JAERI - M
90-057

AUTOMATED FINDER FOR THE CRITICAL CONDITION ON THE LINEAR STABILITY OF FLUID MOTIONS

March 1990

Kaoru FUJIMURA

日 本 原 子 力 研 究 所
Japan Atomic Energy Research Institute

JAERI-Mレポートは，日本原子力研究所が不定期に公刊している研究報告書です。
入手の問合わせは，日本原子力研究所技術情報部情報資料課（〒319-11茨城県那珂郡東海村）あて，お申しこしください。なお，このほかに財団法人原子力弘済会資料センター（〒319-11茨城県那珂郡東海村日本原子力研究所内）で複写による実費頒布をおこなっております。

JAERI-M reports are issued irregularly.
Inquiries about availability of the reports should be addressed to Information Division, Department of Technical Information, Japan Atomic Energy Research Institute, Tokai-mura, Naka-gun, Ibaraki-ken 319-11, Japan.

編集兼発行　日本原子力研究所
印　　刷　㈱原子力資料サービス

JAERI-M 90-057

Automated Finder for the Critical Condition on the Linear Stability of Fluid Motions

Kaoru FUJIMURA

High Temperature Engineering Department
Tokai Research Establishment
Japan Atomic Energy Research Institute
Tokai-mura, Naka-gun, Ibaraki-ken

(Received February 21, 1990)

An automated finder routine for the critical condition on the linear stability of fluid motions is proposed. The Newton-Raphson method was utilized for an iteration to solve nonlinear eigenvalue problems appeared in the analysis. The routine was applied to linear stability problem of a free convection between vertical parallel plates with different non-uniform temperatures as well as a plane Poiseuille flow. An efficiency of the finder routine is demonstrated for several parameter sets, numerically.

Keywords: Hydrodynamic Stability, Linear Critical Condition, Eigenvalue Problems, Newton-Raphson Method

JAERI－M 90－057

流れの安定性に関する臨界条件の決定法

日本原子力研究所東海研究所高温工学部
藤村　薫

（1990年2月21日受理）

流れの線形安定性理論における臨界条件の決定手法を提案した。そこでは解析に現われる非線形固有値問題を解くために，ニュートン・ラプソン法が用いられる。手法は，非一様温度の鉛直平板間に発生する自然対流と平面ポワズイユ流に対する線形安定性の問題に適用され，その有効性が示された。

東海研究所：〒319－11 茨城県那珂郡東海村白方字白根2－4

Contents

1. Introduction 1
2. Procedure to find the critical condition 3
3. Numerical results 9
4. Conclusions 13
References 14
Appendix A Notes on the Program 15
Appendix B Program list 18

目　　次

1. はじめに 1
2. 臨界条件の決定法 3
3. 数値計算結果 9
4. 結　論 13
参考文献 14
付録 A　プログラムについて 15
付録 B　プログラムリスト 18

1. INTRODUCTION

An evaluation of the critical condition is the main objective in linear stability analyses for fluid motions. The governing disturbance equation usually involves the Orr-Sommerfeld equation in it. The latter equation has the form of

$$i\alpha(U-c)(\phi''-\alpha^2\phi)-U''\phi-R^{-1}(\phi^{iv}-2\alpha^2\phi''+\alpha^4\phi)=0, \tag{1}$$

where i is the imaginary unit, U is a basic flow as a function of transverse co-ordinate y ($U=1-y^2$ for plane Poiseuille flow, $U=y$ for plane Couette flow, and so on), c is a complex phase velocity, α is a wavenumber, ϕ is an amplitude function of y for a normal mode, R is the Reynolds number, and the prime denotes a differentiation with respect to y. The boundary conditions for ϕ are imposed as

$$\phi(\pm 1)=\phi'(\pm 1)=0. \tag{2}$$

Equations (1) and (2) provide an eigenvalue problem. The complex phase velocity c is determined as the eigenvalue for prescribed values of (α, R). Linear growth rate is defined by αc_i where c_i is the imaginary part of c while a phase velocity is defined by c_r, the real part of c. If $\alpha c_i=0$, the flow is neutrally stable. The critical condition in the parameter space (α, R) is determined as a pair of the minimum Reynolds number R and corresponding α satisfying that $\alpha c_i=0$, $dR/d\alpha=0$, and $d^2R/d\alpha^2>0$. If the neutral stability curve is double nosed as was obtained in unstably stratified shear flows[1] for instance, the last condition is necessary. If the neutral curve has single nose, on the other hand, the critical condition is given by only two simple conditions $\alpha c_i=0$ and $dR/d\alpha=0$. We consider the latter case here.

Consider a situation $c_r=0$ and $c_i=0$. This is the case for stationary disturbances in plane Couette flow, free shear layer with anti-symmetric velocity profile, and a free convection between vertical parallel plates with different temperatures. Then the equation (1) is simplified as

$$i\alpha U(\phi''-\alpha^2\phi)-U''\phi-R^{-1}(\phi^{iv}-2\alpha^2\phi''+\alpha^4\phi)=0, \tag{3}$$

for the neutral state. The neutral Reynolds number for some specific α can thus be obtained from (3) as an eigenvalue and depicting the neutral stability curve is a simple task. The critical condition for this case is thereafter determined numerically.

The situation $c_r=0$ and $c_i=0$, however, is not the case in general. Indeed, all the traveling wave disturbances have non-zero c_r even in the plane Couette flow. We have to solve (1) and obtain the complex phase velocity, at first. The neutral condition is determined thereafter, and the critical condition is finally calculated. Numerical errors will be accumulated during the numerical determination of the critical condition.

By the way, a solution of the Orr-Sommerfeld equation is known to become singular as $R\rightarrow\infty$ in the critical layer.[2] Behavior of $\phi(y)$ across the critical layer is so rapidly changing that highly accurate numerical treatment is required to handle this equation. Recent progress in the numerical computation guarantees that an expansion of $\phi(y)$ in Chebyshev polynomials $T_n(y)$ or Legendre polynomials $P_n(y)$ provides reliable numerical accuracy.[2,3] The accuracy kept in the expansion in these orthogonal Jacobian polynomials should also be maintained in determining the critical condition.

The objective of the present report is to establish these determination process of the critical condition with extremely high accuracy. We consider about more complicated system than the Orr-Sommerfeld problem : linear stability of a free convection between vertical parallel plates with different non-uniform temperatures. Governing equations are composed of an equation for velocity disturbance and an equation for temperature disturbance. The two equations are coupled through a buoyancy term. The equation for the velocity disturbance involves the Orr-Sommerfeld equation in it.

We describe the way to find the critical condition in §2 and give a brief guide about the numerical program in §3. The program list is given in §4 and sample numerical results are presented in §5.

2. PROCEDURE TO FIND THE CRITICAL CONDITION

Let us briefly give an idea to get the critical condition, at first. The critical state is a special state satisfying $\partial G/\partial\alpha=0$ among the neutral states. Thus the neutral states for some specific values of α are evaluated at first in ordinary critical-condition finder routine. Note that determination of the neutral state requires several iterations, N_1 say, at least. Then the finding of the critical condition by making use of the stability data for the neutral disturbances requires another several iterations, N_2 say, at least. This means that $N_1 \times N_2$ iterations are necessary in order to determine the critical condition. If we assume the critical conditions $c_i=0$ and $\partial G/\partial\alpha=0$ from the very beginning, the number of iterations will be reduced substantially. The former condition can be imposed if we split the disturbance equation in the real and the imaginary parts while the latter condition can be imposed if we differentiate the disturbance equation with respect to α. In the following, we propose such an efficient routine to find the critical condition.

Start from non-dimensionalized disturbance equations of the form of

$$i\alpha(W-c)S\phi-i\alpha W''\phi-G^{-1}S^2\phi+G^{-1}D\theta=0, \tag{4}$$

$$i\alpha(W-c)\theta-(GP)^{-1}S\theta+i\alpha T'\phi-\beta D\phi=0, \tag{5}$$

where α is a wavenumber, c is a complex phase velocity, ϕ is an amplitude function of the normal mode for velocity disturbance, θ is an amplitude function of the normal mode for temperature disturbance, G is the Grashof number, P is the Prandtl number, β is a non-dimensional temperature gradient in the vertical (z) direction, $D\equiv d/dx$, and $S\equiv D^2-\alpha^2$. Homogeneous boundary conditions are imposed at $x=\pm1$: $\phi(\pm1)=\phi'(\pm1)=\theta(\pm1)=0$. The basic fields W and T are functions of x and β as

$$W=-\frac{\mathrm{Im}\, f}{2m^2},\ T=\mathrm{Re}\, f, \tag{6,7}$$

where $f(x,m)=\dfrac{\sinh[(1+i)mx]}{\sinh[(1+i)m]}$, and $m=\left(\dfrac{\beta GP}{4}\right)^{1/4}$.[4,5]

Consider the neutral state. The imaginary part of c then vanishes. Denote the real part of c as c for simplicity and without any confusion. Decompose (4) and (5) into the real and the

imaginary parts of the form of

$$-\alpha(W-c)S\phi_i+\alpha W''\phi_i-G^{-1}S^2\phi_r+G^{-1}D\theta_r=0, \tag{8}$$

$$\alpha(W-c)S\phi_r-\alpha W''\phi_r-G^{-1}S^2\phi_i+G^{-1}D\theta_i=0, \tag{9}$$

$$-\alpha(W-c)\theta_i-(GP)^{-1}S\theta_r-\alpha T'\phi_i-4m^4(GP)^{-1}D\phi_r=0, \tag{10}$$

$$\alpha(W-c)\theta_r-(GP)^{-1}S\theta_i+\alpha T'\phi_r-4m^4(GP)^{-1}D\phi_i=0. \tag{11}$$

Split ϕ_r, ϕ_i, θ_r, θ_i, c, and G into the old values and new corrections as

$$\phi_r\rightarrow\Phi_r+\phi_r,\ \phi_i\rightarrow\Phi_i+\phi_i,\ \theta_r\rightarrow\Theta_r+\theta_r,\ \theta_i\rightarrow\Theta_i+\theta_i,\ c\rightarrow C+c,\ \text{and}\ G^{-1}\rightarrow G^{-1}+g^{-1}.$$

Substitute these expression into (8)-(11) and neglect the nonlinear terms with respect to the correction components. We then have the following linearized equations :

$$-\alpha(W-C)S\phi_i+\alpha W''\phi_i-G^{-1}S^2\phi_r+G^{-1}D\theta_r+\alpha cS\Phi_i-g^{-1}S^2\Phi_r+g^{-1}D\Theta_r$$
$$=-[-\alpha(W-C)S\Phi_i+\alpha W''\Phi_i-G^{-1}S^2\Phi_r+G^{-1}D\Theta_r], \tag{12}$$

$$\alpha(W-C)S\phi_r-\alpha W''\phi_r-G^{-1}S^2\phi_i+G^{-1}D\theta_i-\alpha cS\Phi_r-g^{-1}S^2\Phi_i+g^{-1}D\Theta_i$$
$$=-[\alpha(W-C)S\Phi_r-\alpha W''\Phi_r-G^{-1}S^2\Phi_i+G^{-1}D\Theta_i], \tag{13}$$

$$-\alpha(W-C)\theta_i-(GP)^{-1}S\theta_r-\alpha T'\phi_i-4m^4(GP)^{-1}D\phi_r+\alpha c\Theta_i-(gP)^{-1}S\Theta_r-4m^4(gP)^{-1}D\Phi_r$$
$$=-[-\alpha(W-C)\Theta_i-(GP)^{-1}S\Theta_r-\alpha T'\Phi_i-4m^4(GP)^{-1}D\Phi_r], \tag{14}$$

$$\alpha(W-C)\theta_r-(GP)^{-1}S\theta_i+\alpha T'\phi_r-4m^4(GP)^{-1}D\phi_i-\alpha c\Theta_r-(gP)^{-1}S\Theta_i-4m^4(gP)^{-1}D\Phi_i$$
$$=-[\alpha(W-C)\Theta_r-(GP)^{-1}S\Theta_i+\alpha T'\Phi_r-4m^4(GP)^{-1}D\Phi_i]. \tag{15}$$

The above procedure is not new but the Newton-Raphson method itself.

Expand Φ_r, Φ_i, Θ_r, Θ_i, ϕ_r, ϕ_i, θ_r, and θ_i in Chebyshev polynomials as

$$[\Phi_r, \Phi_i, \phi_r, \phi_i]^T = \sum_{n=0}^{N}[\Phi_r^{(n)}, \Phi_i^{(n)}, \phi_r^{(n)}, \phi_i^{(n)}]^T\ (1-x^2)^2T_n(x), \tag{16}$$

$$[\Theta_r, \Theta_i, \theta_r, \theta_i]^T = \sum_{n=0}^{N}[\Theta_r^{(n)}, \Theta_i^{(n)}, \theta_r^{(n)}, \theta_i^{(n)}]^T\ (1-x^2)T_n(x). \tag{17}$$

We introduce here a normalization condition as $\Phi_r^{(0)}=1$, $\phi_r^{(0)}=\Phi_i^{(0)}=\phi_i^{(0)}=0$. Substituting (16) and (17) into (11)-(15) and making use of the collocation method, we obtain coupled algebraic equation of the form of

$$\mathbf{Bx} = \mathbf{b}, \tag{18}$$

where B is a 4(N+1)×4(N+1) matrix, **x** and **b** are 4(N+1)-vectors, and **x** is expressed by

$$\mathbf{x}=[c, \phi_r^{(1)}, \phi_r^{(2)}, \ldots, \phi_r^{(N)}, g^{-1}, \phi_i^{(1)}, \phi_i^{(2)}, \ldots, \phi_i^{(N)}, \theta_r^{(0)}, \theta_r^{(1)}, \ldots, \theta_r^{(N)}, \theta_i^{(0)}, \theta_i^{(1)}, \ldots, \theta_i^{(N)}]^T. \tag{19}$$

Eq. (18) is readily solved.

This is the explanation about how to find the neutral Grashof number, phase velocity, and corresponding eigenfunction for given values of α. In order to attack the critical condition, we need initial guesses of $\partial\phi/\partial\alpha$ and $\partial c/\partial\alpha$ for the neutral modes as will be shown below. The above procedure is used for this purpose.

Let us now proceed to the next step : determination of the critical condition. The critical condition is considered to be given by $dG/d\alpha=0$ and $d^2G/d\alpha^2>0$. In the following, we neglect the second condition. In order to impose the first condition, we differentiate (4) and (5) with respect to α as

$$[i(W-c)S-i\alpha c_\alpha S+i\alpha(W-c)S_\alpha-iW''-G^{-1}{}_\alpha S^2-G^{-1}S^2{}_\alpha]\phi+G^{-1}{}_\alpha D\theta +[i\alpha(W-c)S-i\alpha W''-G^{-1}S^2]\phi_\alpha+G^{-1}D\theta_\alpha=0, \tag{20}$$

$$[i(W-c)-i\alpha c_\alpha-P^{-1}G^{-1}{}_\alpha S-(GP)^{-1}S_\alpha]\theta+iT'\phi+[i\alpha(W-c)-(GP)^{-1}S]\theta_\alpha +i\alpha T'\phi_\alpha-4m^4(GP)^{-1}D\phi_\alpha=0, \tag{21}$$

where suffix α denotes the differentiation with respect to α, $S_\alpha=-2\alpha$, and $S^2{}_\alpha=-4\alpha D^2+4\alpha^3$. Now we impose the condition $G^{-1}{}_\alpha=0$. Decompose (20) and (21) into real and imaginary part of the form of

$$-(W-c)S\phi_i+\alpha c_\alpha S\phi_i-\alpha(W-c)S_\alpha\phi_i+W''\phi_i-G^{-1}S^2{}_\alpha\phi_r-\alpha(W-c)S\phi_{\alpha i}+\alpha W''\phi_{\alpha i} -G^{-1}S^2\phi_{\alpha r}+G^{-1}D\theta_{\alpha r}=0, \tag{22}$$

$$(W-c)S\phi_r-\alpha c_\alpha S\phi_r+\alpha(W-c)S_\alpha\phi_r-W''\phi_r-G^{-1}S^2{}_\alpha\phi_i+\alpha(W-c)S\phi_{\alpha r}-\alpha W''\phi_{\alpha r} -G^{-1}S^2\phi_{\alpha i}+G^{-1}D\theta_{\alpha i}=0, \tag{23}$$

$$-(W-c)\theta_i+\alpha c_\alpha\theta_i-(GP)^{-1}S_\alpha\theta_r-T'\phi_i-\alpha(W-c)\theta_{\alpha i}-(GP)^{-1}S\theta_{\alpha r}-\alpha T'\phi_{\alpha i}$$
$$-4m^4(GP)^{-1}D\phi_{\alpha r}=0, \tag{24}$$

$$(W-c)\theta_r-\alpha c_\alpha\theta_r-(GP)^{-1}S_\alpha\theta_i+T'\phi_r+\alpha(w-c)\theta_{\alpha r}-(GP)^{-1}S\theta_{\alpha i}+\alpha T'\phi_{\alpha r}$$
$$-4m^4(GP)^{-1}D\phi_{\alpha i}=0. \tag{25}$$

Split the variables in (22)-(25) into the old values and corrections as $\phi_{\alpha r}\rightarrow\Phi_{\alpha r}+\phi_{\alpha r}$, $\phi_{\alpha i}\rightarrow\Phi_{\alpha i}+\phi_{\alpha i}$, $\theta_{\alpha r}\rightarrow\Theta_{\alpha r}+\theta_{\alpha r}$, $\theta_{\alpha i}\rightarrow\Theta_{\alpha i}+\theta_{\alpha i}$, $c_\alpha\rightarrow C_\alpha+c_\alpha$, and $\alpha\rightarrow A+\alpha$. Then the Newton-Raphson procedure gives us the following equations for the correction terms :

$$-(W-C)S\phi_i+AC_\alpha S\phi_i-A(W-C)S_\alpha\phi_i+W''\phi_i-G^{-1}S^2{}_\alpha\phi_r-A(W-C)S\phi_{\alpha i}+AW''\phi_{\alpha i}$$
$$-G^{-1}S^2\phi_{\alpha r}+G^{-1}D\theta_{\alpha r}+cS\Phi_i+\alpha C_\alpha S\Phi_i+Ac_\alpha S\Phi_i+\alpha(W-C)S_\alpha\Phi_i$$
$$+AcS_\alpha\Phi_i-g^{-1}S^2{}_\alpha\Phi_r+AcS\Phi_{\alpha i}+\alpha(W-C)S\Phi_{\alpha i}+\alpha W''\Phi_{\alpha i}-g^{-1}S^2\Phi_{\alpha r}$$
$$+g^{-1}D\Theta_{\alpha r}-\alpha(W-C)S_\alpha\Phi_i+\alpha AC_\alpha S_\alpha\Phi_i-\alpha A(W-c)S_{\alpha\alpha}\Phi_i-G^{-1}\alpha S^2{}_{\alpha\alpha}\Phi_r$$
$$-\alpha A(W-C)S_\alpha\Phi_{\alpha i}-G^{-1}\alpha S^2{}_\alpha\Phi_{\alpha r}$$
$$=[-(W-C)S\Phi_i+AC_\alpha S\Phi_i-A(W-C)S_\alpha\Phi_i+W''\Phi_i-G^{-1}S^2{}_\alpha\Phi_r$$
$$-A(W-C)S\Phi_{\alpha i}+AW''\Phi_{\alpha i}-G^{-1}S^2\Phi_{\alpha r}+G^{-1}D\Theta_{\alpha r}], \tag{26}$$

$$(W-C)S\phi_r-AC_\alpha S\phi_r+A(W-C)S_\alpha\phi_r-W''\phi_r-G^{-1}S^2{}_\alpha\phi_i+A(W-C)S\phi_{\alpha r}-AW''\phi_{\alpha r}$$
$$-G^{-1}S^2\phi_{\alpha i}+G^{-1}D\theta_{\alpha i}-cS\Phi_r-\alpha C_\alpha S\Phi_r-Ac_\alpha S\Phi_r-\alpha(W-C)S_\alpha\Phi_r$$
$$-AcS_\alpha\Phi_r-g^{-1}S^2{}_\alpha\Phi_i-AcS\Phi_{\alpha r}-\alpha(W-C)S\Phi_{\alpha r}-\alpha W''\Phi_{\alpha r}-g^{-1}S^2\Phi_{\alpha i}$$
$$+g^{-1}D\Theta_{\alpha i}+\alpha(W-C)S_\alpha\Phi_r-\alpha AC_\alpha S_\alpha\phi_r+\alpha A(W-c)S_{\alpha\alpha}\Phi_r-G^{-1}\alpha S^2{}_{\alpha\alpha}\Phi_i$$
$$+\alpha A(W-C)S_\alpha\Phi_{\alpha r}-G^{-1}\alpha S^2{}_\alpha\Phi_{\alpha i}$$
$$=[(W-C)S\Phi_r-AC_\alpha S\Phi_r+A(W-C)S_\alpha\Phi_r-W''\Phi_r-G^{-1}S^2{}_\alpha\Phi_i$$
$$+A(W-C)S\Phi_{\alpha r}-AW''\Phi_{\alpha r}-G^{-1}S^2\Phi_{\alpha i}+G^{-1}D\Theta_{\alpha i}], \tag{27}$$

$$-(W-C)\theta_i+AC_\alpha\theta_i-(GP)^{-1}S_\alpha\theta_r-T'\phi_i-A(W-C)\theta_{\alpha i}-(GP)^{-1}S\theta_{\alpha r}-AT'\phi_{\alpha i}$$
$$-4m^4(GP)^{-1}D\phi_{\alpha r}+c\Theta_i+\alpha C_\alpha\Theta_i+Ac_\alpha\Theta_i-(gP)^{-1}S_\alpha\Theta_r-(GP)^{-1}\alpha S_{\alpha\alpha}\Theta_r$$
$$-\alpha(W-C)\Theta_{\alpha i}+Ac\Theta_{\alpha i}-(gP)^{-1}S\Theta_{\alpha r}-(GP)^{-1}\alpha S_\alpha\Theta_{\alpha r}-\alpha T'\Phi_{\alpha i}-4m^4(gP)^{-1}D\Phi_{\alpha r}$$
$$=-[-(W-C)\Theta_i+AC_\alpha\Theta_i-(GP)^{-1}S_\alpha\Theta_r-T'\Phi_i-A(W-C)\Theta_{\alpha i}-(GP)^{-1}S\Theta_{\alpha r}$$
$$-AT'\Phi_{\alpha i}-4m^4(GP)^{-1}D\Phi_{\alpha r}], \tag{28}$$

$$(W-C)\theta_r-AC_\alpha\theta_r-(GP)^{-1}S_\alpha\theta_i+T'\phi_r+A(W-C)\theta_{\alpha r}-(GP)^{-1}S\theta_{\alpha i}+AT'\phi_{\alpha r}$$
$$-4m^4(GP)^{-1}D\phi_{\alpha i}-c\Theta_r-\alpha C_\alpha\Theta_r-Ac_\alpha\Theta_r-(gP)^{-1}S_\alpha\Theta_i-(GP)^{-1}\alpha S_{\alpha\alpha}\Theta_i$$
$$+\alpha(W-C)\Theta_{\alpha r}-Ac\Theta_{\alpha r}-(gP)^{-1}S\Theta_{\alpha i}-(GP)^{-1}\alpha S_\alpha\Theta_{\alpha i}+\alpha T'\Phi_{\alpha r}-4m^4(gP)^{-1}D\Phi_{\alpha i}$$
$$=-[(W-C)\Theta_r-AC_\alpha\Theta_r-(GP)^{-1}S_\alpha\Theta_i+T'\Phi_r+A(W-C)\Theta_{\alpha r}-(GP)^{-1}S\Theta_{\alpha i}$$
$$+AT'\Phi_{\alpha r}-4m^4(GP)^{-1}D\Phi_{\alpha i}], \tag{29}$$

where $S\equiv D^2-A^2$, $S_\alpha=-2A$, $S^2{}_\alpha=-4AD^2+4A^3$ $S_{\alpha\alpha}=-2$, and $S^2{}_{\alpha\alpha}=-4D^2+12\alpha^2$. Eqs. (12)-(15) are altered as

$$-A(W-C)S\phi_i+AW''\phi_i-G^{-1}S^2\phi_r+G^{-1}D\theta_r+AcS\Phi_i-g^{-1}S^2\Phi_r+g^{-1}D\Theta_r$$
$$-\alpha(W-C)S\Phi_i-\alpha A(W-C)S_\alpha\Phi_i+\alpha W''\Phi_i-G^{-1}\alpha S^2{}_\alpha\Phi_r$$
$$=-[-\alpha(W-C)S\Phi_i+\alpha W''\Phi_i-G^{-1}S^2\Phi_r+G^{-1}D\Theta_r], \tag{30}$$

$$A(W-C)S\phi_r-AW''\phi_r-G^{-1}S^2\phi_i+G^{-1}D\theta_i-AcS\Phi_r-g^{-1}S^2\Phi_i+g^{-1}D\Theta_i$$
$$+\alpha(W-C)S\Phi_r+\alpha A(W-C)S_\alpha\phi_r-\alpha W''\Phi_r-G^{-1}\alpha S^2{}_\alpha\phi_i$$
$$=-[\alpha(W-C)S\Phi_r-\alpha W''\Phi_r-G^{-1}S^2\Phi_i+G^{-1}D\Theta_i], \tag{31}$$

$$-A(W-C)\theta_i-(GP)^{-1}S\theta_r-AT'\phi_i-4m^4(GP)^{-1}D\phi_r+Ac\Theta_i-(gP)^{-1}S\Theta_r$$
$$-4m^4(gP)^{-1}D\Phi_r-\alpha(W-C)\Theta_i-(GP)^{-1}\alpha S_\alpha\Theta_r-\alpha T'\Phi_i$$
$$=-[-\alpha(W-C)\Theta_i-(GP)^{-1}S\Theta_r-\alpha T'\Phi_i-4m^4(GP)^{-1}D\Phi_r], \tag{32}$$

$$A(W-C)\theta_r-(GP)^{-1}S\theta_i+AT'\phi_r-4m^4(GP)^{-1}D\phi_i-Ac\Theta_r-(gP)^{-1}S\Theta_i$$
$$-4m^4(gP)^{-1}D\Phi_i+\alpha(W-C)\Theta_r-(GP)^{-1}\alpha S_\alpha\Theta_i+\alpha T'\Phi_r$$
$$=-[\alpha(W-C)\Theta_r-(GP)^{-1}S\Theta_i+\alpha T'\Phi_r-4m^4(GP)^{-1}D\Phi_i]. \tag{33}$$

$\Phi_{\alpha r}$, $\Phi_{\alpha i}$, $\phi_{\alpha r}$, $\phi_{\alpha i}$, $\Theta_{\alpha r}$, $\Theta_{\alpha i}$, $\theta_{\alpha r}$, $\theta_{\alpha i}$ are again expanded in

$$[\Phi_{\alpha r}, \Phi_{\alpha i}, \phi_{\alpha r}, \phi_{\alpha i}]^T=\sum_{n=0}^{N}[\Phi_{\alpha r}{}^{(n)}, \Phi_{\alpha i}{}^{(n)}, \phi_{\alpha r}{}^{(n)}, \phi_{\alpha i}{}^{(n)}]^T\ (1-x^2)^2T_n(x), \tag{34}$$

$$[\Theta_{\alpha r}, \Theta_{\alpha i}, \theta_{\alpha r}, \theta_{\alpha i}]^T=\sum_{n=0}^{N}[\Theta_{\alpha r}{}^{(n)}, \Theta_{\alpha i}{}^{(n)}, \theta_{\alpha r}{}^{(n)}, \theta_{\alpha i}{}^{(n)}]^T\ (1-x^2)T_n(x). \tag{35}$$

The normalization condition for the eigenfunction $\Phi_r^{(0)}=1$, $\phi_r^{(0)}=\Phi_i^{(0)}=\phi_i^{(0)}=0$ yields that $\Phi_{\alpha r}^{(0)}=\phi_{\alpha r}^{(0)}=\Phi_{\alpha i}^{(0)}=\phi_{\alpha i}^{(0)}=0$. Under this condition, we make use of the collocation method and eqs. (26)-(33) are reduced to an algebraic equation of the form of

$$\boldsymbol{B}\,\boldsymbol{x}=\boldsymbol{b}\,, \tag{36}$$

again where $\boldsymbol{B}$ is now a 8(N+1)×8(N+1)-matrix, $\boldsymbol{x}$ and $\boldsymbol{b}$ are 8(N+1)-vectors, and $\boldsymbol{x}$ has the component :

$$\begin{aligned}\boldsymbol{x}=[&c,\ \phi_r^{(1)},\phi_r^{(2)},\ \ldots,\ \phi_r^{(N)},\ g^{-1},\ \phi_i^{(1)},\phi_i^{(2)},\ \ldots,\ \phi_i^{(N)},\\ &\theta_r^{(0)},\ \theta_r^{(1)},\ \ldots,\ \theta_r^{(N)},\ \theta_i^{(0)},\ \theta_i^{(1)},\ \ldots,\ \theta_i^{(N)},\\ &c_\alpha,\ \phi_{\alpha r}^{(1)},\phi_{\alpha r}^{(2)},\ \ldots,\ \phi_{\alpha r}^{(N)},\ \alpha,\ \phi_{\alpha i}^{(1)},\phi_{\alpha i}^{(2)},\ \ldots,\ \phi_{\alpha i}^{(N)},\\ &\theta_{\alpha r}^{(0)},\ \theta_{\alpha r}^{(1)},\ \ldots,\ \theta_{\alpha r}^{(N)},\ \theta_{\alpha i}^{(0)},\ \theta_{\alpha i}^{(1)},\ \ldots,\ \theta_{\alpha i}^{(N)}]^T.\end{aligned} \tag{37}$$

See Appendix B for further detail.

We have shown how to determine the critical condition. But in doing so, we need the initial guesses of $\partial\phi/\partial\alpha$ and $\partial c/\partial\alpha$ for the neutral modes. Suppose that M_1 iterations are necessary to get one neutral mode. Then $2M_1$ iterations are required to evaluate the values of the initial guesses. Moreover, we need M_2 iterations to obtain the critical condition. The total number of iterations are thus $2M_1+M_2$. Compare this value with $N_1 \times N_2$ mentioned at the beginning of this section. Values of N_1, M_1, and M_2 are usually almost the same (≤8) while N_2 is usually much greater than M_2. We thus find the efficiency of the present method.

3. NUMERICAL RESULTS

The efficiency of the present method is demonstrated in this section by showing some numerical results. The first example is the critical condition for $m \to 0$. If $P<12.7$, the critical condition is known to be given by stationary modes while the condition is given by traveling modes for $P>12.7$ according to Korpela, Gözüm, and Baxi,[7] or Bergholz,[5] for example. But more detailed evaluation by Chen & Pearlstein[8] revealed recently that the critical value of P lays between 12.4 and 12.5. The critical Prandtl number problem is particularly important if we are interested in nonlinear interaction between the critical stationary mode and the critical traveling modes.

Numerical evaluation of the critical Grashof numbers for various values of the Prandtl number reveals that the critical Prandtl number is given by P=12.45425644, which is consistent with Chen & Pearlstein's result. All the figures of this value are considered to be significant. Let us first show how fast the iteration of the Newton-Raphson method converges in Table 1, 2, 3, and 4 for this value of P.

The convergence of the expansion in Chebyshev polynomials is checked by changing N in eqs. (13), (14), (31), and (32). The results are demonstrated in Table 5.

Table 1 Convergence of the iteration to determine the traveling neutral mode. N=50, P=12.45425644, α=0.3, and $m \to 0$. Number of iterations is denoted as n.

n	G	c
1	506.1758890942	0.05973981528729
2	524.1308121831	0.05991894720832
3	523.7398769797	0.05991456389165
4	523.7400410733	0.05991456572424
5	523.7400410732	0.05991456572424
6	523.7400410732	0.05991456572424

Table 2 Convergence of the iteration to determine the stationary neutral mode. N=50, P=12.45425644, α=1.3, and m→0. Number of iterations is denoted as n.

n	G	c
1	492.0244032281	1.42×10^{-06}
2	495.4206642589	7.96×10^{-10}
3	495.4225026365	-1.02×10^{-14}
4	-495.4225026253	-4.58×10^{-19}
5	495.4225026253	-1.74×10^{-18}

Table 3 Convergence of the iteration to determine the critical condition for traveling modes. N=50, P=12.45425644, and m→0. Number of iterations is denoted as n.

n	α	G	c
1	0.4033127687324	425.6334226340	0.05937239954478
2	0.3567890627764	470.4442485259	0.05958317712729
3	0.3432149532611	490.7950511620	0.05959970914411
4	0.3424139954082	492.0519802353	0.05959947554744
5	0.3424110375720	492.0563332184	0.05959947420192
6	0.3424110375336	492.0563332753	0.05959947420190
7	0.3424110375336	492.0563332753	0.05959947420190

Table 4 Convergence of the iteration to determine the critical condition for stationary modes. N=50, P=12.45425644, and m→0. Number of iterations is denoted as n.

n	α	G	c
1	1.387915924402	488.5291653467	7.80×10^{-15}
2	1.383151658729	492.0452586859	1.59×10^{-17}
3	1.383147622284	492.0563331220	-7.65×10^{-19}
4	1.383147622728	492.0563331734	-7.50×10^{-19}
5	1.383147622728	492.0563331734	-3.10×10^{-18}

We can thus obtain 12 significant figures for the critical condition (α, G) with N=50. Changing the value of the Prandtl number, we determined its critical value as 12.45425644 at which the critical Grashof number for the stationary mode coincides with that for the traveling mode. We finally mention about CPU time taken by FACOM VP-100 computer and required memory size. It takes 7.89 sec. to get the critical condition for stationary mode with N=50 while 17.72 sec. is taken with N=70. It takes 10.16 sec. to get the critical condition for traveling modes with N=50 while 22.78 sec. is used with N=70. CPU times for stationary modes are different from those for traveling modes because of the difference upon the number of iteration. This calculation requires 6,836 KBytes memory size.

Let us now return back to the Orr-Sommerfeld equation (1) for parallel shear flows. Plane Poiseuille flow is a flow which has been investigated most extensively about its linear and nonlinear stability characteristics mainly based on the Orr-Sommerfeld systems. The critical condition has been evaluated repeatedly by different investigators asymptotically or numerically.[2] The well known critical condition was obtained by Orszag based on the expansion method in Chebyshev polynomials as (α, R, c)=(1.02056, 5772.22, 0.264002).[9] This condition was, however, corrected by Davey as (1.020547, 5772.2218, 0.2640003), [see ref. 2], and the validity of the latter condition was confirmed

Table 5 Convergence of the critical conditions with respect to the truncation level N in the expansion in Chebyshev polynomials. P=12.45425644, m→0.

	Stationary Mode		Traveling Mode		
N	α	G	α	G	c
30	1.383174065777	492.0526077323	0.3424110395874	492.0563306782	0.05959947419722
50	1.383147622728	492.0563331734	0.3424110375336	492.0563332753	0.05959947420190
70	1.383147622729	492.0563331732	0.3424110375336	492.0563332753	0.05959947420190

by the present author[10] : the critical wavenumber lays between 1.0205474 and 1.0205475 while the critical Reynolds number is given by 5772.2218. An usual solver for the eigenvalue problem cannot determine the critical wavenumber, critical Reynolds number, and the critical phase velocity beyond 8 significant figures as far as a double precision algorithm is utilized because the critical condition has to be determined based on an detailed information for c_i although $|c_i| / |c_r| \rightarrow 0$ as we approach the critical condition. The present method, on the other hand, is considered to be superior to the others because it assumes the critical condition $\partial G/\partial\alpha = c_i = 0$ from the beginning, exactly. We therefore need not discuss about precise values of c_i.

Modification of our numerical program for the critical condition of plane Poiseuille flow problem is quite straightforward : just neglect terms involving the temperature disturbances and neglect the disturbance equations for the temperature from eqs. (9)-(12) and (23)-(30). The obtained results are listed in Table 6 with different truncation level N in the Chebyshev expansions.

Table 6 Convergence of the critical conditions for plane Poiseuille flow with respect to the truncation level N in the expansion in Chebyshev polynomials.

N	α	R	c
20	0.986275333613	6778.626535352	0.2526983391600
30	1.020619797714	5769.486620674	0.2640421496037
40	1.020547563732	5772.224242992	0.2640002278853
50	1.020547449726	5772.221809140	0.2640002605989
60	1.020547449286	5772.221816201	0.2640002604789
70	1.020547449285	5772.221816210	0.2640002604788
80	1.020547449285	5772.221816210	0.2640002604788
90	1.020547449285	5772.221816210	0.2640002604788
100	1.020547449285	5772.221816210	0.2640002604788
110	1.020547449285	5772.221816210	0.2640002604788
120	1.020547449285	5772.221816210	0.2640002604788
130	1.020547449285	5772.221816210	0.2640002604788
140	1.020547449285	5772.221816210	0.2640002604788

4. CONCLUSIONS

We proposed the method to determine the critical condition on the linear stability of parallel shear flows. The critical conditions are assumed from the beginning there so that any numerical error associated with the value of small linear growth rate does not appear. We demonstrated the efficiency of the present method for two fundamental basic flows : the free convection between vertical parallel plates with different temperatures and the plane Poiseuille flow.

Notes added in proof:

Professor M. Takashima indicated that the same critical Prandtl number $12.4 < P_c < 12.5$ as the one by Chen & Pearlstein was obtained by M. Takashima and H. Hamabata [J. Phys. Soc. Jpn. 53 (1984) pp.1728-1736].

REFERENCES

1. K. Fujimura & R.E. Kelly : "Stability of unstable stratified shear flow between parallel plates", *Fluid Dynamics Research* **2** (1988) 281-292

2. P.G. Drazin & W.H. Reid : *Hydrodynamic Stability* (Cambridge University Press, 1981)

3. D. Gottlieb & S.A. Orszag : *Numerical Analysis of Spectral Methods : Theory and Applications* (SIAM, 1978)

4. J. Mizushima & K. Gotoh : "The stability of natural convection in a vertical fluid layer", *J. Fluid Mech.* **73** (1976) 65-75

5. R.F. Bergholz : Instability of steady natural convection in a vertical fluid layer", *J. Fluid Mech.* **84** (1978) 743-768

6. W.H. Press, B.P. Flannery, S.A. Teukolsky, & W.T. Vetterling : *Numerical Recipes : The Art of Scientific Computing* (Cambridge University Press, 1986)

7. S.A. Korpela, D. Gözüm, & C.B. Baxi : "On the stability of the conduction regime of natural convection in a vertical slot", *Intl. J. Heat Mass Transfer* **16** (1973) 1683-1690

8. Y-M. Chen & A.J. Pearlstein : "Stability of free-convection flows of variable-viscosity fluids in vertical and inclined slots", *J. Fluid Mech.* **198** (1989) 513-541

9. S.A. Orszag : "Accurate solution of the Orr-Sommerfeld equation", *J. Fluid Mech.* **60** (1971) 689-703

10. K. Fujimura : "The equivalence between two perturbation methods in weakly nonlinear stability theory for parallel shear flows", *Proc. R. Soc. Lond.* A**424** (1989) 373-392

Appendix A NOTES ON THE PROGRAM

This program consists of MAIN, CRIT, NEUTRL, PARA, QR, MTRXC, NORMAL, MODE, EFUNC, CHEB1, CHEBY, MTRXPR, LUDCMP, LUBKSB, and VFLOW. Among them, LUDCMP and LUBKSB are copied from the excellent book '*Numerical Recipes*'.[6] [(C) 1986 by Numerical Recipes Software. Reproduced by permission, from the book Numerical Recipes: The Art of Scientific Computing, published by Cambridge University Press.]

MAIN is a main program which control the computer algorithm.

PARA is called in order to get input data. IC controls an objective of the computation : IC=0 leads us to calculate the critical conditions for different values P at fixed value of m while IC=1 let a computer continue to calculate neutral conditions for different α at fixed pair of parameters (P, m). N denotes the maximum degree of Chebyshev polynomials N in (16), (17), (34), and (35). EM indicates m.

EFUNC makes a table for expansion functions and basic fields for fixed value of m. T1 and T2 are expansion functions $(1-x^2)^2T_n(x)$ and $(1-x^2)T_n(x)$, respectively. T1 and T2 have arrays like (i, j, k). The i indicates an order of derivatives, j indicates the degree of Chebyshev polynomials, and k indicates a location of collocation points. U has arrays like (i, j) where i denotes an order of derivatives of W and j denotes the location of collocation points. The $T'(x_j)$ is given by U(3, j), exceptionally.

Usual eigenvalue problem, eqs. (4) and (5) subject to the homogeneous boundary conditions, have to be solved in a usual technique in order to get an initial guess for c, G, ϕ, and θ. The neutrality of the initial guess is not necessary. The following four subroutines are for this purpose.

MTRXC makes each component of matrices which are the final form of Chebyshev collocation method. If we expand ϕ and θ in Chebyshev polynomials as

$$\phi = \sum_{n=0}^{N} \phi^{(n)} (1-x^2)^2 T_n(x), \ \theta = \sum_{n=0}^{N} \theta^{(n)} (1-x^2) T_n(x),$$

then the resultant disturbance equations (4) and (5) at each collocation points are written concisely as $\mathbf{Ax}=c\mathbf{Bx}$ where A and B are $2(N+1)\times 2(N+1)$ matrices and $\mathbf{x}$ is $2(N+1)$-vector. MTRXC provides all the components of matrices A and B.

QR calls QR-subroutine package which is provided in each computer machine, individually. In the present program, $B^{-1}A$ is created by using a LU decomposition routine, a matrix inversion routine, and a product routine between two matrices. Then the eigenvalue of $B^{-1}A$ and associate eigenfunction are calculated by a QR-routine package provided in FACOM machine.

MODE rearranges the eigenvalues and associated eigenfunctions in order for the linear growth rate to satisfy an inequality $\alpha c_i^{(1)} > \alpha c_i^{(2)} > \alpha c_i^{(3)} > \ldots > \alpha c_i^{(n+1)}$.

NORMAL makes the normalization of the eigenfunction so as to satisfy $\phi^{(0)} = 1$ and $\phi^{(n)} = 0$ for $n \neq 0$.

The above subroutines are called in order to obtain the initial guess for the finding procedure of the neutral mode. Now we have to calculate two neutral phase velocities and neutral eigenfunctions for two values of α, α_1 and α_2 which are very close to each other. Then we make initial guesses for c_α, ϕ_α, and θ_α.

NEUTRL is a subroutine to find the neutral condition following eqs. (12)-(19).

CRIT is a subroutine to find the critical condition following eqs. (26)-(37).

CHEB1 provides expansion functions $(1-x^2)^2 T_n(x)$ and $(1-x^2) T_n(x)$ and their derivatives up to 4-th order at each collocation points. Factors $(1-x^2)^2$ and $(1-x^2)$ are involved in order for the expansion functions to satisfy the homogeneous boundary conditions at $x=\pm 1$ for ϕ and θ.

CHEBY generates the Chebyshev polynomials $T_n(x)$ and their derivatives up to 4-th order at each collocation points.

VFLOW generates the basic flow profile, their 1^{St} and 2^{Nd} derivatives and the 1^{St} derivative of the temperature profile at each collocation points. The nondimensional temperature gradient m is involved as a parameter.

MTRXPR makes a product of matrices.

Appendix B PROGRAM LIST

Program list for FACOM computer is presented. We utilized DCLU, DCLUIV, and DCEIG2 subroutine packages from "SSL2" subroutine library installed in FACOM VP-100 in solving (1) and (2) to get the initial guesses.

```
C
C       FINDER FOR THE CRITICAL CONDITION
C       NEWTON-RAPHSON + CHEBYSHEV COLLOCATION
C       NON-UNIFORM TEMPERATURE DISTRIBUTION IN VERTICAL DIRECTION
C
      PARAMETER (IP1=142,IP2=IP1/2)
      IMPLICIT REAL*8(A-H,P-Z),COMPLEX*16(O)
      DIMENSION OC1(IP1,IP1),OC2(IP1,IP1),OF1(IP1),OF2(IP1)
      DIMENSION OEIG(IP1),OFD(IP1)
      DIMENSION T1(0:4,0:IP2,IP2),T2(0:4,0:IP2,IP2),U(0:3,IP2)
        CALL PARA(1,IC,N,EM,P,G,A)
          N2=2*(N+1)
        CALL EFUNC(N,EM,T1,T2,U)
C--- USUAL EIGENVALUE PROBLEM (NON-NEUTRAL) IS SOLVED TO GET
C    INITIAL GUESS FOR THE NEUTRAL CONDITION FINDER 'NEUTRL'
        CALL MTRXC(N,EM,P,G,A,T1,T2,U,OC1,OC2)
        CALL QR(OC1,OC2,OEIG,N2)
        CALL MODE(N2,OEIG,OC1,OF1)
        CALL NORMAL(OF1)
    1   CALL PARA(2,IC,N,EM,P,G,A)
        IF(A.EQ.0.D0) STOP
C--- NEUTRAL CONDITION FINDER IS CALLED TWICE TO GET INITIAL GUESS
C    FOR THE CRITICAL CONDITION FINDER 'CRIT'
          OE1=OEIG(1)
        CALL NEUTRL(N,EM,P,G,A,OE1,T1,T2,U,OF1)
        IF(IC.EQ.0) GO TO 1
          DA=1.D-4
          A2=A+DA
          OE2=OE1
          G2=G
      DO 2 I=1,IP1
    2     OF2(I)=OF1(I)
        CALL NEUTRL(N,EM,P,G2,A2,OE2,T1,T2,U,OF2)
          OED=(OE2-OE1)/DA
      DO 3 I=1,IP1
    3     OFD(I)=(OF2(I)-OF1(I))/DA
C--- CRITICAL CONDITION FINDER IS CALLED
    4   CALL CRIT(N,EM,P,G,A,OE1,OED,T1,T2,U,OF1,OFD)
        CALL PARA(3,IC,N,EM,P,G1,A)
        IF(P.EQ.0.D0) STOP
        GO TO 4
      END
C
C------ NEWTON METHOD TO SEARCH THE CRITICAL CONDITION ------
      SUBROUTINE CRIT(N,EM,P,G,A,OEIG,OEIGD,T1,T2,U,OF,OG)
      PARAMETER (IP1=142,IP2=IP1/2,IP0=2*IP1,IP=2*IP0)
      IMPLICIT REAL*8(A-H,P-Z),COMPLEX*16(O)
      DIMENSION OF(IP1),OG(IP1)
      DIMENSION T1(0:4,0:IP2,IP2),T2(0:4,0:IP2,IP2),U(0:3,IP2)
      DIMENSION FR(0:4,IP1),FI(0:4,IP1),GR(0:4,IP1),GI(0:4,IP1)
      DIMENSION X(IP,IP),Y(IP),INDX(IP)
          N1=N+1
          N2=2*N1
          N3=3*N1
          N4=4*N1
          N5=5*N1
          N6=6*N1
```

```
      N7=7*N1
      N8=8*N1
      C=DBLE(OEIG)
      DC=DBLE(OEIGD)
      G1=1.D0/G
      EPS=1.D-12
      ITER=0
10 CONTINUE
      ITER=ITER+1
      AS=A**2
      AC=A**3
      AQ=A**4
    IF(ITER.GT.20) WRITE(*,*) 'TOO MANY ITERATIONS'
    IF(ITER.GT.20) GO TO 20
  DO 1 K=0,4
  DO 1 I=1,N1
      FR(K,I)=0.D0
      FI(K,I)=0.D0
      FR(K,N1+I)=0.D0
      FI(K,N1+I)=0.D0
      GR(K,I)=0.D0
      GI(K,I)=0.D0
      GR(K,N1+I)=0.D0
      GI(K,N1+I)=0.D0
  DO 1 J=1,N1
      FR(K,I)=FR(K,I)+DBLE(OF(J))*T1(K,J-1,I)
      FI(K,I)=FI(K,I)+IMAG(OF(J))*T1(K,J-1,I)
      FR(K,N1+I)=FR(K,N1+I)+DBLE(OF(N1+J))*T2(K,J-1,I)
      FI(K,N1+I)=FI(K,N1+I)+IMAG(OF(N1+J))*T2(K,J-1,I)
      GR(K,I)=GR(K,I)+DBLE(OG(J))*T1(K,J-1,I)
      GI(K,I)=GI(K,I)+IMAG(OG(J))*T1(K,J-1,I)
      GR(K,N1+I)=GR(K,N1+I)+DBLE(OG(N1+J))*T2(K,J-1,I)
 1    GI(K,N1+I)=GI(K,N1+I)+IMAG(OG(N1+J))*T2(K,J-1,I)
  DO 2 I=1,IP
      Y(I)=0.D0
  DO 2 J=1,IP
 2    X(I,J)=0.D0
  DO 3 I=1,N1
      Y(I)=A*(U(0,I)-C)*(FI(2,I)-AS*FI(0,I))-A*U(2,I)*FI(0,I)
 *        +G1*(FR(4,I)-2.D0*AS*FR(2,I)+AQ*FR(0,I))
 *        -G1*FR(1,N1+I)
      Y(N1+I)=-A*(U(0,I)-C)*(FR(2,I)-AS*FR(0,I))+A*U(2,I)*FR(0,I)
 *           +G1*(FI(4,I)-2.D0*AS*FI(2,I)+AQ*FI(0,I))
 *           -G1*FI(1,N1+I)
      Y(N2+I)=A*(U(0,I)-C)*FI(0,N1+I)
 *           +G1/P*(FR(2,N1+I)-AS*FR(0,N1+I))+A*U(3,I)*FI(0,I)
 *           +4.D0*EM**4*G1/P*FR(1,I)
      Y(N3+I)=-A*(U(0,I)-C)*FR(0,N1+I)
 *           +G1/P*(FI(2,N1+I)-AS*FI(0,N1+I))-A*U(3,I)*FR(0,I)
 *           +4.D0*EM**4*G1/P*FI(1,I)
      Y(N4+I)=(U(0,I)-C)*(FI(2,I)-AS*FI(0,I))
 *           -A*DC*(FI(2,I)-AS*FI(0,I))+A*(U(0,I)-C)*(-2.D0*A)
 *           *FI(0,I)-U(2,I)*FI(0,I)+G1*(-4.D0*A*FR(2,I)
 *           +4.D0*AC*FR(0,I))-(-A*(U(0,I)-C)*(GI(2,I)-AS*GI(0,I))
 *           +A*U(2,I)*GI(0,I)-G1*(GR(4,I)-2.D0*AS*GR(2,I)+AQ
 *           *GR(0,I)))-G1*GR(1,N1+I)
      Y(N5+I)=-(U(0,I)-C)*(FR(2,I)-AS*FR(0,I))+A*DC*(FR(2,I)
 *            -AS*FR(0,I))-A*(U(0,I)-C)*(-2.D0*A)*FR(0,I)+U(2,I)
 *            *FR(0,I)+G1*(-4.D0*A*FI(2,I)+4.D0*AC*FI(0,I))
 *            -(A*(U(0,I)-C)*(GR(2,I)-AS*GR(0,I))-A*U(2,I)*GR(0,I)
 *            -G1*(GI(4,I)-2.D0*AS*GI(2,I)+AQ*GI(0,I)))
 *            -G1*GI(1,N1+I)
      Y(N6+I)=(U(0,I)-C)*FI(0,N1+I)-A*DC*FI(0,N1+I)
 *           +G1/P*(-2.D0*A)*FR(0,N1+I)+U(3,I)*FI(0,I)
 *           -(-A*(U(0,I)-C)*GI(0,N1+I)-G1/P*(GR(2,N1+I)
 *           -AS*GR(0,N1+I)))+A*U(3,I)*GI(0,I)
 *           +4.D0*EM**4*G1/P*GR(1,I)
```

```
      Y(N7+I)=-(U(0,I)-C)*FR(0,N1+I)+A*DC*FR(0,N1+I)
*            +G1/P*(-2.D0*A)*FI(0,N1+I)-U(3,I)*FR(0,I)
*            -(A*(U(0,I)-C)*GR(0,N1+I)-G1/P*(GI(2,N1+I)
*            -AS*GI(0,N1+I)))-A*U(3,I)*GR(0,I)
*            +4.D0*EM**4*G1/P*GI(1,I)
 DO 3 J=1,N+1
      X(I,J)=-G1*(T1(4,J-1,I)-2.D0*AS*T1(2,J-1,I)
*           +AQ*T1(0,J-1,I))
      X(I,N1+J)=-A*(U(0,I)-C)*(T1(2,J-1,I)-AS*T1(0,J-1,I))
*             +A*U(2,I)*T1(0,J-1,I)
      X(I,N2+J)=G1*T2(1,J-1,I)
      X(I,N3+J)=0.D0
      X(N1+I,J)=A*(U(0,I)-C)*(T1(2,J-1,I)-AS*T1(0,J-1,I))
*             -A*U(2,I)*T1(0,J-1,I)
      X(N1+I,N1+J)=-G1*(T1(4,J-1,I)-2.D0*AS*T1(2,J-1,I)
*             +AQ*T1(0,J-1,I))
      X(N1+I,N2+J)=0.D0
      X(N1+I,N3+J)=G1*T2(1,J-1,I)
      X(N2+I,J)=-4.D0*EM**4*G1/P*T1(1,J-1,I)
      X(N2+I,N1+J)=-A*U(3,I)*T1(0,J-1,I)
      X(N2+I,N2+J)=-G1/P*(T2(2,J-1,I)-AS*T2(0,J-1,I))
      X(N2+I,N3+J)=-A*(U(0,I)-C)*T2(0,J-1,I)
      X(N3+I,J)=A*U(3,I)*T1(0,J-1,I)
      X(N3+I,N1+J)=-4.D0*EM**4*G1/P*T1(1,J-1,I)
      X(N3+I,N2+J)=A*(U(0,I)-C)*T2(0,J-1,I)
      X(N3+I,N3+J)=-G1/P*(T2(2,J-1,I)-AS*T2(0,J-1,I))
      X(N4+I,J)=-G1*(-4.D0*A*T1(2,J-1,I)+4.D0*AC*T1(0,J-1,I))
      X(N4+I,N1+J)=-(U(0,I)-C)*(T1(2,J-1,I)-AS*T1(0,J-1,I))
*             +A*DC*(T1(2,J-1,I)-AS*T1(0,J-1,I))
*             -A*(U(0,I)-C)*(-2.D0*A)*T1(0,J-1,I)
*             +U(2,I)*T1(0,J-1,I)
      X(N4+I,N2+J)=0.D0
      X(N4+I,N3+J)=0.D0
      X(N4+I,N4+J)=-G1*(T1(4,J-1,I)-2.D0*AS*T1(2,J-1,I)
*             +AQ*T1(0,J-1,I))
      X(N4+I,N5+J)=-A*(U(0,I)-C)*(T1(2,J-1,I)-AS*T1(0,J-1,I))
*             +A*U(2,I)*T1(0,J-1,I)
      X(N4+I,N6+J)=G1*T2(1,J-1,I)
      X(N4+I,N7+J)=0.D0
      X(N5+I,J)=(U(0,I)-C)*(T1(2,J-1,I)-AS*T1(0,J-1,I))
*             -A*DC*(T1(2,J-1,I)-AS*T1(0,J-1,I))
*             +A*(U(0,I)-C)*(-2.D0*A)*T1(0,J-1,I)
*             -U(2,I)*T1(0,J-1,I)
      X(N5+I,N1+J)=-G1*(-4.D0*A*T1(2,J-1,I)+4.D0*AC*T1(0,J-1,I))
      X(N5+I,N2+J)=0.D0
      X(N5+I,N3+J)=0.D0
      X(N5+I,N4+J)=A*(U(0,I)-C)*(T1(2,J-1,I)-AS*T1(0,J-1,I))
*             -A*U(2,I)*T1(0,J-1,I)
      X(N5+I,N5+J)=-G1*(T1(4,J-1,I)-2.D0*AS*T1(2,J-1,I)
*             +AQ*T1(0,J-1,I))
      X(N5+I,N6+J)=0.D0
      X(N5+I,N7+J)=G1*T2(1,J-1,I)
      X(N6+I,J)=0.D0
      X(N6+I,N1+J)=-U(3,I)*T1(0,J-1,I)
      X(N6+I,N2+J)=-G1/P*(-2.D0*A)*T2(0,J-1,I)
      X(N6+I,N3+J)=-(U(0,I)-C)*T2(0,J-1,I)+A*DC*T2(0,J-1,I)
      X(N6+I,N4+J)=-4.D0*EM**4*G1/P*T1(1,J-1,I)
      X(N6+I,N5+J)=-A*U(3,I)*T1(0,J-1,I)
      X(N6+I,N6+J)=-G1/P*(T2(2,J-1,I)-AS*T2(0,J-1,I))
      X(N6+I,N7+J)=-A*(U(0,I)-C)*T2(0,J-1,I)
      X(N7+I,J)=U(3,I)*T1(0,J-1,I)
      X(N7+I,N1+J)=0.D0
      X(N7+I,N2+J)=(U(0,I)-C)*T2(0,J-1,I)-A*DC*T2(0,J-1,I)
      X(N7+I,N3+J)=-G1/P*(-2.D0*A)*T2(0,J-1,I)
      X(N7+I,N4+J)=A*U(3,I)*T1(0,J-1,I)
      X(N7+I,N5+J)=-4.D0*EM**4*G1/P*T1(1,J-1,I)
      X(N7+I,N6+J)=A*(U(0,I)-C)*T2(0,J-1,I)
```

```
3       X(N7+I,N7+J)=-G1/P*(T2(2,J-1,I)-AS*T2(0,J-1,I))
 - DO 4 I=1,N1
        X(I,1)=A*(FI(2,I)-AS*FI(0,I))
        X(I,N1+1)=-(FR(4,I)-2.D0*AS*FR(2,I)+AQ*FR(0,I))+FR(1,N1+I)
        X(I,N4+1)=0.D0
        X(I,N5+1)=-(U(0,I)-C)*(FI(2,I)-AS*FI(0,I))+U(2,I)*FI(0,I)
 *              -A*(U(0,I)-C)*(-2.D0*A)*FI(0,I)-G1*(-4.D0*A*FR(2,I)
 *              +4.D0*AC*FR(0,I))
        X(N1+I,1)=-A*(FR(2,I)-AS*FR(0,I))
        X(N1+I,N1+1)=-(FI(4,I)-2.D0*AS*FI(2,I)+AQ*FI(0,I))+FI(1,N1+I)
        X(N1+I,N4+1)=0.D0
        X(N1+I,N5+1)=(U(0,I)-C)*(FR(2,I)-AS*FR(0,I))-U(2,I)*FR(0,I)
 *              +A*(U(0,I)-C)*(-2.D0*A)*FR(0,I)-G1*(-4.D0*A*FI(2,I)
 *              +4.D0*AC*FI(0,I))
        X(N2+I,1)=A*FI(0,N1+I)
        X(N2+I,N1+1)=-1.D0/P*(FR(2,N1+I)-AS*FR(0,N1+I))
 *              -4.D0*EM**4/P*FR(1,I)
        X(N2+I,N4+1)=0.D0
        X(N2+I,N5+1)=-(U(0,I)-C)*FI(0,N1+I)-U(3,I)*FI(0,I)-G1/P
 *              *(-2.D0*A)*FR(0,N1+I)
        X(N3+I,1)=-A*FR(0,N1+I)
        X(N3+I,N1+1)=-1.D0/P*(FI(2,N1+I)-AS*FI(0,N1+I))
 *              -4.D0*EM**4/P*FI(1,I)
        X(N3+I,N4+1)=0.D0
        X(N3+I,N5+1)=(U(0,I)-C)*FR(0,N1+I)+U(3,I)*FR(0,I)-G1/P
 *              *(-2.D0*A)*FI(0,N1+I)
        X(N4+I,1)=(FI(2,I)-AS*FI(0,I))+A*(-2.D0*A)*FI(0,I)
 *              +A*(GI(2,I)-AS*GI(0,I))
        X(N4+I,N1+1)=-(-4.D0*A*FR(2,I)+4.D0*AC*FR(0,I))
 *              -(GR(4,I)-2.D0*AS*GR(2,I)+AQ*GR(0,I))+GR(1,N1+I)
        X(N4+I,N4+1)=A*(FI(2,I)-AS*FI(0,I))
        X(N4+I,N5+1)=DC*(FI(2,I)-AS*FI(0,I))-(U(0,I)-C)*(-2.D0*A)
 *              *FI(0,I)-(U(0,I)-C)*(GI(2,I)-AS*GI(0,I))
 *              +U(2,I)*GI(0,I)-(U(0,I)-C)*(-2.D0*A)*FI(0,I)+A*DC
 *              *(-2.D0*A)*FI(0,I)-A*(U(0,I)-C)*(-2.D0)*FI(0,I)
 *              -G1*(-4.D0*FR(2,I)+12.D0*AS*FR(0,I))-A*(U(0,I)-C)
 *              *(-2.D0*A)*GI(0,I)-G1*(-4.D0*A*GR(2,I)+4.D0*AC
 *              *GR(0,I))
        X(N5+I,1)=-(FR(2,I)-AS*FR(0,I))-A*(-2.D0*A)*FR(0,I)
 *              -A*(GR(2,I)-AS*GR(0,I))
        X(N5+I,N1+1)=-(-4.D0*A*FI(2,I)+4.D0*AC*FI(0,I))
 *              -(GI(4,I)-2.D0*AS*GI(2,I)+AQ*GI(0,I))+GI(1,N1+I)
        X(N5+I,N4+1)=-A*(FR(2,I)-AS*FR(0,I))
        X(N5+I,N5+1)=-DC*(FR(2,I)-AS*FR(0,I))+(U(0,I)-C)*(-2.D0*A)
 *              *FR(0,I)+(U(0,I)-C)*(GR(2,I)-AS*GR(0,I))-U(2,I)
 *              *GR(0,I)+(U(0,I)-C)*(-2.D0*A)*FR(0,I)-A*DC*(-2.D0*A)
 *              *FR(0,I)+A*(U(0,I)-C)*(-2.D0)*FR(0,I)-G1*(-4.D0
 *              *FI(2,I)+12.D0*AS*FI(0,I))+A*(U(0,I)-C)*(-2.D0*A)
 *              *GR(0,I)-G1*(-4.D0*A*GI(2,I)+4.D0*AC*GI(0,I))
        X(N6+I,1)=FI(0,N1+I)+A*GI(0,N1+I)
        X(N6+I,N1+1)=-1.D0/P*(-2.D0*A)*FR(0,N1+I)
 *              -1.D0/P*(GR(2,N1+I)-AS*GR(0,N1+I))
 *              -4.D0*EM**4/P*GR(1,I)
        X(N6+I,N4+1)=A*FI(0,N1+I)
        X(N6+I,N5+1)=DC*FI(0,N1+I)-(U(0,I)-C)*GI(0,N1+I)-U(3,I)
 *              *GI(0,I)-G1/P*(-2.D0)*FR(0,N1+I)-G1/P*(-2.D0*A)
 *              *GR(0,N1+I)
        X(N7+I,1)=-FR(0,N1+I)-A*GR(0,N1+I)
        X(N7+I,N1+1)=-1.D0/P*(-2.D0*A)*FI(0,N1+I)
 *              -1.D0/P*(GI(2,N1+I)-AS*GI(0,N1+I))
 *              -4.D0*EM**4/P*GI(1,I)
        X(N7+I,N4+1)=-A*FR(0,N1+I)
        X(N7+I,N5+1)=-DC*FR(0,N1+I)+(U(0,I)-C)*GR(0,N1+I)+U(3,I)
 *              *GR(0,I)-G1/P*(-2.D0)*FI(0,N1+I)-G1/P*(-2.D0*A)
 *              *GI(0,N1+I)
 4 CONTINUE
      CALL LUDCMP(X,N8,INDX,D)
      CALL LUBKSB(X,N8,INDX,Y)
```

```
      DO 5 I=1,N1
          OF(I)=OF(I)+CMPLX(Y(I),Y(N1+I))
          OF(N1+I)=OF(N1+I)+CMPLX(Y(N2+I),Y(N3+I))
          OG(I)=OG(I)+CMPLX(Y(N4+I),Y(N5+I))
    5     OG(N1+I)=OG(N1+I)+CMPLX(Y(N6+I),Y(N7+I))
          OF(1)=1.D0
          OG(1)=0.D0
          C=C+Y(1)
          G1=G1+Y(N1+1)
          DC=DC+Y(N4+1)
          A=A+Y(N5+1)
          ERR=0.D0
      DO 6 I=1,N8
    6     ERR=ERR+ABS(Y(I))
        WRITE(*,600) A,G,C,ERR
        IF(ERR.GT.EPS) GO TO 10
   20     G=1.D0/G1
        WRITE(*,610) P,A,G,C
          OEIG=C
          OEIGD=DC
  600 FORMAT(15X,1P4E15.6)
  610 FORMAT(1PE15.8,1P3E20.12)
      RETURN
      END
C
C------ NEWTON METHOD TO SEARCH THE NEUTRAL CONDITION ------
      SUBROUTINE NEUTRL(N,EM,P,G,A,OEIG,T1,T2,U,OF)
      PARAMETER (IP1=142,IP2=IP1/2,IP0=2*IP1,IP=2*IP0)
      IMPLICIT REAL*8(A-H,P-Z),COMPLEX*16(O)
      DIMENSION OF(IP1)
      DIMENSION T1(0:4,0:IP2,IP2),T2(0:4,0:IP2,IP2),U(0:3,IP2)
      DIMENSION FR(0:4,IP1),FI(0:4,IP1),X(IP,IP),Y(IP)
      DIMENSION INDX(IP)
          N1=N+1
          N2=2*N1
          N3=3*N1
          N4=4*N1
          C=DBLE(OEIG)
          AS=A**2
          AQ=A**4
          G1=1.D0/G
          EPS=1.D-12
          ITER=0
   10 CONTINUE
          ITER=ITER+1
        IF(ITER.GT.20) WRITE(*,*) 'TOO MANY ITERATIONS'
        IF(ITER.GT.20) GO TO 20
      DO 1 K=0,4
      DO 1 I=1,N1
          FR(K,I)=0.D0
          FI(K,I)=0.D0
          FR(K,N1+I)=0.D0
          FI(K,N1+I)=0.D0
      DO 1 J=1,N1
          FR(K,I)=FR(K,I)+DBLE(OF(J))*T1(K,J-1,I)
          FI(K,I)=FI(K,I)+IMAG(OF(J))*T1(K,J-1,I)
          FR(K,N1+I)=FR(K,N1+I)+DBLE(OF(N1+J))*T2(K,J-1,I)
    1     FI(K,N1+I)=FI(K,N1+I)+IMAG(OF(N1+J))*T2(K,J-1,I)
      DO 2 I=1,IP0
          Y(I)=0.D0
      DO 2 J=1,IP0
    2     X(I,J)=0.D0
      DO 3 I=1,N1
          Y(I)=A*(U(0,I)-C)*(FI(2,I)-AS*FI(0,I))-A*U(2,I)*FI(0,I)
     *        +G1*(FR(4,I)-2.D0*AS*FR(2,I)+AQ*FR(0,I))
     *        -G1*FR(1,N1+I)
          Y(N1+I)=-A*(U(0,I)-C)*(FR(2,I)-AS*FR(0,I))+A*U(2,I)*FR(0,I)
     *           +G1*(FI(4,I)-2.D0*AS*FI(2,I)+AQ*FI(0,I))
```

```
     *            -G1*FI(1,N1+I)
          Y(N2+I)=A*(U(0,I)-C)*FI(0,N1+I)
     *            +G1/P*(FR(2,N1+I)-AS*FR(0,N1+I))+A*U(3,I)*FI(0,I)
     *            +4.D0*EM**4*G1/P*FR(1,I)
          Y(N3+I)=-A*(U(0,I)-C)*FR(0,N1+I)
     *            +G1/P*(FI(2,N1+I)-AS*FI(0,N1+I))-A*U(3,I)*FR(0,I)
     *            +4.D0*EM**4*G1/P*FI(1,I)
      DO 3 J=1,N+1
          X(I,J)=-G1*(T1(4,J-1,I)-2.D0*AS*T1(2,J-1,I)
     *           +AQ*T1(0,J-1,I))
          X(I,N1+J)=-A*(U(0,I)-C)*(T1(2,J-1,I)-AS*T1(0,J-1,I))
     *               +A*U(2,I)*T1(0,J-1,I)
          X(I,N2+J)=G1*T2(1,J-1,I)
          X(I,N3+J)=0.D0
          X(N1+I,J)=A*(U(0,I)-C)*(T1(2,J-1,I)-AS*T1(0,J-1,I))
     *               -A*U(2,I)*T1(0,J-1,I)
          X(N1+I,N1+J)=-G1*(T1(4,J-1,I)-2.D0*AS*T1(2,J-1,I)
     *               +AQ*T1(0,J-1,I))
          X(N1+I,N2+J)=0.D0
          X(N1+I,N3+J)=G1*T2(1,J-1,I)
          X(N2+I,J)=-4.D0*EM**4*G1/P*T1(1,J-1,I)
          X(N2+I,N1+J)=-A*U(3,I)*T1(0,J-1,I)
          X(N2+I,N2+J)=-G1/P*(T2(2,J-1,I)-AS*T2(0,J-1,I))
          X(N2+I,N3+J)=-A*(U(0,I)-C)*T2(0,J-1,I)
          X(N3+I,J)=A*U(3,I)*T1(0,J-1,I)
          X(N3+I,N1+J)=-4.D0*EM**4*G1/P*T1(1,J-1,I)
          X(N3+I,N2+J)=A*(U(0,I)-C)*T2(0,J-1,I)
    3     X(N3+I,N3+J)=-G1/P*(T2(2,J-1,I)-AS*T2(0,J-1,I))
      DO 4 I=1,N1
          X(I,1)=A*(FI(2,I)-AS*FI(0,I))
          X(I,N1+1)=-(FR(4,I)-2.D0*AS*FR(2,I)+AQ*FR(0,I))+FR(1,N1+I)
          X(N1+I,1)=-A*(FR(2,I)-AS*FR(0,I))
          X(N1+I,N1+1)=-(FI(4,I)-2.D0*AS*FI(2,I)+AQ*FI(0,I))+FI(1,N1+I)
          X(N2+I,1)=A*FI(0,N1+I)
          X(N2+I,N1+1)=-1.D0/P*(FR(2,N1+I)-AS*FR(0,N1+I))
     *                 -4.D0*EM**4/P*FR(1,I)
          X(N3+I,1)=-A*FR(0,N1+I)
          X(N3+I,N1+1)=-1.D0/P*(FI(2,N1+I)-AS*FI(0,N1+I))
     *                 -4.D0*EM**4/P*FI(1,I)
    4 CONTINUE
        CALL LUDCMP(X,N4,INDX,D)
        CALL LUBKSB(X,N4,INDX,Y)
      DO 5 I=1,N1
          OF(I)=OF(I)+CMPLX(Y(I),Y(N1+I))
    5     OF(N1+I)=OF(N1+I)+CMPLX(Y(N2+I),Y(N3+I))
          OF(1)=1.D0
          C=C+Y(1)
          G1=G1+Y(N1+1)
          ERR=0.D0
      DO 6 I=1,N4
    6     ERR=ERR+ABS(Y(I))
        WRITE(*,600) G,C,ERR
        IF(ERR.GT.EPS) GO TO 10
   20     G=1.D0/G1
        WRITE(*,610) A,G,C
          OEIG=C
  600 FORMAT(15X,1P3E15.6)
  610 FORMAT(1P3E20.12)
      RETURN
      END
C
C------ INPUT DATA ------
      SUBROUTINE PARA(IR,IC,N,EM,P,G,A)
      IMPLICIT REAL*8(A-H,P-Z),COMPLEX*16(O)
        IF(IR.EQ.1) THEN
        WRITE(*,*) 'IC,N,EM,P,G,A (IC=0:GN ; IC=1:GC)'
        READ(*,*) IC,N,EM,P,G,A
        WRITE(*,600) EM
```

```
        ELSE
        IF(IR.EQ.2) THEN
        WRITE(*,*) 'A'
        READ(*,*) A
        ELSE
        READ(*,*) P
        ENDIF
        ENDIF
  600 FORMAT(' EM=',1PE15.8)
      RETURN
      END
C
C------ CALLING THE QR METHOD IN EACH HOST COMPUTER ------
      SUBROUTINE QR(OC1,OC2,OEIG,N)
      PARAMETER (IP1=142,IP2=IP1/2)
      IMPLICIT REAL*8(A-H,P-Z),COMPLEX*16(O)
      DIMENSION OC1(IP1,IP1),OC2(IP1,IP1)
      DIMENSION OEIG(IP1),OW(IP1)
      DIMENSION ILL(IP1)
C--- CHANGE THE FOLLOWING 4 LINES TO THE APPROPRIATE ONE FOR
C    EACH COMPUTER
        CALL DCLU(OC2,IP1,N,0.0,ILL,IS,OW,ICON)
        CALL DCLUIV(OC2,IP1,N,ILL,ICON)
        CALL MTRXPR(OC2,OC1,N)
        CALL DCEIG2(OC2,IP1,N,1,OEIG,OC1,OW,ILL,ICON)
      RETURN
      END
C
C------ COEFFICIENT MATRICES ------
      SUBROUTINE MTRXC(N,EM,P,G,A,T1,T2,U,OC1,OC2)
      PARAMETER (IP1=142,IP2=IP1/2)
      IMPLICIT REAL*8(A-H,P-Z),COMPLEX*16(O)
      DIMENSION OC1(IP1,IP1),OC2(IP1,IP1)
      DIMENSION T1(0:4,0:IP2,IP2),T2(0:4,0:IP2,IP2),U(0:3,IP2)
          N2=2*(N+1)
      DO 1 I=1,N2
      DO 1 J=1,N2
          OC1(I,J)=0.D0
    1     OC2(I,J)=0.D0
          OI=(0.D0,1.D0)
      DO 2 I=1,N+1
      DO 2 J=1,N+1
          OC1(I,J)=OI*A*U(0,I)*(T1(2,J-1,I)-A**2*T1(0,J-1,I))
     *            -OI*A*U(2,I)*T1(0,J-1,I)-1.D0/G
     *            *(T1(4,J-1,I)-2.D0*A**2*T1(2,J-1,I)
     *            +A**4*T1(0,J-1,I))
          OC1(I,N+1+J)=1.D0/G*T2(1,J-1,I)
          OC1(N+1+I,J)=OI*A*U(3,I)*T1(0,J-1,I)-4.D0*EM**4/(P*G)
     *                *T1(1,J-1,I)
          OC1(N+1+I,N+1+J)=OI*A*U(0,I)*T2(0,J-1,I)
     *                    -1.D0/(G*P)*(T2(2,J-1,I)-A**2*T2(0,J-1,I))
          OC2(I,J)=OI*A*(T1(2,J-1,I)-A**2*T1(0,J-1,I))
          OC2(N+1+I,N+1+J)=OI*A*T2(0,J-1,I)
    2 CONTINUE
      RETURN
      END
C
C------ NORMALIZATION OF THE EIGENFUNCTIONS ------
      SUBROUTINE NORMAL(OF)
      PARAMETER (IP1=142,IP2=IP1/2)
      IMPLICIT REAL*8(A-H,P-Z),COMPLEX*16(O)
      DIMENSION OF(IP1)
      DO 2 I=1,IP1
    2    OF(I)=OF(I)/OF(1)
      RETURN
      END
C
C------ REARRANGEMENT OF THE EIGENVALUES ------
```

```
      SUBROUTINE MODE(N,OEIG,OZ,OF)
      PARAMETER (IP1=142,IP2=IP1/2)
      IMPLICIT REAL*8(A-H,P-Z),COMPLEX*16(O)
      DIMENSION OEIG(IP1),OF(IP1),OZ(IP1,IP1)
      DIMENSION IM(IP1)
      DO 1 I=1,IP1
    1     OF(I)=OEIG(I)
      DO 2 I=1,N
          EMAX=-1.E+10
          IM(I)=0.D0
      DO 3 J=1,N
          JCON=0
      DO 4 K=1,I-1
    4   IF(J.EQ.IM(K)) JCON=1
        IF(JCON.EQ.1) GO TO 3
        IF(IMAG(OF(J)).GT.EMAX) THEN
          EMAX=IMAG(OF(J))
          IM(I)=J
        ENDIF
    3 CONTINUE
    2     OEIG(I)=OF(IM(I))
      DO 5 I=1,IP1
    5     OF(I)=OZ(I,IM(1))
      RETURN
      END
C
C------ EXPANDING FUNCTIONS AND BASIC FLOW PROFILES ------
      SUBROUTINE EFUNC(N,EM,T1,T2,U)
      PARAMETER (IP1=142,IP2=IP1/2)
      IMPLICIT REAL*8(A-H,P-Z),COMPLEX*16(O)
      DIMENSION T1(0:4,0:IP2,IP2),T2(0:4,0:IP2,IP2)
      DIMENSION S1(0:4,0:IP2),S2(0:4,0:IP2),U(0:3,IP2)
          Q1=1.D0
          PAI=2.D0*ASIN(Q1)
          P1=PAI/FLOAT(N+2)
      DO 1 I=1,N+1
          X=COS(FLOAT(I)*P1)
        CALL VFLOW(EM,X,UU0,UU1,UU2,TT0,TT1,TT2)
          U(0,I)=UU0
          U(1,I)=UU1
          U(2,I)=UU2
          U(3,I)=TT1
        CALL CHEB1(N,1,X,S1)
        CALL CHEB1(N,2,X,S2)
      DO 1 K=0,4
      DO 1 J=0,N
          T1(K,J,I)=S1(K,J)
    1     T2(K,J,I)=S2(K,J)
      RETURN
      END
C
C------ INCLUSION OF THE BOUNDARY CONDITIONS ------
      SUBROUTINE CHEB1(N,IC,X,S)
      PARAMETER (IP1=142,IP2=IP1/2)
      IMPLICIT REAL*8(A-H,P-Z),COMPLEX*16(O)
      DIMENSION T(0:4,0:IP2),S(0:4,0:IP2)
        IF(IC.EQ.1) THEN
          F0=(1.D0-X**2)**2
          F1=-4.D0*X+4.D0*X**3
          F2=-4.D0+12.D0*X**2
          F3=24.D0*X
          F4=24.D0
        ELSE
          F0=1.D0-X**2
          F1=-2.D0*X
          F2=-2.D0
          F3=0.D0
          F4=0.D0
```

```
        ENDIF
        CALL CHEBY(N,X,T)
      DO 1 I=0,N
          S(0,I)=F0*T(0,I)
          S(1,I)=F1*T(0,I)+F0*T(1,I)
          S(2,I)=F2*T(0,I)+2.D0*F1*T(1,I)+F0*T(2,I)
          S(3,I)=F3*T(0,I)+3.D0*F2*T(1,I)+3.D0*F1*T(2,I)+F0*T(3,I)
          S(4,I)=F4*T(0,I)+4.D0*F3*T(1,I)+6.D0*F2*T(2,I)
     *          +4.D0*F1*T(3,I)+F0*T(4,I)
    1 CONTINUE
      RETURN
      END
C
C------ CHEBYSHEV POLYNOMIALS ------
      SUBROUTINE CHEBY(N,Y,T)
      PARAMETER (IP1=142,IP2=IP1/2)
      IMPLICIT REAL*8(A-H,P-Z),COMPLEX*16(O)
      DIMENSION T(0:4,0:IP2)
      DO 1 I=0,4
      DO 1 J=0,1
    1     T(I,J)=0.D0
          T(0,0)=1.D0
          T(0,1)=Y
          T(1,1)=1.D0
      DO 2 J=2,N
          T(0,J)=2.D0*Y*T(0,J-1)-T(0,J-2)
          T(1,J)=2.D0*Y*T(1,J-1)-T(1,J-2)+2.D0*T(0,J-1)
          T(2,J)=2.D0*Y*T(2,J-1)-T(2,J-2)+4.D0*T(1,J-1)
          T(3,J)=2.D0*Y*T(3,J-1)-T(3,J-2)+6.D0*T(2,J-1)
    2     T(4,J)=2.D0*Y*T(4,J-1)-T(4,J-2)+8.D0*T(3,J-1)
      RETURN
      END
C
C------ PRODUCT OF MATRICES ------
      SUBROUTINE MTRXPR(OC1,OC2,N)
      PARAMETER (IP1=142,IP2=IP1/2)
      IMPLICIT REAL*8(A-H,P-Z),COMPLEX*16(O)
      DIMENSION OC1(IP1,IP1),OC2(IP1,IP1),OC(IP1,IP1)
      DO 1 I=1,N
      DO 1 J=1,N
          OC(I,J)=0.D0
      DO 1 K=1,N
    1     OC(I,J)=OC(I,J)+OC1(I,K)*OC2(K,J)
      DO 2 I=1,N
      DO 2 J=1,N
    2     OC1(I,J)=OC(I,J)
      RETURN
      END
C
C------ LU DECOMPOSITION FROM 'NUMERICAL RECIPES' [6] ------
      SUBROUTINE LUDCMP(A,N,INDX,D)
      PARAMETER (IP1=142,IP2=IP1/2,IP0=2*IP1,IP=2*IP0,TINY=1.0D-40)
      IMPLICIT REAL*8(A-H,P-Z),COMPLEX*16(O)
      DIMENSION A(IP,IP),INDX(IP),VV(IP)
      D=1.
      DO 12 I=1,N
        AAMAX=0.
        DO 11 J=1,N
          IF (ABS(A(I,J)).GT.AAMAX) AAMAX=ABS(A(I,J))
11      CONTINUE
        IF (AAMAX.EQ.0.) PAUSE 'SINGULAR MATRIX.'
        VV(I)=1./AAMAX
12    CONTINUE
      DO 19 J=1,N
        IF (J.GT.1) THEN
          DO 14 I=1,J-1
            SUM=A(I,J)
            IF (I.GT.1)THEN
```

```
            DO 13 K=1,I-1
              SUM=SUM-A(I,K)*A(K,J)
13          CONTINUE
            A(I,J)=SUM
          ENDIF
14      CONTINUE
      ENDIF
      AAMAX=0.
      DO 16 I=J,N
        SUM=A(I,J)
        IF (J.GT.1)THEN
          DO 15 K=1,J-1
            SUM=SUM-A(I,K)*A(K,J)
15        CONTINUE
          A(I,J)=SUM
        ENDIF
        DUM=VV(I)*ABS(SUM)
        IF (DUM.GE.AAMAX) THEN
          IMAX=I
          AAMAX=DUM
        ENDIF
16    CONTINUE
      IF (J.NE.IMAX)THEN
        DO 17 K=1,N
          DUM=A(IMAX,K)
          A(IMAX,K)=A(J,K)
          A(J,K)=DUM
17      CONTINUE
        D=-D
        VV(IMAX)=VV(J)
      ENDIF
      INDX(J)=IMAX
      IF(J.NE.N)THEN
        IF(A(J,J).EQ.0.)A(J,J)=TINY
        DUM=1./A(J,J)
        DO 18 I=J+1,N
          A(I,J)=A(I,J)*DUM
18      CONTINUE
      ENDIF
19  CONTINUE
    IF(A(N,N).EQ.0.)A(N,N)=TINY
    RETURN
    END
C
C------ BACK SUBSTITUTION FROM 'NUMERICAL RECIPES'[6] ------
    SUBROUTINE LUBKSB(A,N,INDX,B)
    PARAMETER (IP1=142,IP2=IP1/2,IP0=2*IP1,IP=2*IP0)
    IMPLICIT REAL*8(A-H,P-Z),COMPLEX*16(O)
    DIMENSION A(IP,IP),INDX(IP),B(IP)
    II=0
    DO 12 I=1,N
      LL=INDX(I)
      SUM=B(LL)
      B(LL)=B(I)
      IF (II.NE.0)THEN
        DO 11 J=II,I-1
          SUM=SUM-A(I,J)*B(J)
11      CONTINUE
      ELSE IF (SUM.NE.0.) THEN
        II=I
      ENDIF
      B(I)=SUM
12  CONTINUE
    DO 14 I=N,1,-1
      SUM=B(I)
      IF(I.LT.N)THEN
        DO 13 J=I+1,N
```

```
            SUM=SUM-A(I,J)*B(J)
13        CONTINUE
        ENDIF
        B(I)=SUM/A(I,I)
14    CONTINUE
      RETURN
      END
C
C------ U(X) & T(Y) IN NON-UNIFORMLY HEATED CASE -----
      SUBROUTINE VFLOW(EM,X,U0,U1,U2,T0,T1,T2)
      IMPLICIT REAL*8(A-H,M,P-Z),COMPLEX*16(O)
          EMP=EXP(EM)
          EMM=1.D0/EXP(EM)
          CM=COS(EM)
          SM=SIN(EM)
          OI=(0.D0,1.D0)
          OD=EMP*(CM+OI*SM)-EMM*(CM-OI*SM)
          OF=EXP(EM*X)*(COS(EM*X)+OI*SIN(EM*X))
     *      -(COS(EM*X)-OI*SIN(EM*X))/EXP(EM*X)
          OF1=EXP(EM*X)*(COS(EM*X)+OI*SIN(EM*X))
     *      +(COS(EM*X)-OI*SIN(EM*X))/EXP(EM*X)
          OF0=OF/OD
          OF1=(1.D0+OI)*EM*OF1/OD
          OF2=(1.D0+OI)**2*EM**2*OF/OD
          U0=-IMAG(OF0)/(2.D0*EM**2)
          U1=-IMAG(OF1)/(2.D0*EM**2)
          U2=-IMAG(OF2)/(2.D0*EM**2)
          T0=DBLE(OF0)
          T1=DBLE(OF1)
          T2=DBLE(OF2)
      RETURN
      END
```