JOURNAL OF COMPUTATIONAL PHYSICS 15, 98–115 (1974)

Automation of the Derivation of Dispersion Relations.* I. Cold Plasma Case

BERNARD ROSEN

Department of Physics, Stevens Institute of Technology, Hoboken, New Jersey 07030

Received June 26, 1973

We report here on progress in the automation of the derivation of dispersion relations in plasma physics using symbolic manipulation by computer. The techniques developed to date should have application to a wider class of problems, viz., modal analysis of linearized partial differential equations in various branches of physics. The analytic approach can be used exclusively by minor changes in programming but we have found it convenient to use symbolic manipulation as an aid to automatic programming for dealing with physical problems.

The programs include: VECTORS which converts vector equations into scalar ones, ORDERS which performs perturbation expansions and elementary Fourier analysis, DETERM which calculates secular determinants, and PARAM which performs matrix augmentation and truncation required in the analysis of parametric instabilities.

Two examples are given. In the first the instabilities of relativistic E-layers are investigated and in the second we look at the instabilities of an electron beam in a spatially varying magnetic field.

PL/I-FORMAC was the programming medium used.

1. INTRODUCTION

Our knowledge of the behavior of the plasmas is based to a great extent on the results of analyses of the linearized differential equations governing the fields and the charged fluids comprising plasmas. Among the several processes involved in such analyses are:

(a) linearization about equilibrium configurations usually with (implicit) resolution of vectors into components,

(b) Fourier (or more generally, modal) analysis,

(c) calculation of the secular determinant of the resulting linear, algebraic equations,

(d) treatment of mode-coupled equations including truncation.

* This work supported by the National Science Foundation Grant No. GJ-32260.

In this paper we shall present a description of four computer programs that carry out these operations *analytically* and thus relieve the physicist from much of the tedious routine work involved in the derivation of dispersion relations (secular equations). In general the appropriate combination of programs is run in succession, the output of the first becoming part of the input for the second, etc. Each of the programs, all of which are written in FORMAC-PL/1, [1–2] is useful in its own right; together they can carry out the processes mentioned above for problems involving "cold" plasmas. In more mathematical terms the restriction is one to equations in which the *independent* variables are the space–time coordinates. In physical terms it means that a fluid description rather than a full kinetic treatment is used. This level of description is very useful within its domain of validity. A more complete kinetic description given by use of the Vlasov equation is treated in the next paper in this series. (We shall refer to this as (II).)

In the next section we shall briefly describe the operation of each program and in the following two sections we shall illustrate their use by two examples. In the first we derive the secular equation for azimuthal density fluctuations in a thin, relativistic *E*-layer [3]. Secondly, and in more detail, we investigate the instability of an electron beam subject to a magnetic field of constant magnitude whose direction varies sinusoidally in space [4]. These two problems arose during discussions with colleagues as interesting in their own right. During the period of this work the *E*-layer problem has been treated elsewhere [5]. Both problems involve electromagnetic instabilities; the first involves cylindrical geometry and the second exhibits features of the so-called parametric instabilities. In conclusion then there is reason to believe that these problems are in some sense representative.

2. Description of Computer Programs

We shall discuss four programs which, together with their functions, are:

VECTORS—converts vector quantities and scalar quantities formed from vectors into component form,

ORDERS—carries out perturbation expansion and conversion to algebraic form by Fourier or modal analysis,

DETERM—calculates the secular determinant corresponding to the set of linear algebraic equations generated by ORDERS

PARAM—first it augments the equations for the dependent variables at the nominal frequency and wavenumber (ω, k) by those equations governing the modes to which these variables are coupled by virtue of the temporal and/or spatial variations of the zeroth order fields and then it truncates the matrix of coefficients appropriately.

In this section we present a brief description of these programs; a further discussion of the first three is found in Appendix A, and of the program PARAM in Section 4. The operation of the first three routines is illustrated in this section by their application to a well known problem, viz. propagation of electromagnetic waves in an electron plasma containing a uniform magnetic field [6]. The inputs and outputs are summarized in Table I. The equilibrium electron space charge is neutralized by an immobile positive background. The relevent equations are

(1) the equation of motion of the electrons

$$\frac{\partial \vec{v}}{\partial t} + (\vec{v} \cdot \nabla)\,\vec{v} = -\frac{e}{m}(E + \vec{v} \times \vec{B});$$

(2) Poisson's equation

$$\nabla \cdot \hat{E} = 4\pi e(n_0 - n);$$

(3) the Maxwell equations

$$\frac{\partial \vec{E}}{\partial t} = c\nabla \times \vec{B} - 4\pi\vec{j},$$

$$\frac{\partial B}{\partial t} = -c\nabla \times \vec{E}.$$

We have used $\vec{v}$ for the electron (fluid) velocity, n is the electron density, $\vec{E}$ and $\vec{B}$ are the usual electric and magnetic-induction field strengths respectively, e is the magnitude of the electron charge, m is the mass of the electron and c is the speed of light. The current density $\vec{j}$ is given by $-\,ne\vec{v}$. Note that there are ten dynamic variables, viz. the three components each of $\vec{v}$, $\vec{E}$, and $\vec{B}$ and the density for which we have written down the equivalent of ten scalar equations.

In general the wave vector of the electromagnetic waves in our example may have one component along the direction of the field (x-direction) and one component perpendicular (y-direction) so that all dynamical variables are independent of the third coordinate.

A. The program VECTORS [7, 8] recognizes the standard operations of vector analysis—DOT, CROSS, GRAD, CURL, and DIV as well as the operation involved in the construction $(\vec{A} \cdot \nabla)\,\vec{B}$, ($\vec{A}$, $\vec{B}$ vectors). We call this ADVECT. Terms involving these operations are converted into components for a given orthogonal coordinate system in terms of the scale factors. In the case of Cartesian, cylindrical, or spherical coordinates the scale factors are replaced by their values. In other cases the scale factors can be left unspecified or can be given explicit values as the user desires.

TABLE I

Program	Example of input and/or output	Comment
VECTORS	NUMBER-OF-VECTORS = 3, NUMBER-OF-VARIABLES = 1, NSPECIAL = 1, ...	Control information NSPECIAL is the number of special relations to be read in and used NUMBER-OF-VARIABLES is the number of scalars
	'NE'	Name of the scalar variable
	'V' 'E' 'B'	Names of the vectors
	'(J, −N*V)'	Definition of current density
Input	'INDLIST = (X, Y, NULL, T)'	Indicates dependence of fields on *x*, *y*, *t* but not on *z*. Thus all derivatives with respect to *z* are zero
	'DRVTV. (V, T) + ADVECT. (V, V) + E + CROSS. (V, B)'	Equation of motion
	'DIV. (E) − (NE-N0)' 'CURL. (B) − DRVTV. (E, T) − J'	Three of Maxwell's equations in input form
	'CURL. (E) + DRVTV. (B, T)'	The equality to zero is assumed
	⋮	
	'EQUATION (7) = DRV(E3.(X, Y, NULL, T), $(2)) + DRV(B1. (X, Y, NULL, T), $(4))'	
Output	'EQUATION(8) = −DRV(E3. (X, Y, NULL, T), $(1)) + DRV(B2. (X, Y, NULL, T), $(4)) 'EQUATION(9) = DRV(E2. (X, Y, NULL, T), $(1))-DRV(E1. (X, Y, NULL, T), $(2)) +DRV.(B3.(X, Y, NULL, T), $(4))'	Output based on the fourth equation above
	⋮	

ORDERS	ORDER-OF-EXPANSION=1, NUMBER-OF-FIELDS=0, NUMBER-OF-COORDINATES=4,......	Control information
	'E1', 1, 'E2', 1, 'NE', 0,... etc.	Names of dependent variables followed by the lowest order in its expansion
	'EXP(#I*(W.(N)($(4) − T) − KX.(N)($(1) − X) − KY. ($(1) − Y)))' '(KX, KY, NULL, W)'	Used in Fourier analysis
Input	EQUATIONS(1-10) FROM VECTORS 'FIXEDLST = (NULL)'	Zero order density and magnetic field are constants
	⋮	
	'TERM(1, 6, 1) = −#I*K.(1)*B3.(KX, KY, NULL, W)'	Third index indicates order of term
Output	'TERM(2, 6, 1) = −N0.(NULL)*V2.(KX, KY, NULL, W)'	Second index indicates equation number First index is the number of the term
	⋮	
	C(6, 10) = −#I*K.(1)	Coefficient matrix of linearized equations
DETERM Input	NROW = 10; Coefficient Matrix from ORDERS	Dimension of secular matrix
	UNKNOWN = 'K'	Name to be considered variable in polynomial dispersion relation
Output	FORTRAN — coded deck	

The method of conversion is programmed in a recursive manner so that any number of the operations listed above can be nested (in principle). Thus the form: CURL. CURL. (CROSS. (A, B)) is quite acceptable input. Prefix notation is used for the binary operations CROSS, DOT and ADVECT.

The program can be used to provide symbolic output necessary for further formula manipulation as it is here or to provide part of a FORTRAN program for numerical evaluation of the components of the vector equation.

B. ORDERS [9] performs the perturbation expansion of the equations governing the physical situation as well as the conversion from differential to modal form. It is an extension of a program written to carry out orderings on a set of equations. It does so by forming an expansion of each dependent variable in a power series in the ordering parameter, substituting such series into the differential equations and then collecting terms. Provisions exist for the specification of the ordering of parameters occurring in the equations as well as for the ordering of the space-time dependence of external fields present in the equations.

Modal analysis capability is provided in that one can specify the behavior of the perturbative field amplitudes on the coordinates as in

$$f(x, t) \rightarrow \tilde{f}(k, \omega) \exp[-i(\omega t - kx)].$$

Terms that involve products of the dependent variables and/or external fields are replaced by convolution-type terms in which a summation convention is understood to be in effect. Products are recognized by the fact that their logarithmic derivatives are sums.

The perturbation calculation is by no means restricted to first order. The output of the program is the set of differential equations obeyed by lower order equilibria and the transformed equations obeyed by the perturbation amplitudes. The lower order equilibrium equations are *not* solved by these programs, and, in fact, since they are often nonlinear, no effort has been made to *automate* their use in higher-order equations. (In (II) we shall discuss this replacement problem in more detail.) As a side effect of orders the matrix of coefficients of each of the perturbative amplitudes in the several equations is also provided.

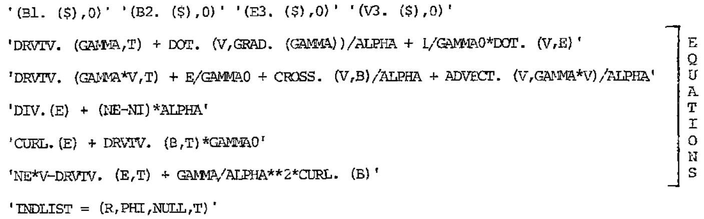
C. The routine DETERM calculates the determinant of a square matrix using the Laplace expansion. When used in conjunction with ORDERS it will produce a term by term expansion of the secular equation (a polynomial in these cases). (For plane wave behavior of the perturbed fields usually employed this is a dispersion relation.) The output terms are segregated as to the power of ω occurring in the terms. If desired the program will also produce part of a FORTRAN-coded program for the numerical evaluation of the coefficients of the (polynomial) secular equation which can be used with a standard root-finding program. In dealing with determinants of large order the analytic form of the secular determinant often is not useful; DETERM is probably most useful when the order is between 3 and 8.

D. The PARAM routine generates equations satisfied by the mode amplitudes at $(\omega + m\omega_0, k + nk_0)$ from those at (ω, k) when m and n range over integer values as designated by input data. It is necessary to consider these mode amplitudes when dealing with mode-coupling problems. This is simply accomplished by means of the FORMAC REPLACE function which replaces a specified quantity, say $W.(1)$ by another such as $(W.(1) + W0)$, in a given expression. In order to deal with a finite matrix of coefficients one must truncate the infinite chain of mode coupled equations. Removal of mode-amplitudes which lie beyond the range of truncation from the equations (for example, the mode amplitude $\tilde{n}(\omega + (m + 1)\,\omega_0)$ which might be in the equation for $\tilde{B}(\omega + m\omega_0))$ is done automatically by including FORMAC STEP functions in appropriate terms. (The STEP function is unity for a user-specified domain and is zero elsewhere.)

The implied convolutions which appear in the input equations were handled on a somewhat ad hoc basis. First, the replacements $\omega_3 \to \omega_1 - \omega_2$ and $k_3 \to k_1 - k_2$ were made. Some of the Fourier components had (Dirac) delta function behavior and hence the remaining convolutions quickly collapsed to a few terms.

3. Application to Relativistic *E*-Layer

We shall give an application of these programs to the calculation of the instability of a thin relativistic *E*-layer against azimuthal perturbations in the electron density. This calculation serves as an example and we make no claim to completeness or to originality. The equations governing the model are shown (Fig. 1) in the form in which they are "fed" into the computer. The first two are the relativistic equations of motion of the electron fluid. The ion motion is ignored. The quantity α is $\omega_p R/c$



```
COORDINATE-SYSTEM='CYLINDRICAL
'NI' 'NE' 'GAMMA'          'V' 'B' 'E'
'(B1. ($),0)' '(B2. ($),0)' '(E3. ($),0)' '(V3. ($),0)'
'DRVTV. (GAMMA,T) + DOT. (V,GRAD. (GAMMA))/ALPHA + 1/GAMMA0*DOT. (V,E)'
'DRVTV. (GAMMA*V,T) + E/GAMMA0 + CROSS. (V,B)/ALPHA + ADVECT. (V,GAMMA*V)/ALPHA'
'DIV.(E) + (NE-NI)*ALPHA'
'CURL.(E) + DRVTV. (B,T)*GAMMA0'
'NE*V-DRVTV. (E,T) + GAMMA/ALPHA**2*CURL. (B)'
'INDLIST = (R,PHI,NULL,T)'
```

Fig. 1. Equations forming part of input to VECTORS.

where ω_p is the electron plasma frequency and R is the equilibrium radius of the layer; γ_0 is the equilibrium value of γ. The next three equations are Maxwell equations; the div(B) equation is satisfied identically in the model.

As can be seen in the figure we have specified a cylindrical coordinate system (r, ϕ, z). The line of input just above the equations states that the field components (B_r, B_ϕ, E_z, and v_z) are zero; the line after the equations states that the field quantities are to be functions of r, ϕ, and t but not z.

The input to ORDERS is the output from VECTORS together with additional information, some of which is shown in Fig. 2. The first line there states that E_ϕ is to be taken as being independent of r and the following line states that the ion density, n_i, is to be an unperturbed function of the radius alone. Following this we have given the assumed functional form of the perturbative amplitudes which is that the (ϕ, t) dependence is given by $\exp[i(\omega t - \mu\phi)]$ while the radial dependence is linear through the layer (with the exception of E_ϕ). The functional form of the radial dependence assumes that the perturbed variable vary linearly across the layer, whose thickness is a, starting from zero value at the inner radius of the layer.

```
'(E2. ($), E2. (NULL,$(2),$(3),$(4)))'

'(NI. ($),NIO. ($(1)))'

'FIXEDLST = ($(1))'

"EXP (#I*(W.(N)* ($(4)-T) - MU.(N)* ($(2) - PHI)))*(1 + ($(1) - R)*R#OVER#A;'

'GAMMA',0  'NE', 0  'E2',1  'V1',1  'V2',0  'E1,0  'B3',0
```

FIG. 2. Essential input into ORDERS.

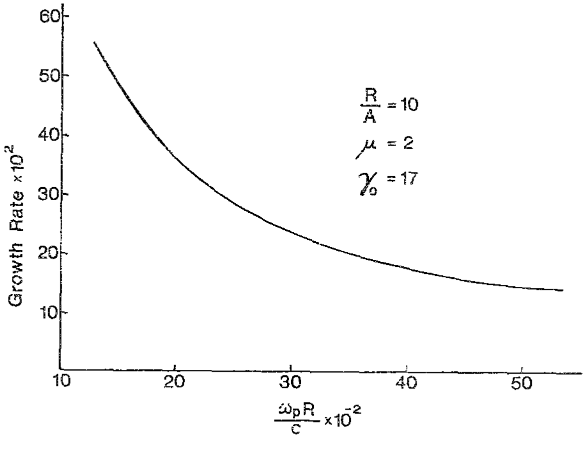


FIG. 3. Growth rate for azimuthal density instability.

R#OVER#A denotes R/a. Finally the last line states the name of the dependent variables. The significance of the numbers following each such variable is that a 0 denotes that the variable has a nonvanishing equilibrium value while a 1 signifies that the variable possesses only a perturbative (in this case, first order) amplitude.

Figure 3 shows numerical results generated by the program that was in turn produced by the action of the three programs we have described here. We have plotted the growth rate of the fastest growing instability for the parameters indicated.

4. Instability of an Electron Beam in a Direction Varying Transverse Magnetic Field

The problem we consider here is the (electromagnetic) instabilities of an electron beam which is moving with velocity V_B transverse to a "spiral staircase" magnetic field given by

$$B^{(0)}(x) \times B0[\cos(k_0 x)\, \hat{e}_y + \sin(k_0 x)\, \hat{e}_z],$$

where the $\hat{e}$'s are unit vectors while $B0$ and k_0 are scalars. (If a vector representing the direction of the field were to be drawn from each point on the axis the locus of the tips would be a helix.) The zeroth order velocity of the beam is

$$V^{(0)}(x) = \hat{e}_x V_B - \frac{\omega_c}{k_0} (B^{(0)}/B0),$$

where ω_c is the cyclotron frequency corresponding to $B0$. $V^{(0)}$ satisfies

$$V^{(0)} \cdot \nabla V^{(0)} = -\frac{e}{m} (V^{(0)} \times B^{(0)}).$$

In this treatment $V_B > \omega/k_0$ by one order of magnitude so that $V_x^{(0)}$ is always greater than zero.

In this case we used the equation of continuity rather than Poisson's equation (the conservation of charge plus the Ampere–Maxwell law imply Poisson's equation). This was done so that each of the equations is of "dynamical" form, i.e., of the form

$$\frac{\partial}{\partial t} \text{(dynamical variable)} = \text{terms without time derivatives.}$$

[redacted]

$$-i\frac{\partial}{\partial t}F_i = \sum_j M_{ij}F_j\,,$$

where the matrix M depends on the parameters and on the wavenumber. The eigenvalues of M are then the possible frequencies of the system and the instability we seek is the eigenvalue with the largest imaginary part. By reducing the problem to an eigenvalue problem we obviate the necessity of first calculating a determinant and then solving an algebraic equation. Furthermore we find the eigenvectors simultaneously and so obtain the mode structure.

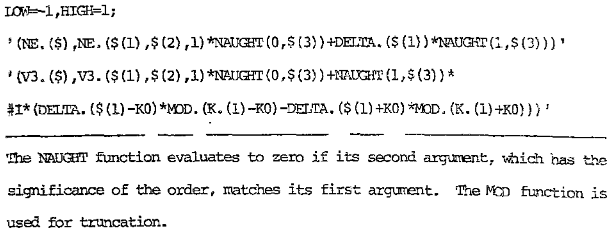
It might be thought to be advantageous to eliminate several of the variables so that the size of the algebraic system is reduced. We found that the resulting denominators in expressions led to difficulties in obtaining a simple polynomial dispersion relation. Furthermore the result of such operations lack the great simplicity and directness of the matrix eigenvalue method.

The input to VECTORS includes the equations of continuity, Newton's equation of motion, Faraday's law, and the Ampere–Maxwell law together with the specification $B_x \equiv 0$. Furthermore the fact that perturbative quantities are functions of x and t alone are also part of the input data.

Following the operation of the VECTORS routine ORDERS performed the requisite linearization together with the substitution

$$f(x, t) \rightarrow \tilde{f}(k, \omega)\exp[-i(\omega t - kx)]$$

for all first-order quantities. Nine nonzero scalar equations result; in some of these the fields (n, v, E, B) at (ω, k) are coupled to fields at $(\omega, k \pm k_0)$ by virtue of the functional form of B_0 and v_0. The equations generated by ORDERS together with the type of information indicated in Fig. 4 are the input to PARAM. The delta functions act as Dirac delta functions and so the (implied) convolution integrals can be calculated by simple algebra. Equations for the mode amplitudes at



```
LOW=-1,HIGH=1;
'(NE.($),NE.($(1),$(2),1)*NAUGHT(0,$(3))+DELTA.($(1))*NAUGHT(1,$(3)))'
'(V3.($),V3.($(1),$(2),1)*NAUGHT(0,$(3))+NAUGHT(1,$(3))*
#I*(DELTA.($(1)-K0)*MOD.(K.(1)-K0)-DELTA.($(1)+K0)*MOD.(K.(1)+K0)))'
```

The NAUGHT function evaluates to zero if its second argument, which has the significance of the order, matches its first argument. The MOD function is used for truncation.

FIG. 4. Examples of input to PARAM.

$(\omega, k + nk_0)$, LOW<n<HIGH, are formed and the MOD functions, which are ultimately replaced by combinations of FORMAC STEP functions (see above), indicated in Fig. 4 are used to truncate the equations appropriately.

The output of PARAM was the nonzero elements of the sparse matrix of coefficients of the field variables. In this particular case the matrix was of dimension 27. The eigenvalues [10] of the secular matrix were determined numerically and we plotted that value with the largest positive imaginary part in Fig. 5. Inspection of the corresponding eigenvectors revealed that the most unstable mode is predominantly a right-hand circularly polarized electromagnetic mode at wavenumber k coupled to an electrostatic oscillation at wavenumber $k + k_0$ with almost no coupling to the mode at $(k - k_0)$. The results of this calculation which was essentially programmed from equations to numerical results has provided some insight into the mode structure so that we can now use a simplified analytic approach.

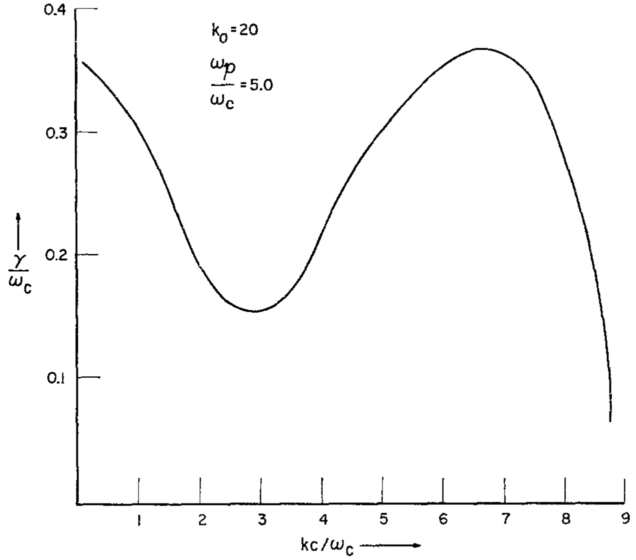


FIG. 5. Growth rate of beam instability.

We assume then that the mode at wavenumber k can be described in terms of a vector potential A and a transverse velocity only. Similarly the mode amplitude at $k + k_0$ is assumed to be well described in terms of a density fluctuation, a fluctuating velocity parallel to the x-axis and an electric field derivable from an electrostatic potential.

In terms of the variables:

$$T = \omega_c t, \qquad X = \omega_c x/c,$$
$$A = e\bar{A}./mc^2, \qquad v = \bar{v}/c,$$
$$B = \bar{B}/B_0\,, \qquad E = \frac{e\bar{E}}{mc\omega_c}\,, \qquad n = \bar{n}/n_0\,, \qquad k_0 = \frac{\bar{k}_0 c}{\omega_c}$$

(in which the barred quantities are the usual ones), we obtain the following results: For the mode amplitudes at k

The equation of motion indicates that $V = A$;
Faraday's law is satisfied identically;
the Ampere–Maxwell law can be written as:

$$\nabla^2 A - \frac{\partial^2 A}{\partial T^2} - \alpha A = \alpha V_{\perp_0} n^+$$

where

$$\alpha \equiv \frac{4\pi n_0 e^2}{m\omega_c{}^2} \equiv \omega_p{}^2/\omega_c{}^2,$$

the subscript $\perp$ indicates perpendicularity to the x-axis and the superscript $+$ indicates that the quantity is evaluated at $k + k_0$. At $k + k_0$ one has: the continuity equation can be written as

$$\frac{dn^+}{dt} = -\frac{\partial V^+}{\partial X}\,,$$

where $d/dt \equiv \partial/\partial T + V_B \partial/\partial X$;
combining Coulomb's law with the Ampere–Maxwell law gives

$$\frac{dE^+}{dt} = \alpha V^+;$$

the equation of motion for the x-component of the velocity is

$$\frac{dV^+}{dT} = -E^+ - V_{\perp_0} \times B - V_\perp \times B_{\perp_0}\,.$$

If we decompose A as

$$A = (e_y + ie_z)\, A_R + (e_y - ie_z)\, A_L$$

and note that

$$B_{\perp_0} = \tfrac{1}{2} e^{ik_0 x}(e_y - ie_z) + \tfrac{1}{2} e^{-ik_0 x}(e_y + ie_z),$$

we discover that within the approximation considered here the right-handed part of A, viz. A_R, is coupled to the electrostatic oscillation at $k + k_0$. The reason for this is as follows: $V_{\perp_0} = -B_{\perp_0}/k_0$ and therefore the nonlinear terms in the equation of motion for the longitudinal velocity can be written as

$$-\frac{(k+k_0)}{2k_0} A \times \hat{e}_L \sim A_R .$$

The density variations at $k + k_0$ contribute to the right-handed current at k because of the zeroth order transverse velocity.

We find the dispersion relation to be

$$(\omega^2 - k^2 - \alpha)((\omega - (k + k_0) V_B)^2 - \alpha) - \frac{\alpha}{2k_0^2} (k + k_0)^2 = 0.$$

The root of this quartic equation (for all values of k_0/ω_c shown in Fig. 5) with the largest positive imaginary part is numerically indistinguishable from the corresponding eigenvalue of the 27 × 27 matrix. Thus, the simplified analytic approach, which was suggested by the numerical results, gives an excellent approximation of the growth rate of this instability within the parameter range (e.g. large k_0).

5. Conclusion

We have discussed the automation of several analytic steps that occur in the derivation of dispersion relations in plasma physics and have exhibited some applications of the techniques that have been developed to date. Our present feeling is that the method is most useful for the automatic programming of the numerical solution of these relations but is by no means restricted to this mode of operation. Indeed the production of a FORTRAN coded subprogram in which the coefficients occurring in the (polynomial) dispersion relation are expressed in terms of the various parameters is an analytic result which has been especially prepared for rapid numerical processing; one is free to try his analytic ability in obtaining a factored form of the dispersion relation.

Appendix A. Description of Program Algorithms

1. *General Comments*

a. FORMAC [1] syntax permits the use of a quantity called an unspecified function, i.e., analogous to the ordinary mathematical statement $z = f(x, y)$ there is the FORMAC statement

$$\text{LET}\,(Z = F \cdot (X, Y)).$$

The number of arguments permitted is, in principle, arbitrary and furthermore the number of arguments attached to a given function name (such as F above) is variable. One can consider that Z is a list (X, Y) of type F. In FORMAC syntax F is the ARG(1, Z), $X =$ ARG(2, Z) and $Y =$ ARG(3, Z). In general the elements of the list can be any construction acceptable on the right hand side of a LET statement.

b. FORMAC variables are global since they are held on a list of so-called REPWORDS within the interpreter. In order to do recursive programming we have found it convenient to introduce an extra index into those variables which should be made local; this index is increased by unity each time a recursive routine is entered and similarly decreased upon exiting from that routine. If the recursive subroutines also use iteration, then some care must be given to the values of iteration indices which should be local *PL*/1 variables but which may have global FORMAC equivalents.

2. *Vectors*

VECTORS, after processing control information such as the number of scalar variables, the type of coordinate system, etc. accepts both scalar and vector equations in that order. Since vectors can occur within divergences or scalar products within a scalar equation *all* equations are scanned for the presence of 'DOT.' and 'DIV.'. If either is found each instance is converted to its expanded form in terms of components. Scalar equations are then converted to "ordinary" form; vector equations are further processed.

By ordinary form we mean that:

(a) Spatial derivatives of vectors are represented in standard FORMAC syntax if symbolic output is desired and by finite differences if FORTRAN output is desired.

[redacted]

The vector equations are handled as follows:

(a) The tensorial form (ith component) of the equation is generated with all summations explicitly carried out.

(b) The explicit components are then calculated for $i = 1$ through the number of dimensions specified.

(c) The equations are converted to ordinary form.

The tensorial form is generated by the action of two recursive routines V$ for vectors and S$ for scalars. V$ searches for the occurrence of 'CROSS.', 'CURL.', 'ADVECT.' and 'GRAD.'. For each such occurrence it passes the elements of the accompanying list to the appropriate function for further processing (V$ for the two "arguments" of CROSS AND ADVECT and for the single element following CURL and S$ for the single argument of GRAD). Similarly S$ searches for 'DOT.' and 'DIV.'. The recursion ends when none of these special names are encountered. The derivatives implied by the CURL, DIV, and GRAD operations are indicated symbolically by use of another unspecified function called DIFFER. In conversion to ordinary form DIFFER is replaced by DERIV for FORMAC output and by finite differences of its first argument for FORTRAN-type output. Processing is done term by term so that routines are iterative as well as recursive. The cyclic and anticyclic permutation of the indices is indicated internally by the replacements $I \to RP.(I)$, $I \to RM.(I)$ a la Symbolic ALGOL II. This our method of delaying evaluation which in ALGOL is effected by use of the call-by-name feature. The internal representation of a vector up to this point is of the form NAME. (INDEX) wherein NAME is the name of the vector and INDEX is I with an arbitrary number (including zero) of RP's and RM's acting upon it.

The generation of components now follows simply by letting I run from 1 through the number of coordinates. The conversion of (nested) RP and RM "function calls" is performed by repeated FORMAC replacements using the replacement CHAIN: (RM. (1), 3, RM. (2), 1, RM. (3), 2, RP. (1), 2, RP. (2), 3, RP. (3), 1).

3. *Orders*

a. Fourier "transformation"

This program can do simple Fourier transforms of differential equations in the sense that replacements of the form: (we denote the function and its transform by the same symbol since no confusion can arise)

$$f(x, t) \to f(k_1, \omega_1) \exp[i(k_1 x - \omega_1 t)] \to f(k_1, \omega_1),$$

$$\frac{\partial f}{\partial t} \to -i\omega f(k_1, \omega) \exp[i(k_1 x - \omega_1 t)] \to -i\omega f(k, \omega), \quad \text{etc.}$$

This change is made in a term containing one such factor to be transformed by using the FORMAC statements:

```
Let (TERM = F. (X, T);

        N = 1;
        NEWLIST = (K(N), W(N));
    TERM = EVAL (TERM, F. ($), F. (NEW LIST)*
            EXP ( #I*((K(N)* ($(1) - X) - W(N)* ($(2) - T))));
```

The last statement is obviously the key one and it is remarkable that this single FORMAC statement by itself performs the replacements we have indicated. The program is written so that NEWLIST and the exponential factor are supplied as data.

During the replacement process terms are handled individually; if this term contains more than one factor which depends on time then we apply the EVAL statement above to each factor depending on time separately with the convention that N now starts at 2 and increases by unity for each such factor. Thus

F. (X, T) + G. (X, T)* H. (X, T) → F. (K(1), W(1)) + G. (K(2), W(2))*
H. (K(3), W(3)).

There is the implication that integrals and/or summations over all k's and ω's with index > 1 are to be carried out and that these summations are subject to the conditions that

$$k(2) + k(3) + \cdots = k(1),\ \omega(2) + \omega(3) + \cdots = \omega(1) \text{ etc.}$$

b. Ordering

Let us denote by ϵ (called # # # # # # in the program) a dimensionless parameter that is supposed to be of the first order of smallness. We define a given ordering of a differential equation(s) in the following manner:

i. each parameter occurring in the equation(s) has its order of smallness defined by the (integer) power of ϵ which its value approximates,

ii. each dependent variable occurring has an expansion in ϵ beginning at some power.

As an example, assume that the equations, dependent variable names, etc. have been read in by the program. For example suppose that we have the following case

$$\frac{\partial f}{\partial t} + \alpha f \frac{\partial g}{\partial x} = 0,$$

$$\frac{\partial g}{\partial t} + \beta f = 0,$$

and furthermore that $\alpha \sim \epsilon$, $\beta \sim \epsilon^2$, $f = \epsilon f_1 + \epsilon^2 f_2 + \cdots$; $g = g_0 + \epsilon g_1 + \cdots$ Also the maximum order we seek is given and is, say, four. The program

(i) forms the sums

$$f = \sum_{n=1}^{4} \epsilon^n f_{(n)}, \qquad g = \sum_{n=0}^{4} \epsilon^n g_{(n)},$$

(ii) replaces α by $\alpha\epsilon$ and β by $\beta\epsilon^2$ in the equations,

(iii) substitute the sums indicated in (i) into the equations, and

(iv) isolates the various orders in ϵ by forming the appropriate derivatives with respect to ϵ evaluated at $\epsilon = 0$. The author has found this method more reliable in general than the use of the COEFF function.

If the FOURIER mode of this program is to be used then the quantities of *lower* order than the order which is being transformed do not necessarily have the full space–time dependence. For example, if we were looking at plasma oscillations the zeroth order density is a constant. This alternate dependence can be specified by the quantity FIXEDLST whose value is a data item. There is only one FIXEDLST; should this prove too restrictive there is an alternate way to introduce quantities, namely through FIELDS which behave like dependent variables except that their exact dependence and expansions must be specified.

IV. *Determ*

As stated in the main text the Laplace expansion is used for calculating determinants. The programming style is recursive with trails ending abruptly if a zero entry is encountered. Each term in the Laplace expansion is written out in FORTRAN code separately. The terms are segregated according to the power of a variable; this variable has a default name of 'W' but it can be changed by the input data to any other name such as 'K' so that the resulting FORTRAN code can be used to solve for $k(\omega)$ rather than for $\omega(k)$.

In order to increase the efficiency of the resulting FORTRAN program a unique name (a FORTRAN variable such as $G231$) is created for each element of the secular determinant. The elements of the determinants are written out in terms of these new FORTRAN variables so that the terms in the dispersion relation require the evaluation of the matrix elements per se only once per iteration.

Acknowledgments

The author is grateful to Professor George Schmidt for introducing him to the E-layer problem and to Professor John Dawson and Dr. Anthony Lin for submitting the beam problem to him. Furthermore, the encouragement offered by Professor Dawson and the continuing hospitality shown by Princeton Plasma Physics Laboratory have been essential ingredients in the completion of this work.

Use was made of computer facilities supported in part by the National Science Foundation at PPL.

The author gratefully acknowledges the support of this work by the NSF under Grant No. GJ-32260.

References

1. J. F. Sammet, Formula manipulation by computer, *in* "Advances in Computers" (F. Alt and M. Rubinoff, Eds.), Vol. 8, Academic Press, New York, 1967.
2. R. Tobey et al., PL/1-FORMAC Interpreter Users Reference Manual, SHARE Contributed Program Library (SHARE 360.D-03.3.004).
3. The immediate point of departure for our considerations was the research note by G. Schmidt, Magnetic compression of astron-type *E*-layers, *Phys. Fluids* **15** (1972), 1540.
4. K. Papadopoulos, private communication.
5. S. N. Mahajan, Ph.D. Thesis, Univ. of Maryland, 1973 (unpublished). For earlier work see K. D. Marx, *Phys. Fluids* **11** (1968), 357.
6. W. P. Allis, S. J. Buchsbaum, and A. Bers, "Waves in Anisotropic Plasmas," M.I.T. Press, Cambridge, MA, 1963.
7. Our approach should be contrasted with that of J. C. Howard and H. Tashjian, An algorithm for deriving the equations of mathematical physics by symbolic manipulation, *Comm. ACM* **11** (1968), 814.
8. M. Petravic, G. Kuo-Petravic, and K. V. Roberts, Automatic Optimization of Symbolic Algol Programs. Culham Report CLM-P 274.
9. B. Rosen and N. K. Winsor, Analytic Orderings Generated by Computer, MATT Report-850, Princeton PPL, Princeton Univ. Princeton, NJ (unpublished).
10. Subroutine ALLMAT by J. Rinzl and R. E. Funderlic, Union Carbide Computing Technology Center Numerical Analysis Library, was used.

Printed in Belgium