# The Dynamics of Interacting Nonlinearities Governing Long Wavelength Driftwave Turbulence

by

DAVID E. NEWMAN

A thesis submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

(PHYSICS)

at the

UNIVERSITY OF WISCONSIN-MADISON

1993

# The Dynamics of Interacting Nonlinearities Governing Long Wavelength Drift Wave Turbulence

David E. Newman

Under the supervision of Associate Professor Paul W. Terry

at the University of Wisconsin - Madison

## Abstract

Because of the ubiquitous nature of turbulence and the vast array of different systems which have turbulent solutions, the study of turbulence is an area of active research. Much present day understanding of turbulence is rooted in the well established properties of homogeneous Navier-Stokes turbulence, which, due to its relative simplicity, allows for approximate analytic solutions. This work examines a group of turbulent systems with marked differences from Navier-Stokes turbulence, and attempts to quantify some of their properties. This group of systems represents a variety of drift wave fluctuations believed to be of fundamental importance in laboratory fusion devices. From extensive simulation of simple local fluid models of long wavelength drift wave turbulence in tokamaks, a reasonably complete picture of the basic properties of spectral transfer and saturation has emerged. These studies indicate that many conventional notions concerning directions of cascades, locality and isotropy of transfer, frequencies of fluctuations, and stationarity of saturation are not valid for moderate to long wavelengths ($k\rho_s \leq 1$). In particular, spectral energy transfer at long wavelengths is dominated by the E×B nonlinearity, which carries energy to short scale (even in 2-D) in a manner that is highly nonlocal and anisotropic. In marked contrast to the canonical self-similar cascade dynamics of Kolmogorov, energy is efficiently passed between modes separated by the entire spectrum range in a correlation time. At short wavelengths, transfer is dominated by the polarization drift nonlinearity. While the standard dual cascade applies in this subrange, it is found that finite spectrum size can produce cascades that are reverse directed (i.e., energy to high k) and are nonconservative in enstrophy and energy similarity ranges (but conservative overall). In regions where both nonlinearities are important, cross-coupling between the nonlinearities gives rise to large nonlinear frequency shifts as well as changes in the spectral dynamics. This profoundly affects the dynamics of saturation by modifying the growth rate and nonlinear transfer rates. These modifications can produce a nonstationary saturated state with large amplitude, long period relaxation oscillations in the energy, spectrum shape, and transport rates.

Acknowledgments

I would like to express my deep appreciation to a number of people for their help in getting to this point in my education. First of all I would like to thank my parents Sally and Ted Newman for exposing me to the beauties of science and encouraging me to pursue and explore these wonders. Next I would like to thank my officemates Chris Watts, Jim Hollenberg, Matt Stoneking, Gene Riggs, Brett Chapman, Earl Scime and Eduardo Fernandez with whom I have enjoyed many discussions on physics as well as life. I believe that the setup of our office with a mixture of experimentalists and theorists is ideal and should not be changed. Thanks also go to Andrew Ware for many valuable answers to last minute questions.

The person who deserves the most credit for the incredible amount of fun that I have had doing research over the last few years is Prof. Paul Terry. I truly can not imagine a better advisor nor a better working relationship. I give him my deepest thanks and fear that all future working relationships will pale in comparison.

I must also thank Geeta Bhatt for help with the typing of this thesis when, predictably, the time got short. In addition, I could not have made it to this point without my boys, Katmai and Kodiak, quite literally standing by me.

Finally my deepest thanks go to my wife and best friend Uma Bhatt whose support has been invaluable and with whom I have been fortunate enough to enjoy daily valuable scientific discussions in addition to life together.

This work was supported by USDOE grant DE-FG02-89ER53291



Table of Contents

List of Figures

Figure page

# Chapter 1
# Introduction

## 1.1 General

The study of turbulence is primarily the study of the dynamics of the nonlinearities governing the turbulence. This area of study has a long and distinguished history spanning over one hundred years of experimentation, analysis and modeling. Despite successes based on the scaling analysis begun with Reynolds and continued by Kolmogorov[1] and others: on analytic closures, physical modeling, computation, and detailed observation, there is still much in turbulence that remains a mystery. Studying the dynamics of multiple turbulent nonlinearities, their interplay with each other and with sources and sinks in their domain, yields a wide variety of fascinating results. Some results and methods of approaching problems are found to be inexplicably general in their applicability while other accepted properties of turbulent systems are found to be limited in their validity.

As a starting place in investigating turbulence it is valuable to use as simple a system as possible in order to fully understand the dynamics without additional complications. One step in this simplification is the use of 2 dimensional models. The validity of such models tends to be much greater than one might expect in a 3 dimensional world. Many situations exist which can cause the reduction of a system from 3-D to 2-D[2]. In neutral fluids, rotation causes a separation of 2 and 3 D dynamics for imbedded turbulence with the correct spatial dimensions and energy. This allows much of the large scale ocean and atmospheric dynamics to be well modeled by 2-D quasi-geostropic models[3]. Stratification can also cause a division of the two types of dynamics with all turbulent fluctuations having a characteristic scale length ($l_e$) greater than the stratification scale ($l_s$) being governed by 2-D dynamics within the plane of stratification. Similarly, most surface physics is explicitly 2-D and as



such is governed by 2-D dynamics. The introduction of a magnetic field makes the dynamics of charged particles along the field intrinsically different from the dynamics perpendicular to the field. This often allows the perpendicular dynamics to be treated as 2-D with little loss of precision. The research described here is primarily focused on plasma turbulence whose 2 dimensional nature is due to the magnetic field which is perpendicular to the plane of the dynamics. Nevertheless much of the insight gained from these studies should be applicable to other (non-plasma) systems.

In addition to serving as a paradigm for a variety of turbulent systems, the models described here strongly suggest the existence of a class of interesting dynamics in many systems that may often be overlooked. This class of dynamics comes from the direct interaction between multiple nonlinearities. A common feature of turbulent systems is the existence of multiple nonlinearities of differing order. The ordering could be based on amplitude, as in a quadratic nonlinearity vs. a cubic nonlinearity, or on scales, as in the nonlinearity having three derivatives vs. one having four derivatives, etc. In general the system is simplified by ignoring the subdominant nonlinearity. While this may be valid in some cases, and is usually a reasonable first step, it may also be ignoring important dynamics in the cross-coupling. These cross-coupling dynamics may not be as subdominant as the dynamics of the neglected nonlinearity and as such may be important to the overall system dynamics.

One of the most cherished concepts in turbulence research is the inertial range. This is a region in k-space where the nonlinear term ($v\cdot\nabla v$ for Navier-stokes turbulence) is much larger then any linear terms ($\nu\nabla^2 v$ for Navier-stokes). Using the existence of such a region the cascade dynamics can be studied both analytically and computationally[4]. Much of the analytic work also assumes that the sources and sinks of energy (i.e., the stirring and viscous damping) are separated by an inertial range of infinite extent. While these concepts are useful for the purpose of getting results, they can sometimes be misleading. In general

if an inertial range exists it is limited in extent and has sources and sinks arranged in a fashion that is less than optimal from the perspective of a theorist. In any physically realistic model the energy will go from the source to the sink irrespective of the preferred dynamics of the nonlinearity governing the cascade. For example a system can be constructed with the characteristic that the nonlinearity prefers to transfer energy to high k from low k. This system can also be arranged so the source is at high k and the sink is at low k. If the system is physically realistic a steep gradient in energy will be set up from source to sink with much more energy in the high k driven modes than in the low k damped modes. The system will, however, still saturate by transferring the energy down the gradient to the sink. The saturation level is likely to be higher then in the opposite case with driving at low k, damping at high k and a preferred nonlinear transfer direction being to high k.

In many systems, including most turbulent plasma models, in addition to the relatively small region (in k space) in which the model is valid, the sources and sinks are distributed. This leads to a small or non-existent true inertial range, adding another level of complexity in the path of understanding the dynamics of the system.

### 1.2 Fusion/Plasma

In the pursuit of fusion as a viable source of energy a number of hurdles must be crossed. One of the most serious of these is the problem of confinement of energy. In order for a self-sustaining nuclear fusion reaction to occur the triple product of density (n), temperature (T), and energy confinement time ($\tau$), $nT\tau$, must be sufficiently large[5]. This means the density must get large enough, the temperature must get high enough, and the energy confinement time must be long enough for an acceptable value of the triple product. Preventing the extension of the confinement time are a number of processes which can be roughly broken down by scale size. At the smallest size scale is classical molecular (or atomic) diffusion. Classical diffusion is a very slow process even at the densities and temperature in the atmosphere. In a typical fusion device such processes would not have a



great effect on the confinement time. At the large scale end of the spectrum are global disruptions. These are usually catastrophic magnetic events which essentially dump all or most of the energy onto the containment shell in a very short amount of time. At the time of a disruption $\tau$ goes to near zero implying no confinement and complete suppression of any fusion reactions. As n and T get larger the stored energy increases and these major disruptions become increasingly serious due to the potential for damage to the shell. Fortunately, regimes of operation have been found in which these disruptions do not seem to occur, thus, removing the problem. This leaves the intermediate scales as the source of the anomalous transport which degrades the confinement[6]. This transport probably does not have the same source everywhere in the device. At some radial locations the transport maybe due to electrostatic fluctuations while at other places it may be due to interacting magnetic islands[7]. Which of the processes is the limiting one is not yet clear and it may be that they all work in concert. Nevertheless it does seem clear that fluctuation driven turbulent transport is important in the system and must be understood.

Large scale (long wavelength) turbulence is an area of intense current research from both the experimental and theoretical perspectives. This research has been particularly focused on the core fluctuations as these fluctuations are thought to be responsible for the transport of energy from the hot core to the outer regions of the plasma. One of the models considered as a viable candidate for the core fluctuations is the dissipative trapped ion convective cell turbulence (DTICC) model. In the late 70's this model was considered[8] and rejected when the catastrophic Bohm-like transport that had been predicted was not realized when successful operation of the Princeton Large Torus in the 4 kev range was achieved. More recent theoretical work[9] suggests that this mode should not give catastrophic transport and is therefore a valid candidate for core fluctuations. Much recent experimental work including beam emission spectroscopy[10] and correlation reflectometry[11] give an indication that long wavelength fluctuations do exist in the core and may be important to transport.

In addition to being useful as a possible candidate to explain core fluctuations, the DTICC model is useful as a starting paradigm for drift wave turbulence. Within drift wave turbulence models two of the most prevalent nonlinearities, and therefore two of the most important, are the E×B and polarization drift nonlinearities. Because DTICC turbulence is easily represented by a 2-D one-field fluid model (Kadomtsev-Pogutse[12]), it readily allows the exploration of the dynamics of the E×B nonlinearity. With straightforward extension the polarization drift nonlinearity (like that in the Hasagawa-Mima equation[13]) can be added. This extends the model to be generic enough in its nonlinear behavior so as to cover most drift wave turbulence. As an example, the Terry-Horton equation[14] has both the E×B and the polarization drift nonlinearities in a one-field model for trapped electron and universal modes. These models easily allow the investigation of the different aspects of these nonlinearities and their interactions. Furthermore they may provide insight into the behavior of turbulent nonlinearities in general.

### 1.3 Summary of Results

This work consists of a series of studies concentrating on understanding the dynamics of two of the nonlinearities which govern drift wave turbulence. With a progression of models each nonlinearity is investigated individually then they are combined to allow the study of interacting nonlinearities. Finally the models are extended to two fields to allow the interplay between the nonlinearities to feed back on the system. This allow the cross-coupling to have a direct effect on the linear growth and transport. The four models are each really just an extension of the previous ones. The first is a one-field Dissipative Trapped Ion Convective Cell (DTICC) model with just the E×B nonlinearity. Next is a model used to study the polarization drift nonlinearity (or the Navier-Stokes nonlinearity). This model is the one-field Hasagawa - Mima (H-M) model. The model combining the two



nonlinearities is the Dissipative Trapped Electron mode (DTEM1) model. This is then extended to a two-field Dissipative Trapped Electron mode (DTEM2) model to allow the self-consistent feedback. In all of these models a variety of diagnostics are used in a range of different simulations to probe the dynamics of the systems. The diagnostics used include time histories of total energy, flux, and enstrophy, time histories of individual mode energy enstrophy, and flux as well as frequency and k spectrum evolutions. A diagnostic was developed to track the nonlinear transfer of energy and enstrophy between modes (or bands of modes). This has proved to be invaluable in deciphering the dynamics over the full range of k space. In the different models a sequence of types of simulations are performed. Some simulations are run with no driving or damping to allow the nonlinearities to relax the system, thereby simulating pure inertial range transfer. Other cases have driven/damped systems run to saturation in which some set of modes are undriven and undamped and which therefore comprise an inertial range. In another set of cases the driven/damped system is allowed to reach saturation at which time a localized energy pulse is inserted. The pulse is allowed to relax and the relaxation is followed to discern the dynamics involved. Finally, scans of parameter space are performed in order to investigate any anomalous behavior. A summary of the main results follows.

The basis for the Kolmogorov spectrum is the assumption that inertial ranges are dominated by a local self-similar transfer or cascade. Using a combination of the Kolmogorov spectrum and equilibrium statistical mechanics it is possible to infer the preferred direction of flow for a given nonlinearity. The dynamics of most turbulent systems are governed by nonlinearities which are themselves isotropic so in the absence of anisotropic driving, damping or geometry the turbulence is also expected to be isotropic. The DTICC model is used to study the dynamics of an anisotropic nonstandard nonlinearity. It is found that this nonlinearity transfers energy nonlocally from low ky to high ky. This leads to a saturated spectrum much flatter in the ky direction than that predicted by

Kolmogorov theory. This should not be surprising as the lack of a local transfer induced similarity range precludes the use of Kolmogorov theory.

Going hand in hand with the concept of local transfer in an inertial range is the idea of conservative cascades. The nonlinearity of the DTICC model (E×B nonlinearity), $\nabla(\partial n/\partial y) \times z \cdot \nabla n$, has only one quadratic invariant (energy) and consequently has only a direct cascade. The H-M model is used to study the dynamics of the polarization drift nonlinearity ( $\nabla \phi \times z \cdot \nabla \nabla^2 \phi$) which has 2 quadratic invariants, energy and enstrophy. These 2 invariants give rise to the same dual cascade that is familiar from 2-D Navier-Stokes turbulence. In this cascade energy cascades conservatively to low k and enstrophy cascades conservatively to high k. The early work of Kraichnan (1964) on these cascades used the infinite extent of the cascade to show that energy is the only quantity cascaded in the low k region and enstrophy is the only quantity cascaded in the high k region. It is found here that if the spectrum is not infinite in extent there is some "improper" transfer. In the energy transfer range some enstrophy is transferred and in the enstrophy transfer range some energy is transferred. The amount of "improper" transfer depends on both the extent of the inertial range and the position, within that range, of the source of free energy (the injection range). It is quite possible to arrange a situation in which more energy or enstrophy flows in the improper direction then in the proper direction. This is important to keep in mind when inferring dynamical properties from the conservation laws of the system.

Nonlinearities are by definition not additive, yet it is assumed that if 2 nonlinearities have different regions of dominance in k space, in the region dominated by one nonlinearity the other nonlinearity can be ignored. The DTEM1 model is used to investigate the effects of two nonlinearities interacting in one system. It is found that at the extremes in which one dominates the other by more then an order of magnitude, it is justifiable to drop the sub-dominant nonlinearity. However in the often large region where the relative strength of the nonlinearities are within an order of magnitude of each other, new dynamics occur which



are not explainable by a linear super-position of the 2 nonlinearities. The "cross-coupling" dynamics are qualitatively different and include both a change in the transfer dynamics and a nonlinear frequency shift. These cross-coupling dynamics occur dominantly in the "crossover" region in which the two nonlinearities are of comparable magnitude. The frequency shift can be as large as a few times $\omega^*$ and is found to be proportional to ky. The shift observed from simulation qualitatively agrees with the shift predicted by closure theory. Such shifts could help explain frequencies observed in computation which seem anomalously shifted away from the frequency expected from linear dispersion relations. The influence of the crossover region creates 5 effective regimes: 1) at the low k extreme, transfer dynamics are governed solely by the E×B nonlinearity; 2) at k values between the crossover region and the region of E×B dominance both the E×B and the cross-coupling dynamics are important; 3) in the crossover region, the characteristic dynamics are dominated by the interaction of the two nonlinearities, these dynamics are largely independent of the E×B and polarization drift dynamics; 4) in the region above the crossover region but below the polarization drift dominated region, both the cross-coupling and the polarization drift dynamics are important; and 5) at the highest k values the polarization drift dynamics dominate. Before ignoring a subdominant nonlinearity it may be important to assure oneself that the nonlinearity of interest is sufficiently dominant so as to dominate the cross-coupling dynamics as well as the subdominant nonlinearity.

In order to self-consistently include the feedback effect of the frequency shift on linear growth and transport fluxes, it is necessary to extend the model beyond a one-field model to a two-field model. The DTEM2 model is used to study these interaction feedbacks. The most striking result from the two-field studies is the lack of a stationary saturated state. As with most driven/damped turbulent systems this system does reach a saturated state dependent on the various parameters. These parameters include growth, damping and the ratio of E×B to polarization drift nonlinearity strength but, for fully

developed turbulence, do not include initial conditions. In this system the saturated state is found to be non-stationary, instead exhibiting oscillations (cycles) with amplitude of order 50% and oscillation period on the order of 10 $\tau_e$ (eddy turnover times). During these cycles most of the relevant quantities also undergo large fluctuations. These include transfer rates and flux as well as frequency and k spectra. These cycles may be understood in terms of a simple heuristic relaxation oscillation model. It is suggestive of an intriguing possibility that this type of oscillation may in fact exist in experiment , but may effectively be "washed out" by averaging over a few cycles in the measurement process. Since these models are all local, it is also possible that spatial measurmental averaging could wash out the fluctuations. If they do exist, it might suggest that transport is a much more locally intermittent event then generally believed.

The organization of the thesis is as follows: chapter 2 contains work on the Dissipative Trapped Ion Convective Cell turbulence model. This work focuses on the novel features of the E×B nonlinearity. Chapter 3 contains analytical and computational results dealing with the effect of finite spectrum size on the polarization drift nonlinearity. This is presented in the context of a Hasagawa - Mima type model which is nearly isomorphic to the Navier - Stokes equation. Chapter 4 deals with the one - field Dissipative Trapped Electron mode turbulence model. This model is used to investigate the interaction between the two nonlinearities (E×B and polarization drift). In chapter 5 the dissipative trapped electron mode model is extended to two fields in order to explore the feedback of the nonlinear cross-coupling. Brief conclusions are presented in chapter 6 followed by Appendix A, which contains some information on the numerics and a commented listing of the main part of the two-field codes. Finally, in Appendix B some calculations are presented on extending Kolmogorov analysis to nonlocal (non self-similar) transfer. It should be noted that each of these chapters is fairly self contained with introductions which elaborate on the brief exposition of concepts given in this introduction.



**References**


1 A. N. Kolmogorov, J. Fluid Mech., **12**, pp 82-85 (1962)
2 M. Lesieur, "Turbulence in Fluids", (Kluwer Academic Publishers, Boston, 1990)
3 J. Pedlosky, "Geophysical Fluid Dynamics", (Springer Verlag, 1979)
4 R.H. Kraichnan, Phys. Fluids **10**, 1417 (1967)
5 J. Hugill, Nucl. Fusion **25**, 331 (1983)
6 J.D. Callen, Phys. Fluids B **2**, 2869 (1990).
7 C.C. Hegna and J.D. Callen, Phys. Fluids B **4**, 1855 (1992).
8 R. Saison, H.K. Wimmel, and F. Sardei, Plasma Physics **20**, 1 (1978).
9 P.H. Diamond and H. Biglari, Phys. Rev. Lett. **65**, 2865 (1990).
10 R.J. Fonck, S.F. Paul, D.R. Roberts, Y.J. Kim, N. Bretz, D. Johnson, R. Nazikian, and G. Taylor, 18th European Conference of Controlled Fusion and Plasma Physics (European Physical Society, Vol. 15C, Part I, 1991) p. I-269.
11 P. Cripwell and A.E. Costley, 18th European Conference of Controlled Fusion and Plasma Physics (European Physical Society, Vol. 15C, Part I, 1991) p. I-17.
12 B.B. Kadomtsev and O.P. Pogutse, Reviews of Plasma Physics Vol 5, edited by M. Leontovich (Consultants Bureou, New York, 1970), p. 387.
13 A. Hasegawa and K. Mima, Phys. Rev. Lett. **39**, 205 (1977).
14 P.W. Terry and W. Horton, Phys. Fluids **25**, 491 (1982).

# Chapter 2
# Dynamics of the E × B Nonlinearity

## 2.1 Introduction

The possibility that turbulence driven by unstable trapped ion modes plays a role in core fluctuations and transport in tokamaks has generally been discounted throughout the past decade. At one time however, trapped ion modes were thought to represent a potentially serious confinement problem for auxiliary heated (multi-kilovolt) plasmas[1]. Indeed, catastrophic Bohm-like transport was predicted. This prediction was premised on a presumed inverse cascade of energy from the already long wavelengths of the unstable fluctuations, and the fact that fluctuation levels would be large, given the long radial correlation length of the turbulence. The successful operation of the Princeton Large Torus device at ion temperatures in excess of 4 Kev without catastrophic confinement problems was generally taken as an indication that trapped ion mode activity was somehow absent from hot auxiliary heated tokamak discharges. Excepting some early efforts to explain this seeming lack of trapped ion mode turbulence[2] and some work on low collisionality ion temperature gradient driven turbulence[3], trapped ion turbulence has generally been ignored.

Recent work on trapped ion mode turbulence based on approximate analytic solution of renormalized Kadomtsev-Pogutse fluid equations[4] contends that these fluctuations can not be discounted as an important component of core turbulence and may in fact comprise the low frequency large amplitude extreme of experimental spectra. Contrary to the predictions of Bohm-like transport[1], this work asserts that trapped ion convective cell turbulence drives transport which is not excessively large, but is comparable in magnitude to the transport produced by trapped electron turbulence. This contention is supported by two



facts. The first is that the turbulent radial flow associated with trapped ion convective cell turbulence is small, offsetting the large fluctuation level in the quadratic moments which determine the transport fluxes. The second is the prediction that spectral energy transfer is not characterized by an inverse cascade, but rather is directed to short wavelengths. The latter precludes catastrophic condensation of energy at the largest scales of the system.

There is mounting experimental evidence that fluctuations with large radial correlation length are present in the core of tokamak plasmas. Observed spectra from scattering diagnostics have long exhibited an increase of spectral energy toward the smallest resolved wavenumbers, typically with no turnover evident over the range of wavenumbers for which measurement is possible. Recently, new fluctuation diagnostics with the capability of providing spatially resolved local measurements of core turbulence have been developed. Both beam emission spectroscopy[5] and correlation reflectometry[6] find evidence (the former in TFTR, and the latter in JET) for fluctuations inside r/a of 0.7 with radial correlation lengths of several centimeters. In both cases, the frequency of the fluctuations is very low or nearly zero, once the rotation induced Doppler shifts are subtracted. Intriguing links with global confinement are evident. While it has not been possible to associate these fluctuations with any given model, trapped ion convective cell turbulence is clearly a candidate.

This chapter describes a numerical study of dissipative trapped ion convective cell turbulence. Motivated by the issue of the energy transfer direction in wavenumber space for trapped ion turbulence, key facets of the spectral energy transfer process are examined. In order to isolate basic physical processes and enable comparison with analytic theory, a simple 2-D single field model is utilized. This model is based on the fluid responses for trapped ions and electrons first used by Kadomtsev and Pogutse. The link between electrons and ions provided by quasineutrality enables a single field description. Rapid trapped particle bounce motion, restricting the development of parallel dynamics, provides

the rationale for a two dimensional treatment. The model incorporates the E×B nonlinearity, the dominant nonlinear transfer mechanism for long wavelength fluid plasma turbulence. Consequently, the turbulent transfer properties of dissipative trapped ion convective cell turbulence, as described by the model, apply to a broader class of long wavelength fluctuations, including those resulting from trapped electron modes.

In addition to determining the direction of energy transfer in wavenumber space, an investigation of other properties relating to the spectral transfer process is described in detail. This includes the degree to which transfer in wavenumber space is local, as implicit in Kolmogorov-type similarity arguments which envision an energy transfer through all scales at precisely the same rate, or nonlocal, and therefore at variance with the standard view of cascades. The degree of isotropy or anisotropy in the energy transfer process and in the spectrum itself is also investigated. Finally, the direction of energy transfer is examined in relation to the quadratic invariants of the nonlinearity and the equilibrium spectrum. The latter ties in with statistical mechanics calculations frequently used to infer the direction of cascades[7,8]. Clearly, these properties impact the spectrum, the turbulence level, and the magnitude of spatial transport. At a more fundamental level, they affect the basic characterization of turbulence and turbulent cascades.

Spectral transfer and its characterization in terms of cascades has long been a central part of the conceptualization of turbulence. In Navier-Stokes turbulence, for example, it is well established that the energy transfer can be represented by a self-similar cascade process. In three dimensions, this process conservatively transfers energy to small scales. In two dimensions, the invariance of an additional quantity, enstrophy, or mean square vorticity, precludes the self-similar transfer of energy to small scale. It is possible to conserve both quantities, however, if enstrophy is transferred to small scale and energy to large scale. By analogy with 2-D Navier-Stokes turbulence, it might be inferred (mistakenly, as will become apparent) that dissipative trapped ion convective cell turbulence



undergoes an inverse energy cascade (cascade to long wavelength). However, it is the simultaneous invariance of energy and enstrophy which directly underlies the dual cascade of 2-D Navier-Stokes turbulence and not the number of dimensions (except through the number of invariants).

It is often possible to infer the direction of spectral transfer from closure equations. These describe the average transfer consistent with the statistical ansatz invoked to obtain the closure. The statistical hypotheses upon which closures are predicated are, in general, very difficult to validate and are known to be violated by fluctuations which are spatially intermittent. Furthermore, a number of other approximations and simplifications typically enter into analytical results obtained from closure equations. In particular, the closure is most often applied to one-point equations. While one-point analyses simplify the determination of a saturation level, they neglect the incoherent transfer process required for energy conservation. For these reasons, other methods for inferring the spectral transfer have been developed.

The most widely used method is based on equilibrium statistical mechanics[7,8]. This method is appealing for its direct use of the dynamical invariants in obtaining equilibrium spectra of the invariant quantities. Its weakness lies in the somewhat tenuous connection between equilibrium quantities and the properties of turbulence, which generally are far from equilibrium. At the minimum, the use of an equilibrium spectrum to infer a spectral transfer direction requires a knowledge of the steady state spectrum set up under forcing and dissipation, and the assumption that nonlinear transfer in the steady state is in the direction which would tend to drive the spectrum towards its equilibrium configuration. An additional weakness arises from the possibility that additional invariants exist which constrain the transfer but are not among those known and included in the calculation of the equilibrium spectrum, thereby compromising the equilibrium prediction. Not withstanding these difficulties, equilibrium statistical mechanics correctly predicts the direction of spectral

transfer in 2-D and 3-D Navier-Stokes turbulence[7,8], and has been used in many other types of turbulence[9-10].

In the present work, nonlinear transfer in the numerical simulation of the Kadomtsev-Pogutse fluid model is directly measured in distinct regions of wavenumber space. In order to eliminate the transfer imposed by any particular wavenumber space distribution of sources and sinks in favor of the conservative transfer produced by the nonlinearity, transfer is determined for undriven/undamped turbulence starting from a finite amplitude initial state with a given spectrum. These results establish the direction of inertial transfer of energy in a steady state whose spectrum is similar to the spectrum chosen as an initial condition. The initial spectrum relaxes under the inertial transfer, producing time-asymptotically an equilibrium spectrum which can be compared with the spectrum predicted by equilibrium statistical mechanics on the basis of the known dynamical invariants. These studies therefore provide a test of the validity of the methodology of equilibrium statistical mechanics for predicting spectral transfer directions, as well as a check on the predictions of the closure theory[4].

The concept of a wavenumber cascade is usually thought of as a local process whereby energy is passed between scales which are adjacent in wavenumber space. The renormalized Kadomtsev-Pogutse equation provides for both local and nonlocal transfer of internal energy $|\tilde{n}_k|^2$ in wavenumber space. According to the renormalized equation,

$$\frac{1}{2}\frac{\partial\,|\tilde{n}_k|^2}{\partial t} - \gamma_k|\tilde{n}_k|^2 + T_k = 0, \qquad (2.1)$$

the evolution of energy in the mode k is governed, apart from linear driving and damping ($\gamma_k$), by a transfer rate $T_k$ with local and nonlocal components. Specifically, the transfer rate, as given from a standard statistical closure, is



$$T_k = \sum_{k'}|\tilde{n}_k|^2|\tilde{n}_{k'}|^2[A_{NL}(k,k') + A(k,k')] + \sum_{p+q=k}|\tilde{n}_p|^2|\tilde{n}_q|^2A(p,q), \qquad (2.2)$$

where

$$A_{NL}(k,k') = \text{const.}\ (\mathbf{k}\times\mathbf{k}'\cdot\mathbf{z})^2\, L^{-1}_{k,k'}\,(k_y'^2 - k_y^2),$$
$$A(k,k') = \text{const.}\ (\mathbf{k}\times\mathbf{k}'\cdot\mathbf{z})^2\, L^{-1}_{k,k'}\,k_y'^2,$$

are coupling coefficients, and reflection symmetry in $k_y$ has been assumed for convenience in this discussion. Aside from the E×B geometrical factor $(\mathbf{k}\times\mathbf{k}'\cdot\mathbf{z})^2$ and the nonlinear response time $L^{-1}_{k,k'}$ (to be defined in the next section), these coefficients are governed by the factors $(k_y'^2 - k_y^2)$ and $k_y'^2$. The factor in $A_{NL}$ indicates transfer which is negligible for k comparable to k', but large for disparate values, i.e., transfer which is nonlocal. Also, the sign of $A_{NL}$ is such that energy is depleted from the mode k when $k_y'^2 > k_y^2$, and deposited into that mode when $k_y'^2 < k_y^2$. This clearly implies transfer to large wavenumber. By contrast, the factor in A is positive definite, indicating transfer which is always out of (rather than into) the mode k, and independent of the relative positions of k and k' in wavenumber space. (Note that for this term, nonlocal transfer is *not* precluded.)

The strong interaction and direct energy exchange between modes of widely disparate wavenumbers suggested by $A_{NL}$ is a fundamental departure from the cascade dynamics implicit in steady state spectra formulated according to the similarity concepts of Kolmogorov. If nonlocal transfer dominates, it is possible that the steady state wavenumber spectrum in an inertial range will strongly deviate from a self-similar spectrum. It is difficult to determine precisely the relative magnitudes of local and nonlocal transfer from renormalized equations without knowing the spectrum. Moreover, nonlocal transfer may be offset by the term A, which is not restricted to purely local triads. In the simulations, nonlocal and local transfer rates are directly measured throughout the relaxation.

The dominant nonlinearity governing mode coupling at long wavelengths is the E×B nonlinearity. This nonlinearity is anisotropic with respect to the two cross-field directions. The anisotropy carries over to the closure equations, where highly anisotropic transfer, particularly nonlocal, is evident. In contrast, the equilibrium spectrum, derived from a single isotropic invariant (energy) is isotropic in the two cross-field directions. This seeming contradiction is examined in detail from the simulation results in order to determine the degree of anisotropy in the spectrum, and local and nonlocal transfer rates.

The results of this chapter are now summarized. Inertial energy transfer by trapped ion convective cell turbulence has been examined numerically for a 2-D Kadomtsev-Pogutse fluid model. Numerical solution of the equations was accomplished with a spectral code containing up to 41×41 modes. The spectrum evolution and transfer rate time histories of spectra initially peaked at low wavenumber were observed with two distinct regimes of evolution in evidence. In the first, there is a strong transfer of energy to high wavenumber occurring over several eddy turnover times and resulting in the relaxation of the spectrum to a configuration with noticeable peaking at high $k_y$ (y is the cross-field direction perpendicular to the inhomogeneity in density). Nonlocal transfer in $k_y$ plays an important role in the relaxation process and is responsible for the peaking in $k_y$. Enstrophy increases throughout this regime. In the second regime the spectrum further relaxes to an approximately equipartitioned state under the action of sloshing in wavenumber space. The final spectrum is roughly consistent with the predictions of equilibrium statistical mechanics based on a single invariant corresponding to the internal energy. A slight peaking in $k_y$ is apparent in this time-asymptotic spectrum. The peaking represents a minor deviation from the predicted equilibrium spectrum, but one which strengthens, rather than weakens, the equilibrium statistical mechanics prediction that energy transfer is to small scale. For an initial spectrum which is flat, net transfer to small scale in the $k_y$ direction is again evident,



but is weaker than that of the peaked spectrum case. This transfer is highly nonlocal, producing a spectrum which is slightly peaked in $k_y$ and similar to the spectrum of the sloshing regime when reached from a peaked initial condition. The flat initial spectrum case is important because stationary turbulence driven at long wavelengths, damped at short wavelengths, and having an inertial range in intermediate scales, results in a stationary spectrum which is only weakly peaked at low k.

Nonlocal transfer in $k_y$ derives from the direct coupling of modes which are widely separated in wavenumber space. For initial spectra with indices $\alpha$ less that 3, where $|n|^2 = k_y^{-\alpha}$, the nonlocal transfer in $k_y$ dominates local transfer, thus invalidating Kolmogorov similarity range arguments for inertial range transfer. For $\alpha > 3$, the coupling between disparate scales is strongly reduced by the large amplitude disparity and transfer is dominantly local until the spectrum has relaxed to $\alpha \sim 3$. The local and nonlocal rates are comparable in the $k_x$ direction. This represents a pronounced anisotropy in the spectral transfer of energy. Because the local transfer tends to be isotropic, anisotropies in the spectrum are less pronounced.

The remainder of this chapter is organized as follows. The basic model, its properties and the basic computational procedure are presented in Sec. II. In Sec. III, the equilibrium spectrum and similarity-range stationary spectrum are derived and the prediction for spectral transfer direction is formulated. The simulation results are detailed in Sec. IV, and conclusions are given in Sec. V.

## 2.2 Basic Equations and Computational Procedure

In order to study nonlinear processes in detail, a simple model is used. This model, based on Kadomtsev-Pogutse equations, treats dissipative trapped ion convective cell turbulence as a turbulent fluid described by a single scalar related to both the fluctuating

density and the flow stream function (electrostatic potential). This model therefore allows contact with the considerable body of knowledge existing for Navier-Stokes turbulence and similar fluid plasma models such as the Hasegawa-Mima equation[11] and its dissipative analogs[12]. At the same time, the model is sufficiently complete to capture many essential elements of trapped ion turbulence. The model equation is given by

$$\frac{\partial \tilde{n}}{\partial t} + D\frac{\partial^2 \tilde{n}}{\partial y^2} + \frac{V_D}{2}\frac{\partial \tilde{n}}{\partial y} + \nu_{eff,e}\,\tilde{n} - \frac{4L_nD}{\varepsilon^{1/2}}\nabla\frac{\partial \tilde{n}}{\partial y}\times z\cdot\nabla\,\tilde{n} = 0\,, \qquad (2.3)$$

where $\tilde{n} = \varepsilon^{1/2}n^{tr}/n_o$ is the normalized trapped ion density, $V_D = \varepsilon^{1/2}(cT_i/eB)L_n^{-1}$ is the effective diamagnetic drift velocity for trapped ions, $D = V_D^2/4\nu_{eff,e}$ is an inverse diffusivity describing the destabilization of trapped ion modes by electron collisions, $\nu_{eff,i}$ is the effective collision frequency of ion-ion collisions, $\nu_{eff,j} = \nu_j/\varepsilon$, $\varepsilon = r/R$ is the inverse aspect ratio parameterizing the fraction of trapped particles, and $L_n$ is the density gradient scale length. This equation incorporates the dynamics of both trapped electrons and trapped ions with quasineutrality providing the link between their densities. The turbulent fluid flow is the E×B flow. This flow couples to the density fluctuations through the E×B advection of the mean density gradient.

The electron dynamics incorporates adiabatic (passing) and nonadiabatic (trapped) electrons. As a result of the very low frequency of the fluctuations, the trapped electrons are collisional, i.e., trapped electrons experience multiple collisions over a fluctuation period. Consequently, electron dynamics are governed by collisional scattering as opposed to nonlinear advection. This results in a linear relation between the fluctuation source (given by the E×B advection of the average density) and the trapped electron density. Trapped electrons are therefore laminar, with a density which is proportional to the potential and 90° out of phase due to the collisions. Electron collisions access the density gradient free energy



through an inverse damping process and thus provide the basic instability which feeds the turbulence.

The combination of electron and ion densities into a single field using the laminar electron response constitutes an "iδ" approximation ($\tilde{n}_e \sim i\delta\phi$, where $\delta = 4L_nD/\varepsilon^{1/2}$ is the nonadiabatic electron response). It is worth noting that because electron inertia is negligible, the nonadiabatic electron response, or function δ, has no explicit dependence on the frequency. Thus, the need to approximate the nonadiabatic response by evaluating an explicit frequency dependence at $\omega_*$ or a linear frequency does not arise. Explicit frequency dependence in other types of drift-wave fluctuations (collisionless trapped electron modes, universal modes, etc.) represents a serious shortcoming of the iδ approximation, but one which does not occur in the present case.

The ion response is hydrodynamic and consists of E×B advection, the polarization drift, and ion-ion collisions. The latter affects the fluctuations at very long wavelengths providing a low k cutoff for the instability. The long wavelengths of the fluctuations restrict the extent to which the polarization drift plays any role in the dynamics. As a consequence, the fluctuations are essentially nondispersive in the energy containing scales fed by the instability. Similarly, the polarization drift nonlinearity ($n_o\nabla\cdot v_p^{(1)}$ where $v_p^{(1)} = B_o^{-1}(c/e)z\times v_E\cdot\nabla v_E$ and $v_E = -\nabla\phi\times z$), familiar from the Hasegawa-Mima equation, is small compared to the E×B nonlinearity ($v_E\cdot\nabla\,\tilde{n}_i$ ). The E×B nonlinearity is nonzero only through the nonadiabatic electron response and produces dissipative coupling which breaks enstrophy conservation[12]. The E×B nonlinearity dominates at long wavelengths where $k\rho < \delta$ (ρ is the ion gyro-radius at the electron temperature). The present study is concerned with the long wavelength limit, so only the E×B nonlinearity is retained in the model. Because energy transfer is toward small scales in regions dominated by the E×B nonlinearity (a principal conclusion of this chapter), energy eventually reaches regions where the polarization drift nonlinearity becomes important and other effects occur, such as

coupling to smaller wavelength trapped electron modes. Future work will address the dynamics at shorter wavelengths where the two nonlinearities participate in the transfer process.

As indicated previously, the Kadomtsev-Pogutse equation, Eq. (2.3), accurately represents many essential elements of trapped ion turbulence. These include the correct electron dissipation-induced linear growth rate of the driving instability. The linear frequency is also correctly represented as nondispersive, in the direction of the electron diamagnetic drift, and offset from the diamagnetic frequency by the trapped particle fraction. Energy transfer is appropriately governed by the E×B nonlinearity. This model does not account for parallel dynamics (and hence radial mode structure) or coupling to short wavelength fluctuations.

The computational method employed is spectral. Thus the time evolution of Fourier modes (coupled by the convolution sum of the E×B nonlinearity) is solved according to

$$\frac{\partial \tilde{n}_k}{\partial t} = Dk_y^2\tilde{n}_k - i\frac{V_D}{2}k_y\tilde{n}_k - \nu_{eff,e}\,\tilde{n}_k - \frac{4iL_nD}{\varepsilon^{1/2}}\sum_{k'}(\mathbf{k}\times\mathbf{k}'\cdot\mathbf{z})k_y'\tilde{n}_{k'}\tilde{n}_{k-k'} \,. \qquad (2.4)$$

A truncation of Fourier space limits the spatial resolution of the solution. Time advancing is accomplished using a Gear-type solver with an explicit Jacobian. The code is purely spectral, treating Eq. (2.4) as it is written, as opposed to pseudo-spectral, which uses fast Fourier transforms to calculate the convolution term in real space where the fields are a simple product. The use of a spectral method limits the resolution relative to that possible with a pseudo-spectral code; however, subtle issues involving dealiasing are thus avoided. In particular, pseudo-spectral methods require dealiasing, a process which distorts the Fourier modes in the shorter wavelength part of the spectrum. The portion of the spectrum which is usable is affected by anisotropies in the nonlinear coupling, and has subtleties



which depend on whether the spectrum represents second order moments such as energy, or higher order moments[13]. In the present work, the energy and enstrophy *transfer* rates are computed. These quantities are calculated from triplet correlations. The effect of dealiasing on third order correlations with anisotropic coupling is not well documented and for the purpose of developing such diagnostics, the spectral environment provides easier implementation and more reliable interpretation. The results presented here are obtained using a maximum of 41×41 modes. For the study of spectrum relaxation and energy transfer in turbulence which is undriven/undamped, this resolution is adequate, as verified by changing the number of modes.

A key feature of the computations reported in this chapter is the calculation of the energy and enstrophy transfer rates $T_k$ and $U_k$ in wavenumber space. Energy and enstrophy are defined as $\Sigma_k|\tilde{n}_k|^2$ and $\Sigma_k k^2|\tilde{n}_k|^2$. Thus, rates of energy and enstrophy transfer from a given mode k are obtained by multiplying the last term of Eq. (2.4) by $\tilde{n}_k^*$ and $k^2\tilde{n}_k^*$ giving respectively

$$T_k = \frac{4iL_nD}{\varepsilon^{1/2}}\,\mathrm{Im}\sum_{k'}(\mathbf{k}\times\mathbf{k}'\cdot\mathbf{z})k_y'\tilde{n}_{k'}\tilde{n}_{k-k'}\tilde{n}_k^* \,, \qquad (2.5)$$

and

$$U_k = \frac{4iL_nD}{\varepsilon^{1/2}}\,\mathrm{Im}\sum_{k'}(\mathbf{k}\times\mathbf{k}'\cdot\mathbf{z})k^2k_y'\tilde{n}_{k'}\tilde{n}_{k-k'}\tilde{n}_k^* \,, \qquad (2.6)$$

These quantities are evaluated at each step of the computation yielding instantaneous rates of transfer between the mode k and all other coupled modes in the spectrum. $T_k$ positive represents a net flow of energy into the mode k; $T_k$ negative represents net outflow. Because energy is conserved by the nonlinear transfer, the energy flow into or out of a mode must be equal to the flow out of or into all remaining modes. Enstrophy, on the other hand, is not conserved. If energy is transferred conservatively to short wavelength, as

occurs in 3-D Navier-Stokes turbulence, it follows from simple dimensional arguments that enstrophy will be generated. If energy flows to long wavelength, the converse will hold. Consequently, the evolution of total enstrophy, obtained from the time history of $\Sigma_k U_k$, provides an additional indicator of energy transfer direction and is the principal enstrophy diagnostic utilized.

The closure representation of the energy transfer rate [Eq. (2.2)] follows from Eq. (2.5) by a straight forward iteration on each of the density fluctuation factors. Consistent with quasi-Gaussian statistics, the iterated density factors, or driven fluctuations, are directly excited by a single triplet interaction, with the remaining triplets acting to nonlinearly decorrelate the interaction. The directly acting triplet is the one yielding closure, e.g.,

$$L_{k-k'}\,\tilde{n}_{k-k'} \;=\; -\frac{4iL_nD}{\varepsilon^{1/2}}\;(\mathbf{k}'\times\mathbf{k}\cdot\mathbf{z})\;(k_y + k_y')\;\tilde{n}_k\,\tilde{n}_{k'}$$

where $L_{k-k'}$ is the nonlinearly broadened propagator, consisting of linear growth, damping, diamagnetic rotation, and the nonlinear decorrelation. Carrying out this procedure yields

$$\begin{aligned} T_k \;=\; & \left(\frac{4L_nD}{\varepsilon^{1/2}}\right)^2 \sum_{\mathbf{k}'} (\mathbf{k}'\times\mathbf{k}\cdot\mathbf{z})^2\; k_y'(k_y'-k_y)\; L^{-1}_{k',k-k'}\,|\tilde{n}_k|^2|\tilde{n}_{k'}|^2 \\ + \; & \left(\frac{4L_nD}{\varepsilon^{1/2}}\right)^2 \sum_{\mathbf{k}'} (\mathbf{k}'\times\mathbf{k}\cdot\mathbf{z})^2\; (k_y'^2 - k_y^2)\; L^{-1}_{k',k-k'}\,|\tilde{n}_k|^2|\tilde{n}_{k'}|^2 \\ + \; & \left(\frac{4L_nD}{\varepsilon^{1/2}}\right)^2 \sum_{\mathbf{p}+\mathbf{q}=\mathbf{k}} (\mathbf{p}\times\mathbf{q}\cdot\mathbf{z})^2\; p_y(p_y-q_y)\; L^{-1}_{k,p,q}\,|\tilde{n}_p|^2|\tilde{n}_q|^2\,, \end{aligned} \qquad (2.7)$$

where $L^{-1}_{k',k-k'} = i\,(\omega_{k'} + \omega_{k-k'} + \omega_{-k} - \gamma_{k'} - \gamma_{k-k'} - \gamma_{-k} + \Delta\omega_{k'} + \Delta\omega_{k-k'} + \Delta\omega_{-k})$ is the triplet interaction correlation time, $\gamma_k = Dk_y^2 - \nu_{eff,e}$ is the linear growth rate, and $\Delta\omega_k = [(4L_nD)^2/\varepsilon]\;\Sigma_{k'}\,(\mathbf{k}'\times\mathbf{k}\cdot\mathbf{z})^2\,k_y'^2\,L^{-1}_{k',k-k'}\,|\tilde{n}_{k'}|^2$ is the nonlinear decorrelation rate.



The first term of Eq. (2.7) comes from the iteration of $\tilde{n}_{k-k'}$ and describes advection of density by the turbulent flow. The third term arises from the iteration of $\tilde{n}_k{}^*$ and incorporates the incoherent transfer process neglected in one-point closures[14]. The second term is obtained by iterating on $\tilde{n}_{k'}$, the fluctuation representing the turbulent flow [$\tilde{v}_{k'} \propto ik_y\tilde{n}_{k'}$]. Consequently, this term provides for the self-consistent back reaction of the density fluctuations on the turbulent flow. In closures of Vlasov-Poisson systems, this term represents the renormalization of the shielding cloud and is frequently neglected because it requires the solution of a nonlinear Poisson equation. In fluid treatments it has a similar meaning through the correspondence between flow and potential which underlies E×B motion. Likewise, it is often neglected in order to avoid a nonlinear eigenvalue problem. The nonlocal transfer produced by this term can be thought of as the interaction between large scale flow (shielding potential) and smaller scale density fluctuations. As noted in the introduction, the nonlocal transfer is unequivocally toward small scale.

In order to reduce the data required to establish the direction of energy transfer and examine other properties of the transfer process, the transfer into or from bands in wavenumber space is measured by summing Eqs. (2.5) and (2.6) over selected k values. The bands are chosen to represent slices in $k_x$ and $k_y$ in order to be sensitive to any anisotropies in the wavenumber space transfer. With the band structure, it is possible to track local energy flow in k-space through adjacent bands. Nonlocal transfer is also easily observed. The definition of local versus nonlocal transfer is somewhat difficult to quantify. Typically, nonlocal transfer is used to signify exchange of energy with modes in the discrete spectrum which are displaced by more than two or three wavenumbers, i.e., an interaction with modes which are not nearest neighbors. However, with such a definition, observed nonlocal transfer could still be compatible with self-similar transfer, provided it admitted a similarity range with a number of these nonlocal steps. In order to identify energy flow which is incompatible with a similarity range, nonlocal transfer will be defined as energy

exchange in interactions spanning more than half (or occasionally one third) of the inertial range. Local transfer will be the difference of the total transfer and the nonlocal transfer, i.e., the flow to modes which are not removed by approximately half of the k-space extent.

### 2.3 Equilibrium and Similarity Range Spectra

In this section, the prediction that trapped ion turbulence produces a transfer of energy to short wavelength[4] is reviewed. This prediction is based on a comparison of the equilibrium spectrum calculated from statistical mechanics, and the stationary spectrum describing the distribution of energy in a driven saturated state. The equilibrium spectrum is derived using the statistical probability distribution function for a canonical ensemble consistent with the known invariants of the nonlinearity. In applying these statistical tools, it is assumed that turbulent transfer in the steady state is in the direction in wavenumber space which would tend to drive the spectrum to the equilibrium configuration if the driving and damping were turned off. The testing of this assumption is one of the objectives of the present study. Clearly, application of this method requires knowledge of the stationary spectrum established by the turbulently moderated balance of driving and damping. In this chapter, the simplest stationary spectrum, consistent with a Kolmogorov-type similarity range, will be adopted as a benchmark. Because the validity of a similarity range spectrum is cast into doubt by the results of the next section, the issue of the appropriate stationary spectrum and its role in the prediction of transfer direction will be revisited at that time.

In the absence of driving and damping, a single nontrivial quadratic invariant is admitted by the Kadomtsev-Pogutse equation. This invariant is the internal energy, $\int d^2x\, \tilde{n}^2$, whose conservation follows from the fact that

$$\int \nabla \frac{\partial \tilde{n}}{\partial y} \times \mathbf{z} \cdot \nabla\, \tilde{n}\, d^2x \;=\; 0\,. \tag{2.8}$$



A second integral, $\int d^2x\, \partial \tilde{n}^2/\partial y$ , is also conserved by the nonlinearity. However, by symmetry, this integral is identically zero for all time, and therefore imposes no constraint on the cascade dynamics. Enstrophy, an invariant of the 2-D Navier-Stokes and Hasegawa-Mima equations, is not conserved by the E×B nonlinearity. The nonconservation of enstrophy arises from the nonadiabatic electrons[12].

With energy as the only nonzero quadratic invariant, the canonical probability distribution of equilibrium states is

$$P \;=\; \exp\left[ -\frac{\beta}{2} \sum_{\mathbf{k}} |\tilde{n}_{\mathbf{k}}|^2 \right] , \tag{2.9}$$

where $\beta$ is an effective inverse temperature. The spectrum is given by the expectation value of the energy in a mode $k_j$ yielding

$$|\tilde{n}_{k_j}|^2 \;=\; \frac{\int |\tilde{n}_{k_j}|^2\, P\, d\tilde{n}_{k_1} \ldots d\tilde{n}_{k_2} \ldots d\tilde{n}_{k_{max}}}{\int P\, d\tilde{n}_{k_1} \ldots d\tilde{n}_{k_2} \ldots d\tilde{n}_{k_{max}}} . \tag{2.10}$$

Equation (2.10) predicts an equipartition of the energy,

$$|\tilde{n}_{k}|^2 \;=\; \frac{1}{\beta} , \tag{2.11}$$

corresponding to a flat spectrum in the 2-D wavenumber space. In addition to equipartition, the equilibrium spectrum is isotropic, a feature which follows from the fact that the lone nonzero quadratic invariant has no explicit anisotropy.

As stated previously, a similarity range stationary spectrum is calculated for comparison with the equilibrium spectrum. Assuming that unstable modes are confined to a

localized region of wavenumber space where $\omega < \omega_{bi}$, the balance of nonlinear transfer of energy $\tilde{n}^2$ at each scale in the spectrum with the net energy input rate requires that $\varepsilon \approx k^3\tilde{n}_k^3$, where $\varepsilon$ is the fixed energy input rate and isotropy has been assumed, i.e., $\partial/\partial x = \partial/\partial y \rightarrow k$. Solving for $\tilde{n}_k^2$ and expressing in terms of an isotropic integral distribution for a single wavenumber k [ $\int E(k)\, dk \approx \tilde{n}^2$ ], the spectrum is

$$E(k) = \varepsilon^{2/3} k^{-3}. \tag{2.12}$$

If the spectrum is anisotropic ($\partial/\partial x \neq \partial/\partial y$), the balance of energy input rate with spectral transfer requires that $\varepsilon = k_y^2 k_x \tilde{n}_k^3$. Writing the spectrum as an integral distribution in two wavenumbers [ $\int E(k_x,k_y)\, dk_x dk_y = \tilde{n}^2$ ] yields

$$E(k_x,k_y) = \varepsilon^{2/3} k_x^{-5/3} k_y^{-7/3}. \tag{2.13}$$

Note that the second spectrum is consistent with the first and reduces to it when $k_x = k_y$. It has an additional power of $k^{-1}$ because it is a distribution of energy in a two-dimensional wavenumber space. In contrast to the equilibrium spectrum, both similarity range spectra are peaked at long wavelength. The anisotropy in $k_x$ and $k_y$ of Eq. (2.13) arises directly from the anisotropy of the E×B nonlinearity. These spectra follow for an inertial cascade, regardless of its direction in wavenumber space. Assuming that the nonlinear interactions of the inertial range attempt to drive the system toward equilibrium, transfer will be in such a direction as to establish the equilibrium spectrum, $E \sim \beta^{-1}$. On this basis, transfer is predicted to be toward short wavelength. If a similarity range does not exist, but the stationary spectrum of driven turbulence remains peaked at the long wavelength, the transfer is again predicted to be toward short wavelength.

The transfer of energy to short wavelength is a consequence of the existence of a single quadratic invariant. By contrast, the 2-D Navier-Stokes and Hasegawa-Mima



equations have two invariants and thus a canonical probability distribution function which depends on the two integrals. The equilibrium spectra calculated from this distribution function have enstrophy weighted toward short wavelengths and energy weighted toward long wavelengths. Consequently, enstrophy in these systems is predicted to flow to short wavelengths and energy to long wavelengths.

## 2.4 Computational Results

The results reported in this section come principally from numerical experiments in which turbulence is initialized with finite amplitude in a variety of spectral distributions and allowed to evolve with no forcing or dissipation. Initial spectra range from $k^{-1}$ to $k^{-6}$, bracketing the similarity range stationary values of $k_x^{-5/3} k_y^{-7/3}$. Evolution is tracked for as many as 100 eddy turnover times. In all cases, the initial spectra relax to an identical time-asymptotic configuration. The instantaneous flow during the spectrum evolution is measured to indicate the flow which would occur in a driven/damped stationary situation having a spectrum like the instantaneous spectrum during relaxation. The time history of energy is calculated throughout the runs as a check on the numerics. Energy is found to be conserved to better than one part in $10^8$ for tens of eddy turnover times. The nonlinear transfer diagnostics have also been benchmarked against hand calculations for a small k space in order to verify their accuracy.

The time-asymptotic spectrum is found to be flat, in very good agreement with the predicted equilibrium spectrum. This agreement is remarkable given that the nonlinearity which drives the relaxation and the measured wavenumber space flow are anisotropic; yet the final spectrum is isotropic and in good agreement with a prediction which takes no account of the anisotropy in the nonlinearity. Figures 2.1-2.3 show a sequence of spectra, starting with an initial spectrum given by $E(k) = k^{-4}$, an intermediate configuration which

occurs a few mean fluctuation time scales after the initial time, and an equipartitioned time-asymptotic spectrum corresponding to a fully relaxed state achieved tens of eddy turnover times after the initial time.

The spectra in Figs. 2.2 and 2.3 represent two distinct regimes of nonlinear transfer. The first is a flow regime characterized by robust transfer to high $k_x$ and $k_y$. The $k_y$ transfer is strongly nonlocal and is responsible for the peak and valley at high and intermediate $k_y$ values, respectively, which is evident in Fig. 2.2. The flow regime terminates 5 - 10 eddy turnover times after the beginning of nonlinear transfer when all energy in the low k modes above the equilibrium value has been transferred to high k, thus depleting the low k spectrum peak. There is a noticeable peak at high $k_y$ in the spectrum at the end of the flow regime. This is due to the relatively greater efficiency of the nonlocal $k_y$ transfer, compared to the local transfer of the $k_x$ flow.



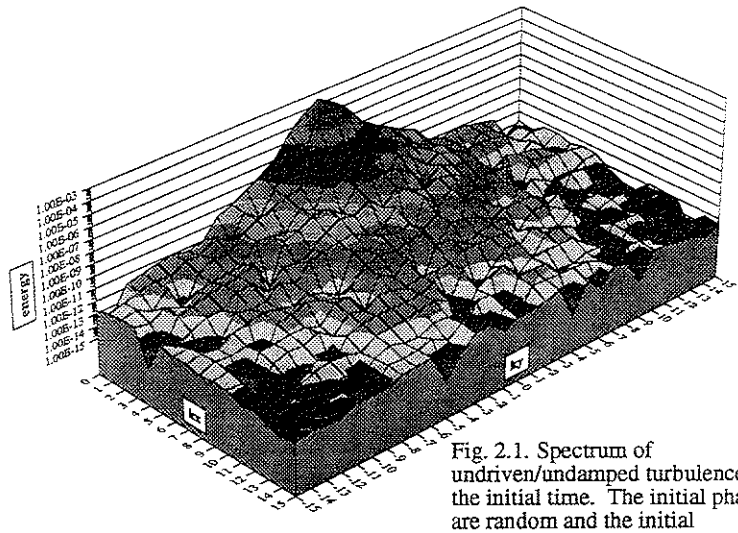


Fig. 2.1. Spectrum of undriven/undamped turbulence at the initial time. The initial phases are random and the initial spectrum fall-off index is $\alpha = -4$.

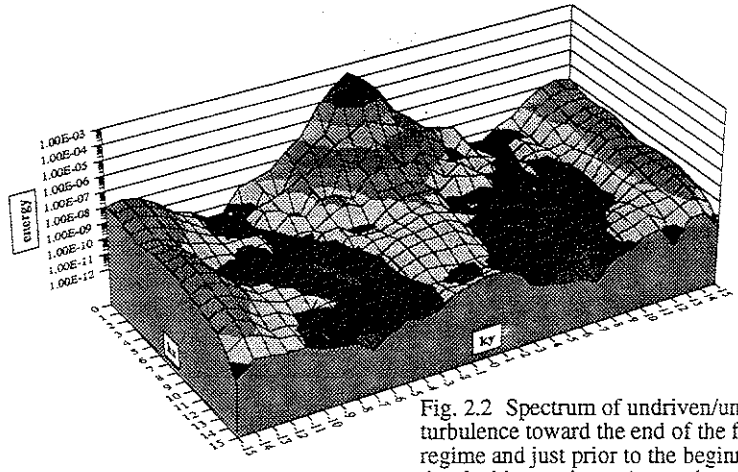


Fig. 2.2 Spectrum of undriven/undamped turbulence toward the end of the flow regime and just prior to the beginning of the sloshing regime. Approximately 5 - 10 correlation times have elapsed from the initial time.

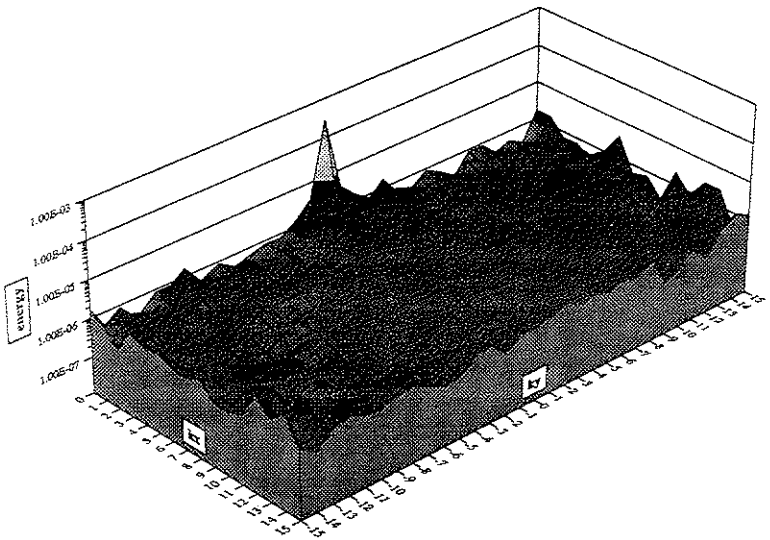


Fig. 2.3 Spectrum of undriven/undamped turbulence in the sloshing regime. The spectrum has been averaged over several correlation times. This spectrum is in the time-asymptotic (relaxed) configuration. (The central peak is the mode kx = ky = 0. Because this mode does not couple to any other mode, it retains its initial energy.)



Because the spectrum at the end of the flow regime is anisotropic and not quite at the equilibrium configuration, a second regime ensues, characterized by sloshing of energy in wavenumber space. The sloshing regime exhibits rapid transfer of energy between modes but with little net transfer. The spectrum is quickly isotropized and flattened by the sloshing. The time-asymptotic spectrum is nearly flat with a slight peak at high $k_y$. The sloshing interaction incorporates a wide range of time scales, from an eddy turnover time to hundreds of eddy turnover times. The sloshing motion roughly produces the equipartitioned spectrum predicted by equilibrium statistical mechanics. However, this regime is probably not relevant to the transfer occurring in a driven stationary state. Insofar as the nonlinear transfer of the steady state is concerned, it is essentially an artifact of the finite (truncated) k-space coupled with the efficient nonlocal transfer, and the fact that there is no dissipation at short wavelengths to absorb the energy nonlinearly transferred from long wavelengths.

While the two regimes of nonlinear transfer have a noticeable effect on the spectrum, their properties are most clearly seen in the energy transfer and enstrophy production diagnostics. Figure 2.4 shows the net energy transfer from bands of constant $k_y$ and $k_x$ in both the long and short wavelength parts of the spectrum. The flow regime is clearly apparent as the period over which there is a continuous outflow from low k, as evidenced by the negative value of $T_k$ for the long wavelength bands, and an inflow to high k, as seen in the positive value of $T_k$ for the short wavelength bands. After t = 2.5, the sloshing regime is reached and the transfer is oscillatory with no net sign apparent. Figure 2.5 indicates that the enstrophy production rate is positive throughout the flow regime and saturates at the transition to the sloshing regime. Positive enstrophy production results from the conservative transfer of energy from long to short wavelength, with the net increase of enstrophy proportional to the ratio $k_{max}^2 / k_{min}^2$. The observed positive enstrophy production rate thus corroborates the results of the energy flow diagnostic. Note that the

negative enstrophy production rate occurring transiently at the onset of sloshing coincides with the first wave of back-transfer.



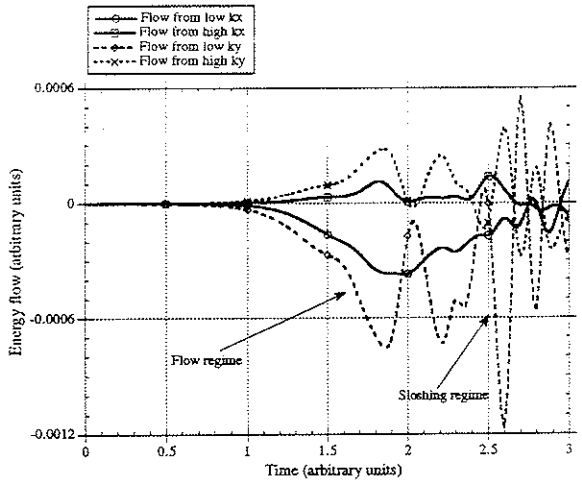


Fig. 2.4 Net energy transfer rates for long and short wavelength bands of constant ky and kx . Energy is seen to flow out of the long wavelength bands (Tk negative) and into the short wavelength bands (Tk positive). Distinct flow and sloshing regimes are identifiable.

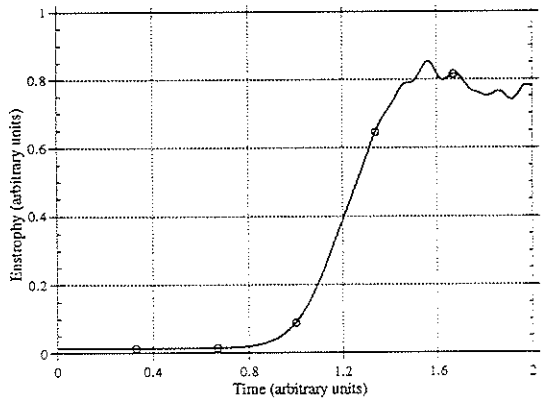


Fig. 2.5 Time evolution of the enstrophy. Enstrophy production is clearly evident in the flow regime, coinciding with energy transfer to long wavelength.

Nonlocal transfer and its magnitude relative to local transfer is now examined. Figure 2.6 shows the time evolution of local and nonlocal transfer rates from a long wavelength $k_y$ = constant band. By symmetry, transfer measured from a $k_y$ band is dominantly the transfer in the $k_y$ direction. From Fig. 2.6, the transfer in the flow regime is almost entirely nonlocal with a small local component. The initial spectrum in this case is $|\tilde{n}_k|^2 \sim k^{-4}$. Given the definition of nonlocal used in this study, this figure indicates that energy is efficiently passing directly from the low $k_y$ modes to modes which are more than one half of the wavenumber space removed. Stated in terms of the wavenumber triangles for the nonlinear interaction of k, k', and k–k', triangles which are highly elongated are strongly favored over equilateral triangles in carrying the energy to high $k_y$. By contrast, Fig. 2.7 indicates that the transfer from a long wavelength band with $k_x$ = constant is divided roughly equally between local and nonlocal components. There is therefore a pronounced anisotropy in the energy transfer process during the flow regime.

The anisotropy in transfer is also evident in the evolution of isodensity contours throughout the relaxation. Figures 2.8 - 2.10 show isodensity contours at the initial time, in the flow regime, and in the sloshing regime when the spectrum has reached equilibrium in an average sense. As expected, the initial isodensity contour plot has just one or two structures, consistent with a spectrum peak at low k. As the spectrum evolves through the flow regime (excitation in the high $k_y$ modes is growing rapidly) the anisotropy in the transfer manifests itself in an anisotropy in the isodensity contours. Small scale structures appear in the $k_y$ direction while few appear in the $k_x$ direction. This can be easily misinterpreted as stretching in the $k_x$ direction, but it should be remembered that the system is evolving by energy transfer from large scale to small scale structures. Hence, it is not stretching, but rather a break-up of the structure in the $k_y$ direction which is occurring. In the sloshing regime, the contours relax to an approximately isotropic configuration, consistent with the isotropic equipartitioned equilibrium spectrum.



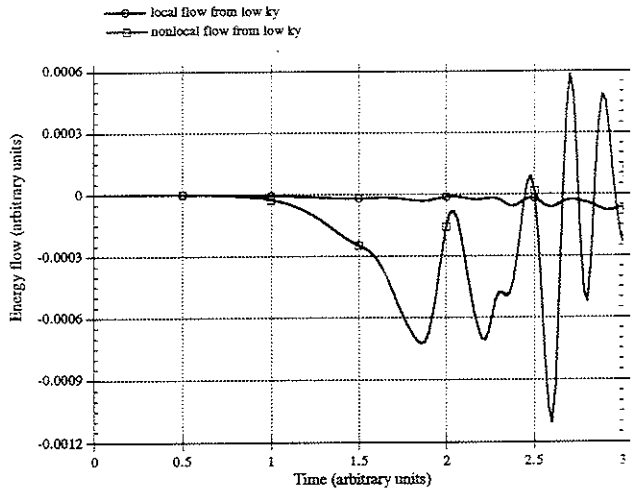


Fig. 2.6 Local and nonlocal transfer rates in the ky direction. Transfer is clearly dominated by the nonlocal process, which here is defined as transfer between coupled modes separated by more than half of the wavenumber space.

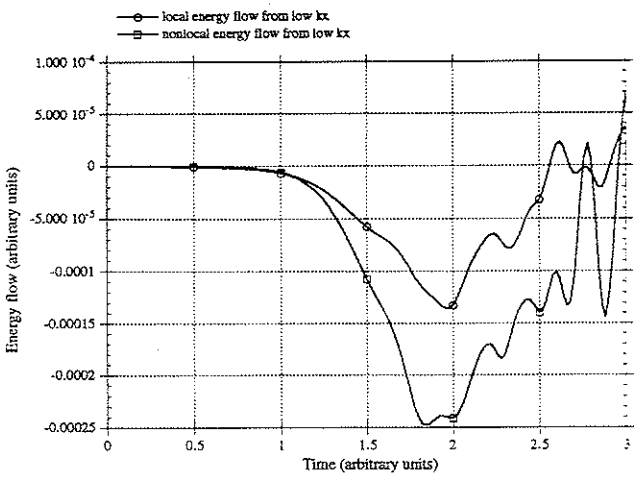


Fig. 2.7 Local and nonlocal transfer rates in the kx direction. The parity of local and nonlocal transfer rates in the kx direction and the disparity of local and nonlocal transfer rates in the ky direction indicates a clear anisotropy in the transfer process.

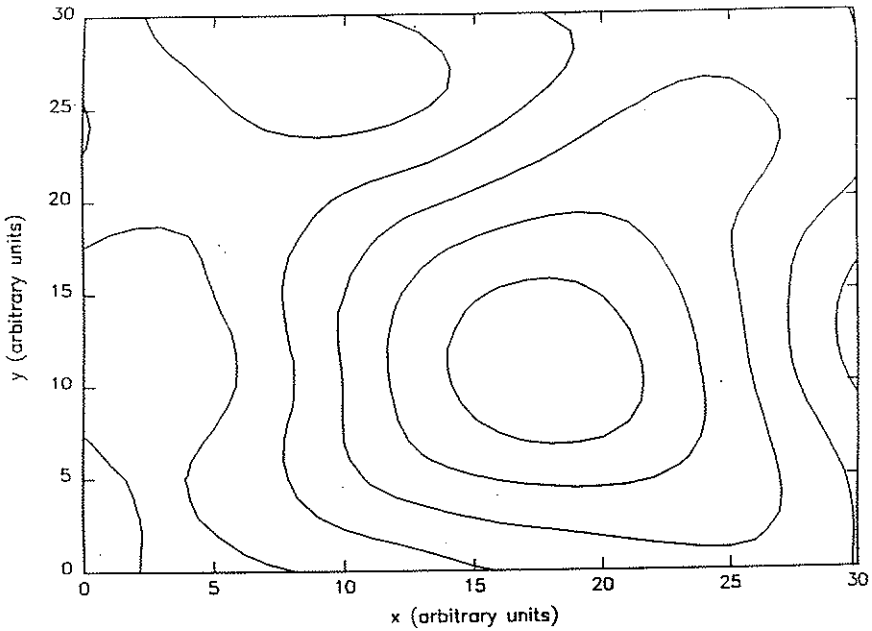


Fig. 2.8 Contours of constant density at the initial time in the relaxation of undriven/undamped turbulence. The peak at low k in the spectrum (Fig. 2.1) is evident in the large scale structure of the contour plot.



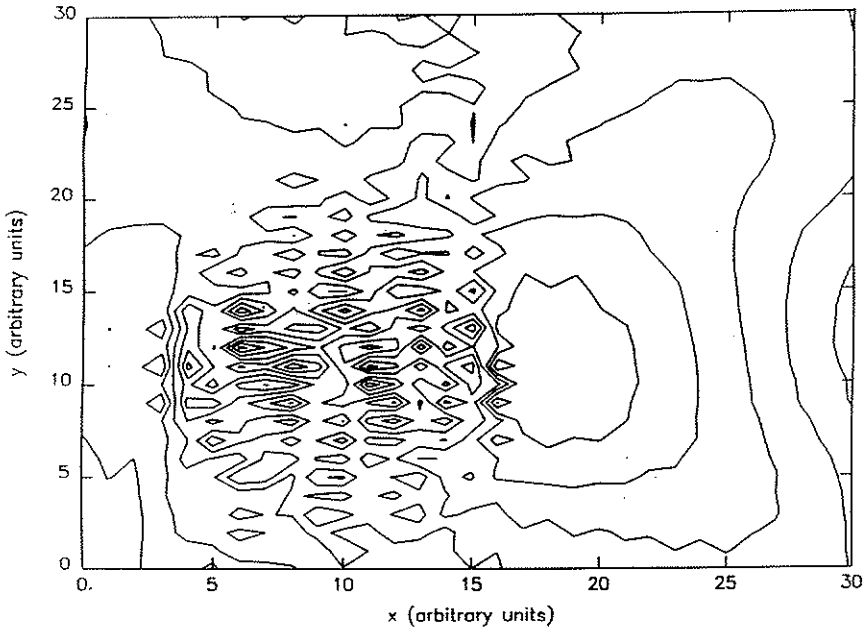


Fig. 2.9 Contours of constant density in the flow regime. Small scale motion has been excited through the nonlinear transfer process with anisotropy due to the effectiveness of nonlocal transfer in the $k_y$ direction.

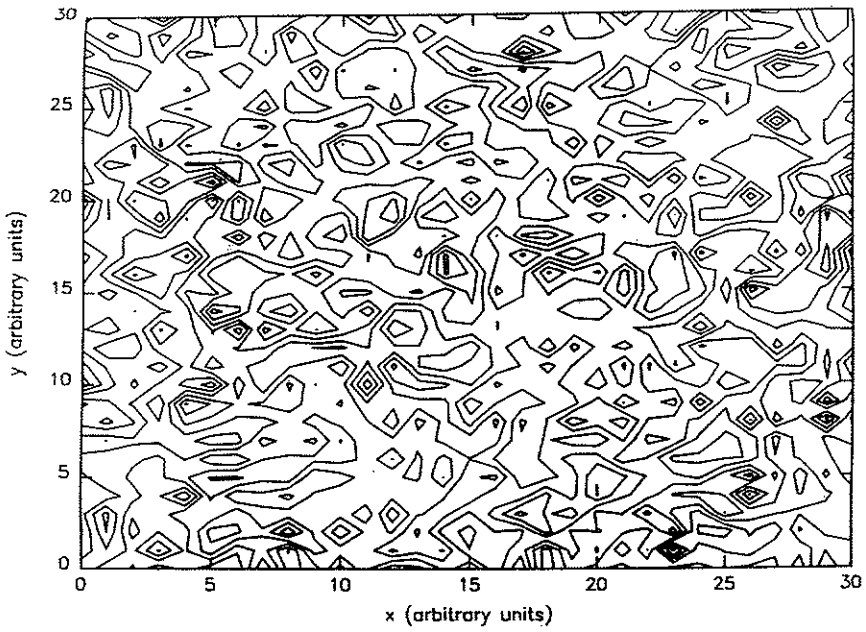


Fig. 2.10 Contours of constant density in the sloshing regime (see Fig. 2.3). Sloshing has isotropized the contours.



Transfer during the sloshing regime tends to be dominantly nonlocal in both the $k_x$ and $k_y$ directions. This is understandable because the mechanism which keeps the transfer local in $k_x$ is suppressed in the sloshing regime. This mechanism is generally responsible for the locality of transfer in most systems, and stems from the amplitude dependence of the transfer coupled with the fact that the amplitude distribution generally falls off with higher k. If the spectrum decays as $n \sim n_o k^{-\alpha}$, then the transfer coupling between modes k and $k' > k$ goes as $T_k \sim n_k^2 n_{k'}^2 \sim n_o^4 k^{-2\alpha}(k'-k)^{-2\alpha}$ [using the closure representation, Eq. (2.2)] . Clearly, for $\alpha > 0$, transfer to neighboring modes $k' \gtrsim k$ is strongly favored over transfer to distant modes ($k' >> k$). However, when the spectrum becomes nearly flat and $\alpha \approx 0$ (as in the case at the end of the flow regime), all modes have equal amplitude and the ratio of nonlocal to local transfer must increase.

This notion suggests that the strong nonlocal transfer in $k_y$ exhibited in Fig. 2.6 should be suppressed if the initial spectrum is made sufficiently steep. Indeed, when the initial spectrum slope exceeds $k^{-3}$, nonlocal transfer is suppressed. In this case, transfer is local, producing a slow relaxation of the spectrum. Once the spectrum has relaxed to the $k^{-3}$ slope, nonlocal transfer quickly begins and relaxation proceeds to the equilibrium spectrum at a much increased rate.

The dominance of nonlocal transfer over local transfer as evidenced in Fig. 2.6 is strongly incompatible with the notion of a self-similar cascade which underlies the Kolmogorov-type spectrum [Eqs. (2.12) and (2.13)]. A similarity range cannot exist, at least in the $k_y$ direction, as energy is transferred out of a mode and across the entire spectrum range in one correlation time. Indeed, the spectrum of steady state turbulence with driven modes at extremely low k, an intermediate inertial range over most of the spectrum, and a hyperviscosity at the highest wavenumbers, bears no resemblance to the similarity range stationary spectrum of Eq. (2.13). Figure 2.11 reveals the stationary spectrum to be nearly flat with slight peaking at low k and strong quenching of the amplitudes in the

dissipation range. While the inertial range is of limited extent in this spectrum (approximately 1 decade), the flatness is a robust feature independent of the strength of driving and the saturated turbulence level. Moreover, theoretical work based on the solution of a two-point equation likewise indicates that a flattening of the spectrum occurs as a result of the nonlocal transfer of the E×B nonlinearity[15]. Because the similarity range stationary spectrum differs so markedly from the numerical spectrum of stationary driven/damped turbulence, it is important to determine the direction of energy transfer in situations with spectra like that of Fig. 2.11. The transfer associated with the spectrum of Fig. 2.11 is found to be directed to large k, as would be expected from the configuration of sources and sinks. Moreover, the transfer from an initially flat spectrum in the undriven undamped case is also toward high k, producing a transfer rate history similar to that of Fig. 2.4, but with reduced magnitude. In this situation, nonlocal transfer proceeds in both directions whereas local transfer is directed to high $k_y$. Consequently, the spectrum develops a slight peak in $k_y$, and a sloshing regime ensues.

It is valid to ask whether the anisotropy in the transfer and the dominantly nonlocal transfer in the $k_y$ direction are artifacts of the size of the k space. While it is not possible to say definitively that it is not an artifact, all evidence suggests that it is real. The wavenumber space has been varied from $13 \times 13$ modes to $41 \times 41$ modes with the nonlocal and anisotropic features becoming more pronounced in the case of a larger k space, rather than the contrary. Furthermore, both the nonlocality and anisotropy are corroborated by the structure of the one-point closure equations and, to a lesser degree, the nonlinearity itself.



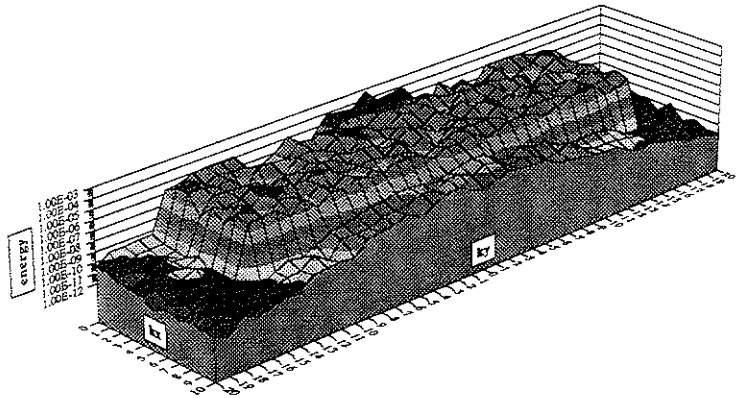


Fig. 2.11 Spectrum of turbulence driven at long wavelength and damped by a hyperviscosity at short wavelength with an inertial range in the intermediate modes. The spectrum is only slightly peaked in the inertial range, a result which differs markedly from the similarity range stationary spectrum [Eq. (2.9)]. The flatness of the spectrum is a result of the nonlocal

## 2.5 Conclusions

A numerical study of dissipative trapped ion convective cell turbulence has been described. This study is based on the spectral solution of the Kadomtsev-Pogutse equation. The primary focus has been on basic physics issues associated with the transfer dynamics of the dominant nonlinearity for long wavelength fluid plasma turbulence, the E×B nonlinearity. The principal results concern the equilibrium spectrum and its relation to statistical mechanics predictions, the direction of energy transfer in k space, and the nonlocality and anisotropy of the transfer process.

Finite amplitude turbulence evolving in the absence of driving and damping has been found to undergo a relaxation in which spectra initially peaked at low k evolve to a flat, equipartitioned and isotropic spectrum. The time-asymptotic spectrum is in good agreement with the predictions of equilibrium statistical mechanics and validates the assertion underlying the statistical mechanics calculation that the sole invariant constraining the dynamics of spectral transfer is the energy. The observed time-asymptotic spectrum also validates the statistical mechanics prediction that for a driven steady state with a spectrum peaked at low k, spectral energy flow will proceed from long to short wavelengths. The agreement with statistical mechanics is remarkable in that the analysis takes no account of a pronounced anisotropy in the nonlinearity and nonlinear transfer rate.

The transfer of energy occurring during the relaxation from a spectrum peaked at long wavelength to the equilibrium is measured to be in the direction of high k. Energy transfer occurs throughout the relaxation phase, with net time average transfer becoming zero when the equilibrium configuration is reached. A monotonically increasing enstrophy coincides with the phase of spectrum relaxation and net energy transfer to short wavelength. Enstrophy production is zero thereafter. With respect to its invariant properties and gross



energy flow characteristics, it is clear that dissipative trapped ion convective cell turbulence is closer to 3-D Navier-Stokes turbulence than to its 2-D counterpart or Hasegawa-Mima drift wave turbulence. On the other hand, long wavelength drift wave turbulence (e.g., trapped electron mode turbulence) is closely related to the present model[12].

On closer inspection, details of the energy transfer are found to deviate dramatically from the self-similar cascade which represents the conventional wisdom for Navier-Stokes turbulence and inertial ranges in general. In particular, a robust and efficient nonlocal energy transfer process is observed which is capable of carrying energy from a low k mode to the other extreme of the spectrum in a correlation time. In the $k_y$ direction this process dominates local cascading except when the spectrum is sufficiently steep. Transfer in the $k_x$ direction is divided equally between local and nonlocal components, implying a strong anisotropy in the transfer process. Nonlocal transfer violates the self-similarity hypothesis of Kolmogorov. Indeed, the inertial range spectrum in a driven/damped steady state is found to differ markedly from the Kolmogorov prediction.

The existence of a direct transfer of energy transfer corroborates a key assertion of Ref. 4 and confirms the real possibility that dissipative convective cell turbulence plays a significant role in core fluctuation activity in hot tokamak discharges. Clearly, further work is needed, particularly in studying the transfer at smaller scales where the polarization drift nonlinearity becomes important and coupling to other short wavelength fluctuations becomes possible. The effect of the nonlocality of transfer on spectra, spatial transport, and other descriptive measures of turbulence also represents an issue requiring additional work.

**References**


1. R. Saison, H.K. Wimmel, and F. Sardei, Plasma Physics **20**, 1 (1978).
2. W. Horton, D.I. Choi, P.W. Terry, and D. Biskamp, Phys. Fluids **23**, 590 (1980).
3. H. Biglari, P.H. Diamond, and P.W. Terry, Phys. Rev. Lett. **60**, 200 (1988).
4. P.H. Diamond and H. Biglari, Phys. Rev. Lett. **65**, 2865 (1990).
5. R.J. Fonck, S.F. Paul, D.R. Roberts, Y.J. Kim, N. Bretz, D. Johnson, R. Nazikian, and G. Taylor, 18th European Conference of Controlled Fusion and Plasma Physics (European Physical Society, Vol. 15C, Part I, 1991) p. I-269.
6. P. Cripwell and A.E. Costley, 18th European Conference of Controlled Fusion and Plasma Physics (European Physical Society, Vol. 15C, Part I, 1991) p. I-17.
7. R.H. Kraichnan, J. Fluid Mech. **67**, part 1, 155 (1975).
8. S.A. Orszag, "Lectures on the Statistical Theory of Turbulence", in *Fluid Dynamics: Les Houches*, Edited by R. Balin and J.-L. Peube, (Gordon and Breach, New York, 1973), p. 235.
9. F.Y. Gang, B.D. Scott, and P.H. Diamond, Phys. Fluids B **1**, 1331 (1989).
10. G.G. Craddock, P.H. Diamond, and P.W. Terry, Phys. Fluids B **3**, 304 (1991).
11. A. Hasegawa and K. Mima, Phys. Rev. Lett. **39**, 205 (1977).
12. P.W. Terry and W. Horton, Phys. Fluids **25**, 491 (1982).
13. G.G. Craddock, private communication.
14. P.W. Terry and P.H. Diamond, Phys. Fluids **28**, 1419 (1985).
15. P.W. Terry and P.H. Diamond, in International Sherwood Fusion Conference, paper 2C13, Seattle, Washington, 1991.




# Chapter 3

# Improper Spectral flow from Polarization Drift Nonlinearity

## 3.1 Introduction

Spectral transfer of energy and other dynamical invariants, such as the enstrophy, or mean squared vorticity[1-3] has long been considered an important aspect of turbulence. In particular, the nature and direction of spectral energy transfer has direct bearing on the way in which instability-driven turbulence is saturated, on the magnitude and shape of the spectrum, and ultimately on the nature of spatial transport produced by the turbulence.

A number of recent studies underscore the importance of spectral transfer. For example, it has recently been shown from closure theory[4] and direct measurement of spectral transfer rates in numerically integrated model equations[1], that dissipative trapped ion convective cell turbulence transfers energy from the long wavelengths of the driving instability to shorter wavelengths. This result has invalidated prior dogma which held that dissipative trapped ion convective cells would transfer energy to longer wavelengths, producing extremely large cell sizes and catastrophic transport. In related studies of broad band dissipative trapped electron mode turbulence[2,3], it has been found that the spectral transfer evinces two distinct subranges at long and short wavelength extremes, separated by a highly complex intermediate subrange. In the long wavelength subrange, energy is transferred to small scales in a process that is distinctly anisotropic and nonlocal in wavenumber space. Significant production of enstrophy accompanies the transfer to shorter wavelength. In the short wavelength subrange, nonlinear transfer very nearly conserves enstrophy. The constraint of two conserved quadratic quantities (energy and enstrophy)

gives rise to an isotropic, local-in-wavenumber space dual cascade with some energy flowing back toward the long wavelength subrange and enstrophy flowing toward shorter scales. These spectral transfer properties produce a distinctive energy spectrum shape $E(k_x,k_y)$, with a flat elliptical plateau in the long wavelength subrange and a fall-off beyond in the short wavelength subrange[2].

The dual cascade observed in the short wavelength subrange of dissipative trapped electron mode turbulence is a manifestation of a transfer process first identified in two-dimensional (2-D) Navier-Stokes turbulence (and, by simple extension, quasi-geostrophic turbulence). Under a dual cascade, it is envisaged that from the scale at which energy and enstrophy are externally injected into a system, the enstrophy is conservatively cascaded to smaller scales through an enstrophy similarity range (i.e., transfer proceeds through every scale at the same rate), and energy is conservatively cascaded in the 'inverse' direction to larger scales through an energy similarity range. The existence of two non-overlapping similarity ranges, one for each of the two conserved quadratic quantities, is posited in order to satisfy the invariance of both the energy and the enstrophy[5,6]. The dual similarity range stationary spectrum, with a $k^{-5/3}$ slope in the energy similarity range and a $k^{-3}$ slope in the enstrophy similarity range, follows directly from the dual cascade hypothesis and simple dimensional arguments and has been observed in 2-D neutral fluid flows[7].

The dual cascade of 2-D Navier-Stokes turbulence has come to be a compelling paradigm for drift wave turbulence, a natural consequence of the near isomorphism of the Hasegawa-Mima[8] equation with the quasi-geostrophic equation. The dual cascade is frequently invoked in a variety of drift wave models in order to infer the character and magnitude of fluctuations at the largest scales of a system, or to explain excitation at scales large compared to those of the driving instability. As such, the dual cascade is a key element of turbulence driven by electron temperature ($\eta_e$) modes[9]. More generally, it is assumed that the dual cascade is an element of spectral transfer for any system possessing



multiple quadratic invariants. Thus it figures prominently in the transfer dynamics of magnetohydrodynamic (MHD) turbulence[10]. As an example, specific properties of the dual cascade of three dimensional MHD have recently been invoked in order to predict ion heating rates in reversed field pinch discharges[11]. It is worth remarking that the dual cascade applies to spectral transfer in an inertial range. An inertial range may or may not exist in plasma turbulence, due to the non-localized nature of plasma driving sources and sinks. Nevertheless, inertial range transfer is a powerful and revealing characterization of the behavior of nonlinearities that are themselves conservative, and therefore would support an inertial range in the absence of dissipation.

The conventional view of the dual cascade process is based on analysis of spectral energy and enstrophy flow in 2-D Navier-Stokes turbulence for an infinite spectrum encompassing wavenumbers from zero to infinity. For a spectrum with a single falloff rate of $k^{-3}$ over its entire range ($0-\infty$), Kraichnan proved that there is a wavenumber independent flow of enstrophy to high k, *and no flow of energy*. Such flow defines an enstrophy similarity range. Likewise, for a spectrum with a single falloff rate of $k^{-5/3}$, there is a wavenumber independent flow of energy to low k, and no flow of enstrophy, defining an energy similarity range. In the $k^{-3}$ spectrum, enstrophy is effectively injected at $k=0$ and removed at $k=\infty$, though no injection scale is specified in the analysis. In the $k^{-5/3}$ spectrum, energy is effectively injected at $k=\infty$ and removed at $k=0$. Both spectra are singular in the sense that there is no physically distinguishable scale that is not either zero or infinity, i.e., the injection scale and the minimum and maximum wavenumber cutoffs are all either zero or infinity. Because these singular spectra are required for the proof, true similarity ranges can strictly be said to occur only in these rather extreme and amorphous spectra.

In fact, the standard stationary inertial range spectrum in 2-D turbulence is not one of the singular spectra analyzed by Kraichnan, but is forced or stirred at a finite scale $k_{inj}^{-1}$. It

also has maximum and minimum wavenumbers $k_{max}$ and $k_{min}$ corresponding to the smallest and largest inertial scales, typically set by dissipation and geometry. For drift wave turbulence, the spectrum is likely to be restricted to a very narrow range, perhaps encompassing no more than a single decade. For such finite spectra, it is crucial to account for the enstrophy carried in the energy flow and visa versa. Thus, a self-similar inverse cascade of energy from $k_{inj}$ to $k_{min}$ results in the destruction of a portion of the enstrophy carried with the energy flow. Enstrophy is destroyed because it is proportional to the square of a spatial derivative (curl) of the flow, which necessarily becomes flatter as energy flows to large scale. This loss can be accommodated in an enstrophy conserving system only if enstrophy is created in some other part of the spectrum. The correct amount of enstrophy can be created if there is a self-similar cascade in the *reverse* direction (from $k_{inj}$ to $k_{max}$) of an appropriate fraction of the energy. Under this scenario, self-similar cascades of energy proceed in both directions, with accompanying nonconservative flows of enstrophy. The portion of energy flowing self-similarly in the reverse direction (to high k) is determined by the constraint of enstrophy conservation. Specifically, the enstrophy generated in the self-similar flow of energy to high k must equal the enstrophy destroyed in the self-similar flow of energy to low k. The above description applies to self-similar energy flow and the enstrophy carried with it. By symmetry, and in order to recover the results of Kraichnan's proof in the appropriate limits, there must also be a self-similar cascade of enstrophy, and it must proceed in both directions so as to yield zero net energy production from the accompanying nonconservative energy flows. This picture represents the simplest way to satisfy energy and enstrophy conservation, to reduce to Kraichnan's results in the appropriate limits, and to account for the changes that occur in one quantity when the other quantity is transferred between different scales.

In the present chapter, spectral flows in numerical realizations of Hasegawa-Mima turbulence in a finite spectrum are measured and found to conform to this picture as



formulated under simple dimensional analysis. The numerical results are obtained from spectral solution of the basic equation, with flow measurements achieved through direct evaluation of the triplet nonlinearity (of the power spectrum evolution equation). The numerical results indicate that significant nonconservative flows of enstrophy occur in both directions, and are consistent with enstrophy carried in self-similar energy flows proceeding in both directions. The reverse energy flow (self-similar energy flow to high k) generates sufficient enstrophy to compensate for the loss of enstrophy resulting from the proper energy flow (self-similar inverse cascade of energy to low k). Likewise, nonconservative flows of energy, consistent with self-similar flows of enstrophy in both directions are observed. The reverse enstrophy flow (to low k) destroys sufficient energy to compensate for the generation of energy by the proper enstrophy flow (to high k). For infinite spectra ($k_{min} \rightarrow 0$, $k_{max} \rightarrow \infty$), the dimensional analysis indicates that the reverse flows of energy and enstrophy vanish, while the nonconservative flows occur only in a narrow band near $k_{inj}$. Outside this band, the flows are self-similar and proceed according to the standard dual cascade hypothesis.

### 3.2 Dimensional Analysis

Three constraints govern the flow of energy and enstrophy in a finite spectrum $k_{min} < k_{inj} < k_{max}$. The flow configuration must account for changes in enstrophy (energy) due to the transfer of energy (enstrophy), energy and enstrophy must be conserved, and the flow configuration must reduce to Kraichnan's results in the proper singular spectrum limits. These constraints are most easily accommodated by dividing the flow of each invariant quantity (energy and enstrophy) into two components, a component that is transferred self-similarly (from which the correct self-similar transfer is recovered in the singular spectrum limits) and a locally (in wavenumber space) non-conserved component

representing the energy or enstrophy carried by the self-similar flow of the other quantity. At the injection scale, all inputted energy and enstrophy must be partitioned into either of these components, so that

$$E_O = E_C + E_N\,, \qquad \Omega_O = \Omega_C + \Omega_N, \tag{3.1}$$

where $E_O$ is the injected energy, $E_C$ is the amount of injected energy carried in self-similar energy flows, and $E_N$ is the amount of injected energy in the non-conserving energy flows representing the energy carried by the self-similar *enstrophy* flows. Similar definitions apply to the inputted enstrophy $\Omega_O$. Because enstrophy is mean squared vorticity, the Fourier enstrophy $\Omega(k)$ (defined as the enstrophy of a Fourier mode of wavenumber k) of any flow with Fourier energy E(k) is given by $\Omega(k) = k^2 E(k)$. Thus the partition $E_O = E_C + E_N$, with $E_N$ being the portion of injected energy carried by the self-similar enstrophy cascade, implies that

$$E_N = k_{inj}^{-2}\,\Omega_C. \tag{3.2}$$

Likewise, we have

$$\Omega_N = k_{inj}^{2}\,E_C. \tag{3.3}$$

Consider now the self-similar energy flow. Let $E_p$ be the energy carried in a proper energy conserving cascade from $k_{inj}$ to $k_{min}$. A quantity of enstrophy $E_p k_{inj}^2$ is carried in this cascade at the scale $k_{inj}^{-1}$, but dwindles to $E_p k_{min}^2$ as the energy reaches $k_{min}$, a consequence of the smoother gradients associated with the large scale $k_{min}^{-1}$. Obviously, the proper cascade of energy $E_p$ between $k_{inj}$ and $k_{min}$ produces a net loss of enstrophy of



magnitude $E_p(k_{inj}^2 - k_{min}^2)$. This loss is the end result of the nonconservative enstrophy flow associated with the proper energy cascade. To assure overall conservation of enstrophy, enstrophy equal to the amount lost must be generated somewhere in the spectrum. This can occur if a portion of energy $E_r$ cascades conservatively in the reverse sense from $k_{inj}$ to $k_{max}$. The steeper gradients associated with a flow of energy $E_r$ at a scale $k_{max}^{-1}$ results in a net production of enstrophy of magnitude $E_r(k_{max}^2 - k_{inj}^2)$. Equating the net loss of enstrophy resulting from the proper cascade of energy with the net production of enstrophy resulting from the reverse cascade, the energies $E_p$ and $E_r$ are

$$E_p = E_c \frac{(k_{max}^2 - k_{inj}^2)}{(k_{max}^2 - k_{min}^2)} = E_c \frac{(1 - I^2)}{(1 - R^2)}\,, \tag{3.4}$$

$$E_r = E_c \frac{(k_{inj}^2 - k_{min}^2)}{(k_{max}^2 - k_{min}^2)} = E_c \frac{(I^2 - R^2)}{(1 - R^2)}\,, \tag{3.5}$$

where the total amount of conservatively cascaded energy $E_c = E_r + E_p$ is split into proper and reverse components, $I = k_{inj}/k_{max}$ and $R = k_{min}/k_{max}$. From these expressions it is obvious that as $k_{max} \to \infty$ with $k_{inj}$ remaining finite, $E_p \to E_c$ and $E_r \to 0$, yielding a unidirectional self-similar flow to low k, consistent with the dual cascade hypothesis for $k_{inj}$ finite.

These arguments can be repeated for the conservative spectral flow of enstrophy. If $\Omega_p$ is the portion of enstrophy undergoing a proper self-similar cascade from $k_{inj}$ to $k_{max}$, energy carried in this flow decreases from its value $\Omega_p k_{inj}^{-2}$ at the scale $k_{inj}$ to $\Omega_p k_{max}^{-2}$ at $k_{max}$, i.e., a net amount of energy $\Omega_p(k_{inj}^{-2} - k_{max}^{-2})$ is lost in the conservative enstrophy transfer. Consequently, there must be a portion of enstrophy $\Omega_r$ reverse-cascaded to $k_{min}$, which produces a net increase of energy $\Omega_r(k_{min}^{-2} - k_{inj}^{-2})$. Equating the net loss and gain of energy in order to maintain energy conservation, the quantities of enstrophy cascaded in the proper and reverse directions are

$$\Omega_p = \Omega_c \frac{(k_{min}^{-2} - k_{inj}^{-2})}{(k_{min}^{-2} - k_{max}^{-2})} = \Omega_c \frac{(I^2 - R^2)}{(1 - R^2)} \frac{1}{I^2}, \tag{3.6}$$

$$\Omega_r = \Omega_c \frac{(k_{inj}^{-2} - k_{max}^{-2})}{(k_{min}^{-2} - k_{max}^{-2})} = \Omega_c \frac{(1 - I^2)}{(1 - R^2)} \frac{R^2}{I^2}, \tag{3.7}$$

where $\Omega_c = \Omega_p + \Omega_r$ is the total amount of enstrophy conservatively cascaded. Again, $k_{min} \rightarrow 0$ with $k_{inj}$ remaining finite implies that $\Omega_r \rightarrow 0$ and $\Omega_p \rightarrow \Omega_c$.

In the above discussions, the fraction of total injected energy that gets carried by the conservative enstrophy cascade $E_N$, is not specified relative to the fraction of energy going into the conservative energy cascade $E_c$. A reasonable hypothesis for this partition, based loosely on similarity and statistical homogeneity arguments, follows from

$$\Omega_c = E_c k_{inj}^2. \tag{3.8}$$

Note that for an infinite spectrum ($k_{max} = \infty$, $k_{min} = 0$) the ansatz of Eq. (3.8) applies solely to the proper cascades, i.e., spectral flows proceeding in opposite directions away from $k_{inj}$; otherwise it applies to flows moving in both directions. Equation (3.8) fixes the ratio of conservative to nonconservative flow since the nonconservative energy flow is governed by the conservative enstrophy flow, $E_n = \Omega_c k_{inj}^{-2}$. Thus we have $E_n = E_c$. From these considerations, it is apparent that the nonconserved flows in either similarity range are significant. Even for the infinite spectrum ($k_{max} = \infty$, $k_{min} = 0$, with $k_{inj}$ finite), where the self-similar or conserved flows are entirely proper, the nonconserved flows remain, and they proceed in the reverse sense. The enstrophy in the nonconserved enstrophy flow decreases like $k^{-2}$ as it cascades toward $k_{max} = \infty$. Similarly, the energy in the nonconserved energy flow decreases like $k^2$ as it cascades toward $k_{min} = 0$. Consequently the nonconserved flow is effectively confined to a region around $k_{inj}$.



Outside this region, flow is largely self-similar and proper, consistent with the dual cascade hypothesis. Clearly, for finite spectra of limited extent, departures from the dual cascade picture are significant, arising from both the large nonconserved flows, as well as the reverse self-similar flows. The fact that reverse self-similar flows can be large is illustrated in Figs. 3.1 and 3.2, which plot the conserved energy and enstrophy flows [Eqs. (3.4) - (3.7)] as a function of the injection scale, $I = k_{inj}/k_{max}$ for $R = k_{min}/k_{max} = 0.1$.

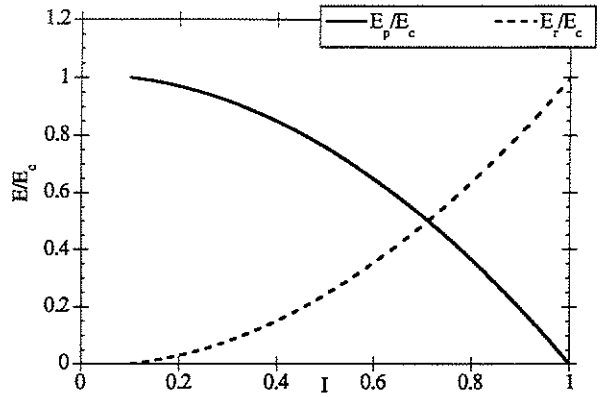


Fig. 3.1 The magnitude of the normalized self-similar energy flows in the proper and reverse directions as a function of I = kinj/kmax. Normalization is with respect to the total energy carried in self-similar flows.

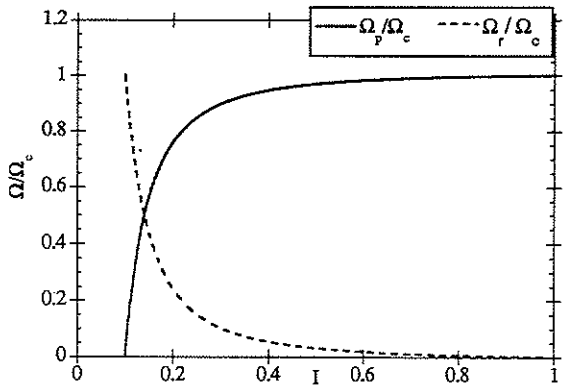


Fig. 3.2 The magnitude of the normalized self-similar enstrophy flows in the proper and reverse directions as a function of I = kinj/kmax. Normalization is with respect to the total enstrophy carried in self-similar flows.



In the limit R→0, I→0 (R/I→0), the spectrum is infinite and all of the conservatively cascaded energy and enstrophy flow in the proper directions. Away form this limit significant fractions of the injected energy and enstrophy can flow in the reverse directions. In particular, for $I > 2^{-1/2}(1 + R^2)^{1/2}$, the reverse energy flow exceeds the proper energy flow. Thus, when $k_{inj}$ is somewhat more than half the distance from $k_{min}$ to $k_{max}$, both enstrophy and energy flow primarily to high k, a result strikingly at variance with the standard dual cascade picture.

### 3.3 Numerical Analysis

In order to determine the extent to which spectral flow conforms to the heuristic description of the previous section, the Hasegawa-Mima equation is solved numerically and spectral flow is measured for several spectrum configurations. The Hasegawa-Mima equation,

$$\frac{\partial}{\partial t}(1 - \nabla_\perp^2 \rho_s^2)\,\hat{\phi} + V_D \frac{\partial}{\partial y}\hat{\phi} + \rho_s^3 C_s \nabla\hat{\phi}\times\mathbf{z}\cdot\nabla\nabla_\perp^2\hat{\phi} = 0, \qquad (3.9)$$

describes collective drift wave fluctuations supported by fluid ions and adiabatic electrons linked through quasineutrality. Here, $\hat{\phi}$ is the electrostatic potential, $V_D = (cT_e/eB)L_n^{-1}$ is the diamagnetic drift velocity representing E×B advection of the background density gradient, $\rho_s = (T_e/m_i)^{1/2}(eB/m_ic)^{-1}$ is the ion gyroradius evaluated at the electron temperature, and $C_s = (T_e/m_i)^{1/2}$ is the ion sound speed. Besides the diamagnetic drift, ion motion is governed by the polarization drift, which produces both the linear dispersion and the nonlinearity. Physically, the polarization drift nonlinearity represents E×B advection of the E×B flow vorticity and corresponds to the nonlinearity of 2-D Navier-Stokes turbulence. The assumption of adiabatic electrons ($n_e = \hat{\phi}$) eliminates linear instability, particle

transport, and the enstrophy invariance-breaking E×B nonlinearity[12]. As is well known, the Hasegawa-Mima equation conserves both the energy $E = \int [\, |\hat{\phi}|^2 + |\nabla\hat{\phi}|^2 \,]\, dxdy$ and the enstrophy $\Omega = \int [\, |\nabla\hat{\phi}|^2 + |\nabla^2\hat{\phi}|^2 \,]\, dxdy$, and produces a condensation of energy at long wavelengths[8]. The latter is a manifestation that some energy undergoes an inverse cascade.

The Hasegawa-Mima equation is solved spectrally by numerically integrating the coupled ordinary differential equations for the time evolution of the amplitudes of the spatial Fourier series representation of $\hat{\phi}$. The wavenumber space is truncated to a finite number of modes with 41×41 representing the largest spectral domain. Finite differencing in time is accomplished with a gear method. Spectral flow of energy is measured by determining the rate of spectral energy transfer

$$T_k = -\rho_s^3 C_s \, \mathrm{Re} \sum_{k'} \mathbf{k}\times\mathbf{k}'\cdot\mathbf{z}\ (k_\perp - k_\perp')^2\ \hat{\phi}_{k'}\,\hat{\phi}_{k-k'}\hat{\phi}_k^*. \tag{3.10}$$

The quantity $T_k$ represents the rate at which energy is deposited into or removed from the mode k by the nonlinear transfer. For $T_k < 0$, energy is transferred from the mode to other parts of the wavenumber spectrum, while for $T_k > 0$, energy is deposited into the mode from other parts of the spectrum. Typically $T_k$ is summed over the modes in a band in wavenumber space, thus giving a measure of the transfer rate into or out of the band. By examining the transfer rate for all bands, it is usually possible to track the flow of energy through the spectrum. Enstrophy flow is determined by measuring the rate of enstrophy transfer,

$$U_k = \rho_s^3 C_s \, \mathrm{Re} \sum_{k'} \mathbf{k}\times\mathbf{k}'\cdot\mathbf{z}\ k^2(k_\perp - k_\perp')^2\ \hat{\phi}_{k'}\,\hat{\phi}_{k-k'}\hat{\phi}_k^*. \tag{3.11}$$

Again, a band structure is utilized and the transfer into or out of all bands is observed.



Spectral transfer is examined under two spectrum configurations. In one, an initial spectrum of randomly phased finite amplitude fluctuations is allowed to relax to a quasi-equilibrium configuration, a state characterized by no net transfer in the time averaged sense. A large potential amplitude pulse is then applied at $k_{inj}$ and the subsequent prompt transfer of energy and enstrophy is tracked throughout the spectrum. In a second configuration, transfer is monitored under conditions more akin to a driven/damped steady state. Here, the system is coherently forced at $k_{inj}$ with dissipative sinks at $k_{min}$ and $k_{max}$. After a steady state is established, a large pulse is again applied at $k_{inj}$, and the subsequent prompt transfer is observed. In both configurations, the prompt flow from the perturbative pulse is large compared to any pre-existing flows. This ensures observation of the transfer characteristics of the nonlinearity, independent of the spatial arrangement or the relative strengths of the sources and sinks. This is necessary because in the steady state, the arrangement and strengths of the sources and sinks can dominate the flow pattern. Both spectrum configurations yield flow patterns that are essentially the same.

The behavior evident in the computational flow patterns is generally more complicated than the transfer of the simple picture developed in the previous section. Figure 3.3 shows the rate of energy transfer from a range of $k_y$ = constant bands spanning wavenumber space. The transfer rate (vertical axis) is effectively an average obtained by summing discrete values of the instantaneous transfer rate over a period covering several nonlinear interaction times. The error bars denote the standard deviation from the mean value plotted. A large negative spike occurs at $k = k_{inj}$, indicating transfer out of the band with the large amplitude pulse. Positive values on either side of $k_{inj}$ indicate that energy is transferred both toward $k_{min}$ and $k_{max}$. The transfer is predominantly toward $k_{min}$; the steady rise in going to $k_{min}$ is produced both by condensation and by the reverse flow of enstrophy to $k_{min}$. The flow of energy to high k diminishes as k increases, consistent with the destruction of energy carried in the proper cascade of enstrophy. The rate of decrease is

roughly consistent with the $k^{-2}$ scaling of the dimensional analysis. The run was terminated before there was any condensation of enstrophy or energy at $k_{max}$.



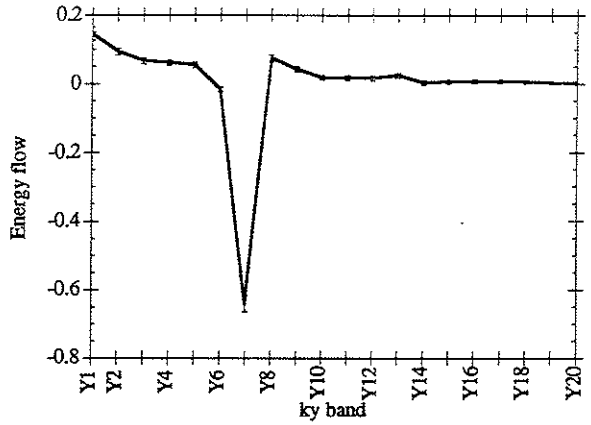


Fig. 3.2 Energy transfer rate for bands with ky = constant in numerical solutions of Hasegawa-Mima turbulence. Transfer is from a large perturbative pulse (at kinj = 7) applied to a quasi-equilibrium spectrum.
The error bars represent the standard deviation of the energy transfer rate about its mean value.

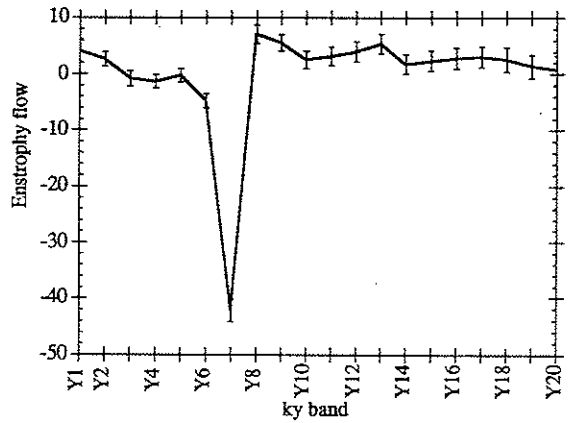


Fig. 3.3 Enstrophy transfer rate for the same case as Fig. 3.2.

The flow pattern for enstrophy is displayed in Fig. 3.4. Again enstrophy is seen to flow from $k_{inj}$ in both directions. Here, however, more enstrophy flows toward high k than to low k. Condensation of enstrophy at $k_{min}$ is clear evidence of a reverse enstrophy cascade but masks the nonconservative enstrophy flow associated with the proper energy cascade. The flow of enstrophy to high k is roughly constant, within error bars, over the enstrophy similarity range. It is worth noting that a self-similar (steady state) cascade would produce zero net transfer into or out of any band. Here the flow is the transient response to the perturbative pulse and represents the propagation of the bulk of enstrophy in the pulse to high k, before condensation at $k_{max}$.

It is likely that differences between the results of Figs. 3.3 and 3.4 and the simple relations derived in the previous section stem both from limitations in the numerical work and the extreme simplicity of the dimensional analysis. In particular, the spectrum configuration is often arranged with $k_{inj}$ as the geometric mean of $k_{max}$ and $k_{min}$. Among other things, this choice makes the fractional proper cascaded energies and enstrophies comparable ($E_p/E_c \approx \Omega_p/\Omega_c$). Moreover, there is some evidence that this scale represents a natural break point for dual cascades in relaxing spectra with no forcing or perturbative pulse. Under such circumstances, $k_{inj}$ is typically closer to $k_{min}$ than it is to $k_{max}$. Consequently, there is considerable energy condensation at $k_{min}$ before the transfer to high k reaches $k_{max}$. Because condensation produces spectrum changes that then affect transfer, it becomes difficult to track the upward transfer all the way to $k_{max}$ once condensation has begun at $k_{min}$. A second limitation results from the analysis of transfer into or out of bands of constant $k_x$ or $k_y$. This band structure permits the examination of anisotropies of the transfer rate. However, for the Hasegawa-Mima equation, the transfer is found to be isotropic, in which case the constant $k_x$ or $k_y$ band structure has the unwanted effect of potentially distorting measurement of transfer. For example, self-similar transfer of energy *within* a $k_x$ = constant band (from large $k_y$ to small $k_y$) can decrease the enstrophy of the



band, independent of the transfer occurring between bands. Finally, the observed flow patterns are affected by random fluctuations of the nonlinear interaction. Even though the flow patterns are time averaged, some random component remains after averaging.

Each of the above difficulties suggests improvements for future computational work. However, there is considerable agreement between the simple model and the results of this imperfect computational analysis. Certainly, the measured flow patterns are considerably altered from those envisioned in the standard dual cascade hypothesis. Energy and enstrophy are transferred away from $k_{inj}$ in *both* directions. Moreover, there is evidence that the flow in a given direction away from $k_{inj}$ is not completely self-similar. Also, runs with inertial ranges varied by over a factor of 3 indicate that the flow patterns tend toward the standard dual cascade configuration as the spectral range increases, i.e., the magnitude of the reverse (conservative) flows decreases relative to that of the proper flows.

### 3.4 Conclusions and Discussion

The notion that energy and enstrophy respectively undergo self-similar cascades to long and short wavelengths in turbulence conserving these quantities has been shown to be appropriate only for infinite spectra ($k_{min}\rightarrow 0$, $k_{max}\rightarrow\infty$) away from the injection scale. Because enstrophy (energy) is carried in wavenumber space by a self-similar energy flow (enstrophy flow) and increases or decreases depending on the direction of the flow, injected energy and enstrophy must flow from the injection scale in both directions. The energy lost in the proper self-similar cascade of enstrophy to high k is then compensated by the energy gained in a reverse self-similar cascade of enstrophy to low k. A similar statement applies to enstrophy lost and gained from proper and reverse energy cascades to low and high k respectively. These constraints have been incorporated into simple scaling expressions from which the magnitude of proper, reverse, and the non self-similar flows are obtained. The standard dual cascade results are recovered from these expressions in the limit $k_{min}\rightarrow 0$ and $k_{max}\rightarrow\infty$. These expressions therefore yield the dominant flow pattern of the infinite spectrum, a result often inferred from statistical mechanics arguments.

Large reverse energy flows are predicted when $k_{inj}$ is near $k_{max}$. This case is instructive in reference to $\eta_e$ turbulence[9], where it has been asserted that there is an inverse cascade of energy to scales given by $c/\omega_{pe}$ (where $\omega_{pe}$ is the electron plasma frequency) from the smaller scales of collective excitation at $\rho_e$ (the electron gyroradius). With driving already at very small scales, it is likely that dissipation occurs at scales only slightly above $\rho_e$, in which case Fig. 3.1 suggests that the dominant energy transfer would be toward short wavelengths, not the longer $c/\omega_{pe}$ scales. Because enstrophy also flows to high k, the dominant non self-similar energy flow would also be toward high k.

This type of consideration clearly demonstrates that the practice of invoking a standard dual cascade for spectra with an inertial range bounded between limits not widely



separated in wavenumber space may not be valid. Even if damping is restricted to a region outside maximum and minimum cutoffs, the location of the scale at which fluctuations are excited within the region will play a significant role in the direction of energy flow. Moreover, because sources and sinks may in fact be distributed, with no true inertial range, determining the k space flow ultimately requires a knowledge of the spectrum. This, in turn, requires solution of the appropriate two-point equations, taking account of both the distributions of sources and sinks and the spectral properties of the nonlinearities.

## References


1. D.E. Newman, P.W. Terry, and P.H. Diamond, Phys. Fluids B **4**, 599 (1992).
2. D.E. Newman, P.W. Terry, P.H. Diamond, and Y.-M. Liang, Phys. Fluids B.
3. Y.-M. Liang, P.H. Diamond, X.H. Wang, D.E. Newman, and P.W. Terry, Phys. Fluids B.
4. P.H. Diamond and H. Biglari, Phys. Rev. Lett. **65**, 2865 (1990).
5. R.H. Kraichnan, Phys. Fluids **10**, 1417 (1967).
6. C.E. Leith, Phys. Fluids **11**, 671 (1968).
7. M. Gharib and P. Derango, Physica D **37**, 406 (1989).
8. A. Hasegawa and K. Mima, Phys. Rev. Lett. **39**, 205 (1977).
9. W. Horton, B.-G. Hong, T. Tajima, and N. Bekki, Comments Plasma Phys. Controlled Fusion **13**, 207 (1990).
10. A. Pouquet, U. Frisch, and J. Léorat, J. Fluid Mech. **77**, 321 (1976).
11. N. Mattor, P.W. Terry, and S.C. Prager, Comments Plasma Phys. Controlled Fusion **15**, 65 (1992).
12. P.W. Terry and W. Horton, Phys. of Fluids **25**, 491 (1982).




# Chapter 4

# Dynamics of Coupled Nonlinearities

## 4.1 Introduction

Recent analytical work indicates that trapped ion convective cell turbulence is a viable candidate for long wavelength fluctuation activity in hot, auxiliary heated core plasmas[1]. Beyond its promise as a core fluctuation and transport model, however, trapped ion convective cell turbulence provides an instructive paradigm for long wavelength turbulence and the spectral transfer properties that ultimately govern its saturation, spectral distribution of energy, and transport[1,2]. Its usefulness as a paradigm follows in part from the fact that trapped ion convective cell turbulence can be described by a one-field fluid model, e.g., the Kadomtsev-Pogutse equation, thus allowing a succinct and transparent representation of the nonlinear mode coupling process. More importantly, a generalization of the Kadomtsev-Pogutse model provides a mode coupling representation with two nonlinearities, the E×B and polarization drift nonlinearities, and is therefore generic to drift wave type fluctuations. For example, the E×B and polarization drift nonlinearities are the nonlinearities that appear in the one-fluid description (Terry-Horton) of turbulence associated with dissipative trapped electron and universal modes[7].

Chapter 2 presented theoretical and computational analysis of purely trapped ion convective cell turbulence that has focused on the long wavelength regime, where nonlinear transfer is dominated by the E×B nonlinearity. These studies demonstrated that the transfer of energy in wavenumber space is directed toward short wavelength. The direct cascade of energy is possible because the conservative transfer of energy is not subject to any additional constraints, i.e., besides the energy, the E×B nonlinearity conserves no other

nontrivial quadratic quantity. In addition to a direct cascade of energy, the E×B nonlinearity produces energy transfer which is anisotropic[2] and highly nonlocal in wavenumber space[1,2]. The occurrence of strong nonlocal transfer is a marked departure from the self-similar cascade dynamics inherent in the scaling arguments of Kolmogorov. The anisotropy of transfer follows from an absence of symmetry induced by the $k_y$ dependence of the nonadiabatic electron response in the E×B nonlinearity. The anisotropy manifests itself as a transfer which is strongly nonlocal in the $k_y$ direction (cross-field direction perpendicular to the density gradient) but characterized by comparable local and nonlocal transfer rates in the $k_x$ (gradient) direction.

The long wavelength regime of trapped ion convective cell turbulence contrasts strongly with a more familiar drift wave turbulence paradigm, the Hasegawa-Mima model[8]. In the Hasegawa-Mima equation, neglect of nonadiabatic electron dynamics eliminates the E×B nonlinearity and leaves the polarization drift nonlinearity (normally subdominant to the E×B nonlinearity in the long wavelength limit) as the sole spectral transfer mechanism. While the neglect of nonadiabatic electron dynamics excludes the possibility of describing either instability or transport, it does produce a model which is nearly isomorphic to the quasi-geostrophic equation. Accordingly, *two* dynamical invariants are admitted by the polarization drift nonlinearity, i.e., the energy and the enstrophy, or mean squared vorticity. In order to satisfy both constraints, a dual cascade process is required with the energy undergoing an inverse cascade or transfer to long wavelength, as in two-dimensional Navier-Stokes turbulence. The cascade dynamics of the Hasegawa-Mima equation are representative of the conventional view of spectral energy transfer in 2-D plasma turbulence. In particular, the notions of the inverse cascade and local transfer in wavenumber space are pervasive in heuristic descriptions of saturation and spectral dynamics.

With nonadiabatic electrons, spectral transfer is affected by both the E×B and polarization drift nonlinearities, and enstrophy conservation, in a strict sense, is broken.



However, the polarization drift nonlinearity involves a higher derivative of the fluctuating potential than the E×B nonlinearity. Consequently, the magnitude of the polarization drift nonlinearity becomes much larger than the magnitude of the E×B nonlinearity at short wavelengths, while the opposite holds at long wavelengths. It would seem reasonable, therefore, to predict the existence of spectral ranges in the short and long wavelength limits of the spectrum in which the transfer dynamics is dominated by one or the other of the nonlinearities. This would mean there is a spectral range at long wavelengths where energy is transferred to smaller scales and in which the total enstrophy within the range evolves on the timescale of the nonlinear interaction. There would also be a range at short wavelengths where energy is transferred to larger scales and enstrophy is approximately conserved on the timescale of nonlinear interactions. One of the primary aims of this chapter is the testing of this hypothesis and the characterization of spectral transfer properties in both the intermediate range where the two nonlinearities play an active role and over the larger spectrum as a whole. Such a study is of direct relevance to trapped ion and electron turbulence because the direct transfer of energy generated at long wavelengths ( $\omega<\omega_{bi}$ ) will inevitably carry energy to a spectrum region where the polarization drift nonlinearity is important, i.e. the dissipative trapped electron mode (DTEM) regime where $\omega_{be}>\omega>\omega_{bi}$ .

It is also of considerable interest to examine the interplay of the two disparate transfer processes in terms of the locality and anisotropy of spectral transfer. In particular, as mentioned above, the E×B nonlinearity, acting alone, is known to produce transfer that is both nonlocal and anisotropic[1,2]. On the other hand, the polarization drift nonlinearity is isotropic in form, and by analogy with the two-dimensional Navier-Stokes equation, should transfer energy locally in wavenumber space[9], consistent with the notion of a Kolmogorov-type similarity range and cascade.

In this chapter, broad-band nonlinear transfer dynamics is examined for dissipative trapped electron mode turbulence (DTEM), including both the E×B and polarization drift nonlinearities. This work is an outgrowth of a previous study which considered only the E×B nonlinearity[2], and is based on numerical solution of a model equation utilizing direct measurement of local and nonlocal spectral transfer rates. The computational work is accomplished with a spectral code containing up to 41×41 modes. The simulations examine two general situations: 1) the relaxation of the spectral energy distribution from an initial finite amplitude configuration in the absence of driving and damping; or 2) the evolution and saturation of the spectrum starting from infinitesimal amplitudes driven by unstable modes at low wavenumber and damped modes at high wavenumber. The first situation permits measurement of the nonlinear transfer rate independent of any particular wavenumber space configuration of sources and sinks, consistent with a chosen general spectrum shape. This situation also enables comparison of the stationary spectrum achieved by relaxation from an initial state with the predictions of equilibrium statistical mechanics. The second situation addresses the transfer characteristic of the most likely arrangement of sources and sinks. In this case, the net energy transfer in a saturated state (assuming one occurs) is not in question, since it will necessarily proceed from source to sink. Rather, these studies will determine the spectrum shape and examine nonlinear transfer in subranges of wavenumber space, focusing on the issues of direction, locality, and isotropy.

A striking visualization of the energy transfer in the saturated state may be produced by subjecting the saturated spectrum to a large, localized (in wavenumber space) perturbation and observing the subsequent relaxation of the spectrum to its original stationary configuration. Using this technique it is possible to determine the direction, locality, and isotropy of transfer is various subranges of the spectrum. This technique is the spectral analog of perturbative transport studies using modulated gas feed, heat pulse propagation, etc. to infer the locality, direction, and magnitude of spatial transport. As



such, it represents a reasonably accessible means of examining spectral transfer and cascades in experiment. Tests of this technique in simulated turbulence are reported herein.

The principal results of this chapter are now summarized. It is found that the polarization drift nonlinearity, acting alone, produces transfer that is both isotropic and local. Both properties follow directly from symmetries of the nonlinearity that are not present in the pure Kadomtsev-Pogutse model. With both nonlinearities acting simultaneously, subranges exist in which the transfer is virtually indistinguishable from the transfer of the dominant nonlinearity, if acting alone. Enstrophy generation is shown to be an effective indicator of the dominance of one nonlinearity over the other in these cases. If the maximum and minimum wavenumbers restrict the spectrum to a range where the E×B nonlinearity dominates throughout, enstrophy production is significant and increases on a timescale of a few eddy turnover times. Energy is transferred to high k in an anisotropic and nonlocal process characteristic of cases in which the polarization drift nonlinearity is entirely absent. If the spectrum is fixed to a range in which the polarization drift nonlinearity dominates throughout, enstrophy production is weak, with an e-folding time much larger than the eddy turnover time. Here a dual cascade is evident and transfer is isotropic and local. A third subrange exists and is accessed by determining the maximum and minimum wavenumbers so that both nonlinearities are roughly comparable over most of the spectrum (i.e., neither dominates). In this subrange, enstrophy production is moderate. Spectral transfer tends to be isotropic but retains a strong nonlocal component. In this situation, the local transfer develops an anisotropy which offsets the anisotropy of the nonlocal flow. When undriven, undamped turbulence evolves from an initial spectrum peaked at low k, the spectrum relaxes to a time-asymptotic state that remains peaked at long wavelength but has two subranges with distinct spectral falloff rates. The subranges separate at the point where the two nonlinearities are equal, with a slightly steeper slope in the high k range than in the low k range. Energy flow is toward higher k in the low k

subrange, with the converse true in the high k subrange. Present limits on k-space resolution make it difficult to extend this subrange to sufficiently large and small wavenumber to allow each nonlinearity to dominate at the extremes of the spectrum. It is anticipated that a larger subrange would accentuate, in each of its extremes, both the spectrum falloff disparity, and the differences in energy transfer direction.

For turbulence driven by unstable modes at long wavelength and damped by a hyperviscous damping at small wavelength, there is a noticeable difference in the wavenumber spectra which occur when either of the nonlinearities is acting alone and the spectrum which occurs in the subrange where when both nonlinearities are present and comparable. Whereas the spectrum of the E×B nonlinearity tends to be flat[2], it falls off toward high k with both nonlinearities. There is a discernible change in the falloff rate at the wavenumber where the two nonlinearities are equal, with a flatter falloff in the longer wavelength part of the spectrum. Distinct differences in the transfer rate are observed. In interpreting the transfer diagnostics, it appears that transfer is more sensitive to the cross-coupling of the nonlinearities than is the spectrum. Consequently, transfer behavior in either subrange with the unmodified signature of the dominant nonlinearity is more difficult to discern in the relatively small wavenumber space of the present simulations. Nevertheless, the following statements represent the qualitative behavior of the transfer with both nonlinearities in the steady state. In the E×B subrange, transfer is nonlocal. Above a critical wavenumber (related to the wavenumber where the two nonlinearities are equal), nonlocal transfer becomes weak relative to the total transfer. Thus, there is a polarization subrange with local transfer. A dual cascade is evident in this region of k space and accounts for the change in spectral index in crossing from one subrange to the other.

A novel and potentially important aspect of the interplay between the two nonlinearities is evident in the frequency spectra of individual modes. In the regime where both nonlinearities are comparable, a large shift of the frequency spectrum peak is observed.



For shorter wavelength modes, the shift is to higher frequency and can be many times the diamagnetic frequency. If either nonlinearity is absent, or contributes only weakly to the transfer dynamics, the shift is small. Theoretically, a spectrum shift is found in the renormalized response function[10]. This shift arises from the cross-coupling of the nonlinearities through the driven fluctuations. Since the E×B nonlinearity has one fewer spatial derivatives than the polarization drift nonlinearity, the cross-coupling term is 90° out of phase with the eddy damping decrement, and thus enters as a shift in the frequency spectrum peak (as opposed to a broadening). The occurrence of a frequency shift has a potentially significant impact on stability and transport[10], as well as the interpretation of fluctuation measurements. A derivation of the frequency shift and its effect on nonlinear stability, and therefore on the spectrum, is explored in Ref. 10. The self-consistent effect of the shift on mode fluctuation levels cannot be described by the present computational model, and will be addressed in future work.

The chapter is organized as follows. The model and its properties are presented and discussed in the next section. Section III describes the spectral transfer properties. To facilitate interpretation of the results, a subsection examines the locality and isotropy of transfer of the polarization drift nonlinearity. In Sec. IV, measurements of the frequency spectrum are presented and discussed. The conclusions are given in Sec. V.

## 4.2 Basic Model and Properties

In this section the basic model is presented and the dynamical coupling of the two nonlinearities is analyzed from the perspective of the mathematical structure of the model. This analysis examines integral invariants, symmetries in the structure of the nonlinearities, and the renormalized mode equations.

The model utilized for this study is a trapped particle fluid equation. This model couples the laminar dynamics of collisional trapped electrons with hydrodynamic ions through the quasineutrality condition. The derivation of this equation and a discussion of its details and limitations have been presented elsewhere[1,2,7,10]. In previous work, the polarization drift nonlinearity was neglected by considering only the long wavelength extreme of the spectrum[1,2]. Here, the polarization drift nonlinearity is included, yielding the model equation:

$$\frac{\partial \tilde{n}}{\partial t} + D\frac{\partial^2 \tilde{n}}{\partial y^2} + V_D\frac{\partial \tilde{n}}{\partial y} + \nu_i \tilde{n} - L_n D\,\nabla\frac{\partial \tilde{n}}{\partial y}\times\mathbf{z}\cdot\nabla\,\tilde{n}$$

$$+ \rho_s C_s\,\nabla\,\tilde{n}\times\mathbf{z}\cdot\nabla\,\rho_s^2\nabla^2\,\tilde{n} + \mu\nabla^4\,\tilde{n} = 0\,, \qquad (4.1)$$

where $\tilde{n}$ is the fluctuating ion density, $V_D = (cT_e/eB)L_n^{-1}$ is the diamagnetic drift velocity, $D = \varepsilon^{1/2}V_D^2(1+3\eta_e/2)/\nu_{eff,e}$ is a negative diffusivity describing the destabilization of DTEM modes by electron collisions, $\varepsilon$ is the trapped electron fraction, $\eta_e = d\ln T/d\ln n$ is the electron temperature gradient parameter, $\nu_i$ models long wavelength collisional damping, $\nu_{eff,e} = \nu_e/\varepsilon$, $\mu$ is the coefficient of the hyper-viscosity introduced to model strong damping at high k, $L_n$ is the density gradient scale length, $\rho_s = (cT_e/eB)/C_s$ is the ion gyroradius evaluated at the electron temperature, and $C_s = (T_e/m_i)^{1/2}$ is the ion sound speed. In keeping with the emphasis of the present chapter on nonlinear transfer effects, the linear polarization drift



term responsible for linear dispersion has not been included in Eq. (4.1). This allows significant savings in computation time. Inclusion of this term introduces straight-forward modifications of subsequent equations and relations (for example, the energy and enstrophy defined below are modified when dispersion is included[10]). However, the basic mechanisms governing transfer and the concepts describing this process are unaltered. Indeed, numerical solutions with and without the linear polarization term included were found to be qualitatively the same for the spectral ranges studied herein.

The first nonlinearity appearing in Eq. (4.1) is the E×B nonlinearity, arising from $\mathbf{v}_E\cdot\nabla\,\tilde{n}_i$, where $\mathbf{v}_E = -(c/B_0)\nabla\phi\times\mathbf{z}$ is the E×B drift. The second nonlinearity is the polarization drift nonlinearity, and arises from $n_0\nabla\cdot\mathbf{v}_p^{(1)}$, where $\mathbf{v}_p^{(1)} = B_o^{-1}(m_ic/e)\mathbf{z}\times\mathbf{v}_E\cdot\nabla\mathbf{v}_E$ is the nonlinear polarization drift. The E×B nonlinearity requires a nonadiabatic electron response (provided by the trapped electrons), whereas the polarization drift nonlinearity derives from the ion polarization drift.

From Eq. (4.1), it is apparent that, by virtue of its additional spatial derivative, the polarization drift nonlinearity dominates the E×B nonlinearity at very short wavelengths. The converse holds at long wavelengths. The nominal crossover point is given by the wavenumber at which the two nonlinearities are equal. Assuming rough isotropy, so that $\nabla_\perp \approx \partial/\partial y$, this wavenumber is given by $k\rho_s \approx \delta = C_s/L_n\nu_{eff,e} \equiv k_o\rho_s$. Because the nonlinearities are characterized, not by a single spatial scale, but by a triad interaction consisting of three waves of differing wavelengths, it is more realistic to identify a region, centered about the crossover wavenumber, in which the two nonlinearities are comparable, rather than to speak of a single wavenumber at which the two are equal.

Due to the presence of the E×B nonlinearity, a single quadratic invariant, the energy, is admitted by Eq. (4.1) in the absence of driving and damping, i.e.,

$$\frac{1}{2}\frac{\partial}{\partial t}\int \tilde{n}^2\, d^2x \;=\; D\int\left(\frac{\partial\tilde{n}}{\partial y}\right)^2 d^2x \;-\; \nu_{eff,i}\int \tilde{n}^2\, d^2x \;-\; \mu\int(\nabla^2\tilde{n})^2\, d^2x, \qquad (4.2)$$

where $E = \int \tilde{n}^2\, d^2x$ is the energy and the terms on the right hand side represent the dissipative source (inverse electron damping) and sinks (ion-ion collisions and hyper-viscosity). Only if the E×B nonlinearity is absent, is there a second invariant, the enstrophy, defined by $\Omega = \int |\nabla\tilde{n}|^2\, d^2x$. From Eq. (4.1), the enstrophy evolution is given by

$$\frac{1}{2}\frac{\partial}{\partial t}\int |\nabla\tilde{n}|^2\, d^2x \;-\; D\int\left|\nabla\frac{\partial\tilde{n}}{\partial y}\right|^2 d^2x \;+\; \nu_{eff,i}\int|\nabla\tilde{n}|^2\, d^2x \;+\; \mu\int(\nabla^3\tilde{n})^2\, d^2x$$

$$= \;-L_n D\int\nabla^2\tilde{n}\,\nabla\frac{\partial\tilde{n}}{\partial y}\times z\cdot\nabla\,\tilde{n}\; d^2x\;. \qquad (4.3)$$

The term on the right hand side describes the generation or destruction of enstrophy associated with the conservative transfer of energy by the E×B nonlinearity. As previously indicated, the E×B nonlinearity in isolation transfers energy to short wavelengths, given a spectrum which is peaked at long wavelength or flat, and therefore drives robust production of enstrophy[2]. This term is present even in spectral ranges where the polarization drift nonlinearity dominates ($k > k_o$). Consequently, enstrophy is not conserved even when the E×B nonlinearity is weak compared to the polarization drift nonlinearity. However, in such a case, the E×B nonlinearity accounts for proportionately less of the total energy transfer. Because enstrophy production is tied to energy transfer by the E×B nonlinearity, it can be expected that the importance of enstrophy production in the cascade dynamics diminishes for $k > k_o$. This fact is also apparent in comparing the enstrophy production rate



$$\gamma_\Omega \;=\; \left[\frac{1}{2}\int|\nabla\tilde{n}|^2\, d^2x\right]^{-1} L_n D\int\nabla^2\tilde{n}\,\nabla\frac{\partial\tilde{n}}{\partial y}\times z\cdot\nabla\,\tilde{n}\; d^2x\;, \qquad (4.4)$$

with the nonlinear interaction rate or eddy turnover rate. For $k < k_o$, the eddy turnover rate is controlled by the dominant E×B nonlinearity and can be expected to be comparable to the enstrophy generation rate. For $k > k_o$ the eddy turnover rate is controlled by the larger polarization drift nonlinearity, while the enstrophy generation rate is tied to the weaker E×B nonlinearity. Consequently, there will be little enstrophy generation on the nonlinear interaction or nonlinear transfer timescale. This fact suggests a convenient diagnostic of the strength of the polarization drift nonlinearity in the form of the enstrophy dissipation rate $\hat{\gamma}_\Omega$ normalized to the mean oscillation time of an intermediate scale mode. Results from the simulations will show $\hat{\gamma}_\Omega^{-1}$ to range from 2-3 in the E×B subrange to hundreds in the polarization subrange.

Energy transfer that is anisotropic and nonlocal in wavenumber space is a robust feature of the E×B nonlinearity, but at variance with the conventional picture of the cascade process. On the other hand, the polarization drift nonlinearity is of the same form as the advective nonlinearity of the vorticity evolution equation of Navier-Stokes turbulence. Therefore, it is reasonable to postulate that the polarization drift nonlinearity produces transfer which is local and isotropic. It is possible to associate these features with symmetries in the structure of the nonlinear coupling. These symmetries are most transparent in the Fourier representation of the nonlinearities. Transforming Eq. (4.1), the evolution of mode amplitudes is given by

$$\frac{\partial\tilde{n}_k}{\partial t} \;-\; Dk_y^2\,\tilde{n}_k \;+\; iV_D\,k_y\,\tilde{n}_k \;+\; \nu_{eff,i}\,\tilde{n}_k \;+\; \mu k^4\tilde{n}_k \;+\; N_k^{(E\times B)} \;+\; N_k^{(Pol)} \;=\; 0, \qquad (4.5)$$

where

$$N_k^{(E\times B)} = \frac{i}{2} L_n D \sum_{k'} \mathbf{k}\times\mathbf{k}'\cdot\mathbf{z}\,[k_y' - (k_y - k_y')]\,\tilde{n}_{k'}\,\tilde{n}_{k-k'} \equiv \sum_{k'} \chi^{(E\times B)}_{k,k'}\,\tilde{n}_{k'}\,\tilde{n}_{k-k'}, \tag{4.6}$$

$$N_k^{(Pol)} = \frac{1}{2}\rho_s^3 C_s \sum_{k'} \mathbf{k}\times\mathbf{k}'\cdot\mathbf{z}\,[(k_\perp - k_\perp')^2 - k_\perp'^2]\,\tilde{n}_{k'}\,\tilde{n}_{k-k'}.$$

$$\equiv \sum_{k'} \chi^{(Pol)}_{k,k'}\,\tilde{n}_{k'}\,\tilde{n}_{k-k'}. \tag{4.7}$$

From Eqs. (4.6) and (4.7), the lack of anisotropy in the E×B nonlinearity is evident in the appearance of the factor $k_y' - (k_y - k_y')$, whereas the polarization drift nonlinearity is manifestly isotropic.

Differences between the two nonlinearities regarding the locality of transfer in wavenumber space can also be deduced from Eqs. (4.6) and (4.7). From Eq. (4.6), the E×B coupling is proportional to $k_y' - (k_y - k_y') = 2k_y' - k_y$. For a nonlocal triad consisting of a long wavelength fluctuation $k_y$ interacting with short wavelength fluctuations $k_y'$ and $k_y - k_y'$ ($k_y \ll k_y', k_y - k_y'$), this factor is proportional to the large wavenumber $2k_y'$. For local triads ($k_y \sim k_y' \sim k_y - k_y'$), $2k_y' - k_y \sim k_y$, i.e., the factor is proportional to the small wavenumber. Clearly, the E×B coupling favors nonlocal interaction. Note that this predilection for nonlocal coupling is not isotropic, but applies solely to the displacement in the $k_y$ direction. By contrast, the comparable coupling factor in the polarization drift nonlinearity is $(k_\perp - k_\perp')^2 - k_\perp'^2 \sim k_\perp^2$ (spectrum symmetries eliminate the contribution $-2k_\perp k_\perp'$). Because this factor involves the squares of the wavenumbers $k_\perp'$ and $k_\perp - k_\perp'$, the large wavenumber $k_\perp'^2$ cancels and the factor is proportional to the small wavenumber $k_\perp^2$. Hence, the polarization drift nonlinearity, unlike the E×B nonlinearity, has no special weighting that favors nonlocal coupling (beyond the universal factors $\mathbf{k}\times\mathbf{k}'\cdot\mathbf{z}$ and the spectral energy distribution $\tilde{n}_{k'}\,\tilde{n}_{k-k'}$). The symmetries in the nonlinear couplings



responsible for this distinction between the E×B and polarization drift nonlinearities derive from the nonadiabatic electron response of the E×B nonlinearity ($\phi \sim \delta\partial\tilde{n}_e/\partial y$) and the adiabatic electron response of the polarization drift nonlinearity ($\phi \sim \tilde{n}$). These symmetries carry directly over to the energy transfer rates and give rise to a nonlocal transfer rate by the E×B nonlinearity which dramatically exceeds the local transfer rate, while for the polarization drift nonlinearity, local and nonlocal transfer rates tend to be comparable.

The existence of the frequency spectrum shift is also apparent from Eqs. (4.6) and (4.7) under iteration of the $\tilde{n}_{k-k'}$ factor in each expression. In the iteration procedure, $\tilde{n}_{k-k'}$ is replaced by the formal solution of Eq. (4.5) (with $k \to k-k'$). This solution is obtained by placing $N^{(E\times B)}_{k-k'} + N^{(Pol)}_{k-k'}$ on the right hand side of Eq. (4.5) and inverting the temporal operator of the left hand side. Under standard statistical closures, the sum in the $N_{k-k'}$ terms of Eq. (4.5) is restricted to the directly interacting triplet giving

$$\tilde{n}_{k-k'} = -2\Delta\omega_{k-k'}^{-1}\left[\chi^{(Pol)}_{k-k',-k'} + \chi^{(E\times B)}_{k-k',-k'}\right]\tilde{n}_{-k'}\,\tilde{n}_k\,, \tag{4.8}$$

where $\Delta\omega_k^{-1}$ is the inversion of the first four terms of Eq. (4.5). Substitution of Eq. (4.10) into Eqs. (4.6) and (4.7) gives the coherent nonlinear response from each of the nonlinearities. Combining, $\tilde{n}_k$ evolves according to

$$\frac{\partial\tilde{n}_k}{\partial t} - Dk_y^2\,\tilde{n}_k + \frac{iV_D}{2}k_y\,\tilde{n}_k + \nu_{eff,i}\,\tilde{n}_k + \mu k^4\tilde{n}_k - \nu_k^{nl}\,\tilde{n}_k = 0, \tag{4.9}$$

where

$$\nu_k^{nl} = 2\sum_{k'}\left[\chi^{(Pol)}_{k,k'} + \chi^{(E\times B)}_{k,k'}\right]\left[\chi^{(Pol)}_{k-k',-k'} + \chi^{(E\times B)}_{k-k',-k'}\right]|\tilde{n}_{-k'}|^2 / \Delta\omega_k^{-1}. \tag{4.10}$$

Note that $\chi^{(Pol)}$ is 90° out of phase with $\chi^{(E\times B)}$. Consequently, if the propagator $\Delta\omega_k^{-1}$ is treated as purely real the diagonal terms $\chi^{(E\times B)}_{k,k'}\,\chi^{(E\times B)}_{k-k',-k'}$ and $\chi^{(Pol)}_{k,k'}\,\chi^{(Pol)}_{k-k',-k'}$ produce coherent damping while the off-diagonal terms $\chi^{(E\times B)}_{k,k'}\,\chi^{(Pol)}_{k-k',-k'}$ and $\chi^{(Pol)}_{k,k'}\,\chi^{(E\times B)}_{k-k',-k'}$ produce a phase oscillation. Accordingly, the diagonal terms are related to the spectrum linewidth[11] while the off-diagonal terms contribute to the frequency of the spectrum peak[10]. Physically, the frequency shift arises from the interaction of a driven fluctuation $\tilde{n}_{k-k'}$ which is 90° out of phase with the test and beat waves at k and k'. Clearly it is cross-coupling of modes driven by each of the two nonlinearities that produces the shift, with no shift occurring if either of the nonlinearities is absent.

Substituting for $\chi^{(E\times B)}$ and $\chi^{(Pol)}$ in Eq. (4.10), it is seen that

$$\text{Im}\,\nu_k^{nl} = -L_n D\rho_s^3 C_s \sum_{k'} (\mathbf{k}\times\mathbf{k}'\cdot\mathbf{z})^2\, k_y\,[2k_\perp^2 - k_\perp'^2]\,|n_{k'}|^2 \,/\, \text{Re}(\Delta\omega_k^{-1}), \qquad (4.11)$$

corresponding to an upshift for intermediate modes $k_\perp^2 \gtrsim \langle k_\perp^2\rangle$. The qualitative features of this predicted shift, specifically its sign and its $k_y$ dependence, will be examined in Sec. IV.

### 4.3 Spectral Transfer

In this section the spectral flow of energy over the spectrum range is described. The transfer properties are inferred from the time evolution of the wavenumber spectrum and the time evolution of the energy and enstrophy transfer rates from (or to) selected bands in wavenumber space[2]. The energy and enstrophy transfer rates due to the E×B nonlinearity are given in Ref. 2. With the addition of the polarization drift nonlinearity, the evolution of the energy and enstrophy of the mode k is described by



$$\frac{1}{2}\frac{\partial|\tilde{n}_k|^2}{\partial t} - Dk_y^2\,|\tilde{n}_k|^2 + \nu_{eff,i}\,|\tilde{n}_k|^2 + \mu k^4\,|\tilde{n}_k|^2 = T_k, \qquad (4.12)$$

$$\frac{1}{2}\frac{\partial\, k^2|\tilde{n}_k|^2}{\partial t} - Dk_y^2\,k^2|\tilde{n}_k|^2 + \nu_{eff,i}\,k^2|\tilde{n}_k|^2 + \mu k^6\,|\tilde{n}_k|^2 = U_k, \qquad (4.13)$$

where

$$T_k = L_n D\,\text{Im}\sum_{k'}(\mathbf{k}\times\mathbf{k}'\cdot\mathbf{z})k_y'\tilde{n}_{k'}\tilde{n}_{k-k'}\tilde{n}_k^*$$
$$- \rho_s^3 C_s\,\text{Re}\sum_{k'}\mathbf{k}\times\mathbf{k}'\cdot\mathbf{z}\;(k_\perp - k_\perp')^2\;\tilde{n}_{k'}\,\tilde{n}_{k-k'}\tilde{n}_k^* \qquad (4.14)$$

and

$$U_k = L_n D\,\text{Im}\sum_{k'}(\mathbf{k}\times\mathbf{k}'\cdot\mathbf{z})k^2k_y'\tilde{n}_{k'}\tilde{n}_{k-k'}\tilde{n}_k^*$$

$$- \rho_s^3 C_s\,\text{Re}\sum_{k'}\mathbf{k}\times\mathbf{k}'\cdot\mathbf{z}\;k^2(k_\perp - k_\perp')^2\;\tilde{n}_{k'}\,\tilde{n}_{k-k'}\tilde{n}_k^* \qquad (4.15)$$

are respectively the energy and enstrophy transfer rates from the mode k. Note that summing $U_k$ over all k yields the enstrophy production rate, Eq. (4.4). Summing $T_k$ and $U_k$ over a band of wavenumbers (typically with $k_y$ = constant or $k_x$ = constant) yields the transfer from the given band. Moreover, by selecting only certain k' values in the sums, the transfer rates can be formulated to identify transfer between coupled triplets with wavenumbers which span more than a significant fraction of the spectrum (typically 1/3 to 1/2). This allows a quantitative assessment of the relative magnitudes of local and nonlocal transfer. Before describing the flow induced by the E×B and polarization drift nonlinearities together, it is useful to establish a reference in terms of the flow properties under each of the nonlinearities in isolation.

## A. Transfer Properties of Each Nonlinearity in Isolation

Spectral flow by the E×B nonlinearity was described in Ref. 2 and is briefly summarized. In absence of driving and damping, spectra initially peaked at low k relax to the equipartitioned state predicted by statistical mechanics. The relaxation is accomplished by a prompt nonlocal flow to the largest $k_y$ bands (defined by $k_y \sim k_{y\,max}$), producing, within a few eddy turnover times, spectra with peaks at high $|k_y|$. Subsequent sloshing motion excites the intermediate wavenumbers. The energy transfer rate between nonlocally displaced $k_y$ bands exceeds the transfer between closely neighboring (local) bands by an order of magnitude. Local and nonlocal transfer rates in the $k_x$ direction are comparable. In both directions, transfer proceeds to high k when the spectrum is initially peaked. When driven by modes at low k and damped by modes at high k with an intermediate inertial range, the stationary wavenumber spectrum is remarkably flat over the inertial range, falling off only at the damped modes. This dramatic deviation of the spectrum from that predicted by similarity arguments is another indication of strong nonlocal transfer.

When the polarization drift nonlinearity is the sole nonlinear coupling mechanism, the equilibrium spectrum peaks at low k. Figure 4.1 shows the equilibrium spectrum (no driving or damping) reached from an initial spectrum with $|n_k|^2 \sim k^{-3}$ for 41×41 modes.



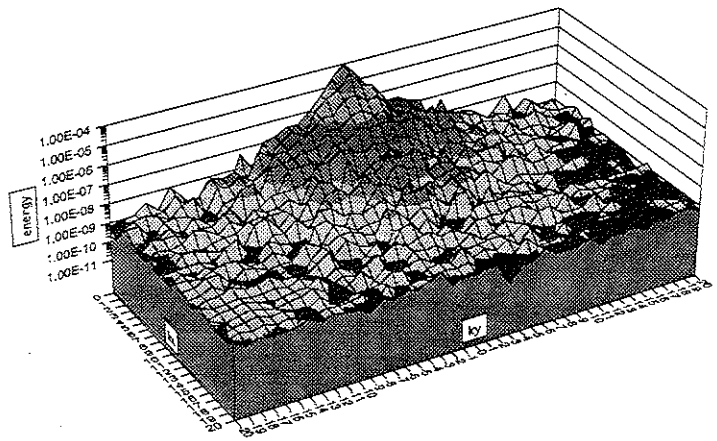


Fig. 4.1 Equilibrium spectrum of undriven/undamped turbulence with the only polarization drift nonlinearity. This spectrum is the relaxed state that has evolved from an initial spectrum with a fall-off index of a = -4. The initial spectrum is identical to the initial spectrum shown in Fig. 1 of Ref. 2.

Relative to the initial spectrum, the equilibrium spectrum is more peaked at lower wavenumbers and flatter at large wavenumbers, indicating an inverse cascade of energy to low k, but also some energy transfer to high k. Indeed, the energy transfer rate reveals such a transfer pattern. From Fig. 4.2, the transfer rate $T_k$ is positive at the lowest wavenumbers and negative somewhat higher, indicating a primary transfer into the lowest wavenumbers from the higher band. The higher band also engages in a smaller subsidiary transfer to even higher wavenumbers, as seen in the positive peak just above the negative feature. In the upper half of the spectrum there is little net transfer above the noise level as defined by the standard deviation of the scatter in the transfer history.

The transfer to low k is consistent with the dynamical constraints imposed by the conservation of energy and enstrophy. The smaller forward cascade might appear to be a violation of these constraints. However, as shown elsewhere[12], some energy must be transferred in the forward direction in order to generate sufficient enstrophy to compensate for the enstrophy destroyed by the inverse energy cascade. The amount of energy transferred to high k decreases to zero as the largest wavenumber goes to infinity. For the rather small wavenumber spaces considered here, however, this energy can be appreciable. This process contributes to the existence of two fall-off rates in the equilibrium spectrum, a feature not anticipated by equilibrium statistical mechanics.

As discussed in the previous section, the symmetries of the coupling coefficient of the polarization drift nonlinearity suggest that nonlocal transfer should be no larger than the local transfer. In fact, as evidenced by Fig. 4.3, the local transfer rate from the band with negative transfer in Fig. 4.2 exceeds the nonlocal transfer rate by a factor of two or three. Moreover, the transfer rates in the $k_x$ and $k_y$ directions are found to be comparable to within a factor of two, so that transfer is isotropic, as anticipated from the form of the nonlinearity.



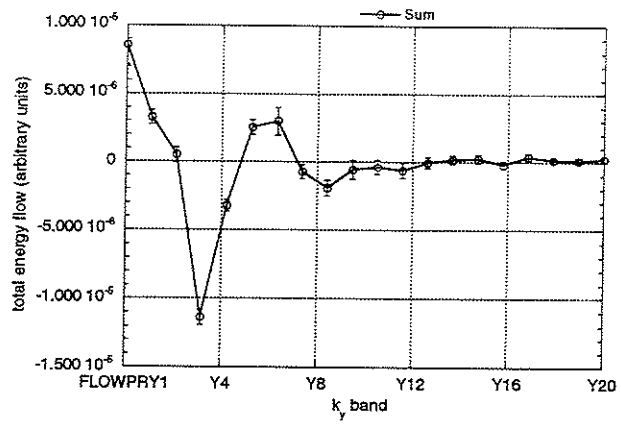


Fig. 4.2 Time averaged energy transfer rate during the relaxation leading to the spectrum of Fig. 4.1. The error bars indicate the standard deviation from the mean value plotted in the figure.

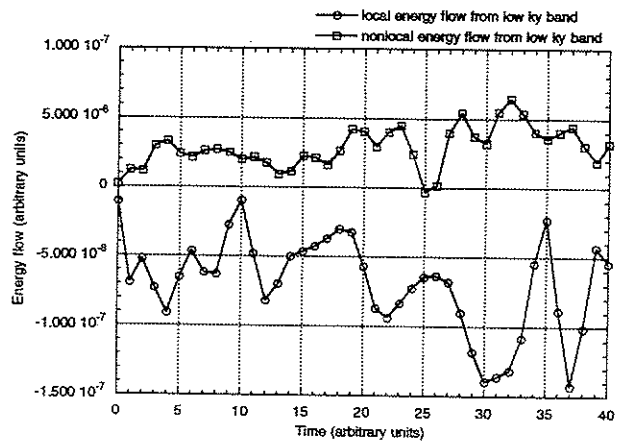


Fig. 4.3 Local and nonlocal energy transfer rates from the band with negative transfer (energy outflow) during the relaxation leading to the spectrum of Fig. 4.1.

## B. Transfer Properties with Both Nonlinearities

With both nonlinearities present, the transfer of energy and enstrophy is characterized by three general principles. These are now presented. 1) Excepting the effects represented by the other two principles, each nonlinearity transfers energy and enstrophy throughout the spectrum as it would if acting alone. Because the E×B nonlinearity dominates at low k ($k<k_o$), the net transfer in the long wavelength subrange is similar to that produced by the E×B nonlinearity alone, i.e., the transfer is to high k with a strong nonlocal component in $k_y$. While the E×B dynamics are similar in the high k subrange, the transfer they produce is small by comparison with that of the polarization drift nonlinearity, which drives a dual cascade via local, isotropic transfer.

2) With the two nonlinearities acting together, the spectral distribution of energy $|\tilde{n}_k|^2$ is modified relative to its configuration with a single nonlinearity. Since $T_k$ and $U_k$ depend on amplitudes $\tilde{n}_{k'}$, $\tilde{n}_{k-k'}$, and $\tilde{n}_k^*$ [Eqs. (14) and (15)], the transfer is also modified relative to its behavior for a single nonlinearity. In each of the subranges, the spectra, both the equilibrium and the driven/damped stationary spectrum, tend to look like the spectra of the dominant nonlinearity, when acting alone. Consequently, transfer well within each subrange continues to be characterized by the transfer of the dominant nonlinearity in isolation. Significant modifications are thus restricted to the crossover region around $k_o$ where the two nonlinearities are comparable.

3) The spectrum shift, Eq. (4.10), affects the transfer in three ways. a) The spectrum shift contributes to the three wave phase decorrelation just as a linear frequency mismatch contributes to the decorrelation. The shift mismatch $\nu_{k'} + \nu_{k-k'} - \nu_k$ is finite, due to the dispersion of the shift, and more pronounced for nonlocal triads than local triads. The shift-induced phase decorrelation will therefore limit nonlocal transfer. However, this effect



is a factor only in the crossover region. Outside this region, the shift-induced decorrelation is small relative to the phase decorrelation caused by the eddy damping of the dominant nonlinearity. b) In a strong turbulence regime, the frequency shift in the response function of the driven fluctuation gives rise to a nonresonant contribution [off-diagonal terms in Eq. (4.10)] to the eddy damping which can, in effect, be comparable to the resonant (diagonal) contribution. In such a case, the transfer, or equivalently, the energetics, is directly affected by the shift. c) The frequency shift affects the nonlinear stability (or free energy extraction) by modifying Re $\omega$ in the eigenmode potential. This effect is most simply displayed in typical drift wave growth rates, which contain the factor $\omega-\omega_*$, with instability for $\omega < \omega_*$. The role of the frequency shift on nonlinear stability, as it pertains to dissipative drift waves, is explored in Ref. (10). As noted previously, this effect is not present in the present computational model but will be examined computationally in the next chapter.

It should be apparent from the above that the transfer dynamics of the crossover region is the most difficult to analyze. Figures (4.4) and (4.5) depict the evolution of the spectrum in the crossover region under non driven, non damped relaxation, with $k_o$ slightly larger than the median k. Figure (4.4) is the spectrum a few eddy turnover times after commencing the relaxation from a initial $k^{-3}$ distribution. Significant nonlocal transfer to high k modes is evident in the peaks which have developed at high |k|. This peaking feature is isotropic in k and contrasts with the peaks that occur at high $k_y$ (but not $k_x$) at comparable times with the E×B nonlinearity alone. The nonlocal transfer is an unmistakable signature of the E×B nonlinearity, whereas the tendency toward isotropy is a feature of the polarization drift nonlinearity. Isotropization of the high k peaks is accomplished by transfer modifications induced by the spectrum distribution (principle 2). The final equilibrium spectrum, Fig. (4.5), is mostly flat with a narrow peak at low k. The peak is produced by the inverse cascade of the polarization drift nonlinearity. It is more narrow

than the peak which occurs with the polarization drift nonlinearity alone [Fig. (4.1)] because of the competing transfer to high k produced by the E×B nonlinearity.



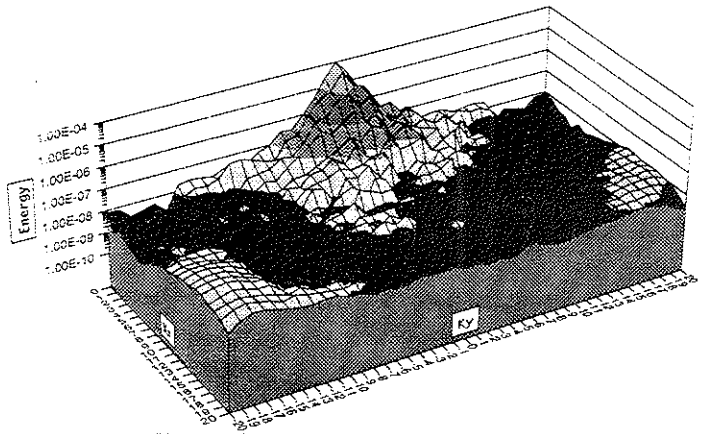


Fig. 4.4 Spectrum of undriven/undamped turbulence with both the E×B and polarization drift nonlinearities (E×B nonlinearity dominates weakly). This is the spectrum that occurs several eddy turnover times after the initiation of evolution.

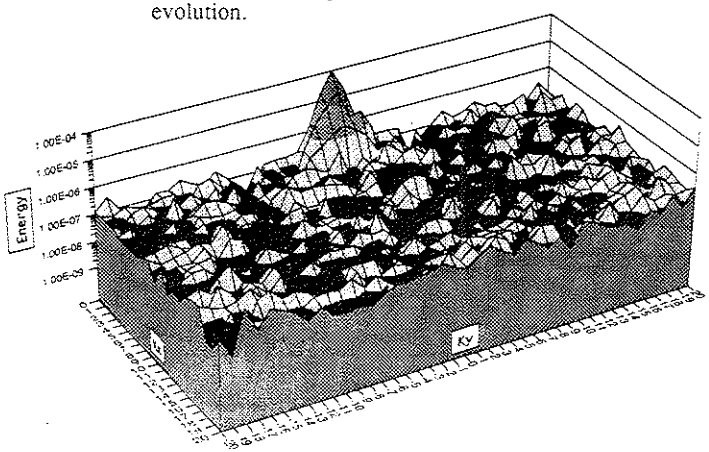


Fig. 4.5 Final (time asymptotic) spectrum for undriven/undamped turbulence with both the E×B and polarization drift nonlinearities (E×B nonlinearity dominates weakly).

The flat portion is attributable to the E×B nonlinearity, which tends to drive the spectrum to equipartition. In this regard, it is interesting to note that the spectrum predicted by equilibrium statistical mechanics is the equipartitioned spectrum, since enstrophy is not conserved in this case. Statistical mechanics is unable to account for the inverse cascade because the E×B nonlinearity breaks enstrophy invariance even though significant enstrophy conserving transfer by the polarization drift nonlinearity occurs.

The degree of nonconservation of enstrophy, or more precisely, the rate of enstrophy generation, is a measure of the extent of energy transfer to high k by the E×B nonlinearity. In Fig. (4.6), the enstrophy evolution under non driven, non damped relaxation is plotted as a function of time for four spectrum configurations. These are the E×B and polarization subranges, in which $k_o$ falls at either the extreme low end (polarization subrange) or high end of the spectrum (E×B subrange); and two crossover region configurations, one in which $k_o$ is slightly to the right of center and one in which $k_o$ falls slightly to the left of center. In the E×B subrange there is significant enstrophy generation, and the timescale of enstrophy production is a few eddy turnover times. These facts indicate that the transfer dynamics is dominated by the forward cascade of the E×B nonlinearity. In contrast, the small net enstrophy increase and long (many eddy turnover times) generation timescale evident in the polarization subrange, indicate that the forward transfer by the E×B nonlinearity is of minor overall importance. In the crossover region, the same general trend is evident, with the net enstrophy production and production rate being roughly proportional to the fraction of k space below $k_o$. Taken together, these results suggest that in a spectrum encompassing both subranges, there is significant non-enstrophy conserving forward transfer from $k_{min}$ to $k_o$ and somewhat beyond, but that its magnitude relative to the net transfer rapidly diminishes well beyond $k_o$.



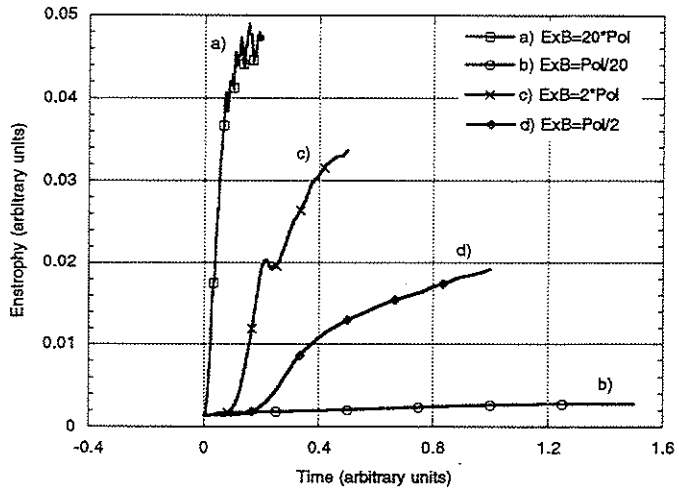


Fig. 4.6 Time evolution of the enstrophy for four spectrum configurations: a) spectrum primarily in the E×B subrange, b) spectrum primarily in the polarization drift subrange, c) spectrum of crossover region with slightly more of k-space below the crossover wavenumber ko, d) spectrum of crossover region with slightly more of k-space above the crossover wavenumber ko. The enstrophy production rate is only comparable to the nonlinear interaction (eddy turnover) time in a) and c).

It is obviously important to examine the transfer dynamics in a saturated spectrum achieved through the dynamic balance of sources, sinks and transfer. For dissipative trapped ion convective cells, turbulence is driven at long wavelength (below $k_o$); at longer wavelengths still, ion-ion collisions provide a sink. The robust forward transfer of the E×B nonlinearity can be expected to carry energy beyond $k_o$. At some scale the energy is dissipated ( high k damping). While the existence of a steady state requires a net transfer from sources to sinks, the spectral energy distribution allowing saturation, and the extent of nonlocality and anisotropy in the transfer process must still be determined. In Fig. (4.7), the stationary spectrum is plotted for a configuration in which $k_o$ is centrally located between $k_{min}$ and $k_{max}$. Unstable modes occur in the low k region with a stable band at the lowest wavenumbers, representing ion-ion collisional damping. A broad inertial range extends to the highest wavenumbers where a hyperviscous damping provides a high k sink. Figure (4.7) reveals a spectrum with sharp peaks at the wavenumbers of the driven modes, surrounded by a plateau-like region of roughly elliptical shape. Beyond the plateau, the spectrum falls off sharply to the edge. The edge of the plateau corresponds roughly to the crossover point $k_o$. Comparison with the stationary spectrum of the E×B nonlinearity alone, which is flat all the way out to the viscously damped modes[2], indicates that significant nonlocal transfer to high k is the dominant process in the plateau region. Anisotropic transfer is the cause of the elliptical shape of the plateau. There is also transfer to the collisionally damped modes at low k, producing a steep drop-off in going below the driven wavenumbers. This spectrum indicates that the characteristics of the E×B and polarization drift nonlinearities effectively dominate in the appropriate ranges of the spectrum.



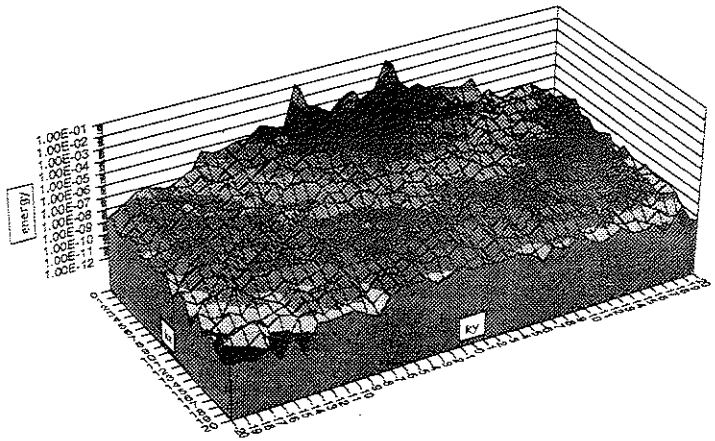


Fig. 4.7 Spectrum of turbulence driven at long wavelength and damped by a hyperviscosity at short wavelength with an inertial range in the intermediate modes. There is also damping at the smallest wavenumbers, representing the effect of ion-ion

The transfer that underlies the spectrum in Fig. (4.7) is graphically visualized by applying an energy impulse at a specified wavenumber and observing the subsequent spectrum evolution as it relaxes back to its stationary state. To interpret this diagnostic in cases with both the nonlinearities present, it is helpful to apply it first to the saturated spectra of each nonlinearity in isolation. Figure (4.8) illustrates the relaxation of the spectrum under the E×B nonlinearity starting from the stationary spectrum with a large impulse (dark black in color) localized to a rectangular annulus in k space. Shortly after the impulse, prompt nonlocal transfer carries the energy to all parts of the spectrum. The filamentary structure indicates anisotropic transfer. In the last frame, the stationary spectrum is re-established at the original level. (To optimize contrast, the color scale in the first frames is different from that of later frames.) In contrast, the same experiment with the polarization drift nonlinearity produces a slower diffusive-like spreading of energy in k space, with the inverse cascade restoring the central peak, as shown in Fig. (4.9). When both nonlinearities are combined, as in Fig. (4.10), an impulse in the E×B subrange produces prompt nonlocal but isotropic transfer throughout the spectrum. Subsequent frames show a slower diffusive inverse cascade which restores the central peak. In Fig. (4.11), the impulse is applied at high k in the polarization subrange. The spreading of energy is slower and diffusive throughout the evolution. It is readily apparent that when both nonlinearities are combined, the relaxation of the impulse (at least initially) carries the unmistakable signature of the subrange in which the impulse originates.



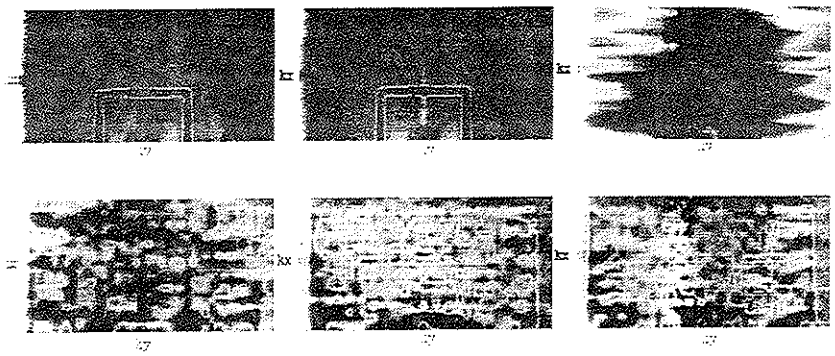

Fig. 4.8 Evolution of the spectrum with driving and damping (under the E×B nonlinearity only) subsequent to a impulse of energy localized to a rectanguiar annulus in k space. Time goes sequentially from left to right and down. The 0,0 mode is at the center bottom of each snap shot and the scale is from dark gray (near center) signifying the most energy through white to black signifying the least energy.

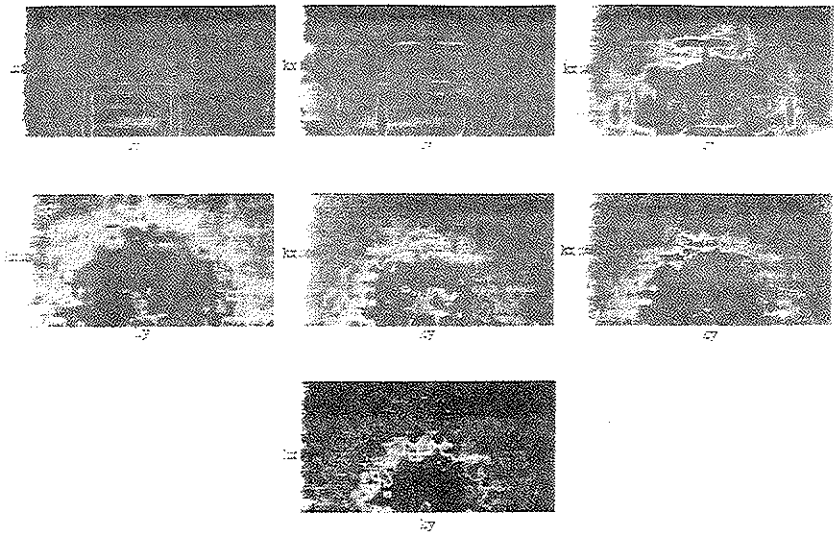

Fig. 4.9 Evolution of the spectrum with driving and damping (under the polarization drift nonlinearity only) subsequent to a impulse of energy localized to a rectangular annulus in k space. Time goes sequentially from left to right and down. The 0.0 mode is at the center bottom of each snap shot and the scale is from dark gray (near center) signifying the most energy through white to black signifying the least energy.



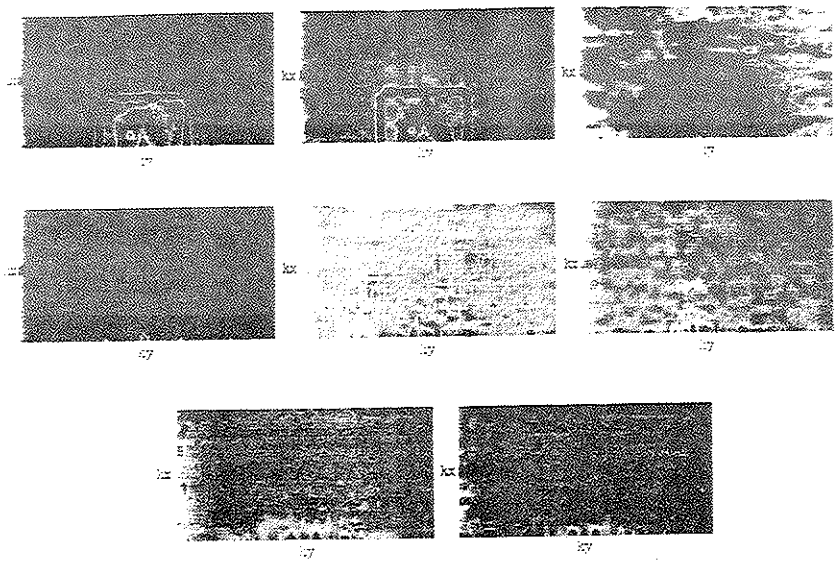

Fig. 4.10 Evolution of the spectrum with driving and damping and both nonlinearities subsequent to a impulse of energy localized to a rectangular annulus at low k in the ExB subrange. Time goes sequentially from left to right and down. The 0.0 mode is at the center bottom of each snap shot and the scale is from dark gray (near center) signifying the most energy through white to black signifying the least energy.

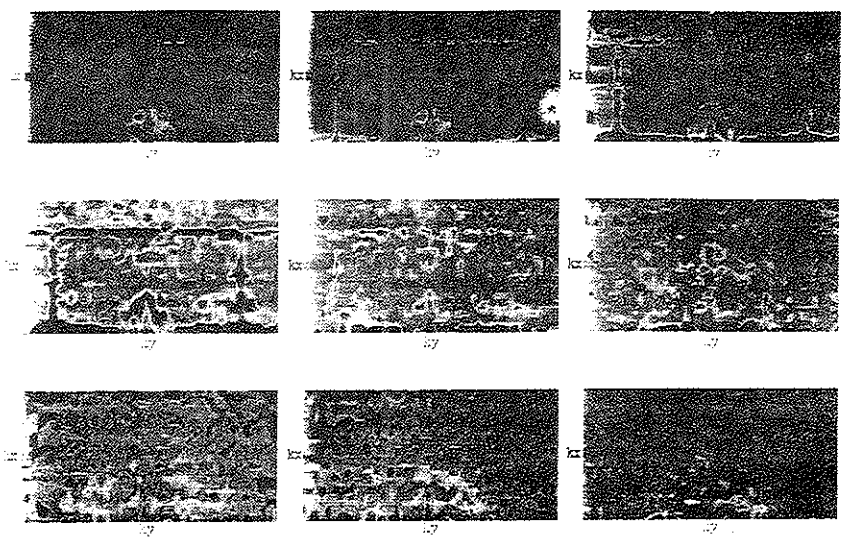

Fig. 4.11 Evolution of the spectrum with driving and damping and both nonlinearities subsequent to a impulse of energy localized to a rectangular annulus at high k in the polarization drift subrange. Time goes sequentially from left to right and down. The 0.0 mode is at the center bottom of each snap shot and the scale is from dark gray (near center) signifying the most energy through white to black signifying the least energy.



## 4.4 Frequency Spectrum

The frequency spectrum is generally taken to signify the frequency dependent part of the power spectrum $S_k(\omega)$, for k fixed. In terms of the solution of a spectral representation of the model, i.e., the solution of Eq. (4.5), the frequency spectrum is just the Fourier transform of the temporal autocorrelation function, obtained from the time history of the mode k,

$$S_k(\omega) = \frac{1}{\sqrt{2\pi}} \int d\tau \exp(-i\omega\tau) \langle \tilde{n}_k(t')\tilde{n}_k^*(t'+\tau) \rangle, \tag{4.16}$$

where $\langle\ \rangle$ is the average over a suitably chosen ensemble.

In this section, frequency spectra are examined for a variety of modes and conditions with the objective of establishing qualitatively the existence of a frequency shift induced by the coupling of the E×B and polarization drift nonlinearities. From Eq. (4.9), it is apparent that the spectrum will contain a shift, given by Eq. (4.11). However, it should also be noted that the spectrum is a correlation, and the correlation of $\tilde{n}_k(t')$ with $\tilde{n}_k^*(t'+\tau)$ for small time differences $\tau$ is not captured by simply substituting into Eq. (4.16) the solution of Eq. (4.9), an approximate one-time, one-point equation[13]. Specifically, Eq. (4.9) represents the coherent response, but does not capture the incoherent emission associated with small scale, short time correlation. While the *dissipative part* of the incoherent emission ultimately balances the coherent decay in a steady state inertial range, and thus provides a link between the two components (and a way of determining the linewidth[14]), the reactive component of the incoherent emission is not related to the reactive part of the coherent response, Im $\nu_k$, in any simple way. Thus, it is simplistic to expect that the frequency spectrum, and the frequency shift in particular, are totally described by the solution of Eq. (4.11)[15]. Indeed, as will be apparent, the frequency spectra are complicated and not readily interpreted as a

localized, single peaked function. Nevertheless it *is* apparent that a large shift in the mean frequency of the spectrum occurs, that the shift requires the presence of both nonlinearities, and that other features of Eq. (4.11) are present.

The frequency spectra of both saturated (driven/damped) turbulence and relaxing (no driving or damping) turbulence have been determined. In general, the spectra of saturated turbulence are much broader than the spectra of relaxing turbulence, and the shifts relative to the linewidths are accordingly less pronounced. The spectra of relaxing turbulence, on the other hand, are not stationary. Computing the spectra from a time history which covers the entire relaxation phase, these spectra have a feature associated with the relaxation. Since the relaxation takes many eddy turnover times, and the timescale of the shift is typically less than an eddy turnover time, the nonstationarity of the spectrum poses no particular difficulty in examining the shifts. At most, the nonstationarity is responsible for intrinsic broadening of the spectrum, but by an amount that is small relative to the magnitude of the shifts. For the above reasons, the spectra shown in the figures are for relaxing turbulence.

Figure (4.12) shows the frequency spectra of a mode with small wavenumber for relaxing turbulence with the E×B nonlinearity only, the polarization drift nonlinearity only, and both nonlinearities. In the latter case there is a marked shift of the spectrum peak to higher frequency. In all cases, the peaks are in the electron diamagnetic direction. In Fig. (4.13), the spectra for a mode with large wavenumber are displayed. For the cases with E×B and polarization drift nonlinearities only, the spectra are peaked at the small frequency corresponding to the (linear) diamagnetic rotation [third term, Eq. (4.5)]. With the two nonlinearities combined, the spectrum becomes highly complicated, acquiring a small reproducible feature in the ion direction, and a much larger feature at high frequency in the electron direction. The mean frequency $\bar{\omega}_k = \int d\omega\, \omega\, S_k(\omega)$ is dramatically raised, as is the width, $\Delta\omega_k^2 = \int d\omega\, (\omega^2 - \bar{\omega}_k^2)\, S_k(\omega)$, relative to the cases with a single nonlinearity. For the combined case of Fig. (4.13), $\bar{\omega}_k \cong 2\Delta\omega_k$. From these figures, it is clear that there is a



marked frequency shift associated with the coexistence of the two nonlinearities. The magnitude of the shift, in absolute terms, is larger for the mode with larger wavenumber. For the shifted spectra of Figs. (4.12) and (4.13), $\bar{\omega}_{k\ large}/\bar{\omega}_{k\ small} \approx 25$, a number of the same magnitude as the ratio of the wavenumbers. Both of these features are consistent with the shift described by Eq. (4.13). The same general features are also found in the spectra of steady state (driven/damped) turbulence computed numerically for the same model. However, due to the very broad linewidths of the steady state case, these features are less pronounced.

Given the complexity of the frequency spectrum, especially for high k, a more quantitative comparison with theory is beyond the scope of the present work. The next chapter will focus on a theoretical understanding of the complex features and inclusion of the feedback of spectrum shifts on the mode stability in the simulations. The existence of a large frequency shift at the longer wavelengths required for the E×B nonlinearity is interesting in light of the apparent observation of long wavelength fluctuations with large phase velocity ($v_{ph} > \omega_*/k$) in the Texas Experimental Tokamak (TEXT)[16].

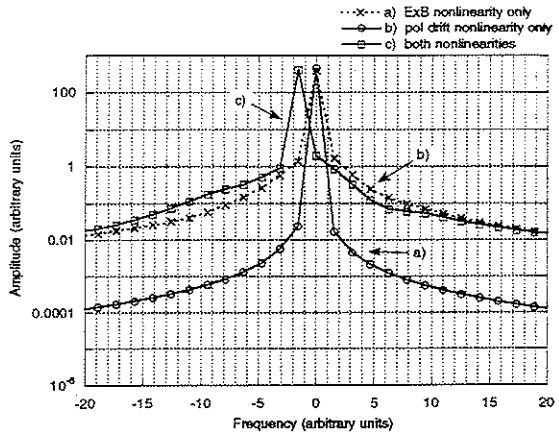


Fig. 4.12 Frequency spectra of a mode with small wavenumber during relaxation without driving or damping. The spectra correspond to a) ExB nonlinearity only, b) polarization drift nonlinearity only, c) both nonlinearities

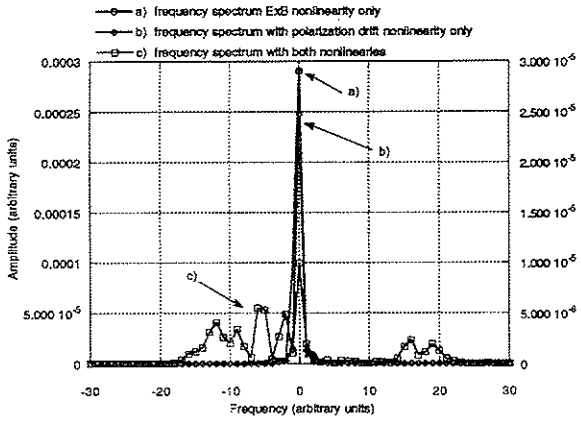


Fig. 4.13 Frequency spectra of a mode with large wavenumber during relaxation without driving or damping. The spectra correspond to a) ExB nonlinearity only, b) polarization drift nonlinearity only, c) both nonlinearities



## 4.5 Conclusions

The spectral transfer dynamics of drift wave turbulence over the broad wavenumber range incorporating both the ExB and polarization drift nonlinearities has been explored. This work is based on numerical solution of the single field dissipative trapped ion convective cell turbulence model[1,2] and describes the spectral transfer resulting from unstable trapped ion modes at long wavelength. The present computational model is based on an "iδ" approximation with laminar electron dynamics resulting from collisional, nonadiabatic electrons. Consequently, the present computational model neglects the dynamical feedback of finite amplitude nonlinear-induced frequencies on mode stability.

A variety of diagnostics, including wavenumber and frequency spectra, spectrum histories, energy and enstrophy transfer rates, and spectrum impulses, are employed to characterize the transfer. It is found that the ExB and polarization drift nonlinearities transfer energy (and enstrophy) much as they would in isolation. Consequently, there are two subranges corresponding to spectral regions in which one nonlinearity or the other is dominant. In the low k ExB subrange, the net energy transfer is directed to higher k with a very large nonlocal component in the $k_y$ direction. There is significant generation of enstrophy on the timescale of a few eddy turnover times. In the high k polarization drift subrange, transfer is local in k. There is evidence for a dual cascade, and enstrophy generation is weak, requiring many eddy turnover times for any change in magnitude. The near conservation of enstrophy on dynamical timescales is consistent with the observed inverse cascade of energy. For saturated turbulence which is driven at low k, damped at lower k and at high k, with an inertial range providing a bridge between the driven wavenumbers and the short wavelength damping, the spectrum is flat in the ExB subrange and falls off in the polarization drift subrange. Nonlocal transfer is responsible for the flatness of the spectrum at low k.

In the crossover region where the two nonlinearities are comparable there is a rich cross-coupling. Energy is transferred nonlocally but the transfer tends to be isotropic. A significant shift in the peak of the frequency spectrum occurs. This shift is directly attributable to the coupling of the two nonlinearities, as it disappears if either nonlinearity is set to zero. The shift increases with increasing wavenumber. Both of these effects are consistent with the theoretical shift based on the cross-coupling of the two nonlinearities in the renormalized response function. In the present work, model limitations have prevented treatment of the back-reaction of the shift on the mode dynamics.

The visualization of energy flow by following the spectral relaxation after application of a localized impulse energy at a given wavenumber is found to provide a reliable and graphic indication of energy flow, indicating accurately its direction, its locality in wavenumber space and whether transfer is isotropic.

In the next chapter, work on this model will include the frequency shift feedback and its modification of the mode stability structure. In addition, an investigation will be made of structures which are stable solutions of one nonlinearity and their persistence in the presence of both nonlinearities.



**References**


1. P.H. Diamond and H. Biglari, Phys. Rev. Lett. **65**, 2865 (1990).
2. D.E. Newman, P.W. Terry, and P.H. Diamond, Phys. Fluids B **4**, 599 (1992).
3. R.J. Fonck, S.F. Paul, D.R. Roberts, Y.J. Kim, N. Bretz, D. Johnson, R. Nazikian, and G. Taylor, *18th European Conference of Controlled Fusion and Plasma Physics*, Berlin, Germany, edited by P. Bachmann and D.C. Robinson (European Physical Society, Geneva, Vol. 15C, Part I, 1991) p. I-269.
4. P. Cripwell and A.E. Costley, *18th European Conference of Controlled Fusion and Plasma Physics*, Berlin, Germany, edited by P. Bachmann and D.C. Robinson (European Physical Society, Geneva, Vol. 15C, Part I, 1991) p. I-17.
5. P.M. Schoch, J.C. Forster, W.C. Jennings, and R.L. Hickok, Rev. Sci. Instrum. **59**, 1825 (1986).
6. D.L. Brower, W.A. Peebles, and N.C. Luhmann, Nucl. Fusion **27**, 2055 (1987).
7. P.W. Terry and W. Horton, Phys. Fluids **25**, 491 (1982).
8. A. Hasegawa and K. Mima, Phys. Rev. Lett. **39**, 205 (1977).
9. R.H. Kraichnan, J. Fluid Mech. **47**, part 3, 525 (1971).
10. Y.-M. Liang, P.H. Diamond, X.H. Wang, D.E. Newman, and P.W. Terry, "A Two-Nonlinearity Model of Dissipative Drift Wave Turbulence", Phys. Fluids B, submitted.
11. P.W. Terry and P.H. Diamond, in Statistical Physics and Chaos in Fusion Plasmas, ed. by W. Horton and L. Reichl (Wiley, N.Y., 1984) p. 335; and P.W. Terry and P.H. Diamond, Phys. Fluids **28**, 1419 (1985).
12. P.W. Terry and D.E. Newman, Bull. Amer. Phys. Soc. **37** (1992).
13. T. Boutros-Ghali and T.H. Dupree, Phys. Fluids **24**, 1839 (1981).
14. The line width is not directly given by the coherent damping. In steady state coherent damping is balanced by driving and incoherent emission and all of these processes affect the spectrum. This is discussed in the articles of Ref. 11.
15. This is further discussed in N. Mattor and P.W. Terry, Phys. Fluids B **4**, 1126 (1992).
16. D.W. Ross, R.V. Bravenec, Ch. P. Ritz, M. L. Sloan, J.R. Thompson, A.J. Wootton, P.M. Schoch, J.W. Heard, T.P. Crowley, R.L. Hickok, V. Simcic, D.L. Brower, W.A. Peebles, and N.C. Luhmann, Jr., Phys. Fluids B **3**, 2251 (1991).

# Chapter 5
# Multiple Field effects

## 5.1 Introduction

One of the more interesting and novel features of models with two nonlinearities is the cross-coupling dynamics of the two nonlinearities[1]. In the previous chapter, cross-coupling dynamics was represented analytically as a frequency shift because the propagator was taken to be real. In fact the cross-coupling dynamics has two parts: the nonlinear frequency shift from the cross terms of equation (4.10) multiplied by the real part of the inverse propagator $\Delta\omega_k^{-1}$; and a dynamical effect from the cross terms multiplied by the imaginary part of the inverse propagator. The dynamical effect changes the transfer dynamics and therefore changes the saturated (or instantaneous) spectrum. This can then change the spectrum weighted average k. All of which can, in principle, affect quantities such as transport. The frequency shift has a more subtle consequence. Through the propagator it also modifies the transfer dynamics, moreover, there can be an effect on the mode stability (linear stability). This change of mode stability is an outgrowth of the inertia of the adiabatic electrons which give rise to a frequency change in the form of a $(\omega - \omega^*)$ term in the dispersion relation. The model used previously, which was a one-field model, has the linear growth spectrum ($\gamma_k$) fixed. This type of model does not permit the frequency shifts to impact the linear stability. In order to investigate the ramifications of such a feedback, a two-field model for dissipative trapped electron modes is used[2]. In this model, two fields are evolved, the potential ($\phi$) and election density ($n_e$), with all of the quadratic



nonlinearities intrinsic to this problem. This model permits the investigation of the dynamics in both the strongly dissipative regime, where the one-field model is valid, and the collisionless regime. The two main benefits of using a two-field model are the ability to investigate the interplay between the frequency shifts and the linear growth spectrum, and a thorough exploration of the transport properties of this type of mode. In the one-field model the phase relation between n and $\phi$ is set and therefore, the transport ($\backslash\mathbf{B}\backslash bc\backslash< ( v n ) = < \nabla\phi \times z\ n >$) is an artificially constrained quantity. In the two-field model, it is possible to examine the evolution of the flux under the nonlinear processes that govern the correlation of n with v and determine how the individual modes or regions of modes contribute to the net flux. For example, do some regions in k space contribute an inward flux while others give an outward flux?

As before, a model with distinct subranges will be investigated ; the long wavelength extreme of the spectrum is dominated by the E×B nonlinearity while at the high k, shorter wavelength extreme is dominated by the polarization drift nonlinearity. The previous computational and analytic results from the one-field model confirm the existence of the two extremes, but also point out the importance of the intermediate regime. In this region of k space, neither the E×B nor the polarization drift nonlinearity dominate completely and the interplay (cross-coupling) becomes very important. The earlier work with the one-field model showed that in the long wavelength subrange there exists a direct, non-local, anisotropic cascade of energy to high $k_y$. In the short wavelength (polarization drift dominated) subrange, a dual cascade analogous to 2-Dimensional Navier-Stokes was found with an inverse cascade of energy to low k and a direct cascade of enstrophy (mean squared vorticity) to high k. The cascades due to the polarization drift nonlinearity were largely local

and isotropic in nature, in agreement with the well known Navier-Stokes dynamics. In the crossover region the dynamics are mixed but not a superposition of the two extremes: the net dynamics tends to be a more isotropic, yet a non-local direct cascade of energy to high k remains in force. In addition to the change in transfer dynamics, a large frequency shift, resulting from the imaginary part of the cross-coupling, is found.

It is important to keep in mind that many or most physical systems contain multiple nonlinearities and while it is true that one nonlinearity may dominate the dynamics in a given region of k space, at some point a region may exist in which multiple nonlinearities of comparable importance co-exist . In some systems, DTEM for example, the dynamics at the extremes, where just one nonlinearity dominates, conspire to increase the importance of the crossover region. In the case of DTEM, the E×B nonlinearity tends to transfer energy into the crossover region from low k while the polarization drift nonlinear inverse cascades energy into the crossover from high k. These combined cascades increase the total energy in the crossover region and act to make the crossover dynamics very important for the total system dynamics.

In this work many of the techniques used in the earlier studies are applied to determine if the same dynamics hold in the more complete system. The spectral flow diagnostic is used to describe the transfer dynamics of the two nonlinearities, i.e., whether it is, local or non-local, direct or inverse, and isotropic or anisotropic. Additionally, the frequency and k spectra are analyzed temporally, to investigate secular and quasi-periodic variations. As before, the computational work is done using a spectral code containing up to 41 by 41 modes. Two general types of simulations are performed: 1) scanning of collisionality while holding other factors roughly constant; and 2) scanning of the ratio of



relative strength of nonlinearities while holding the collisionality ( $\nu$ ) fixed. In both cases some adjustment of damping is made in order to investigate the effect of varying saturation levels. These two classes of simulations permit the investigation of a variety of phenomena. In the first case one effectively scans the growth rate and the importance of having two-fields. In the high $\nu$ region this permits verification of the results of the one-field model while still including a much more realistic growth spectrum. In the second set of simulations it is possible to study the effect of the cross-coupling terms on growth rates and spectrum shapes in a self consistent way. Here the focus is on the average and instantaneous rates of energy extraction, spectrum shape and transfer dynamics.

The main results in this chapter are now summarized. It is found that while the results from the one-field model can be reproduced in the high $\nu$ regime of the two-field model, there are additional new phenomena intrinsic to multi-field models. Despite the fact that in the two-field model the nonlinearities are in different fields (E×B in the density field and polarization drift nonlinearity in the potential field) the model behaves very much like the one-field model when $\nu$ is large. The nonlinearities cascade in the same fashion with the E×B nonlinearity transferring energy nonlocally to high $k_y$ in the region it dominates. Once again it is in the cross over region that the dynamics are different. The total (net) transfer in the cross over region is dominantly a direct cascade to high k. However it is more isotropic than when the E×B nonlinearity acts alone and it is more non-local than when the polarization drift nonlinearity acts alone. As before, in this subrange, there is marked evidence of a large change in the frequency dynamics. In addition to this expected dynamics, a variety of new phenomena are found. The most significant of these is the nonstationary nature of the saturated quantities. In general, in turbulent systems it is

assumed that once saturation is reached quantities such as mode amplitude, frequency spectrum, transport and k spectrum are constant when averaged over a few eddy turnover times. This, in some sense, is the definition of a saturated steady state. In the present system something very different is observed - large-scale global fluctuations. These fluctuations in total energy, enstrophy, and flux are characterized by both their large (>30%) amplitude and their long period ($> 5\tau_e$). During the global fluctuations most of the local (local in k space) quantities undergo marked change. The k spectrum fluctuates, the frequency spectrum fluctuates, the transfer rates fluctuate, and the transport fluctuates. Distinct phases of these fluctuation cycles are identifiable. These phases can be broadly defined as the preliminary phase (at the bottom of a cycle), growth phase on the up cycle of the fluctuation, turnover phase (at the peak of a cycle), and the crash phrase (on the declining side of a cycle). The k space regime in which these fluctuations are most visible is the crossover region where both nonlinearities are comparable. During the cycle the frequency spectrum for a mode in the center of the crossover range undergoes a metamorphosis from no shift to upshift to downshift while the energy in those modes grows, then transfers out, and finally crashes. It must be kept in mind that if time histories are averaged over many eddy turnover times (>20), these fluctuation can be lost or overlooked.

The organization of this chapter is as follows: The model and its properties are presented in section II along with the reduction to the one-field model. Section III presents a brief heuristic explanation for the cycling fluctuations. Section IV has the results of the two-field model. Finally, section V presents conclusions.



## 5.2 Basic Model and Properties

In this section the basic model is briefly presented and some of its main properties discussed. This is done in the context of comparing and contrasting the present model with the one-field model of the same system. This analysis examines the linear structure, integral invariants, and the fourier transformed mode equations.

The model utilized for this study is a set of trapped particle fluid equations. This model couples the dynamics of collisional trapped electrons with hydrodynamic ions through the quasineutrality condition. The derivation of this equation and a discussion of its details and limitations have been presented elsewhere. In previous work, the inertial response of the electrons was neglected by considering only the most dissipative extreme of collisionality. Here, the electron nonlinearity and inertia terms are included, yielding the model equations:

$$(1 - \sqrt{\varepsilon} - \rho_s^2\nabla^2)\frac{\partial\phi}{\partial t} + V_D\frac{\partial\phi}{\partial y} - \sqrt{\varepsilon}\,V_D\,(1+\alpha\eta_e)\frac{\partial\tilde{n}_e}{\partial y} - \sqrt{\varepsilon}\nu_e\tilde{n} + \sqrt{\varepsilon}\nu_e\phi$$

$$+ \rho_s C_s\,\nabla\phi\times\mathbf{z}\cdot\nabla\,\rho_s^2\nabla^2\phi + \mu\nabla^4\phi = 0\,, \qquad (5.1)$$

and

$$\frac{\partial\tilde{n}_e}{\partial t} + V_D\,(1+\alpha\eta_e)\frac{\partial\tilde{n}_e}{\partial y} + \nu_{eff}\,\tilde{n}_e - C_s\,\rho_s^3\nabla\phi\times\mathbf{z}\cdot\nabla\,\tilde{n}_e - \nu_{eff}\,\phi = 0\,, \qquad (5.2)$$

where $\tilde{n}_e = n_e + \phi$ is the fluctuating electron density plus the fluctuating potential, $V_D = (cT_e/eB)L_n^{-1}$ is the diamagnetic drift velocity, $D = \varepsilon^{1/2}V_D^2(1+\alpha\eta_e)/\nu_{eff,e}$ is a negative

diffusivity describing the destabilization of DTEM modes by electron collisions, $\varepsilon$ is the trapped electron fraction, $\eta_e = d\ln T/d\ln n$ is the electron temperature gradient parameter, $\nu_i$ models long wavelength collisional damping, $\nu_{eff,e} = \nu_e/\varepsilon$, $\mu$ is the coefficient of the hyper-viscosity introduced to model strong damping at high k, $L_n$ is the density gradient scale length, $\rho_s = (cT_e/eB)/C_s$ is the ion gyroradius evaluated at the electron temperature, and $C_s = (T_e/m_i)^{1/2}$ is the ion sound speed. The substitution of $\bar{n}$ for $n_e + \phi$ is for computational simplicity. The primitive equation for $n_e$ has a time derivative of $\phi$ in addition to the time derivative of $n_e$, making the use of explicit solvers impossible. This change in no way alters the dynamics of the problem and was made at this stage for consistency with the computational work. In contrast with the previous work (chapter 4), the linear polarization drift term responsible for linear dispersion has been included in Eq. (5.1). The basic mechanisms governing transfer and the concepts describing this process are however unaltered. Indeed, numerical solutions with and without the linear polarization term included were found to be qualitatively the same for spectral ranges studied herein.

The nonlinearity appearing in Eq. (5.2) is the $E \times B$ nonlinearity, arising from $\mathbf{v}_E \cdot \nabla \bar{n}_i$, where $\mathbf{v}_E = -(c/B_o)\nabla\phi \times \mathbf{z}$ is the $E \times B$ drift. The nonlinearity which appears in Eq. (5.1) is the polarization drift nonlinearity, and arises from $n_o \nabla \cdot \mathbf{v}_p^{(1)}$, where $\mathbf{v}_p^{(1)} = B_o^{-1}(m_i c/e)\mathbf{z} \times \mathbf{v}_E \cdot \nabla \mathbf{v}_E$ is the nonlinear polarization drift. The $E \times B$ nonlinearity requires a nonadiabatic electron response (provided by the trapped electrons), whereas the polarization drift nonlinearity derives from the ion polarization drift.

From Eqs. (5.1) and (5.2), it is again apparent that, by virtue of its additional spatial derivative, the polarization drift nonlinearity dominates the E×B nonlinearity at very short wavelengths. The converse holds at long wavelengths. The nominal crossover point is given by the wavenumber at which the two nonlinearities are equal. This crossover point is however also dependent on the effective electron collisionality ($\nu_{eff,e}$). Assuming rough



isotropy, so that $\nabla_\perp \approx \partial/\partial y$, this wavenumber is given by $k\rho_s \approx \delta = C_s/L_n\nu_{eff,e} \equiv k_o\rho_s$. Because the nonlinearities are characterized, not by a single spatial scale, but by a triad interaction consisting of three waves of differing wavelengths, it is more realistic to identify a region, centered about the crossover wavenumber, in which the two nonlinearities are comparable, rather than to speak of a single wavenumber at which the two are equal. It is within this region that the cross-coupling dynamics are dominant.

Due to the presence of the E×B nonlinearity, a single quadratic invariant, the energy, is admitted by this system in the absence of driving and damping. As previously discussed, the E×B nonlinearity in isolation transfers energy to short wavelengths, given a spectrum which is peaked at long wavelength or flat, and therefore drives robust production of enstrophy[2]. This term is present even in spectral ranges where the polarization drift nonlinearity dominates ($k > k_o$). Consequently, enstrophy is not conserved even when the E×B nonlinearity is weak compared to the polarization drift nonlinearity. However, in such a case, the E×B nonlinearity accounts for proportionately less of the total energy transfer. Because enstrophy production is tied to energy transfer by the E×B nonlinearity, it can be expected that the importance of enstrophy production in the cascade dynamics diminishes for $k > k_o$. For $k < k_o$, the eddy turnover rate is controlled by the dominant E×B nonlinearity and can be expected to be comparable to the enstrophy generation rate. For $k > k_o$ the eddy turnover rate is controlled by the larger polarization drift nonlinearity, while the enstrophy generation rate is tied to the weaker E×B nonlinearity. Consequently, there will be little enstrophy generation on the nonlinear interaction or nonlinear transfer timescale.

As described in chapter 2, energy transfer that is anisotropic and nonlocal in wavenumber space is a robust feature of the E×B nonlinearity, but at variance with the conventional picture of the cascade process. On the other hand, the polarization drift nonlinearity, investigated in chapter 3, is of the same form as the advective nonlinearity of the vorticity evolution equation of Navier-Stokes turbulence. Therefore, as expected, the

polarization drift nonlinearity produces transfer which is local and isotropic. It is possible to associate these features with symmetries in the structure of the nonlinear coupling. These symmetries are most transparent in the Fourier representation of the nonlinearities when simplified to the one-field model (see chapter 4). The polarization drift nonlinearity in Eq. (5.6) has the same symmetry structure as in the one-field case. The transfer dynamics of the E×B nonlinearity Eq. (5.5) are not as transparent in this representation. This is due to the cross field nature of the nonlinearity and the transformation made to $\tilde{n}$ from $n_e$ and $\phi$. Transforming Eqs. (5.1) and (5.2), the evolution of mode amplitudes is given by

$$\frac{\partial \phi_k}{\partial t}(1 - \sqrt{\varepsilon + k_\perp^2 \rho_s^2}) - i\sqrt{\varepsilon}\, V_D\, (1+\alpha\eta_e)\, k_y \phi_k + iV_D\, k_y \phi_k + \nu_{eff}\,(\phi_k - \tilde{n}_k) + \mu k^4 \phi_k + N_k^{(Pol)} = 0, \tag{5.3}$$

$$\frac{\partial \tilde{n}_k}{\partial t} + iV_D\,(1+\alpha\eta_e)\, k_y \phi_k + \nu_{eff,i}\,\tilde{n}_k - \nu_{eff,i}\,\phi_k + N_k^{(E\times B)} = 0, \tag{5.4}$$

where

$$N_k^{(E\times B)} = \frac{1}{2}\rho_s C_s \sum_{k'} \mathbf{k}\times\mathbf{k}'\cdot\mathbf{z}\ \phi_{k'}\,\tilde{n}_{k-k'} \equiv \sum_{k'} \chi^{(E\times B)}_{k,k'}\,\phi_{k'}\,\tilde{n}_{k-k'}, \tag{5.5}$$

$$N_k^{(Pol)} = \frac{1}{2}\rho_s^3 C_s \sum_{k'} \mathbf{k}\times\mathbf{k}'\cdot\mathbf{z}\,[(k_\perp - k_\perp')^2 - k_\perp'^2]\,\phi_{k'}\,\phi_{k-k'}$$

$$\equiv \sum_{k'} \chi^{(Pol)}_{k,k'}\,\phi_{k'}\,\phi_{k-k'}, \tag{5.6}$$

As discussed in chapter 3 and 4 the symmetries in the nonlinear couplings responsible for this distinction between the E×B and polarization drift nonlinearities derive from the nonadiabatic electron response of the E×B nonlinearity ($\phi \sim \delta\partial\tilde{n}_e/\partial y$) and the



adiabatic electron response of the polarization drift nonlinearity ($\phi \sim \tilde{n}$). These symmetries carry directly over to the energy transfer rates and give rise to a nonlocal transfer rate by the E×B nonlinearity which dramatically exceeds the local transfer rate, while for the polarization drift nonlinearity, local and nonlocal transfer rates tend to be comparable.

The linear dispersion relation for this system is given by:

$$(1 - \sqrt{\varepsilon + \rho^2 k^2})\,\omega^2 + (i\nu - v_D\, ky - i\rho^2 k^2 \nu\ \sqrt{\varepsilon} + i\nu\ (1 + \rho^2 k^2) + \sqrt{\varepsilon}\, v_D\, ky\, \beta)\,\omega - i\, v_D\, ky\, \nu = 0 \tag{5.7}$$

with

$$\beta = (1+\alpha\eta_e)$$

This relation has relatively complicated higher order structure but to first order the solutions are very simple. The real part of the solution which gives the linear frequency is linear for low k and the rolls over at higher k due to the linear polarization drift term (Fig. 5.1). The imaginary part of the solution gives the linear growth rate and to first order, or at low k, the solution is given by $\gamma \propto k_y^2$ (Fig. 5.2). These first order linear solutions agree with the linear dispersion relation for the one-field iδ model exactly. This is not surprising as the reduction from the two-field model to the one-field model is simply a matter of dropping the inertial terms in the electron density equation and then solving the 2 equations in a high $\nu$ limit.

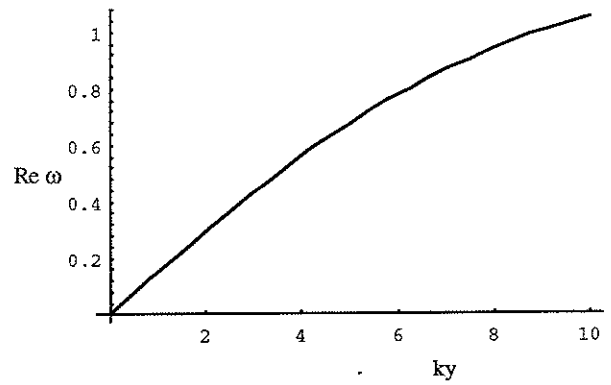


Fig. 5.1 The real part of the dispersion relation.
This gives the frequency spectrum as a function of ky.

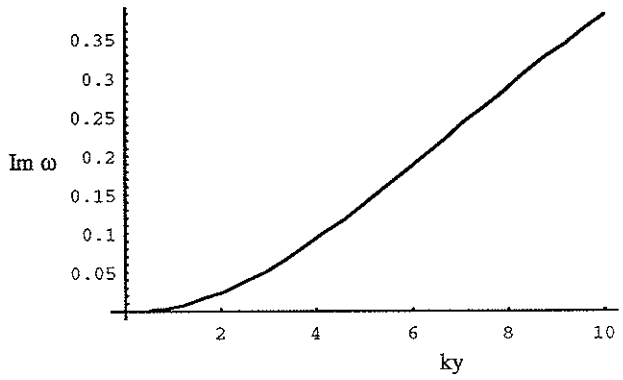


Fig. 5.2 The imaginary part of the dispersion relation.
This gives the growth rate as a function of ky



## 5.3 Heuristics of the Cycling

Relaxation oscillations are relatively ubiquitous in nature, existing in systems as simple as a bow drawn over a violin string to systems as complex as sawteeth in a fusion device. In most of these cases there is a competition between forces driving the system away from its preferred state and restoring forces. In addition to these two competing forces some form of inertia is needed to take the system through and beyond its equilibrium. In the case of the violin string the oscillation is driven by the difference between the static and the dynamic coefficients of friction and the inertia of the string. While the cycles in the two-field model are not as simple as the violin string it is instructive to go through a few scenarios for the mechanism behind the two-field oscillations. First, two simple easily pictured mechanisms will be outlined; then, a third more realistic mechanism combining the first two will be outlined. The first scenario is a simple feedback of the frequency shift on the linear growth rate with overshoot due to spectrum inertia. In this case, the system starts from small amplitude. From closure calculations the frequency shift is found to be amplitude dependent, ($\omega_s \propto \sum_{k'} k_y (2k_y^2 - k'^2_y) |n_{k'}|^2$) , which suggests that at low amplitude there is little or no frequency shift which allows the linear growth to proceed. As the modes gain in amplitude the frequency shift becomes more pronounced. Since the linear growth rate is $\gamma \propto [\omega_* (1 + \alpha \eta_e) - \omega]$ , when $\omega > \omega_* (1 + \alpha \eta_e)$ the growth becomes negative and the mode is stabilized. Therefore at some point in the growth the mode amplitudes become large enough to stabilize the mode and the growth ends. The mode then loses energy until the amplitude is small enough to turn off the frequency shift which in turn reactivates the

growth rate. The spectrum inertia is required to force the mode amplitudes to overshoot the amplitude which would cause a frequency shift giving marginal stability. The system could then undergo stable oscillations around the marginally stable amplitudes.

The second mechanism is similar to the first except that the frequency shift is replaced by a change in the dynamics due to the cross-coupling, i.e., $\chi^{(E\times B)}_{k,k'}\ \chi^{(Pol)}_{k-k',-k'}$ Im $\Delta\omega_k^{-1}$. In this case as the system grows from small amplitude the dynamics of the transfer are dominated by the individual nonlinearities. This is because the cross-coupling dynamic shift is also dependent on the amplitude. As the most unstable modes (assumed to be the middle k modes near the crossover) grow higher than the surrounding modes the cross-coupling dynamics shift starts to dominate over the standard dynamics. This allows energy to be transferred rapidly out of these high amplitude modes to the damped modes at either end of the k spectrum. As the energy is transferred out, the mode energy decreases eventually reaching the level at which the cross-coupling dynamics are less important than the dynamics of the individual nonlinearities. At that point the mode starts linearly (and nonlinearly) growing again. Once again the spectrum (or transfer) inertia is relied on to force an overshoot of the mode amplitudes allowing the cycling to occur.

The final scenario combines the first 2 and appears more realistic for this model. Again, the system starts from low amplitude, as the amplitudes grow the frequency shift grows once again having a feedback effect on the growth rate. The shift in transfer dynamics becomes important at a larger amplitude because it depends on the nonresonant part of the propagator (i.e., Im $\Delta\omega$) and is, as such, dependent on the frequency shift. As the frequency shift becomes large enough to stabilize the mode, the transfer dynamics shift rapidly moves the energy out of the crossover modes and into new modes. This forces the amplitude of the new energy bearing modes to grow rapidly causing them to undergo similar



shifts in transfer dynamics, eventually transferring the energy to dissipative modes. While the energy is being cascaded to dissipative modes the energy in the linearly most unstable modes falls well below the marginal stability amplitude due to the combined effects of the frequency shift induced stabilization and the shifted transfer dynamics. The cycle then restarts. Many of the elements in this mechanism can be inferred from the diagnostics in the computational studies, as will be shown in the next section.

### 5.4 Computational results

In this section the computational results from the two-field model are described. In the two-field computations, as with the one-field work, a number of computational diagnostic tools were used. The nonlinear transfer diagnostic is based on the energy evolution for an individual model k. The equations for the energy evolution are:

$$\frac{(1-\sqrt{e})}{2}\frac{\partial|\phi_k|^2}{\partial t} - \sqrt{e}\,\nu\,|n_k\,\phi_k| + \sqrt{e}\,\nu\,|\phi_k|^2 = T_k^{(Pol)} \qquad 5.8$$

and

$$\frac{1}{2}\frac{\partial|n_k|^2}{\partial t} + \nu\,|n_k|^2 - \nu\,|\phi_k\,n_k| = T_k^{(E\times B)} \qquad 5.9$$

with

$$T_k^{(Pol)} = c_s\,\rho_s\,\mathrm{Re}\sum_{k'}(k\times k'\cdot z)\,(k-k')^2\,n_{k'},\ n_{k-k'},\ n_k^{*} \qquad 5.10$$

and

$$T_k^{(E\times B)} = c_s\,\rho_s\,\mathrm{Re}\sum_{k'}(k\times k'\cdot z)\,\phi_{k'},\ n_{k-k'},\ \phi_k^{*} \qquad 5.11$$

Here $T_k^{(E\times B)}$ is defined as the $E\times B$ nonlinear transfer rate and $T_k^{(Pol)}$ is the polarization drift nonlinear transfer rate. If $T_k$ is negative (positive) that implies nonlinear

transfer out of (into) the mode k. For simplicity of data storage this information is not kept for all modes individually; instead, the flow from (to) bands in the $k_x$ and $k_y$ directions are determined. The flow is defined as local if the interacting modes are within |m| modes of k where m is taken to be between 1/3 and 1/2 of the total k spectrum size. Nonlocal flow is all flow not within the region defined by m. In most cases m is chosen to be large in order to insure that the local/nonlocal differentiation is equivalent to local in the Kolmogorov inertial range sense.

In addition to the mode energy/enstrophy transfer diagnostic, global energies ($|n_k|^2_{tot}$, $|\phi_k|^2_{tot}$, $|n_k|^2_{tot} + |\phi_k|^2_{tot}$) and flux ($<n\nabla\phi>$) are tracked. While the global quantities are useful, and in fact first showed the cycling, the local (local being defined as consisting of a single mode) quantities are more instructive and are therefore followed in detail. Mode energy, flux for a given k and frequency spectra for individual modes are all analyzed and used in conjunction with the transfer rates to investigate the dynamics within each field as well as the cross correlations between fields. Finally, it has been found to be very useful to animate sequences of 2-D k spectra and real space isocontours. These animation sequences often give an idea as to which dynamics dominate in a given temporal region.

With the exception of runs done for the purpose of testing the code, all of the computation was done with driving and damping turned on. Typically the linear growth spectrum was peaked at intermediate k with viscosity (or hyperviscosity) decreasing the growth rate at higher k. This results in a growth rate ($\gamma$) that is negative at lowest and highest wavenumbers and positive at intermediate $k_y$ with $k_x = 0$. When the simulations are started from infinitesimal initial amplitudes, a linear growth regime is observed followed by



the onset of nonlinear interactions which reduces the net growth rate (linear plus nonlinear) of the most linearly unstable modes and increases the net growth rate of the less unstable modes. As is common with turbulent simulations, saturation does not occur if the total growth rate, ( $\gamma_T = \sum_k \gamma_k$) is positive, while saturated "steady" states whose saturation amplitude is sensitively dependent on $\gamma_T$ exist for negative $\gamma_T$.

In the limit of large $\nu$ ($\nu >> \omega$) the model is reducible to the one-field model. Consistent with this, the dynamics of the computational simulations of the two-field model are virtually indistinguishable from the one-field results in the high $\nu$ limit. In the region of k space in which the polarization drift nonlinearity dominates the flow is largely local in nature and exhibits a dual cascade. The spectrum is dominantly peaked at low k so the energy bearing modes are below the driving or maximally unstable k. There is very little or no frequency shift for the modes strongly dominated by the polarization drift nonlinearity; however,it is on the polarization drift dominated side of the crossover region that the frequency shift reaches its maximum. In the region dominated by the E×B nonlinearity the transfer is predominately nonlocal and anisotropic, and energy is directly transferred to high $k_y$ modes. The transfer in the $k_x$ direction is also direct but more local. In this regime there is no appreciable frequency shift and the dynamics of the E×B nonlinearity dominate the system dynamics. The k spectrum in the E×B dominated region is anisotropic with the peak at the most unstable modes. This leads to a spectrum plateau which is broadened in the $k_y$ direction and is much flatter then the spectrum in the polarization drift nonlinearity dominated region.

When $\nu \gtrsim \omega$ the dynamics of the ExB and polarization drift subranges remains nearly identical to the results from the one-field model; however, new phenomena are

observed in the cross-over region. This is because the same basic dynamics are now free to have repercussions in the two-field model that were prohibited in the one-field model. As in the high k (polarization drift nonlinearity region) and the low k (E×B nonlinearity dominated region) regimes, the crossover region displays dynamics similar to the one-field dynamics; however unlike the one-field model those dynamics are not temporally stationary. Typically, when a turbulent system reaches saturation the system is assumed to be stationary on a time scale of a few eddy turnover times. This stationarity is best seen by averaging over a few eddy turnover times after which all the significant quantities such as total energy, mode energy, enstrophy, frequency spectrum, transfer rates, etc., are constant. In the two-field model this stationarity simply does not exist in the regimes for which both nonlinearities are important. Instead, a long period fluctuation (cycling) occurs around the saturation level. The dynamics of these fluctuations will be discussed next. But first it should be emphasized that in the proper regime (i.e. high $\nu_{eff}$) the iδ model (fixed phase, one-field model) does a very good job with the nonlinear dynamics of the system. The specific details that an iδ model can not handle by its very construction are the feedback effects due to cross-coupling. These effects modify the linear growth rate as well as the flux. The flux is particularly poorly represented by the iδ model as by construction, iδ models fix the cross correlation or phase relation between n and $\phi$ by δ. This fixes the flux amplitude to the saturation amplitude without allowing any variation due to phase correlation or decorrelation of the various modes. This correlation is found to vary during cycling events and is not even constant in the various individual k modes. The effect on the linear growth rate is more subtle. Since this effect comes from feedback of the frequency shifts onto the growth, it is difficult to decouple this effect from changes in the transfer dynamics. It is however apparent that there is still a large frequency shift and that shift is nonstationary



over a cycling event. It appears that the frequency shifts do cause the most unstable modes to become less unstable, closer to marginally stable, however without more diagnostic evaluation it is not possible confirm this. This work will be reported at a later date.

In the more moderate region of $\nu_{eff}$ ($\nu \sim \omega > \omega_*$) there is a large nonequipartition of energy between $n_e$ and $\phi$ with $\phi >> n_e$. This causes the E×B nonlinearity to be relatively weakened compared to the polarization drift nonlinearity and allows the E×B nonlinearity to undergo a shift in dynamics to a more local more isotropic flow. This also tends to weaken the fluctuations. Therefore, it is the region in parameter space between $\nu >> \omega$ and $\nu \sim \omega$ that is investigated in order to characterize the dynamics of the cycles. This is the region of maximum cycling. The cycling is observed for a wide range of parameters and is characterized by large amplitude fluctuations (> 15% around the mean level), see (Fig. 5.3), with a long period ($\tau_{cycle} > 8\tau_e$). At either extreme in the $\nu_{eff}$ spectrum ($\nu >> \omega$, or $\nu < \omega$) the amplitude of the fluctuations is decreased.

While the cycling is a robust feature of the two-field model, the structure and characterization of the cycles depends on a number of factors. For an example, the cycling amplitude can be controlled by varying the relative strength of the nonlinear transfer to the linear growth rate. This change in amplitude then modifies the characteristic behavior of the cycles. When the amplitude of the fluctuations is decreased, the period of the cycles (Fig. 5.4) is increased. During a cycling event there is typically a burst of flux. This burst of flux (Fig. 5.5), ($\sum_k \nu_k n_k^*$) usually leads the characteristic energy fluctuation in the dominant modes. These early dominant modes are usually the middle k modes which are near, or slightly above the crossover, and are just below the most linearly unstable modes. At the end of a cycle, in the low phase, the phase relation between n and $\phi$ of the low k

modes goes to 0. This causes the E×B nonlinearity (Fig. 5.6) to cease transfer and the flux from the modes to go to 0.



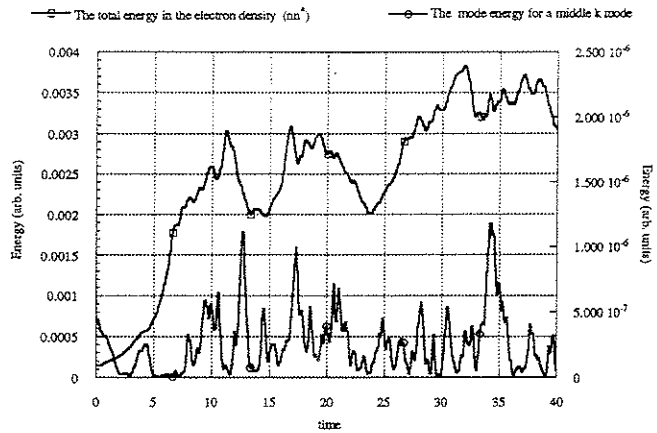


Fig. 5.3 During a cycling event the total energy is seen to fluctuate by more than 15%. The period is 5 to 10 "eddy turnover times" as defined by a mode fluctuation time.

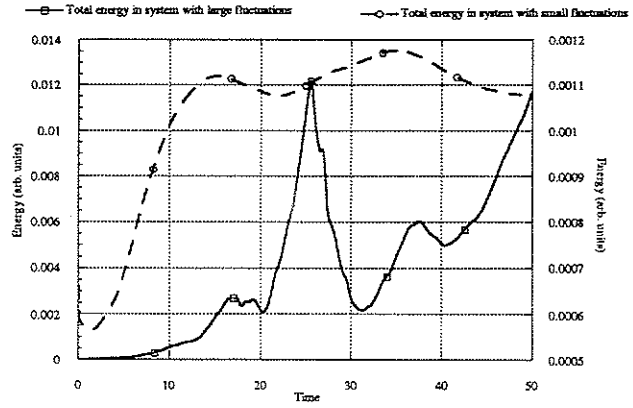


Fig. 5.4 The system with the large amplitude cycles has a longer more regular period. Note that the scales are different for the 2 systems.

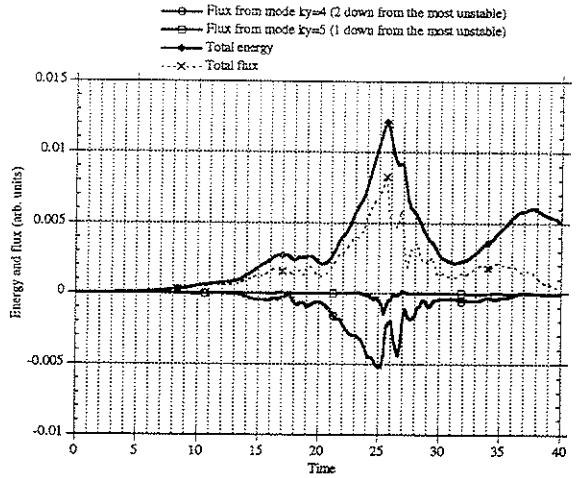


Fig. 5.5 The flux from the 2 most active modes "bursts" coincident with the fluctuation. The burst in the most active mode leads the fluctuation by a small phase.

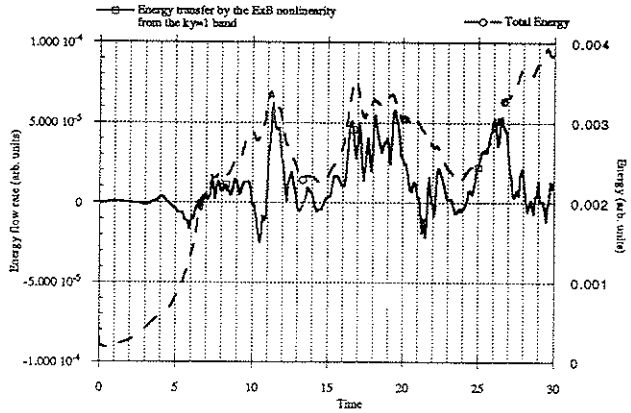


Fig. 5.6 Total energy evolution with the ExB flow evolution during fluctuations. The ExB transfer goes to zero at the end of a cycle then starts again. In fig 5.5 the same behaviour in shown for the flux from a mode



These occur both $< v_k n_k^* >$ (flux) and $\nabla n \times z \cdot \nabla \phi$ (ExB nonlinearity) are maximum when the phase shift between n and $\phi$ is $\pi/2$, and zero when n and $\phi$ are in phase. The phase is then rescrambled on a nonlinear time scale, i.e., approximately one eddy turnover time. The flux and nonlinear transfer then resume.

In the course of a cycle the modes that undergo the most significant changes are the middle ky modes early to midway through a cycle and then low in $k_x$ propagating to high $k_x$ at the end of a cycle. During such an event the frequency spectrum for the middle to low middle $k_y$ modes undergoes large changes. It is difficult to resolve frequency spectrum fluctuations on the time scale of a cycling fluctuation because a cycle period is of order 10 eddy turnover times. This means that each phase of the cycle lasts ~ 3 $\tau_e$ and even then may not be stationary. Nevertheless it is possible to extract some useful spectra assuming the existence of two, three, and four distinct phases in the cycle, and using conditional sampling techniques to extract spectra. These simply assume that one cycle is like another in terms of the relationship between the frequency spectra and the phase of a cycle. Combining multiple partial cycles allows one to create a record of 10s of eddy turnover times for frequency analysis. These low middle ky modes show (Fig. 5.7) a frequency downshift early in the cycle followed by a region of little or no shift, followed by a large upshift. These shifts would be consistent with the suggestion that the linear growth is reduced at the top of the cycle and is then turned back on at the bottom. It is worth noting here that by averaging over a few cycles (~30$\tau_e$) one would miss the nonstationarity of the frequency spectrum and would simply see a broadened frequency spectrum. Unfortunately the explanation of the cycling is probably not as simple as the mechanism which simply has the frequency shift

feeding back on the growth rate . This can be seen from the transfer diagnostic and the k spectrum evolution. The spectrum evolves as follows during a cycle see (Figs. 5.8-5.10):

1) The middle ky (most unstable) modes increase in amplitude,

2) There is a large burst of inverse $k_y$ cascade with a smaller amount of direct cascade in the ky direction, both due to the polarization drift nonlinearity. This causes the spectrum to flatten in the ky direction,

3) A burst of direct cascade in the kx direction due to the polarization drift nonlinearity is nearly coincident with the cross-field transfer from the E×B nonlinearity. This takes most of the energy to the high k dissipation region and brings the cycle full circle.

In real space these cycles can also be characterized by distinct phases. Structures which are elongated in the x direction appear, grow, and propagate in the y direction they then rapidly break apart and start up again (Figs. 5.11-5.13). The eccentricity of these structures (their x/y dimension) is partially determined by where the crossover from E×B dominated to polarization drift dominated is located relative to the most unstable mode. If the most unstable modes are in the E×B nonlinearity dominated region then the structures are narrower in the y direction due to increased direct, nonlocal transfer to high ky.



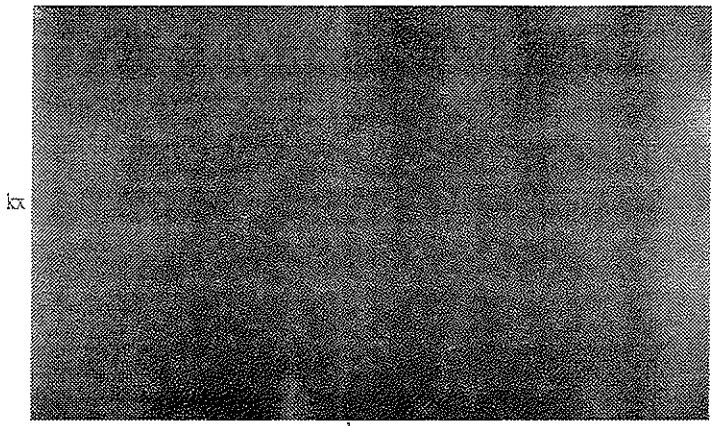


Fig. 5.8 This k spectrum shows the energy dominantly in the middle ky modes. This spectrum is at the begining of a cycle. Darkest color is highest value. The 0,0 mode is at the bottom center.

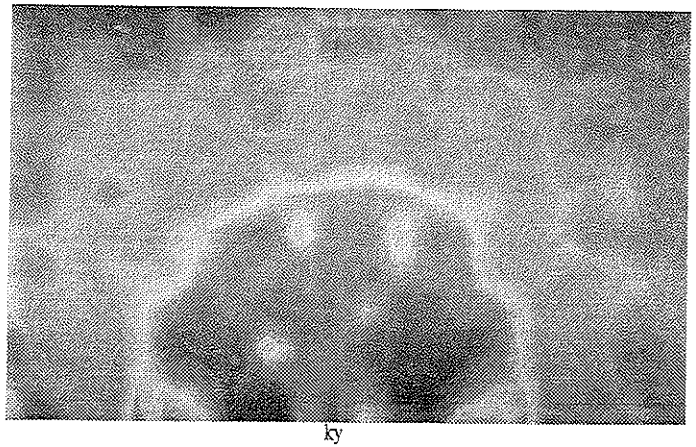


Fig. 5.9 This k spectrum is in the middle of a cycle with energy moving from the early dominant modes to lower ky modes. Some energy can be seen moving to higher ky but this magnitude is much less. The scales and coordinates are the same as for fig. 5.8 with the colors cycling back to dark for the lowest amplitudes.



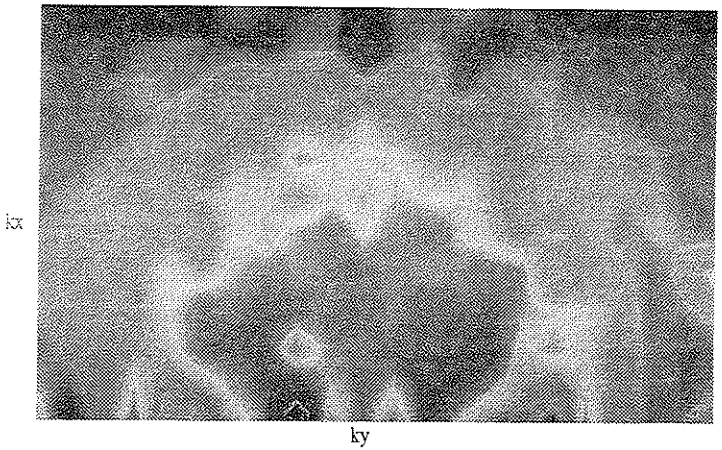


Fig. 5.10 This k spectrum is at the end of a cycle, showing the energy being transfered out to high kx with some spreading in ky also. The scales are the same as for fig 5.9.

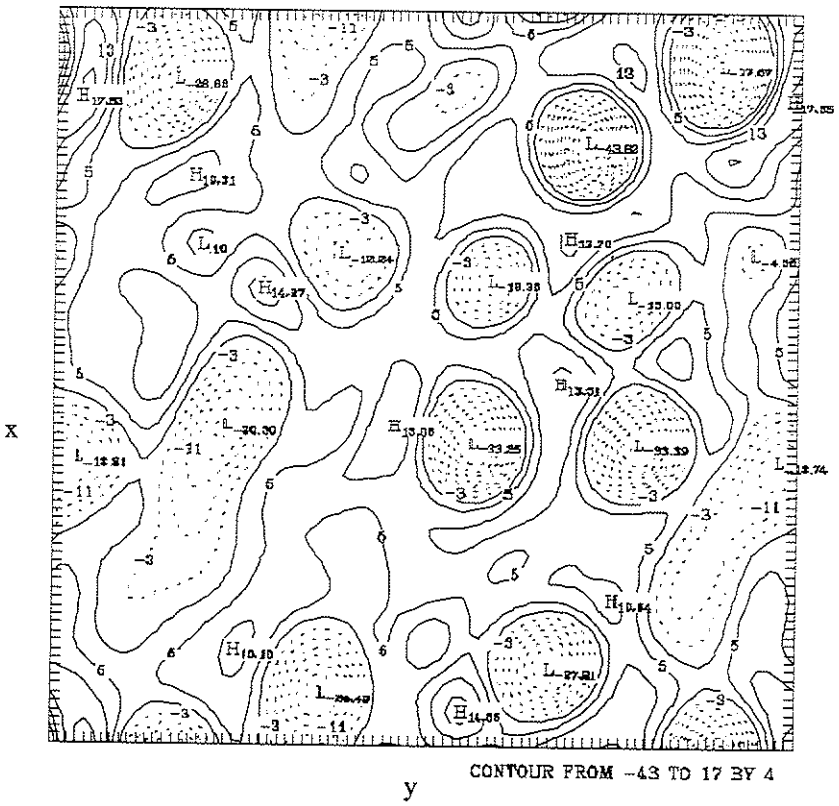


Fig. 5.11 Real space isodensity contours early in a cycle. The structures are relatively isotropic and homogeneous.



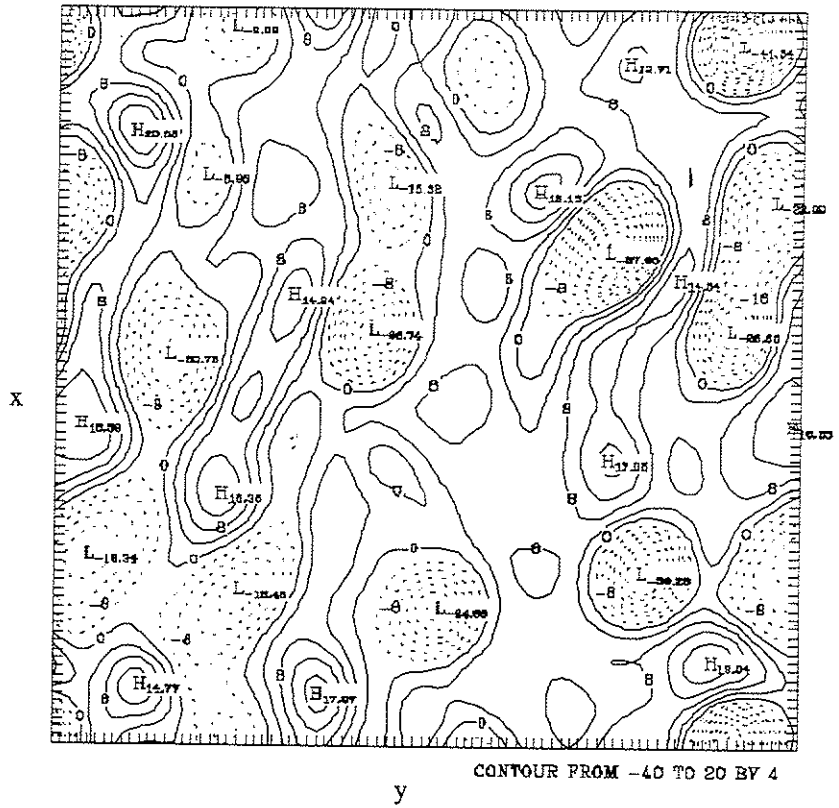


Fig. 5.12 Mid way through a cycle there is noticable elongation in the x direction. The Structures propagate in the y direction.

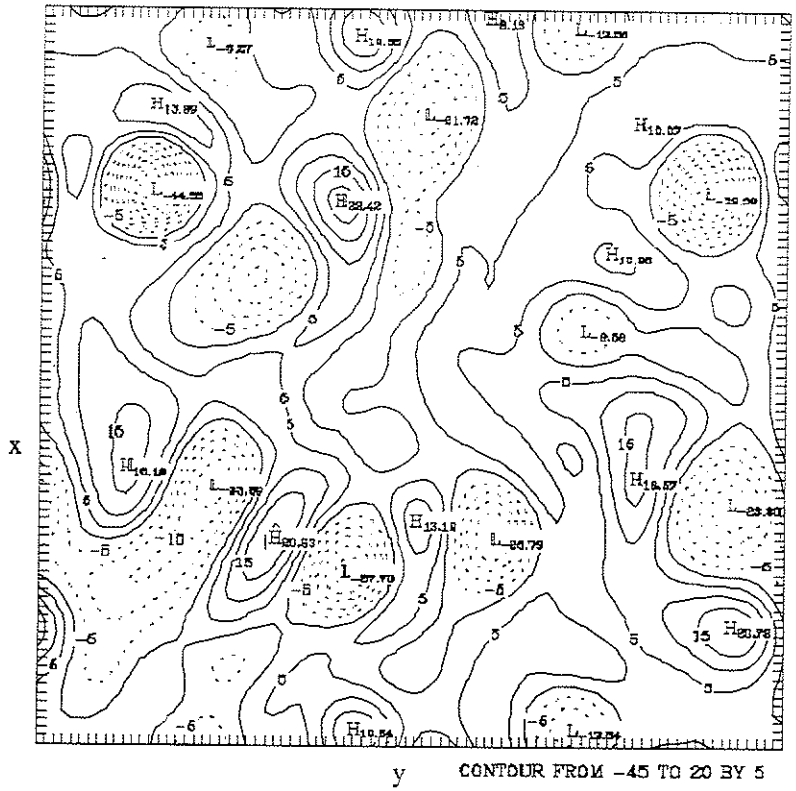


Fig. 5.13 Isocontours at the end of a cycle. The elongated structures in fig. 5.12 are breaking up and the system is becoming more isotropic.



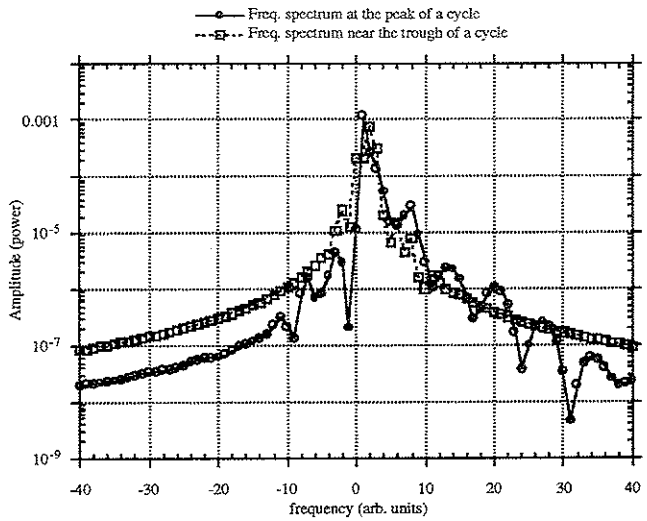


Fig. 5.7 Note the mean frequency up-shift in the peak phase of a cycle. This would tend to decrease the growth rate relative to the trough phase.

## 5.5 Conclusions

Using the same variety of diagnostics as employed for the one-field model, including transfer rates, wavenumber and frequency spectra, mode, spectrum and flux histories, the dynamics of a two-field model has been investigated. It is found that the E×B and polarization drift nonlinearities behave as they would alone in the region of k space in which they dominate. Both the nonlocal anisotropic direct cascade of the E×B nonlinearity and the local, isotropic, dual cascade of the polarization drift nonlinearity are observed. The two-field model dynamics reduces to the one-field dynamics in the high $\nu_{eff}$ limit as expected. Most significantly, large scale fluctuations (cycling) are found to spontaneously occur in cases which include the cross over region in the system. This cycling is observed on a ~$10\tau_e$ time scale with oscillation levels of 15-80% of the base saturation level. The existence of these fluctuations calls into question the validity of the assumption that saturated turbulence is also stationary or steady state. Fluctuations are observed in both the frequency and k spectrum coincident with the global fluctuations. These observations combined with the fact that bursting of transfer and flux are also coincident with the cycles suggests a relaxation type oscillation which is not inconsistent with a heuristic model that was outlined. Characteristics observable in real space include structures elongated in the x (radial) direction. These structures propagate poloidally and break up, then reform on a cycle time scale. It is important to note that if an observer is sampling across multiples cycle the cycles may be washed out giving rise to an apparently broadened spectrum, but missing much of the important dynamics.



## References

1. Y.-M. Liang, P.H. Diamond, X.H. Wang, D.E. Newman, and P.W. Terry, "A Two-Nonlinearity Model of Dissipative Drift Wave Turbulence", Phys. Fluids B, submitted.

2 G.G. Craddock et al, Transport Task Force Meeting Newport RI 1993

# Chapter 6
# Conclusions

Throughout the study of turbulence there are a number of generally accepted principles upon which much of turbulence theory has been built. These basic precepts include the idea of a classic inertial range, isotropy of systems without anisotropic driving, conservative cascades, stationarity of saturated systems and a rough additive principle of nonlinearities. While these ideas largely work very well in the systems for which they were developed, the question of their broader use arises.

Using a series of simple models with extensive diagnostics designed for investigating the nonlinear dynamics, those basic assumptions are investigated in more complex systems then the "simple" Navier-Stokes system in which they were originaly developed. The models used have been, in order of increasing complexity,as follows. A one-field model for Dissipative Trapped Ion Convective Cell (DTICC) turbulence which has one nonlinearity (E×B) and is used to study the dynamics of an anisotropic nonlinearity. A one-field model for Hasagawa - Mima (H-M) turbulence which also has one nonlinearity (polarization drift) and is used to study dual cascade dynamics. A one-field model of Dissipative Trapped Electron mode (DTEM1) turbulence with both the E×B and the polarization drift nonlinearities which lends itself to the investigation of the interplay between the nonlinearities. And finally a two-field two-nonlinearity model of dissipative trapped electron model turbulence (DTEM2) is used to investigate feedback of two nonlinear effects and nonstationarity. In all of these models the diagnostics used stress the transfer dynamics and included time histories of total energy and enstrophy as well as time histories for individual modes. These are supplemented with direct transfer diagnostics in addition to



frequency and k spectra evolution. With these tools it is possible to investigate the detailed dynamics of the mode evolution and spectral transfer.

The basis for the Kolmogorov spectrum is the assumption that inertial ranges are dominated by a local self-similar transfer or cascade. Using the Kolmogorov spectrum and equilibrium statistical mechanics it is possible to infer the preferred direction of flow for a given nonlinearity. The dynamics of most turbulent systems are governed by nonlinearities which are themselves isotropic so that in the absence of anisotropic driving, damping or geometry the turbulence is also expected to be isotropic. The DTICC model is used to study the dynamics of an anisotropic, nonstandard nonlinearity. It is found that this nonlinearity transfers energy nonlocally from low ky to high ky. This leads to a saturated spectrum much flatter in the ky direction than that predicted by Kolmogorov theory. This should not be surprising as the lack of a local transfer induced similarity range precludes the use of Kolmogorov theory.

Going hand in hand with the concept of local transfer in an inertial range is the idea of conservative cascades. The H-M model is used to study the dynamics of the polarization drift nonlinearity ( $\nabla \phi \times z \cdot \nabla \nabla^2 \phi$) which has two quadratic invariants, energy and enstrophy. These two invariants give rise to the same dual cascade that is familiar from 2-D Navier-Stokes turbulence. In this cascade energy is transferred conservatively to low k and enstrophy is transferred conservatively to high k. The conventional description of the dual cascade was formulated by Kraichnan who used a spectrum of infinite extent to show that energy alone is cascaded in the low k region and enstrophy alone is cascaded in the high k region. It is found here that if the spectrum is not infinite in extent there is some "improper" transfer. That is, in the energy transfer range some enstrophy is transferred and in the enstrophy transfer range some energy is transferred. The amount of "improper" transfer depends on both the extent of the inertial range and the position within that range of the source of free energy (the injection range). It is quite possible to arrange a situation in

which more energy or enstrophy flows in the improper direction then in the proper direction. This is important to keep in mind when analyzing system dynamics from the conservative cascade point of view.

Nonlinearities by definition cannot be linearly superimposed as if they were independent, yet it is assumed that if two nonlinearities have different regions of dominance in k space, in the region dominated by one nonlinearity the other nonlinearity can be ignored. The DTEM1 model was used to investigate the effects of two nonlinearities interacting in one system. It was found that at the extremes in which one dominates the other by more then an order of magnitude it is justifiable to drop the sub-dominant nonlinearity. However, in the often large region where the nonlinearities relative strength are within an order of magnitude of each other, new dynamics occur which are not explainable by a linear super-position of the two nonlinearities. The "cross-coupling" dynamics are qualitatively different and include both a change in the transfer dynamics and a nonlinear frequency shift. The frequency shift can be as large as a few times $\omega^*$ and is found to be proportional to $k_y$. The shift observed from simulation qualitatively agrees with the shift predicted by closure theory. Such shifts could help explain experimentally observed frequencies which seem anomalously shifted away from the frequency expected from linear dispersion relations. The influence of the crossover region creates five effective regimes: 1) at the low k extreme transfer dynamics are governed solely by the E×B nonlinearity; 2) at k values between the crossover region and the region of E×B dominance both the $E \times B$ and the cross-coupling dynamics are important; 3) in the crossover region, the characteristic dynamics are dominated by the interaction of the two nonlinearities and these dynamics are largely independent of the dynamics of the E×B and polarization drift nonlinearities when acting alone; 4) in the region above the crossover region but below the polarization drift dominated region, both the interaction of the nonlinearities and the polarization drift dynamics are important; and 5) at the highest k values the polarization drift



dynamics dominate. Before ignoring a subdominant nonlinearity it may be important to assure oneself that the nonlinearity of interest is sufficiently dominant so as to dominate both the cross-coupling dynamics as well as the subdominant nonlinearity.

In order to self consistently include the feedback effect of the frequency shift on the linear growth and particle flux, it is necessary to extend the model beyond a one-field model to a two-field model. The DTEM2 model is used to study these interaction feedbacks. The most striking result from the two-field studies is the lack of a stationary saturated state. As with most driven damped turbulent systems, this system does reach a saturated state dependent on the various parameters. These parameters include growth, damping and the ratio of E×B to polarization drift nonlinearity strength but, in the case of fully developed turbulence, do not include initial conditions. In this system the saturated state is found to be non-stationary, exhibiting oscillations (cycles) with amplitude of order 50% and oscillation period on the order of 10 eddy turnover times. During these cycles most of the relevant quantities also undergo large fluctuations. These include flux and transfer rates as well as frequency and k spectra. These cycles may be understood in terms of a simple heuristic relaxation oscillation model. It is possible that this type of oscillation may in fact exist in experiment, but is effectively "washed out" by diagnostics that average over a few cycles. Since these models are all local, it is also possible that spatial averaging could wash out the fluctuations. If they do exist it might suggest that transport is a much more locally intermittent event then generally believed.

In conclusion, an outgrowth of this thesis is that there are three areas in which general turbulence lore may not be valid. There do exist nonlinearities for which transfer is not local in k space (E×B nonlinearity) in disagreement with Kolmogorov scaling. It is not always justified to discard subdominant nonlinearities because of the fundamentally new dynamics that can arise from cross-coupling of multiple nonlinearities. And finally, saturated states are not always stationary and can infact exhibit complex cycling with

ramifications for many observable quantities. It is important when dealing with turbulent nonlinearities to be cognizant of these large variety of dynamical behaviors that can be displayed. One should not get complacent in the process of extrapolating from a well known system to an unknown one.



# APPENDIX A

## Computational Model descriptions

The computational results in this thesis are really a compilation of results from four codes of two different types. The first code was a one-field spectral code with one nonlinearity (E×B). This code used an IMSL routines (DB2QBF) to do the time stepping. When the second nonlinearity was added the code was changed to a more portable solver of the same type called LSODE by Alan C. Hindmarsh at LLNL. At that time the Jacobian of the system was explicitly entered and the flow diagnostics where added. The next code was a pseudo-spectral version of the one-field two nonlinearity code. This code was made in order to increase the k space sizes which were practical to run (as pseudo-spectral is more efficient) as well as to serve as a check on the validity of the spectral results. Finally, the two-field two nonlinearity code, as a spectral code. Spectral codes were used because their accuracy is limited only by the specified precision and the number of modes in the truncation. They work by Fourier transforming (in space) the PDE and truncating the Fourier representation at some value of k. This process is exact for most PDE's, if k is taken to go to infinity, which is clearly not possible. However, in most systems there is some physical mechanism which realistically cuts off the k. This can be as simple as kinematic viscosity (or hyperviscosity) or it could be some complicated mechanism by which the model of interest loses validity at high k and must couple to new modes which may be parameterized as a damping or driving at high k. We have typically used both methods combined into one by including viscosity or hyperviscosity as a high k cutoff to simulate either real viscosity or an energy sink into coupling with some new higher k mode. The main drawback of pure spectral codes is that nonlinearities become a convolution in k

space, i.e. $\sum_{k'} k \times k' \cdot z \, [(k_\perp - k_\perp')^2 - k_\perp'^2] \, \phi_{k'} \phi_{k-k'}$ , for each mode.

Clearly, doing $k_n^2$ computations is much less efficient than doing $k_n \ln k_n$. Therefore, the pseudo-spectral method was developed. Pseudo-spectral codes have exactly the same accuracy as spectral codes, as long as the k space they are operating in is 50% larger. The idea behind pseudo spectral codes is that new Fast Fourier Transform methods are efficient enough to made it advantageous to move back and forth between real space and k space with the products (nonlinear terms) being evaluated in real space and the time-advancing (as well as the linear terms) being computed in k space. The computational requirements for the spectral codes versus pseudo spectral codes goes as $n^2$ versus n ln n where n is the total number of modes simulated. This difference approaches a very significant level rather rapidly. Unfortunately, since we are interested in the flow in k space which intrinsically requires calculation of the convolution, we are generally unable to benefit from pseudo spectral codes except as a check of the spectral method.

The systems explored here are extremely stiff systems. This means physically that there are many different time scales simultaneously important in the problem. Mathematically this implies that the Jacobian is sparse. Computationally this means that most simpler time (integrating) methods such as Runga-Kutta do not work well. Even simple predictor-corrector routines had trouble converging hence, the method used was one specifically written for stiff systems. The method employed is the Gear method in the LSODE solver package. This package is an adaptive time stepping predictor-corrector implementation of the Gear solver and is available to the public from netlib. With this routine one is able to obtain results with a fixed time step while allowing the solver to use a variable time step. This is important for some post processing such as frequency analysis. Typically we try to choose a time step that gave approximately 10 points per eddy turnover



time outputted which normally gives approximately 10 computational time steps per outputted time step.

The code itself is portable enough to run on any machine with ANSI compatible FORTRAN 77 or FORTRAN 90 and has been run on everything from a Macintosh II to a Cray C-90. Most of the computation was performed on a DEC station 5000-240, a Cray-2, a Cray-YMP, and a Cray C-90 with some work done on a DEC station 3000-400 AXP ( an alpha box). The largest spectral simulations (51x51) were performed on the C-90 and took up to ten days of CPU time. Most of the FORTRAN post processing such as the frequency spectra analysis routines are run on the DEC station 5000. The final post processing (the imaging) is performed mainly on Apple Macintosh computers. The time plots are made with Kalidagraph, the 3-D plots are made with Wingz, and the movies are made with NCSA Image. The real space contour plots are created on the DEC station using NCAR graphics.

Below is a listing of the spectral code for the 2 field runs:

```
c      main program spec 2disnonlinfull june 6 1993 22:03:03
c     This version of the code is for the full two field version of
c     the dissipative trapped electron  modes.  To use it the full set
c     LSODE subroutines need to be compiled with this main code.
c     In this version the ExB nonlinearity is in the DEN equation
c     and the pol. drift nonlinearity is in the PSI equation.
c     !!To change the maximum size of the kspace kxmax and kymax must
c     !!be changed in every parameter statment in which it appears.
c          set array dimensions and take care of converting
c          from the array name to kx, ky name.  declare all



       parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
c *** parameter statement for specout stuff************************
c     This defines which modes have their full temporal history outputed
       parameter(isav=1,indx1=kxm1-1,indy1=kym1-1,indx2=kxm1-2,
     + indy2=kym1,indx3=kxm1,indy3=kym1-2,
     + indx4=kym1-2,indy4=kym1-2,indx5=kxmax+1+kxmax/2,
     + indy5=kymax+1+kymax/2,indx6=kxm1,indy6=kym1,indx7=kxm2,
     + indy7=kym2+kymax,indx8=kxm2+kxmax,indy8=kym2,
     + indx9=kxm2+1,indy9=kym2+1,indx10=kxm2+2,indy10=kym2+2,
     + indx11=kxm2+2,indy11=kym2,indx12=kxm2,indy12=kym2+2,
```

```
     + indx13=kxmax+1+kxmax/2,indy13=kym2,indx14=kxm2,
     + indy14=kymax+1+kymax/2,indx15=kxmax+1+kxmax/2,indy15=kym2+2,
     + indx16=kxm2+2,indy16=kymax+1+kymax/2,indx17=kxmax/2,
     + indy17=kymax+1+kymax/2,indx18=kxmax+1,indy18=kymax+3,
     + indx19=kxmax+1,indy19=kymax+4,indx20=kxmax+1,indy20=kymax+5,
     + indx21=kxmax+1,indy21=kymax+6,indx22=kxmax+1,indy22=kymax+7,
     + indx23=kxmax+1,indy23=kymax+8,indx24=kxmax+1,indy24=kymax+9,
     + indx25=kxmax+1,indy25=kymax+10,indx26=kxmax+1,indy26=kymax+11,
     + indx27=kxmax+1,indy27=kymax+12,indx28=kxmax+1,indy28=kymax+13,
     + indx29=kxmax+1,indy29=kymax+14,indx30=kxmax+1,indy30=kymax+15,
     + indx31=kxmax+1,indy31=kymax+16,indx32=kxmax+1,indy32=kymax+17,
     + indx33=kxmax+1,indy33=kymax+18,indx34=kxmax+1,indy34=kymax+19,
     + indx35=kxmax+1,indy35=kymax+20,indy36=kymax+1,indx36=kxmax+3,
     + indy37=kymax+1,indx37=kxmax+4,indy38=kymax+1,indx38=kxmax+5,
     + indy39=kymax+1,indx39=kxmax+6,indy40=kymax+1,indx40=kxmax+7,
     + indy41=kymax+1,indx41=kxmax+8,indy42=kymax+1,indx42=kxmax+9,
     + indy43=kymax+1,indx43=kxmax+10,indy44=kymax+1,indx44=kxmax+11,
     + indy45=kymax+1,indx45=kxmax+12,indy46=kymax+1,indx46=kxmax+13,
     + indy47=kymax+1,indx47=kxmax+14,indy48=kymax+1,indx48=kxmax+15,
     + indy49=kymax+1,indx49=kxmax+16,indy50=kymax+1,indx50=kxmax+17,
     + indy51=kymax+1,indx51=kxmax+18,indy52=kymax+1,indx52=kxmax+19,
     + indy53=kymax+1,indx53=kxmax+20)

c     The complex variable declarations

       complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
       complex spec1(30000),spec2(30000),spec3(30000),spec4(30000),
     +spec5(30000),spec6(30000),spec7(30000),spec8(30000),spec9(30000),
     +spec10(30000),spec11(30000),spec12(30000),spec13(30000),
     +spec14(30000),spec15(30000),spec16(30000),spec17(30000),
     +spec18(30000),spec19(30000),spec20(30000),spec21(30000),
     +spec22(30000),spec23(30000),spec24(30000),spec25(30000),
     +spec26(30000),spec27(30000),spec28(30000),spec29(30000),
     +spec31(30000),spec32(30000),spec33(30000),spec34(30000),
     +spec35(30000),spec36(30000),spec37(30000),spec38(30000),
     +spec39(30000),spec30(30000),spec40(30000),spec41(30000),
     +spec42(30000),spec43(30000),spec44(30000),spec45(30000),
     +spec46(30000),spec47(30000),spec48(30000),spec49(30000),
     +spec50(30000),spec51(30000),spec52(30000),spec53(30000)

c     The real variable declarations

       real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos

c     The 3 common block declarations

       common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
```



```
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx,pulhgt,kxpul,kypul,
     + ratio2,kbreak
       common /arrays/
     + psi,den
       common /arrdim/
     + nt,ntstep,pesu

       integer to(15),ti(15),timer(15)
c          dimension the plot variables and then open all the
c          devices used for output or input including unit 1 which
c          is the cray unit for the screen

       real time(10000),plot1(10000),plot2(10000),plot3(10000)
c       call link("unit1=(terminal)//")
      open(4,file='nonindat',status='unknown')
      open(10,file='nonoutdat',status='unknown',recl=500)
      open(7,file='nondat',status='unknown',recl=5000)
      open(8,file='nonpsidat',status='unknown',recl=5000)
      open(9,file='nonrrordat',status='unknown')
      open(11,file='nonspecp',status='unknown',recl=15000)
      open(75,file='nonspecn',status='unknown',recl=15000)
      open(12,file='nonflowe',status='unknown',recl=10000)
      open(13,file='noniflowe',status='unknown',recl=10000)
      open(14,file='nonspkek',status='unknown')
      open(15,file='nonflowp',status='unknown',recl=10000)
      open(16,file='noniflowp',status='unknown',recl=10000)

      open(17,file='nonensdat',status='unknown',recl=5000)
      open(18,file='nonndat',status='unknown',recl=5000)
      open(19,file='nondendat',status='unknown',recl=5000)


      open(21,file='specout1',status='unknown')
      open(22,file='specout2',status='unknown')
      open(23,file='specout3',status='unknown')
      open(24,file='specout4',status='unknown')
      open(25,file='specout5',status='unknown')
      open(26,file='specout6',status='unknown')
      open(27,file='specout7',status='unknown')
      open(28,file='specout8',status='unknown')
      open(29,file='specout9',status='unknown')
      open(30,file='specout10',status='unknown')
      open(31,file='specout11',status='unknown')
      open(32,file='specout12',status='unknown')
      open(33,file='specout13',status='unknown')
      open(34,file='specout14',status='unknown')
c     open(35,file='specout15',status='unknown')
      open(36,file='specout16',status='unknown')
      open(37,file='specout17',status='unknown')
      open(38,file='specout18',status='unknown')
```

```
      open(39,file='specout19',status='unknown')
      open(40,file='specout20',status='unknown')
      open(41,file='specout21',status='unknown')
      open(42,file='specout22',status='unknown')
      open(43,file='specout23',status='unknown')
      open(44,file='specout24',status='unknown')
      open(45,file='specout25',status='unknown')
      open(46,file='specout26',status='unknown')
      open(47,file='specout27',status='unknown')
      open(48,file='specout28',status='unknown')
      open(49,file='specout29',status='unknown')
      open(50,file='specout30',status='unknown')
      open(51,file='specout31',status='unknown')
      open(52,file='specout32',status='unknown')
      open(53,file='specout33',status='unknown')
      open(54,file='specout34',status='unknown')
      open(55,file='specout35',status='unknown')
      open(56,file='specout36',status='unknown')
      open(57,file='specout37',status='unknown')
      open(58,file='specout38',status='unknown')
c     open(59,file='specout39',status='unknown')
c     open(60,file='specout40',status='unknown')
c     open(61,file='specout41',status='unknown')
c     open(62,file='specout42',status='unknown')
c     open(63,file='specout43',status='unknown')
c     open(64,file='specout44',status='unknown')
c     open(65,file='specout45',status='unknown')
c     open(66,file='specout46',status='unknown')
c     open(67,file='specout47',status='unknown')
c     open(68,file='specout48',status='unknown')
c     open(69,file='specout49',status='unknown')
c     open(70,file='specout50',status='unknown')
c     open(71,file='specout51',status='unknown')
c     open(72,file='specout52',status='unknown')
c     open(73,file='specout53',status='unknown')
      open(74,file='gamaout',status='unknown',recl=5000)



c     open(20,file='restart')

c      close(20)

c initialize
c      ierr=izchgtim(to)

c     The subroutine initial sets up and initializes the various arrays
c     and reads in the values of the setable parameters

      call initial

c     This is the begining of the main loop in which the timesteping
c     occurs.  After the actual time stepping routine is called (in
c     timead) the various dianstic and printing routines are called.
c     They are called at varying intervals.  The intervals are
c     determined by the variables nwrt and nwrs.
```



```
c time step

      do 20 i=1,ntstep
      ncl=i
      mid=ntstep/2
      fin=ntstep

c     solve ode's doing the actual time step integrations

      call timead

c     either save or don't depending on the value of isav
c     the psi and den arrays at each (or some) time step
      if(isav .ne. 1)then
      call writdat
      elseif((isav .eq. 1) .and.((ncl .eq. 0).or.
     +  (mod(ncl,10*nwrt) .eq. 0)))then
      call writdat
      endif
c      call writinit

c write the output complex timeseries for a selection of k space modes

      spec1(i)=psi(indx1,indy1)
      spec2(i)=psi(indx2,indy2)
      spec3(i)=psi(indx3,indy3)
      spec4(i)=psi(indx4,indy4)
      spec5(i)=psi(indx5,indy5)
      spec6(i)=psi(indx6,indy6)
      spec7(i)=psi(indx7,indy7)
      spec8(i)=psi(indx8,indy8)
      spec9(i)=psi(indx9,indy9)
      spec10(i)=psi(indx10,indy10)
      spec11(i)=psi(indx11,indy11)
      spec12(i)=psi(indx12,indy12)
      spec13(i)=psi(indx13,indy13)
      spec14(i)=psi(indx14,indy14)
      spec15(i)=psi(indx15,indy15)
      spec16(i)=psi(indx16,indy16)
      spec17(i)=psi(indx17,indy17)
      spec18(i)=psi(indx18,indy18)
      spec19(i)=psi(indx19,indy19)
      spec20(i)=psi(indx20,indy20)
      spec21(i)=psi(indx21,indy21)
      spec22(i)=psi(indx22,indy22)
      spec23(i)=psi(indx23,indy23)
      spec24(i)=psi(indx24,indy24)
      spec25(i)=psi(indx25,indy25)
      spec26(i)=psi(indx26,indy26)
      spec27(i)=psi(indx27,indy27)
      spec28(i)=psi(indx28,indy28)
      spec29(i)=psi(indx29,indy29)
      spec30(i)=psi(indx30,indy30)
      spec31(i)=psi(indx31,indy31)
      spec32(i)=psi(indx32,indy32)
      spec33(i)=psi(indx33,indy33)
```

```
      spec34(i)=psi(indx34,indy34)
      spec36(i)=psi(indx36,indy36)
      spec37(i)=psi(indx37,indy37)
      spec38(i)=psi(indx38,indy38)
      spec39(i)=psi(indx39,indy39)
      spec35(i)=psi(indx35,indy35)
      spec40(i)=psi(indx40,indy40)
      spec41(i)=psi(indx41,indy41)
      spec42(i)=psi(indx42,indy42)
      spec43(i)=psi(indx43,indy43)
      spec44(i)=psi(indx44,indy44)
      spec45(i)=psi(indx45,indy45)
      spec46(i)=psi(indx46,indy46)
      spec47(i)=psi(indx47,indy47)
      spec48(i)=psi(indx48,indy48)
      spec49(i)=psi(indx49,indy49)
      spec50(i)=psi(indx50,indy50)
      spec51(i)=psi(indx51,indy51)
      spec52(i)=psi(indx52,indy52)
      spec53(i)=psi(indx53,indy53)


      n1=i
      eta1=d
      nue1=nue
      epsi1=epsi
      dt1=dt
      ntstep1=ntstep
      call spwrite(spec1,spec2,spec3,spec4,spec5,spec6,
     + spec7,spec8,spec9,spec10,spec11,spec12,spec13,
     + spec14,spec15,spec16,spec17,spec18,spec19,spec20,
     + spec21,spec22,spec23,spec24,spec25,spec26,spec27,
     + spec28,spec29,spec30,spec31,spec32,spec33,spec34,
     + spec35,spec36,spec37,spec38,spec39,spec40,spec41,
     + spec42,spec43,spec44,spec45,spec46,spec47,spec48,
     + spec49,spec50,spec51,spec52,spec53,n1,eta1,nue1,epsi1,
     + dt1,ntstep1)


c       get energy's and mean sq flux, growth rates

      call energy

c          every nwrt steps write to the plot arrays and to
c          various spectum output arrays

      if(mod(ncl,nwrt).eq.0) then

c          write to print diagnostics arrays

      time(i)=t
      plot2(i)=ek

c          write to spectum output arrays

      call writeit
```



```
       call writesp
       endif

c  write the a selection of flux and psi**2 and den**2 arrays every
c     nwrs timesteps
c     call the flow diagnostic routines and write the results every
c     nwrs timesteps
c       write(6,*)("i=",i)

        call writear
        call kflowe
        call kflowp

        if((ncl .eq. 1) .or. (mod(ncl,nwrs).eq.0))then
        call writksp
        call storear
        call writflwe
        call writflwp
       endif

  20    continue

c      finish up
c 100    ierr=izchgtim(ti)
c        if(ti(13) .le. 0) goto 100
c        do 200 i=1,13
c 200    timer(i)=ti(i)-to(i)
c        overlap=real(timer(1))/real(timer(13))

c         write(6,*) (timer(i),i=1,13)
c         write(6,*)"overlap=",overlap
c         write(10,*) (timer(i),i=1,13)
c         write(10,*)"overlap=",overlap

      call final
c      call plot(time,plot1,plot2,plot3)
      write(6,*)"program terminated normally"
      stop
      end


c*************************************************************

      subroutine spwrite(spec1,spec2,spec3,spec4,spec5,spec6,
     + spec7,spec8,spec9,spec10,spec11,spec12,spec13,
     + spec14,spec15,spec16,spec17,spec18,spec19,spec20,
     + spec21,spec22,spec23,spec24,spec25,spec26,spec27,
     + spec28,spec29,spec30,spec31,spec32,spec33,spec34,
     + spec35,spec36,spec37,spec38,spec39,spec40,spec41,
     + spec42,spec43,spec44,spec45,spec46,spec47,spec48,
     + spec49,spec50,spec51,spec52,spec53,n,eta,nue,epsi,
     + dt,ntstep)

c     This subroutine saves the mode histories for a selected modes
c     it also calculates the total growth rates for some modes
```

```
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)

      complex spec1(30000),spec2(30000),spec3(30000),spec4(30000),
     + spec5(30000),spec6(30000),spec7(30000),spec8(30000),
     + spec9(30000),spec10(30000),spec11(30000),spec12(30000),
     + spec13(30000),spec14(30000),spec15(30000),spec16(30000),
     + spec17(30000),ic,
     +spec18(30000),spec19(30000),spec20(30000),spec21(30000),
     +spec22(30000),spec23(30000),spec24(30000),spec25(30000),
     +spec26(30000),spec27(30000),spec28(30000),spec29(30000),
     +spec31(30000),spec32(30000),spec33(30000),spec34(30000),
     +spec35(30000),spec36(30000),spec37(30000),spec38(30000),
     +spec39(30000),spec30(30000),spec40(30000),spec41(30000),
     +spec42(30000),spec43(30000),spec44(30000),spec45(30000),
     +spec46(30000),spec47(30000),spec48(30000),spec49(30000),
     +spec50(30000),spec51(30000),spec52(30000),spec53(30000)
        real a,b,gama(30000,30)

        i=n
        write(21,*)spec1(i)
        write(22,*)spec2(i)
        write(23,*)spec3(i)
        write(24,*)spec4(i)
        write(25,*)spec5(i)
        write(26,*)spec6(i)
        write(27,*)spec7(i)
        write(28,*)spec8(i)
        write(29,*)spec9(i)
        write(30,*)spec10(i)
        write(31,*)spec11(i)
        write(32,*)spec12(i)
        write(33,*)spec13(i)
        write(34,*)spec14(i)
c       write(35,*)spec15(i)
        write(36,*)spec16(i)
        write(37,*)spec17(i)
        write(38,*)spec18(i)
        write(39,*)spec19(i)
        write(40,*)spec20(i)
        write(41,*)spec21(i)
        write(42,*)spec22(i)
        write(43,*)spec23(i)
        write(44,*)spec24(i)
        write(45,*)spec25(i)
        write(46,*)spec26(i)
        write(47,*)spec27(i)
        write(48,*)spec28(i)
        write(49,*)spec29(i)
        write(50,*)spec30(i)
        write(51,*)spec31(i)
        write(52,*)spec32(i)
        write(53,*)spec33(i)
        write(54,*)spec34(i)
```



```
        write(55,*)spec35(i)
        write(56,*)spec36(i)
        write(57,*)spec37(i)
        write(58,*)spec38(i)
c       write(59,*)spec39(i)
c       write(60,*)spec40(i)
c       write(61,*)spec41(i)
c       write(62,*)spec42(i)
c       write(63,*)spec43(i)
c       write(64,*)spec44(i)
c       write(65,*)spec45(i)
c       write(66,*)spec46(i)
c       write(67,*)spec47(i)
c       write(68,*)spec48(i)
c       write(69,*)spec49(i)
c       write(70,*)spec50(i)
c       write(71,*)spec51(i)
c       write(72,*)spec52(i)
c       write(73,*)spec53(i)

        if(n.eq.ntstep)then
        ic=(0.,1.)
        a=sqrt(epsi)/nue
        b=a*eta
        ky5=kym2+kymax/2
        ky9=kym2+1
        ky10=kym2+2
        ky18=kym2+2
        ky19=kym2+3
        ky20=kym2+4
        ky21=kym2+5
        ky22=kym2+6
        ky23=kym2+7
        ky24=kym2+8
        ky25=kym2+9
        ky26=kym2+10
        ky27=kym2+11
        do 100 i=1,ntstep-2
          gama(i,1)=real(((-a*(spec5(i)-2.*spec5(i+1)
     +  +spec5(i+2))/dt**2)-(b*ky5*ic*(spec5(i+2)-spec5(i))
     +  /2.*dt))/spec5(i+1))
          gama(i,2)=real(((-a*(spec9(i)-2.*spec9(i+1)
     +  +spec9(i+2))/dt**2)-(b*ky9*ic*(spec9(i+2)-spec9(i))
     +  /2.*dt))/spec9(i+1))
          gama(i,3)=real(((-a*(spec10(i)-2.*spec10(i+1)
     +  +spec10(i+2))/dt**2)-(b*ky10*ic*(spec10(i+2)-spec10(i))
     +  /2.*dt))/spec10(i+1))
          gama(i,4)=real(((-a*(spec18(i)-2.*spec18(i+1)
     +  +spec18(i+2))/dt**2)-(b*ky18*ic*(spec18(i+2)-spec18(i))
     +  /2.*dt))/spec18(i+1))
          gama(i,5)=real(((-a*(spec19(i)-2.*spec19(i+1)
     +  +spec19(i+2))/dt**2)-(b*ky19*ic*(spec19(i+2)-spec19(i))
     +  /2.*dt))/spec19(i+1))
          gama(i,6)=real(((-a*(spec20(i)-2.*spec20(i+1)
     +  +spec20(i+2))/dt**2)-(b*ky20*ic*(spec20(i+2)-spec20(i))
     +  /2.*dt))/spec20(i+1))
```

```
        gama(i,7)=real(((-a*(spec21(i)-2.*spec21(i+1)
     +  +spec21(i+2))/dt**2)-(b*ky21*ic*(spec21(i+2)-spec21(i))
     +  /2.*dt))/spec21(i+1))
        gama(i,8)=real(((-a*(spec22(i)-2.*spec22(i+1)
     +  +spec22(i+2))/dt**2)-(b*ky22*ic*(spec22(i+2)-spec22(i))
     +  /2.*dt))/spec22(i+1))
        gama(i,9)=real(((-a*(spec23(i)-2.*spec23(i+1)
     +  +spec23(i+2))/dt**2)-(b*ky23*ic*(spec23(i+2)-spec23(i))
     +  /2.*dt))/spec23(i+1))
        gama(i,10)=real(((-a*(spec24(i)-2.*spec24(i+1)
     +  +spec24(i+2))/dt**2)-(b*ky24*ic*(spec24(i+2)-spec24(i))
     +  /2.*dt))/spec24(i+1))
        gama(i,11)=real(((-a*(spec25(i)-2.*spec25(i+1)
     +  +spec25(i+2))/dt**2)-(b*ky25*ic*(spec25(i+2)-spec25(i))
     +  /2.*dt))/spec25(i+1))
        gama(i,12)=real(((-a*(spec26(i)-2.*spec26(i+1)
     +  +spec26(i+2))/dt**2)-(b*ky26*ic*(spec26(i+2)-spec26(i))
     +  /2.*dt))/spec26(i+1))
        gama(i,13)=real(((-a*(spec27(i)-2.*spec27(i+1)
     +  +spec27(i+2))/dt**2)-(b*ky27*ic*(spec27(i+2)-spec27(i))
     +  /2.*dt))/spec27(i+1))
100     continue
        do 110 i=1,ntstep-2

        write(74,*) (gama(i,j),j=1,13)

110     continue
        endif


        return
        end

c*************************************************************
        subroutine initial
        parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
        complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
        real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
        common /parms/
     + vl,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
```



```
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
        common /arrays/
     + psi,den
        common /arrdim/
     + nt,ntstep,pesu
c
        character*24 fdate
        external fdate
c       set quantities
        t=0.
        ekold=0.
        enold=0.


        write(10,*)fdate()

c   read in parameters
        call readit
c  initial perturbation
        call pert
c  write initial perturbation
        call energy
        write(10,*) " ncl  t  ek  en  ed  et  flux  phase gamk  gamen"
        write(6,*) " ncl  t  ek  en  ed  et  flux  phase gamk  gamen"
        write(11,*) "ekk(0,0)  ekk(0,1)  ekk(2)  ekk(3)  ekk(4)  ekk(1,0)
     + ekk(2)  ekk(3)  ekk(4)  ekk(3,3)"
        write(12,300) (m,m=1,kfxmax),(m,m=1,kfymax)
        write(13,400) (m,m=1,kfxmax),(m,m=1,kfymax)
        write(15,301) (m,m=1,kfxmax),(m,m=1,kfymax)
        write(16,401) (m,m=1,kfxmax),(m,m=1,kfymax)

        call writdat
        call writeit
        call writear
        call writksp
        call storear
c
 300     format(3('flowerx',i2,2x),3('flowery',i2,2x))
 400     format(3('floweix',i2,2x),3('floweiy',i2,2x))
 301     format(3('flowprx',i2,2x),3('flowpry',i2,2x))
 401     format(3('enflwpx',i2,2x),3('enflowpy',i2,2x))
 500     format(a8,3x,a8)
        return
        end


c*************************************************************

        subroutine readit
        parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
        complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
```

```
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
       real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
       common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx,pulhgt,kxpul,kypul,
     + ratio2,kbreak
       common /arrays/
     + psi,den
       common /arrdim/
     + nt,ntstep,pesu
c      set defaults
c
c           choose whether you want to enter the data parameters
c           or use the ones in the code

       integer ichoice
       write (6,*) "type 2 for your file indat to enter the data,"
       write (6,*) "type 1 if you want to enter the data, type o "
       write (6,*) "if you want to use the values in the code"
       read (5,*) ichoice

       if (ichoice .eq. 0)then

c           input from within code the various data parameters

        data dt/0.01/,ntstep/100/,nwrt/10/,nwrs/100/,tol/0.001/
        data v1/1./,gama1/1./,gama2/1./,epsi/.001/,kxg/1/,kyg/1/
        data amp/2./,kxs/1/,kys/1/,ratio/1./,d/1./,dout/0./,nue/1./
        data init/2/,psi0/.02/,flreal/.1/,flima/.1/,loc/1/
        data dmid/0./,gama3/1./,kxnorm/1./,kynorm/1./,le/1./
        data leout/1./,lemid/1./,lpout/1./,lpmid/1./,lp/1./
        data pulhgt/10./,kxpul/4/,kypul/4/,kbreak/4/,ratio2/2./
        data emass/1./,dfirst/0./,rhos/.01/
c


        elseif (ichoice .eq. 1)then

c             or interactively input the various data parameters

       write (6,*) "enter the values comma delimited"
       write (6,*) "dt,ntstep,nwrt,nwrs,tol,kxnorm,kynorm,loc"
       read (5,*) dt,ntstep,nwrt,nwrs,tol,kxnorm,kynorm,loc
       write (6,*) "v1,gama1,gama2,gama3,epsi,d,dout,kxg,kyg"
       read (5,*) v1,gama1,gama2,gama3,epsi,d,dout,kxg,kyg
```



```
       write (6,*) "le,lemid,leout,lp,lpmid,lpout"
       read (5,*) le,lemid,leout,lp,lpmid,lpout
       write (6,*) "amp(mult of tol to get atol),dmid,kxs,kys,
     + ratio(init slope),nue,emass,dfirst"
       read (5,*) amp,dmid,kxs,kys,ratio,nue,emass,dfirst
       write (6,*) "init,psi0,flreal,flima"
       read (5,*) init,psi0,flreal,flima
       write (6,*) "pulhgt,kxpul,kypul,kbreak,ratio2,rhos"
       read (5,*) pulhgt,kxpul,kypul,kbreak,ratio2,rhos


       else

c             or from input file (device 4)

c


        read (4,*)dt,ntstep,nwrt,nwrs,tol,kxnorm,kynorm,loc
        read (4,*)v1,gama1,gama2,gama3,epsi,d,dout,kxg,kyg
        read (4,*)le,lemid,leout,lp,lpmid,lpout
        read (4,*)amp,dmid,kxs,kys,ratio,nue,emass,dfirst
        read (4,*)init,psi0,flreal,flima
        read (4,*)pulhgt,kxpul,kypul,kbreak,ratio2,rhos


        endif

c            namelist input

        namelist /runp/ dt,ntstep,nwrt,nwrs,tol,kxnorm,kynorm,loc
        namelist /physp/ v1,gama1,gama2,gama3,epsi,d,dout,kxg,kyg
        namelist /physp2/ le,lemid,leout,lp,lpmid,lpout
        namelist /intilp/ amp,dmid,kxs,kys,ratio,nue,emass,dfirst
        namelist /pertin/ init,psi0,flreal,flima
        namelist /pulse/ pulhgt,kxpul,kypul,kbreak,ratio2,rhos

c


        write(6,*)'xsize by ysize ',kxm1,' by ',kym1
        write (6,runp)
        write (6,physp)
        write (6,physp2)
        write (6,intilp)
        write (6,pertin)
        write (6,pulse)

        write (10,*)'xsize by ysize ',kxm1,' by ',kym1
        write (10,runp)
        write (10,physp)
        write (10,physp2)
        write (10,intilp)
        write (10,pertin)
        write (10,pulse)
c
       return
       end
```

```
c*************************************************************


      subroutine endit(ifail,cout)

c           called if there is an error in the intergrater
c           see the integrater documentation for specific meaning

      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold
      common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
      write(10,*) ifail,cout
      write(6,*) ifail,cout
  100  format(1x,'error',2x,'ifail= ',i6,' cout(1)= ',i6)
      write(6,*) "dgeqrf error termination"
      stop
      end



c*************************************************************

      subroutine writeit
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      parameter(iscreen=1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
```



```
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
      write(10,200) ncl,t,ek,en,ed,et,flux,phase,gamk,gamen
      if(iscreen .ne. 1)then
      write(6,200) ncl,t,ek,en,ed,et,flux,phase,gamk,gamen
      endif
  200  format(1x,i6,9(1x,e12.4))
      return
      end

c*************************************************************

      subroutine writesp
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
```

```
      real ekk(kxm1,kym1),edd(kxm1,kym1),nphi(kxm1,kym1)
c  calculate the energy in each mode
      do 20 j=1,kym1
        do 10 i=1,kxm1
          xk=kxnorm*(i-kxm2)
          yk=kynorm*(j-kym2)
          ak=sqrt(xk**2+yk**2)
          ekk(i,j)=psi(i,j)*conjg(psi(i,j))*2.
          edd(i,j)=den(i,j)*conjg(den(i,j))*2.
          nphi(i,j)=aimag(ak*den(i,j)*conjg(psi(i,j)))*2.
 10     continue
 20   continue
      write(11,*) (ekk(kxm2,j),j=kym2+1,kym1),(ekk(i,kym2),
     +  i=kxm2+1,kxm1),ekk(kxm2+2,kym2+2),ekk(kxm2+3,kym2+3),
     +  ekk(kxm1,kym1),(nphi(kxm2,j),j=kym2+1,kym1),
     + (nphi(i,kym2),i=kxm2+1,kxm1)
      write(75,*) (edd(kxm2,j),j=kym2+1,kym1),(edd(i,kym2),
     +  i=kxm2+1,kxm1),edd(kxm2+2,kym2+2),edd(kxm2+3,kym2+3),
     +  edd(kxm1,kym1)


 200   format( 50(e13.6,1x) )
      return
      end

c*************************************************************

      subroutine writear

      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
```



```
      real ekk(kxm1,kym1),eden(kxm1,kym1)
c  calculate the energy in each mode

      do 20 j=1,kym1
        do 10 i=1,kxm1
          xk=kxnorm*(i-kxm2)
          yk=kynorm*(j-kym2)
          ekk(i,j)=psi(i,j)*conjg(psi(i,j))*2.+ekkold(i,j)
          ekkold(i,j)=ekk(i,j)
          eden(i,j)=den(i,j)*conjg(den(i,j))*2.+edenold(i,j)
          edenold(i,j)=eden(i,j)


 10     continue
 20   continue
      return
      end
c*************************************************************
      subroutine writdat

      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu

      do 20 j=1,kym1

       write(7,*)(psi(i,j),i=1,kxm1)
       write(18,*)(den(i,j),i=1,kxm1)

 20   continue
      write(7,*)
      write(7,*)
      write(18,*)
```

```
      write(18,*)

      return
      end

c*************************************************************

c       subroutine writinit

c       use main
c       open(20,file='restart')

c       do 20 i=1,kxm1

c        write(20,*)(psi(i,j),j=1,kym1)

c  20   continue

c       close(20)
c       return
c       end

c*************************************************************

      subroutine storear
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + vl,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
      real ekk(kxm1,kym1),ensk(kxm1,kym1),eden(kxm1,kym1)
      do 20 j=1,kym1
        do 10 i=1,kxm1

        xk=kxnorm*(i-kxm2)
```



```
        yk=kynorm*(j-kym2)
        ak=xk**2+yk**2

        if(ncl .le. 10)then
        ekk(i,j)=ekkold(i,j)
        ensk(i,j)=ekkold(i,j)*ak
        eden(i,j)=edenold(i,j)
        else
        ekk(i,j)=ekkold(i,j)/real(nwrs)
        ensk(i,j)=ekkold(i,j)*ak/real(nwrs)
        eden(i,j)=edenold(i,j)/real(nwrs)
        endif
        ekkold(i,j)=0.0
        edenold(i,j)=0.0

  10    continue
  20  continue

c  write the energies in array form
      do 30 i=kxm2,kxm1

        write(8,*) (ekk(i,j),j=1,kym1)
        write(17,*) (ensk(i,j),j=1,kym1)
        write(19,*) (eden(i,j),j=1,kym1)
  30  continue

      write(8,*)
      write(8,*)

      write(17,*)
      write(17,*)

      write(19,*)
      write(19,*)


  200   format(1p50e13.5)
      return
      end

c*************************************************************
      subroutine writksp
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
```

```
     + vl,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
       common /arrays/
     + psi,den
       common /arrdim/
     + nt,ntstep,pesu
       real ekk(kxm1,kym1)

       do 20 j=1,kym1
         do 10 i=1,kxm1

           xk=kxnorm*(i-kxm2)
           yk=kynorm*(j-kym2)

         if(ncl .le. 10)then
         ekk(i,j)=ekkold(i,j)
         else
         ekk(i,j)=ekkold(i,j)/real(nwrs)
         endif

  10     continue
  20   continue

c  write the k**2 vs em,ek and flux in array form
       do 30 i=kxm2,kxm1
        do 40 j=kym2,kym1

           xk=kxnorm*(i-kxm2)
           yk=kynorm*(j-kym2)
           ak=xk**2+yk**2

         write(14,*) ekk(i,j) ,  ak
  40    continue
  30   continue

       write(14,*)
       write(14,*)

  200   format(50(e12.4,1x))

       return
       end

c*************************************************************

       subroutine writeer(h,ier)
       parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
       complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
```



```
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
       real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
       common /parms/
     + vl,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
       common /arrays/
     + psi,den
       common /arrdim/
     + nt,ntstep,pesu
       if (ier .eq. 33) then
       write(9,*) 'h=',h,'ier=',ier,'tol may be too
     + low or equations too stiff, t=t+h'
       write(6,*) 'h=',h,'ier=',ier,'tol may be too
     + low or equations too stiff, t=t+h'
       elseif (ier .eq. 66 .or. ier .eq. 67) then
       write(9,*) 'h=',h,'ier=',ier,'h was decreased to
     + force convergence'
       write(6,*) 'h=',h,'ier=',ier,'h was decreased to
     + force convergence'
       else
       write(9,*) 'h=',h,'ier=',ier,'unknown error'
       write(6,*) 'h=',h,'ier=',ier,'unknown error'
       endif
       return
       end

c*************************************************************

       subroutine energy
       parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
       complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
       real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
       common /parms/
```

```
     + vl,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
       common /arrays/
     + psi,den
       common /arrdim/
     + nt,ntstep,pesu
       real emin,ekin,msfin,esin
       integer pesuin
       namelist /maginp/ em,ek,msf,es,pesu

c  initialization


       ek=0.
       en=0.
       ed=0.
       et=0.
       es=1.0e-25
       flux=0.
       phase=0.
       number=0


c   calculate energies
       i1=1
       i2=kxmax
       j1=kym2
       j2=kym1
  10    do 20 i=i1,i2
           xk=kxnorm*(i-kxm2)
          do 20 j=j1,j2
           yk=kynorm*(j-kym2)
           ak=xk**2+yk**2
           number=number+1

           ek=ek+psi(i,j)*conjg(psi(i,j))*2
           en=en+psi(i,j)*conjg(psi(i,j))*ak*2
           ed=ed+den(i,j)*conjg(den(i,j))*2
           flux=flux-sqrt(ak)*aimag(den(i,j)*conjg(psi(i,j)))*2
           phase=phase+atan(aimag(den(i,j))/real(den(i,j)))-
     +     atan(aimag(psi(i,j))/real(psi(i,j)))


  20    continue
       if(xk .eq. kxmax*kxnorm .and. yk .eq. kymax*kynorm)goto 30
       i1=kxm2
       i2=kxm1
       j1=kym2+1
       goto 10
  30     continue
```



```
        et=ek+ed
        phase=phase/number
c  calculate growth rates

       apsi=amax1((en+enold)/2,es)
       bpsi=amax1((ek+ekold)/2,es)

       gamen=((en-enold)/dt)/apsi
       gamk=((ek-ekold)/dt)/bpsi
c save old energies

       ekold=ek
       enold=en
c
       return
       end

c*************************************************

       subroutine kflowe
       parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
       complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
       real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
       common /parms/
     + vl,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
       common /arrays/
     + psi,den
       common /arrdim/
     + nt,ntstep,pesu
       complex u(kxm1,kym1),ekxflwe,ekyflwe,ekxloce,ekxnone,
     +  ekyloce,ekynone,enyeloc,enyenon,enyeflw,enxeloc,enxenon,
     +  enxeflw,v(kxm1,kym1)
       i1=1
       i2=kxm1
       j1=1
       j2=kym1

       do 20 i=i1,i2
```

```
        do 20 j=j1,j2
          u(i,j)=den(i,j)
          v(i,j)=den(i,j)
20      continue

      do 30 m=1,kfxmax

      kxl=m

      call kflowe2(u,v,kxl,ekxloce,ekxnone,enxeloc,enxenon)

      ekxflwe=ekxloce+ekxnone
      enxeflw=enxeloc+enxenon

      flowexl(m)=flowexl(m)+ekxloce
      enleflwx(m)=enleflwx(m)+enxeloc
      flowexn(m)=flowexn(m)+ekxnone
      enneflwx(m)=enneflwx(m)+enxenon
      flowex(m)=flowex(m)+ekxflwe
      eneflowx(m)=eneflowx(m)+enxeflw

30      continue

      do 40 m1=1,kfymax

      kyl=m1

      call kflowe1(u,v,kyl,ekyloce,ekynone,enyeloc,enyenon)

      ekyflwe=ekyloce+ekynone
      enyeflw=enyeloc+enyenon

      floweyl(m1)=floweyl(m1)+ekyloce
      enleflwy(m1)=enleflwy(m1)+enyeloc
      floweyn(m1)=floweyn(m1)+ekynone
      enneflwy(m1)=enneflwy(m1)+enyenon
      flowey(m1)=flowey(m1)+ekyflwe
      eneflowy(m1)=eneflowy(m1)+enyeflw

40    continue


      return

      end


c*************************************************

      subroutine kflowe1(u,v,kyl,ekyloce,ekynone,enyeloc,enyenon)
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
```



```
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + vl,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
      complex u(kxm1,kym1),dnyloce,dnynone,ekyloce,ekynone,ic,
     +  enyeloc,enyenon,v(kxm1,kym1)
      integer kyl
      ekyloce=(0.,0.)
      ekynone=(0.,0.)
      enyeloc=(0.,0.)
      enyenon=(0.,0.)
       ic=(0.,1.)
      i1=1
      i2=kxm1
      j1=1
      j2=kym1
10      do 20 i=i1,i2
          kxx=i-kxm2
          do 20 j=j1,j2
          kyy=j-kym2

          ak=(kyy*kynorm)**2+(kxx*kxnorm)**2

          if(iabs(kyy) .le. kyl)then

          call yflowe(u,v,kyl,kxx,kyy,dnyloce,dnynone)

          ekyloce=ekyloce-le*d*.5*dnyloce*conjg(v(i,j))
          ekynone=ekynone-le*d*.5*dnynone*conjg(v(i,j))
          enyeloc=enyeloc-le*d*.5*ak*dnyloce*conjg(v(i,j))
          enyenon=enyenon-le*d*.5*ak*dnynone*conjg(v(i,j))

          endif

20      continue

c       if (kxx .eq. kxmax .and. kyy .eq. kymax) goto 30
c       i1=kxm2
c       i2=kxm1
c       j1=2
```

```
c       goto 10

  30    return

        end


c*****************************************************
        subroutine kflowe2(u,v,kxl,ekxloce,ekxnone,enxeloc,enxenon)
        parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
        complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
        real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
        common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
        common /arrays/
     + psi,den
        common /arrdim/
     + nt,ntstep,pesu
        complex u(kxm1,kym1),dnxloce,dnxnone,ekxloce,ekxnone,ic,
     +  enxeloc,enxenon,v(kxm1,kym1)
        integer kxl
        ekxloce=(0.,0.)
        ekxnone=(0.,0.)
        enxeloc=(0.,0.)
        enxenon=(0.,0.)
        ic=(0.,1.)
        il=1
        i2=kxmax
        j1=1
        j2=kym2
  60      do 70 i=il,i2
            kxx=i-kxm2
            do 70 j=j1,j2
            kyy=j-1

            ak=(kyy*kynorm)**2+(kxx*kxnorm)**2

            if(iabs(kxx) .le. kxl)then
```



```
            call xflowe(u,v,kxl,kxx,kyy,dnxloce,dnxnone)

            ekxloce=ekxloce-le*d*.5*dnxloce*conjg(v(i,j+kymax))
            ekxnone=ekxnone-le*d*.5*dnxnone*conjg(v(i,j+kymax))
            enxeloc=enxeloc-le*d*.5*ak*dnxloce*conjg(v(i,j+kymax))
            enxenon=enxenon-le*d*.5*ak*dnxnone*conjg(v(i,j+kymax))

            endif

  70     continue
        if (kxx .eq. kxmax .and. kyy .eq. kymax) goto 80
        il=kxm2
        i2=kxm1
        j1=2
        goto 60

  80    return

        end


c*****************************************************
        subroutine yflowe(u,v,kyl,kxx,kyy,wyloce,wynone)
        parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
        complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
        real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
        common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
        common /arrays/
     + psi,den
        common /arrdim/
     + nt,ntstep,pesu

        complex u(kxm1,kym1),v(kxm1,kym1),wyloce,wynone,
     +  wylloce,wylnone
        integer kyl
c intialization
c process
```

```
      wyloce=(0.0,0.0)
      wynone=(0.0,0.0)
      i1=1
      i2=kxm1+kxx
      if (kxx .gt.  0) then
      i1=kxx+1
      i2=kxm1
      end if


      j1=1
      j2=kym1+kyy
      if(kyy .gt. 0)then
      j1=kyy+1
      j2=kym1
      endif

           do 20 i=i1,i2
           do 20 j=j1,j2
             kx1=i-kxm2
             ky1=j-kym2
             kxmod=kxx-kx1
             kymod=kyy-ky1
             kxm=kxm2+kxmod
             kym=kym2+kymod
      if(((iabs(ky1) .gt. ky1) .or. (iabs(kyy-ky1) .gt. ky1)).and.
     +  ((iabs(ky1-ky1).le.loc).or.(iabs((kyy-ky1)-ky1).le.loc)))then

c convolute

          wy1loce=(u(i,j)*v(kxm,kym)-u(kxm,kym)*v(i,j))
          wy1loce=wy1loce*kxnorm*kynorm*(ky1*kxx-kx1*kyy)*.5
          wyloce=wyloce+wy1loce

      elseif(((iabs(ky1) .gt. ky1) .or. (iabs(kyy-ky1) .gt. ky1)).and.
     +  ((iabs(ky1-ky1).gt.loc).or.(iabs((kyy-ky1)-ky1).gt.loc)))then
          wy1none=(u(i,j)*v(kxm,kym)-u(kxm,kym)*v(i,j))
          wy1none=wy1none*kynorm*kxnorm*(ky1*kxx-kx1*kyy)*.5
          wynone=wynone+wy1none
      endif

 20       continue

      return

      end


c*************************************************
      subroutine xflowe(u,v,kx1,kxx,kyy,wxloce,wxnone)
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpx1(kfxmax),flowpxn(kfxmax),flowpy1(kfymax),flowpyn(kfymax),
```



```
     + flowex1(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
       real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
       common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowex1,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpx1,flowpxn,flowpy1,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
       common /arrays/
     + psi,den
       common /arrdim/
     + nt,ntstep,pesu

       complex u(kxm1,kym1),v(kxm1,kym1),wxloce,wxnone,
     +  wx1loce,wx1none
        integer kx1
c intialization
c process

      wxloce=(0.0,0.0)
      wxnone=(0.0,0.0)
      i1=1
      i2=kxm1+kxx
      if (kxx .gt.  0) then
      i1=kxx+1
      i2=kxm1
      end if
           do 20 i=i1,i2
           do 20 j=kyy+1,kym1
             kx1=i-kxm2
             ky1=j-kym2
             kxmod=kxx-kx1
             kymod=kyy-ky1
             kxm=kxm2+kxmod
             kym=kym2+kymod

      if(((iabs(kx1) .gt. kx1) .or. (iabs(kxx-kx1) .gt. kx1)).and.
     +  ((iabs(kx1-kx1).le.loc).or.(iabs((kxx-kx1)-kx1).le.loc)))then

c convolute

          wx1loce=(u(i,j)*v(kxm,kym)-u(kxm,kym)*v(i,j))
          wx1loce=wx1loce*kynorm*kxnorm*(ky1*kxx-kx1*kyy)*.5
          wxloce=wxloce+wx1loce

      elseif(((iabs(kx1) .gt. kx1) .or. (iabs(kxx-kx1) .gt. kx1)).and.
     +  ((iabs(kx1-kx1).gt.loc).or.(iabs((kxx-kx1)-kx1).gt.loc)))then
```

```
        wx1none=(u(i,j)*v(kxm,kym)-u(kxm,kym)*v(i,j))
        wx1none=wx1none*kynorm*kxnorm*(ky1*kxx-kx1*kyy)*.5
        wxnone=wxnone+wx1none

      endif
 20       continue

      return

      end



c*****************************************************
      subroutine writflwe
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
      real flowerx(kfxmax),floweix(kfxmax),flowery(kfymax),
     + floweiy(kfymax),flwerxl(kfxmax),flwerxn(kfxmax),flweixl(kfxmax),
     + flweixn(kfxmax),flweryl(kfymax),flweryn(kfymax),
     + flweiyl(kfymax),flweiyn(kfymax)

      do 30 m=1,kfxmax
       flowerx(m)=real(flowex(m))/real(nwrs)
       floweix(m)=real(eneflowx(m))/real(nwrs)
       flwerxl(m)=real(flowexl(m))/real(nwrs)
       flwerxn(m)=real(flowexn(m))/real(nwrs)
       flweixl(m)=real(enleflwx(m))/real(nwrs)
       flweixn(m)=real(enneflwx(m))/real(nwrs)
 30    continue
```



```
      do 50 m1=1,kfymax
       flowery(m1)=real(flowey(m1))/real(nwrs)
       floweiy(m1)=real(eneflowy(m1))/real(nwrs)
       flweryl(m1)=real(floweyl(m1))/real(nwrs)
       flweryn(m1)=real(floweyn(m1))/real(nwrs)
       flweiyl(m1)=real(enneflwy(m1))/real(nwrs)
       flweiyn(m1)=real(enneflwy(m1))/real(nwrs)
 50    continue

      write(12,*)(flowerx(m),m=1,kfxmax),(flowery(m),m=1,kfymax),
     +  (flwerxl(m),m=1,kfxmax),(flweryl(m),m=1,kfymax),
     +  (flwerxn(m),m=1,kfxmax),(flweryn(m),m=1,kfymax)
      write(13,*)(floweix(m),m=1,kfxmax),(floweiy(m),m=1,kfymax),
     +  (flweixl(m),m=1,kfxmax),(flweiyl(m),m=1,kfymax),
     +  (flweixn(m),m=1,kfxmax),(flweiyn(m),m=1,kfymax)


      do 20 j=1,kfxmax
      do 20 j1=1,kfymax
      flowex(j)=(0.,0.)
      flowey(j1)=(0.,0.)
      flowexl(j)=(0.,0.)
      flowexn(j)=(0.,0.)
      floweyl(j1)=(0.,0.)
      floweyn(j1)=(0.,0.)

      eneflowx(j)=(0.,0.)
      enleflwx(j)=(0.,0.)
      enneflwx(j)=(0.,0.)
      eneflowy(j1)=(0.,0.)
      enleflwy(j1)=(0.,0.)
      enneflwy(j1)=(0.,0.)

 20   continue

 200   format(50(e12.4,1x))
      return

      end

c*****************************************************

      subroutine kflowp
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
```

```
      common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
      complex u(kxm1,kym1),ekxflwp,ekyflwp,ekxlocp,ekxnonp,
     + ekylocp,ekynonp,enyloc,enynon,enyflw,enxloc,enxnon,enxflw
      i1=1
      i2=kxm1
      j1=1
      j2=kym1

      do 20 i=i1,i2
         do 20 j=j1,j2
           u(i,j)=psi(i,j)
20       continue

      do 30 m1=1,kfymax

      ky1=m1
      call kflowp1(u,ky1,ekylocp,ekynonp,enyloc,enynon)

      ekyflwp=ekylocp+ekynonp
      enyflw=enyloc+enynon

      flowpy1(m1)=flowpy1(m1)+ekylocp
      enlflwy(m1)=enlflwy(m1)+enyloc
      flowpyn(m1)=flowpyn(m1)+ekynonp
      ennflwy(m1)=ennflwy(m1)+enynon
      flowpy(m1)=flowpy(m1)+ekyflwp
      enflowy(m1)=enflowy(m1)+enyflw

30       continue

      do 40 m=1,kfxmax

      kx1=m

      call kflowp2(u,kx1,ekxlocp,ekxnonp,enxloc,enxnon)

      ekxflwp=ekxlocp+ekxnonp
      enxflw=enxloc+enxnon

      flowpx1(m)=flowpx1(m)+ekxlocp
      enlflwx(m)=enlflwx(m)+enxloc
      flowpxn(m)=flowpxn(m)+ekxnonp
      ennflwx(m)=ennflwx(m)+enxnon
      flowpx(m)=flowpx(m)+ekxflwp
```



```
      enflowx(m)=enflowx(m)+enxflw

40       continue


      return

      end


c*****************************************************

      subroutine kflowp1(u,ky1,ekylocp,ekynonp,enyloc,enynon)
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpx1(kfxmax),flowpxn(kfxmax),flowpy1(kfymax),flowpyn(kfymax),
     + flowex1(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
      complex u(kxm1,kym1),dnylocp,dnynonp,ekylocp,ekynonp,ic,
     +  enyloc,enynon
      integer ky1
      ekylocp=(0.,0.)
      ekynonp=(0.,0.)
      enyloc=(0.,0.)
      enynon=(0.,0.)
       ic=(0.,1.)
      i1=1
      i2=kxm1
      j1=1
      j2=kym1
10      do 20 i=i1,i2
          kxx=i-kxm2
          do 20 j=j1,j2
          kyy=j-kym2
```

```
          ak=(kyy*kynorm)**2+(kxx*kxnorm)**2

          if(iabs(kyy) .le. kyl)then

          call yflowp(u,u,kyl,kxx,kyy,dnylocp,dnynonp)

          ekylocp=ekylocp-lp*d*.5*dnylocp*conjg(u(i,j))
     + /(1.+ak*rhos**2)
          eknonp=eknonp-lp*d*.5*dnynonp*conjg(u(i,j))
     + /(1.+ak*rhos**2)
          enyloc=enyloc-lp*d*.5*ak*dnylocp*conjg(u(i,j))
     + /(1.+ak*rhos**2)
          enynon=enynon-lp*d*.5*ak*dnynonp*conjg(u(i,j))
     + /(1.+ak*rhos**2)

          endif

  20    continue

c       if (kxx .eq. kxmax .and. kyy .eq. kymax) goto 30
c       i1=kxm2
c       i2=kxm1
c       j1=2
c       goto 10

  30    return

        end


c*****************************************************
        subroutine kflowp2(u,kxl,ekxlocp,ekxnonp,enxloc,enxnon)
        parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
        complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
        real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
        common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
        common /arrays/
     + psi,den
```



```
        common /arrdim/
     + nt,ntstep,pesu
        complex u(kxm1,kym1),dnxlocp,dnxnonp,ekxlocp,ekxnonp,ic,
     +  enxloc,enxnon
        integer kxl
        ekxlocp=(0.,0.)
        ekxnonp=(0.,0.)
        enxloc=(0.,0.)
        enxnon=(0.,0.)
        ic=(0.,1.)
        i1=1
        i2=kxmax
        j1=1
        j2=kym2
  60      do 70 i=i1,i2
            kxx=i-kxm2
            do 70 j=j1,j2
            kyy=j-1

            ak=(kyy*kynorm)**2+(kxx*kxnorm)**2

            if(iabs(kxx) .le. kxl)then

            call xflowp(u,u,kxl,kxx,kyy,dnxlocp,dnxnonp)

            ekxlocp=ekxlocp-lp*d*.5*dnxlocp*conjg(u(i,j+kymax))
     + /(1.+ak*rhos**2)
            eknonp=eknonp-lp*d*.5*dnxnonp*conjg(u(i,j+kymax))
     + /(1.+ak*rhos**2)
            enxloc=enxloc-lp*d*.5*ak*dnxlocp*conjg(u(i,j+kymax))
     + /(1.+ak*rhos**2)
            enxnon=enxnon-lp*d*.5*ak*dnxnonp*conjg(u(i,j+kymax))
     + /(1.+ak*rhos**2)

            endif

  70     continue
        if (kxx .eq. kxmax .and. kyy .eq. kymax) goto 80
        i1=kxm2
        i2=kxm1
        j1=2
        goto 60

  80    return

        end


c*****************************************************
        subroutine yflowp(u,v,kyl,kxx,kyy,wylocp,wynonp)
        parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
        complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
```

```
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
       real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
       common /parms/
     + vl,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
       common /arrays/
     + psi,den
       common /arrdim/
     + nt,ntstep,pesu

       complex u(kxm1,kym1),v(kxm1,kym1),wylocp,wynonp,
     +  wyllocp,wylnonp
        integer kyl
c intialization
c process

       wylocp=(0.0,0.0)
       wynonp=(0.0,0.0)
       i1=1
       i2=kxm1+kxx
       if (kxx .gt.  0) then
       i1=kxx+1
       i2=kxm1
       end if

       j1=1
       j2=kym1+kyy
       if(kyy .gt. 0)then
       j1=kyy+1
       j2=kym1
       endif

            do 20 i=i1,i2
            do 20 j=j1,j2
              kx1=i-kxm2
              ky1=j-kym2
              kxmod=kxx-kx1
              kymod=kyy-ky1
              kxm=kxm2+kxmod
              kym=kym2+kymod

              ak1=kx1**2+ky1**2
              ak=kxmod**2+kymod**2

       if(((iabs(ky1) .gt. ky1) .or. (iabs(kyy-ky1) .gt. ky1)).and.
```



```
     +  ((iabs(ky1-kyl).le.loc).or.(iabs((kyy-ky1)-kyl).le.loc)))then

c convolute

          wyllocp=(u(i,j)*v(kxm,kym)*ak1-u(kxm,kym)*v(i,j)*
     +    ak)*kxnorm*kynorm
          wyllocp=wyllocp*kxnorm*kynorm*(ky1*kxx-kx1*kyy)*.5
          wylocp=wylocp+wyllocp

       elseif(((iabs(ky1) .gt. ky1) .or. (iabs(kyy-ky1) .gt. ky1)).and.
     +  ((iabs(ky1-ky1).gt.loc).or.(iabs((kyy-ky1)-kyl).gt.loc)))then
          wylnonp=(u(i,j)*v(kxm,kym)*ak1-u(kxm,kym)*v(i,j)*
     +    ak)*kxnorm*kynorm
          wylnonp=wylnonp*kxnorm*kynorm*(ky1*kxx-kx1*kyy)*.5
          wynonp=wynonp+wylnonp
       endif

  20        continue

       return

       end


c*************************************************
       subroutine xflowp(u,v,kxl,kxx,kyy,wxlocp,wxnonp)
       parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
       complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
       real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
       common /parms/
     + vl,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
       common /arrays/
     + psi,den
       common /arrdim/
     + nt,ntstep,pesu

       complex u(kxm1,kym1),v(kxm1,kym1),wxlocp,wxnonp,
     +  wxllocp,wxlnonp
        integer kxl
```

```
c intialization
c process

      wxlocp=(0.0,0.0)
      wxnonp=(0.0,0.0)
      i1=1
      i2=kxm1+kxx
      if (kxx .gt.  0) then
      i1=kxx+1
      i2=kxm1
      end if
           do 20 i=i1,i2
           do 20 j=kyy+1,kym1
             kx1=i-kxm2
             ky1=j-kym2
             kxmod=kxx-kx1
             kymod=kyy-ky1
             kxm=kxm2+kxmod
             kym=kym2+kymod

             ak1=kx1**2+ky1**2
             ak=kxmod**2+kymod**2

     if(((iabs(kx1) .gt. kx1) .or. (iabs(kxx-kx1) .gt. kx1)).and.
    +  ((iabs(kx1-kx1).le.loc).or.(iabs((kxx-kx1)-kx1).le.loc)))then

c convolute

          wx1locp=(u(i,j)*v(kxm,kym)*ak1-u(kxm,kym)*v(i,j)*
    +     ak)*kxnorm*kynorm
          wx1locp=wx1locp*kxnorm*kynorm*(ky1*kxx-kx1*kyy)*.5
          wxlocp=wxlocp+wx1locp

     elseif(((iabs(kx1) .gt. kx1) .or. (iabs(kxx-kx1) .gt. kx1)).and.
    +  ((iabs(kx1-kx1).gt.loc).or.(iabs((kxx-kx1)-kx1).gt.loc)))then

          wx1nonp=(u(i,j)*v(kxm,kym)*ak1-u(kxm,kym)*v(i,j)*
    +     ak)*kxnorm*kynorm
          wx1nonp=wx1nonp*kxnorm*kynorm*(ky1*kxx-kx1*kyy)*.5
          wxnonp=wxnonp+wx1nonp

     endif
 20        continue

      return

      end



c*************************************************
      subroutine writflwp
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
```



```
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
       real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
       common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
       common /arrays/
     + psi,den
       common /arrdim/
     + nt,ntstep,pesu
       real flowprx(kfxmax),flowpix(kfxmax),flowpry(kfymax),
     + flowpiy(kfymax),flwprxl(kfxmax),flwprxn(kfxmax),flwpixl(kfxmax),
     + flwpixn(kfxmax),flwpryl(kfymax),flwpryn(kfymax),
     + flwpiyl(kfymax),flwpiyn(kfymax)

       do 30 m=1,kfxmax
        flowprx(m)=real(flowpx(m))/real(nwrs)
        flowpix(m)=real(enflowx(m))/real(nwrs)
        flwprxl(m)=real(flowpxl(m))/real(nwrs)
        flwprxn(m)=real(flowpxn(m))/real(nwrs)
        flwpixl(m)=real(enlflwx(m))/real(nwrs)
        flwpixn(m)=real(ennflwx(m))/real(nwrs)
 30     continue


       do 50 m1=1,kfymax
        flowpry(m1)=real(flowpy(m1))/real(nwrs)
        flowpiy(m1)=real(enflowy(m1))/real(nwrs)
        flwpryl(m1)=real(flowpyl(m1))/real(nwrs)
        flwpryn(m1)=real(flowpyn(m1))/real(nwrs)
        flwpiyl(m1)=real(enlflwy(m1))/real(nwrs)
        flwpiyn(m1)=real(ennflwy(m1))/real(nwrs)
 50     continue

       write(15,*)(flowprx(m),m=1,kfxmax),(flowpry(m),m=1,kfymax),
     +  (flwprxl(m),m=1,kfxmax),(flwpryl(m),m=1,kfymax),
     +  (flwprxn(m),m=1,kfxmax),(flwpryn(m),m=1,kfymax)
       write(16,*)(flowpix(m),m=1,kfymax),(flowpiy(m),m=1,kfymax),
     +  (flwpixl(m),m=1,kfxmax),(flwpiyl(m),m=1,kfymax),
     +  (flwpixn(m),m=1,kfxmax),(flwpiyn(m),m=1,kfymax)


      do 20 j=1,kfxmax
      do 20 j1=1,kfymax
```

```
      flowpx(j)=(0.,0.)
      flowpy(j1)=(0.,0.)
      flowpxl(j)=(0.,0.)
      flowpxn(j)=(0.,0.)
      flowpyl(j1)=(0.,0.)
      flowpyn(j1)=(0.,0.)

      enflowx(j)=(0.,0.)
      enlflwx(j)=(0.,0.)
      ennflwx(j)=(0.,0.)
      enflowy(j1)=(0.,0.)
      enlflwy(j1)=(0.,0.)
      ennflwy(j1)=(0.,0.)

 20   continue

 200   format(50(e12.4,1x))
      return

      end



c*******************************************************
      subroutine pert
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx,pulhgt,kxpul,kypul,
     + ratio2,kbreak
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu

      external ranl
c   initialization
```



```
c   perturb psi
      a1=0.
      a2=0. .
      i1=1
      i2=kxm2
      j1=kym2
      j2=kym1
      iflag=-10
c    for flat initial profile
      if (init .eq. 1)then
 10    do 20 i=i1,i2
          xk=kxnorm*(i-kxm2)
        do 20 j=j1,j2
         yk=kynorm*(j-kym2)
         ak=sqrt(xk**2+yk**2)
         if((iabs(i-kxm2) .eq. 3 .and. iabs(j-kym2) .le. 3) .or.
     +    (iabs(j-kym2) .eq. 3 .and. iabs(i-kxm2) .le. 3))then
         psi(i,j)=cmplx(1000.*flreal,1000.*flima)
         den(i,j)=cmplx(1000.*flreal,1000.*flima)
         else
         psi(i,j)=cmplx(flreal,flima)
         den(i,j)=cmplx(flreal,flima)
         endif
         if (i .eq. kxm2 .and. j .eq. kym2) goto 20
         psi(2*kxm2-i,2*kym2-j)=conjg(psi(i,j))
         den(2*kxm2-i,2*kym2-j)=conjg(den(i,j))
 20    continue
      if(xk .eq.  kxmax*kxnorm  .and.  yk  .eq.  kymax*kynorm)goto 30
      i1=kxm2+1
      i2=kxm1
      j1=kym2+1
      goto 10
c    for regular peaked initial profile
      elseif(init .eq. 2) then
 11    do 21 i=i1,i2
          xk=kxnorm*(i-kxm2)
        do 21 j=j1,j2
         yk=kynorm*(j-kym2)
         ak=sqrt(xk**2+yk**2)
         a1=a1+psi0
         psi(i,j)=ratio*cmplx(a1,a1)
         den(i,j)=ratio*cmplx(a1,a1)
         if (i .eq. kxm2 .and. j .eq. kym2) goto 21
         psi(2*kxm2-i,2*kym2-j)=conjg(psi(i,j))
         den(2*kxm2-i,2*kym2-j)=conjg(den(i,j))
 21    continue
      if(xk .eq.  kxmax*kxnorm  .and.  yk  .eq.  kymax*kynorm)goto 30
      i1=kxm2+1
      i2=kxm1
      j1=kym2+1
      goto 11

c  for peaked initial profile with random phases
      elseif(init .eq. 3)then
cfpp$ cncall
 12    do 22 i=i1,i2
```

```
          xk=kxnorm*(i-kxm2)
         do 22 j=j1,j2
          yk=kynorm*(j-kym2)
          ak=sqrt(xk**2+yk**2)
          a1=psi0/(1.+ak**ratio)
          a2=a1*2.*sin(2.*3.141592654*ran1(iflag))
          a3=sqrt(abs(2.*a1**2-a2**2))
          a4=psi0/(1.+ak**ratio)
          a5=a4*2.*sin(2.*3.141592654*ran1(iflag))
          a6=sqrt(abs(2.*a4**2-a5**2))

          psi(i,j)=1.*cmplx(a3,a2)
          den(i,j)=1.*cmplx(a6,a5)
          if (i .eq. kxm2 .and. j .eq. kym2) goto 22
          psi(2*kxm2-i,2*kym2-j)=conjg(psi(i,j))
          den(2*kxm2-i,2*kym2-j)=conjg(den(i,j))
 22     continue
       if(xk .eq.  kxmax*kxnorm  .and.  yk  .eq.  kymax*kynorm)goto 30
       i1=kxm2+1
       i2=kxm1
       j1=kym2+1
       goto 12

c  for restart from file initspec
       elseif(init .eq. 4)then
       open(20,file='initspec',status='unknown')

       do 44 i=1,kxm1
         read(20,*)(psi(i,j),j=1,kym1)
         read(20,*)(den(i,j),j=1,kym1)
 44    continue


c  for restart from file initspec with pulse of height pulhgt and
c in position kypul and kxpul (the pulse is a band)

       elseif(init .eq. 5)then
       open(20,file='initspec',status='unknown')

       do 45 i=1,kxm1
         read(20,*)(psi(i,j),j=1,kym1)
         read(20,*)(den(i,j),j=1,kym1)
 45    continue

       do 46 i=1,kxm1
        do 46 j=1,kym1

       if((iabs(i-kxm2) .eq. kxpul .and. iabs(j-kym2) .le. kypul) .or.
     +     (iabs(j-kym2) .eq. kypul .and. iabs(i-kxm2) .le. kxpul))then
       psi(i,j)=psi(i,j)*pulhgt
       den(i,j)=den(i,j)*pulhgt
       else
       psi(i,j)=psi(i,j)*1
       den(i,j)=den(i,j)*1
       endif
 46    continue
```



```
c  for peaked initial profile with random phases with 2 slopes
       elseif(init .eq. 6)then
cfpp$ cncall
        b1=(real(kbreak))**(ratio2-ratio)
  52    do 62 i=i1,i2
           xk=kxnorm*(i-kxm2)
          do 62 j=j1,j2
           yk=kynorm*(j-kym2)
           ak=sqrt(xk**2+yk**2)
           if((iabs(i-kxm2) .le. kbreak) .and.
     +        (iabs(j-kym2) .le. kbreak))then
           a1=psi0/(1.+ak**ratio)
           a2=a1*2.*sin(2.*3.141592654*ran1(iflag))
           a3=sqrt(abs(2.*a1**2-a2**2))
           psi(i,j)=1.*cmplx(a3,a2)
           den(i,j)=1.*cmplx(a3,a2)
           else
           a4=b1*psi0/(1.+ak**ratio2)
           a5=a4*2.*sin(2.*3.141592654*ran1(iflag))
           a6=sqrt(abs(2.*a4**2-a5**2))
           psi(i,j)=1.*cmplx(a6,a5)
           den(i,j)=1.*cmplx(a6,a5)
           endif
           if (i .eq. kxm2 .and. j .eq. kym2) goto 62
           psi(2*kxm2-i,2*kym2-j)=conjg(psi(i,j))
           den(2*kxm2-i,2*kym2-j)=conjg(den(i,j))
  62    continue
       if(xk .eq.  kxmax*kxnorm  .and.  yk  .eq.  kymax*kynorm)goto 30
       i1=kxm2+1
       i2=kxm1
       j1=kym2+1
       goto 52
       endif


  30   return
       end


c*************************************************
       function ran1(idum)
       dimension r(97)
       parameter (m1=259200,ia1=7141,ic1=54773,rm1=3.8580247e-6)
       parameter (m2=134456,ia2=8121,ic2=28411,rm2=7.4373773e-6)
       parameter (m3=243000,ia3=4561,ic3=51349)
       data iff /0/
       if (idum.lt.0.or.iff.eq.0) then
         iff=1
         ix1=mod(ic1-idum,m1)
         ix1=mod(ia1*ix1+ic1,m1)
         ix2=mod(ix1,m2)
         ix1=mod(ia1*ix1+ic1,m1)
         ix3=mod(ix1,m3)
```

```
        do 11 j=1,97
          ix1=mod(ia1*ix1+ic1,m1)
          ix2=mod(ia2*ix2+ic2,m2)
          r(j)=(float(ix1)+float(ix2)*rm2)*rm1
11              continue
        idum=1
      endif
      ix1=mod(ia1*ix1+ic1,m1)
      ix2=mod(ia2*ix2+ic2,m2)
      ix3=mod(ia3*ix3+ic3,m3)
      j=1+(97*ix3)/m3
      if(j.gt.97.or.j.lt.1)pause
      ran1=r(j)
      r(j)=(float(ix1)+float(ix2)*rm2)*rm1
      return
       end


c*****************************************************
      subroutine psilin(dnl,u,v,kxx,kyy)
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
      complex u,ic,dnl,v
c initialization
      ic=(0.,1.)
      xk=kxnorm*kxx
      yk=kynorm*kyy
      if (iabs(kyy) .le. kyg .and. iabs(kxx) .le. kxg) then
c construct linear portion

      g1=gama1
      d1=dfirst*nue
```



```
      dnl=-1.*ic*yk*v1*u + d1*sqrt(epsi)*v-g1*u
      dnl=dnl+ic*v1*sqrt(epsi)*d*yk*u - nue*sqrt(epsi)*u
c
      elseif(iabs(kyy) .gt. kys .or. iabs(kxx) .gt. kxs)then
c construct linear portion
      g1=gama3
      d1=dout*nue
      dnl=-1.*ic*yk*v1*u - g1*u*(yk**2+xk**2)**2+d1*sqrt(epsi)*v
      dnl=dnl+ic*v1*sqrt(epsi)*d*yk*u- nue*sqrt(epsi)*u
      else

c construct linear portion
      g1=gama2
      d1=dmid*nue
      dnl=-1.*ic*yk*v1*u - g1*u*(yk**2+xk**2)**2+d1*sqrt(epsi)*v
      dnl=dnl+ic*v1*sqrt(epsi)*d*yk*u- nue*sqrt(epsi)*u
      endif

      return
      end

c*****************************************
      subroutine fpsi(kx,ky,u,v,psikp)
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
      complex u(kxm1,kym1),v(kxm1,kym1),psikp,dn,dnl,pin,din,
     + dn1,dn2,ic
      ic=(0.,1.)
      xk=kxnorm*kx
      yk=kynorm*ky
c     calculate dn/dx=psikp at kx, ky
      pin=u(kx+kxm2,ky+kym2)
```

```
      din=v(kx+kxm2,ky+kym2)
      ak1=1.+ (xk**2+yk**2)*rhos**2

      call psilin(dn1,pin,din,kx,ky)
      call conv2(u,u,kx,ky,dn2)
      psikp=dn1-lp*d*.5*dn2
      psikp=psikp/((1.-sqrt(epsi))*ak1)
c
      return
      end


c*********************************************
      subroutine fden(kx,ky,u,v,dn1)
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
      complex u(kxm1,kym1),v(kxm1,kym1),pin,din,psikp,dn1,dnl,ic

c initialization
      ic=(0.,1.)
      yk=kynorm*ky

      pin=u(kx+kxm2,ky+kym2)
      din=v(kx+kxm2,ky+kym2)

c construct linear portion

      d1=d
      dnl=(-1.*ic*yk*d1*v1*pin + nue*pin - nue*din)/emass

      call conv(u,v,kx,ky,dn1)
      dnl=dnl-le*d*.5*dnl
```



```
      return
      end


c*********************************************
      subroutine fcn1(neq,x,y,yprime)
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
      real y(nn),yprime(nn)
      complex psikp,nkp,u(kxm1,kym1),denkp,v(kxm1,kym1),
     + pin,din
c initialization
      kount=0
      kount1=kxm1*kymax+kxmax
      kountd=2*(kxm1*kymax+kxmax)
      kountd1=3*(kxm1*kymax+kxmax)
c convert y to psi,den
      call invert(y,u,v)
c get yprime
      i1=1
      i2=kxmax
      j1=1
      j2=kym2
  10    do 20 i=i1,i2
          ik=i-kxm2
          do 20 j=j1,j2
          jk=j-1
           kount=kount+1
c get dn/dt=fpsi
           pin=u(ik+kxm2,jk+kym2)
```

```
            din=v(ik+kxm2,jk+kym2)
            call fpsi(ik,jk,u,v,psikp)
            call fden(ik,jk,u,v,denkp)
            yprime(kount)=real(psikp)
            yprime(kount+kountl)=aimag(psikp)
            yprime(kount+kountd)=real(denkp)
            yprime(kount+kountdl)=aimag(denkp)

  20     continue
         if (ik .eq. kxmax .and. jk .eq. kymax) goto 30
         i1=kxm2
         i2=kxm1
         j1=2
         goto 10
c
  30     return
         end



c*****************************************************
         subroutine fcn2(neq,x,y,ml,mu,pd,nrpd)
         parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
       + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
       + kfxmax=kxmax-1,kfymax=kymax-1)
         complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
       + flowpy(kfymax),
       + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
       + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
       + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
       + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
       + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
       + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
         real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
       + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
         common /parms/
       + vl,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
       + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
       + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
       + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
       + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
       + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
       + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
       + enneflwy,enleflwy,enneflwx,enleflwx
         common /arrays/
       + psi,den
         common /arrdim/
       + nt,ntstep,pesu
          integer mm
         real x,y(nn),pd(nrpd,nn),g
         complex u(kxm1,kym1),v(kxm1,kym1)
c   the jacobian
c  initialization
         kount=0
         kountm=kxm1*kymax+kxmax
```



```
         kountd=2*(kxm1*kymax+kxmax)
         kountdm=3*(kxm1*kymax+kxmax)
         lount=0
         lountm=kxm1*kymax+kxmax
         lountd=2*(kxm1*kymax+kxmax)
         lountdm=3*(kxm1*kymax+kxmax)
         ab=1./(1.-sqrt(epsi))
         call invert(y,u,v)
c  process
         m1=1
         m2=kxmax
         n1=1
         n2=kym2

         i1=1
         i2=kxmax
         j1=1
         j2=kym2
  55       do 60 m=m1,m2
              kx=(m-kxm2)*kxnorm
              do 50 n=n1,n2
               ky=(n-1)*kynorm
               kount=0
               lount=lount+1
  25           do 30 i=i1,i2
                  kx1=(i-kxm2)*kxnorm
                  do 20 j=j1,j2
                   ky1=(j-1)*kynorm
                   kount=kount+1
                   kxarr=m
                   kyarr=n+kymax
                   kx1arr=i
                   ky1arr=j+kymax
           kxxarr=kxarr-kx1arr
           kyyarr=kyarr-ky1arr
           ab1=ab/(1.+(kx**2+ky**2)*rhos**2)
         if (iabs(ky) .le. kyg .and. iabs(kx) .le. kxg)then
           g=gama1
           d1=dfirst*nue
         elseif (iabs(ky) .gt. kys .or. iabs(kx) .gt. kxs)then
           g=gama3*(ky**2+kx**2)**2
           d1=dout*nue

         else
            g=gama2*(ky**2+kx**2)**2
            d1=dmid*nue
         endif

          ak1=kx1**2+ky1**2
          ak=(kx-kx1)**2+(ky-ky1)**2

          if((m .eq. i) .and. (n .eq. j))then

            pd(lount,kount)=(-1.*g- nue*sqrt(epsi))*ab1
            pd(lount,kount+kountm)=(1.*vl*ky-sqrt(epsi)*d1*ky)*ab1
            pd(lount,kount+kountd)=(sqrt(epsi)*nue)*ab1
```

```
      pd(lount,kount+kountdm)=0.

      pd(lount+lountd,kount)=(-1.*v1*ky+sqrt(epsi)*d1*ky)*ab1
      pd(lount+lountd,kount+kountm)=(-1.*g- nue*sqrt(epsi))*ab1
      pd(lount+lountd,kount+kountd)=0.
      pd(lount+lountd,kount+kountdm)=(sqrt(epsi)*nue)*ab1


      pd(lount+lountm,kount)= 1.*nue
      pd(lount+lountm,kount+kountm)=1.*v1*d*ky/emass
      pd(lount+lountm,kount+kountd)=-1.*nue/emass
      pd(lount+lountm,kount+kountdm)=0.

      pd(lount+lountdm,kount)=-1.*v1*d*ky/emass
      pd(lount+lountdm,kount+kountm)= 1.*nue
      pd(lount+lountdm,kount+kountd)=0.
      pd(lount+lountdm,kount+kountdm)=-1.*nue/emass


      else

      if((kxxarr .le. 0) .or. (kxxarr .gt. kxm1) .or.
     +   (kyyarr .le. 0) .or. (kyyarr .gt. kym1))then

      pd(lount,kount)=0.
      pd(lount,kount+kountm)=0.
      pd(lount,kount+kountd)=0.
      pd(lount,kount+kountdm)=0.

      pd(lount+lountm,kount)=0.
      pd(lount+lountm,kount+kountm)=0.
      pd(lount+lountm,kount+kountd)=0.
      pd(lount+lountm,kount+kountdm)=0.

      pd(lount+lountd,kount)=0.
      pd(lount+lountd,kount+kountm)=0.
      pd(lount+lountd,kount+kountd)=0.
      pd(lount+lountd,kount+kountdm)=0.

      pd(lount+lountdm,kount)=0.
      pd(lount+lountdm,kount+kountm)=0.
      pd(lount+lountdm,kount+kountd)=0.
      pd(lount+lountdm,kount+kountdm)=0.

      else

      pd(lount,kount)=(
     ++lp*d*.5*(kx1*ky-ky1*kx)*real(u(kxxarr,kyyarr))
     +  *(ak1-ak))*ab1
      pd(lount,kount+kountm)=(
     ++lp*d*.5*(kx1*ky-ky1*kx)*aimag(u(kxxarr,kyyarr))
     +  *(ak1-ak))*ab1
      pd(lount,kount+kountd)=0.
      pd(lount,kount+kountdm)=0.
```



```
      pd(lount+lountm,kount)=(
     ++lp*d*.5*(kx1*ky-ky1*kx)*aimag(u(kxxarr,kyyarr))
     +  *(ak1-ak))*ab1
      pd(lount+lountm,kount+kountm)=(
     ++lp*d*.5*(kx1*ky-ky1*kx)*real(u(kxxarr,kyyarr))
     +  *(ak1-ak))*ab1
      pd(lount+lountm,kount+kountd)=0.
      pd(lount+lountm,kount+kountdm)=0.



      pd(lount+lountd,kount)=(-1.*le*d*.5*(kx1*ky-ky1*kx)
     +  *real(v(kxxarr,kyyarr))*2.)
      pd(lount+lountd,kount+kountm)=(-1.*le*d*.5*(kx1*ky-ky1*kx)
     +  *aimag(v(kxxarr,kyyarr))*2.)
      pd(lount+lountd,kount+kountd)=(-1.*le*d*.5*(kx1*ky-ky1*kx)
     +  *real(u(kxxarr,kyyarr))*2.)
      pd(lount+lountd,kount+kountdm)=(-1.*le*d*.5*(kx1*ky-ky1*kx)
     +  *aimag(u(kxxarr,kyyarr))*2.)

      pd(lount+lountdm,kount)=(-1.*le*d*.5*(kx1*ky-ky1*kx)
     +  *aimag(v(kxxarr,kyyarr))*2.)
      pd(lount+lountdm,kount+kountm)=(-1.*le*d*.5*(kx1*ky-ky1*kx)
     +  *real(v(kxxarr,kyyarr))*2.)
      pd(lount+lountdm,kount+kountd)=(-1.*le*d*.5*(kx1*ky-ky1*kx)
     +  *aimag(u(kxxarr,kyyarr))*2.)
      pd(lount+lountdm,kount+kountdm)=(-1.*le*d*.5*(kx1*ky-ky1*kx)
     +  *real(u(kxxarr,kyyarr))*2.)


      endif
      endif

 20    continue
 30    continue
      if ((kx1arr .eq. kxm1)  .and. (ky1arr .eq. kym1))goto 35
      i1=kxm2
      i2=kxm1
      j1=2
      goto 25

c
 35   if ((kxarr .eq. kxm1)  .and. (kyarr .eq. kym1))goto 70
      i1=1
      i2=kxmax
      j1=1
 50    continue
 60    continue
      m1=kxm2
      m2=kxm1
      n1=2
      goto 55

 70   return
      end
```

```
c*******************************************************
      subroutine timead
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + vl,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
      real y(nn),tf,rwork(22+9*nn+nn**2)
      integer iwork(20+nn),liw,lrw
      external fcn1,fcn2
c intialization
      liw=nn+20
      lrw=22+9*nn+nn**2
       if(t .eq. 0.0)then
        itask=1
        istate=1
        iopt=0
        mf=21
        itol=1
        tola=tol*amp
       endif

      h=dt/100

      tf=t+dt
c convert psi to y
      call convert(psi,den,y)
c timestep
      call lsode(fcn1,nn,y,t,tf,itol,tol,tola,itask,istate,iopt,rwork,
     +   lrw,iwork,liw,fcn2,mf)
c check for problems
      if(istate .le. 0) call endit(istate,rwork(11))
      write(9,60)iwork(11),iwork(12),iwork(13)
      write(9,*)'istate = ',istate
```



```
c convert y back to psi
      call invert(y,psi,den)
c
 60   format(/12h no. steps =,i4, 11h no. f-s =,i4,
     + 11h no. j-s =,i4)
      return
      end


c*******************************************************

      subroutine conv(u,v,kxx,kyy,w)
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + vl,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
      complex u(kxm1,kym1),v(kxm1,kym1),w,w1
c intialization
c process
      w=(0.0,0.0)
      i1=1
      i2=kxm1+kxx
      if (kxx .gt.  0) then
      i1=kxx+1
      i2=kxm1
      end if
           do 20 i=i1,i2
           do 20 j=kyy+1,kym1
             kx1=i-kxm2
             ky1=j-kym2
             kxmod=kxx-kx1
             kymod=kyy-ky1
             kxm=kxm2+kxmod
             kym=kym2+kymod
```

```
c convolute
                w1=(u(i,j)*v(kxm,kym)-u(kxm,kym)*v(i,j))
                w1=w1*kynorm*kxnorm*kynorm*(ky1*kxx-kx1*kyy)*.5
                w=w+w1
  20          continue
c
        return
        end


c*************************************************

        subroutine conv2(u,v,kxx,kyy,w)
        parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
      + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
      + kfxmax=kxmax-1,kfymax=kymax-1)
        complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
      + flowpy(kfymax),
      + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
      + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
      + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
      + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
      + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
      + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
        real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
      + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
        common /parms/
      + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
      + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
      + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
      + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
      + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
      + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
      + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
      + enneflwy,enleflwy,enneflwx,enleflwx
        common /arrays/
      + psi,den
        common /arrdim/
      + nt,ntstep,pesu
        complex u(kxm1,kym1),v(kxm1,kym1),w,w1
c intialization
c process
        w=(0.0,0.0)
        i1=1
        i2=kxm1+kxx
        if (kxx .gt.  0) then
        i1=kxx+1
        i2=kxm1
        end if
             do 20 i=i1,i2
             do 20 j=kyy+1,kym1
               kx1=i-kxm2
               ky1=j-kym2
               kxmod=kxx-kx1
               kymod=kyy-ky1
```



```
               kxm=kxm2+kxmod
               kym=kym2+kymod

               ak1=kx1**2+ky1**2
               ak=kxmod**2+kymod**2

c convolute
               w1=(u(i,j)*v(kxm,kym)*ak1-u(kxm,kym)*v(i,j)*
     +         ak)*kxnorm*kynorm
               w1=w1*kxnorm*kynorm*(ky1*kxx-kx1*kyy)*.5
               w=w+w1
  20          continue
c
        return
        end


c*************************************************
        subroutine invert(y,u,v)
        parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
      + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
      + kfxmax=kxmax-1,kfymax=kymax-1)
        complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
      + flowpy(kfymax),
      + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
      + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
      + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
      + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
      + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
      + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
        real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
      + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
        common /parms/
      + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
      + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
      + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
      + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
      + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
      + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
      + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
      + enneflwy,enleflwy,enneflwx,enleflwx
        common /arrays/
      + psi,den
        common /arrdim/
      + nt,ntstep,pesu
        complex u(kxm1,kym1),u1,v(kxm1,kym1),v2
        real y(nn)
c  initialization
        kount=0
        kountm=kxm1*kymax+kxmax
        kountd=2*(kxm1*kymax+kxmax)
        kountdm=3*(kxm1*kymax+kxmax)
c  process
        i1=1
        i2=kxmax
```

```
      j1=1
      j2=kym2
 10     do 20 i=i1,i2
          kxarr=i
          do 20 j=j1,j2
          kyarr=kymax+j
          kount=kount+1
          u1=cmplx(y(kount),y(kount+kountm))
          u(kxarr,kyarr)=u1
          v2=cmplx(y(kount+kountd),y(kount+kountdm))
          v(kxarr,kyarr)=v2

          ix=kxm1-i+1
          jx=kym2-j+1
          u(ix,jx)=conjg(u1)
          v(ix,jx)=conjg(v2)
 20     continue
      if (kxarr .eq. kxm1  .and. kyarr .eq. kym1)goto 30
      i1=kxm2
      i2=kxm1
      j1=2
      goto 10
c
 30    return
      end


c*************************************************
      subroutine convert(u,v,y)
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
      common /arrays/
     + psi,den
      common /arrdim/
     + nt,ntstep,pesu
      complex u(kxm1,kym1),u1,v(kxm1,kym1),v2
```



```
      real y(nn)
c  initialization
      kount=0
      kountm=kxm1*kymax+kxmax
      kountd=2*(kxm1*kymax+kxmax)
      kountdm=3*(kxm1*kymax+kxmax)
c  process
      i1=1
      i2=kxmax
      j1=1
      j2=kym2
 10     do 20 i=i1,i2
          kxarr=i
          do 20 j=j1,j2
          kyarr=kymax+j
          kount=kount+1
          y(kount)=real(u(kxarr,kyarr))
          y(kount+kountm)=aimag(u(kxarr,kyarr))
          y(kount+kountd)=real(v(kxarr,kyarr))
          y(kount+kountdm)=aimag(v(kxarr,kyarr))


 20     continue
      if(kxarr .eq. kxm1 .and. kyarr .eq. kym1) goto 30
      i1=kxm2
      i2=kxm1
      j1=2
      goto 10
c
 30   return
      end


c*************************************************
      subroutine final
      parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
      complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
      real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
      common /parms/
     + v1,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
```

```
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
       common /arrays/
     + psi,den
       common /arrdim/
     + nt,ntstep,pesu
       write(6,100)
       write(10,100)
  100   format(4x,' das ende')
c
       return
       end



c**********************************************
       subroutine plot(time,plot1,plot2,plot3)
       parameter(kxmax=10,kymax=10,kxm1=2*kxmax+1,kym1=2*kymax+1,
     + kxm2=kxmax+1,kym2=kymax+1,nn=4*(kxm1*kymax+kxmax),
     + kfxmax=kxmax-1,kfymax=kymax-1)
       complex psi(kxm1,kym1),den(kxm1,kym1),flowpx(kfxmax),
     + flowpy(kfymax),
     + flowpxl(kfxmax),flowpxn(kfxmax),flowpyl(kfymax),flowpyn(kfymax),
     + flowexl(kfxmax),flowexn(kfxmax),floweyl(kfymax),floweyn(kfymax),
     + flowex(kfxmax),flowey(kfymax),enflowx(kfxmax),enflowy(kfymax),
     + ennflwy(kfymax),enlflwy(kfymax),ennflwx(kfxmax),enlflwx(kfxmax),
     + eneflowx(kfxmax),eneflowy(kfymax),enneflwy(kfymax),
     + enleflwy(kfymax),enneflwx(kfxmax),enleflwx(kfxmax)
       real kxnorm,kynorm,ekkold(kxm1,kym1),le,lp,leout,lemid,lpout,
     + lpmid,ratio2,edenold(kxm1,kym1),nue,emass,dfirst,rhos
       common /parms/
     + vl,gama1,gama2,epsi,le,lp,kxg,kyg,kxnorm,kynorm,kxs,kys,
     + nwrt,nwrs,t,dt,tol,ek,gamk,flux,ncl,ratio,psi0,d,dmid,dout,loc,
     + init,flreal,flima,ekkold,emkold,flexold,flowex,flowey,gama3,
     + flowexl,flowexn,floweyl,floweyn,lpmid,lpout,lemid,leout,phase,
     + flpxold,flowpx,flowpy,flowpxl,flowpxn,flowpyl,edenold,nue,
     + amp,flowpyn,en,gamen,ed,et,ekold,enold,enflowx,enflowy,emass,
     + ennflwy,enlflwy,ennflwx,enlflwx,eneflowx,eneflowy,dfirst,rhos,
     + enneflwy,enleflwy,enneflwx,enleflwx
       common /arrays/
     + psi,den
       common /arrdim/
     + nt,ntstep,pesu
       real time(4000),plot1(4000),plot2(4000),
     +  plot3(4000),ploty(4000,3)
c
       do 30 i=1,ntstep
        ploty(i,1)=plot1(i)
        ploty(i,2)=plot2(i)
        ploty(i,3)=plot3(i)

 30     continue
c  replacing the old vax plotting progam with nmfecc's quikdraw routines
c       call opngks
c       call ezmxy(time,ploty,4000,3,4000,
c     + 'energy vs time$')
```