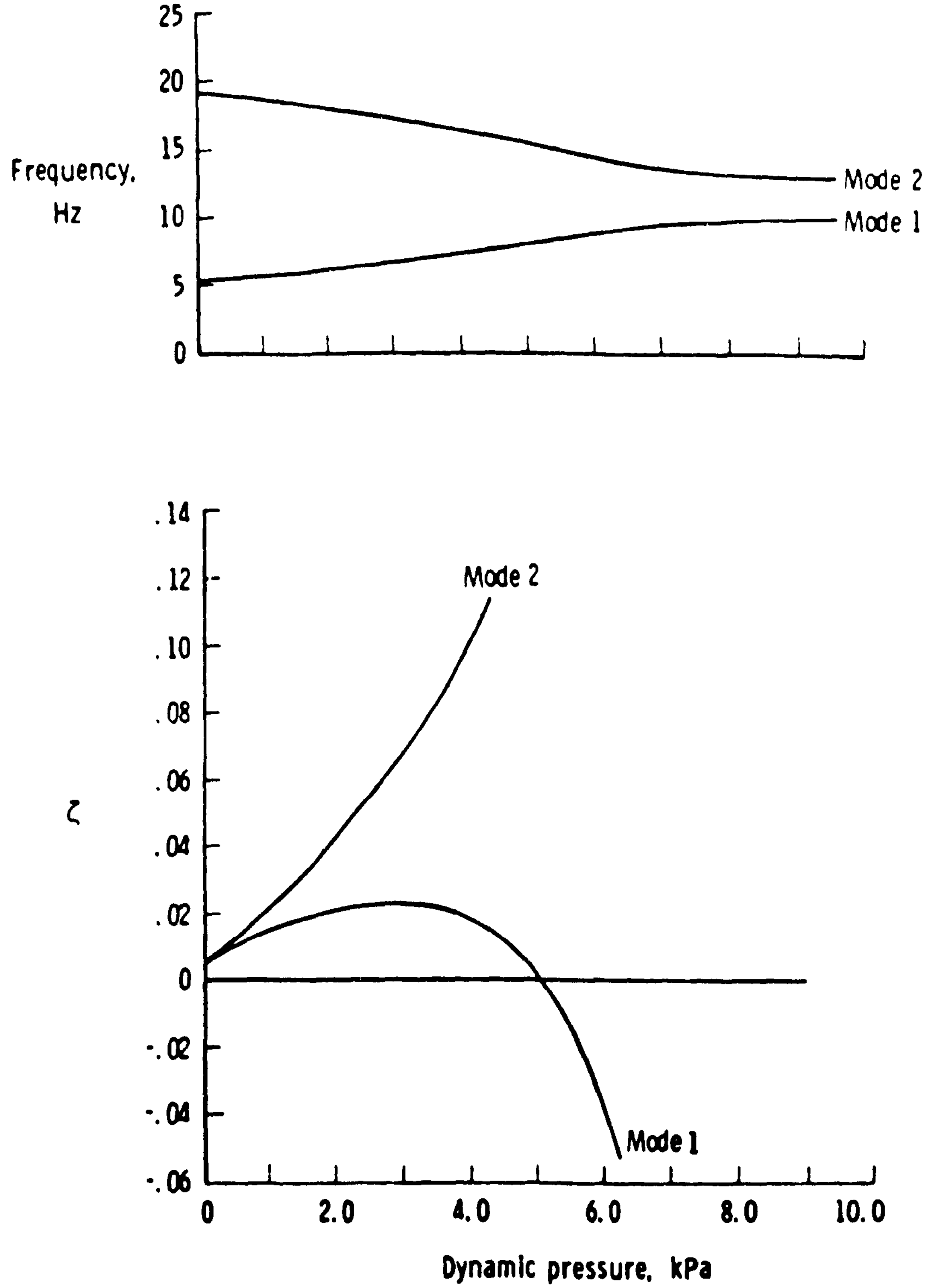


Figure 3 Eigenvalue Solution : Damping and Frequency versus Dynamic Pressure (Abel 1979)

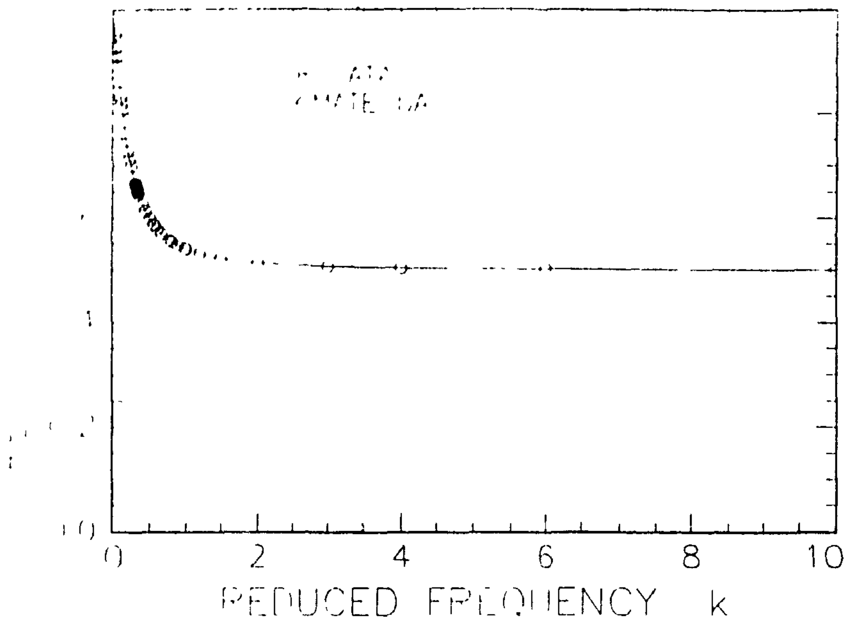


Figure 4a-1 Comparison between the Rational Function Approximation of $C(k)$ and the True $C(k)$, Real Part, as a Function of Reduced Frequency, $0.0 < k < 10.0$

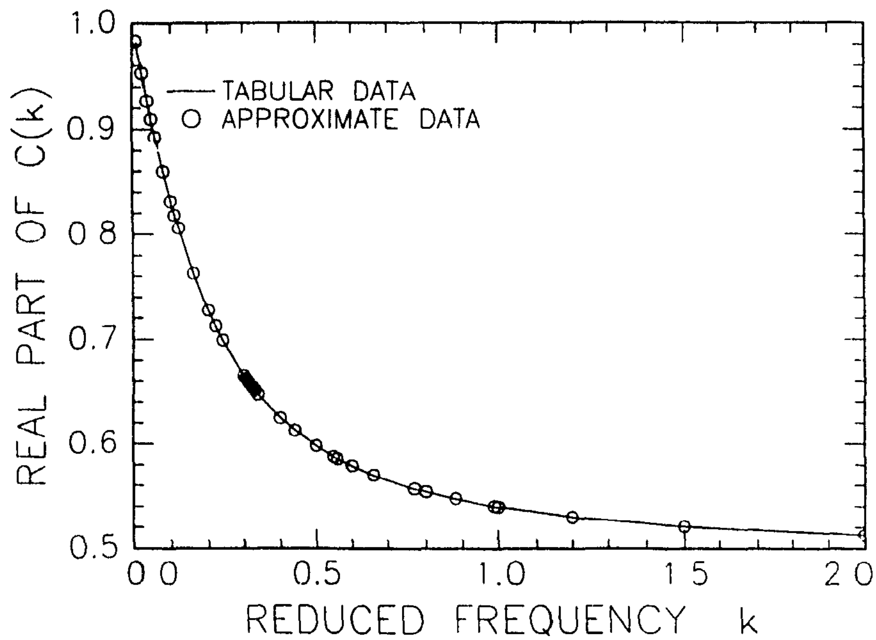


Figure 4a-2 Comparison between the Rational Function Approximation of $C(k)$ and the True $C(k)$, Real Part, as a Function of Reduced Frequency, $0.0 < k < 2.0$ - a magnification of figure 4a-1

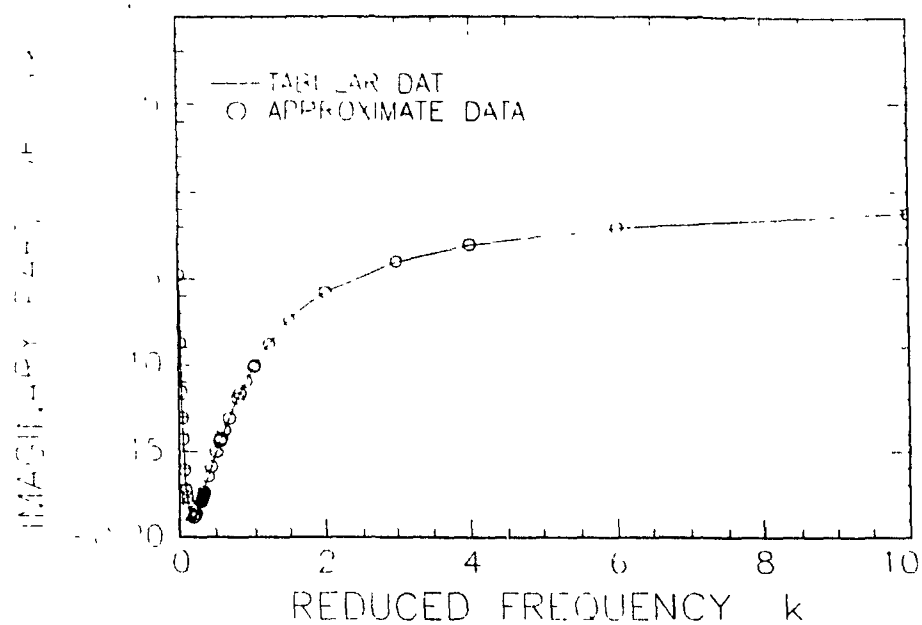


Figure 4a-3 Comparison between the Rational Function Approximation of $C(k)$ and the True $C(k)$, Imaginary Part, as a Function of Reduced Frequency, $0.0 < k < 10.0$

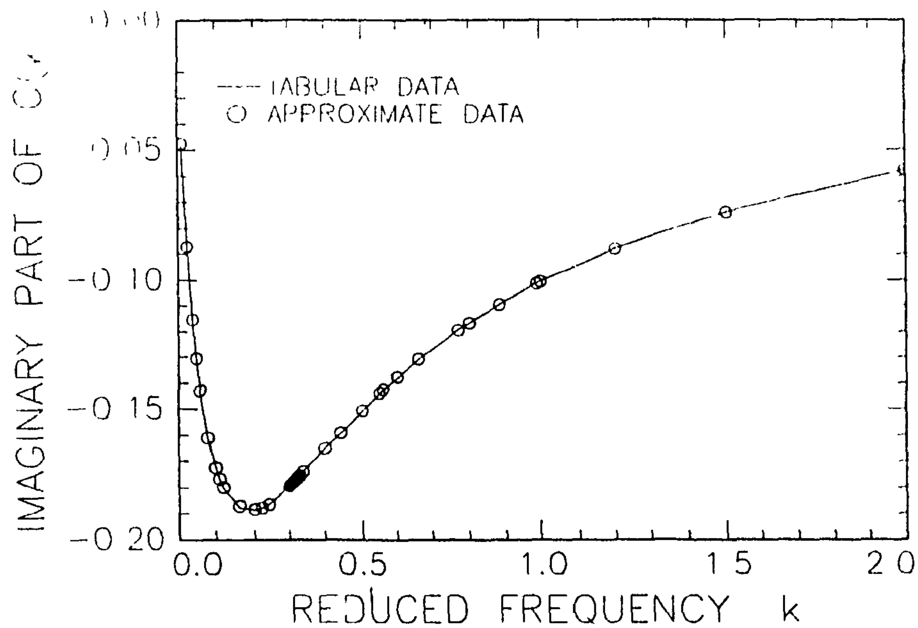


Figure 4a-4 Comparison between the Rational Function Approximation of $C(k)$ and the True $C(k)$, Imaginary Part, as a Function of Reduced Frequency, $0.0 < k < 2.0$ - a magnification of figure 4a-3

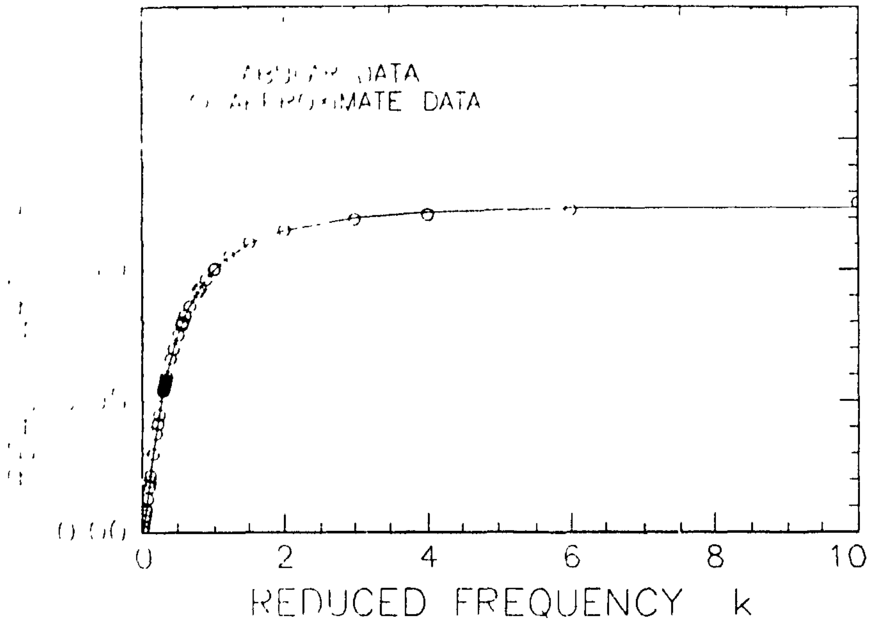


Figure 4b-1 Comparison between the Rational Function Approximation of $C(k)ik$ and the True $C(k)ik$, Real Part, as a Function of Reduced Frequency, $0.0 < k < 10.0$.

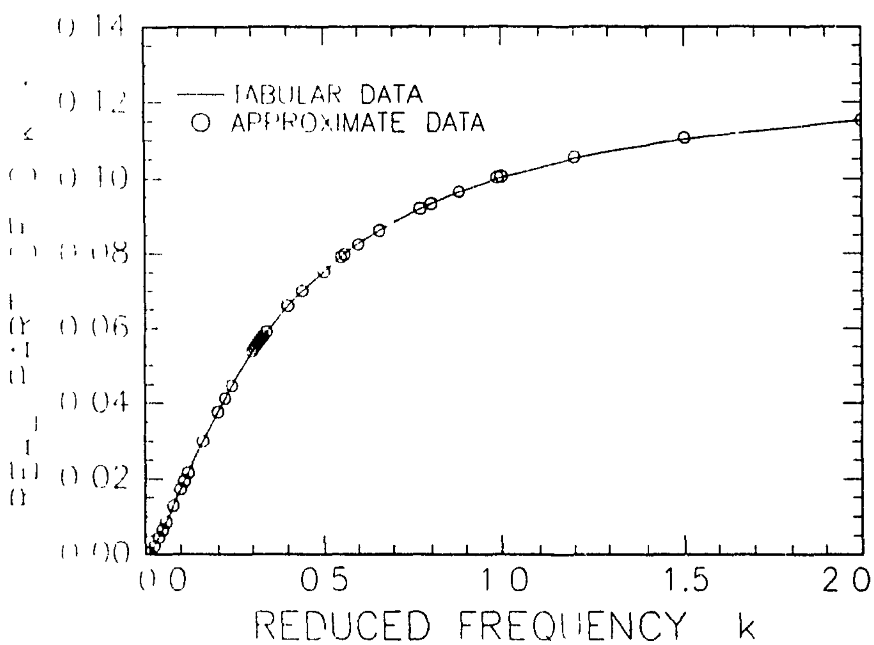


Figure 4b-2 Comparison between the Rational Function Approximation of $C(k)ik$ and the True $C(k)ik$, Real Part, as a Function of Reduced Frequency, $0.0 < k < 2.0$ - a magnification of figure 4b-1

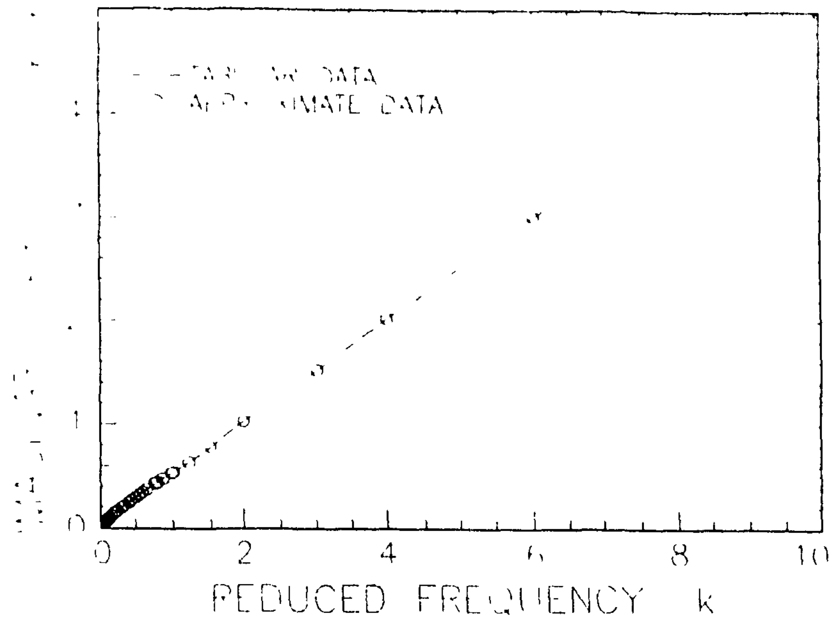


Figure 4b-3 Comparison between the Rational Function Approximation of $C(k)ik$ and the True $C(k)ik$, Imaginary Part, as a Function of Reduced Frequency, $0.0 < k < 10.0$

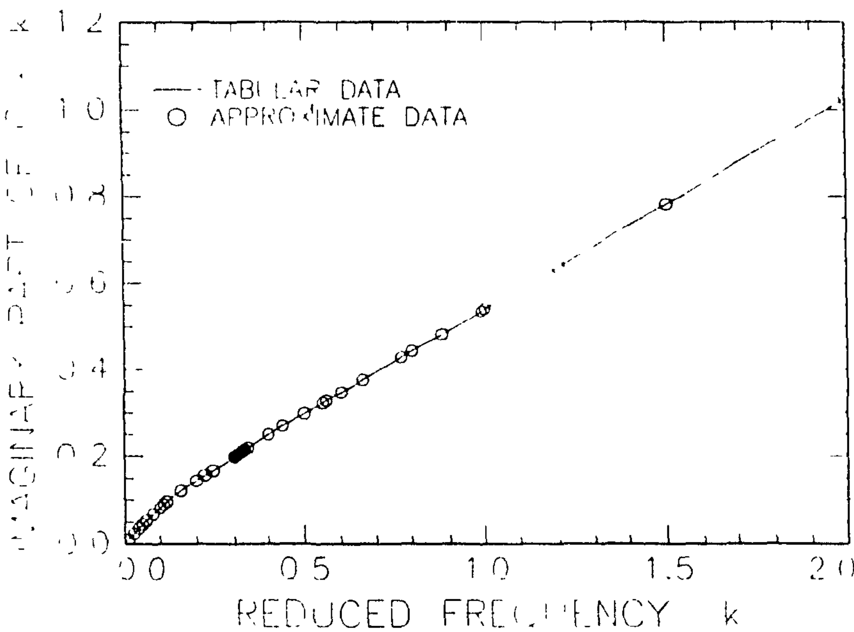


Figure 4b-4 Comparison between the Rational Function Approximation of $C(k)ik$ and the True $C(k)ik$, Imaginary Part, as a Function of Reduced Frequency, $0.0 < k < 2.0$ - a magnification of figure 4b-3

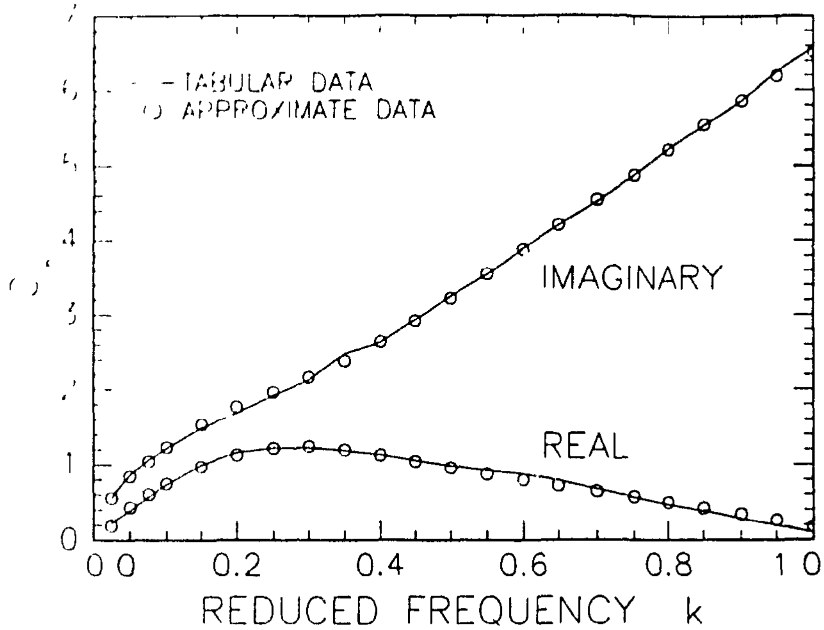


Figure 5a-1 Comparison between the Rational Function Approximation of C_{l_h} and the True C_{l_h} as a Function of Reduced Frequency, $0.025 \leq k \leq 1.0$, at $M = 0.85$ for a NACA 64A006 Airfoil.

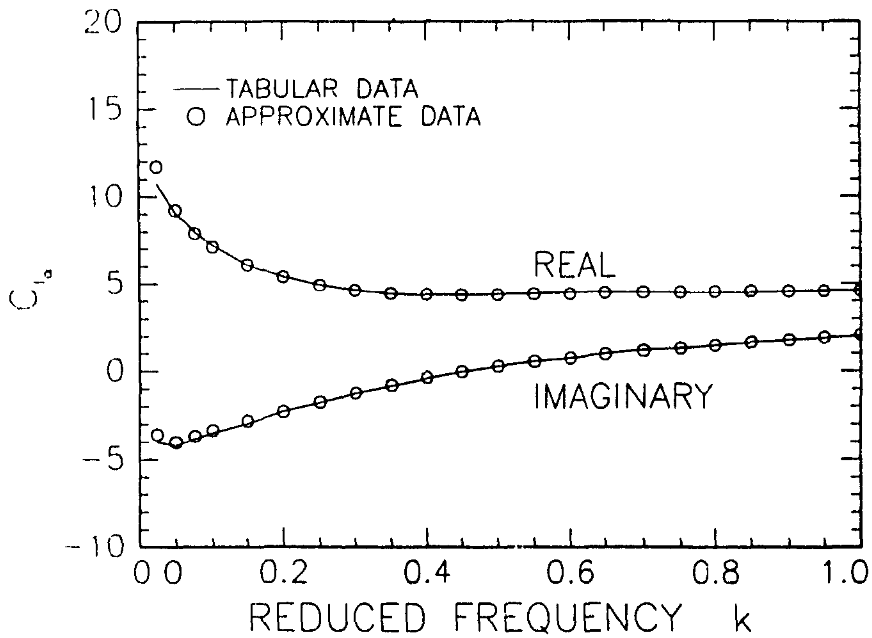


Figure 5b-1 Comparison between the Rational Function Approximation of C_{l_α} and the True C_{l_α} as a Function of Reduced Frequency, $0.025 \leq k \leq 1.0$, at $M = 0.85$ for a NACA 64A006 Airfoil.

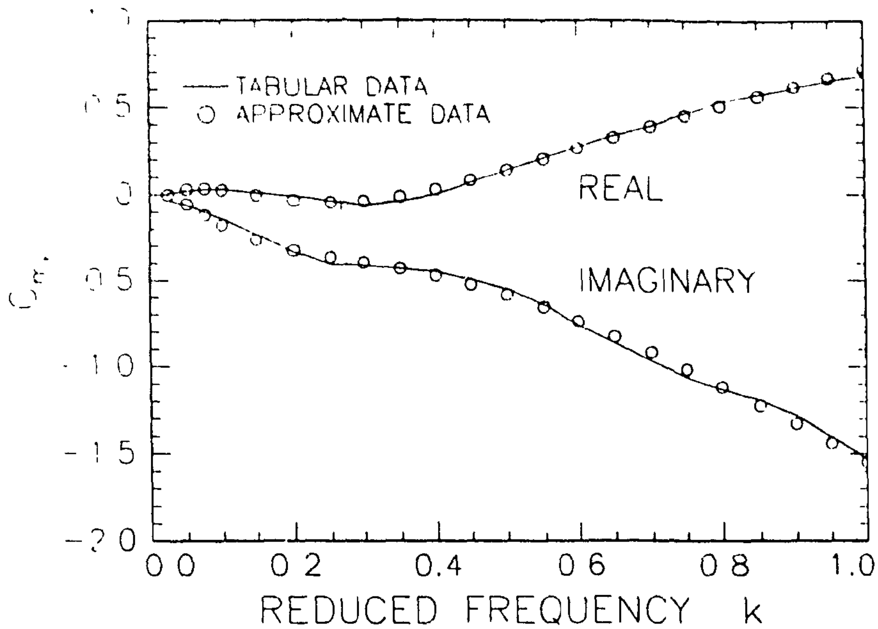


Figure 5c-1 Comparison between the Rational Function Approximation of C_{m_h} and the True C_{m_h} as a Function of Reduced Frequency, $0.025 \leq k \leq 1.0$, at $M = 0.85$ for a NACA 64A006 Airfoil.

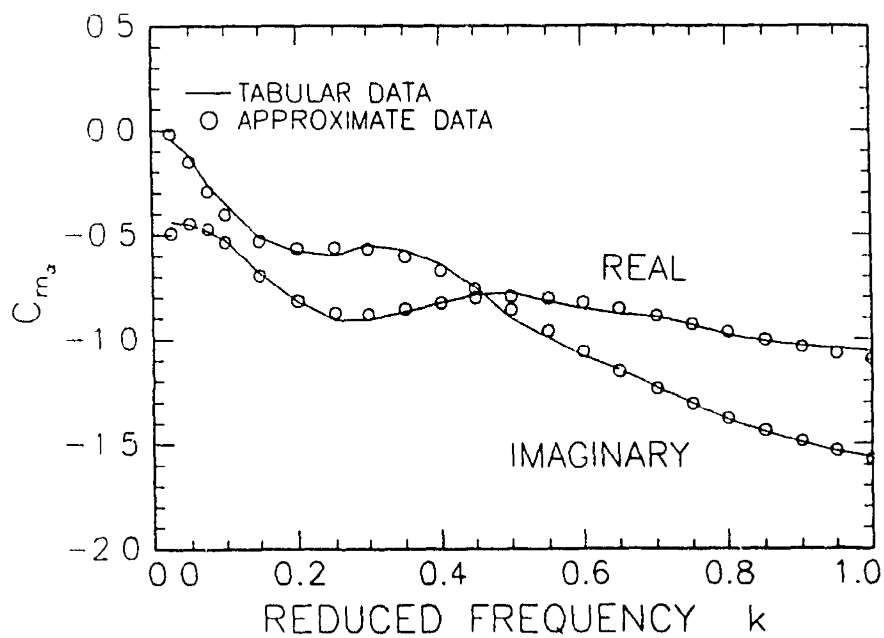


Figure 5d-1 Comparison between the Rational Function Approximation of C_{m_α} and the True C_{m_α} as a Function of Reduced Frequency, $0.025 \leq k \leq 1.0$, at $M = 0.85$ for a NACA 64A006 Airfoil.

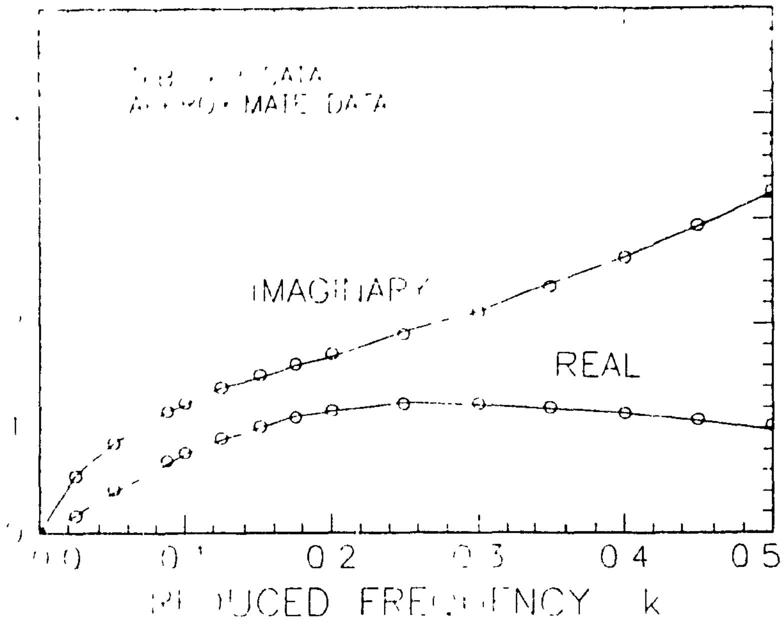


Figure 5a-2 Comparison between the Rational Function Approximation of C_{l_h} and the True C_{l_h} as a Function of Reduced Frequency, $0.0 \leq k \leq 0.5$, at $M = 0.85$ for a NACA 64A006 Airfoil.

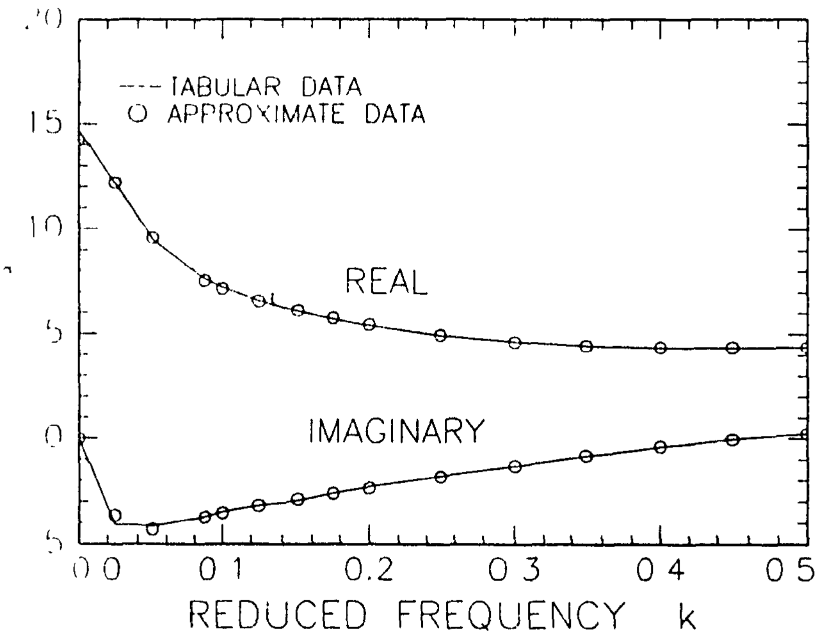


Figure 5b-2 Comparison between the Rational Function Approximation of C_{l_α} and the True C_{l_α} as a Function of Reduced Frequency, $0.0 \leq k \leq 0.5$, at $M = 0.85$ for a NACA 64A006 Airfoil.

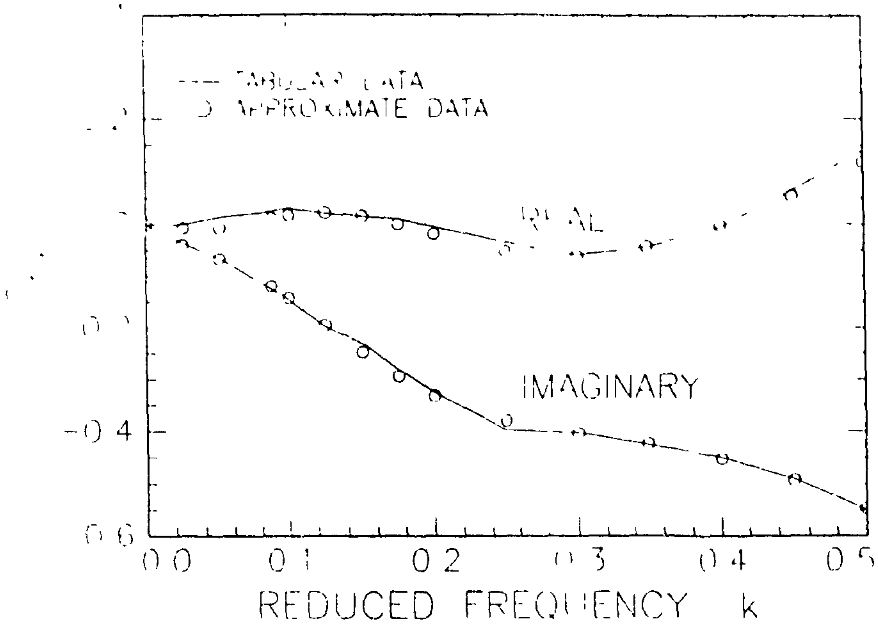


Figure 5c-2 Comparison between the Rational Function Approximation of C_{m_h} and the True C_{m_h} as a Function of Reduced Frequency, $0.0 \leq k \leq 0.5$, at $M = 0.85$ for a NACA 64A006 Airfoil

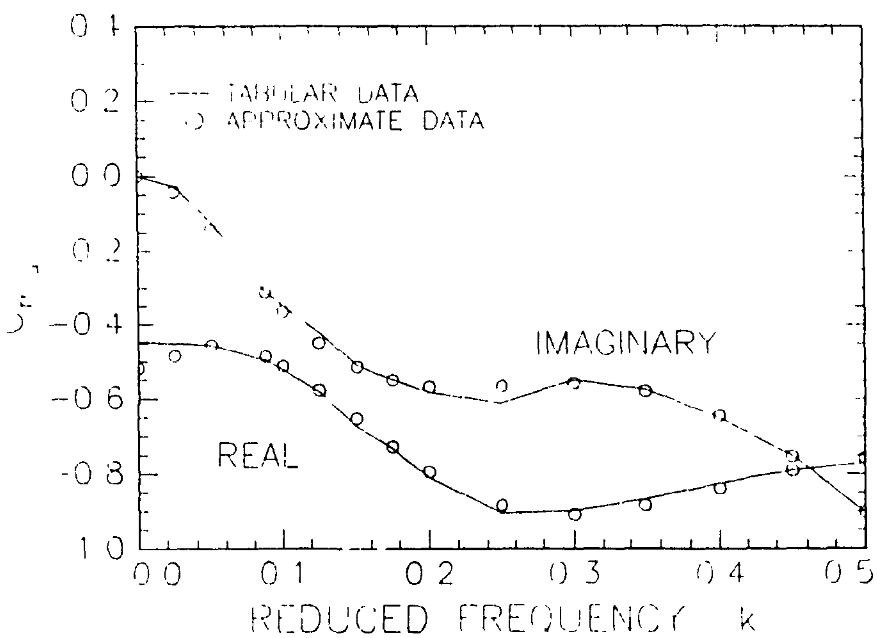


Figure 5d-2 Comparison between the Rational Function Approximation of C_{m_α} and the True C_{m_α} as a Function of Reduced Frequency, $0.0 \leq k \leq 0.5$ at $M = 0.85$ for a NACA 64A006 Airfoil

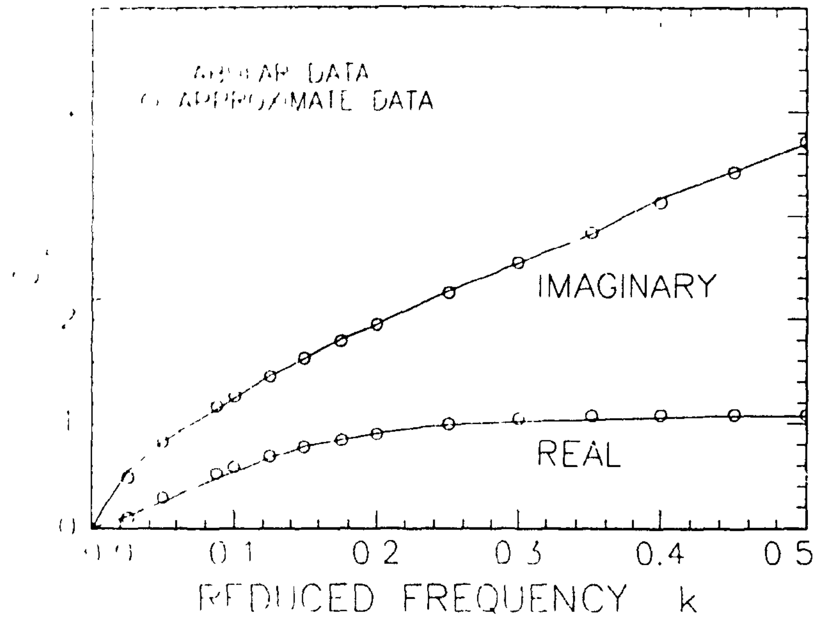


Figure 6a Comparison between the Rational Function Approximation of C'_{l_h} and the True C'_{l_h} as a Function of Reduced Frequency, $0.0 \leq k \leq 0.5$, at $M = 0.80$ for a NACA 64A006 Airfoil.

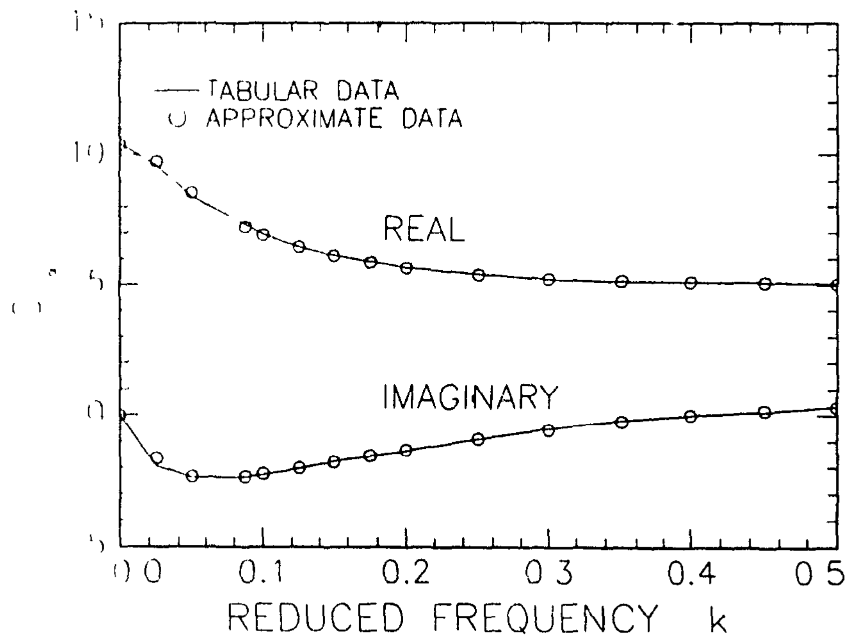


Figure 6b. Comparison between the Rational Function Approximation of C'_{l_α} and the True C'_{l_α} as a Function of Reduced Frequency, $0.0 \leq k \leq 0.5$, at $M = 0.80$ for a NACA 64A006 Airfoil.

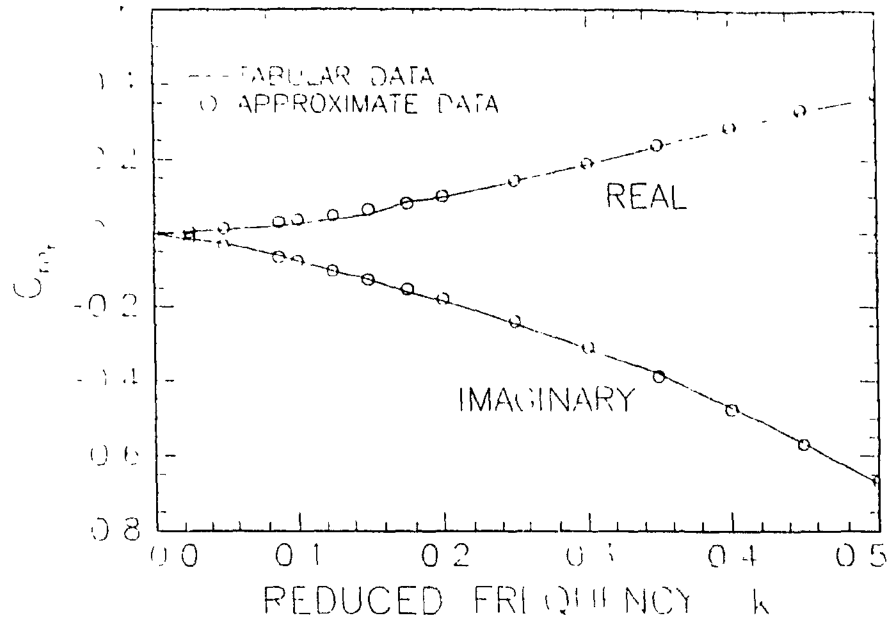


Figure 6c Comparison between the Rational Function Approximation of C_{m_h} and the True C_{m_h} as a Function of Reduced Frequency, $0.0 \leq k \leq 0.5$, at $M = 0.80$ for a NACA 64A006 Airfoil.

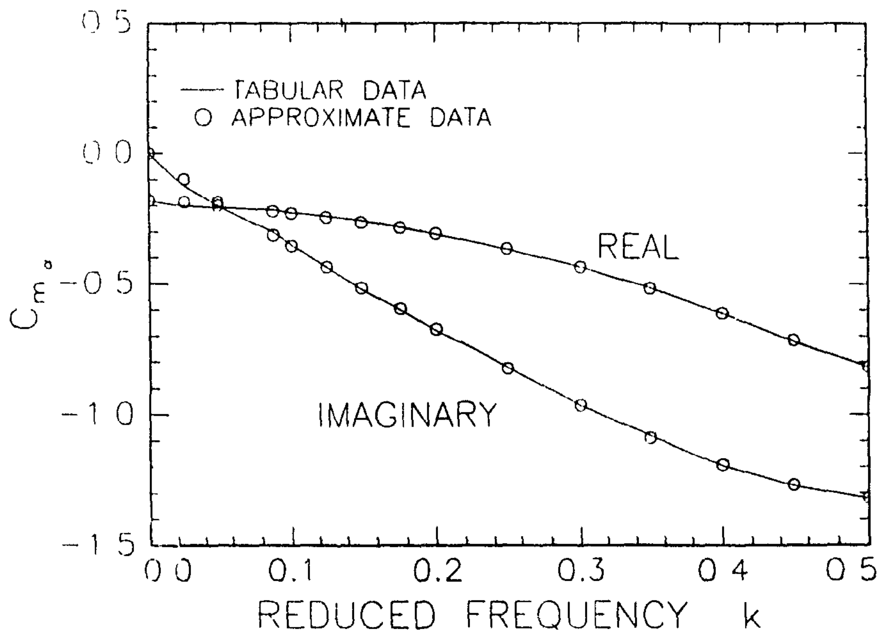


Figure 6d Comparison between the Rational Function Approximation of C_{m_α} and the True C_{m_α} as a Function of Reduced Frequency, $0.0 \leq k \leq 0.5$, at $M = 0.80$ for a NACA 64A006 Airfoil.

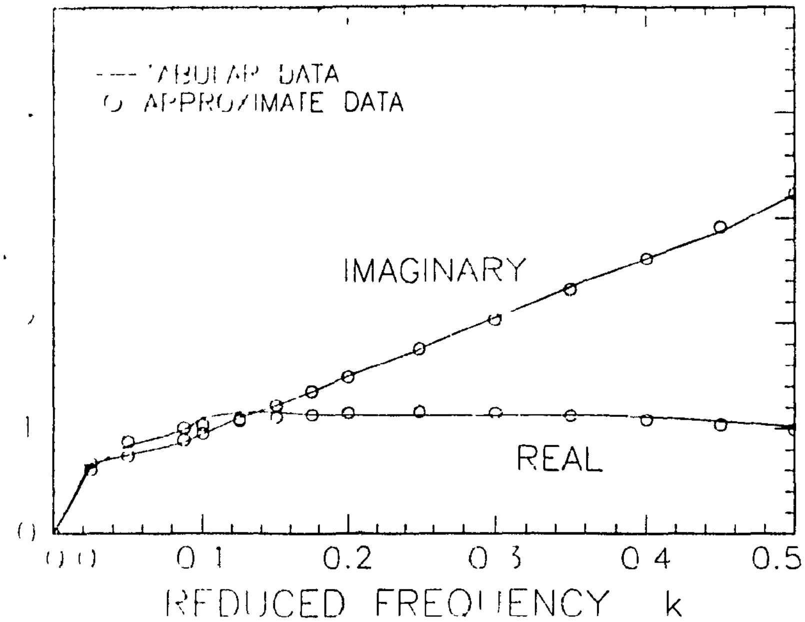


Figure 7a Comparison between the Rational Function Approximation of C_{l_h} and the True C_{l_h} as a Function of Reduced Frequency, $0.0 \leq k \leq 0.5$, at $M = 0.875$ for a NACA 64A006 Airfoil.

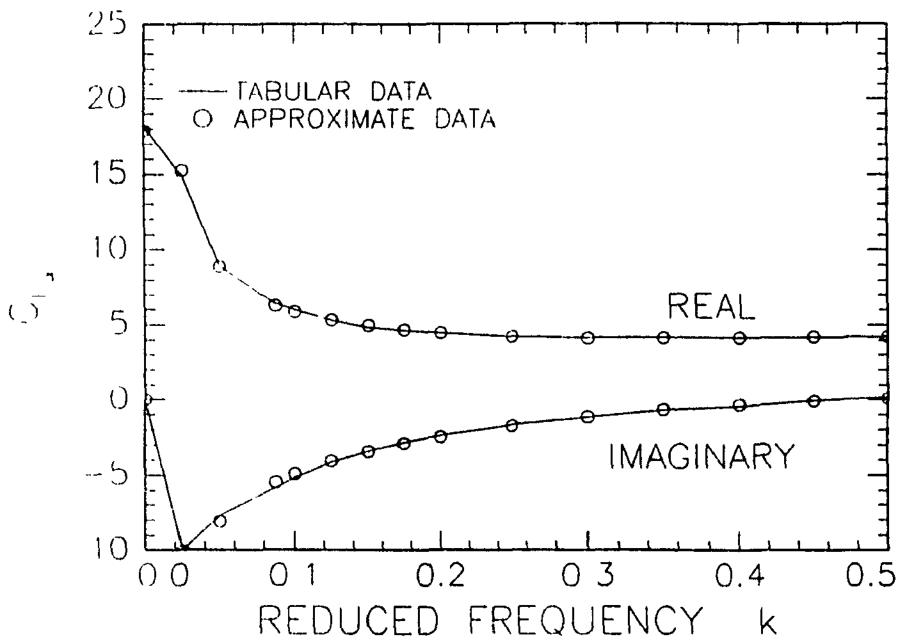


Figure 7b Comparison between the Rational Function Approximation of C_{l_α} and the True C_{l_α} as a Function of Reduced Frequency, $0.0 \leq k \leq 0.5$, at $M = 0.875$ for a NACA 64A006 Airfoil.

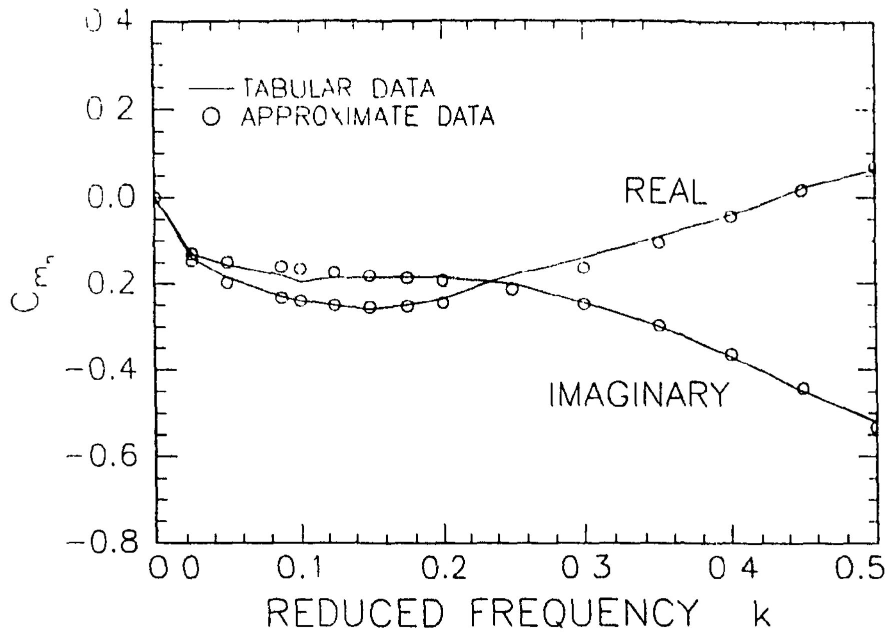


Figure 7c Comparison between the Rational Function Approximation of C_{m_h} and the True C_{m_h} as a Function of Reduced Frequency, $0.0 \leq k \leq 0.5$, at $M = 0.875$ for a NACA 64A006 Airfoil.

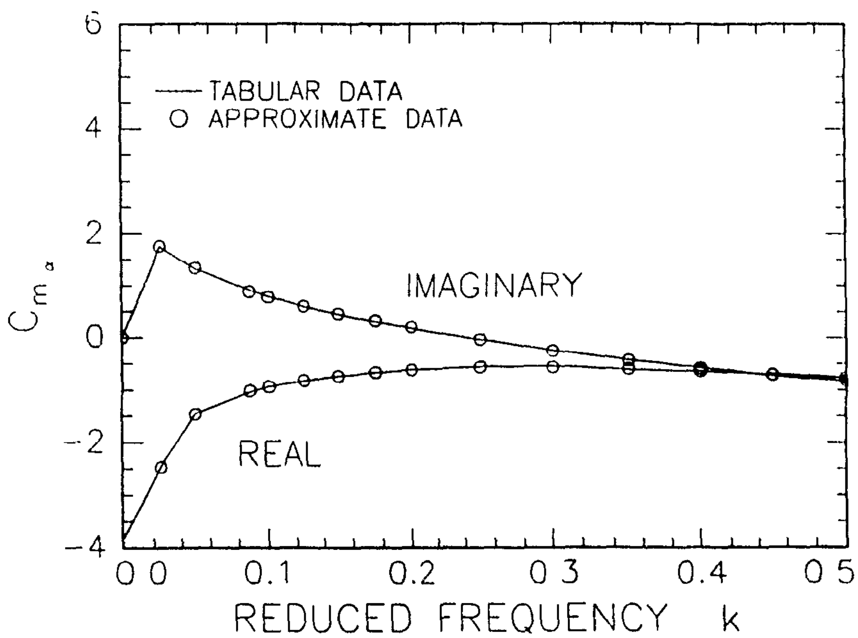


Figure 7d Comparison between the Rational Function Approximation of C_{m_α} and the True C_{m_α} as a Function of Reduced Frequency, $0.0 \leq k \leq 0.5$, at $M = 0.875$ for a NACA 64A006 Airfoil.

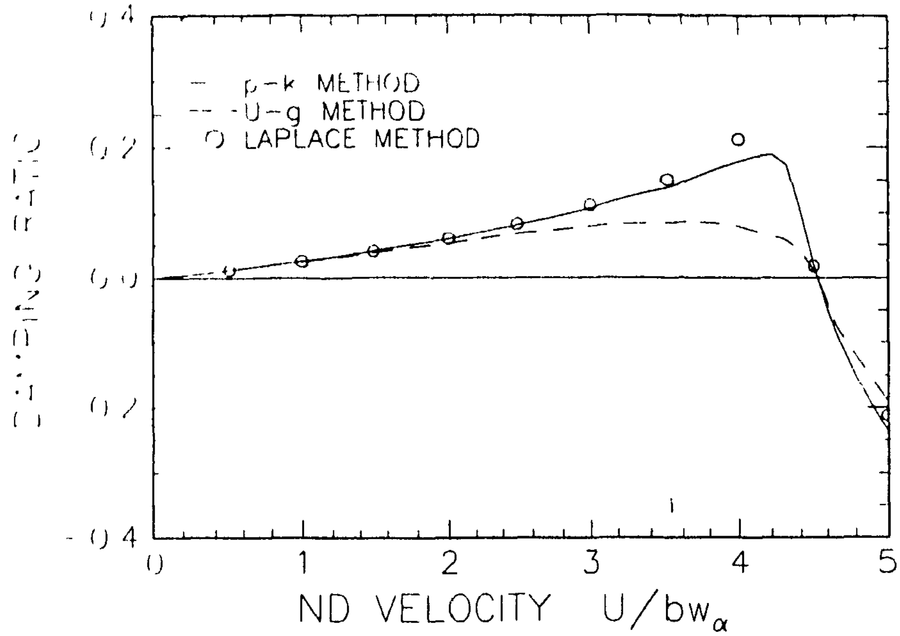


Figure 8a-1 Comparison of $U-g$ and $p-k$ with *Laplace* generated graphs of Damping Ratio, Root 1, versus Non-Dimensional Velocity, $\bar{U}$. Case Study #1 : $\mu = 50.0$, $r_\alpha = 0.5$, $a = -0.5$, $w_h/w_\alpha = 0.2$, Flutter Velocity, $\bar{U} = 4.53$

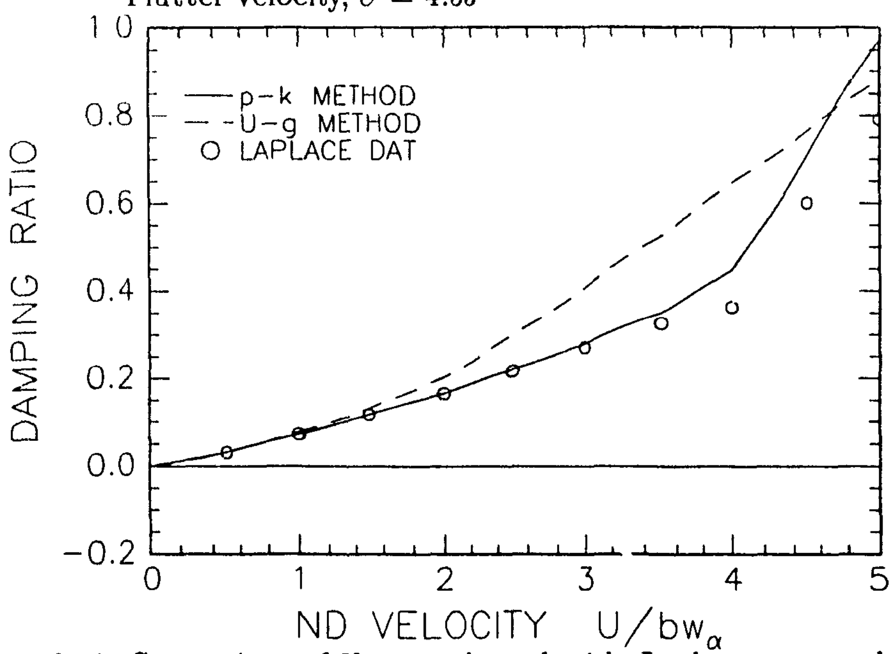


Figure 8a-2 Comparison of $U-g$ and $p-k$ with *Laplace* generated graphs of Damping Ratio, Root 2, versus Non-Dimensional Velocity, $\bar{U}$. Case Study #1 : $\mu = 50.0$, $r_\alpha = 0.5$, $a = -0.5$, $w_h/w_\alpha = 0.2$,

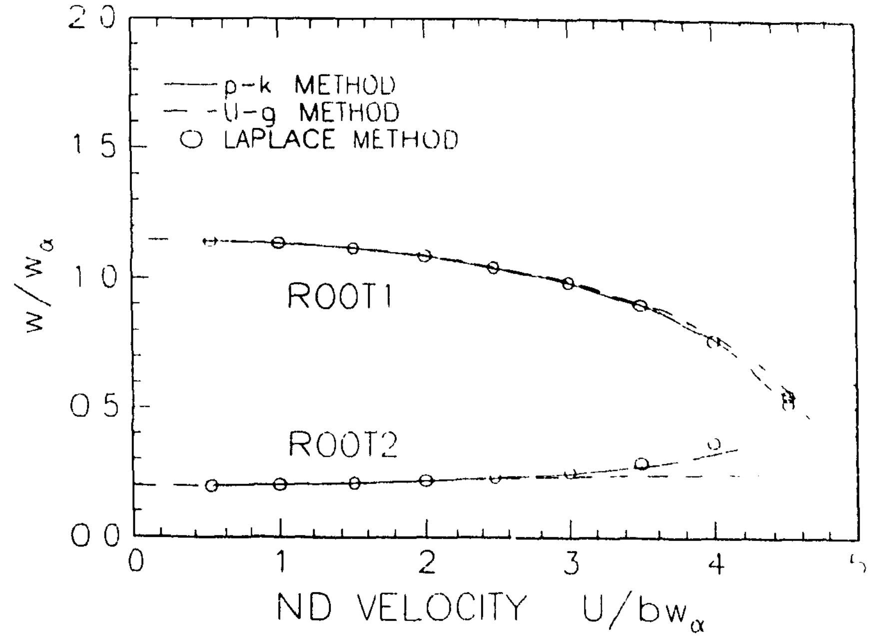


Figure 8a-3 Comparison of $U-g$ and $p-k$ with *Laplace* generated graphs of Frequency Ratio, w/w_α, versus Non-Dimensional Velocity, $\bar{U}$. Case Study #1 : $\mu = 50.0$, $r_\alpha = 0.5$, $a = -0.5$, $w_h/w_\alpha = 0.2$,

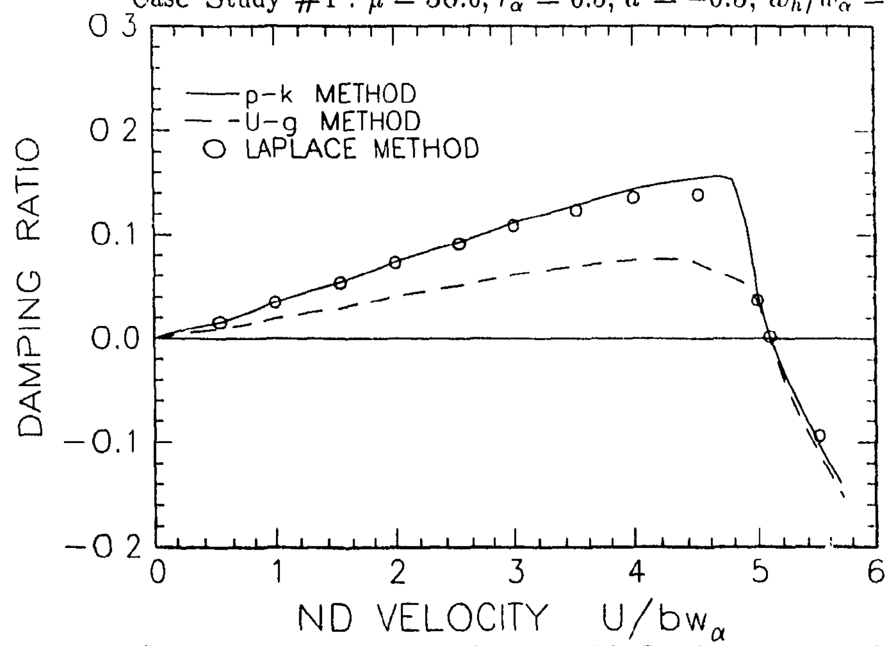


Figure 8b-1 Comparison of $U-g$ and $p-k$ with *Laplace* generated graphs of Damping Ratio, Root 1, versus Non-Dimensional Velocity, $\bar{U}$. Case Study #2 : $\mu = 50.0$, $r_\alpha = 0.6$, $a = -0.6$, $w_h/w_\alpha = 0.4$, Flutter Velocity, $\bar{U} = 5.11$

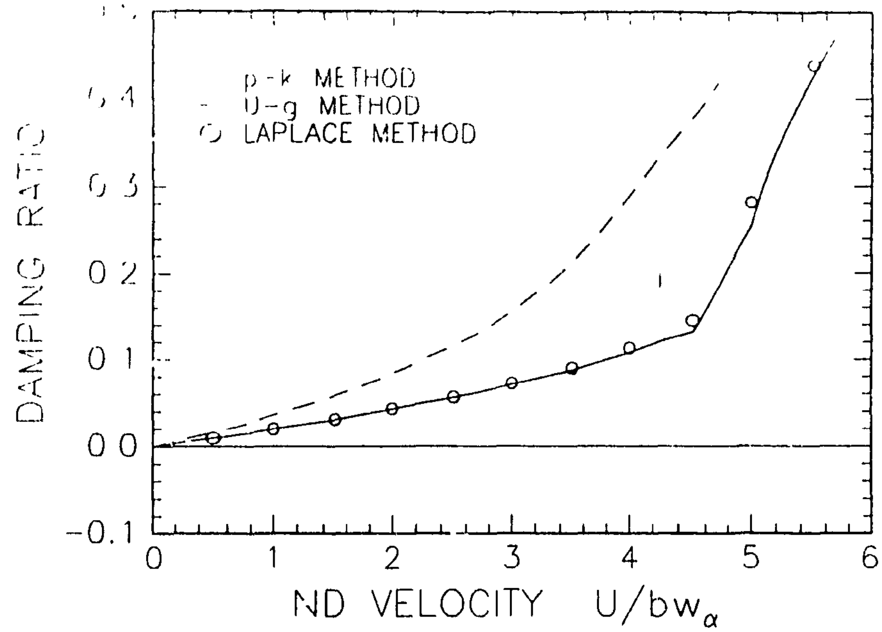


Figure 8b-2 Comparison of $U-g$ and $p-k$ with *Laplace* generated graphs of Damping Ratio, Root 2, versus Non-Dimensional Velocity, $\bar{U}$. Case Study #2 : $\mu = 50.0$, $r_\alpha = 0.6$, $a = -0.6$, $w_h/w_\alpha = 0.4$,

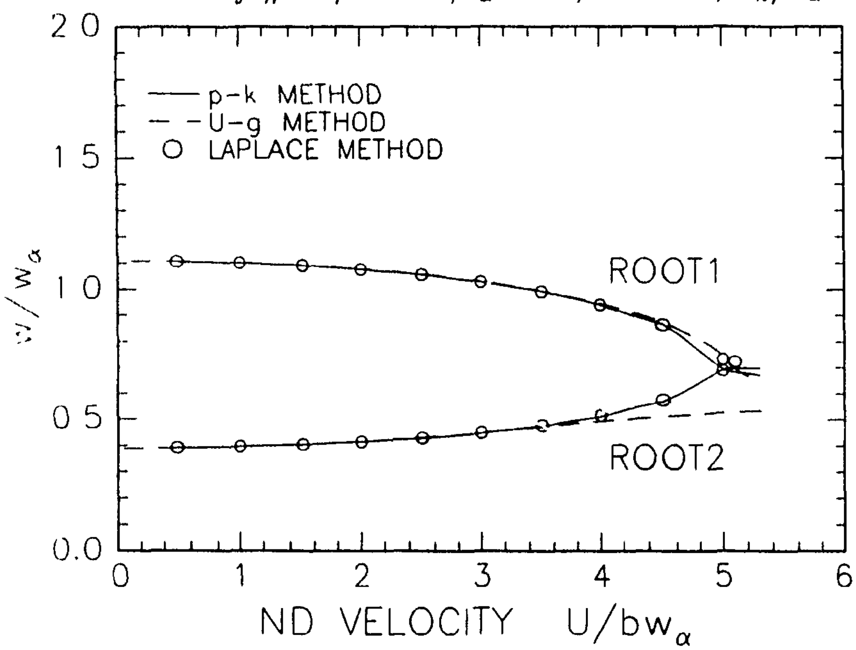


Figure 8b-3 Comparison of $U-g$ and $p-k$ with *Laplace* generated graphs of Frequency Ratio, w/w_α, versus Non-Dimensional Velocity, $\bar{U}$. Case Study #2 : $\mu = 50.0$, $r_\alpha = 0.6$, $a = -0.6$, $w_h/w_\alpha = 0.4$,

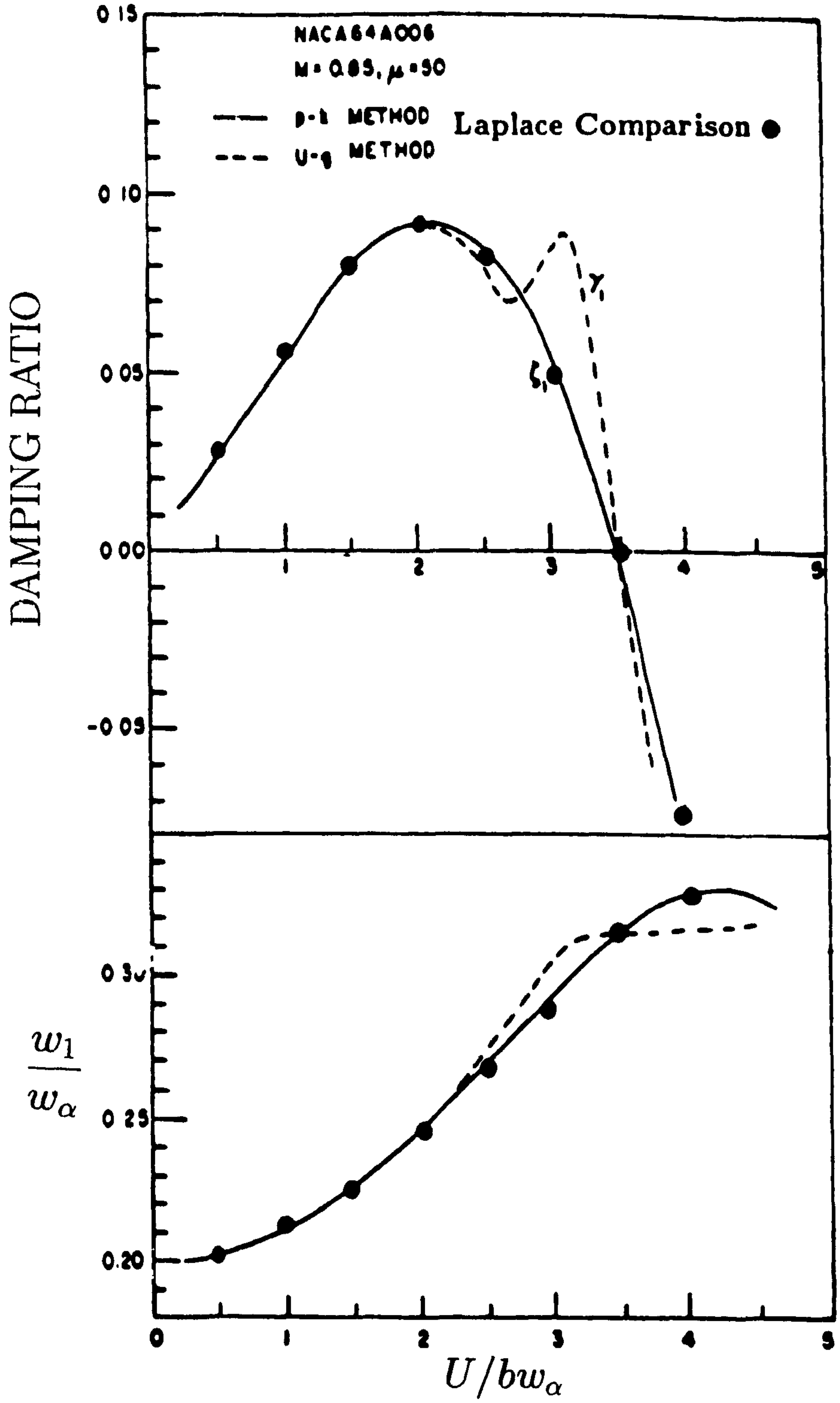


Figure 9a-1 Comparison of $U - g$ and $p - k$ with *Laplace* generated graphs of Root1 Damping and Frequency Ratio versus Non-Dimensional Velocity, $\bar{U}$, at $M = 0.85$ ($\mu = 50.0$, $r_\alpha = 0.5$, $a = -0.5$, $w_h/w_\alpha = 0.2$; Flutter Velocity, $\bar{U} = 3.46$).

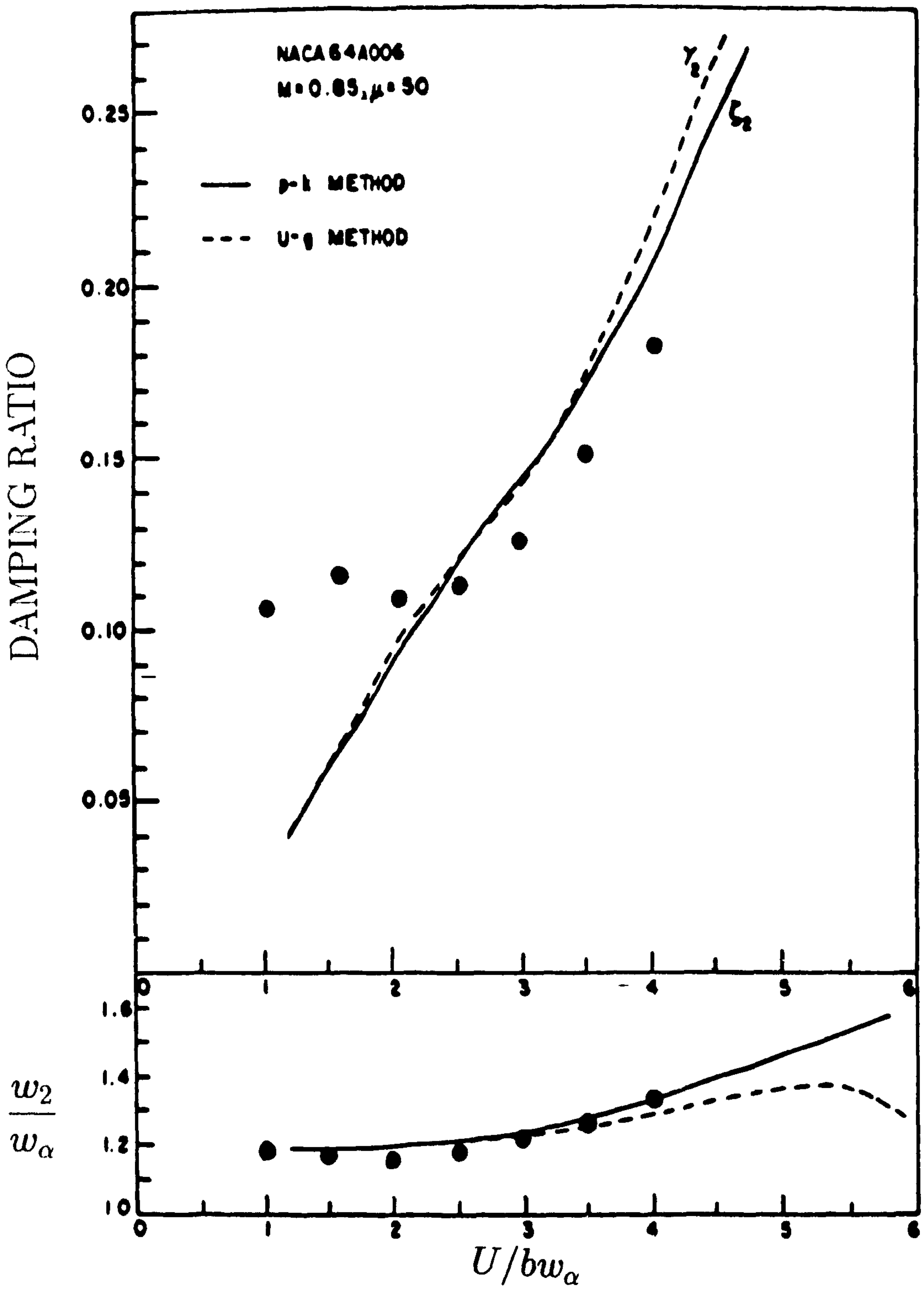


Figure 9a-2 Comparison of $U-g$ and $p-k$ with *Laplace* generated graphs of Root2 Damping and Frequency Ratio versus Non-Dimensional Velocity, $\bar{U}$, at $M = 0.85$. ($\mu = 50.0$, $r_\alpha = 0.5$, $a = -0.5$, $w_h/w_\alpha = 0.2$; Flutter Velocity, $\bar{U} = 3.46$).

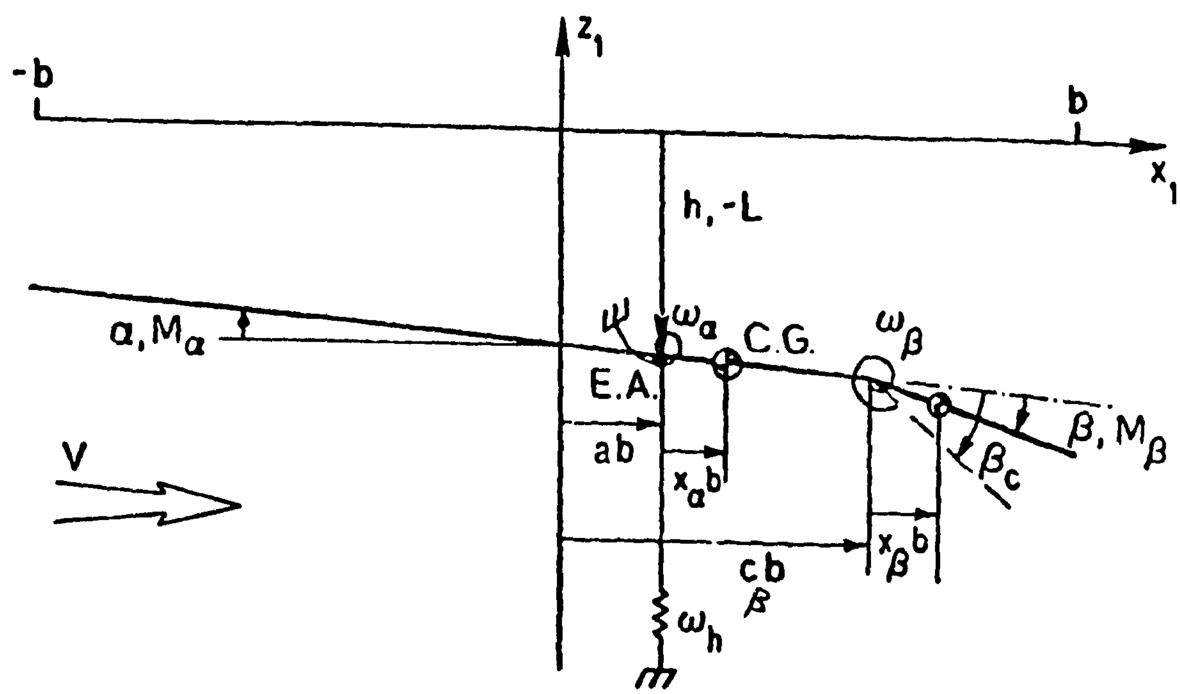


Figure 10 Schematic Representation of Airfoil with Freedom to Move in Torsional and Bending Directions with a Flap (3DOF). (Karpel 1981)

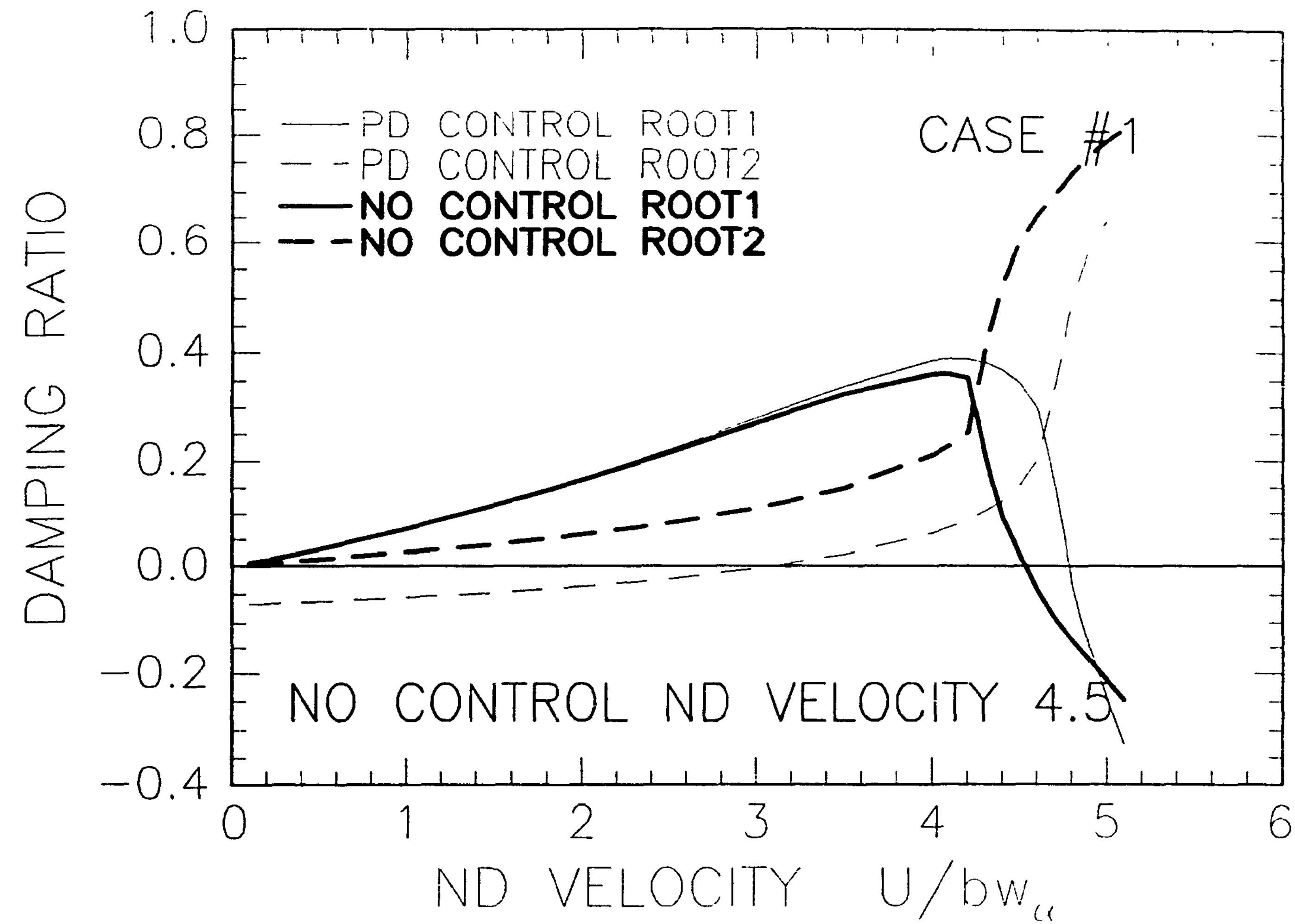


Figure 11 Implementation of Active Control for Incompressible Regime.

Case Study #1 : $\mu = 50.0$, $r_\alpha = 0.5$, $a = -0.5$, $w_h/w_\alpha = 0.2$

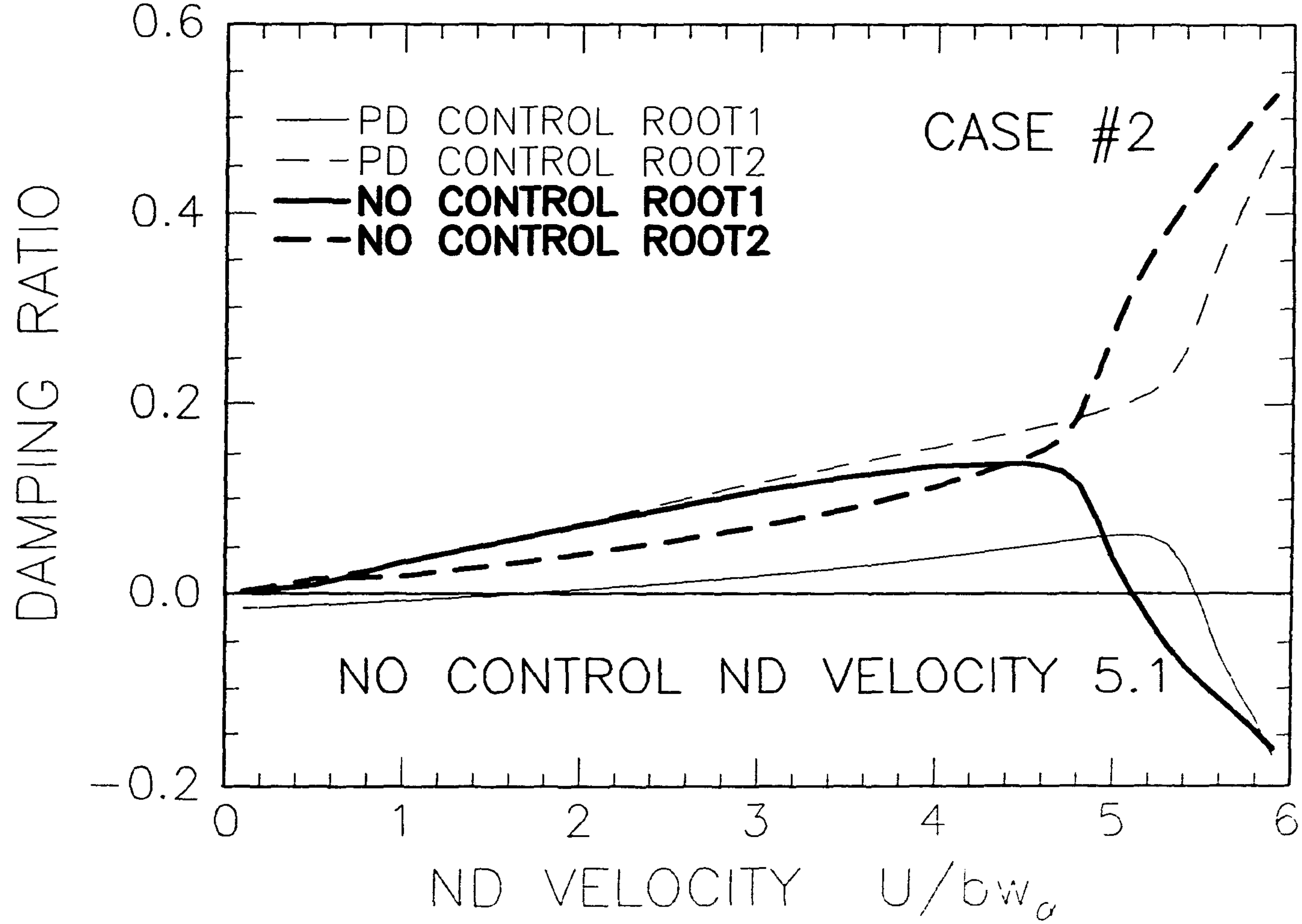


Figure 12 Implementation of Active Control for Incompressible Regime.

Case Study #2 : $\mu = 50.0$, $r_\alpha = 0.6$, $a = -0.6$, $w_h/w_\alpha = 0.1$

Appendix A : Objective Error Function

Many forms of an error function can be used

$Q(ik) = F + iG$ rational function approximation

$Q(ik) = F + iG$ tabular data (discrete values of reduced frequency)

Error #1 (one used in study)

$$err(ik) = \sum \frac{|\bar{Q}(ik) - Q(ik)|^2}{max\{1, |Q(ik)|^2\}}$$

Error #2

$$err(ik) = \sum \frac{|\bar{Q}(ik) - Q(ik)|^2}{\{|Q(ik)|^2\}}$$

Error #3

$$err(ik) = \sum \left\{ \frac{(\bar{F} - F)^2}{F^2} + \frac{(\bar{G} - G)^2}{G^2} \right\}$$

Error #4

$$err(ik) = \sum \left\{ abs\left[\frac{(\bar{F} - F)}{F}\right] + abs\left[\frac{(\bar{G} - G)}{G}\right] \right\}$$

where $|Q(ik)|^2 = F^2 + G^2$

other variations of the error objective function could include

Error #5

$$err(ik) = \sum \frac{|\, Q(ik) - Q(ik)\,|^2}{max\,\{1, |\, Q(ik)\,|^2\}\; k}$$

Error #6

$$err(ik) = \sum \frac{|\, Q(ik) - Q(ik)\,|^2}{max\,\{1, |\, Q(ik)\,|^2\}\; k^2}$$

where $|\, Q(ik)\,|^2 = F^2 + G^2$.

Analysis has showed that the error function, given by Error #1, resulted in the best comparison of $\bar{U}$, $w \,/\, w_\alpha$ and k to Lee (1984). For example, the table below considers a reduced frequency range of $0.0 \leq k \leq 0.5$, with forced constraints for all aerodynamic coefficients at $k = 0.0$.

Error Type	*Reduced Frequency Range, k*	$\bar{U}$	$w \,/\, w_\alpha$	k_{cr}
#1	$0.0 \leq k \leq 0.5$	3.45	0.309	0.090
#2	$0.0 \leq k \leq 0.5$	3.36	0.303	0.090
#3	$0.0 \leq k \leq 0.5$	3.42	0.305	0.089
#4	$0.0 \leq k \leq 0.5$	3.38	0.302	0.089

Note: Depending on the original nature of the tabular data, one of the other presented error functions may have resulted in better results. It was decided to implement the same error function as used by Adams and Tiffany (1988), namely Error Type #1

Appendix B : Linear Coefficient Optimization - Least Squares Methodology

The linear coefficients, $A_0, A_1, \ldots, A_6$, are optimized by utilizing a least square methodology. In order to obtain a minimum error function, the derivative of the error function is taken with respect to each linear coefficient, and written as

$$\frac{\partial err(ik)}{\partial(A_j)} = 0 \ .$$

The error function is defined as

$$err(ik) = \sum \frac{|\ \tilde{Q}(ik) - Q(ik)\ |^2}{M(ik)} \ ,$$

where

$$M(ik) = max\left\{1, |\ Q(ik)\ |^2\right\} \ .$$

The quantities $\tilde{Q}(ik)$ and $Q(ik)$ represent the rational function approximation and the exact value, respectively, at a given reduced frequency. The term $M(ik)$ is used to normalize the aerodynamic tabular data, such that certain points do not receive larger than normal weighting. Note, the error function is defined as the total normalized sum for all the given values of the unsteady aerodynamic tabular data.

The following analysis considers an approximating function of only three lag terms, hence linear coefficients $A_0, A_1, \ldots, A_5$. The extension to additional lag terms is straight forward.

Making the following substitutions for $\tilde{Q}(ik)$ and $Q(ik)$,

$$Q(ik) = A_0 + A_1 ik + A_2(ik)^2 + A_3\frac{ik}{ik + b_1} + A_4\frac{ik}{ik + b_2} + A_5\frac{ik}{ik + b_3}$$

and

$$Q(ik) = F + iG \, ,$$

into the above equations gives

$$\frac{\partial \left[\sum \frac{\left| \left(A_0 + A_1 ik + A_2 (ik)^2 + A_3 \frac{ik}{ik+b_1} + A_4 \frac{ik}{ik+b_2} + A_5 \frac{ik}{ik+b_3} \right) - (F + iG) \right|^2}{M(ik)} \right]}{\partial (A_j)} = 0$$

Using $i^2 = -1$, and cleaning up the denominator of terms of the form $ik + b_j$, where $j = 1 \; to \; 3$, gives

$$\frac{ik}{ik + b_j} = \frac{ik}{ik + b_j} \left(\frac{ik - b_j}{ik - b_j} \right) = \frac{k^2 + ikb_j}{k^2 + b_j^2}$$

The numerator term, given by

$$\left[\sum \frac{\left| \left(A_0 + A_1 ik + A_2 (ik)^2 + A_3 \frac{ik}{ik+b_1} + A_4 \frac{ik}{ik+b_2} + A_5 \frac{ik}{ik+b_3} \right) - (F + iG) \right|^2}{M(ik)} \right] ,$$

can then be rewritten to give

$$\left[\sum \frac{\left| \left(A_0 + A_1 ik - A_2 (k)^2 + A_3 \frac{k^2+ikb_1}{k^2+b_1^2} + A_4 \frac{k^2+ikb_2}{k^2+b_2^2} + A_5 \frac{k^2+ikb_3}{k^2+b_3^2} \right) - (F + iG) \right|^2}{M(ik)} \right]$$

This can be further simplified to

$$\sum \frac{\left| (a + ib) \right|^2}{M(ik)} = \sum \frac{(a^2 + b^2)}{M(ik)} ,$$

where

$$a = A_0 - A_2 k^2 + A_3 \frac{k^2}{k^2 + b_1^2} + A_4 \frac{k^2}{k^2 + b_2^2} + A_5 \frac{k^2}{k^2 + b_3^2} - F ,$$

and

$$b = A_1 k + A_3 \frac{kb_1}{k^2 + b_1^2} + A_4 \frac{kb_2}{k^2 + b_2^2} + A_5 \frac{kb_3}{k^2 + b_3^2} - G$$

This can be further simplified to give

$$a^2 = A_0^2 + A_2^2 k^4 + A_3^2 B_1^2 + A_4^2 B_2^2 + A_5^2 B_3^2 - 2A_0 A_2 k^2 - 2A_0 F + 2A_0 A_3 B_1 +$$

$$2A_0A_4B_2+2A_0A_5B_3+2A_2k^2F-2FA_3B_1-2FA_4B_2-2FA_5B_3+2A_3B_1A_4B_2+2A_3B_1A_5B_3+$$

$$2A_4B_2A_5B_3+F^2\ ,$$

and

$$b^2 = A_1^2k^2 + A_3^2\frac{B_1^2b_1^2}{k^2} + A_4^2\frac{B_2^2b_2^2}{k^2} + A_5^2\frac{B_3^2b_3^2}{k^2} - 2A_1kG + 2A_1A_3B_1b_1 + 2A_1A_4B_2b_2 +$$

$$2A_1A_5B_3b_3 - 2GA_3\frac{B_1b_1}{k} - 2GA_4\frac{B_2b_2}{k} - 2GA_5\frac{B_3b_3}{k} + 2A_3A_4\frac{B_1b_1B_2b_2}{k^2} + 2A_3A_5\frac{B_1b_1B_3b_3}{k^2} +$$

$$2A_4A_5\frac{B_2b_2B_3b_3}{k^2}\ ,$$

where

$$B_j = \frac{k^2}{k^2+b_j^2}\ ,\ j = 1\ to\ 3\ .$$

Hence, taking the derivative of the error function with respect to the linear coefficient A_0 gives

$$\left[\sum\frac{2A_0-2A_2k^2-2F+2A_3B_1+2A_4B_2+2A_5B_3}{M(ik)}\right] = 0\ ,$$

or

$$\sum\frac{A_0-A_2k^2+A_3B_1+A_4B_2+A_5B_3}{M(ik)} = \sum\frac{F}{M(ik)}\ .$$

This can also be written as

$$[A']\{x\} = [B']\ ,$$

where

$$[A'] = \left[\ 1\ ,\ \ 0\ ,\ -k^2\ ,\ \ B_1\ ,\ \ B_2\ ,\ \ B_3\right],$$

$$[B'] = [\ F\ ]$$

and

$$\{x\} = \{\ A_0,\ A_1,\ A_2,\ \cdots,\ A_5\ \}\ .$$

Similarly, taking the derivative with respect to the coefficients A_j, $j = 1$ *to* 5 yields a set of linear algebraic equations , that can be written as

$$[A]\,\{x\} = [B]$$

where

$$[A] = \sum \frac{1}{M(ik)} \begin{bmatrix} 1 & 0 & -k^2 & B_1 & \cdots & B_3 \\ 0 & k^2 & 0 & B_1 b_1 & \cdots & B_3 b_3 \\ -k^2 & 0 & k^3 & -k^2 B_1 & \cdots & -k^2 B_3 \\ B_1 & B_1 b_1 & -k^2 B_1 & B_1 B_1 (1 + \frac{b_1 b_1}{k^2}) & \cdots & B_1 B_3 (1 + \frac{b_1 b_3}{k^2}) \\ B_2 & B_2 b_2 & -k^2 B_2 & B_2 B_1 (1 + \frac{b_2 b_1}{k^2}) & \cdots & B_2 B_3 (1 + \frac{b_2 b_3}{k^2}) \\ B_3 & B_3 b_3 & -k^2 B_3 & B_3 B_1 (1 + \frac{b_3 b_1}{k^2}) & \cdots & B_3 B_3 (1 + \frac{b_3 b_3}{k^2}) \end{bmatrix}$$

$$[B] = \sum \frac{1}{M(ik)} \left[F\,, \;\; kG \;\; -k^2 F\,, \;\; F B_1 + \frac{G B_1 b_1}{k}\,, \;\; \cdots\,, \;\; F B_3 + \frac{G B_3 b_3}{k} \right]^T .$$

and

$$\{x\} = \{\ A_0,\ A_1,\ A_2,\ \cdots,\ A_5\ \}\ .$$

Appendix C : Nonlinear Coefficient Optimization - Simplex Method

The best way to explain the simplex method is through an example. Consider a function f consisting of two independent variables ($n = 2$), say X_1 and X_2. This problem requires the function evaluation $f(X_1, X_2)$ at three $(n + 1)$ different sets of values to produce three vertices. Let these initial simplex vertices be denoted as A, B and C, which would form a triangular simplex (see figure). In general, the simplex will not be regular in shape. The simplex method then considers the function evaluations at these three points, and determines the largest, say for argument, it is $f(A)$, and that $f(C) < f(B) < f(A)$. It now takes a series of steps, to move the point of the simplex, which has the largest value, A, through the opposite face of the simplex, to produce a new point that will replace A. The initial step taken, is to reflect the point A through the *geometric* centroid of the two vertices, B and C (the midpoint), resulting in a reflection point E. In general, the centroid is defined as the centroid of all the points $(X_1)_i$, $(X_2)_i$, ... ; not including the point of the highest function value. Note, that the lengths AD and DE are equal. Depending on the magnitude of the function value at E, the simplex will do an extension to F, contraction to H, a lesser contraction to G, or maintain the initial reflection to form BCE. The process continues until convergence occurs.

Criteria:

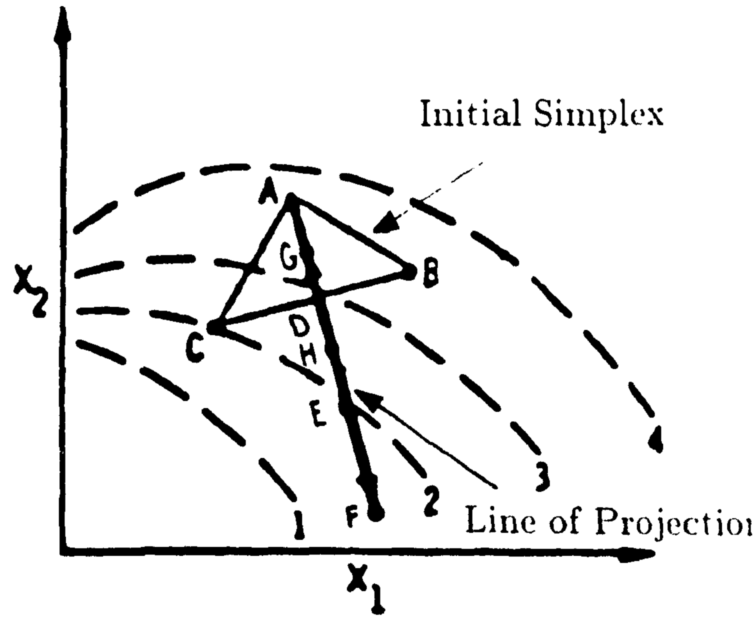
1) if $f(E) < f(C)$ locate F (extension point) along AD of length DE

2) if $f(E) > f(A)$ locate G (interior contraction point) midpoint of AD

3) if $f(B) < f(E) < f(A)$ locate H (exterior contraction point) midpoint of DE

4) if $f(C) < f(E) < f(B)$ keep E (reflection point) new simplex is BCE

Simplex Method : $f(X_1, X_2)$

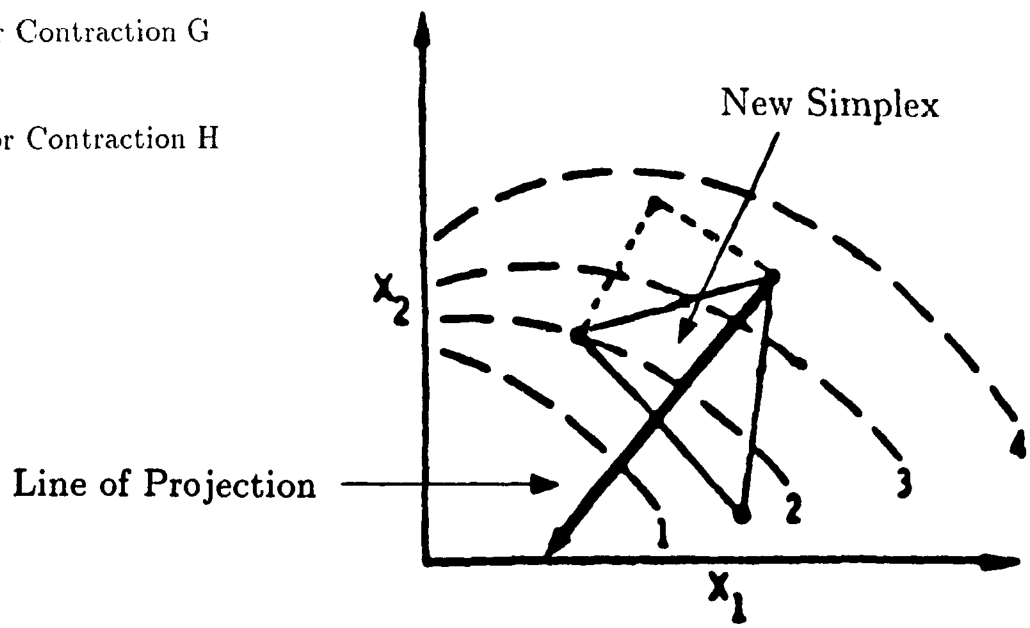


Reflection Point E

Extension Point F

Interior Contraction G

Exterior Contraction H



Appendix D : Function Optimization for Incompressible Regime

Nasa-Langley approximate in Laplace form

$$[N] = [A_0] + \frac{b}{U}[A_1]\,s + (\frac{b}{U})^2 [A_2]\,s^2 +$$

$$[A_3]\frac{s}{s+\frac{U}{b}b_1} + [A_4]\frac{s}{s+\frac{U}{b}b_2} + [A_5]\frac{s}{s+\frac{U}{b}b_3} + [A_6]\frac{s}{s+\frac{U}{b}b_4}\ .$$

Approximation of Theodorsen Function $C(k)$

$$[N] = [0.99828] + \frac{b}{U}[-0.00004]\,s + (\frac{b}{U})^2 [0.00000]\,s^2 +$$

$$[-0.04120]\frac{s}{s+\frac{U}{b}(0.01544)} + [-0.16289]\frac{s}{s+\frac{U}{b}(0.08429)} +$$
$$[-0.22909]\frac{s}{s+\frac{U}{b}(0.25595)} + [-0.06459]\frac{s}{s+\frac{U}{b}(0.76511)}\ .$$

Approximation of $C(k) \ * \ ik$

$$[N] = [-0.00001] + \frac{b}{U}[0.50037]\,s + (\frac{b}{U})^2 [-0.00004]\,s^2 +$$

$$[0.00061]\frac{s}{s+\frac{U}{b}(0.01544)} + [0.01380]\frac{s}{s+\frac{U}{b}(0.08429)} +$$
$$[0.05848]\frac{s}{s+\frac{U}{b}(0.25595)} + [0.04963]\frac{s}{s+\frac{U}{b}(0.76511)}\ .$$

note : lag terms are equivalent

SEQUENTIAL SIMPLEX : PROBLEM MINIMIZATION
INCOMPRESSIBLE REGIME
GLOBAL OPTIMIZATION
RANGE 0.00 < k < 10.00

N (1, 1) = C(k)

k	EXACT REAL	IMAG	APPROXIMATE REAL	IMAG
10.000000,	.50060,	-.01240,	.50057,	-.01257,
6.000000,	.50170,	-.02060,	.50182,	-.02047,
4.000000,	.50370,	-.03050,	.50372,	-.03025,
3.000000,	.50630,	-.04000,	.50619,	-.03976,
2.000000,	.51290,	-.05770,	.51271,	-.05764,
1.500000,	.52100,	-.07360,	.52083,	-.07373,
1.200000,	.53000,	-.08770,	.52994,	-.08798,
1.000000,	.53940,	-.10030,	.53959,	-.10051,
.990000,	.54000,	-.10100,	.54019,	-.10122,
.880000,	.54740,	-.10950,	.54767,	-.10963,
.800000,	.55410,	-.11650,	.55443,	-.11655,
.770000,	.55700,	-.11930,	.55733,	-.11934,
.660000,	.56990,	-.13080,	.57012,	-.13063,
.600000,	.57880,	-.13780,	.57895,	-.13757,
.560000,	.58570,	-.14280,	.58579,	-.14253,
.550000,	.58760,	-.14410,	.58764,	-.14381,
.500000,	.59790,	-.15070,	.59787,	-.15049,
.440000,	.61300,	-.15920,	.61278,	-.15904,
.400000,	.62500,	-.16500,	.62477,	-.16499,
.340000,	.64690,	-.17380,	.64683,	-.17397,
.330000,	.65120,	-.17520,	.65108,	-.17543,
.325000,	.65350,	-.17590,	.65328,	-.17614,
.320000,	.65580,	-.17660,	.65552,	-.17686,
.315000,	.65810,	-.17730,	.65782,	-.17756,
.310000,	.66040,	-.17790,	.66017,	-.17825,
.305000,	.66270,	-.17860,	.66257,	-.17894,
.300000,	.66500,	-.17930,	.66502,	-.17961,
.240000,	.69890,	-.18620,	.69927,	-.18633,
.220000,	.71250,	-.18770,	.71297,	-.18770,
.200000,	.72760,	-.18860,	.72802,	-.18841,
.160000,	.76280,	-.18760,	.76288,	-.18703,
.120000,	.80630,	-.18010,	.80584,	-.17977,
.110000,	.81880,	-.17660,	.81822,	-.17652,
.100000,	.83200,	-.17230,	.83138,	-.17244,
.080000,	.86040,	-.16040,	.86027,	-.16092,
.060000,	.89200,	-.14260,	.89253,	-.14285,
.050000,	.90900,	-.13050,	.90957,	-.13041,
.040000,	.92670,	-.11600,	.92684,	-.11525,
.025000,	.95450,	-.08720,	.95305,	-.08710,
.010000,	.98240,	-.04820,	.98349,	-.04764,

continued

N (1, 2) = C(k)*ik

k	EXACT REAL	IMAG	APPROXIMATE REAL	IMAG
10.000000,	.12400,	5.00600,	.12582,	5.00914,
6.000000,	.12360,	3.01020,	.12292,	3.01116,
4.000000,	.12200,	2.01480,	.12110,	2.01468,
3.000000,	.12000,	1.51890,	.11938,	1.51835,
2.000000,	.11540,	1.02580,	.11536,	1.02526,
1.500000,	.11040,	.78150,	.11065,	.78113,
1.200000,	.10524,	.63600,	.10561,	.63585,
1.000000,	.10030,	.53940,	.10053,	.53954,
.990000,	.09999,	.53460,	.10023,	.53474,
.880000,	.09636,	.48171,	.09649,	.48191,
.800000,	.09320,	.44328,	.09325,	.44352,
.770000,	.09186,	.42889,	.09190,	.42912,
.660000,	.08633,	.37613,	.08621,	.37626,
.600000,	.08268,	.34728,	.08253,	.34735,
.560000,	.07997,	.32799,	.07980,	.32802,
.550000,	.07926,	.32318,	.07908,	.32318,
.500000,	.07535,	.29895,	.07522,	.29892,
.440000,	.07005,	.26972,	.06995,	.26961,
.400000,	.06600,	.25000,	.06597,	.24989,
.340000,	.05909,	.21995,	.05912,	.21990,
.330000,	.05782,	.21490,	.05786,	.21483,
.325000,	.05717,	.21239,	.05722,	.21229,
.320000,	.05651,	.20986,	.05656,	.20974,
.315000,	.05585,	.20730,	.05590,	.20719,
.310000,	.05515,	.20472,	.05523,	.20463,
.305000,	.05447,	.20212,	.05455,	.20206,
.300000,	.05379,	.19950,	.05385,	.19948,
.240000,	.04469,	.16774,	.04469,	.16780,
.220000,	.04129,	.15675,	.04127,	.15683,
.200000,	.03772,	.14552,	.03766,	.14558,
.160000,	.03002,	.12205,	.02991,	.12204,
.120000,	.02161,	.09676,	.02156,	.09669,
.110000,	.01943,	.09007,	.01941,	.09000,
.100000,	.01723,	.08320,	.01723,	.08313,
.080000,	.01283,	.06883,	.01286,	.06882,
.060000,	.00856,	.05352,	.00855,	.05356,
.050000,	.00653,	.04545,	.00650,	.04548,
.040000,	.00464,	.03707,	.00459,	.03708,
.025000,	.00218,	.02386,	.00215,	.02382,
.010000,	.00048,	.00982,	.00046,	.00983,

Appendix E : Rational Function Optimization - Transonic Regime

Comparison between the rational function approximation of the aerodynamic derivatives and the true aerodynamics as a function of reduced frequency, $0.025 < k < 1.000$, at $M = 0.85$.

C_{l_h}, C_{l_α}, C_{m_h} and C_{m_α}

SEQUENTIAL SIMPLEX : PROBLEM MINIMIZATION

TRANSONIC REGIME

C_{l_h}

k	EXACT		APPROXIMATE	
	REAL	IMAG	REAL	IMAG
.025000,	.23000,	.56000,	.21658,	.56611,
.050000,	.40000,	.87000,	.41419,	.84979,
.075000,	.58000,	1.07000,	.59756,	1.06853,
.100000,	.74000,	1.22000,	.75517,	1.24000,
.150000,	1.00000,	1.49000,	.98456,	1.50109,
.200000,	1.17000,	1.69000,	1.11518,	1.72130,
.250000,	1.23000,	1.92000,	1.17742,	1.94017,
.300000,	1.22000,	2.12000,	1.19509,	2.17025,
.350000,	1.19000,	2.48000,	1.18273,	2.41433,
.400000,	1.13000,	2.63000,	1.14896,	2.67229,
.450000,	1.06000,	2.93000,	1.09904,	2.94340,
.500000,	.99000,	3.25000,	1.03633,	3.22697,
.550000,	.92000,	3.54000,	.96317,	3.52249,
.600000,	.88000,	3.89000,	.88125,	3.82966,
.650000,	.79000,	4.20000,	.79193,	4.14829,
.700000,	.68000,	4.52000,	.69631,	4.47832,
.750000,	.57000,	4.88000,	.59538,	4.81976,
.800000,	.47000,	5.22000,	.49004,	5.17262,
.850000,	.39000,	5.53000,	.38110,	5.53699,
.900000,	.29000,	5.87000,	.26936,	5.91290,
.950000,	.20000,	6.25000,	.15556,	6.30041,
1.000000,	.10000,	6.58000,	.04045,	6.69956,

C_{m_h}

k	EXACT		APPROXIMATE	
	REAL	IMAG	REAL	IMAG
.025000,	.01000,	-.03000,	-.01235,	-.01705,
.050000,	.02000,	-.06000,	.03307,	-.05976,
.075000,	.03000,	-.10000,	.03738,	-.12445,
.100000,	.03000,	-.14000,	.02280,	-.17765,
.150000,	.01000,	-.23000,	-.01110,	-.24884,
.200000,	-.01000,	-.33000,	-.02820,	-.29719,
.250000,	-.04000,	-.40000,	-.02752,	-.33989,
.300000,	-.06000,	-.41000,	-.01311,	-.38366,
.350000,	-.04000,	-.43000,	.01179,	-.43100,
.400000,	.00000,	-.45000,	.04502,	-.48292,
.450000,	.07000,	-.49000,	.08512,	-.53991,
.500000,	.14000,	-.55000,	.13108,	-.60228,
.550000,	.21000,	-.64000,	.18211,	-.67029,
.600000,	.27000,	-.76000,	.23757,	-.74417,
.650000,	.34000,	-.86000,	.29690,	-.82415,
.700000,	.39000,	-.97000,	.35958,	-.91043,
.750000,	.46000,	-1.06000,	.42512,	-1.00321,
.800000,	.53000,	-1.13000,	.49304,	-1.10268,
.850000,	.57000,	-1.19000,	.56289,	-1.20898,
.900000,	.61000,	-1.28000,	.63420,	-1.32226,
.950000,	.65000,	-1.40000,	.70653,	-1.44262,
1.000000,	.68000,	-1.53000,	.77943,	-1.57016,

continued

C_{l_α}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.025000,	10.79000,	-4.01000,	10.80926,	-4.16547,
.050000,	9.10000,	-4.21000,	9.23528,	-4.01598,
.075000,	8.00000,	-3.82000,	8.06712,	-3.82265,
.100000,	7.21000,	-3.49000,	7.20308,	-3.55867,
.150000,	6.09000,	-2.96000,	6.03660,	-2.93090,
.200000,	5.41000,	-2.29000,	5.34470,	-2.29399,
.250000,	4.93000,	-1.79000,	4.93036,	-1.71875,
.300000,	4.62000,	-1.28000,	4.67882,	-1.21716,
.350000,	4.43000,	-.81000,	4.52562,	-.78120,
.400000,	4.36000,	-.41000,	4.43443,	-.39950,
.450000,	4.35000,	-.03000,	4.38410,	-.06209,
.500000,	4.40000,	.27000,	4.36177,	.23873,
.550000,	4.43000,	.55000,	4.35925,	.50872,
.600000,	4.50000,	.74000,	4.37104,	.75210,
.650000,	4.50000,	1.00000,	4.39332,	.97197,
.700000,	4.48000,	1.18000,	4.42326,	1.17064,
.750000,	4.48000,	1.31000,	4.45867,	1.34983,
.800000,	4.48000,	1.47000,	4.49781,	1.51085,
.850000,	4.49000,	1.63000,	4.53923,	1.65470,
.900000,	4.56000,	1.80000,	4.58165,	1.78215,
.950000,	4.60000,	1.91000,	4.62398,	1.89385,
1.000000,	4.62000,	2.02000,	4.66521,	1.99031,

C_{m_α}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.025000,	-.43700,	-.04100,	-.43992,	-.01892,
.050000,	-.45100,	-.12100,	-.45702,	-.16849,
.075000,	-.48200,	-.26200,	-.50534,	-.27825,
.100000,	-.52300,	-.34800,	-.56402,	-.35625,
.150000,	-.68100,	-.51100,	-.66848,	-.45194,
.200000,	-.81700,	-.58000,	-.74100,	-.51230,
.250000,	-.90800,	-.59800,	-.78859,	-.56393,
.300000,	-.90600,	-.54800,	-.82047,	-.61610,
.350000,	-.86900,	-.57400,	-.84285,	-.67156,
.400000,	-.82300,	-.64000,	-.85947,	-.73091,
.450000,	-.78600,	-.74800,	-.87262,	-.79411,
.500000,	-.77800,	-.90000,	-.88376,	-.86094,
.550000,	-.81800,	-.99200,	-.89383,	-.93120,
.600000,	-.85200,	-1.08000,	-.90351,	-1.00472,
.650000,	-.87800,	-1.14300,	-.91328,	-1.08137,
.700000,	-.89200,	-1.23000,	-.92354,	-1.1610[illegible]
.750000,	-.93000,	-1.30700,	-.93459,	-1.24377,
.800000,	-.97800,	-1.38100,	-.94671,	-1.32940,
.850000,	-1.00800,	-1.43900,	-.96013,	-1.41796,
.900000,	-1.03100,	-1.49500,	-.97508,	-1.50942,
.950000,	-1.04300,	-1.53400,	-.99176,	-1.60377,
1.000000,	-1.05800,	-1.56400,	-1.01035,	-1.70098,

SEQUENTIAL SIMPLEX : PROBLEM MINIMIZATION

TRANSONIC REGIME

APPROXIMANT FUNCTION N (1, 1) = C_{l_h}

A0	A1	A2	A3	A4	A5	A6
-7907.23428	13.851605	-1.182972	7907.335646	.299359	1.150099	-22.698018

lag1	lag2	lag3	lag4	error	
.000000273	.042626307	.132115292	2.737040606	.065486271	.065486271

APPROXIMANT FUNCTION N (1, 2) = C_{l_α}

A0	A1	A2	A3	A4	A5	A6
170329.1714	-4.694754	1.921693	-170317.2824	-3.613661	-4.501366	23.17

lag1	lag2	lag3	lag4	error	
.000000273	.042626307	.132115292	2.737040606	.181049841	.306154896

APPROXIMANT FUNCTION N (2, 1) = C_{m_h}

A0	A1	A2	A3	A4	A5	A6
3001.540919	-6.292480	.797560	-3001.600689	.227531	-.334033	14.757

lag1	lag2	lag3	lag4	error	
.000000273	.042626307	.132115292	2.737040606	.059618785	.125105055

APPROXIMANT FUNCTION N (2, 2) = C_{m_α}

A0	A1	A2	A3	A4	A5	A6
-6706.7085	-3.347323	.739087	6706.26103	.105898	-.565209	5.315633

lag1	lag2	lag3	lag4	error	
.000000273	.042626307	.132115292	2.737040606	.159675096	.465829992

Appendix F : Function Optimization for Transonic Regime

Nasa-Langley approximate in Laplace form

$$[N] = [A_0] + \frac{b}{U}[A_1]\,s + (\frac{b}{U})^2[A_2]\,s^2 +$$

$$[A_3]\frac{s}{s+\frac{U}{b}b_1} + [A_4]\frac{s}{s+\frac{U}{b}b_2} + [A_5]\frac{s}{s+\frac{U}{b}b_3} + [A_6]\frac{s}{s+\frac{U}{b}b_4}\ .$$

Approximation of Transonic Coefficients of Lift and Moment with respect to plunge and pitch, for a reduced frequency range of $0.0 \leq k \leq 0.5$ with forced constraints for C_{l_h} and C_{m_h} at $k = 0.0$, at $M = 0.85$.

$$\begin{bmatrix} C_{l_h} & C_{l_\alpha} \\ C_{m_h} & C_{m_\alpha} \end{bmatrix} = \begin{bmatrix} 0.0000 & 14.236 \\ 0.0000 & -0.5168 \end{bmatrix} + \frac{b}{U}\begin{bmatrix} 7.3520 & 0.9578 \\ -1.9059 & -4.3942 \end{bmatrix} s + (\frac{b}{U})^2 \begin{bmatrix} -0.2652 & 1.25818 \\ -0.2936 & 2.7440 \end{bmatrix} s^2 +$$

$$\begin{bmatrix} -8561.67 & 404976 \\ 158360 & 217699 \end{bmatrix} \frac{s}{s+\frac{U}{b}(0.27893)} + \begin{bmatrix} 0.5930 & -8.14577 \\ -0.05984 & 0.16511 \end{bmatrix} \frac{s}{s+\frac{U}{b}(0.01365)} +$$

$$\begin{bmatrix} 8707.6 & -408964 \\ -159903 & -219881 \end{bmatrix} \frac{s}{s+\frac{U}{b}(0.27899)} + \begin{bmatrix} -146.23 & 3987.4 \\ 1543.9 & 2184.1 \end{bmatrix} \frac{s}{s+\frac{U}{b}(0.28587)}\ .$$

note : lag terms are equivalent

SEQUENTIAL SIMPLEX : PROBLEM MINIMIZATION
TRANSONIC REGIME M=0.85
GLOBAL OPTIMIZATION
FORCED CONSTRAINTS N(r,c)=0.0 AT k=0.0
RANGE 0.00 < k < 0.50
ERROR #1

C_{l_h}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	.00000,	.00000,	.00000,	.00000,
.025000,	.17000,	.56000,	.16792,	.54877,
.050000,	.39000,	.87000,	.41847,	.86344,
.087500,	.66000,	1.17000,	.69680,	1.16498,
.100000,	.76000,	1.23000,	.77287,	1.24726,
.125000,	.89000,	1.39000,	.90794,	1.39206,
.150000,	1.01000,	1.49000,	1.02060,	1.51454,
.175000,	1.12000,	1.61000,	1.10957,	1.62079,
.200000,	1.17000,	1.69000,	1.17459,	1.71761,
.250000,	1.25000,	1.93000,	1.23994,	1.90689,
.300000,	1.23000,	2.12000,	1.24034,	2.11637,
.350000,	1.19000,	2.39000,	1.20249,	2.35917,
.400000,	1.15000,	2.64000,	1.14690,	2.63547
.450000,	1.07000,	2.93000,	1.08659,	2.94005,
.500000,	.98000,	3.25000,	1.02883,	3.26655,

C_{l_α}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	14.70000,	.00000,	14.23567,	.00000,
.025000,	12.20000,	-4.05000,	12.21288,	-3.64541,
.050000,	9.50000,	-4.15000,	9.56427,	-4.30539,
.087500,	7.63000,	-3.70000,	7.53586,	-3.72869,
.100000,	7.22000,	-3.47000,	7.15123,	-3.52318,
.125000,	6.61000,	-3.18000,	6.57096,	-3.17275,
.150000,	6.09000,	-2.94000,	6.12655,	-2.88115,
.175000,	5.72000,	-2.59000,	5.75592,	-2.61606,
.200000,	5.41000,	-2.29000,	5.43806,	-2.35648,
.250000,	4.92000,	-1.79000,	4.94237,	-1.82723,
.300000,	4.63000,	-1.30000,	4.62099,	-1.29981,
.350000,	4.43000,	-.83000,	4.44320,	-.81140,
.400000,	4.35000,	-.41000,	4.36678,	-.38607,
.450000,	4.35000,	-.03000,	4.35258,	-.03010,
.500000,	4.40000,	.28000,	4.37100,	.26139,

continued

C_{m_h}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	.00000,	.00000,	.00000,	.00000,
.025000,	.00100,	-.03500,	-.00729,	-.03935,
.050000,	.01500,	-.06500,	-.00727,	-.06574,
.087500,	.02500,	-.12500,	.01171,	-.11930,
.100000,	.03000,	-.14500,	.01725,	-.14295,
.125000,	.02000,	-.19800,	.02145,	-.19500,
.150000,	.01500,	-.23000,	.01512,	-.24731,
.175000,	.01000,	-.28000,	.00030,	-.29389,
.200000,	-.00500,	-.32500,	-.01874,	-.33146,
.250000,	-.03500,	-.39700,	-.05219,	-.37922,
.300000,	-.06000,	-.40300,	-.06191,	-.40363,
.350000,	-.04500,	-.42500,	-.04310,	-.42338,
.400000,	-.00500,	-.44800,	-.00139,	-.45102,
.450000,	.06500,	-.49000,	.05504,	-.49191,
.500000,	.14000,	-.55000,	.11927,	-.54679,

C_{m_α}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	-.44500,	.00000,	-.51684,	.00000,
.025000,	-.45000,	-.02700,	-.48373,	-.04150,
.050000,	-.45500,	-.12200,	-.45477,	-.14213,
.087500,	-.49500,	-.31100,	-.48600,	-.31323,
.100000,	-.52000,	-.34800,	-.51147,	-.36418,
.125000,	-.58100,	-.42200,	-.57665,	-.45028,
.150000,	-.67200,	-.50800,	-.65200,	-.51226,
.175000,	-.73200,	-.54800,	-.72737,	-.55013,
.200000,	-.81000,	-.58200,	-.79435,	-.56754,
.250000,	-.90500,	-.61000,	-.88350,	-.56578,
.300000,	-.89800,	-.55000,	-.90772,	-.55768,
.350000,	-.86800,	-.57500,	-.88429,	-.57878,
.400000,	-.82400,	-.64800,	-.83843,	-.64393,
.450000,	-.79000,	-.75000,	-.79150,	-.75384,
.500000,	-.77000,	-.90300,	-.75798,	-.90251,

SEQUENTIAL SIMPLEX : PROBLEM MINIMIZATION
TRANSONIC REGIME M=0.85
GLOBAL OPTIMIZATION
FORCED CONSTRAINTS
RANGE 0.00 < k < 0.50
ERROR #1

C_{l_h}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	.00000,	.00000,	.00000,	.00000,
.025000,	.17000,	.56000,	.19383,	.55795,
.050000,	.39000,	.87000,	.43656,	.85319,
.087500,	.66000,	1.17000,	.70846,	1.17109,
.100000,	.76000,	1.23000,	.79040,	1.25555,
.125000,	.89000,	1.39000,	.93541,	1.39534,
.150000,	1.01000,	1.49000,	1.04994,	1.50834,
.175000,	1.12000,	1.61000,	1.13440,	1.60795,
.200000,	1.17000,	1.69000,	1.19295,	1.70330,
.250000,	1.25000,	1.93000,	1.25110,	1.90029,
.300000,	1.23000,	2.12000,	1.25543,	2.11892,
.350000,	1.19000,	2.39000,	1.22593,	2.36403,
.400000,	1.15000,	2.64000,	1.17512,	2.63603,
.450000,	1.07000,	2.93000,	1.11128,	2.93413,
.500000,	.98000,	3.25000,	1.04035,	3.25724,

C_{l_α}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	14.70000,	.00000,	14.70000,	.00000,
.025000,	12.20000,	-4.05000,	12.16128,	-3.97370,
.050000,	9.50000,	-4.15000,	9.41145,	-4.23129,
.087500,	7.63000,	-3.70000,	7.67286,	-3.66816,
.100000,	7.22000,	-3.47000,	7.29733,	-3.52885,
.125000,	6.61000,	-3.18000,	6.66079,	-3.25681,
.150000,	6.09000,	-2.94000,	6.14647,	-2.96663,
.175000,	5.72000,	-2.59000,	5.73753,	-2.66267,
.200000,	5.41000,	-2.29000,	5.41665,	-2.35786,
.250000,	4.92000,	-1.79000,	4.97158,	-1.77935,
.300000,	4.63000,	-1.30000,	4.70202,	-1.26197,
.350000,	4.43000,	-.83000,	4.54025,	-.80489,
.400000,	4.35000,	-.41000,	4.44697,	-.39879,
.450000,	4.35000,	-.03000,	4.39915,	-.03395,
.500000,	4.40000,	.28000,	4.38285,	.29772,

continued

C_{m_h}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	.00000,	.00000,	.00000,	.00000,
.025000,	.00100,	-.03500,	-.04127,	-.05079,
.050000,	.01500,	-.06500,	-.02884,	-.04927,
.087500,	.02500,	-.12500,	.01021,	-.12957,
.100000,	.03000,	-.14500,	.00937,	-.16165,
.125000,	.02000,	-.19800,	-.00401,	-.21692,
.150000,	.01500,	-.23000,	-.02225,	-.25906,
.175000,	.01000,	-.28000,	-.03854,	-.29126,
.200000,	-.00500,	-.32500,	-.05051,	-.31708,
.250000,	-.03500,	-.39700,	-.06049,	-.35882,
.300000,	-.06000,	-.40300,	-.05448,	-.39514,
.350000,	-.04500,	-.42500,	-.03590,	-.42987,
.400000,	-.00500,	-.44800,	-.00707,	-.46418,
.450000,	.06500,	-.49000,	.03062,	-.49834,
.500000,	.14000,	-.55000,	.07634,	-.53233,

C_{m_α}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	-.44500,	.00000,	-.44500,	.00000,
.025000,	-.45000,	-.02700,	-.47202,	-.06448,
.050000,	-.45500,	-.12200,	-.42613,	-.12063,
.087500,	-.49500,	-.31100,	-.43443,	-.33035,
.100000,	-.52000,	-.34800,	-.47130,	-.39057,
.125000,	-.58100,	-.42200,	-.56157,	-.47416,
.150000,	-.67200,	-.50800,	-.64879,	-.51749,
.175000,	-.73200,	-.54800,	-.72016,	-.53602,
.200000,	-.81000,	-.58200,	-.77337,	-.54202,
.250000,	-.90500,	-.61000,	-.83305,	-.54527,
.300000,	-.89800,	-.55000,	-.84809,	-.56012,
.350000,	-.86800,	-.57500,	-.83581,	-.59821,
.400000,	-.82400,	-.64800,	-.80830,	-.66360,
.450000,	-.79000,	-.75000,	-.77420,	-.75757,
.500000,	-.77000,	-.90300,	-.74002,	-.88020,

LAPLACE METHOD

TRANSONIC M=0.85
FORCED CONSTRAINTS
RANGE 0.00 < k < 0.50

u	r(alp)	ah	wben/wtor
50.0	.5	-.5	.2

ND VELOCITY .500

-.14470	1.28135	
-.14470	-1.28135	
-.60746	.00000	
-.67443	.00000	
-.00511	-.20192	
-.00511	.20192	
-.05524	-.00255	
-.05524	.00255	
-.02989	.00322	
-.02989	-.00322	
-.02296	.00203	
-.02296	-.00203	
perfomance index is	1.75297	.00000

ND VELOCITY 3.300

-6.29379	.00000	
-4.01529	.00000	
-.19525	1.25302	
-.19525	-1.25302	
-.00033	-.30122	
-.00033	.30122	
-.34065	-.18074	
-.34065	.18074	
-.17917	-.11029	
-.17917	.11029	
-.06246	-.06004	
-.06246	.06004	
perfomance index is	2.72761	.00000

ND VELOCITY 3.400

-6.49084	.00000	
-4.13698	.00000	
-.20562	1.26275	
-.20562	-1.26275	
.00355	-.30563	INSTABILITY
-.35100	-.18800	ND VELOCITY = 3.3[illegible]
-.35100	.18800	
.00355	.30563	
-.18377	-.11362	
-.18377	.11362	
-.06142	-.06271	
-.06142	.06271	
perfomance index is	4.31713	.00000

SEQUENTIAL SIMPLEX : PROBLEM MINIMIZATION
TRANSONIC REGIME M=0.85
GLOBAL OPTIMIZATION
FORCED CONSTRAINTS N(r,c)=0.0 AT k=0.0
RANGE 0.00 < k < 1.00
ERROR #1

C_{l_h}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	.00000,	.00000,	.00000,	.00000,
.025000,	.17000,	.56000,	.16253,	.54899,
.050000,	.39000,	.87000,	.41431,	.86878,
.087500,	.66000,	1.17000,	.69195,	1.16922,
.100000,	.76000,	1.23000,	.76583,	1.25158,
.125000,	.89000,	1.39000,	.89576,	1.39867,
.150000,	1.01000,	1.49000,	1.00408,	1.52567,
.175000,	1.12000,	1.61000,	1.09045,	1.63721,
.200000,	1.17000,	1.69000,	1.15459,	1.73882,
.250000,	1.25000,	1.93000,	1.22118,	1.93316,
.300000,	1.23000,	2.12000,	1.22287,	2.14008,
.350000,	1.19000,	2.39000,	1.18306,	2.37316,
.400000,	1.15000,	2.64000,	1.12048,	2.63369,
.450000,	1.07000,	2.93000,	1.04718,	2.91750,
.500000,	.98000,	3.25000,	.96983,	3.21913,
.550000,	.92000,	3.54000,	.89166,	3.53366,
.600000,	.88000,	3.89000,	.81392,	3.85723,
.650000,	.79000,	4.20000,	.73684,	4.18702,
.700000,	.68000,	4.52000,	.66016,	4.52100,
.750000,	.57000,	4.88000,	.58345,	4.85778,
.800000,	.47000,	5.22000,	.50621,	5.19641,
.850000,	.39000,	5.53000,	.42799,	5.53621,
.900000,	.29000,	5.87000,	.34837,	5.87675,
.950000,	.20000,	6.25000,	.26700,	6.21770,
1.000000,	.10000,	6.58000,	.18357,	6.55886,

C_{l_α}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	14.70000,	.00000,	13.94730,	.00000,
.025000,	12.20000,	-4.05000,	12.17360,	-3.42799,
.050000,	9.50000,	-4.15000,	9.67237,	-4.23117,
.087500,	7.63000,	-3.70000,	7.57972,	-3.78514,
.100000,	7.22000,	-3.47000,	7.16425,	-3.58634,
.125000,	6.61000,	-3.18000,	6.53843,	-3.22036,
.150000,	6.09000,	-2.94000,	6.07511,	-2.89669,
.175000,	5.72000,	-2.59000,	5.70821,	-2.60052,
.200000,	5.41000,	-2.29000,	5.40820,	-2.31982,
.250000,	4.92000,	-1.79000,	4.95806,	-1.78414,
.300000,	4.63000,	-1.30000,	4.66424,	-1.27997,
.350000,	4.43000,	-.83000,	4.48829,	-.81771,
.400000,	4.35000,	-.41000,	4.39670,	-.40553,
.450000,	4.35000,	-.03000,	4.36154,	-.04479,
.500000,	4.40000,	.28000,	4.36154,	.26823,
.550000,	4.43000,	.55000,	4.38166,	.53973,
.600000,	4.50000,	.74000,	4.41178,	.77649,
.650000,	4.50000,	1.00000,	4.44540,	.98490,
.700000,	4.48000,	1.18000,	4.47847,	1.17048,
.750000,	4.48000,	1.31000,	4.50862,	1.33784,
.800000,	4.48000,	1.47000,	4.53451,	1.49071,
.850000,	4.49000,	1.63000,	4.55549,	1.63206,
.900000,	4.56000,	1.80000,	4.57130,	1.76425,
.950000,	4.60000,	1.91000,	4.58194,	1.88914,
1.000000,	4.62000,	2.02000,	4.58755,	2.00820,

continued

C_{m_h}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	.00000,	.00000,	.00000,	.00000,
.025000,	.00100,	-.03500,	.01514,	- 01722,
.050000,	.01500,	-.06500,	.03491,	-.05826,
.087500,	.02500,	-.12500,	.04444,	-.13429,
.100000,	.03000,	-.14500,	.04338,	-.15945,
.125000	.02000,	-.19800,	.03688,	-.20736,
.150000,	.01500,	-.23000,	.02622,	-.25074,
.175000,	.01000,	-.28000,	.01345,	-.28853,
.200000,	-.00500,	-.32500,	.00069,	-.32037,
.250000,	-.03500,	-.39700,	-.01725,	-.36818,
.300000,	-.06000,	-.40300,	-.01698,	-.40259,
.350000,	-.04500,	-.42500,	.00424,	-.43438,
.400000,	-.00500,	-.44800,	.04351,	-.47188,
.450000,	.06500,	-.49000,	.09567,	-.51964,
.500000,	.14000,	-.55000,	.15561,	-.57908,
.550000,	.21000,	-.64000,	.21928,	-.64973,
.600000,	.27000,	-.76000,	.28384,	-.73020,
.650000,	.34000,	-.86000,	.34752,	-.81879,
.700000,	.39000,	-.97000,	.40931,	-.91387,
.750000,	.46000,	-1.06000,	.46873,	-1.01398,
.800000,	.53000,	-1.13000,	.52566,	-1.11794,
.850000,	.57000,	-1.19000,	.58017,	-1.22475,
.900000,	.61000,	-1.28000,	.63245,	-1.33366,
.950000,	.65000,	-1.40000,	.68275,	-1.44407,
1.000000,	.68000,	-1.53000,	.73133,	-1.55552,

C_{m_α}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	-.44500,	.00000,	-.57389,	.00000,
.025000,	-.45000,	-.02700,	-.50236,	-.00277,
.050000,	-.45500,	-.12200,	-.43591,	-.12894,
.087500,	-.49500,	-.31100,	-.48192,	-.34358,
.100000,	-.52000,	-.34800,	-.51936,	-.39784,
.125000,	-.58100,	-.42200,	-.60391,	-.47587,
.150000,	-.67200,	-.50800,	-.68543,	-.51920,
.175000,	-.73200,	-.54800,	-.75284,	-.53819,
.200000,	-.81000,	-.58200,	-.80187,	-.54342,
.250000,	-.90500,	-.61000,	-.84842,	-.54545,
.300000,	-.89800,	-.55000,	-.84946,	-.56636,
.350000,	-.86800,	-.57500,	-.83350,	-.61570,
.400000,	-.82400,	-.64800,	-.81867,	-.68765,
.450000,	-.79000,	-.75000,	-.81277,	-.77282,
.500000,	-.77000,	-.90300,	-.81723,	-.86341,
.550000,	-.81800,	-.99200,	-.83063,	-.95435,
.600000,	-.85200,	-1.08000,	-.85067,	-1.04290,
.650000,	-.87800,	-1.14300,	-.87516,	-1.12786,
.700000,	-.89200,	-1.23000,	-.90235,	-1.20894,
.750000,	-.93000,	-1.30700,	-.93092,	-1.28636,
.800000,	-.97800,	-1.38100,	-.95998,	-1.36052,
.850000,	-1.00800,	-1.43900,	-.98895,	-1.43193,
.900000,	-1.03100,	-1.49500,	-1.01747,	-1.50106,
.950000,	-1.04300,	-1.53400,	-1.04534,	-1.56834,
1.000000,	-1.05800,	-1.56400,	-1.07247,	-1.63414,

LAPLACE METHOD

TRANSONIC M=0.85
FORCED CONSTRAINTS AT N(r,c)=0.0 AT k=0.0
RANGE 0.00 < k < 1.00

u	r(alp)	ah	wben/wtor
50.0	.5	-.5	.2

ND VELOCITY .500

-.01753	-1.17101	
-.01753	1.17101	
-.00509	.20181	
-.00509	-.20181	
-.19517	.03653	
-.17830	.03354	
-.19517	-.03653	
-.17830	-.03354	
-.13698	.00000	
-.12781	.00000	
-.02340	.00000	
-.02273	.00000	
perfomance index is	.75683	.00000

ND VELOCITY 3.100

-.19085	1.23731	
-.19085	-1.23731	
-1.36708	.69722	
-1.36708	-.69722	
-1.16682	.25217	
-1.16682	-.25217	
-.74436	.00000	
-.00338	.29255	
-.00338	-.29255	
-.35884	.00000	
-.20286	.00000	
-.07687	.00000	
perfomance index is	8.89727	.00000

ND VELOCITY 3.200

-.19992	1.24432	
-.19992	-1.24432	
-1.41678	.72501	
-1.41678	-.72501	
-1.20460	.26032	
-1.20460	-.26032	
-.76779	.00000	
.00058	.29752	INSTABILITY
.00058	-.29752	ND VELOCITY = 3.19
-.36035	.00000	
-.21351	.00000	
-.07676	.00000	
perfomance index is	15.95851	.00000

SEQUENTIAL SIMPLEX : PROBLEM MINIMIZATION
TRANSONIC REGIME M=0.85
GLOBAL OPTIMIZATION
FORCED CONSTRAINTS
RANGE 0.00 < k < 1.00

C_{l_h}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	.00000,	.00000,	.00000,	.00000,
.025000,	.17000,	.56000,	.20041,	.56379,
.050000,	.39000,	.87000,	.43868,	.85147,
.087500,	.66000,	1.17000,	.70126,	1.18815,
.100000,	.76000,	1.23000,	.78739,	1.27735,
.125000,	.89000,	1.39000,	.94081,	1.41745,
.150000,	1.01000,	1.49000,	1.05738,	1.52418,
.175000,	1.12000,	1.61000,	1.13708,	1.61709,
.200000,	1.17000,	1.69000,	1.18649,	1.70825,
.250000,	1.25000,	1.93000,	1.22244,	1.90616,
.300000,	1.23000,	2.12000,	1.20838,	2.13362,
.350000,	1.19000,	2.39000,	1.16854,	2.38805,
.400000,	1.15000,	2.64000,	1.11535,	2.66342,
.450000,	1.07000,	2.93000,	1.05501,	2.95416,
.500000,	.98000,	3.25000,	.99046,	3.25592,
.550000,	.92000,	3.54000,	.92301,	3.56554,
.600000,	.88000,	3.89000,	.85313,	3.88073,
.650000,	.79000,	4.20000,	.7808[illegible],	4.19990,
.700000,	.68000,	4.52000,	.70617,	4.52191,
.750000,	.57000,	4.88000,	.62879,	4.84597,
.800000,	.47000,	5.22000,	.54853,	5.17148,
.850000,	.39000,	5.53000,	.46518,	5.49806,
.900000,	.29000,	5.87000,	.37857,	5.82540,
.950000,	.20000,	6.25000,	.28853,	6.15329,
1.000000,	.10000,	6.58000,	.19492,	6.48156,

C_{l_α}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	14.70000,	.00000,	14.70000,	.00000,
.025000,	12.20000,	-4.05000,	12.09142,	-4.07232,
.050000,	9.50000,	-4.15000,	9.23969,	-4.20057,
.087500,	7.63000,	-3.70000,	7.67001,	-3.60840,
.100000,	7.22000,	-3.47000,	7.30929,	-3.51304,
.125000,	6.61000,	-3.18000,	6.64489,	-3.30380,
.150000,	6.09000,	-2.94000,	6.08270,	-3.02828,
.175000,	5.72000,	-2.59000,	5.63848,	-2.70889,
.200000,	5.41000,	-2.29000,	5.30123,	-2.37741,
.250000,	4.92000,	-1.79000,	4.86628,	-1.75049,
.300000,	4.63000,	-1.30000,	4.63458,	-1.21168,
.350000,	4.43000,	-.83000,	4.51518,	-.76053,
.400000,	4.35000,	-.41000,	4.45752,	-.38090,
.450000,	4.35000,	-.03000,	4.43406,	-.05620,
.500000,	4.40000,	.28000,	4.42969,	.22711,
.550000,	4.43000,	.55000,	4.43604,	.47927,
.600000,	4.50000,	.74000,	4.44844,	.70785,
.650000,	4.50000,	1.00000,	4.46424,	.91840,
.700000,	4.48000,	1.18000,	4.48194,	1.11500,
.750000,	4.48000,	1.31000,	4.50072,	1.30069,
.800000,	4.48000,	1.47000,	4.52013,	1.47773,
.850000,	4.49000,	1.63000,	4.53991,	1.64784,
.900000,	4.56000,	1.80000,	4.55997,	1.81234,
.950000,	4.60000,	1.91000,	4.58025,	1.97223,
1.000000,	4.62000,	2.02000,	4.60076,	2.12832,

continued

C_{m_h}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	.00000,	.00000,	.00000,	.00000,
.025000,	.00100,	-.03500,	-.08536,	-.07477,
.050000,	.01500,	-.06500,	-.07246,	-.03012,
.087500,	.02500,	-.12500,	.00470,	-.13296,
.100000,	.03000,	-.14500,	-.00026,	-.17806,
.125000,	.02000,	-.19800,	-.03268,	-.24522,
.150000,	.01500,	-.23000,	-.06879,	-.28304,
.175000,	.01000,	-.28000,	-.09475,	-.30307,
.200000,	-.00500,	-.32500,	-.10802,	-.31524,
.250000,	-.03500,	-.39700,	-.10344,	-.33744,
.300000,	-.06000,	-.40300,	-.07271,	-.37041,
.350000,	-.04500,	-.42500,	-.02907,	-.41710,
.400000,	-.00500,	-.44800,	.02019,	-.47542,
.450000,	.06500,	-.49000,	.07148,	-.54248,
.500000,	.14000,	-.55000,	.12324,	-.61574,
.550000,	.21000,	-.64000,	.17495,	-.69328,
.600000,	.27000,	-.76000,	.22659,	-.77370,
.650000,	.34000,	-.86000,	.27836,	-.85600,
.700000,	.39000,	-.97000,	.33054,	-.93951,
.750000,	.46000,	-1.06000,	.38344,	-1.02376,
.800000,	.53000,	-1.13000,	.43733,	-1.10842,
.850000,	.57000,	-1.19000,	.49246,	-1.19328,
.900000,	.61000,	-1.28000,	.54905,	-1.27819,
.950000,	.65000,	-1.40000,	.60729,	-1.36306,
1.000000,	.68000,	-1.53000,	.66733,	-1.44782,

C_{m_α}

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	-.44500,	.00000,	-.44500,	.00000,
.025000,	-.45000,	-.02700,	-.45251,	-.05573,
.050000,	-.45500,	-.12200,	-.40544,	-.12522,
.087500,	-.49500,	-.31100,	-.42450,	-.33564,
.100000,	-.52000,	-.34800,	-.46298,	-.39195,
.125000,	-.58100,	-.42200,	-.55008,	-.46639,
.150000,	-.67200,	-.50800,	-.62755,	-.50379,
.175000,	-.73200,	-.54800,	-.68670,	-.52187,
.200000,	-.81000,	-.58200,	-.72856,	-.53235,
.250000,	-.90500,	-.61000,	-.77490,	-.55295,
.300000,	-.89800,	-.55000,	-.79240,	-.58606,
.350000,	-.86800,	-.57500,	-.79722,	-.63383,
.400000,	-.82400,	-.64800,	-.79794,	-.69385,
.450000,	-.79000,	-.75000,	-.79890,	-.76310,
.500000,	-.77000,	-.90300,	-.80217,	-.83905,
.550000,	-.81800,	-.99200,	-.80866,	-.91976,
.600000,	-.85200,	-1.08000,	-.81870,	-1.00384,
.650000,	-.87800,	-1.14300,	-.83234,	-1.09027,
.700000,	-.89200,	-1.23000,	-.84948,	-1.17836,
.750000,	-.93000,	-1.30700,	-.86999,	-1.26760,
.800000,	-.97800,	-1.38100,	-.89371,	-1.35764,
.850000,	-1.00800,	-1.43900,	-.92048,	-1.44823,
.900000,	-1.03100,	-1.49500,	-.95018,	-1.53920,
.950000,	-1.04300,	-1.53400,	-.98269,	-1.63041,
1.000000,	-1.05800,	-1.56400,	-1.01789,	-1.72178,

LAPLACE METHOD

TRANSONIC M=0.85
FORCED CONSTRAINTS
RANGE 0.00 < k < 1.00

u	r(alp)	ah	wben/wtor
50.0	.5	-.5	.2

ND VELOCITY .500

-.02388	1.17850	
-.02388	-1.17850	
-.00516	-.20181	
-.00516	.20181	
-.17990	.00227	
-.17990	-.00227	
-.04345	-.00962	
-.04345	.00962	
-.03331	.01123	
-.03331	-.01123	
-.02539	.00376	
-.02539	-.00376	
performance index is	.78187	.00000

ND VELOCITY 3.300

-.21773	-1.24873	
-.21773	1.24873	
-1.28855	-.01047	
-1.28855	.01047	
-.34557	-.25499	
-.00193	-.29898	
-.34557	.25499	
-.00193	.29898	
-.15908	-.13442	
-.04041	-.07268	
-.15908	.13442	
-.04041	.07268	
perfomance index is	8.34966	.00000

ND VELOCITY 3.400

-.22938	-1.25594	
-.22938	1.25594	
-1.34034	.00000	
-1.31940	.00000	
-.35652	-.26509	
.00131	-.30338	INSTABILITY
-.35652	.26509	ND VELOCITY = 3.36
.00131	.30338	
-.16341	-.13823	
-.16341	.13823	
-.03761	-.07519	
-.03761	.07519	
perfomance index is	8.98850	.00000

Appendix G : Incompressible Regime - Eigenvalue Solutions

Eigenvalue solutions for the incompressible regime (Case #1) incorporating the $V-g$, $p-k$ and Laplace methods.

UG METHOD CASE #1

u	r(alp)	ah	wben/wtor
50.0	.5	-.5	.2

k	U/b*wtor	w/wtor	g	damping ratio	
10.000,	.01970,	.19698,	-.00209,	.00104,	
10.000,	.11493,	1.14932,	-.00569,	.00284,	
6.000,	.03283,	.19699,	-.00348,	.00174,	
6.000,	.19150,	1.14900,	-.00948,	.00474,	
4.000,	.04925,	.19701,	-.00524,	.00262,	
4.000,	.28709,	1.14837,	-.01423,	.00712,	
2.000,	.09856,	.19711,	-.01067,	.00533,	
2.000,	.57247,	1.14494,	-.02861,	.01431,	
1.000,	.19746,	.19746,	-.02238,	.01119,	
1.000,	1.13102,	1.13102,	-.05796,	.02898,	
.880,	.22452,	.19758,	-.02580,	.01290,	
.880,	1.27906,	1.12558,	-.06604,	.03302,	
.800,	.24711,	.19769,	-.02871,	.01436,	
.800,	1.40063,	1.12050,	-.07275,	.03638,	
.600,	.33024,	.19814,	-.03995,	.01998,	
.600,	1.82975,	1.09785,	-.09686,	.04843,	
.500,	.39712,	.19856,	-.04954,	.02477,	
.500,	2.15096,	1.07548,	-.11494,	.05747,	
.400,	.49815,	.19926,	-.06485,	.03243,	
.400,	2.58976,	1.03590,	-.13857,	.06929,	
.300,	.66874,	.20062,	-.09274,	.04637,	
.300,	3.19652,	.95896,	-.16476,	.08238,	
.200,	1.02048,	.20410,	-.15760,	.07880,	
.200,	3.97729,	.79546,	-.16106,	.08053,	
.150,	1.39164,	.20875,	-.23812,	.11906,	
.150,	4.35545,	.65332,	-.09675,	.04837,	
.140,	1.50237,	.21033,	-.26508,	.13254,	
.140,	4.41935,	.61871,	-.07038,	.03519,	
.130,	1.63333,	.21233,	-.29866,	.14933,	
.130,	4.47879,	.58224,	-.03713,	.01856,	
.125,	1.70828,	.21354,	-.31888,	.15944,	
.125,	4.50682,	.56335,	-.01709,	.00854,	
.120,	1.79092,	.21491,	-.34199,	.17100,	
.120,	4.53418,	.54410,	.00554,	-.00277,	INSTABILITY
.110,	1.98367,	.21820,	-.40100,	.20050,	U/b*wtor = 4.53
.110,	4.59055,	.50496,	.06404,	-.03202,	
.100,	2.22456,	.22246,	-.48395,	.24197,	
.100,	4.66272,	.46627,	.14335,	-.07167,	
.050,	4.97074,	.24854,	-1.75199,	.87599,	
.050,	7.19299,	.35965,	1.11015,	-.55508,	

PK METHOD CASE #1

u	r(alp)	ah	wben/wtor
50.0	.5	-.5	.2

k	U/b*wtor	damping	w	damping ratio	
.3966,	.5000,	-.00640,	.19917,	.03212	
2.2909,	.5000,	-.01436,	1.14593,	.01253	
.2034,	1.0000,	-.01487,	.20335,	.07295	
1.1348,	1.0000,	-.02958,	1.13484,	.02606	
.1394,	1.5000,	-.02492,	.20916,	.11830	
.7434,	1.5000,	-.04621,	1.11514,	.04141	
.1087,	2.0000,	-.03677,	.21735,	.16679	
.5425,	2.0000,	-.06450,	1.08521,	.05933	
.0916,	2.5000,	-.05169,	.22893,	.22026	
.4169,	2.5000,	-.08437,	1.04258,	.08066	
.3275,	3.0000,	-.10547,	.98336,	.10664	
.0820,	3.0000,	-.07180,	.24614,	.28002	
.2570,	3.5000,	-.12613,	.89945,	.13887	
.0781,	3.5000,	-.10247,	.27359,	.35074	
.1923,	4.0000,	-.13992,	.76981,	.17882	
.0809,	4.0000,	-.16167,	.32367,	.44684	
.1784,	4.1000,	-.13846,	.73202,	.18586	
.0821,	4.1000,	-.18312,	.33747,	.47693	
.1633,	4.2000,	-.13313,	.68655,	.19036	
.0838,	4.2000,	-.21078,	.35084,	.51500	
.1456,	4.3000,	-.11102,	.62712,	.17433	
.0833,	4.3000,	-.24824,	.35793,	.56990	
.1306,	4.4000,	-.05886,	.57403,	.10201	
.0805,	4.4000,	-.29190,	.35396,	.63623	
.1228,	4.5000,	-.01093,	.55238,	.01977	
.0749,	4.5000,	-.33777,	.33681,	.70811	
.1177,	4.6000,	.02507,	.54086,	-.04629	INSTABILITY
.0678,	4.6000,	-.38361,	.31138,	.77642	U/b*wtor = 4.53
.1133,	4.7000,	.05465,	.53225,	-.10213	
.0599,	4.7000,	-.43058,	.28056,	.83783	
.1094,	4.8000,	.08015,	.52451,	-.15106	
.0499,	4.8000,	-.48151,	.23879,	.89589	
.1056,	4.9000,	.10301,	.51721,	-.19532	
.0390,	4.9000,	-.53891,	.19045,	.94285	
.1020,	5.0000,	.12384,	.50991,	-.23600	
.0284,	5.0000,	-.60286,	.14083,	.97378	
.0986,	5.1000,	.14317,	.50263,	-.27394	
.0208,	5.1000,	-.66517,	.10505,	.98776	

LAPLACE METHOD CASE #1

u	r(alp)	ah	wben/wtor
50.0	.5	-.5	.2

ND VELOCITY .50

-.01437	1.14595
-.01437	-1.14595
-.00637	-.19921
-.00637	.19921
-.38147	.00000
-.38256	.00000
-.12656	.00000
-.12798	.00000
-.04200	.00000
-.04214	.00000
-.00772	.00000
-.00772	.00000

perfomance index is 1.13157 .00000

ND VELOCITY 1.00

-.02966	1.13486
-.02966	-1.13486
-.76212	.00000
-.76511	.00000
-.01482	.20360
-.01482	-.20360
-.24879	.00000
-.25595	.00000
-.08316	.00000
-.08429	.00000
-.01544	.00000
-.01543	.00000

perfomance index is 1.22729 .00000

ND VELOCITY 1.50

-.04649	-1.11520
-.04649	1.11520
-1.14241	.00000
-1.14767	.00000
-.02480	.20984
-.02480	-.20984
-.36693	.00000
-.38393	.00000
-.12267	.00000
-.12643	.00000
-.02316	.00000
-.02313	.00000

perfomance index is 2.29740 .00000

ND VELOCITY 2.00

-.06529	1.08538
-1.52250	.00000
-1.53023	.00000
-.06529	-1.08538
-.51190	.00000
-.48020	.00000

```
     -.03664          .21879
     -.03664         -.21879
     -.15957          .00000
     -.16857          .00000
     -.03088          .00000
     -.03080          .00000
perfomance index is       28.13492         .00000

ND VELOCITY  2.50
    -1.90251          .00000
    -1.91278          .00000
     -.08644        -1.04289
     -.08644         1.04289
     -.63988          .00000
     -.58718          .00000
     -.05141         -.23195
     -.05141          .23195
     -.19235          .00000
     -.21072          .00000
     -.03860          .00000
     -.03843          .00000
perfomance index is        2.70799         .00000

ND VELOCITY  3.00
    -2.28251          .00000
    -2.29534          .00000
     -.11024         -.98348
     -.11024          .98348
     -.76786          .00000
     -.68609          .00000
     -.07108          .25240
     -.07108         -.25240
     -.25286          .00000
     -.21817          .00000
     -.04632          .00000
     -.04600          .00000
perfomance index is        2.86213         .00000

ND VELOCITY  3.50
    -2.66251          .00000
    -2.67790          .00000
     -.13678          .89843
     -.13678         -.89843
     -.89583          .00000
     -.77454          .00000
     -.09940          .28811
     -.09940         -.28811
     -.29500          .00000
     -.23173          .00000
     -.05404          .00000
     -.05348          .00000
perfomance index is        2.21510         .00000

ND VELOCITY  4.00
    -3.04253          .00000
    -3.06045          .00000
     -.16495         -.76095
     -.16495          .76095
    -1.02381          .00000
     -.84867          .00000
```

```
   -.14324          -.36783
   -.14324           .36783
   -.33714           .00000
   -.22548           .00000
   -.06176           .00000
   -.06081           .00000
perfomance index is        2.43926          .00000

ND VELOCITY  4.10
  -3.11854           .00000
  -3.13696           .00000
   -.16997           .71810
   -.16997          -.71810
  -1.04940           .00000
   -.86123           .00000
   -.15509           .39867
   -.15509          -.39867
   -.34557           .00000
   -.22157           .00000
   -.06331           .00000
   -.06225           .00000
perfomance index is        2.98730          .00000

ND VELOCITY  4.20
  -3.19455           .00000
  -3.21348           .00000
  -1.07500           .00000
   -.87286           .00000
   -.17319          -.65974
   -.17319           .65974
   -.16957           .44503
   -.16957          -.44503
   -.35400           .00000
   -.21693           .00000
   -.06485           .00000
   -.06368           .00000
perfomance index is        2.48161          .00000

ND VELOCITY  4.40
  -3.34656           .00000
  -3.36650           .00000
   -.05134           .55387
   -.32922           .52710
  -1.12619           .00000
   -.89292           .00000
   -.05134          -.55387
   -.32922          -.52710
   -.37086           .00000
   -.20615           .00000
   -.06794           .00000
   -.06651           .00000
perfomance index is        6.60422          .00000

ND VELOCITY  4.50
  -3.42257           .00000
  -3.44301           .00000
  -1.15178           .00000
   -.01016          -.54976
   -.39043          -.51942
   -.90108           .00000
```

```
     -.01016           .54976
     -.39043           .51942
     -.37929           .00000
     -.20035           .00000
     -.06948           .00000
     -.06791           .00000
perfomance index is          2.05162          .00000

ND VELOCITY   4.60
    -3.49858           .00000
    -3.51952           .00000
    -1.17738           .00000
      .02177           .54546       INSTABILITY
     -.44317           .51200       U/b*wtor = 4.53
     -.90778           .00000
      .02177          -.54546
     -.44317          -.51200
     -.38771           .00000
     -.06929           .00000
     -.07103           .00000
     -.19451           .00000
perfomance index is          5.28303          .00000

ND VELOCITY   4.80
    -3.65060           .00000
    -3.67254           .00000
      .07249           .53602
      .07249          -.53602
    -1.22857           .00000
     -.53799           .49817
     -.91590           .00000
     -.53799          -.49817
     -.40457           .00000
     -.07200           .00000
     -.07412           .00000
     -.18314           .00000
perfomance index is          2.20679          .00000

ND VELOCITY   5.00
    -3.80262           .00000
    -3.82557           .00000
      .11382          -.52557
      .11382           .52557
    -1.27976           .00000
     -.62745          -.48580
     -.62745           .48580
     -.91517           .00000
     -.42143           .00000
     -.17267           .00000
     -.07461           .00000
     -.07720           .00000
perfomance index is          1.71618          .00000
```

Appendix H : Transonic Regime - Laplace Generated Eigenvalues for $M = 0.85$

This sections gives the Laplace generated eigenvalues for $M = 0.85$.

LAPLACE METHOD TRANSONIC M=0.85

u	r(alp)	ah	wben/wtor
50.0	.5	-.5	.2

ND VELOCITY .500

-.08315	1.27662	
-.08315	-1.27662	
-.00511	-.20185	
-.00511	.20185	
-.18737	-.01797	
-.18737	.01797	
-.12884	-.03121	
-.12884	.03121	
-.11139	-.01459	
-.11139	.01459	
-.02172	.00019	
-.02172	-.00019	
perfomance index is	1.34192	.00000

ND VELOCITY 1.000

-.13682	-1.22225	
-.13682	1.22225	
-.01143	.21140	
-.01143	-.21140	
-.47016	.00000	
-.41480	.00000	
-.24390	-.08059	
-.24390	.08059	
-.19773	-.03599	
-.19773	.03599	
-.04279	.00145	
-.04279	-.00145	
perfomance index is	1.91492	.00000

ND VELOCITY 1.500

-.14304	-1.17204	
-.14304	1.17204	
-.91862	.00000	
-.60947	.00000	
-.01825	.22490	
-.01825	-.22490	
-.35435	.13235	
-.35435	-.13235	
-.27054	.05787	
-.27054	-.05787	
-.06248	.00460	
-.06248	-.00460	
perfomance index is	3.55296	.00000

ND VELOCITY 2.000

-.13090	-1.16269
-1.42867	.00000
-.13090	1.16269
-.81080	.00000
-.02269	.24257

-.02269	-.24257	
-.46358	.18321	
-.46358	-.18321	
-.33331	.07634	
-.33331	-.07634	
-.08006	-.01006	
-.08006	.01006	
perfomance index is	3.54698	.00000

ND VELOCITY 2.500

-1.93877	.00000	
-.13300	-1.18652	
-.13300	1.18652	
-1.01285	.00000	
-.02228	.26434	
-.02228	-.26434	
-.57278	.23294	
-.57278	-.23294	
-.38902	.08819	
-.38902	-.08819	
-.09496	-.01771	
-.09496	.01771	
perfomance index is	4.61556	.00000

ND VELOCITY 3.000

-2.43536	.00000	
-.15539	1.22983	
-.15539	-1.22983	
-1.21510	.00000	
-.68232	-.28189	
-.68232	.28189	
-.01448	-.28849	
-.01448	.28849	
-.44104	.09158	
-.44104	-.09158	
-.10696	.02704	
-.10696	-.02704	
perfomance index is	3.86013	.00000

ND VELOCITY 3.100

-2.53320	.00000	
-.16210	-1.24029	
-.16210	1.24029	
-1.25556	.00000	
-.70427	-.29162	
-.70427	.29162	
-.01186	-.29332	
-.01186	.29332	
-.45129	-.09119	
-.45129	.09119	
-.10903	.02906	
-.10903	-.02906	
perfomance index is	5.05791	.00000

ND VELOCITY 3.200

-2.63062	.00000
-.16948	1.25130
-.16948	-1.25130
-1.29602	.00000
-.72623	-.30133

-.72623	.30133	
-.00889	-.29807	
-.00889	.29807	
-.46152	-.09044	
-.46152	.09044	
-.11100	.03110	
-.11100	-.03110	
perfomance index is	5.61231	.00000

ND VELOCITY 3.300

-2.72764	.00000	
-.17749	-1.26286	
-.17749	1.26286	
-1.33649	.00000	
-.74821	-.31102	
-.74821	.31102	
-.00554	-.30271	
-.00554	.30271	
-.47177	-.08933	
-.47177	.08933	
-.11287	.03319	
-.11287	-.03319	
perfomance index is	24.33285	.00000

ND VELOCITY 3.400

-2.82428	.00000	
-.18609	-1.27495	
-.18609	1.27495	
-1.37696	.00000	
-.77020	.32070	
-.77020	-.32070	
-.00184	.30721	
-.00184	-.30721	
-.48205	.08786	
-.48205	-.08786	
-.11466	.03530	
-.11466	-.03530	
perfomance index is	16.47175	.00000

ND VELOCITY 3.500

-2.92058	.00000	
-.19525	1.28757	
-1.41743	.00000	
-.19525	-1.28757	INSTABILITY
-.79221	-.33037	ND VEL = 3.45
-.79221	.33037	
.00221	.31152	
.00221	-.31152	
-.49236	.08603	
-.49236	-.08603	
-.11637	.03744	
-.11637	-.03744	
perfomance index is	5.16679	.00000

ND VELOCITY 4.000

-3.39765	.00000
-1.61980	.00000
-.24794	-1.35846
-.24794	1.35846
-.90240	.37857

-.90240	-.37857	
.02714	-.32936	
.02714	.32936	
-.54475	-.07102	
-.54475	.07102	
-.12386	.04847	
-.12386	-.04847	
perfomance index is	2.57841	.00000

Appendix I : Transonic Regime - Comparison of Rational Function Optimization

Comparison of rational function optimization for $M = 0.80$, $M = 0.85$ and $M = 0.875$.

TRANSONIC AERODYNAMICS

		M=0.85	M=0.80	M=0.875
N(1,1)	C_{l_h}	.000000	.000000	.000000
		7.351972	1.908267	6.759715
		-.265177	3.804198	.499484
		-8561.672085	-95004.636200	3.652575
		.593074	.668869	.076634
		8707.625432	15317.270050	.928264
		-146.231038	79694.444592	-3.828033
N(1,2)	C_{l_α}	14.235672	10.400895	17.891303
		.957753	6.483311	-.319441
		1.258183	-.637080	2.425840
		404976.770760	199406.181992	-21.350260
		-8.145772	-4.472213	11.313079
		-408964.029690	-31963.187732	-23.973478
		3987.367830	-167451.070982	22.699002
N(2,1)	C_{m_h}	.000000	.000000	.000000
		-1.905869	-1.614667	-1.979172
		-.293633	-0.378487	.006105
		158360.047503	10060.571210	-3.656111
		-.059842	.024373	-.004016
		-159903.225805	-1624.362035	-.219576
		1543.931881	-8436.118582	4.435226
N(2,2)	C_{m_α}	-.516839	-.178982	-4.220761
		-4.394286	2.247185	-1.072839
		2.744002	-2.311694	.007098
		217699.240354	21872.971701	7.958546
		.165112	-.038852	-.309693
		-219881.401614	-3648.306001	3.606871
		2184.097622	-18230.087698	-8.442002

OPTIMAL

	M=0.85	M=0.80	M=0.875
LAG TERMS	.27892661	.54004054	.30284145
	.04365735	.06187302	.00461111
	.27899139	.54678439	.02192557
	.28586712	.53878120	.35119036
ERROR	.00139602	.00106544	.00236629

Appendix J : 3DOF Incompressible Unsteady Aerodynamics

Expressions for Lift force, Pitching Moment and Hinge Moment

$$P_w = 1 - 2\imath\frac{V}{bu}(F + \imath G)$$

$$P_\phi = \frac{1}{2} - \imath\frac{V}{bu}[1 + 2(F + \imath G)] - 2\left(\frac{V}{bu}\right)^2 (F + \imath G)$$

$$P_\beta = -\frac{T_1}{\pi} + \imath\frac{V}{bu}\frac{T_4}{\pi} - \imath\frac{V}{bu}\frac{T_{11}}{\pi}(F + \imath G) - 2\left(\frac{V}{bu}\right)^2 \frac{T_{10}}{\pi}(F + \imath G)$$

$$M_w = \frac{1}{2}$$

$$M_\phi = \frac{3}{8} - \imath\frac{V}{bu}$$

$$M_\beta = -\frac{T_7}{\pi} - \left(c_\beta + \frac{1}{2}\right)\frac{T_1}{\pi} + \imath\frac{V}{bu}\frac{T_4 - \frac{2}{3}(\sqrt{1-c_\beta^2})^3}{\pi} - \left(\frac{V}{bw}\right)^2 \frac{T_4 + T_{10}}{\pi}$$

$$T_w = -\frac{T_1}{\pi} - \imath\frac{V}{bu}\frac{T_{12}}{\pi}(F + \imath G)$$

$$T_\phi = -\frac{1}{\pi}\left[T_7 + \left(c_\beta + \frac{1}{2}\right)T_1\right] + \imath\frac{V}{bu}\frac{\frac{2}{3}(\sqrt{1-c_\beta^2})^3 + 2T_1 + T_4}{2\pi} - \imath\frac{V}{bw}\frac{T_{12}}{\pi}(F + \imath G) - \left(\frac{V}{bu}\right)^2 \frac{T_{12}}{\pi}(F + \imath G)$$

$$T_\beta = -\frac{T_3}{\pi^2} + \imath\frac{V}{bu}\frac{T_4 T_{11}}{2\pi^2} - \imath\frac{V}{bu}\frac{T_{11}T_{12}}{2\pi^2}(F + \imath G) - \left(\frac{V}{bw}\right)^2 \frac{T_5 - T_4 T_{10}}{\pi^2} - \left(\frac{V}{bu}\right)^2 \frac{T_{10}T_{12}}{\pi^2}(F + \imath G)$$

$$T_1 = -\frac{1}{3}\sqrt{1 - c_\beta^2}(2 + c_\beta^2) + c_\beta cos^{-1}c_\beta$$

$$T_2 = c_\beta(1 - c_\beta^2) - \sqrt{1 - c_\beta^2}(1 + c_\beta^2)cos^{-1}c_\beta + c_\beta(cos^{-1}c_\beta)^2$$

$$T_3 = -\left(\frac{1}{8} + c_\beta^2\right)(cos^{-1}c_\beta)^2 + \frac{1}{4}c_\beta\sqrt{1 - c_\beta^2}cos^{-1}c_\beta(7 + 2c_\beta^2) - \frac{1}{8}(1 - c_\beta^2)(5c_\beta^2 + 4)$$

$$T_4 = -cos^{-1}c_\beta + c_\beta\sqrt{1 - c_\beta^2}$$

$$T_5 = -(1 - c_\beta^2) - (cos^{-1}c_\beta)^2 + 2c_\beta\sqrt{1 - c_\beta^2}cos^{-1}c_\beta$$

$$T_6 = T_2$$

$$T_7 = -\left(\frac{1}{8} + c_\beta^2\right) cos^{-1}c_\beta + \frac{1}{8}c_\beta\sqrt{1 - c_\beta^2}(7 + 2c_\beta^2)$$

$$T_8 = -\frac{1}{3}\sqrt{1 - c_\beta^2}(2c_\beta^2 + 1) + c_\beta cos^{-1}c_\beta$$

$$T_9 = \frac{1}{2}\left[\frac{1}{3}(\sqrt{1 - c_\beta^2})^3 + aT_4\right]$$

$$T_{10} = \sqrt{1 - c_\beta^2} + cos^{-1}c_\beta$$

$$T_{11} = cos^{-1}c_\beta(1 - 2c_\beta) + \sqrt{1 - c_\beta^2}(2 - c_\beta)$$

$$T_{12} = \sqrt{1 - c_\beta^2}(2 + c_\beta) - cos^{-1}c_\beta(2c_\beta + 1)$$

$$T_{13} = \frac{1}{2}[-T_7 - (c_\beta - a)T_1]$$

$$T_{14} = \frac{1}{16} + \frac{1}{2}ac_\beta$$

Appendix K : Control Transformation Matrix

Various Airfoil Configurations (transformation)

For an airfoil of specific geometric configuration, Nissim and Abel determined the optimal parameters for a PD transfer function control. For an airfoil flap having a length equal to 20% of the chord ($c_\beta = 0.6$, see figure 10), and having displacement measured at 30% chord (measured from leading edge, gives $a = -0.4$), they found the corresponding values

$$[\ t\ ] = [\ t_{11}\ ,\ t_{12}\ ] = [\ 0.0\ ,\ -1.86\ ]\ ,$$

$$[\ t^*\ ] = [\ t_{11}^*\ ,\ t_{12}^*\ ] = [\ 4.0\ ,\ 3.20\ ]\ .$$

In this study an airfoil flap having a length equal to 20% of the chord ($c_\beta = 0.6$, see figure 10) and having displacement measured at 25% chord (corresponding value of $a = -0.5$) was considered. A transformation was required to account for the difference in position at which the measurements were made (Nissim 1971; Nissim 1977)

The transformation matrix can be written as

$$[\ \ ]_s = [\ \ ]_{opt} \begin{bmatrix} 1 & -x_s - 0.4 \\ 0 & 1 \end{bmatrix}$$

where $[\ \ ]_{opt}$ is the original matrix values Nissim and Abel obtained for the specific geometry of an airfoil flap having a length equal to 20% of the chord ($c_\beta = 0.6$, and having displacement measured at 30% chord (measured from leading edge, gives $a = -0.4$), namely

$$[\ t\ ] = [\ t_{11}\ ,\ t_{12}\ ] = [\ 0.0\ ,\ -1.86\ ]\ ,$$

$$[\, t^* \,] = [\, t^*_{11} \quad t^*_{12} \,] = [\, 4.0 \;.\; 3.20 \,]$$

The value of x_s is analogous to the quantity a, that is used in this thesis. Note when a value of $a = -0.4$ is used in the transformation matrix, the original matrix results are correctly obtained.

In this study an airfoil flap having a length equal to 20% of the chord ($c_3 = 0.6$, see figure 10) and having displacement measured at 25% chord (corresponding value of $a = -0.5$) was considered. A transformation was completed to account for the difference in position of measurement by:

$$[\, t \,]_s = [\, t \,]_{opt} \begin{bmatrix} 1 & -x_s - 0.4 \\ 0 & 1 \end{bmatrix}$$

$$[\, t \,]_s = [\, 0 \quad -1.86 \,]_{opt} \begin{bmatrix} 1 & -(-0.5) - 0.4 \\ 0 & 1 \end{bmatrix} = [\, 0 \quad -1.86 \,]$$

and

$$[\, t^* \,]_s = [\, t^* \,]_{opt} \begin{bmatrix} 1 & -x_s - 0.4 \\ 0 & 1 \end{bmatrix}$$

$$[\, t^* \,]_s = [\, 4.0 \quad 3.2 \,]_{opt} \begin{bmatrix} 1 & -(-0.5) - 0.4 \\ 0 & 1 \end{bmatrix} = [\, 4.0 \quad 3.6 \,]$$

Appendix L : Computer Program Listings

PROGRAM	INPUT FILES	OUTPUT FILES	COMMENTS
1 UG.FOR	UGTHEO.DAT CONST.DAT	UGDATA.OUT	eigenvalue analysis flutter calculation COMP.BAT (FORTRAN) (incompressible)
2 PK.FOR	PKTHEO.DAT CONST1.DAT	PKDATA.OUT	eigenvalue analysis flutter calculation COMP.BAT (FORTRAN) (incompressible)
3 SIMPN.FOR	N11.DAT N12.DAT	LAG.OUT LAPLACE.DAT DATA.OUT	rational function approximation (incompressible)
4 SIMPTRN.FOR	N11.DAT N12.DAT N21.DAT N22.DAT	LAG.OUT LAPLACE.DAT DATA.OUT	rational function approximation (transonic)
5 LAPINCOM.FOR	LAPLACE.DAT CONST6.DAT	EIGEN.OUT	eigenvalue analysis flutter calculation COMP.BAT (FORTRAN) (incompressible)
6 LAPTRAN.FOR	LAPLACE.DAT CONST6.DAT	EIGEN.OUT	eigenvalue analysis flutter calculation COMP.BAT (FORTRAN) (transonic)
7 LAPPDOP.FOR	LAPLACE.DAT CONST6.DAT	OPTIMAL.OUT DATA.OUT	eigenvalue analysis flutter calculation COMP.BAT (FORTRAN) (incompressible)

```
$storage:2
$floatcalls
$debug
C-----------------------------------------------------------------------
C    U-g EIGENVALUE SOLUTION
C-----------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-Z)
      INTEGER ib,n,ia,ijob,iz,ier,infer,i,j
      COMPLEX*16 eval(2),A(2,2),B(2,2),EIGA(2),EIGB(2),Z(2,2),WK(3,6)
      REAL k(40),kc(40),F(40),G(40),u
      REAL abar(2,40),bbar(2,40),ndvel(2,40),ndw(2,40)
      REAL gdamp(2,40),dratio(2,40)
      INTEGER pt,pts
C
C Read in constants.
      OPEN (UNIT=1,FILE='CONST.DAT',STATUS='OLD')
           READ (1,*) ia,ib,iz,n,ijob
      CLOSE (UNIT=1)
C
C open required files
      OPEN (UNIT=2,FILE='CONST.OUT',STATUS='NEW')
      OPEN (UNIT=4,FILE='UGEIGEN.OUT',STATUS='NEW')
C
C required constants
                u=50.0
                wrat=0.2
                r=0.5
                ah=-0.5
C
                x=0.25
                semic=1.0
                wtor=1.0
                pts=21
C
C Theodorsen function C(k)= F+iG given by k, where k=kc/2
C thus if kc=0.2,  k=0.1 therefore require C(k)=C(0.1)
      OPEN (UNIT=3,FILE='UGTHEO.DAT',STATUS='OLD')
          do 50 pt=1,pts
             READ (3,*) k(pt),F(pt),G(pt)
             kc(pt)=2.0*k(pt)
 50       continue
      CLOSE (UNIT=3)
C
C formation of COMPLEX matrices [A] and [B] to
C solve eigenvalue problem given by [A]{x} = lamba [B]{x}
          do 100 pt=1,pts
             ar11=0.25*u*kc(pt)**2+0.25*kc(pt)**2+kc(pt)*G(pt)
             ai11=-kc(pt)*F(pt)
             ar12=0.25*x*u*kc(pt)**2-0.25*ah*kc(pt)**2-2*F(pt)+
     1       (0.5-ah)*kc(pt)*G(pt)
             ai12=-0.5*kc(pt)-2*G(pt)-(0.5-ah)*kc(pt)*F(pt)
             ar21=0.25*x*u*kc(pt)**2-0.25*ah*kc(pt)**2-(ah+0.5)
     1       *kc(pt)*G(pt)
```

```
               ai21=(ah+0.5)*kc(pt)*F(pt)
               ar22=0.25*r**2*u*kc(pt)**2+(0.125+ah**2)*0.25*kc(pt)**2
     1         +2*(ah+0.5)*F(pt)-(ah+0.5)*(0.5-ah)*kc(pt)*G(pt)
               ai22=-(0.5-ah)*0.5*kc(pt)+2*(ah+0.5)*G(pt)+(ah+0.5)
     1         *(0.5-ah)*kc(pt)*F(pt)
                  a(1,1)=dcmplx(ar11,ai11)
                  a(1,2)=dcmplx(ar12,ai12)
                  a(2,1)=dcmplx(ar21,ai21)
                  a(2,2)=dcmplx(ar22,ai22)
               br11=wrat**2
               bi11=0.0
               br12=0.0
               bi12=0.0
               br21=0.0
               bi21=0.0
               br22=r**2
               bi22=0.0
                  b(1,1)=dcmplx(br11,bi11)
                  b(1,2)=dcmplx(br12,bi12)
                  b(2,1)=dcmplx(br21,bi21)
                  b(2,2)=dcmplx(br22,bi22)
C
C subrountine for eigenvalue calculation
C abar : real part of eval( )
C bbar : imaginary part of eval( )
      CALL EIGZC(A,IA,B,IB,N,IJOB,EIGA,EIGB,Z,IZ,WK,INFER,IER)
          DO 5 I=1,n
 5             EVAL(I)=EIGA(I)/EIGB(I)
               write(2,19) k(pt),F(pt),G(pt)
 19            format('   k  ',f5.3,'      F + iG ',2f12.7)
               write(2,20) ar11,ai11,ar12,ai12,ar21,ai21
               write(2,20) ar22,ai22,br11,br22
 20            format(10f12.7)
               write(2,21)wk(1,1)
 21            format('  perfomance index is', 2f10.3)
               write(2,22)eval(1)
                  abar(1,pt)=dreal(eval(1))
                  bbar(1,pt)=dimag(eval(1))
               write(2,22)eval(2)
                  abar(2,pt)=dreal(eval(2))
                  bbar(2,pt)=dimag(eval(2))
               write(4,22)k(pt),eval(1)
               write(4,22)k(pt),eval(2)
          do 6 i=1,n
 6             write(2,22) (z(i,j),j=1,n)
 22            format(8f12.5)
 100      continue
      CLOSE (UNIT=3)
C
C calculations and data table
C ndvel     : nd velocity                         vel/(b*wtor)
C ndw       : nd frequency                         w2/wtor
C gdamp     : structural damping coefficient         g
C dratio    : damping ratio                         -g/2
```

```
      OPEN (UNIT=5,FILE='UGDATA.OUT',STATUS='NEW')
      OPEN (UNIT=6,FILE='UGDAMP.OUT',STATUS='NEW')
      OPEN (UNIT=7,FILE='UGFREQ.OUT',STATUS='NEW')
             write(5,32)
             write(5,121)
             write(5,121)
             write(5,33)
             write(5,121)
             write(5,34) u,r,ah,wrat
             write(5,121)
             write(5,120)
             write(5,121)
 32          format('  UG METHOD ')
 33          format('   u           r(alp)     ah     wben/wtor')
 34          format(f7.1,4f10.1)
120          format('   k           U/b*wtor      w/wtor        g   ',
     1'   damping ratio ')
121          format('   ')
          do 140 pt=1,pts
          do 130 i=1,2
             ndvel(i,pt)=(u/abar(i,pt))**0.5
             ndw(i,pt)=k(pt)*(u/abar(i,pt))**0.5
             gdamp(i,pt)=bbar(i,pt)/abar(i,pt)
             dratio(i,pt)=-gdamp(i,pt)/2
C printing data to to be graphed to file 'UGDATA.OUT'
             write(5,125) k(pt),ndvel(i,pt),
     1       ndw(i,pt),gdamp(i,pt),dratio(i,pt)
             write(6,125) ndvel(i,pt),dratio(i,pt)
             write(7,125) ndvel(i,pt),ndw(i,pt)
 125         format(f6.3,',',f13.5,',',f12.5,',',f9.5,','
     1       ,f9.5,',',f12.5,',',f12.5)
 130      continue
 140      continue
           CLOSE (UNIT=7)
           CLOSE (UNIT=6)
           CLOSE (UNIT=5)
      END
```

INPUT FILE : UGTHEO.DAT

THEODORSEN FUNCTION C(k)

10.00	0.5006	-0.0124
6.000	0.5017	-0.0206
4.000	0.5037	-0.0305
3.000	0.5063	-0.0400
2.000	0.5129	-0.0577
1.500	0.5210	-0.0736
1.200	0.5300	-0.0877
1.000	0.5394	-0.1003
0.800	0.5541	-0.1165
0.600	0.5788	-0.1378
0.500	0.5979	-0.1507
0.400	0.6250	-0.1650
0.300	0.6550	-0.1793
0.200	0.7276	-0.1886
0.150	0.7737	-0.1857
0.130	0.7954	-0.1820
0.110	0.8188	-0.1766
0.100	0.8320	-0.1723
0.050	0.9090	-0.1305
0.025	0.9545	-0.0872
0.010	0.9824	-0.0482
k	F	iG

INPUT FILE : CONST.DAT

2 2 2 2 2

OUTPUT FILE : UGDATA.OUT

UG METHOD

u	r(alp)	ah	wben/wtor
50.0	.5	-.5	.2

k	U/b*wtor	w/wtor	g	damping ratio
10.000,	.01970,	.19698,	-.00209,	.00104,
10.000,	.11493,	1.14932,	-.00569,	.00284,
6.000,	.03283,	.19699,	-.00348,	.00174,
6.000,	.19150,	1.14900,	-.00948,	.00474,
4.000,	.04925,	.19701,	-.00524,	.00262,
4.000,	.28709,	1.14837,	-.01423,	.00712,
3.000,	.06568,	.19704,	-.00703,	.00351,
3.000,	.38249,	1.14748,	-.01901,	.00950,
2.000,	.09856,	.19711,	-.01067,	.00533,
2.000,	.57247,	1.14494,	-.02861,	.01431,
1.500,	.13147,	.19721,	-.01444,	.00722,
1.500,	.76090,	1.14135,	-.03831,	.01916,
1.200,	.16444,	.19732,	-.01834,	.00917,
1.200,	.94726,	1.13672,	-.04810,	.02405,
1.000,	.19746,	.19746,	-.02238,	.01119,
1.000,	1.13102,	1.13102,	-.05796,	.02898,
.800,	.24711,	.19769,	-.02871,	.01436,
.800,	1.40063,	1.12050,	-.07275,	.03638,
.600,	.33024,	.19814,	-.03995,	.01998,
.600,	1.82975,	1.09785,	-.09686,	.04843,
.500,	.39712,	.19856,	-.04954,	.02477,
.500,	2.15096,	1.07548,	-.11494,	.05747,
.400,	.49815,	.19926,	-.06485,	.03243,
.400,	2.58976,	1.03590,	-.13857,	.06929,
.300,	.66874,	.20062,	-.09274,	.04637,
.300,	3.19652,	.95896,	-.16476,	.08238,
.200,	1.02048,	.20410,	-.15760,	.07880,
.200,	3.97729,	.79546,	-.16106,	.08053,
.150,	1.39164,	.20875,	-.23812,	.11906,
.150,	4.35545,	.65332,	-.09675,	.04837,
.130,	1.63333,	.21233,	-.29866,	.14933,
.130,	4.47879,	.58224,	-.03713,	.01856,
.110,	1.98367,	.21820,	-.40100,	.20050,
.110,	4.59055,	.50496,	.06404,	-.03202,
.100,	2.22456,	.22246,	-.48395,	.24197,
.100,	4.66272,	.46627,	.14335,	-.07167,
.050,	4.97074,	.24854,	-1.75199,	.87599,
.050,	7.19299,	.35965,	1.11015,	-.55508,
.025,	10.37782,	.25945,	-4.24223,	2.12111,
.025,	13.98072,	.34952,	2.64402,	-1.32201,
.010,	27.41923,	.27419,	-12.28424,	6.14212,
.010,	34.48513,	.34485,	6.79696,	-3.39848,

```
$storage:2
$floatcalls
$debug
C-------------------------------------------------------------------
C    p-k EIGENVALUE SOLUTION
C-------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-Z)
      INTEGER ib,n,ia,ijob,iz,ier,infer,i,j
      COMPLEX*16 eval(4),A(4,4),B(4,4),EIGA(4),EIGB(4),Z(4,4),WK(6,12)
      REAL k(40),kc(40),F(40),G(40),u,vel
      REAL abar(4),bbar(4),ndvel(4,40),ndw(4,40)
      REAL gdamp(4,40),dratio(4,40),kg,fg,gg,step
      INTEGER temp,it,pt,pts,test,soln1,isoln,check,soln(2)
C
C Read in constants.
      OPEN (UNIT=1,FILE='CONST1.DAT',STATUS='OLD')
           READ (1,*) ia,ib,iz,n,ijob
      CLOSE (UNIT=1)
C
C open required files
      OPEN (UNIT=2,FILE='CONST.OUT',STATUS='NEW')
      OPEN (UNIT=4,FILE='PKDATA.OUT',STATUS='NEW')
      OPEN (UNIT=5,FILE='PKDAMP.OUT',STATUS='NEW')
      OPEN (UNIT=6,FILE='PKFREQ.OUT',STATUS='NEW')
C
C required constants
              u=50.0
              wrat=0.2
              r=0.5
              ah=-0.5
C
              x=0.25
              semic=1.0
              wtor=1.0
              wben=wrat*wtor
              pts=41
C
          write(4,32)
          write(4,35)
          write(4,35)
          write(4,33)
          write(4,35)
          write(4,34) u,r,ah,wrat
          write(4,35)
          write(4,36)
          write(4,35)
 32       format('  PK METHOD ')
 33       format('   u          r(alp)     ah      wben/wtor')
 34       format(f7.1,4f10.1)
 35       format(' ')
 36       format('   k    ND VEL    b (damping)      w/wtor  ')
C
C Theodorsen function C(k)= F+iG given by k, where k=kc/2
```

```
C thus if kc=0.2,  k=0.1 therefore require C(k)=C(0.1)
      OPEN (UNIT=3,FILE='PKTHEO.DAT',STATUS='OLD')
          do 50 pt=1,pts
             READ (3,*) k(pt),F(pt),G(pt)
             kc(pt)=2.0*k(pt)
 50       continue
      CLOSE (UNIT=3)
C
C
C iterate velocity
             vel=0.0
             step =0.5
 60          vel=vel+step
             if(vel/(semic*wtor).gt.5.0) goto 120
          do 95 test=1,2
             check=0
             count=0
             kg=((wtor+wben)/2)*semic/vel
C
C interpolation of reduced frequency, F and G
 51          tempw=vel*kg/semic
             it=1
 52          it=it+1
             if(kg.lt.0.01) kg=0.01
             if(k(it).lt.kg) goto 53
             goto 52
 53          Fg=f(it-1)+(f(it)-f(it-1))*(kg-k(it-1))/(k(it)-k(it-1))
             Gg=g(it-1)+(g(it)-g(it-1))*(kg-k(it-1))/(k(it)-k(it-1))
 54          format(' ',8f12.5)
C
C formation of COMPLEX matrices [A] and [B] to
C solve eigenvalue problem given by [A]{x} = lambda [B]{x}
             ar11=0.0
             ai11=0.0
             ar12=0.0
             ai12=0.0
             ar13=wrat**2*wtor**2
             ai13=0.0
             ar14=(2/u)*(vel/semic)**2*Fg
             ai14=(2/u)*(vel/semic)**2*Gg
             ar21=0.0
             ai21=0.0
             ar22=0.0
             ai22=0.0
             ar23=0.0
             ai23=0.0
             ar24=r**2*wtor**2-2*(ah+0.5)*(Fg/u)*(vel/semic)**2
             ai24=-2*(ah+0.5)*(Gg/u)*(vel/semic)**2
             ar31=-1.0
             ai31=0.0
             ar32=0.0
             ai32=0.0
             ar33=0.0
             ai33=0.0
```

```
ar34=0.0
ai34=0.0
ar41=0.0
ai41=0.0
ar42=-1.0
ai42=0.0
ar43=0.0
ai43=0.0
ar44=0.0
ai44=0.0
   a(1,1)=dcmplx(ar11,ai11)
   a(1,2)=dcmplx(ar12,ai12)
   a(1,3)=dcmplx(ar13,ai13)
   a(1,4)=dcmplx(ar14,ai14)
   a(2,1)=dcmplx(ar21,ai21)
   a(2,2)=dcmplx(ar22,ai22)
   a(2,3)=dcmplx(ar23,ai23)
   a(2,4)=dcmplx(ar24,ai24)
   a(3,1)=dcmplx(ar31,ai31)
   a(3,2)=dcmplx(ar32,ai32)
   a(3,3)=dcmplx(ar33,ai33)
   a(3,4)=dcmplx(ar34,ai34)
   a(4,1)=dcmplx(ar41,ai41)
   a(4,2)=dcmplx(ar42,ai42)
   a(4,3)=dcmplx(ar43,ai43)
   a(4,4)=dcmplx(ar44,ai44)
br11=1+1/u
bi11=0.0
br12=x-ah/u
bi12=0.0
br13=2*(Fg/u)*(vel/semic)
bi13=2*(Gg/u)*(vel/semic)
br14=((1/u)+2*(Fg/u)*(0.5-ah))*(vel/semic)
bi14=(2*(Gg/u)*(0.5-ah))*(vel/semic)
br21=x-ah/u
bi21=0.0
br22=r**2+(0.125+ah**2)/u
bi22=0.0
br23=-2*(ah+0.5)*(Fg/u)*(vel/semic)
bi23=-2*(ah+0.5)*(Gg/u)*(vel/semic)
br24=((0.5-ah)/u-2*(ah+0.5)*(Fg/u)*(0.5-ah))*(vel/semic)
bi24= (-2*(ah+0.5)*(Gg/u)*(0.5-ah))*(vel/semic)
br31=0.0
bi31=0.0
br32=0.0
bi32=0.0
br33=1.0
bi33=0.0
br34=0.0
bi34=0.0
br41=0.0
bi41=0.0
br42=0.0
bi42=0.0
```

```
                br43=0.0
                bi43=0.0
                br44=1.0
                bi44=0.0
                   b(1,1)=dcmplx(-br11,-bi11)
                   b(1,2)=dcmplx(-br12,-bi12)
                   b(1,3)=dcmplx(-br13,-bi13)
                   b(1,4)=dcmplx(-br14,-bi14)
                   b(2,1)=dcmplx(-br21,-bi21)
                   b(2,2)=dcmplx(-br22,-bi22)
                   b(2,3)=dcmplx(-br23,-bi23)
                   b(2,4)=dcmplx(-br24,-bi24)
                   b(3,1)=dcmplx(-br31,-bi31)
                   b(3,2)=dcmplx(-br32,-bi32)
                   b(3,3)=dcmplx(-br33,-bi33)
                   b(3,4)=dcmplx(-br34,-bi34)
                   b(4,1)=dcmplx(-br41,-bi41)
                   b(4,2)=dcmplx(-br42,-bi42)
                   b(4,3)=dcmplx(-br43,-bi43)
                   b(4,4)=dcmplx(-br44,-bi44)
C
C subrountine for eigenvalue calculation
C abar : real part of eval( )
C bbar : imaginary part of eval( )
      CALL EIGZC(A,IA,B,IB,N,IJOB,EIGA,EIGB,Z,IZ,WK,INFER,IER)
C
          do 5 I=1,n
 5            EVAL(I)=EIGA(I)/EIGB(I)
 21           format('  perfomance index is', 2f10.3)
c              write(2,22)eval(1)
                abar(1)=dreal(eval(1))
                bbar(1)=dimag(eval(1))
c              write(2,22)eval(2)
                abar(2)=dreal(eval(2))
                bbar(2)=dimag(eval(2))
c              write(2,22)eval(3)
                abar(3)=dreal(eval(3))
                bbar(3)=dimag(eval(3))
c              write(2,22)eval(4)
                abar(4)=dreal(eval(4))
                bbar(4)=dimag(eval(4))
C
C obtaining eigenvalues with positive imaginary parts for analysis
              j=0
          do 14 isoln=1,4
              if (bbar(isoln).gt.0.0) goto 12
              goto 14
 12           j=j+1
              soln(j)=isoln
 14       continue
              if(check.eq.0) goto 23
              if(test.eq.1.and.abs(tempw-bbar(soln(1))).lt.abs(
     1        tempw-bbar(soln(2)))) goto 23
              if(test.eq.2.and.abs(tempw-bbar(soln(2))).lt.abs(
```

```
     1          tempw-bbar(soln(1)))) goto 23
                if(test.eq.2) goto 19
                temp=soln(1)
                soln(1)=soln(2)
                soln(2)=temp
                goto 23
 19             temp=soln(2)
                soln(2)=soln(1)
                soln(1)=temp
C
C convergence check
 23             check=1
                w=dimag(eval(soln(test)))
                if(abs(w-tempw).lt.0.001) goto 94
                kg=w*semic/vel
                goto 51
C
C writing final data to file
 94             write(4,27)kg,vel/(semic*wtor),dreal(eval(soln(test))),
     1          dimag(eval(soln(test)))/wtor
                write(5,27) vel/(semic*wtor),-dreal(eval(soln(test)))/
     1          ((dreal(eval(soln(test))))**2+(dimag(eval(soln(test))))
     1          **2)**0.5
                write(6,27) vel/(semic*wtor),dimag(eval(soln(test)))/wtor
 27             format(f6.4,',',f7.4,',',f12.5,',',f12.5)
 22             format(8f12.5)
 95         continue
C
C changing velocity step size
                if(vel/(semic*wtor).ge.5.0) step=0.1
C                if(vel/(semic*wtor).ge.4.8) step=0.05
C                if(vel/(semic*wtor).ge.5.0) step=0.5
c                if(vel/(semic*wtor).ge.6.0) goto 120
 100            goto 60
 120     CLOSE (UNIT=4)
 200     CLOSE (UNIT=2)
 300         END
```

INPUT FILE : PKTHEO.DAT

THEODORSEN FUNCTION C(k)

k	F	iG
10.00	0.5006	-0.0124
6.000	0.5017	-0.0206
4.000	0.5037	-0.0305
3.000	0.5063	-0.0400
2.000	0.5129	-0.0577
1.500	0.5210	-0.0736
1.200	0.5300	-0.0877
1.000	0.5394	-0.1003
0.990	0.5400	-0.1010
0.880	0.5474	-0.1095
0.800	0.5541	-0.1165
0.770	0.5570	-0.1193
0.660	0.5699	-0.1308
0.600	0.5788	-0.1378
0.560	0.5857	-0.1428
0.550	0.5876	-0.1441
0.500	0.5979	-0.1507
0.440	0.6130	-0.1592
0.400	0.6250	-0.1650
0.340	0.6469	-0.1738
0.330	0.6512	-0.1752
0.325	0.6535	-0.1759
0.320	0.6558	-0.1766
0.315	0.6581	-0.1773
0.310	0.6604	-0.1779
0.305	0.6627	-0.1786
0.300	0.6650	-0.1793
0.240	0.6989	-0.1862
0.220	0.7125	-0.1877
0.200	0.7276	-0.1886
0.160	0.7628	-0.1876
0.120	0.8063	-0.1801
0.110	0.8188	-0.1766
0.100	0.8320	-0.1723
0.080	0.8604	-0.1604
0.060	0.8920	-0.1426
0.050	0.9090	-0.1305
0.040	0.9267	-0.1160
0.025	0.9545	-0.0872
0.010	0.9824	-0.0482
0.000	1.0000	-0.0000

INPUT FILE : CONST1.DAT

4 4 4 4 2

OUTPUT FILE : PKDATA.OUT

PK METHOD

u	r(alp)	ah	wben/wtor
50.0	.5	-.5	.2

k	ND VEL	b (damping)	w/wtor
.3966,	.5000,	-.00640,	.19917
2.2909,	.5000,	-.01436,	1.14593
.2034,	1.0000,	-.01487,	.20335
1.1348,	1.0000,	-.02958,	1.13484
.1394,	1.5000,	-.02492,	.20916
.7434,	1.5000,	-.04621,	1.11514
.1087,	2.0000,	-.03677,	.21735
.5425,	2.0000,	-.06450,	1.08521
.0916,	2.5000,	-.05169,	.22893
.4169,	2.5000,	-.08437,	1.04258
.3275,	3.0000,	-.10547,	.98336
.0820,	3.0000,	-.07180,	.24614
.2570,	3.5000,	-.12613,	.89945
.0781,	3.5000,	-.10247,	.27359
.1923,	4.0000,	-.13992,	.76981
.0809,	4.0000,	-.16167,	.32367
.1228,	4.5000,	-.01093,	.55238
.0749,	4.5000,	-.33777,	.33681
.1020,	5.0000,	.12384,	.50991
.0284,	5.0000,	-.60286,	.14083

```
C-----------------------------------------------------------------------
C     SEQUENTIAL SIMPLEX PROGRAM   (MINIMIZATION PROBLEM SOLVING)
C     RATIONAL FUNCTION APPROXIMATION - INCOMPRESSIBLE REGIME
C-----------------------------------------------------------------------
C
C     OPERATION
C     REQUIRE: 1) INPUT OF  ICOUNT, N, REQMIN, START(I),STEP(I)
C              2) USER SPECIFIED FUNCTION SUBPROGRAM (DOUBLE PRECISION
C                 FUNCTION FN(X) - MINIMIZATION FUNCTION)
C
C
      DOUBLE PRECISION START(20),STEP(20),XMIN(20),
     1XSEC(20),YNEWLO,YSEC,REQMIN
C
      OPEN(UNIT=6,FILE='DATA.OUT')
C
C       ***** INPUT REQUIRED *****************
C       *                                    *
C       *   ICOUNT,N,REQMIN,START(N),STEP(N) *
C       *                                    *
C       **************************************
C
C LET N=4 AT ALL TIMES, SIMPLY SET START(I)=0.0 AND STEP(I)=0.0
C
      ICOUNT=260
      N=4
      REQMIN=0.0000000000001
      START(1)=0.015
      START(2)=0.084464515
      START(3)=0.25635
      START(4)=0.76550587
      STEP(1)=0.0105
      STEP(2)=0.01
      STEP(3)=0.05
      STEP(4)=0.1
C
C
      DO 60 I=1,N
      XMIN(I)=0.D0
      XSEC(I)=0.D0
 60   CONTINUE
      YNEWLO=0.D0
      YSEC=0.D0
C
C              CALL NELDER-MEAD SUBROUTINE
C
      CALL NELMIN(N,START,XMIN,XSEC,YNEWLO,YSEC,
     1REQMIN,STEP,ICOUNT)
C
C              OUTPUT FROM PROGRAM
C
      WRITE(6,64)
      WRITE(6,65) ICOUNT
      WRITE(6,75)
```

```
      WRITE(6,77)
      DO 79 I=1,N
 79   WRITE(6,80) I,XSEC(I),XMIN(I)
      WRITE(6,82)
      WRITE(6,83)
      WRITE(6,84) YSEC,YNEWLO
C
 64   FORMAT(6X,1H //,1H ,42H SEQUENTIAL SIMPLEX : PROBLEM MINIMIZATION)
 65   FORMAT(1H //,1H ,I5,12H TRIALS USED/)
 75   FORMAT(1H ,21X,9HESTIMATES/)
 77   FORMAT(6X,1H ,9HPARAMETER,7X,12HNEXT-TO-BEST,8X,
     14HBEST/)
 80   FORMAT(1H ,I5,2F20.7)
 82   FORMAT(6X,1H //,1H ,6X,15HFUNCTION VALUES/)
 83   FORMAT(6X,1H ,5X,13H NEXT-TO-BEST,8X,4HBEST/)
 84   FORMAT(6X,1H ,2F15.9)
      STOP
      END
C
C             NELDER-MEAD SUBROUTINE
C
      SUBROUTINE NELMIN(N,START,XMIN,XSEC,YNEWLO,YSEC,
     1REQMIN,STEP,ICOUNT)
      DOUBLE PRECISION START(N),STEP(N),XMIN(N),
     1XSEC(N),YNEWLO,YSEC,REQMIN,P(20,21),PSTAR(20),
     2P2STAR(20),PBAR(20),Y(20),DN,Z,YLO,RCOEFF,
     3YSTAR,ECOEFF,Y2STAR,CCOEFF,FN,DABIT,DCHK,
     4COORD1,COORD2
      DATA RCOEFF/1.0D0/,ECOEFF/2.0D0/,CCOEFF/0.5D0/
      KCOUNT=ICOUNT
      ICOUNT=0
C
      IF(REQMIN.LE.0.D0) ICOUNT=ICOUNT-1
      IF(N.LE.0) ICOUNT=ICOUNT-10
      IF(N.GT.20) ICOUNT=ICOUNT-10
      IF(ICOUNT.LT.0) RETURN
C
      DABIT=2.04607D-35
      BIGNUM=1.0D38
      KONVGE=5
      XN=FLOAT(N)
      DN=DFLOAT(N)
      NN=N+1
C
C     CONSTRUCTION OF INITIAL SIMPLEX
C
 1001 DO 1 I=1,N
    1 P(I,NN)=START(I)
      Y(NN)=FN(START)
      ICOUNT=ICOUNT+1
      DO 2 J=1,N
      DCHK=START(J)
      START(J)=DCHK+STEP(J)
      DO 3 I=1,N
```

```
    3 P(I,J)=START(I)
      Y(J)=FN(START)
      ICOUNT=ICOUNT+1
    2 START(J)=DCHK
C
C      SIMPLEX CONSTRUCTION COMPLETE
C
C      FIND HIGHEST AND LOWEST Y VALUES
C      YNEWLO (Y(IHI)) INDICATES THE VERTEX OF
C      THE SIMPLEX TO BE REPLACED
C
 1000 YLO=Y(1)
      YNEWLO=YLO
      ILO=1
      IHI=1
      DO 5 I=2,NN
      IF(Y(I).GE.YLO) GOTO 4
      YLO=Y(I)
      ILO=I
    4 IF(Y(I).LE.YNEWLO) GOTO 5
      YNEWLO=Y(I)
      IHI=I
    5 CONTINUE
C
C       PERFORM CONVERGENCE CHECKS ON FUNCTION
C
      DCHK=(YNEWLO+DABIT)/(YLO+DABIT)-1.DO
      IF(DABS(DCHK).LT.REQMIN) GOTO 900
C
      KONVGE=KONVGE-1
      IF(KONVGE.NE.0) GOTO 2020
      KONVGE=5
C
C       CHECK CONVERGENCE OF COORDINATES ONLY
C       EVERY 5 SIMPLEXES
C
      DO 2015 I=1,N
      COORD1=P(I,1)
      COORD2=COORD1
      DO 2010 J=2,NN
      IF(P(I,J).GE.COORD1) GOTO 2005
      COORD1=P(I,J)
 2005 IF(P(I,J).LE.COORD2) GOTO 2010
      COORD2=P(I,J)
 2010 CONTINUE
      DCHK=(COORD2+DABIT)/(COORD1+DABIT)-1.DO
      IF(DABS(DCHK).GT.REQMIN) GOTO 2020
 2015 CONTINUE
      GOTO 900
 2020 IF(ICOUNT.GE.KCOUNT) GOTO 900
C
C        CALCULATE PBAR, THE CENTROID OF THE
C        SIMPLEX VERTICES EXCEPTING THAT WITH
C        Y VALUE YNEWLO
```

```
C
      DO 7 I=1,N
      Z=0.0D0
      DO 6 J=1,NN
    6 Z=Z+P(I,J)
      Z=Z-P(I,IHI)
    7 PBAR(I)=Z/DN
C
C        REFLECTION THROUGH THE CENTROID
C
      DO 8 I=1,N
    8 PSTAR(I)=(1.0D0+RCOEFF)*PBAR(I)-RCOEFF*P(I,IHI)
      YSTAR=FN(PSTAR)
      ICOUNT=ICOUNT+1
      IF(YSTAR.GE.YLO) GOTO 12
      IF(ICOUNT.GE.KCOUNT) GOTO 19
C
C        SUCCESSFUL REFLECTION, SO EXTENSION
C
      DO 9 I=1,N
    9 P2STAR(I)=ECOEFF*PSTAR(I)+(1.0D0-ECOEFF)*PBAR(I)
      Y2STAR=FN(P2STAR)
      ICOUNT=ICOUNT+1
C
C        RETAIN EXTENSION OR CONTRACTION
C
      IF(Y2STAR.GE.YSTAR) GOTO 19
   10 DO 11 I=1,N
   11 P(I,IHI)=P2STAR(I)
      Y(IHI)=Y2STAR
      GOTO 1000
C
C        NO EXTENSION
C
   12 L=0
      DO 13 I=1,NN
      IF(Y(I).GT.YSTAR) L=L+1
   13 CONTINUE
      IF(L.GT.1) GOTO 19
      IF(L.EQ.0) GOTO 15
C
C        CONTRACTION ON THE REFLECTION SIDE OF THE
C        CENTROID
C
      DO 14 I=1,N
   14 P(I,IHI)=PSTAR(I)
      Y(IHI)=YSTAR
C
C        CONTRACTION ON THE Y(IHI) SIDE OF THE CENTROID
C
   15 IF(ICOUNT.GE.KCOUNT) GOTO 900
      DO 16 I=1,N
   16 P2STAR(I)=CCOEFF*P(I,IHI)+(1.0D0-CCOEFF)*PBAR(I)
      Y2STAR=FN(P2STAR)
```

```
      ICOUNT=ICOUNT+1
      IF(Y2STAR.LT.Y(IHI)) GOTO 10
C
C        CONTRACT THE WHOLE SIMPLEX
C
      DO 18 J=1,NN
      DO 17 I=1,N
      P(I,J)=(P(I,J)+P(I,ILO))*0.5D0
   17 XMIN(I)=P(I,J)
      Y(J)=FN(XMIN)
   18 CONTINUE
      ICOUNT=ICOUNT+NN
      IF(ICOUNT.LT.KCOUNT) GOTO 1000
      GOTO 900
C
C        RETAIN REFLECTION
C
C    19 CONTINUE
   19 DO 20 I=1,N
   20 P(I,IHI)=PSTAR(I)
      Y(IHI)=YSTAR
      GOTO 1000
C
C        SELECT THE TWO BEST FUNCTION VALUES (YNEWLO
C        AND YSEC) AND THEIR COORDS. (XMIN AND XSEC)
C
  900 DO 23 J=1,NN
      DO 22 I=1,N
   22 XMIN(I)=P(I,J)
      Y(J)=FN(XMIN)
   23 CONTINUE
      YNEWLO=BIGNUM
      DO 24 J=1,NN
      IF(Y(J).GE.YNEWLO) GOTO 24
      YNEWLO=Y(J)
      IBEST=J
   24 CONTINUE
      Y(IBEST)=BIGNUM
      YSEC=BIGNUM
      DO 25 J=1,NN
      IF(Y(J).GE.YSEC) GOTO 25
      YSEC=Y(J)
      ISEC=J
   25 CONTINUE
      DO 26 I=1,N
      XMIN(I)=P(I,IBEST)
      XSEC(I)=P(I,ISEC)
   26 CONTINUE
      RETURN
      END
C
C    APPROXIMATING FUNCTION : PADE APPROXIMATE WITH LAG TERMS
C
       DOUBLE PRECISION FUNCTION FN(lag)
```

```
      IMPLICIT REAL*8 (A-Z)
      COMPLEX*16 ckquad,cklag,ckbar(40,2,2),p
      DOUBLE PRECISION LAG(4)
      REAL*8 w(40),k(40,2,2),F(40,2,2),G(40,2,2),a(9,18),b(9,1),x(9,1)
      REAL*8 mn(40)
      INTEGER i,m,in,nst,nf,j,prt,totlag,c,r,count,icount
      INTEGER D,COL,ROW
C
C
        OPEN(UNIT=1,FILE='LAG.OUT')
        OPEN(UNIT=10,FILE='LAPLACE.DAT')
C
C
C     ********** INPUT REQUIRED **********************
C     *                                              *
C     *   OPEN REQUIRED INPUT TABULAR DATA FILES     *
C     *   MINIMUM ERROR DESIRED FOR TABLE            *
C     *   TOTAL # OF ITERATIONS  ICOUNT              *
C     *   INPUT NUMBER OF LAG TERMS REQUIRED         *
C     *   # OF POINTS (m) IN DATA FILES              *
C     *                                              *
C     ************************************************
C
         icount=260+5
         minerr=0.0000167
         totlag=4
         m=40
C
         count=count+1
         if(count.gt.1) goto 88
C
C   FILES CONTAIN   k(i,r,c), F(i,r,c), G(i,r,c)
C   k(i) REDUCED FREQUENCY
C   F(i) REAL
C   G(i) IMAGINARY
C
C APPROXIMATING C(k) AND C(k)*ik
        OPEN(UNIT=2,FILE='N11.DAT',STATUS='OLD')
        OPEN(UNIT=4,FILE='N12.DAT',STATUS='OLD')
        do 150 i=1,m
            READ(2,*) k(i,1,1),F(i,1,1),G(i,1,1)
            READ(4,*) k(i,1,2),F(i,1,2),G(i,1,2)
 150    continue
        CLOSE(UNIT=4)
        CLOSE(UNIT=2)
C
C   REQUIRED CONSTANTS
C           matrix [A] ( size  in x in )
 88         nst=4
            in=nst+(totlag-1)
            nf=2*in
C
C NOTE: lag(1) <> lag(2) <> lag(3) <> lag(4)
C       lag(1), lag(2), lag(3), lag(4) must be all non-negative
```

```
C BOUNDARY (CONSTRAINT) CONDITIONS
          do 156 i=1,totlag
             if(lag(i).lt.0.0) lag(i)=i*0.0001
          do 157 j=1,totlag-1
             if(i.eq.j) goto 157
             if(lag(i).eq.lag(j)) goto 163
             goto 157
 163         print 164, j
 164         format(i5,' EQUAL LAG TERMS REQUIRES IMMEDIATE STOP')
             stop
 157      continue
 156      continue
C
             FN=0.0
C
          do 750 c=1,2
             r=1
             err=0.0
C    RESETTING MATRIX ELEMENT VALUES TO ZERO
 390      do 410 i=1,in
          do 400 j=1,nf
             a(i,j)=0.0
 400      continue
             b(i,1)=0.0
             x(i,1)=0.0
 410      continue
C
C    CREATING MATRICES [A] AND [B] :   SOLVE   [A] {x} = [B]
          do 175 i=1,m
             bk1=k(i,r,c)**2/(k(i,r,c)**2+lag(1)**2)
             if(totlag.eq.1) goto 415
             bk2=k(i,r,c)**2/(k(i,r,c)**2+lag(2)**2)
             if(totlag.eq.2) goto 415
             bk3=k(i,r,c)**2/(k(i,r,c)**2+lag(3)**2)
             if(totlag.eq.3) goto 415
             bk4=k(i,r,c)**2/(k(i,r,c)**2+lag(4)**2)
 415         mn(i)=F(i,r,c)**2+G(i,r,c)**2
             if(mn(i).lt.1.0) mn(i)=1.0
             w(i)=1.0
             w(i)=w(i)/mn(i)
             a(1,1)=a(1,1)+w(i)*(1.0)
             a(1,2)=a(1,2)+w(i)*(0.0)
             a(1,3)=a(1,3)+w(i)*(-k(i,r,c)**2)
             a(1,4)=a(1,4)+w(i)*(bk1)
             a(1,5)=a(1,5)+w(i)*(bk2)
             a(1,6)=a(1,6)+w(i)*(bk3)
             a(1,7)=a(1,7)+w(i)*(bk4)
             a(2,1)=a(2,1)+w(i)*(0.0)
             a(2,2)=a(2,2)+w(i)*(k(i,r,c)**2)
             a(2,3)=a(2,3)+w(i)*(0.0)
             a(2,4)=a(2,4)+w(i)*(bk1*lag(1))
             a(2,5)=a(2,5)+w(i)*(bk2*lag(2))
             a(2,6)=a(2,6)+w(i)*(bk3*lag(3))
             a(2,7)=a(2,7)+w(i)*(bk4*lag(4))
```

```
      a(3,1)=a(3,1)+w(i)*(-k(i,r,c)**2)
      a(3,2)=a(3,2)+w(i)*(0.0)
      a(3,3)=a(3,3)+w(i)*(k(i,r,c)**4)
      a(3,4)=a(3,4)+w(i)*(-k(i,r,c)**2*bk1)
      a(3,5)=a(3,5)+w(i)*(-k(i,r,c)**2*bk2)
      a(3,6)=a(3,6)+w(i)*(-k(i,r,c)**2*bk3)
      a(3,7)=a(3,7)+w(i)*(-k(i,r,c)**2*bk4)
      a(4,1)=a(4,1)+w(i)*(bk1)
      a(4,2)=a(4,2)+w(i)*(bk1*lag(1))
      a(4,3)=a(4,3)+w(i)*(-k(i,r,c)**2*bk1)
      a(4,4)=a(4,4)+w(i)*(bk1**2*(1+(lag(1)**2/k(i,r,c)**2)))
      a(4,5)=a(4,5)+w(i)*(bk1*bk2*(1+(lag(1)*lag(2)/
     1     k(i,r,c)**2)))
      a(4,6)=a(4,6)+w(i)*(bk1*bk3*(1+(lag(1)*lag(3)/
     1     k(i,r,c)**2)))
      a(4,7)=a(4,7)+w(i)*(bk1*bk4*(1+(lag(1)*lag(4)/
     1     k(i,r,c)**2)))
      a(5,1)=a(5,1)+w(i)*(bk2)
      a(5,2)=a(5,2)+w(i)*(bk2*lag(2))
      a(5,3)=a(5,3)+w(i)*(-k(i,r,c)**2*bk2)
      a(5,4)=a(5,4)+w(i)*(bk1*bk2*(1+(lag(1)*lag(2)/
     1     k(i,r,c)**2)))
      a(5,5)=a(5,5)+w(i)*(bk2*bk2*(1+(lag(2)*lag(2)/
     1     k(i,r,c)**2)))
      a(5,6)=a(5,6)+w(i)*(bk3*bk2*(1+(lag(3)*lag(2)/
     1     k(i,r,c)**2)))
      a(5,7)=a(5,7)+w(i)*(bk4*bk2*(1+(lag(4)*lag(2)/
     1     k(i,r,c)**2)))
      a(6,1)=a(6,1)+w(i)*(bk3)
      a(6,2)=a(6,2)+w(i)*(bk3*lag(3))
      a(6,3)=a(6,3)+w(i)*(-k(i,r,c)**2*bk3)
      a(6,4)=a(6,4)+w(i)*(bk1*bk3*(1+(lag(1)*lag(3)/
     1     k(i,r,c)**2)))
      a(6,5)=a(6,5)+w(i)*(bk2*bk3*(1+(lag(2)*lag(3)/
     1     k(i,r,c)**2)))
      a(6,6)=a(6,6)+w(i)*(bk3*bk3*(1+(lag(3)*lag(3)/
     1     k(i,r,c)**2)))
      a(6,7)=a(6,7)+w(i)*(bk4*bk3*(1+(lag(4)*lag(3)/
     1     k(i,r,c)**2)))
      a(7,1)=a(7,1)+w(i)*(bk4)
      a(7,2)=a(7,2)+w(i)*(bk4*lag(4))
      a(7,3)=a(7,3)+w(i)*(-k(i,r,c)**2*bk4)
      a(7,4)=a(7,4)+w(i)*(bk1*bk4*(1+(lag(1)*lag(4)/
     1     k(i,r,c)**2)))
      a(7,5)=a(7,5)+w(i)*(bk2*bk4*(1+(lag(2)*lag(4)/
     1     k(i,r,c)**2)))
      a(7,6)=a(7,6)+w(i)*(bk3*bk4*(1+(lag(3)*lag(4)/
     1     k(i,r,c)**2)))
      a(7,7)=a(7,7)+w(i)*(bk4*bk4*(1+(lag(4)*lag(4)/
     1     k(i,r,c)**2)))
      b(1,1)=b(1,1)+w(i)*(f(i,r,c))
      b(2,1)=b(2,1)+w(i)*(g(i,r,c)*k(i,r,c))
      b(3,1)=b(3,1)+w(i)*(-f(i,r,c)*k(i,r,c)**2)
      b(4,1)=b(4,1)+w(i)*(bk1*(f(i,r,c)+g(i,r,c)*
```

```
     1        lag(1)/k(i,r,c)))
              b(5,1)=b(5,1)+w(i)*(bk2*(f(i,r,c)+g(i,r,c)*
     1        lag(2)/k(i,r,c)))
              b(6,1)=b(6,1)+w(i)*(bk3*(f(i,r,c)+g(i,r,c)*
     1        lag(3)/k(i,r,c)))
              b(7,1)=b(7,1)+w(i)*(bk4*(f(i,r,c)+g(i,r,c)*
     1        lag(4)/k(i,r,c)))
 175      continue
 205          format(8f10.4)
 206          format('  ')
C     PROGRAM INVMAT
C
C     A(N,NF) -- CHANGE BOTH THE DECLARATION I.E REAL A(N,NF), AS WELL
C     AS THE DECLARED VALUES OF N & NF. ALSO CHANGE THE NF VALUE IN THE
C     FORMAT STATEMENT AT THE END. I.E (NFF6.2)
C
      DO 4 ROW=1,IN
     DO 3 COL=(IN+1),NF
       A(ROW,COL)=0.0
       IF((ROW+IN).EQ.COL) A(ROW,COL)=1.0
3    CONTINUE
4     CONTINUE
      DO 50 D=1,IN
     TEMP=A(D,D)
     DO 5 COL=D,NF
       A(D,COL)=A(D,COL)/TEMP
5    CONTINUE
     DO 15 ROW=D+1,IN
       TEMP=A(ROW,D)
       DO 10 COL=D,NF
         A(ROW,COL)=A(ROW,C   ,-TEMP*A(D,COL)
10     CONTINUE
15   CONTINUE
50    CONTINUE
      DO 100 D=IN,2,-1
     DO 80 ROW=D-1,1,-1
       TEMP=A(ROW,D)
       DO 70 COL=D,NF
         A(ROW,COL)=A(ROW,COL)-TEMP*A(D,COL)
70     CONTINUE
80   CONTINUE
100   CONTINUE
      DO 120 ROW=1,IN
      DO 125 COL=IN+1,NF
      X(ROW,1)=X(ROW,1)+A(ROW,COL)*B(COL-IN,1)
125    CONTINUE
120    CONTINUE
130   FORMAT(' ',9F15.5)
C CALCULATION OF APPROXIMATE FUNCTION
C
          do 300 i=1,m
             p=dcmplx(0.0,k(i,r,c))
             ckquad=x(1,1)+x(2,1)*p+x(3,1)*p**2
             cklag=x(4,1)*p/(p+lag(1))+x(5,1)*p/(p+lag(2))+x(6,1)*
```

```
     1      p/(p+lag(3))+x(7,1)*p/(p+lag(4))
            ckbar(i,r,c)=ckquad+cklag
            mn(i)=F(i,r,c)**2+G(i,r,c)**2
            if(mn(i).lt.1.0) mn(i)=1.0
            err=err+((dreal(ckbar(i,r,c))-f(i,r,c))**2+
     1      (dimag(ckbar(i,r,c))-g(i,r,c))**2)/mn(i)
 300     continue
 301        FN=FN+err
 305        format(6X,f12.6,',',f10.5,',',f10.5,',',f10.5,',',
     1      f10.5,',',2f10.5)
            print 634, count
 500     continue
 600     if(count.lt.icount-1) goto 750
         write (1,602) r,c
         write (1,604)
         write (1,604)
         write (1,603)
         write (1,604)
         write (1,605) x(1,1),x(2,1),x(3,1),x(4,1)
         write (1,604)
         write (1,601)
         write (1,604)
         write (1,605) x(5,1),x(6,1),x(7,1)
         write (1,604)
         write (1,604)
         write (1,606)
         write (1,604)
         write (1,635) lag(1),lag(2),lag(3),lag(4),err,FN
         write (1,604)
         write (1,604)
         write (1,610)
         write (1,611)
         write (1,604)
         write (1,604)
 602     format(6X,' APPROXIMANT FUNCTION       N (',i2,',',i2,' )')
 603     format(5x,'        A0             A1            A2            A3')
 601     format(5x,'        A4             A5            A6')
 604     format('  ')
 605     format(5x,8f16.8)
 634     format(2i5,7f12.9)
 635     format(5x,7f12.9)
 606     format(5x,'       lag1          lag2          lag3          lag4',
     1   '          error')
 610     format(6X,'    k              EXACT                     APPROXIMATE')
 611     format(6X,'                   REAL          IMAG          REAL       ',
     1   ' IMAG')
         do 650 prt=1,m
         write(1,305) k(prt,r,c),f(prt,r,c),g(prt,r,c),ckbar(prt,r,c)
 650     continue
C
C CREATE FILE 'LAPLACE.DAT' CONTAINS b1, b2, ..., A0, A1, ...
         write (10,605) x(1,1),x(2,1),x(3,1),x(4,1),x(5,1),
     1   x(6,1),x(7,1)
         if(r.eq.1.and.c.eq.2) goto 584
```

```
      goto 750
 584  write (10,635) lag(1),lag(2),lag(3),lag(4),FN
C
 750  continue
      RETURN
 800  END
```

INPUT FILE : N11.DAT

10.000	0.5006	-0.0124
6.000	0.5017	-0.0206
4.000	0.5037	-0.0305
3.000	0.5063	-0.0400
2.000	0.5129	-0.0577
1.500	0.5210	-0.0736
1.200	0.5300	-0.0877
1.000	0.5394	-0.1003
0.990	0.5400	-0.1010
0.880	0.5474	-0.1095
0.800	0.5541	-0.1165
0.770	0.5570	-0.1193
0.660	0.5699	-0.1308
0.600	0.5788	-0.1378
0.560	0.5857	-0.1428
0.550	0.5876	-0.1441
0.500	0.5979	-0.1507
0.440	0.6130	-0.1592
0.400	0.6250	-0.1650
0.340	0.6469	-0.1738
0.330	0.6512	-0.1752
0.325	0.6535	-0.1759
0.320	0.6558	-0.1766
0.315	0.6581	-0.1773
0.310	0.6604	-0.1779
0.305	0.6627	-0.1786
0.300	0.6650	-0.1793
0.240	0.6989	-0.1862
0.220	0.7125	-0.1877
0.200	0.7276	-0.1886
0.160	0.7628	-0.1876
0.120	0.8063	-0.1801
0.110	0.8188	-0.1766
0.100	0.8320	-0.1723
0.080	0.8604	-0.1604
0.060	0.8920	-0.1426
0.050	0.9090	-0.1305
0.040	0.9267	-0.1160
0.025	0.9545	-0.0872
0.010	0.9824	-0.0482

DATA TABLE C(k)

k **F** iG

INPUT FILE : N12.DAT

10.0000000	.1240000	5.0060000
6.0000000	.1236000	3.0102000
4.0000000	.1220000	2.0148000
3.0000000	.1200000	1.5189000
2.0000000	.1154000	1.0258000
1.5000000	.1104000	.7815000
1.2000000	.1052400	.6360000
1.0000000	.1003000	.5394000
.9900000	.0999900	.5346000
.8800000	.0963600	.4817120
.8000000	.0932000	.4432800
.7700000	.0918610	.4288900
.6600000	.0863280	.3761340
.6000000	.0826800	.3472800
.5600000	.0799680	.3279920
.5500000	.0792550	.3231800
.5000000	.0753500	.2989500
.4400000	.0700480	.2697200
.4000000	.0660000	.2500000
.3400000	.0590920	.2199460
.3300000	.0578160	.2148960
.3250000	.0571675	.2123875
.3200000	.0565120	.2098560
.3150000	.0558495	.2073015
.3100000	.0551490	.2047240
.3050000	.0544730	.2021235
.3000000	.0537900	.1995000
.2400000	.0446880	.1677360
.2200000	.0412940	.1567500
.2000000	.0377200	.1455200
.1600000	.0300160	.1220480
.1200000	.0216120	.0967560
.1100000	.0194260	.0900680
.1000000	.0172300	.0832000
.0800000	.0128320	.0688320
.0600000	.0085560	.0535200
.0500000	.0065250	.0454500
.0400000	.0046400	.0370680
.0250000	.0021800	.0238625
.0100000	.0004820	.0098240

DATA TABLE **C(k)*ik**

k	**-G*k**	**iF*k**

OUTPUT FILE : DATA.OUT

SEQUENTIAL SIMPLEX : PROBLEM MINIMIZATION

260 TRIALS USED

ESTIMATES

PARAMETER	NEXT-TO-BEST	BEST
1	.0154409	.0154409
2	.0842865	.0842865
3	.2559507	.2559507
4	.7651135	.7651136

FUNCTION VALUES

NEXT-TO-BEST	BEST
.000011782	.000011782

OUTPUT FILE : LAG.OUT

APPROXIMANT FUNCTION N (1, 1)

A0	A1	A2	A3	
.99828226	-.00003562	.00000474	-.04120375	

A4	A5	A6
-.16289333	-.22908745	-.06459371

lag1	lag2	lag3	lag4	error
.015440852	.084286467	.255950673	.765113621	.000009348

.000009348

k	EXACT REAL	IMAG	APPROXIMATE REAL	IMAG
10.000000,	.50060,	-.01240,	.50057,	-.01257,
6.000000,	.50170,	-.02060,	.50182,	-.02047,
4.000000,	.50370,	-.03050,	.50372,	-.03025,
3.000000,	.50630,	-.04000,	.50619,	-.03976,
2.000000,	.51290,	-.05770,	.51271,	-.05764,
1.500000,	.52100,	-.07360,	.52083,	-.07373,
1.200000,	.53000,	-.08770,	.52994,	-.08798,
1.000000,	.53940,	-.10030,	.53959,	-.10051,
.990000,	.54000,	-.10100,	.54019,	-.10122,

.880000,	.54740,	-.10950,	.54767,	-.10963,
.800000,	.55410,	-.11650,	.55443,	-.11655,
.770000,	.55700,	-.11930,	.55733,	-.11934,
.660000,	.56990,	-.13080,	.57012,	-.13063,
.600000,	.57880,	-.13780,	.57895,	-.13757,
.560000,	.58570,	-.14280,	.58579,	-.14253,
.550000,	.58760,	-.14410,	.58764,	-.14381,
.500000,	.59790,	-.15070,	.59787,	-.15049,
.440000,	.61300,	-.15920,	.61278,	-.15904,
.400000,	.62500,	-.16500,	.62477,	-.16499,
.340000,	.64690,	-.17380,	.64683,	-.17397,
.330000,	.65120,	-.17520,	.65108,	-.17543,
.325000,	.65350,	-.17590,	.65328,	-.17614,
.320000,	.65580,	-.17660,	.65552,	-.17686,
.315000,	.65810,	-.17730,	.65782,	-.17756,
.310000,	.66040,	-.17790,	.66017,	-.17825,
.305000,	.66270,	-.17860,	.66257,	-.17894,
.300000,	.66500,	-.17930,	.66502,	-.17961,
.240000,	.69890,	-.18620,	.69927,	-.18633,
.220000,	.71250,	-.18770,	.71297,	-.18770,
.200000,	.72760,	-.18860,	.72802,	-.18841,
.160000,	.76280,	-.18760,	.76288,	-.18703,
.120000,	.80630,	-.18010,	.80584,	-.17977,
.110000,	.81880,	-.17660,	.81822,	-.17652,
.100000,	.83200,	-.17230,	.83138,	-.17244,
.080000,	.86040,	-.16040,	.86027,	-.16092,
.060000,	.89200,	-.14260,	.89253,	-.14285,
.050000,	.90900,	-.13050,	.90957,	-.13041,
.040000,	.92670,	-.11600,	.92684,	-.11525,
.025000,	.95450,	-.08720,	.95305,	-.08710,
.010000,	.98240,	-.04820,	.98349,	-.04764,

APPROXIMANT FUNCTION N (1, 2)

A0	A1	A2	A3	
-.00001355	.50037475	-.00003629	.00061076	

A4	A5	A6
.01380489	.05848218	.04963449

lag1	lag2	lag3	lag4	error
.015440852	.084286467	.255950673	.765113621	.000002434

.000011782

k	EXACT REAL	IMAG	APPROXIMATE REAL	IMAG
10.000000,	.12400,	5.00600,	.12582,	5.00914,
6.000000,	.12360,	3.01020,	.12292,	3.01116,

4.000000,	.12200,	2.01480,	.12110,	2.01468,
3.000000,	.12000,	1.51890,	.11938,	1.51835,
2.000000,	.11540,	1.02580,	.11536,	1.02526,
1.500000,	.11040,	.78150,	.11065,	.78113,
1.200000,	.10524,	.63600,	.10561,	.63585,
1.000000,	.10030,	.53940,	.10053,	.53954,
.990000,	.09999,	.53460,	.10023,	.53474,
.880000,	.09636,	.48171,	.09649,	.48191,
.800000,	.09320,	.44328,	.09325,	.44352,
.770000,	.09186,	.42889,	.09190,	.42912,
.660000,	.08633,	.37613,	.08621,	.37626,
.600000,	.08268,	.34728,	.08253,	.34735,
.560000,	.07997,	.32799,	.07980,	.32802,
.550000,	.07926,	.32318,	.07908,	.32318,
.500000,	.07535,	.29895,	.07522,	.29892,
.440000,	.07005,	.26972,	.06995,	.26961,
.400000,	.06600,	.25000,	.06597,	.24989,
.340000,	.05909,	.21995,	.05912,	.21990,
.330000,	.05782,	.21490,	.05786,	.21483,
.325000,	.05717,	.21239,	.05722,	.21229,
.320000,	.05651,	.20986,	.05656,	.20974,
.315000,	.05585,	.20730,	.05590,	.20719,
.310000,	.05515,	.20472,	.05523,	.20463,
.305000,	.05447,	.20212,	.05455,	.20206,
.300000,	.05379,	.19950,	.05385,	.19948,
.240000,	.04469,	.16774,	.04469,	.16780,
.220000,	.04129,	.15675,	.04127,	.15683,
.200000,	.03772,	.14552,	.03766,	.14558,
.160000,	.03002,	.12205,	.02991,	.12204,
.120000,	.02161,	.09676,	.02156,	.09669,
.110000,	.01943,	.09007,	.01941,	.09000,
.100000,	.01723,	.08320,	.01723,	.08313,
.080000,	.01283,	.06883,	.01286,	.06882,
.060000,	.00856,	.05352,	.00855,	.05356,
.050000,	.00653,	.04545,	.00650,	.04548,
.040000,	.00464,	.03707,	.00459,	.03708,
.025000,	.00218,	.02386,	.00215,	.02382,
.010000,	.00048,	.00982,	.00046,	.00983,

OUTPUT FILE : LAPLACE.DAT

```
       .99828226      -.00003562       .00000474      -.04120375
 -.16289332      -.22908745      -.06459372

      -.00001355       .50037475      -.00003629       .00061076
  .01380489       .05848217       .04963449

     .015440851  .084286462  .255950657  .765113549  .000011782
```

```
C ---------------------------------------------------------------
C     SEQUENTIAL SIMPLEX PROGRAM    (MINIMIZATION PROBLEM SOLVING)
C     RATIONAL FUNCTION APPROXIMATION - TRANSONIC REGIME
C ---------------------------------------------------------------
C
C     OPERATION
C     REQUIRE: 1) INPUT OF  ICOUNT, N, REQMIN, START(I),STEP(I)
C              2) USER SPECIFIED FUNCTION SUBPROGRAM (DOUBLE PRECISION
C                 FUNCTION FN(X) - MINIMIZATION FUNCTION)
C
C
      DOUBLE PRECISION START(20),STEP(20),XMIN(20),
     1XSEC(20),YNEWLO,YSEC,REQMIN
C
      OPEN(UNIT=6,FILE='DATA.OUT')
C
C       ***** INPUT REQUIRED ******************
C       *                                     *
C       *   ICOUNT,N,REQMIN,START(N),STEP(N)  *
C       *                                     *
C       ***************************************
C
C LET N=4 AT ALL TIMES, SIMPLY SET START(I)=0.0 AND STEP(I)=0.0
C
      ICOUNT=450
      N=4
      REQMIN=0.000000000001
      START(1)=0.000592052
      START(2)=0.07383689
      START(3)=0.27305258
      START(4)=0.47381410
      STEP(1)=0.1
      STEP(2)=0.1051
      STEP(3)=0.210412
      STEP(4)=0.310323
C
C
      DO 60 I=1,N
      XMIN(I)=0.D0
      XSEC(I)=0.D0
 60   CONTINUE
      YNEWLO=0.D0
      YSEC=0.D0
C
C               CALL NELDER-MEAD SUBROUTINE
C
      CALL NELMIN(N,START,XMIN,XSEC,YNEWLO,YSEC,
     1REQMIN,STEP,ICOUNT)
C
C                OUTPUT FROM PROGRAM
C
      WRITE(6,64)
      WRITE(6,65) ICOUNT
      WRITE(6,75)
```

```
      WRITE(6,77)
      DO 79 I=1,N
 79   WRITE(6,80) I,XSEC(I),XMIN(I)
      WRITE(6,82)
      WRITE(6,83)
      WRITE(6,84) YSEC,YNEWLO
C
 64   FORMAT(6X,1H //,6x,1H ,
     142H SEQUENTIAL SIMPLEX : PROBLEM MINIMIZATION)
 65   FORMAT(6X,1H //,6x,1H ,I5,12H TRIALS USED/)
 75   FORMAT(6X,1H ,21X,9HESTIMATES/)
 77   FORMAT(6X,1H ,9HPARAMETER,7X,12HNEXT-TO-BEST,8X,
     14HBEST/)
 80   FORMAT(6X,1H ,I5,2F20.7)
 82   FORMAT(6X,1H //,1H ,6X,15HFUNCTION VALUES/)
 83   FORMAT(6X,1H ,5X,13H NEXT-TO-BEST,8X,4HBEST/)
 84   FORMAT(6X,1H ,2F15.9)
      STOP
      END
C
C            NELDER-MEAD SUBROUTINE
C
      SUBROUTINE NELMIN(N,START,XMIN,XSEC,YNEWLO,YSEC,
     1REQMIN,STEP,ICOUNT)
      DOUBLE PRECISION START(N),STEP(N),XMIN(N),
     1XSEC(N),YNEWLO,YSEC,REQMIN,P(20,21),PSTAR(20),
     2P2STAR(20),PBAR(20),Y(20),DN,Z,YLO,RCOEFF,
     3YSTAR,ECOEFF,Y2STAR,CCOEFF,FN,DABIT,DCHK,
     4COORD1,COORD2
      DATA RCOEFF/1.0D0/,ECOEFF/2.0D0/,CCOEFF/0.5D0/
      KCOUNT=ICOUNT
      ICOUNT=0
C
      IF(REQMIN.LE.0.D0) ICOUNT=ICOUNT-1
      IF(N.LE.0) ICOUNT=ICOUNT-10
      IF(N.GT.20) ICOUNT=ICOUNT-10
      IF(ICOUNT.LT.0) RETURN
C
      DABIT=2.04607D-35
      BIGNUM=1.0D38
      KONVGE=5
      XN=FLOAT(N)
      DN=DFLOAT(N)
      NN=N+1
C
C     CONSTRUCTION OF INITIAL SIMPLEX
C
 1001 DO 1 I=1,N
    1 P(I,NN)=START(I)
      Y(NN)=FN(START)
      ICOUNT=ICOUNT+1
      DO 2 J=1,N
      DCHK=START(J)
      START(J)=DCHK+STEP(J)
```

```
      DO 3 I=1,N
    3 P(I,J)=START(I)
      Y(J)=FN(START)
      ICOUNT=ICOUNT+1
    2 START(J)=DCHK
C
C       SIMPLEX CONSTRUCTION COMPLETE
C
C       FIND HIGHEST AND LOWEST Y VALUES
C       YNEWLO (Y(IHI)) INDICATES THE VERTEX OF
C       THE SIMPLEX TO BE REPLACED
C
 1000 YLO=Y(1)
      YNEWLO=YLO
      ILO=1
      IHI=1
      DO 5 I=2,NN
      IF(Y(I).GE.YLO) GOTO 4
      YLO=Y(I)
      ILO=I
    4 IF(Y(I).LE.YNEWLO) GOTO 5
      YNEWLO=Y(I)
      IHI=I
    5 CONTINUE
C
C        PERFORM CONVERGENCE CHECKS ON FUNCTION
C
      DCHK=(YNEWLO+DABIT)/(YLO+DABIT)-1.D0
      IF(DABS(DCHK).LT.REQMIN) GOTO 900
C
      KONVGE=KONVGE-1
      IF(KONVGE.NE.0) GOTO 2020
      KONVGE=5
C
C        CHECK CONVERGENCE OF COORDINATES ONLY
C        EVERY 5 SIMPLEXES
C
      DO 2015 I=1,N
      COORD1=P(I,1)
      COORD2=COORD1
      DO 2010 J=2,NN
      IF(P(I,J).GE.COORD1) GOTO 2005
      COORD1=P(I,J)
 2005 IF(P(I,J).LE.COORD2) GOTO 2010
      COORD2=P(I,J)
 2010 CONTINUE
      DCHK=(COORD2+DABIT)/(COORD1+DABIT)-1.D0
      IF(DABS(DCHK).GT.REQMIN) GOTO 2020
 2015 CONTINUE
      GOTO 900
 2020 IF(ICOUNT.GE.KCOUNT) GOTO 900
C
C         CALCULATE PBAR, THE CENTROID OF THE
C         SIMPLEX VERTICES EXCEPTING THAT WITH
```

```
C         Y VALUE YNEWLO
C
      DO 7 I=1,N
      Z=0.0D0
      DO 6 J=1,NN
    6 Z=Z+P(I,J)
      Z=Z-P(I,IHI)
    7 PBAR(I)=Z/DN
C
C         REFLECTION THROUGH THE CENTROID
C
      DO 8 I=1,N
    8 PSTAR(I)=(1.0D0+RCOEFF)*PBAR(I)-RCOEFF*P(I,IHI)
      YSTAR=FN(PSTAR)
      ICOUNT=ICOUNT+1
      IF(YSTAR.GE.YLO) GOTO 12
      IF(ICOUNT.GE.KCOUNT) GOTO 19
C
C         SUCCESSFUL REFLECTION, SO EXTENSION
C
      DO 9 I=1,N
    9 P2STAR(I)=ECOEFF*PSTAR(I)+(1.0D0-ECOEFF)*PBAR(I)
      Y2STAR=FN(P2STAR)
      ICOUNT=ICOUNT+1
C
C         RETAIN EXTENSION OR CONTRACTION
C
      IF(Y2STAR.GE.YSTAR) GOTO 19
   10 DO 11 I=1,N
   11 P(I,IHI)=P2STAR(I)
      Y(IHI)=Y2STAR
      GOTO 1000
C
C         NO EXTENSION
C
   12 L=0
      DO 13 I=1,NN
      IF(Y(I).GT.YSTAR) L=L+1
   13 CONTINUE
      IF(L.GT.1) GOTO 19
      IF(L.EQ.0) GOTO 15
C
C         CONTRACTION ON THE REFLECTION SIDE OF THE
C         CENTROID
C
      DO 14 I=1,N
   14 P(I,IHI)=PSTAR(I)
      Y(IHI)=YSTAR
C
C         CONTRACTION ON THE Y(IHI) SIDE OF THE CENTROID
C
   15 IF(ICOUNT.GE.KCOUNT) GOTO 900
      DO 16 I=1,N
   16 P2STAR(I)=CCOEFF*P(I,IHI)+(1.0D0-CCOEFF)*PBAR(I)
```

```
      Y2STAR=FN(P2STAR)
      ICOUNT=ICOUNT+1
      IF(Y2STAR.LT.Y(IHI)) GOTO 10
C
C        CONTRACT THE WHOLE SIMPLEX
C
      DO 18 J=1,NN
      DO 17 I=1,N
      P(I,J)=(P(I,J)+P(I,ILO))*0.5D0
   17 XMIN(I)=P(I,J)
      Y(J)=FN(XMIN)
   18 CONTINUE
      ICOUNT=ICOUNT+NN
      IF(ICOUNT.LT.KCOUNT) GOTO 1000
      GOTO 900
C
C        RETAIN REFLECTION
C
C   19 CONTINUE
   19 DO 20 I=1,N
   20 P(I,IHI)=PSTAR(I)
      Y(IHI)=YSTAR
      GOTO 1000
C
C        SELECT THE TWO BEST FUNCTION VALUES (YNEWLO
C        AND YSEC) AND THEIR COORDS. (XMIN AND XSEC)
C
  900 DO 23 J=1,NN
      DO 22 I=1,N
   22 XMIN(I)=P(I,J)
      Y(J)=FN(XMIN)
   23 CONTINUE
      YNEWLO=BIGNUM
      DO 24 J=1,NN
      IF(Y(J).GE.YNEWLO) GOTO 24
      YNEWLO=Y(J)
      IBEST=J
   24 CONTINUE
      Y(IBEST)=BIGNUM
      YSEC=BIGNUM
      DO 25 J=1,NN
      IF(Y(J).GE.YSEC) GOTO 25
      YSEC=Y(J)
      ISEC=J
   25 CONTINUE
      DO 26 I=1,N
      XMIN(I)=P(I,IBEST)
      XSEC(I)=P(I,ISEC)
   26 CONTINUE
      RETURN
      END
C
C    APPROXIMATING FUNCTION : PADE APPROXIMATE WITH LAG TERMS
C
```

```
      DOUBLE PRECISION FUNCTION FN(lag)
      IMPLICIT REAL*8 (A-Z)
      COMPLEX*16 ckquad,cklag,ckbar(40,2,2),p
      DOUBLE PRECISION LAG(4)
      REAL*8 w(40),k(40,2,2),F(40,2,2),G(40,2,2),a(9,18),b(9,1),x(9,1)
      REAL*8 mn(40)
      INTEGER i,m,in,nst,nf,j,prt,totlag,c,r,count,icount
      INTEGER D,COL,ROW
C
C
        OPEN(UNIT=1,FILE='LAG.OUT')
        OPEN(UNIT=10,FILE='LAPLACE.DAT')
C
C
C     ********** INPUT REQUIRED **********************
C     *                                              *
C     *   OPEN REQUIRED INPUT TABULAR DATA FILES     *
C     *   TOTAL # OF ITERATIONS  ICOUNT (2)          *
C     *   # OF POINTS (m) IN FILES                   *
C     *   CHECK FORCED CONDITIONS                    *
C     *                                              *
C     ************************************************
C
         icount=450+5
         minerr=0.0000167
         totlag=4
         m=15
         count=count+1
         if(count.gt.1) goto 88
C
C    FILES CONTAIN   k(i,r,c), F(i,r,c), G(i,r,c)
C    k(i) REDUCED FREQUENCY
C    F(i) REAL
C    G(i) IMAGINARY
C
C APPROXIMATING UNSTEADY AERODYNAMICS
        OPEN(UNIT=2,FILE='N11.DAT',STATUS='OLD')
        OPEN(UNIT=4,FILE='N12.DAT',STATUS='OLD')
        OPEN(UNIT=7,FILE='N21.DAT',STATUS='OLD')
        OPEN(UNIT=8,FILE='N22.DAT',STATUS='OLD')
        do 150 i=1,m
            READ(2,*) k(i,1,1),F(i,1,1),G(i,1,1)
            READ(4,*) k(i,1,2),F(i,1,2),G(i,1,2)
            READ(7,*) k(i,2,1),F(i,2,1),G(i,2,1)
            READ(8,*) k(i,2,2),F(i,2,2),G(i,2,2)
 150    continue
        CLOSE(UNIT=8)
        CLOSE(UNIT=7)
        CLOSE(UNIT=4)
        CLOSE(UNIT=2)
C
C    REQUIRED CONSTANTS
C           matrix [A] ( size  in x in )
 88         nst=4
```

```
            in=nst+(totlag-1)
            nf=2*in
C
C NOTE: lag(1) <> lag(2) <> lag(3) <> lag(4)
C      lag(1), lag(2), lag(3), lag(4) must be all non-negative
C BOUNDARY (CONSTRAINT) CONDITIONS
         do 156 i=1,totlag
            if(lag(i).lt.0.0) lag(i)=i*0.000000001
         do 157 j=1,totlag-1
            if(i.eq.j) goto 157
            if(lag(i).eq.lag(j)) goto 163
            goto 157
 163        print 164, j
 164        format(i5,' EQUAL LAG TERMS REQUIRES IMMEDIATE STOP')
            stop
 157     continue
 156     continue
C
            FN=0.0
C
         do 750 r=1,2
         do 700 c=1,2
            err=0.0
C    RESETTING MATRIX ELEMENT VALUES TO ZERO
 390     do 410 i=1,in
         do 400 j=1,nf
            a(i,j)=0.0
 400     continue
            b(i,1)=0.0
            x(i,1)=0.0
 410     continue
C
C    CREATING MATRICES [A] AND [B] :  SOLVE   [A] {x} = [B]
         do 175 i=1,m
            if(k(i,r,c).eq.0.0) goto 175
            bk1=k(i,r,c)**2/(k(i,r,c)**2+lag(1)**2)
            if(totlag.eq.1) goto 415
            bk2=k(i,r,c)**2/(k(i,r,c)**2+lag(2)**2)
            if(totlag.eq.2) goto 415
            bk3=k(i,r,c)**2/(k(i,r,c)**2+lag(3)**2)
            if(totlag.eq.3) goto 415
            bk4=k(i,r,c)**2/(k(i,r,c)**2+lag(4)**2)
 415        mn(i)=F(i,r,c)**2+G(i,r,c)**2
            if(mn(i).lt.1.0) mn(i)=1.0
            w(i)=1.0
            w(i)=w(i)/mn(i)
            a(1,1)=a(1,1)+w(i)*(1.0)
            a(1,2)=a(1,2)+w(i)*(0.0)
            a(1,3)=a(1,3)+w(i)*(-k(i,r,c)**2)
            a(1,4)=a(1,4)+w(i)*(bk1)
            a(1,5)=a(1,5)+w(i)*(bk2)
            a(1,6)=a(1,6)+w(i)*(bk3)
            a(1,7)=a(1,7)+w(i)*(bk4)
            a(2,1)=a(2,1)+w(i)*(0.0)
```

```
      a(2,2)=a(2,2)+w(i)*(k(i,r,c)**2)
      a(2,3)=a(2,3)+w(i)*(0.0)
      a(2,4)=a(2,4)+w(i)*(bk1*lag(1))
      a(2,5)=a(2,5)+w(i)*(bk2*lag(2))
      a(2,6)=a(2,6)+w(i)*(bk3*lag(3))
      a(2,7)=a(2,7)+w(i)*(bk4*lag(4))
      a(3,1)=a(3,1)+w(i)*(-k(i,r,c)**2)
      a(3,2)=a(3,2)+w(i)*(0.0)
      a(3,3)=a(3,3)+w(i)*(k(i,r,c)**4)
      a(3,4)=a(3,4)+w(i)*(-k(i,r,c)**2*bk1)
      a(3,5)=a(3,5)+w(i)*(-k(i,r,c)**2*bk2)
      a(3,6)=a(3,6)+w(i)*(-k(i,r,c)**2*bk3)
      a(3,7)=a(3,7)+w(i)*(-k(i,r,c)**2*bk4)
      a(4,1)=a(4,1)+w(i)*(bk1)
      a(4,2)=a(4,2)+w(i)*(bk1*lag(1))
      a(4,3)=a(4,3)+w(i)*(-k(i,r,c)**2*bk1)
      a(4,4)=a(4,4)+w(i)*(bk1**2*(1+(lag(1)**2/k(i,r,c)**2)))
      a(4,5)=a(4,5)+w(i)*(bk1*bk2*(1+(lag(1)*lag(2)/
     1       k(i,r,c)**2)))
      a(4,6)=a(4,6)+w(i)*(bk1*bk3*(1+(lag(1)*lag(3)/
     1       k(i,r,c)**2)))
      a(4,7)=a(4,7)+w(i)*(bk1*bk4*(1+(lag(1)*lag(4)/
     1       k(i,r,c)**2)))
      a(5,1)=a(5,1)+w(i)*(bk2)
      a(5,2)=a(5,2)+w(i)*(bk2*lag(2))
      a(5,3)=a(5,3)+w(i)*(-k(i,r,c)**2*bk2)
      a(5,4)=a(5,4)+w(i)*(bk1*bk2*(1+(lag(1)*lag(2)/
     1       k(i,r,c)**2)))
      a(5,5)=a(5,5)+w(i)*(bk2*bk2*(1+(lag(2)*lag(2)/
     1       k(i,r,c)**2)))
      a(5,6)=a(5,6)+w(i)*(bk3*bk2*(1+(lag(3)*lag(2)/
     1       k(i,r,c)**2)))
      a(5,7)=a(5,7)+w(i)*(bk4*bk2*(1+(lag(4)*lag(2)/
     1       k(i,r,c)**2)))
      a(6,1)=a(6,1)+w(i)*(bk3)
      a(6,2)=a(6,2)+w(i)*(bk3*lag(3))
      a(6,3)=a(6,3)+w(i)*(-k(i,r,c)**2*bk3)
      a(6,4)=a(6,4)+w(i)*(bk1*bk3*(1+(lag(1)*lag(3)/
     1       k(i,r,c)**2)))
      a(6,5)=a(6,5)+w(i)*(bk2*bk3*(1+(lag(2)*lag(3)/
     1       k(i,r,c)**2)))
      a(6,6)=a(6,6)+w(i)*(bk3*bk3*(1+(lag(3)*lag(3)/
     1       k(i,r,c)**2)))
      a(6,7)=a(6,7)+w(i)*(bk4*bk3*(1+(lag(4)*lag(3)/
     1       k(i,r,c)**2)))
      a(7,1)=a(7,1)+w(i)*(bk4)
      a(7,2)=a(7,2)+w(i)*(bk4*lag(4))
      a(7,3)=a(7,3)+w(i)*(-k(i,r,c)**2*bk4)
      a(7,4)=a(7,4)+w(i)*(bk1*bk4*(1+(lag(1)*lag(4)/
     1       k(i,r,c)**2)))
      a(7,5)=a(7,5)+w(i)*(bk2*bk4*(1+(lag(2)*lag(4)/
     1       k(i,r,c)**2)))
      a(7,6)=a(7,6)+w(i)*(bk3*bk4*(1+(lag(3)*lag(4)/
     1       k(i,r,c)**2)))
```

```
             a(7,7)=a(7,7)+w(i)*(bk4*bk4*(1+(lag(4)*lag(4)/
     1       k(i,r,c)**2)))
             b(1,1)=b(1,1)+w(i)*(f(i,r,c))
             b(2,1)=b(2,1)+w(i)*(g(i,r,c)*k(i,r,c))
             b(3,1)=b(3,1)+w(i)*(-f(i,r,c)*k(i,r,c)**2)
             b(4,1)=b(4,1)+w(i)*(bk1*(f(i,r,c)+g(i,r,c)*
     1       lag(1)/k(i,r,c)))
             b(5,1)=b(5,1)+w(i)*(bk2*(f(i,r,c)+g(i,r,c)*
     1       lag(2)/k(i,r,c)))
             b(6,1)=b(6,1)+w(i)*(bk3*(f(i,r,c)+g(i,r,c)*
     1       lag(3)/k(i,r,c)))
             b(7,1)=b(7,1)+w(i)*(bk4*(f(i,r,c)+g(i,r,c)*
     1       lag(4)/k(i,r,c)))
 175       continue
 205         format(8f10.4)
 206         format('  ')
C     PROGRAM INVMAT
C
C     A(N,NF) -- CHANGE BOTH THE DECLARATION I.E REAL A(N,NF), AS WELL
C     AS THE DECLARED VALUES OF N & NF. ALSO CHANGE THE NF VALUE IN THE
C     FORMAT STATEMENT AT THE END. I.E (NFF6.2)
C
C
      DO 4 ROW=1,IN
     DO 3 COL=(IN+1),NF
       A(ROW,COL)=0.0
       IF((ROW+IN).EQ.COL) A(ROW,COL)=1.0
3    CONTINUE
4     CONTINUE
      DO 50 D=1,IN
     TEMP=A(D,D)
     DO 5 COL=D,NF
       A(D,COL)=A(D,COL)/TEMP
5    CONTINUE
     DO 15 ROW=D+1,IN
       TEMP=A(ROW,D)
       DO 10 COL=D,NF
         A(ROW,COL)=A(ROW,COL)-TEMP*A(D,COL)
10     CONTINUE
15   CONTINUE
50    CONTINUE
      DO 100 D=IN,2,-1
     DO 80 ROW=D-1,1,-1
       TEMP=A(ROW,D)
       DO 70 COL=D,NF
         A(ROW,COL)=A(ROW,COL)-TEMP*A(D,COL)
70     CONTINUE
80   CONTINUE
100   CONTINUE
      DO 120 ROW=1,IN
      DO 125 COL=IN+1,NF
      X(ROW,1)=X(ROW,1)+A(ROW,COL)*B(COL-IN,1)
125    CONTINUE
120    CONTINUE
```

```
130   FORMAT(' ',9F15.5)
C
C FORCED AGREEMENT WITH TABULAR DATA AT k=0.0
             if(r.eq.1.and.c.eq.1.and.k(1,1,1).eq.0.0) x(1,1)=F(1,r,c)
             if(r.eq.2.and.c.eq.1.and.k(1,2,1).eq.0.0) x(1,1)=F(1,r,c)
c            if(r.eq.1.and.c.eq.2.and.k(1,1,2).eq.0.0) x(1,1)=F(1,r,c)
c            if(r.eq.2.and.c.eq.2.and.k(1,2,2).eq.0.0) x(1,1)=F(1,r,c)
C CALCULATION OF APPROXIMATE FUNCTION
C
          do 300 i=1,m
             p=dcmplx(0.0,k(i,r,c))
             ckquad=x(1,1)+x(2,1)*p+x(3,1)*p**2
             cklag=x(4,1)*p/(p+lag(1))+x(5,1)*p/(p+lag(2))+x(6,1)*
     1       p/(p+lag(3))+x(7,1)*p/(p+lag(4))
             ckbar(i,r,c)=ckquad+cklag
c error calculation
             mn(i)=F(i,r,c)**2+G(i,r,c)**2
             if(mn(i).lt.1.0) mn(i)=1.0
c square error (complete) #1 #2
             err=err+(((dreal(ckbar(i,r,c))-f(i,r,c))**2+
     1       (dimag(ckbar(i,r,c))-g(i,r,c))**2)/mn(i))/m
c % square error (separate) #3
c             err=err+((dreal(ckbar(i,r,c))-f(i,r,c))**2/f(i,r,c)**2+
c     1       (dimag(ckbar(i,r,c))-g(i,r,c))**2/g(i,r,c)**2)/m
c % error (separate) #4
c             err=err+(abs((dreal(ckbar(i,r,c))-f(i,r,c))/f(i,r,c))+
c     1       abs((dimag(ckbar(i,r,c))-g(i,r,c))/g(i,r,c)))/m
c
 300      continue
 301         FN=FN+err
c            print 305, x(1,1),x(2,1),x(3,1),x(4,1),x(5,1),x(6,1),x(7,1)
 305         format(6X,f12.6,',',f10.5,',',f10.5,',',f10.5,',',
     1       f10.5,',',2f10.5)
c             print 634, r,c,lag(1),lag(2),lag(3),lag(4),err,FN
             print 634, count
 500      continue
 600      if(count.lt.icount-2) goto 700
          write (1,604)
          write (1,602) r,c
          write (1,604)
          write (1,604)
          write (1,603)
          write (1,604)
          write (1,605) x(1,1),x(2,1),x(3,1),x(4,1)
          write (1,604)
          write (1,601)
          write (1,604)
          write (1,605) x(5,1),x(6,1),x(7,1)
          write (1,604)
          write (1,604)
          write (1,606)
          write (1,604)
          write (1,635) lag(1),lag(2),lag(3),lag(4),err,FN
          write (1,604)
```

```
          write (1,604)
          write (1,610)
          write (1,611)
          write (1,604)
          write (1,604)
C
C CREATE FILE 'LAPLACE.DAT' CONTAINS b1, b2, ..., A0, A1, ...
          write (10,605) x(1,1),x(2,1),x(3,1),x(4,1),x(5,1),
     1    x(6,1),x(7,1)
          if(r.eq.2.and.c.eq.2) goto 584
          goto 585
 584      write (10,635) lag(1),lag(2),lag(3),lag(4),FN
C
 602      format(6X,' APPROXIMANT FUNCTION        N (',i2,',',i2,' )')
 603      format(5x,'       A0              A1              A2',
     1    '       A3')
 601      format(5x,'       A4              A5              A6')
 604      format('  ')
 605      format(5x,8f16.6)
 634      format(2i5,7f12.9)
 635      format(7x,7f11.8)
 606      format(7x,'    lag1       lag2       lag3       lag4',
     1    '       error')
 610      format(6X,'      k            EXACT                ',
     1    'APPROXIMATE')
 611      format(6X,'               REAL       IMAG          REAL  ',
     1    '    IMAG')
 585      do 650 prt=1,m
          write(1,305) k(prt,r,c),f(prt,r,c),g(prt,r,c),ckbar(prt,r,c)
 650      continue
 700      continue
 750      continue
          RETURN
 800      END
```

INPUT FILE : N11.DAT

0.00	0.00	0.00
0.025	0.17	0.56
0.050	0.39	0.87
0.0875	0.66	1.17
0.100	0.76	1.23
0.125	0.89	1.39
0.150	1.01	1.49
0.175	1.12	1.61
0.200	1.17	1.69
0.25	1.25	1.93
0.3	1.23	2.12
0.35	1.19	2.39
0.4	1.15	2.64
0.45	1.07	2.93
0.5	0.98	3.25

DATA TABLE

Cl(h)

k Real Imaginary

INPUT FILE : N12.DAT

0.000	14.70	-0.00
0.025	12.20	-4.05
0.050	9.50	-4.15
0.0875	7.63	-3.70
0.100	7.22	-3.47
0.125	6.61	-3.18
0.150	6.09	-2.94
0.175	5.72	-2.59
0.200	5.41	-2.29
0.25	4.92	-1.79
0.3	4.63	-1.30
0.35	4.43	-0.83
0.4	4.35	-0.41
0.45	4.35	-0.03
0.5	4.40	0.28

DATA TABLE

Cl(alpha)

k Real Imaginary

INPUT FILE : N21.DAT

0.000	0.000	0.000
0.025	0.001	-0.035
0.050	0.015	-0.065
0.0875	0.025	-0.125
0.100	0.030	-0.145
0.125	0.02	-0.198
0.150	0.015	-0.23
0.175	0.010	-0.280
0.200	-0.005	-0.325
0.25	-0.035	-0.397
0.3	-0.060	-0.403
0.35	-0.045	-0.425
0.4	-0.005	-0.448
0.45	0.065	-0.490
0.5	0.14	-0.55

DATA TABLE

Cm(h)

k Real Imaginary

INPUT FILE : N22.DAT

0.000	-0.445	0.000
0.025	-0.450	-0.027
0.050	-0.455	-0.122
0.0875	-0.495	-0.311
0.100	-0.520	-0.348
0.125	-0.581	-0.422
0.150	-0.672	-0.508
0.175	-0.732	-0.548
0.200	-0.810	-0.582
0.25	-0.905	-0.610
0.3	-0.898	-0.550
0.35	-0.868	-0.575
0.4	-0.824	-0.648
0.45	-0.790	-0.750
0.5	-0.770	-0.903

DATA TABLE

Cm(alpha)

k Real Imaginary

OUTPUT FILE : DATA.OUT

SEQUENTIAL SIMPLEX : PROBLEM MINIMIZATION

451 TRIALS USED

ESTIMATES

PARAMETER	NEXT-TO-BEST	BEST
1	.2789266	.2789266
2	.0436573	.0436573
3	.2789914	.2789914
4	.2858671	.2858671

FUNCTION VALUES

NEXT-TO-BEST	BEST
.001414830	.001396017

OUTPUT FILE : LAG.OUT

APPROXIMANT FUNCTION N (1, 1)

A0	A1	A2	A3
.000000	7.351972	-.265177	-8561.672085

A4	A5	A6
.593074	8707.625432	-146.231038

lag1	lag2	lag3	lag4	error	
.27892661	.04365735	.27899139	.28586712	.00020817	.00020817

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	.00000,	.00000,	.00000,	.00000,
.025000,	.17000,	.56000,	.16792,	.54877,
.050000,	.39000,	.87000,	.41847,	.86344,
.087500,	.66000,	1.17000,	.69680,	1.16498,
.100000,	.76000,	1.23000,	.77287,	1.24726,

.125000,	.89000,	1.39000,	.90794,	1.39206,
.150000,	1.01000,	1.49000,	1.02060,	1.51454,
.175000,	1.12000,	1.61000,	1.10957,	1.62079,
.200000,	1.17000,	1.69000,	1.17459,	1.71761,
.250000,	1.25000,	1.93000,	1.23994,	1.90689,
.300000,	1.23000,	2.12000,	1.24034,	2.11637,
.350000,	1.19000,	2.39000,	1.20249,	2.35917,
.400000,	1.15000,	2.64000,	1.14690,	2.63547,
.450000,	1.07000,	2.93000,	1.08659,	2.94005,
.500000,	.98000,	3.25000,	1.02883,	3.26655,

APPROXIMANT FUNCTION N (1, 2)

A0	A1	A2	A3
14.235672	.957753	1.258183	404976.770760

A4	A5	A6
-8.145772	-408964.029690	3987.367830

lag1	lag2	lag3	lag4	error	
.27892661	.04365735	.27899139	.28586712	.00020294	.00041111

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	14.70000,	.00000,	14.23567,	.00000,
.025000,	12.20000,	-4.05000,	12.21288,	-3.64541,
.050000,	9.50000,	-4.15000,	9.56427,	-4.30539,
.087500,	7.63000,	-3.70000,	7.53586,	-3.72869,
.100000,	7.22000,	-3.47000,	7.15123,	-3.52318,
.125000,	6.61000,	-3.18000,	6.57096,	-3.17275,
.150000,	6.09000,	-2.94000,	6.12655,	-2.88115,
.175000,	5.72000,	-2.59000,	5.75592,	-2.61606,
.200000,	5.41000,	-2.29000,	5.43806,	-2.35648,
.250000,	4.92000,	-1.79000,	4.94237,	-1.82723,
.300000,	4.63000,	-1.30000,	4.62099,	-1.29981,
.350000,	4.43000,	-.83000,	4.44320,	-.81140,
.400000,	4.35000,	-.41000,	4.36678,	-.38607,
.450000,	4.35000,	-.03000,	4.35258,	-.03010,
.500000,	4.40000,	.28000,	4.37100,	.26139,

APPROXIMANT FUNCTION N (2, 1)

A0	A1	A2	A3
.000000	-1.905869	-.293633	158360.047503

A4	A5	A6
-.059842	-159903.225805	1543.931881

lag1	lag2	lag3	lag4	error	
.27892661	.04365735	.27899139	.28586712	.00019833	.00060945

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	.00000,	.00000,	.00000,	.00000,
.025000,	.00100,	-.03500,	-.00729,	-.03935,
.050000,	.01500,	-.06500,	-.00727,	-.06574,
.087500,	.02500,	-.12500,	.01171,	-.11930,
.100000,	.03000,	-.14500,	.01725,	-.14295,
.125000,	.02000,	-.19800,	.02145,	-.19500,
.150000,	.01500,	-.23000,	.01512,	-.24731,
.175000,	.01000,	-.28000,	.00030,	-.29389,
.200000,	-.00500,	-.32500,	-.01874,	-.33146,
.250000,	-.03500,	-.39700,	-.05219,	-.37922,
.300000,	-.06000,	-.40300,	-.06191,	-.40363,
.350000,	-.04500,	-.42500,	-.04310,	-.42338,
.400000,	-.00500,	-.44800,	-.00139,	-.45102,
.450000,	.06500,	-.49000,	.05504,	-.49191,
.500000,	.14000,	-.55000,	.11927,	-.54679,

APPROXIMANT FUNCTION N (2, 2)

A0	A1	A2	A3
-.516839	-4.394286	2.744002	217699.240354

A4	A5	A6
.165112	-219881.401614	2184.097622

lag1	lag2	lag3	lag4	error	
.27892661	.04365735	.27899139	.28586712	.00078657	.00139602

k	EXACT REAL	EXACT IMAG	APPROXIMATE REAL	APPROXIMATE IMAG
.000000,	-.44500,	.00000,	-.51684,	.00000,
.025000,	-.45000,	-.02700,	-.48373,	-.04150,
.050000,	-.45500,	-.12200,	-.45477,	-.14213,
.087500,	-.49500,	-.31100,	-.48600,	-.31323,

```
.100000,    -.52000,    -.34800,    -.51147,    -.36418,
.125000,    -.58100,    -.42200,    -.57665,    -.45028,
.150000,    -.67200,    -.50800,    -.65200,    -.51226,
.175000,    -.73200,    -.54800,    -.72737,    -.55013,
.200000,    -.81000,    -.58200,    -.79435,    -.56754,
.250000,    -.90500,    -.61000,    -.88350,    -.56578,
.300000,    -.89800,    -.55000,    -.90772,    -.55768,
.350000,    -.86800,    -.57500,    -.88429,    -.57878,
.400000,    -.82400,    -.64800,    -.83843,    -.64393,
.450000,    -.79000,    -.75000,    -.79150,    -.75384,
.500000,    -.77000,    -.90300,    -.75798,    -.90251,
```

OUTPUT FILE : LAPLACE.DAT

```
              .000000          7.351972          -.265177     -8561.672085
  .593074      8707.625432          -146.231038

            14.235672           .957753          1.258183    404976.770760
-8.145772   -408964.029690          3987.367830

              .000000         -1.905869          -.293633    158360.047503
 -.059842   -159903.225805          1543.931881

             -.516839         -4.394286          2.744002    217699.240354
  .165112   -219881.401614          2184.097622

      .27892661   .04365735   .27899139   .28586712   .00139602
```

```
$storage:2
$floatcalls
$debug
C-----------------------------------------------------------------------
C  LAPLACE EIGENVALUE SOLUTION
C  FLUTTER CALCULATION (INCOMPRESSIBLE REGIME)
C-----------------------------------------------------------------------
C
      IMPLICIT REAL*8 (A-Z)
      INTEGER ib,n,ia,ijob,iz,ier,infer,i,j
      COMPLEX*16 eval(12),A(12,12),B(12,12),EIGA(12),EIGB(12),Z(12,12)
      COMPLEX*16 WK(21,42)
      REAL*8 u,vel,atemp1,atemp2,atemp3,atemp4
      REAL*8 finv(7,2,2),f(8,2,2),m(2,2),c(2,2),k(2,2)
      REAL*8 a0(2,2),a1(2,2),a2(2,2),a3(2,2),a4(2,2),a5(2,2),a6(2,2)
      real*8 n1(2,7)
      INTEGER y
C
C Read in constants.
      OPEN (UNIT=1,FILE='CONST6.DAT',STATUS='OLD')
      READ (1,*) ia,ib,iz,n,ijob
C                12 12 12 12 2
      CLOSE (UNIT=1)
C
C open required files
      OPEN (UNIT=4,FILE='EIGEN.OUT',STATUS='NEW')
C
C required constants
             u=50.0
             wrat=0.2
             r=0.5
             ah=-0.5
C
             xtor=0.25
             semic=1.0
             wtor=1.0
             wben=wrat*wtor
C
C required constants
C
      OPEN (UNIT=10,FILE='LAPLACE.DAT',STATUS='OLD')
          do 25 i=1,2
      READ (10,*) n1(i,1),n1(i,2),n1(i,3),n1(i,4),n1(i,5),
     1            n1(i,6),n1(i,7)
 25       continue
      READ (10,*) b1,b2,b3,b4,err
      CLOSE (UNIT=10)
C
C titles and initial parameter listing to file
             write(4,32)
             write(4,121)
             write(4,121)
             write(4,33)
             write(4,121)
```

```
            write(4,34) u,r,ah,wrat
            write(4,121)
            write(4,121)
 32         format(6x,' LAPLACE METHOD   INCOMPRESSIBLE (THEODORSEN)')
 33         format(6x,'  u          r(alp)     ah     wben/wtor')
 34         format(6x,f7.1,4f10.1)
 121        format('   ')
C
C FOUR LAG TERMS
C
C PADE APPROXIMANT  THEODORSEN FUNCTION    C(k)  AND  C(k)*ik
C
C term Q(1,1): C(k)*ik
C
            a0(1,1)= n1(2,1)
            a1(1,1)= n1(2,2)
            a2(1,1)= n1(2,3)
            a3(1,1)= n1(2,4)
            a4(1,1)= n1(2,5)
            a5(1,1)= n1(2,6)
            a6(1,1)= n1(2,7)
C
C term Q(1,2): C(k)*ik*(0.5-ah)+C(k)
C
            a0(1,2)= n1(2,1)*(0.5-ah)+n1(1,1)
            a1(1,2)= n1(2,2)*(0.5-ah)+n1(1,2)
            a2(1,2)= n1(2,3)*(0.5-ah)+n1(1,3)
            a3(1,2)= n1(2,4)*(0.5-ah)+n1(1,4)
            a4(1,2)= n1(2,5)*(0.5-ah)+n1(1,5)
            a5(1,2)= n1(2,6)*(0.5-ah)+n1(1,6)
            a6(1,2)= n1(2,7)*(0.5-ah)+n1(1,7)
C
C term Q(2,1): C(k)*ik*(-(ah+0.5))
C
            a0(2,1)= n1(2,1)*(-0.5-ah)
            a1(2,1)= n1(2,2)*(-0.5-ah)
            a2(2,1)= n1(2,3)*(-0.5-ah)
            a3(2,1)= n1(2,4)*(-0.5-ah)
            a4(2,1)= n1(2,5)*(-0.5-ah)
            a5(2,1)= n1(2,6)*(-0.5-ah)
            a6(2,1)= n1(2,7)*(-0.5-ah)
C
C term Q(2,2): C(k)*ik*(-(ah+0.5)*(0.5-ah))-(ah+0.5)*C(k)
C
            a0(2,2)= n1(2,1)*(-0.5-ah)*(0.5-ah)-(ah+0.5)*n1(1,1)
            a1(2,2)= n1(2,2)*(-0.5-ah)*(0.5-ah)-(ah+0.5)*n1(1,2)
            a2(2,2)= n1(2,3)*(-0.5-ah)*(0.5-ah)-(ah+0.5)*n1(1,3)
            a3(2,2)= n1(2,4)*(-0.5-ah)*(0.5-ah)-(ah+0.5)*n1(1,4)
            a4(2,2)= n1(2,5)*(-0.5-ah)*(0.5-ah)-(ah+0.5)*n1(1,5)
            a5(2,2)= n1(2,6)*(-0.5-ah)*(0.5-ah)-(ah+0.5)*n1(1,6)
            a6(2,2)= n1(2,7)*(-0.5-ah)*(0.5-ah)-(ah+0.5)*n1(1,7)
C
C
C
```

```
                 b1234=b1+b2+b3+b4
                 b123=b1+b2+b3
                 b124=b1+b2+b4
                 b134=b1+b3+b4
                 b234=b2+b3+b4
                 bt123=b1*b2*b3
                 bt124=b1*b2*b4
                 bt134=b1*b3*b4
                 bt234=b2*b3*b4
                 bq1234=b1*b2*b3*b4
                 bt1234=b1*b2*b3+b1*b2*b4+b1*b3*b4+b2*b3*b4
                 bd1234=b1*b2+b1*b3+b2*b3+b1*b4+b2*b4+b3*b4
                 bd123=b1*b2+b1*b3+b2*b3
                 bd124=b1*b2+b1*b4+b2*b4
                 bd134=b1*b3+b1*b4+b3*b4
                 bd234=b2*b3+b3*b4+b2*b4
c
c iterate velocity
                 vel=0.0
                 step =0.5
 60              vel=vel+step
                 if(vel/(semic*wtor).gt.5.0) goto 120
                 wf=vel/semic
                 xvar=2*wf**2/u
c
c formation of matrices :  [M], [C], [K], [D] and [E]
                 m(1,1)=1+1/u
                 m(1,2)=xtor-ah/u
                 m(2,1)=xtor-ah/u
                 m(2,2)=r**2+(0.125+ah**2)/u
                 c(1,1)=0.0
                 c(1,2)=wf/u
                 c(2,1)=0.0
                 c(2,2)=(0.5-ah)*wf/u
                 k(1,1)=wrat**2*wtor**2
                 k(1,2)=0.0
                 k(2,1)=0.0
                 k(2,2)=r**2*wtor**2
c
c formation of polynomial coefficients :  [F7], [F6], [F5], [F4]
c                                         [F3], [F2], [F1]
c
c   [F7]s6+[F6]s5+ ... +[F2]s+[F1] = 0
              do 200 i=1,2
              do 100 j=1,2
c                calculation of [F7]
                 f(7,i,j)=m(i,j)+xvar*(1/wf)**2*a2(i,j)
c                calculation of [F6]
                 f(6,i,j)=c(i,j)+wf*b1234*m(i,j)+xvar*(1/wf)*(a1(i,j)+
     1           b1234*a2(i,j))
c                calculation of [F5]
                 f(5,i,j)=k(i,j)+wf*b1234*c(i,j)+wf**2*bd1234*m(i,j)+
     1           xvar*(a0(i,j)+b1234*a1(i,j)+bd1234*a2(i,j)+a3(i,j)+
```

```
     1        a4(i,j)+a5(i,j)+a6(i,j))
C             calculation of [F4]
              f(4,i,j)=wf*b1234*k(i,j)+wf**2*bd1234*c(i,j)+wf**3*bt1234*
     1        m(i,j)+xvar*wf*(b1234*a0(i,j)+bd1234*a1(i,j)+bt1234*
     1        a2(i,j)+b234*a3(i,j)+b134*a4(i,j)+b124*a5(i,j)+
     1        b123*a6(i,j))
C             calculation of [F3]
              f(3,i,j)=wf**2*bd1234*k(i,j)+wf**3*bt1234*c(i,j)+wf**4*
     1        bq1234*m(i,j)+xvar*wf**2*(bd1234*a0(i,j)+bt1234*a1(i,j)+
     1        bq1234*a2(i,j)+bd234*a3(i,j)+bd134*a4(i,j)+bd124*
     1        a5(i,j)+bd123*a6(i,j))
C             calculation of [F2]
              f(2,i,j)=wf**3*bt1234*k(i,j)+wf**4*bq1234*c(i,j)+
     1        xvar*wf**3*(bt1234*a0(i,j)+bq1234*a1(i,j)+bt234*a3(i,j)+
     1        bt134*a4(i,j)+bt124*a5(i,j)+bt123*a6(i,j))
C             calculation of [F1]
              f(1,i,j)=wf**4*bq1234*k(i,j)+xvar*wf**4*bq1234*a0(i,j)
 100      continue
 200      continue
C
C             calculation of inverse  ( inv[F8] )
              f7det=f(7,1,1)*f(7,2,2)-f(7,1,2)*f(7,2,1)
              if(f7det.eq.0.0) goto 210
              goto 220
 210          write(4,215)
 215          format(' DETERMINANT EQUALS ZERO - INVESTIGATE INPUT ',
     1        'CONSTANT ... ah ')
              goto 120
 220          finv(7,1,1)=f(7,2,2)/f7det
              f(8,1,1)=finv(7,1,1)
              finv(7,2,2)=f(7,1,1)/f7det
              f(8,2,2)=finv(7,2,2)
              finv(7,1,2)=-f(7,1,2)/f7det
              f(8,1,2)=finv(7,1,2)
              finv(7,2,1)=-f(7,2,1)/f7det
              f(8,2,1)=finv(7,2,1)
C
C formation of matrices [A] and [B] to
C solve eigenvalue problem given by [A]{x} = lambda [B]{x}
          do 400 i=1,n
          do 300 j=1,n
              a(i,j)=dcmplx(0.0,0.0)
              if(i.eq.j+2) a(i,j)=dcmplx(1.0,0.0)
              b(i,j)=dcmplx(0.0,0.0)
              if(i.eq.j) b(i,j)=dcmplx(1.0,0.0)
 300      continue
 400      continue
          do 500 i=1,n/2
              y=n/2+1-i
              atemp1=-(finv(7,1,1)*f(y,1,1)+finv(7,1,2)*f(y,2,1))
              a(1,2*i-1)=dcmplx(atemp1,0.0)
              atemp2=-(finv(7,1,1)*f(y,1,2)+finv(7,1,2)*f(y,2,2))
              a(1,2*i)=dcmplx(atemp2,0.0)
              atemp3=-(finv(7,2,1)*f(y,1,1)+finv(7,2,2)*f(y,2,1))
```

```
              a(2,2*i-1)=dcmplx(atemp3,0.0)
              atemp4=-(finv(7,2,1)*f(y,1,2)+finv(7,2,2)*f(y,2,2))
              a(2,2*i)=dcmplx(atemp4,0.0)
 500       continue
C
 748          format(6x,2f24.6)
 750          format(6x,28e8.1)
 749          format(6x,i3)
 760          format('  ')
C
C subrountine for eigenvalue calculation
C abar : real part of eval( )
C bbar : imaginary part of eval( )
      CALL EIGZC(A,IA,B,IB,N,IJOB,EIGA,EIGB,Z,IZ,WK,INFER,IER)
          do 5 I=1,n
 5            EVAL(I)=EIGA(I)/EIGB(I)
C printout of velocity and associated eigenvalues
              write(4,23) vel/(semic*wtor)
          do 6 i=1,n
              write(4,22) eval(i)
 6        continue
               write(4,21) wk(1,1)
               write(4,121)
 21           format(6x,'  perfomance index is', 2f15.5)
 23           format(6x,' ND VELOCITY ',f6.3)
 22           format(6x,8f17.5)
 28           format(6x,f17.5,',',f17.5,',',f17.5)
C
C
C changing velocity step size
c              if(vel/(semic*wtor).ge.3.5) step=0.1
c              if(vel/(semic*wtor).ge.5.0) step=0.1
c              if(vel/(semic*wtor).ge.6.0) step=0.5
c              if(vel/(semic*wtor).ge.6.0) goto 120
 700           goto 60
 120  CLOSE (UNIT=4)
 800  END
```

INPUT FILE : LAPLACE.DAT

```
    .99828226        -.00003562         .00000474        -.04120375
-.16289333       -.22908745        -.06459371

    -.00001355        .50037475        -.00003629         .00061076
.01380489        .05848218          .04963449

    .015440852   .084286467   .255950673   .765113621   .000011782
```

INPUT FILE : CONST6.DAT

```
12   12   12   12   2
```

OUTPUT FILE : EIGEN.OUT

LAPLACE METHOD   INCOMPRESSIBLE (THEODORSEN)

u	r(alp)	ah	wben/wtor
50.0	.5	-.5	.2

ND VELOCITY .500

-.01437	1.14595	
-.01437	-1.14595	
-.00637	-.19921	
-.00637	.19921	
-.38147	.00000	
-.38256	.00000	
-.12656	.00000	
-.12798	.00000	
-.04200	.00000	
-.04214	.00000	
-.00772	.00000	
-.00772	.00000	
perfomance index is	1.25299	.00000

ND VELOCITY 1.000

-.02966	1.13486	
-.02966	-1.13486	
-.76212	.00000	
-.76511	.00000	
-.01482	.20360	
-.01482	-.20360	
-.24879	.00000	
-.25595	.00000	
-.08316	.00000	
-.08429	.00000	
-.01544	.00000	
-.01543	.00000	
perfomance index is	1.50717	.00000

ND VELOCITY 1.500

-.04649	-1.11520	
-.04649	1.11520	
-1.14241	.00000	
-1.14767	.00000	
-.02480	.20984	
-.02480	-.20984	
-.36693	.00000	
-.38393	.00000	
-.12267	.00000	
-.12643	.00000	
-.02316	.00000	
-.02313	.00000	
perfomance index is	3.53341	.00000

ND VELOCITY 2.000

-.06529	1.08538	
-1.52250	.00000	
-1.53023	.00000	
-.06529	-1.08538	
-.51190	.00000	
-.48021	.00000	
-.03664	.21879	
-.03664	-.21879	
-.15957	.00000	
-.16857	.00000	
-.03088	.00000	
-.03080	.00000	
perfomance index is	17.25293	.00000

ND VELOCITY 2.500

-1.90251	.00000	
-1.91278	.00000	
-.08644	-1.04289	
-.08644	1.04289	
-.63988	.00000	
-.58718	.00000	
-.05141	-.23195	
-.05141	.23195	
-.19235	.00000	
-.21072	.00000	
-.03860	.00000	
-.03843	.00000	
perfomance index is	1.84973	.00000

ND VELOCITY 3.000

-2.28251	.00000	
-2.29534	.00000	
-.11024	-.98348	
-.11024	.98348	
-.76785	.00000	
-.68609	.00000	
-.07108	.25240	
-.07108	-.25240	
-.25286	.00000	
-.21816	.00000	
-.04632	.00000	
-.04600	.00000	
perfomance index is	3.86263	.00000

ND VELOCITY 3.500

-2.66251	.00000
-2.67790	.00000
-.13678	.89843
-.13678	-.89843
-.89583	.00000
-.77455	.00000
-.09940	.28811

-.09940	-.28811	
-.29500	.00000	
-.23173	.00000	
-.05404	.00000	
-.05348	.00000	
perfomance index is	2.67030	.00000

ND VELOCITY 4.000

-3.04253	.00000	
-3.06045	.00000	
-.16495	-.76095	
-.16495	.76095	
-1.02380	.00000	
-.84867	.00000	
-.14323	-.36783	
-.14323	.36783	
-.33715	.00000	
-.22548	.00000	
-.06176	.00000	
-.06081	.00000	
perfomance index is	3.78368	.00000

ND VELOCITY 4.500

-3.42257	.00000	
-3.44301	.00000	
-1.15178	.00000	
-.01016	-.54976	
-.39043	-.51943	
-.90109	.00000	
-.01016	.54976	
-.39043	.51943	
-.37929	.00000	
-.20035	.00000	
-.06948	.00000	
-.06791	.00000	
perfomance index is	3.19514	.00000

ND VELOCITY 5.000

-3.80262	.00000	
-3.82557	.00000	
.11382	-.52557	
.11382	.52557	
-1.27975	.00000	
-.62744	-.48581	
-.62744	.48581	
-.91518	.00000	
-.42143	.00000	
-.17266	.00000	
-.07461	.00000	
-.07720	.00000	
perfomance index is	2.91782	.00000

```
$storage:2
$floatcalls
$debug
C-----------------------------------------------------------------------
C  LAPLACE EIGENVALUE SOLUTION
C  FLUTTER CALCULATION (TRANSONIC REGIME)
C-----------------------------------------------------------------------
      IMPLICIT REAL*8 (A-Z)
      INTEGER ib,n,ia,ijob,iz,ier,infer,i,j
      COMPLEX*16 eval(12),A(12,12),B(12,12),EIGA(12),EIGB(12),Z(12,12)
      COMPLEX*16 WK(21,42)
      REAL*8 u,vel,atemp1,atemp2,atemp3,atemp4
      REAL*8 finv(7,2,2),f(8,2,2),m(2,2),c(2,2),k(2,2),d(2,2),e(2,2)
      REAL*8 a0(2,2),a1(2,2),a2(2,2),a3(2,2),a4(2,2),a5(2,2),a6(2,2)
      INTEGER y
C
C Read in constants.
      OPEN (UNIT=1,FILE='CONST6.DAT',STATUS='OLD')
      READ (1,*) ia,ib,iz,n,ijob
C                12 12 12 12 2
      CLOSE (UNIT=1)
C
C open required files
      OPEN (UNIT=4,FILE='EIGEN.OUT',STATUS='NEW')
C
C required constants
             u=50.0
             r=0.5
             ah=-0.5
             wrat=0.2
C
             xtor=0.25
             semic=1.0
             wtor=1.0
             wben=wrat*wtor
             pi=4.0*atan(1.0)
C
C required constants (approximation)
      OPEN (UNIT=10,FILE='LAPLACE.DAT',STATUS='OLD')
          do 50 i=1,2
          do 25 j=1,2
      READ (10,*) a0(i,j),a1(i,j),a2(i,j),a3(i,j),a4(i,j),
     1            a5(i,j),a6(i,j)
 25       continue
 50       continue
      READ (10,*) b1,b2,b3,b4,err
      CLOSE (UNIT=10)
C
C titles and initial parameter listing to file
             write(4,32)
             write(4,121)
             write(4,121)
             write(4,33)
             write(4,121)
```

```
              write(4,34) u,r,ah,wrat
              write(4,121)
              write(4,121)
 32           format(6X,'  LAPLACE METHOD       TRANSONIC  M=0.85')
 33           format(6X,'   u            r(alp)      ah       wben/wtor')
 34           format(6X,f7.1,4f10.1)
 121          format('    ')
C
C
C term Q(1,1): N(1,1) = Clh/2
C
              a0(1,1)=    a0(1,1)/2.0
              a1(1,1)=    a1(1,1)/2.0
              a2(1,1)=    a2(1,1)/2.0
              a3(1,1)=    a3(1,1)/2.0
              a4(1,1)=    a4(1,1)/2.0
              a5(1,1)=    a5(1,1)/2.0
              a6(1,1)=    a6(1,1)/2.0
C
C term Q(1,2): N(1,2) = Cltor
C
              a0(1,2)=    a0(1,2)
              a1(1,2)=    a1(1,2)
              a2(1,2)=    a2(1,2)
              a3(1,2)=    a3(1,2)
              a4(1,2)=    a4(1,2)
              a5(1,2)=    a5(1,2)
              a6(1,2)=    a6(1,2)
C
C term Q(2,1): N(2,1) = Cmh*(-1)
C
              a0(2,1)=    a0(2,1)*(-1.0)
              a1(2,1)=    a1(2,1)*(-1.0)
              a2(2,1)=    a2(2,1)*(-1.0)
              a3(2,1)=    a3(2,1)*(-1.0)
              a4(2,1)=    a4(2,1)*(-1.0)
              a5(2,1)=    a5(2,1)*(-1.0)
              a6(2,1)=    a6(2,1)*(-1.0)
C
C term Q(2,2): N(2,2) = Cmtor*(-2)
C
              a0(2,2)=    a0(2,2)*(-2.0)
              a1(2,2)=    a1(2,2)*(-2.0)
              a2(2,2)=    a2(2,2)*(-2.0)
              a3(2,2)=    a3(2,2)*(-2.0)
              a4(2,2)=    a4(2,2)*(-2.0)
              a5(2,2)=    a5(2,2)*(-2.0)
              a6(2,2)=    a6(2,2)*(-2.0)
C
C
C
              b1234=b1+b2+b3+b4
              b123=b1+b2+b3
              b124=b1+b2+b4
```

```
            b134=b1+b3+b4
            b234=b2+b3+b4
            bt123=b1*b2*b3
            bt124=b1*b2*b4
            bt134=b1*b3*b4
            bt234=b2*b3*b4
            bq1234=b1*b2*b3*b4
            bt1234=b1*b2*b3+b1*b2*b4+b1*b3*b4+b2*b3*b4
            bd1234=b1*b2+b1*b3+b2*b3+b1*b4+b2*b4+b3*b4
            bd123=b1*b2+b1*b3+b2*b3
            bd124=b1*b2+b1*b4+b2*b4
            bd134=b1*b3+b1*b4+b3*b4
            bd234=b2*b3+b3*b4+b2*b4
c
c iterate velocity
            vel=0.0
            step =0.5
 60         vel=vel+step
            if(vel/(semic*wtor).gt.4.0) goto 120
            wf=vel/semic
            xvar=(1/pi)*wf**2/u
c
c formation of matrices :  [M], [C], [K], [D] and [E]
            m(1,1)=1.0
            m(1,2)=xtor
            m(2,1)=xtor
            m(2,2)=r**2
            c(1,1)=0.0
            c(1,2)=0.0
            c(2,1)=0.0
            c(2,2)=0.0
            k(1,1)=wben**2
            k(1,2)=0.0
            k(2,1)=0.0
            k(2,2)=r**2*wtor**2
c
c formation of polynomial coefficients :  [F7], [F6], [F5], [F4]
c                                         [F3], [F2], [F1]
c
c   [F7]s6+[F6]s5+ ... +[F2]s+[F1] = 0
         do 200 i=1,2
         do 100 j=1,2
c           calculation of [F7]
            f(7,i,j)=m(i,j)+xvar*(1/wf)**2*a2(i,j)
c           calculation of [F6]
            f(6,i,j)=c(i,j)+wf*b1234*m(i,j)+xvar*(1/wf)*(a1(i,j)+
     1      b1234*a2(i,j))
c           calculation of [F5]
            f(5,i,j)=k(i,j)+wf*b1234*c(i,j)+wf**2*bd1234*m(i,j)+
     1      xvar*(a0(i,j)+b1234*a1(i,j)+bd1234*a2(i,j)+a3(i,j)+
     1      a4(i,j)+a5(i,j)+a6(i,j))
c           calculation of [F4]
            f(4,i,j)=wf*b1234*k(i,j)+wf**2*bd1234*c(i,j)+wf**3*bt1234*
     1      m(i,j)+xvar*wf*(b1234*a0(i,j)+bd1234*a1(i,j)+bt1234*
```

```
     1        a2(i,j)+b234*a3(i,j)+b134*a4(i,j)+b124*a5(i,j)+
     1        b123*a6(i,j))
c             calculation of [F3]
              f(3,i,j)=wf**2*bd1234*k(i,j)+wf**3*bt1234*c(i,j)+wf**4*
     1        bq1234*m(i,j)+xvar*wf**2*(bd1234*a0(i,j)+bt1234*a1(i,j)+
     1        bq1234*a2(i,j)+bd234*a3(i,j)+bd134*a4(i,j)+bd124*
     1        a5(i,j)+bd123*a6(i,j))
c             calculation of [F2]
              f(2,i,j)=wf**3*bt1234*k(i,j)+wf**4*bq1234*c(i,j)+
     1        xvar*wf**3*(bt1234*a0(i,j)+bq1234*a1(i,j)+bt234*a3(i,j)+
     1        bt134*a4(i,j)+bt124*a5(i,j)+bt123*a6(i,j))
c             calculation of [F1]
              f(1,i,j)=wf**4*bq1234*k(i,j)+xvar*wf**4*bq1234*a0(i,j)
 100       continue
 200       continue
c
c             calculation of inverse  ( inv[F8] )
              f7det=f(7,1,1)*f(7,2,2)-f(7,1,2)*f(7,2,1)
              if(f7det.eq.0.0) goto 210
              goto 220
 210          write(4,215)
 215          format(' DETERMINANT EQUALS ZERO - INVESTIGATE INPUT ',
     1        'CONSTANT ... ah ')
              goto 120
 220          finv(7,1,1)=f(7,2,2)/f7det
              f(8,1,1)=finv(7,1,1)
              finv(7,2,2)=f(7,1,1)/f7det
              f(8,2,2)=finv(7,2,2)
              finv(7,1,2)=-f(7,1,2)/f7det
              f(8,1,2)=finv(7,1,2)
              finv(7,2,1)=-f(7,2,1)/f7det
              f(8,2,1)=finv(7,2,1)
c
c formation of matrices [A] and [B] to
c solve eigenvalue problem given by [A]{x} = lambda [B]{x}
           do 400 i=1,n
           do 300 j=1,n
              a(i,j)=dcmplx(0.0,0.0)
              if(i.eq.j+2) a(i,j)=dcmplx(1.0,0.0)
              b(i,j)=dcmplx(0.0,0.0)
              if(i.eq.j) b(i,j)=dcmplx(1.0,0.0)
 300       continue
 400       continue
           do 500 i=1,n/2
              y=n/2+1-i
              atemp1=-(finv(7,1,1)*f(y,1,1)+finv(7,1,2)*f(y,2,1))
              a(1,2*i-1)=dcmplx(atemp1,0.0)
              atemp2=-(finv(7,1,1)*f(y,1,2)+finv(7,1,2)*f(y,2,2))
              a(1,2*i)=dcmplx(atemp2,0.0)
              atemp3=-(finv(7,2,1)*f(y,1,1)+finv(7,2,2)*f(y,2,1))
              a(2,2*i-1)=dcmplx(atemp3,0.0)
              atemp4=-(finv(7,2,1)*f(y,1,2)+finv(7,2,2)*f(y,2,2))
              a(2,2*i)=dcmplx(atemp4,0.0)
 500       continue
```

```
C
C subrountine for eigenvalue calculation
C abar : real part of eval( )
C bbar : imaginary part of eval( )
      CALL EIGZC(A,IA,B,IB,N,IJOB,EIGA,EIGB,Z,IZ,WK,INFER,IER)
          do 5 I=1,n
 5           EVAL(I)=EIGA(I)/EIGB(I)
C
C printout of velocity and associated eigenvalues
             write(4,24)
             write(4,23) vel/(semic*wtor)
          do 6 i=1,n
             write(4,22) eval(i)
 6        continue
             write(4,21) wk(1,1)
 21          format(6x,'  perfomance index is', 2f15.5)
 23          format(6x,' ND VELOCITY ',f6.3)
 22          format(6x,8f17.5)
 24          format(' ')
C
C changing velocity step size
c             if(vel/(semic*wtor).ge.3.0) step=0.1
c             if(vel/(semic*wtor).ge.3.5) step=0.5
c             if(vel/(semic*wtor).ge.4.5) step=0.5
c             if(vel/(semic*wtor).ge.5.0) step=0.5
 700          goto 60
 120  CLOSE (UNIT=4)
 800  END
```

INPUT FILE : LAPLACE.DAT

```
            .000000         7.351972          -.265177    -8561.672085
  .593074    8707.625432     -146.231038

          14.235672          .957753          1.258183    404976.770760
-8.145772  -408964.029690    3987.367830

            .000000        -1.905869          -.293633    158360.047503
 -.059842  -159903.225805    1543.931881

           -.516839        -4.394286          2.744002    217699.240354
  .165112  -219881.401614    2184.097622

       .27892661  .04365735  .27899139  .28586712  .00139602
```

INPUT FILE : CONST6.DAT

```
12  12  12  12  2
```

OUTPUT FILE : EIGEN.OUT

```
          LAPLACE METHOD       TRANSONIC  M=0.85

           u          r(alp)      ah      wben/wtor

           50.0         .5        -.5        .2


         ND VELOCITY   .500
                   -.08315         1.27662
                   -.08315        -1.27662
                   -.00511         -.20185
                   -.00511          .20185
                   -.18737         -.01797
                   -.18737          .01797
                   -.12884         -.03121
                   -.12884          .03121
                   -.11139         -.01459
                   -.11139          .01459
                   -.02172          .00019
                   -.02172         -.00019
          perfomance index is         1.34192          .00000

         ND VELOCITY  1.000
                   -.13682        -1.22225
                   -.13682         1.22225
                   -.01143          .21140
                   -.01143         -.21140
                   -.47016          .00000
                   -.41480          .00000
                   -.24390         -.08059
                   -.24390          .08059
                   -.19773         -.03599
                   -.19773          .03599
                   -.04279          .00145
                   -.04279         -.00145
          perfomance index is         1.91492          .00000

         ND VELOCITY  1.500
                   -.14304        -1.17204
                   -.14304         1.17204
                   -.91862          .00000
                   -.60947          .00000
                   -.01825          .22490
                   -.01825         -.22490
                   -.35435          .13235
                   -.35435         -.13235
                   -.27054          .05787
                   -.27054         -.05787
                   -.06248          .00460
                   -.06248         -.00460
          perfomance index is         3.55296          .00000
```

ND VELOCITY 2.000

-.13090	-1.16269	
-1.42867	.00000	
-.13090	1.16269	
-.81080	.00000	
-.02269	.24257	
-.02269	-.24257	
-.46358	.18321	
-.46358	-.18321	
-.33331	.07634	
-.33331	-.07634	
-.08006	-.01006	
-.08006	.01006	
perfomance index is	3.54698	.00000

ND VELOCITY 2.500

-1.93877	.00000	
-.13300	-1.18652	
-.13300	1.18652	
-1.01285	.00000	
-.02228	.26434	
-.02228	-.26434	
-.57278	.23294	
-.57278	-.23294	
-.38902	.08819	
-.38902	-.08819	
-.09496	-.01771	
-.09496	.01771	
perfomance index is	4.61556	.00000

ND VELOCITY 3.000

-2.43536	.00000	
-.15539	1.22983	
-.15539	-1.22983	
-1.21510	.00000	
-.68232	-.28189	
-.68232	.28189	
-.01448	-.28849	
-.01448	.28849	
-.44104	.09158	
-.44104	-.09158	
-.10696	.02704	
-.10696	-.02704	
perfomance index is	3.86013	.00000

ND VELOCITY 3.500

-2.92058	.00000
-.19525	1.28757
-1.41743	.00000
-.19525	-1.28757
-.79221	-.33037
-.79221	.33037
.00221	.31152

```
           .00221          -.31152
          -.49236           .08603
          -.49236          -.08603
          -.11637           .03744
          -.11637          -.03744
 perfomance index is        5.60378          .00000

ND VELOCITY  4.000
         -3.39765           .00000
         -1.61980           .00000
          -.24794         -1.35846
          -.24794          1.35846
          -.90240           .37857
          -.90240          -.37857
           .02714          -.32936
           .02714           .32936
          -.54475          -.07102
          -.54475           .07102
          -.12386           .04847
          -.12386          -.04847
 perfomance index is        3.19862          .00000
```

```
C ----------------------------------------------------------------------
C     LAPLACE EIGENVALUE SOLUTION
C     IMPLEMENTATION OF ACTIVE CONTROL
C     FLUTTER CALCULATION (INCOMPRESSIBLE REGIME)
C ----------------------------------------------------------------------
C
C     OPERATION
C     REQUIRE: 1) INPUT OF  ICOUNT, N, REQMIN, START(I),STEP(I)
C              2) USER SPECIFIED FUNCTION SUBPROGRAM (DOUBLE PRECISION
C                 FUNCTION FN(X) - MINIMIZATION FUNCTION)
C
C
      DOUBLE PRECISION START(20),STEP(20),XMIN(20),
     1XSEC(20),YNEWLO,YSEC,REQMIN
C
      OPEN(UNIT=6,FILE='DATA.OUT',STATUS='NEW')
C
C
C
C       ***** INPUT REQUIRED ********************************
C       *                                                   *
C       *   ICOUNT,REQMIN,START(N),STEP(N)                  *
C       *                                                   *
C       *   velndnc : no control non-dimensional velocity   *
C       *   step : velocity step size                       *
C       *   u : airfoil-air mass ratio                      *
C       *   wrat : frequency ratio (wben/wtor)              *
C       *   r : non-dimensional distance                    *
C       *   ah : non-dimensional distance                   *
C       *                                                   *
C       *   OUTPUT FILE : OPTIMAL.OUT                       *
C       *                                                   *
C       *****************************************************
C
C
C
C LET N=4 AT ALL TIMES, SIMPLY SET START(I)=0.0 AND STEP(I)=0.0
C
C
      ICOUNT=8
      N=4
      REQMIN=0.000000001
      START(1)=0.0
      START(2)=-0.3
      START(3)=4.0
      START(4)=3.6
      STEP(1)=0.1
      STEP(2)=-0.1
      STEP(3)=0.1
      STEP(4)=0.1
C
C
      DO 60 I=1,N
      XMIN(I)=0.D0
```

```
      XSEC(I)=0.D0
 60   CONTINUE
      YNEWLO=0.D0
      YSEC=0.D0
C
C              CALL NELDER-MEAD SUBROUTINE
C
      CALL NELMIN(N,START,XMIN,XSEC,YNEWLO,YSEC,
     1REQMIN,STEP,ICOUNT)
C
C              OUTPUT FROM PROGRAM
C
      WRITE(6,64)
      WRITE(6,65) ICOUNT
      WRITE(6,75)
      WRITE(6,77)
      DO 79 I=1,N
 79   WRITE(6,80) I,XSEC(I),XMIN(I)
      WRITE(6,82)
      WRITE(6,83)
      WRITE(6,84) YSEC,YNEWLO
C
 64   FORMAT(1H //,1H ,42H SEQUENTIAL SIMPLEX : PROBLEM MINIMIZATION)
 65   FORMAT(1H //,1H ,I5,12H TRIALS USED/)
 75   FORMAT(1H ,21X,9HESTIMATES/)
 77   FORMAT(1H ,9HPARAMETER,7X,12HNEXT-TO-BEST,8X,
     14HBEST/)
 80   FORMAT(1H ,I5,2F20.7)
 82   FORMAT(1H //,1H ,6X,15HFUNCTION VALUES/)
 83   FORMAT(1H ,5X,13H NEXT-TO-BEST,8X,4HBEST/)
 84   FORMAT(1H ,2F15.9)
      STOP
      END
C
C             NELDER-MEAD SUBROUTINE
C
      SUBROUTINE NELMIN(N,START,XMIN,XSEC,YNEWLO,YSEC,
     1REQMIN,STEP,ICOUNT)
      DOUBLE PRECISION START(N),STEP(N),XMIN(N),
     1XSEC(N),YNEWLO,YSEC,REQMIN,P(20,21),PSTAR(20),
     2P2STAR(20),PBAR(20),Y(20),DN,Z,YLO,RCOEFF,
     3YSTAR,ECOEFF,Y2STAR,CCOEFF,FN,DABIT,DCHK,
     4COORD1,COORD2
      DATA RCOEFF/1.0D0/,ECOEFF/2.0D0/,CCOEFF/0.5D0/
      KCOUNT=ICOUNT
      ICOUNT=0
C
      IF(REQMIN.LE.0.D0) ICOUNT=ICOUNT-1
      IF(N.LE.0) ICOUNT=ICOUNT-10
      IF(N.GT.20) ICOUNT=ICOUNT-10
      IF(ICOUNT.LT.0) RETURN
C
      DABIT=2.04607D-35
      BIGNUM=1.0D38
```

```
      KONVGE=5
      XN=FLOAT(N)
      DN=DFLOAT(N)
      NN=N+1
C
C     CONSTRUCTION OF INITIAL SIMPLEX
C
 1001 DO 1 I=1,N
    1 P(I,NN)=START(I)
      Y(NN)=FN(START)
      ICOUNT=ICOUNT+1
      DO 2 J=1,N
      DCHK=START(J)
      START(J)=DCHK+STEP(J)
      DO 3 I=1,N
    3 P(I,J)=START(I)
      Y(J)=FN(START)
      ICOUNT=ICOUNT+1
    2 START(J)=DCHK
C
C      SIMPLEX CONSTRUCTION COMPLETE
C
C      FIND HIGHEST AND LOWEST Y VALUES
C      YNEWLO (Y(IHI)) INDICATES THE VERTEX OF
C      THE SIMPLEX TO BE REPLACED
C
 1000 YLO=Y(1)
      YNEWLO=YLO
      ILO=1
      IHI=1
      DO 5 I=2,NN
      IF(Y(I).GE.YLO) GOTO 4
      YLO=Y(I)
      ILO=I
    4 IF(Y(I).LE.YNEWLO) GOTO 5
      YNEWLO=Y(I)
      IHI=I
    5 CONTINUE
C
C       PERFORM CONVERGENCE CHECKS ON FUNCTION
C
      DCHK=(YNEWLO+DABIT)/(YLO+DABIT)-1.D0
      IF(DABS(DCHK).LT.REQMIN) GOTO 900
C
      KONVGE=KONVGE-1
      IF(KONVGE.NE.0) GOTO 2020
      KONVGE=5
C
C       CHECK CONVERGENCE OF COORDINATES ONLY
C       EVERY 5 SIMPLEXES
C
      DO 2015 I=1,N
      COORD1=P(I,1)
      COORD2=COORD1
```

```
      DO 2010 J=2,NN
      IF(P(I,J).GE.COORD1) GOTO 2005
      COORD1=P(I,J)
 2005 IF(P(I,J).LE.COORD2) GOTO 2010
      COORD2=P(I,J)
 2010 CONTINUE
      DCHK=(COORD2+DABIT)/(COORD1+DABIT)-1.D0
      IF(DABS(DCHK).GT.REQMIN) GOTO 2020
 2015 CONTINUE
      GOTO 900
 2020 IF(ICOUNT.GE.KCOUNT) GOTO 900
C
C        CALCULATE PBAR, THE CENTROID OF THE
C        SIMPLEX VERTICES EXCEPTING THAT WITH
C        Y VALUE YNEWLO
C
      DO 7 I=1,N
      Z=0.0D0
      DO 6 J=1,NN
    6 Z=Z+P(I,J)
      Z=Z-P(I,IHI)
    7 PBAR(I)=Z/DN
C
C        REFLECTION THROUGH THE CENTROID
C
      DO 8 I=1,N
    8 PSTAR(I)=(1.0D0+RCOEFF)*PBAR(I)-RCOEFF*P(I,IHI)
      YSTAR=FN(PSTAR)
      ICOUNT=ICOUNT+1
      IF(YSTAR.GE.YLO) GOTO 12
      IF(ICOUNT.GE.KCOUNT) GOTO 19
C
C        SUCCESSFUL REFLECTION, SO EXTENSION
C
      DO 9 I=1,N
    9 P2STAR(I)=ECOEFF*PSTAR(I)+(1.0D0-ECOEFF)*PBAR(I)
      Y2STAR=FN(P2STAR)
      ICOUNT=ICOUNT+1
C
C        RETAIN EXTENSION OR CONTRACTION
C
      IF(Y2STAR.GE.YSTAR) GOTO 19
   10 DO 11 I=1,N
   11 P(I,IHI)=P2STAR(I)
      Y(IHI)=Y2STAR
      GOTO 1000
C
C        NO EXTENSION
C
   12 L=0
      DO 13 I=1,NN
      IF(Y(I).GT.YSTAR) L=L+1
   13 CONTINUE
      IF(L.GT.1) GOTO 19
```

```
      IF(L.EQ.0) GOTO 15
C
C        CONTRACTION ON THE REFLECTION SIDE OF THE
C        CENTROID
C
      DO 14 I=1,N
   14 P(I,IHI)=PSTAR(I)
      Y(IHI)=YSTAR
C
C        CONTRACTION ON THE Y(IHI) SIDE OF THE CENTROID
C
   15 IF(ICOUNT.GE.KCOUNT) GOTO 900
      DO 16 I=1,N
   16 P2STAR(I)=CCOEFF*P(I,IHI)+(1.0D0-CCOEFF)*PBAR(I)
      Y2STAR=FN(P2STAR)
      ICOUNT=ICOUNT+1
      IF(Y2STAR.LT.Y(IHI)) GOTO 10
C
C        CONTRACT THE WHOLE SIMPLEX
C
      DO 18 J=1,NN
      DO 17 I=1,N
      P(I,J)=(P(I,J)+P(I,ILO))*0.5D0
   17 XMIN(I)=P(I,J)
      Y(J)=FN(XMIN)
   18 CONTINUE
      ICOUNT=ICOUNT+NN
      IF(ICOUNT.LT.KCOUNT) GOTO 1000
      GOTO 900
C
C        RETAIN REFLECTION
C
C   19 CONTINUE
   19 DO 20 I=1,N
   20 P(I,IHI)=PSTAR(I)
      Y(IHI)=YSTAR
      GOTO 1000
C
C        SELECT THE TWO BEST FUNCTION VALUES (YNEWLO
C        AND YSEC) AND THEIR COORDS. (XMIN AND XSEC)
C
  900 DO 23 J=1,NN
      DO 22 I=1,N
   22 XMIN(I)=P(I,J)
      Y(J)=FN(XMIN)
   23 CONTINUE
      YNEWLO=BIGNUM
      DO 24 J=1,NN
      IF(Y(J).GE.YNEWLO) GOTO 24
      YNEWLO=Y(J)
      IBEST=J
   24 CONTINUE
      Y(IBEST)=BIGNUM
      YSEC=BIGNUM
```

```
      DO 25 J=1,NN
      IF(Y(J).GE.YSEC) GOTO 25
      YSEC=Y(J)
      ISEC=J
   25 CONTINUE
      DO 26 I=1,N
      XMIN(I)=P(I,IBEST)
      XSEC(I)=P(I,ISEC)
   26 CONTINUE
      RETURN
      END
C
C
C-----------------------------------------------------------------------
C  LAPLACE SOLUTION (EIGENVALUE SOLUTION)
C                      THEODORSEN:  7th order in s
C                                   approximate   C(k)   &   C(k)*ik
C-----------------------------------------------------------------------
C
      DOUBLE PRECISION FUNCTION FN(lag)
      IMPLICIT REAL*8 (A-Z)
      DOUBLE PRECISION lag(4),t(1,2),tst(1,2)
      INTEGER ib,n,ia,ijob,iz,ier,infer,i,j
      COMPLEX*16 eval(14),A(14,14),B(14,14),EIGA(14),EIGB(14),Z(14,14)
      COMPLEX*16 WK(24,45)
      REAL*8 u,vel,velndnc,velck,atemp1,atemp2,atemp3,atemp4
      REAL*8 trm1,trm2,trm3,trm4,trm5,trm6,trm7,trm8
      REAL*8 trm9,trm10,trm11,trm12,trm13,trm14
      REAL*8 finv(8,2,2),f(9,2,2),m(2,2),c(2,2),k(2,2)
      REAL*8 mc(2,1),cc(2,1),kc(2,1)
      REAL*8 a0(2,2),a1(2,2),a2(2,2),a3(2,2),a4(2,2),a5(2,2),a6(2,2)
      REAL*8 a0c(2,1),a1c(2,1),a2c(2,1),a3c(2,1)
      REAL*8 a4c(2,1),a5c(2,1),a6c(2,1)
      REAL*8 n1(2,7)
      INTEGER y,check
C
C transfer function constants
            t(1,1)=LAG(1)
            t(1,2)=LAG(2)
            tst(1,1)=LAG(3)
            tst(1,2)=LAG(4)
C
C
            velndnc=4.53
            step=0.05
            u=50.0
            wrat=0.2
            r=0.5
            ah=-0.5
C
            ta=0.0012
            tb=0.5484
            cst=50000.0
            ds=cst*ta/tb
```

```
                velndbig=2*velndnc
C
C Read in constants.
      OPEN (UNIT=1,FILE='CONST6.DAT',STATUS='OLD')
      READ (1,*) ia,ib,iz,n,ijob
C                 14 14 14 14 2
      CLOSE (UNIT=1)
C
C open required files
      check=check+1
      if(check.gt.1) goto 66
      OPEN (UNIT=4,FILE='EIGEN.OUT',STATUS='NEW')
      OPEN (UNIT=5,FILE='OPTIMAL.OUT',STATUS='NEW')
C required constants
             pi=3.14159
C
             xtor=0.25
             semic=1.0
             wtor=1.0
             wben=wrat*wtor
C
             xbeta=0.25
             rbeta=0.5
             c1=0.6
C
C required constants
C
             t1=-(1/3)*(1-c1**2)**0.5*(2+c1**2)+c1*acos(c1)
             t2=c1*(1-c1**2)-(1-c1**2)**0.5*(1+c1**2)*acos(c1)+c1*
     1       (acos(c1))**2
             t3=-((1/8)+c1**2)*(acos(c1))**2+0.25*c1*(1-c1**2)**0.5*
     1       acos(c1)*(7+2*c1**2)-(1/8)*(1-c1**2)*(5*c1**2+4)
             t4=-acos(c1)+c1*(1-c1**2)**0.5
             t5=-(1-c1**2)-(acos(c1))**2+2*c1*(1-c1**2)**0.5*acos(c1)
             t6=t2
             t7=-((1/8)+c1**2)*acos(c1)-(1/8)*c1*(1-c1**2)**0.5*
     1       (7+2*c1**2)
             t8=-(1/3)*(1-c1**2)**0.5*(2*c1**2+1)+c1*acos(c1)
             t9=0.5*((1/3)*(1-c1**2)**1.5+ah*t4)
             t10=(1-c1**2)**0.5+acos(c1)
             t11=acos(c1)*(1-2*c1)+(1-c1**2)**0.5*(2-c1)
             t12=(1-c1**2)**0.5*(2+c1)-acos(c1)*(2*c1+1)
             t13=0.5*(-t7-(c1-ah)*t1)
             t14=(1/16)+0.5*ah*c1
C required constants
C
      OPEN (UNIT=10,FILE='LAPLACE.DAT',STATUS='OLD')
          do 25 i=1,2
      READ (10,*) n1(i,1),n1(i,2),n1(i,3),n1(i,4),n1(i,5),
     1                n1(i,6),n1(i,7)
 25       continue
      READ (10,*) b1,b2,b3,b4,err
      CLOSE (UNIT=10)
C
```

```
C FOUR LAG TERMS
C
C PADE APPROXIMANT  THEODORSEN FUNCTION    C(k)  AND  C(k)*ik
C
C term Q(1,1): C(k)*ik
C
                a0(1,1)= n1(2,1)
                a1(1,1)= n1(2,2)
                a2(1,1)= n1(2,3)
                a3(1,1)= n1(2,4)
                a4(1,1)= n1(2,5)
                a5(1,1)= n1(2,6)
                a6(1,1)= n1(2,7)
C
C term Q(1,2): C(k)*ik*(0.5-ah)+C(k)
C
                a0(1,2)= n1(2,1)*(0.5-ah)+n1(1,1)
                a1(1,2)= n1(2,2)*(0.5-ah)+n1(1,2)
                a2(1,2)= n1(2,3)*(0.5-ah)+n1(1,3)
                a3(1,2)= n1(2,4)*(0.5-ah)+n1(1,4)
                a4(1,2)= n1(2,5)*(0.5-ah)+n1(1,5)
                a5(1,2)= n1(2,6)*(0.5-ah)+n1(1,6)
                a6(1,2)= n1(2,7)*(0.5-ah)+n1(1,7)
C
C term Q(2,1): C(k)*ik*(-(ah+0.5))
C
                a0(2,1)= n1(2,1)*(-0.5-ah)
                a1(2,1)= n1(2,2)*(-0.5-ah)
                a2(2,1)= n1(2,3)*(-0.5-ah)
                a3(2,1)= n1(2,4)*(-0.5-ah)
                a4(2,1)= n1(2,5)*(-0.5-ah)
                a5(2,1)= n1(2,6)*(-0.5-ah)
                a6(2,1)= n1(2,7)*(-0.5-ah)
C
C term Q(2,2): C(k)*ik*(-(ah+0.5)*(0.5-ah))-(ah+C.5)*C(k)
C
                a0(2,2)= n1(2,1)*(-0.5-ah)*(0.5-ah)-(ah+0.5)*n1(1,1)
                a1(2,2)= n1(2,2)*(-0.5-ah)*(0.5-ah)-(ah+0.5)*n1(1,2)
                a2(2,2)= n1(2,3)*(-0.5-ah)*(0.5-ah)-(ah+0.5)*n1(1,3)
                a3(2,2)= n1(2,4)*(-0.5-ah)*(0.5-ah)-(ah+0.5)*n1(1,4)
                a4(2,2)= n1(2,5)*(-0.5-ah)*(0.5-ah)-(ah+0.5)*n1(1,5)
                a5(2,2)= n1(2,6)*(-0.5-ah)*(0.5-ah)-(ah+0.5)*n1(1,6)
                a6(2,2)= n1(2,7)*(-0.5-ah)*(0.5-ah)-(ah+0.5)*n1(1,7)
C
C        flap control
C
C term Qc(1,1): ik*C(k)*t11/(2*pi)+t10*C(k)/pi
C
                a0c(1,1)= n1(2,1)*t11/(2*pi)+n1(1,1)*t10/pi
                a1c(1,1)= n1(2,2)*t11/(2*pi)+n1(1,2)*t10/pi
                a2c(1,1)= n1(2,3)*t11/(2*pi)+n1(1,3)*t10/pi
                a3c(1,1)= n1(2,4)*t11/(2*pi)+n1(1,4)*t10/pi
                a4c(1,1)= n1(2,5)*t11/(2*pi)+n1(1,5)*t10/pi
                a5c(1,1)= n1(2,6)*t11/(2*pi)+n1(1,6)*t10/pi
```

```
              a6c(1,1)= n1(2,7)*t11/(2*pi)+n1(1,7)*t10/pi
c
c term Qc(2,1) : -(0.5+a)*(ik*C(k)*t11/(2*pi)+t10*C(k)/pi)
              a0c(2,1)= (n1(2,1)*t11/(2*pi)+n1(1,1)*t10/pi)*(-0.5-ah)
              a1c(2,1)= (n1(2,2)*t11/(2*pi)+n1(1,2)*t10/pi)*(-0.5-ah)
              a2c(2,1)= (n1(2,3)*t11/(2*pi)+n1(1,3)*t10/pi)*(-0.5-ah)
              a3c(2,1)= (n1(2,4)*t11/(2*pi)+n1(1,4)*t10/pi)*(-0.5-ah)
              a4c(2,1)= (n1(2,5)*t11/(2*pi)+n1(1,5)*t10/pi)*(-0.5-ah)
              a5c(2,1)= (n1(2,6)*t11/(2*pi)+n1(1,6)*t10/pi)*(-0.5-ah)
              a6c(2,1)= (n1(2,7)*t11/(2*pi)+n1(1,7)*t10/pi)*(-0.5-ah)
c
c
c
              b1234=b1+b2+b3+b4
              b123=b1+b2+b3
              b124=b1+b2+b4
              b134=b1+b3+b4
              b234=b2+b3+b4
              bt123=b1*b2*b3
              bt124=b1*b2*b4
              bt134=b1*b3*b4
              bt234=b2*b3*b4
              bq1234=b1*b2*b3*b4
              bt1234=b1*b2*b3+b1*b2*b4+b1*b3*b4+b2*b3*b4
              bd1234=b1*b2+b1*b3+b2*b3+b1*b4+b2*b4+b3*b4
              bd123=b1*b2+b1*b3+b2*b3
              bd124=b1*b2+b1*b4+b2*b4
              bd134=b1*b3+b1*b4+b3*b4
              bd234=b2*b3+b3*b4+b2*b4
c
c
c titles and initial parameter listing to file
 66           write(4,34) t(1,1),t(1,2)
              write(4,34) tst(1,1),tst(1,2)
              write(4,32)
              write(4,121)
              write(4,121)
              write(4,33)
              write(4,121)
              write(4,34) u,r,ah,wrat,velndnc
 32           format(6x,' LAPLACE METHOD  INCOMPRESSIBLE (THEODORSEN)')
 33           format(6x,'   u           r(alp)     ah     wben/wtor   '
    1                 'no control velocity')
 34           format(6x,f7.1,3f10.1,f8.2)
 121          format('    ')
c
c iterate velocity
              vel=velndnc+0.1
              velck=velndnc*1.3
 60           vel=vel+step
              if(vel/(semic*wtor).gt.velck) goto 800
              wf=vel/semic
              xvar=2*wf**2/u
c
```

```
C formation of matrices :  [M], [C], [K], [D] and [E]
              m(1,1)=1+1/u
              m(1,2)=xtor-ah/u
              m(2,1)=xtor-ah/u
              m(2,2)=r**2+(0.125+ah**2)/u
              c(1,1)=0.0
              c(1,2)=wf/u
              c(2,1)=0.0
              c(2,2)=(0.5-ah)*wf/u
              k(1,1)=wrat**2*wtor**2
              k(1,2)=0.0
              k(2,1)=0.0
              k(2,2)=r**2*wtor**2
              mc(1,1)=xbeta-t1/(pi*u)
              mc(2,1)=rbeta**2+(c1-ah)*xbeta+(-t7-(c1+0.5)*t1+
     1        (0.5-ah)*t1)/(pi*u)
              cc(1,1)=-wf*t4/(u*pi)
              cc(2,1)=(wf/u*pi)*(-(t4-(2/3)*(1-c1**2)**1.5)+(0.5+ah)*t4)
              kc(1,1)=0.0
              kc(2,1)=(wf**2/u)*(t4+t10)/pi
C
C formation of polynomial coefficients :  [F8], [F7], [F6], [F5], [F4]
C                                         [F3], [F2], [F1]
C
C   [F8]s7+[F7]s6+[F6]s5+ ... +[F2]s+[F1] = 0
           do 200 i=1,2
           do 100 j=1,2
C             calculation of [F8]
                      trm1=m(i,j)+xvar*(1/wf)**2*a2(i,j)
                      trm2=mc(i,1)+xvar*(1/wf)**2*a2c(i,1)
              f(8,i,j)=trm1+trm2*t(1,j)+ds*trm2*tst(1,j)
C             calculation of [F7]
                      trm3=c(i,j)+wf*b1234*m(i,j)+xvar*(1/wf)*
     1                (a1(i,j)+b1234*a2(i,j))
                      trm4=cc(i,1)+wf*b1234*mc(i,1)+xvar*(1/wf)*
     1                (a1c(i,1)+b1234*a2c(i,1))
              f(7,i,j)=trm3+trm4*t(1,j)+ds*trm4*tst(1,j)+cst*trm1+
     1        cst*trm2*t(1,j)
C             calculation of [F6]
                      trm5=k(i,j)+wf*b1234*c(i,j)+wf**2*bd1234*
     1                m(i,j)+xvar*(a0(i,j)+b1234*a1(i,j)+bd1234*
     1                a2(i,j)+a3(i,j)+a4(i,j)+a5(i,j)+a6(i,j))
                      trm6=kc(i,1)+wf*b1234*cc(i,1)+wf**2*bd1234*
     1                mc(i,1)+xvar*(a0c(i,1)+b1234*a1c(i,1)+bd1234*
     1                a2c(i,1)+a3c(i,1)+a4c(i,1)+a5c(i,1)+a6c(i,1))
              f(6,i,j)=trm5+trm6*t(1,j)+ds*trm6*tst(1,j)+cst*
     1        trm3+cst*trm4*t(1,j)
C             calculation of [F5]
                      trm7=wf*b1234*k(i,j)+wf**2*bd1234*c(i,j)+wf**3*
     1                bt1234*m(i,j)+xvar*wf*(b1234*a0(i,j)+bd1234*
     1                a1(i,j)+bt1234*a2(i,j)+b234*a3(i,j)+b134*a4(i,j)+
     1                b124*a5(i,j)+b123*a6(i,j))
                      trm8=wf*b1234*kc(i,1)+wf**2*bd1234*cc(i,1)+
     1                wf**3*bt1234*mc(i,1)+xvar*wf*(b1234*a0c(i,1)+
```

```
     1                bd1234*a1c(i,1)+bt1234*a2c(i,1)+b234*a3c(i,1)+
     1                b134*a4c(i,1)+b124*a5c(i,1)+b123*a6c(i,1))
            f(5,i,j)=trm7+trm8*t(1,j)+ds*trm8*tst(1,j)+cst*
     1      trm5+cst*trm6*t(1,j)
c           calculation of [F4]
                  trm9=wf**2*bd1234*k(i,j)+wf**3*bt1234*c(i,j)+
     1            wf**4*bq1234*m(i,j)+xvar*wf**2*(bd1234*a0(i,j)+
     1            bt1234*a1(i,j)+bq1234*a2(i,j)+bd234*a3(i,j)+
     1            bd134*a4(i,j)+bd124*a5(i,j)+bd123*a6(i,j))
                  trm10=wf**2*bd1234*kc(i,1)+wf**3*bt1234*cc(i,1)+
     1            wf**4*bq1234*mc(i,1)+xvar*wf**2*(bd1234*a0c(i,1)+
     1            bt1234*a1c(i,1)+bq1234*a2c(i,1)+bd234*a3c(i,1)+
     1            bd134*a4c(i,1)+bd124*a5c(i,1)+bd123*a6c(i,1))
            f(4,i,j)=trm9+trm10*t(1,j)+ds*trm10*tst(1,j)+cst*
     1      trm7+cst*trm8*t(1,j)
c           calculation of [F3]
                  trm11=wf**3*bt1234*k(i,j)+wf**4*bq1234*c(i,j)+
     1            xvar*wf**3*(bt1234*a0(i,j)+bq1234*a1(i,j)+
     1            bt234*a3(i,j)+bt134*a4(i,j)+bt124*a5(i,j)+
     1            bt123*a6(i,j))
                  trm12=wf**3*bt1234*kc(i,1)+wf**4*bq1234*cc(i,1)+
     1            xvar*wf**3*(bt1234*a0c(i,1)+bq1234*a1c(i,1)+
     1            bt234*a3c(i,1)+bt134*a4c(i,1)+bt124*a5c(i,1)+
     1            bt123*a6c(i,1))
            f(3,i,j)=trm11+trm12*t(1,j)+ds*trm12*tst(1,j)+cst*
     1      trm9+cst*trm10*t(1,j)
c           calculation of [F2]
                  trm13=wf**4*bq1234*k(i,j)+xvar*wf**4*bq1234*
     1            a0(i,j)
                  trm14=wf**4*bq1234*kc(i,1)+xvar*wf**4*bq1234*
     1            a0c(i,1)
            f(2,i,j)=trm13+trm14*t(1,j)+ds*trm14*tst(1,j)+cst*
     1      trm11+cst*trm12*t(1,j)
c           calculation of [F1]
            f(1,i,j)=cst*(wf**4*bq1234*k(i,j)+xvar*wf**4*bq1234*
     1      a0(i,j))+cst*(wf**4*bq1234*kc(i,1)+xvar*wf**4*bq1234*
     1      a0c(i,1))*t(1,j)
 100      continue
 200      continue
c
c           calculation of inverse  ( inv[F9] )
            f8det=f(8,1,1)*f(8,2,2)-f(8,1,2)*f(8,2,1)
            if(f8det.eq.0.0) goto 210
            goto 220
 210        write(4,215)
 215        format(' DETERMINANT EQUALS ZERO - INVESTIGATE INPUT ',
     1      'CONSTANT ... ah ')
            goto 800
 220        finv(8,1,1)=f(8,2,2)/f8det
            f(9,1,1)=finv(8,1,1)
            finv(8,2,2)=f(8,1,1)/f8det
            f(9,2,2)=finv(8,2,2)
            finv(8,1,2)=-f(8,1,2)/f8det
            f(9,1,2)=finv(8,1,2)
```

```
              finv(8,2,1)=-f(8,2,1)/f8det
              f(9,2,1)=finv(8,2,1)
c
C formation of matrices [A] and [B] to
C solve eigenvalue problem given by [A]{x} = lambda [B]{x}
          do 400 i=1,n
          do 300 j=1,n
              a(i,j)=dcmplx(0.0,0.0)
              if(i.eq.j+2) a(i,j)=dcmplx(1.0,0.0)
              b(i,j)=dcmplx(0.0,0.0)
              if(i.eq.j) b(i,j)=dcmplx(1.0,0.0)
 300      continue
 400      continue
          do 500 i=1,n/2
              y=n/2+1-i
              atemp1=-(finv(8,1,1)*f(y,1,1)+finv(8,1,2)*f(y,2,1))
              a(1,2*i-1)=dcmplx(atemp1,0.0)
              atemp2=-(finv(8,1,1)*f(y,1,2)+finv(8,1,2)*f(y,2,2))
              a(1,2*i)=dcmplx(atemp2,0.0)
              atemp3=-(finv(8,2,1)*f(y,1,1)+finv(8,2,2)*f(y,2,1))
              a(2,2*i-1)=dcmplx(atemp3,0.0)
              atemp4=-(finv(8,2,1)*f(y,1,2)+finv(8,2,2)*f(y,2,2))
              a(2,2*i)=dcmplx(atemp4,0.0)
 500      continue
C
C
C subrountine for eigenvalue calculation
C abar : real part of eval( )
C bbar : imaginary part of eval( )
      CALL EIGZC(A,IA,B,IB,N,IJOB,EIGA,EIGB,Z,IZ,WK,INFER,IER)
          do 5 I=1,n
 5            EVAL(I)=EIGA(I)/EIGB(I)
C
C printout of velocity and associated eigenvalues
C flutter velocity check
              vndcrit=0.0
          do 6 i=1,n
c              write(4,22) eval(i)
              if(dreal(eval(i)).gt.0.0001) vndcrit=vel/(semic*wtor)
C     1        dreal(eval(i)),dimag(eval(i))/wtor
 6        continue
               FN=velndbig-vndcrit
               perc=((vndcrit-velndnc)/velndnc)*100
               if(vndcrit.gt.0.0) goto 779
 21           format(6x,' performance index is', 2f15.5)
 23           format(6x,' ND VELOCITY ',f6.3,' Ta ',f6.4)
 22           format(6x,8f17.5)
 28           format(6x,f17.5,',',f17.5,',',f17.5)
 41           format(6x,'NO CONTROL VELOCITY ',f5.2,'  CRITICAL ',
     1        'VELOCITY ',f5.2,'  % INCREASE ',F5.2,' %')
 42           format(6x,' Ta ',f7.4,'   Tb ',f7.4)
 43           format(6x,' [ t ]   ',2f14.4)
 44           format(6x,' [ tst ] ',2f14.4)
C
```

```
700        goto 60
779        write(5,41) velndnc,vndcrit,perc
           write(5,42) ta,tb
           write(5,43) t(1,1),t(1,2)
           write(5,44) tst(1,1),tst(1,2)
           write(5,121)
789   RETURN
800   END
```

INPUT FILE : LAPLACE.DAT

```
    .99828226      -.00003562        .00000474       -.04120375
-.16289333      -.22908745      -.06459371

    -.00001355         .50037475        -.00003629          .00061076
.01380489       .05848218       .04963449

    .015440852  .084286467  .255950673  .765113621  .000011782
```

INPUT FILE : CONST6.DAT

```
14  14  14  14  2
```

OUTPUT FILE : OPTIMAL.OUT

```
NO CONTROL VELOCITY  4.53  CRITICAL VELOCITY  4.78   % INCREASE  5.52 %
     Ta    .0012    Tb    .5484
     [ t ]              .0000         -.3000
     [ tst ]           4.0000         3.6000

NO CONTROL VELOCITY  4.53  CRITICAL VELOCITY  4.68   % INCREASE  3.31 %
     Ta    .0012    Tb    .5484
     [ t ]              .1000         -.3000
     [ tst ]           4.0000         3.6000

NO CONTROL VELOCITY  4.53  CRITICAL VELOCITY  4.68   % INCREASE  3.31 %
     Ta    .0012    Tb    .5484
     [ t ]              .0000         -.4000
     [ tst ]           4.0000         3.6000

NO CONTROL VELOCITY  4.53  CRITICAL VELOCITY  4.78   % INCREASE  5.52 %
     Ta    .0012    Tb    .5484
     [ t ]              .0000         -.3000
     [ tst ]           4.1000         3.6000

NO CONTROL VELOCITY  4.53  CRITICAL VELOCITY  4.78   % INCREASE  5.52 %
     Ta    .0012    Tb    .5484
     [ t ]              .0000         -.3000
     [ tst ]           4.0000         3.7000

NO CONTROL VELOCITY  4.53  CRITICAL VELOCITY  4.73   % INCREASE  4.42 %
     Ta    .0012    Tb    .5484
     [ t ]              .0500         -.3000
     [ tst ]           4.0500         3.6000

NO CONTROL VELOCITY  4.53  CRITICAL VELOCITY  4.88   % INCREASE  7.73 %
     Ta    .0012    Tb    .5484
     [ t ]              .0000         -.3500
     [ tst ]           4.0500         3.6000

NO CONTROL VELOCITY  4.53  CRITICAL VELOCITY  4.78   % INCREASE  5.52 %
     Ta    .0012    Tb    .5484
     [ t ]              .0000         -.3000
     [ tst ]           4.1000         3.6000

NO CONTROL VELOCITY  4.53  CRITICAL VELOCITY  4.78   % INCREASE  5.52 %
     Ta    .0012    Tb    .5484
     [ t ]              .0000         -.3000
     [ tst ]           4.0500         3.6500

NO CONTROL VELOCITY  4.53  CRITICAL VELOCITY  4.78   % INCREASE  5.52 %
     Ta    .0012    Tb    .5484
     [ t ]              .0000         -.3000
     [ tst ]           4.0500         3.6000
```

```
NO CONTROL VELOCITY  4.53  CRITICAL VELOCITY  4.73   % INCREASE  4.42 %
        Ta    .0012    Tb    .5484
        [ t ]               .0500          -.3000
        [ tst ]            4.0500          3.6000

NO CONTROL VELOCITY  4.53  CRITICAL VELOCITY  4.88   % INCREASE  7.73 %
        Ta    .0012    Tb    .5484
        [ t ]               .0000          -.3500
        [ tst ]            4.0500          3.6000

NO CONTROL VELOCITY  4.53  CRITICAL VELOCITY  4.78   % INCREASE  5.52 %
        Ta    .0012    Tb    .5484
        [ t ]               .0000          -.3000
        [ tst ]            4.1000          3.6000

NO CONTROL VELOCITY  4.53  CRITICAL VELOCITY  4.78   % INCREASE  5.52 %
        Ta    .0012    Tb    .5484
        [ t ]               .0000          -.3000
        [ tst ]            4.0500          3.6500

NO CONTROL VELOCITY  4.53  CRITICAL VELOCITY  4.78   % INCREASE  5.52 %
        Ta    .0012    Tb    .5484
        [ t ]               .0000          -.3000
        [ tst ]            4.0500          3.6000
```

OUTPUT FILE : DATA.OUT

SEQUENTIAL SIMPLEX : PROBLEM MINIMIZATION

12 TRIALS USED

ESTIMATES

PARAMETER	NEXT-TO-BEST	BEST
1	.0000000	.0000000
2	-.3000000	-.3500000
3	4.1000000	4.0500000
4	3.5999999	3.5999999

FUNCTION VALUES

NEXT-TO-BEST	BEST
4.280000206	4.180000205