Two-Species Plasma Fluid Tests with 2-D Fourier Trasform Analysis

Richard Datwyler

Utah State University

NIMROD Team Meeting, Logan, Utah August 2011

Outline

Introduction

Few Highlights of current work

- A two species fluid model is developed and implemented using NIMROD's finite difference scheme. Particular note is made to:
    - Single advance of one array with all variables
    - Crank Nicholson advance utilized
    - Two fluid species without quasi neutrality assumption and use of displacement current
    - 2-dimensional Fourier analysis detailing dispersion relationships

Two Species Fluid Equations

As a standard fluid equation analysis, take the velocity moments to solve for the first three fluid equations

$$\frac{\partial n_s}{\partial t} + \nabla \cdot n_s \mathbf{u}_s = \frac{\delta n_s}{\delta t}$$

$$n_s m_s [\frac{\partial \mathbf{u}_s}{\partial t} + (\mathbf{u}_s \cdot \nabla)\mathbf{u}_s] + \nabla p_s + \nabla \cdot \tau_s - n_s e_s(\mathbf{E} + \mathbf{u}_s \times \mathbf{B}) = \frac{\delta \mathbf{M}_s}{\delta t}$$

$$\frac{\partial \frac{3}{2} p_s}{\partial t} + (\mathbf{u}_s \cdot \nabla)\frac{3}{2} p_s + \frac{5}{2} p_s (\nabla \cdot \mathbf{u}_s) + \nabla \cdot \mathbf{q}_s + \tau_s : \nabla \mathbf{u}_s = \frac{\delta \mathbf{E}_s}{\delta t}$$

Flow Modifications

Make some modification to these equations by:

- adding a diffusivity term to the continuity equation
- similarly a dissipative term in the form of kinematic viscosity is added to the flow advances
- in addition the flow equations have a collisional, resistive term.
- the adiabatic equation of state

$$\frac{\partial \mathbf{u}_s}{\partial t} + (\mathbf{u}_s \cdot \nabla)\mathbf{u}_s + \frac{\nabla(n_s T_s)}{n_s m_s} - \frac{e_s}{m_s}(\mathbf{E} + \mathbf{u}_s \times \mathbf{B}) + C_{visc}\nabla^2 \mathbf{u}_s = \frac{\eta n_{e_o} e^2 n_e}{n_s m_s}(\mathbf{u}_t - \mathbf{u}_s)$$

$$\frac{\partial n_s}{\partial t} + \nabla \cdot n_s \mathbf{u}_s + D_{diff}\nabla^2 n_s = 0$$

$$\frac{\partial T_s}{\partial t} + (\mathbf{u}_s \cdot \nabla) T_s + \frac{2}{3} T_s (\nabla \cdot \mathbf{u}_s) = 0$$

Field Equations

With these fluid equation we now add the field equations:

$$\frac{\partial \mathbf{E}}{\partial t} - c^2(\nabla \times \mathbf{B}) = -\sum_s \frac{e_s n_s \mathbf{u}_s}{\varepsilon_0}$$

$$\nabla \cdot \mathbf{E} = \sum_s \frac{e_s n_s}{\varepsilon_0}$$

$$\frac{\partial \mathbf{B}}{\partial t} + (\nabla \times \mathbf{E}) = 0$$

$$\nabla \cdot \mathbf{B} = 0$$

Field Modifications

To produce a perfectly hyperbolic form of Ampere's law and Faraday's law we add a corrective potential. This adds a dispersion correction and works to enforce Gauss's law and div B terms as so:

$$\frac{\partial \mathbf{E}}{\partial t} - c^2(\nabla \times \mathbf{B}) + \zeta c^2 \nabla \phi = -\sum_s \frac{e_s n_s \mathbf{u}_s}{\varepsilon_0}$$

$$\frac{\partial \phi}{\partial t} + \zeta(\nabla \cdot \mathbf{E}) = \zeta \sum_s \frac{e_s n_s}{\varepsilon_0}$$

$$\frac{\partial \mathbf{B}}{\partial t} + (\nabla \times \mathbf{E}) + \xi \nabla \varphi = 0$$

$$\frac{\partial \varphi}{\partial t} + \xi c^2 \nabla \cdot \mathbf{B} = 0$$

Collecting all of these equations, both fluid and field, results in a system of 18 coupled equations.

Implicit vs. Explicit Advances

Before considering the implementation of these equations into the NIMROD code structure, let us discuss two other topics:

- De-dimensionalization and implicit vs. explicit formulation.
- Due to the size and nature of this 18 dimensional array there needed to be the ability to test the methods of advancing the equations. To do this a centering coefficient was added to the equations to allow either implicit or explicit advances to be possible.
- The trick comes when two or more terms are explicitly multiplied together.

Continuity Example

One such term is, $\nabla \cdot (n_s \mathbf{u}_s)$.

Using a centering coefficient this term appears as the following in the continuity equation

$$\theta_c[\nabla \cdot (n_s^{t+1} \mathbf{u}_s^{t+1})] = -(1-\theta_c)\nabla \cdot (n_s^t \mathbf{u}_s^t)$$

Defining $\Delta A = A^{t+1} - A^t$, and then substituting in $A^{t+1} = \Delta A + A^t$ and foiling properly gives:

$$\theta_c[\nabla \cdot (\Delta n_s \Delta \mathbf{u}_s) + \nabla \cdot (\Delta n_s \mathbf{u}_s^t) + \nabla \cdot (n_s^t \Delta \mathbf{u}) + \nabla \cdot (n_s^t \mathbf{u}_s^t)] = -(1-\theta_c)\nabla \cdot (n_s^t \mathbf{u}_s^t)$$

Following a similar approach done in NIMROD by C. Sovinec take only Newton-like steps (the perturbed variable times the previous time step variable remains while the higher order dual perturbed variables are truncated) and canceling like terms gives,

$$\theta_c[\nabla \cdot (\Delta n_s \mathbf{u}_s^t) + \nabla \cdot (n_s^t \Delta \mathbf{u})] = -\nabla \cdot (n_s^t \mathbf{u}_s^t)$$

Gradient of Pressure Example

Similarly a term with 3 variables, $\frac{\nabla n_s T_s}{n_s}$. First use the product rule to simplify this down to a gradient of single terms, $\frac{\nabla n_s T_s}{n_s} = \nabla T_s + \frac{T_s \nabla n_s}{n_s}$.

$$\theta_c(\frac{T_s^{t+1} \nabla n_s^{t+1}}{n_s^{t+1}}) = -(1-\theta_c)\frac{T_s^t \nabla n_s^t}{n_s^t}$$

and also using $A^{t+1} = \Delta A + A^t$ as before gives

$$\theta_c[\frac{(\Delta T_s + T_s^t)\nabla(\Delta n_s + n_s^t)}{(\Delta n_s + n_s^t)}] = -(1-\theta_c)\frac{T_s^t \nabla n_s^t}{n_s^t}$$

Using a binomial expansion for the denominator and factoring the numerator leads to

$$\theta_c \frac{1}{n_s^t}(1 - \frac{\Delta n_s}{n_s^t})[\Delta T_s \nabla(\Delta n_s) + T_s^t \nabla(\Delta n_s) + \Delta T_s \nabla n_s^t + T_s^t \nabla n_s^t] = -(1-\theta_c)\frac{T_s^t \nabla n_s^t}{n_s^t}$$

$$\theta_c[\frac{T_s^t \nabla(\Delta n_s)}{n_s^t} + \frac{\Delta T_s \nabla n_s^t}{n_s^t} - \frac{(T_s^t \nabla n_s^t)\Delta n_s}{(n_s^t)^2}] = -\frac{T_s^t \nabla n_s^t}{n_s^t}$$

Implicit and Explicit Flow Equations

Similar reductions for the terms $q_s n_s \mathbf{u}_s$, $\mathbf{u}_s \cdot \nabla \mathbf{u}_s$, $\mathbf{u}_s \times \mathbf{B}$, $T_s \nabla \cdot \mathbf{u}_s$, and $\mathbf{u}_s \cdot \nabla T_s$ give:

$$\frac{\partial \Delta \mathbf{E}}{\partial t} - \theta_c [c^2 (\nabla \times \Delta \mathbf{B}) - \zeta c^2 \nabla \Delta \phi - \sum_s \frac{e_s (\Delta n_s \mathbf{u}_s^t + n_s^t \Delta \mathbf{u}_s)}{\varepsilon_0}]$$

$$= +c^2 (\nabla \times \mathbf{B}^t) - \zeta c^2 \nabla \phi^t - \sum_s \frac{e_s n_s^t \mathbf{u}_s^t}{\varepsilon_0}$$

$$\frac{\partial \Delta \phi}{\partial t} + \theta_c [\zeta (\nabla \cdot \Delta \mathbf{E}) - \zeta \sum_s \frac{e_s \Delta n_s}{\varepsilon_0}] = -\zeta (\nabla \cdot \mathbf{E}^t) + \zeta \sum_s \frac{e_s n_s^t}{\varepsilon_0}$$

$$\frac{\partial \Delta \mathbf{B}}{\partial t} + \theta_c [(\nabla \times \Delta \mathbf{E}) + \xi \nabla \Delta \varphi] = -(\nabla \times \mathbf{E}^t) - \xi \nabla \varphi^t$$

$$\frac{\partial \Delta \varphi}{\partial t} + \theta_c \xi c^2 \nabla \cdot \Delta \mathbf{B} = -\xi c^2 \nabla \cdot \mathbf{B}^t$$

Implicit and Explicit Flow Equations

and:

$$\frac{\partial \Delta \mathbf{u}_s}{\partial t} + \theta_c[(\mathbf{u}_s^t \cdot \nabla)\Delta \mathbf{u}_s + (\Delta \mathbf{u}_s \cdot \nabla)\mathbf{u}_s^t - \frac{e_s}{m_s}(\Delta \mathbf{E} + \mathbf{u}_s^t \times \Delta \mathbf{B} + \Delta \mathbf{u}_s \times \mathbf{B}^t)$$
$$+ \frac{\nabla \Delta T_s}{m_s} + \frac{T_s^t \nabla \Delta n_s}{m_s n_s^t} + \frac{\Delta T_s \nabla n_s^t}{m_s n_s^t} - \frac{(T_s^t \nabla n_s^t)\Delta n_s}{m_s (n_s^t)^2}]$$
$$= -(\mathbf{u}_s^t \cdot \nabla)\mathbf{u}_s^t - \frac{\nabla T_s^t}{m_s} - \frac{T_s^t \nabla n_s^t}{m_s n_s^t} + \frac{e_s}{m_s}(\mathbf{E}^t + \mathbf{u}_s^t \times \mathbf{B}^t)$$

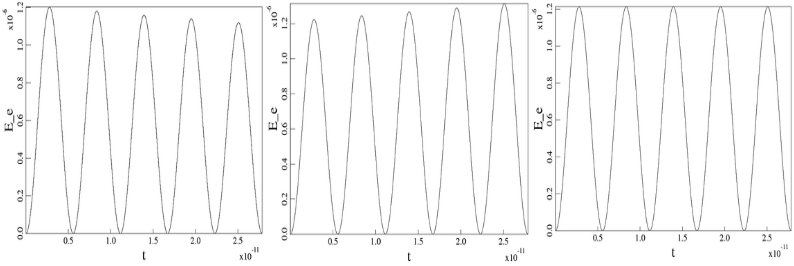
$$\frac{\partial \Delta n_s}{\partial t} + \theta_c \nabla \cdot (n_s^t \Delta \mathbf{u}_s + \Delta n_s \mathbf{u}_s^t) = -\nabla \cdot n_s^t \mathbf{u}_s^t$$

$$\frac{\partial \Delta T_s}{\partial t} + \theta_c[(\mathbf{u}_s^t \cdot \nabla)\Delta T_s + (\Delta \mathbf{u}_s \cdot \nabla) T_s^t + \frac{2}{3} T_s^t (\nabla \cdot \Delta \mathbf{u}_s) + \frac{2}{3}\Delta T_s (\nabla \cdot \mathbf{u}_s^t)]$$
$$= -(\mathbf{u}_s^t \cdot \nabla) T_s^t - \frac{2}{3} T_s^t (\nabla \cdot \mathbf{u}_s^t)$$

Graphical Results

Comparing the different types of advances we note that the implicit advance shrinks in times, while the explicit advance grows in time.

When a Crank Nicholson approach is used the results are stable, neither growing or shrinking.



De-dimensionalization

Due to the multitude of terms in the previous set of equations, we will assume that all the equations have been written with both implicit and explicit parts from here on out.

- Now we move on to making the equations non-dimensional. This is needed to help augment the fact that these equations are advanced simultaneously and in MKS units there are great differences in magnitude.
- To do this we define:

$$\mathbf{r} \longrightarrow r_0 \mathbf{r}' \quad or \quad \nabla \longrightarrow \frac{1}{r_o} \nabla'$$

$$\mathbf{u} \longrightarrow c\mathbf{u}'$$

$$t \longrightarrow \frac{r_0}{c} t'$$

De-dimensionalization

Similarly, define dimensionless variables and their normalizing coefficients for **E**, **B**, ϕ, φ, T, and n.

$$\mathbf{E} \longrightarrow cB_0\mathbf{E}'$$

$$\mathbf{B} \longrightarrow B_0\mathbf{B}'$$

$$T_s \longrightarrow T_0 T_s'$$

$$n \longrightarrow n_0 n'$$

$$\phi \longrightarrow B_0\phi'$$

$$\varphi \longrightarrow cB_0\varphi'$$

Non-dimensional Equations

$$\frac{c^2 B_o}{r_0}\frac{\partial \mathbf{E}'}{\partial t'} - \frac{B_o}{r_0}c^2(\nabla' \times \mathbf{B}') + \frac{B_o}{r_0}\zeta c^2 \nabla' \phi' = -c n_o \sum_s \frac{e_s n_s' \mathbf{u}_s'}{\varepsilon_0}$$

$$\frac{cB_o}{r_0}\frac{\partial \phi'}{\partial t'} + \frac{cB_o}{r_0}\zeta(\nabla' \cdot \mathbf{E}') = n_o \zeta \sum_s \frac{e_s n_s'}{\varepsilon_0}$$

$$\frac{cB_o}{r_0}\frac{\partial \mathbf{B}'}{\partial t'} + \frac{cB_o}{r_0}(\nabla' \times \mathbf{E}') + \frac{cB_o}{r_0}\xi \nabla' \varphi' = 0$$

$$\frac{c^2 B_o}{r_0}\frac{\partial \varphi'}{\partial t'} + \frac{B_o}{r_0}\xi c^2 \nabla' \cdot \mathbf{B}' = 0$$

$$\frac{c^2}{r_0}\frac{\partial \mathbf{u}_s'}{\partial t'} + \frac{c^2}{r_0}(\mathbf{u}_s' \cdot \nabla')\mathbf{u}_s' + \frac{n_o T_o}{n_o r_0}\frac{\nabla'(n_s' T_s')}{n_s' m_s} - cB_0 \frac{e_s}{m_s}(\mathbf{E}' + \mathbf{u}_s' \times \mathbf{B}') + \frac{c}{r_o^2} C_{visc} \nabla'^2 \mathbf{u}_s'$$

$$= \frac{c n_o^2 e^2 C_{elecd}}{n_o c^2 \varepsilon_o m_s} n_{eo}' \frac{n_e'}{n_s'}(\mathbf{u}_t' - \mathbf{u}_s')$$

$$\frac{n_0 c}{r_0}\frac{\partial n_s'}{\partial t'} + \frac{n_0 c}{r_0}\nabla' \cdot n_s' \mathbf{u}_s' + \frac{r_o}{c n_o}\frac{n_o}{r_o^2} D_{visc} \nabla'^2 n_s' = 0$$

$$\frac{T_o c}{r_0}\frac{\partial T_s'}{\partial t'} + \frac{T_o c}{r_0}(\mathbf{u}_s' \cdot \nabla') T_s' + \frac{2}{3}\frac{T_o c}{r_0} T_s'(\nabla' \cdot \mathbf{u}_s') = 0$$

Non-dimensional Equations

$$\frac{\partial \mathbf{E}}{\partial t} - (\nabla \times \mathbf{B}) + \zeta \nabla \phi = -\sum_s \frac{r_0 n_o}{c B_0} \frac{e_s n_s \mathbf{u}_s}{\varepsilon_o}$$

$$\frac{\partial \phi}{\partial t} + \zeta (\nabla \cdot \mathbf{E}) = \zeta \sum_s \frac{r_0 n_o}{c B_0} \frac{e_s n_s}{\varepsilon_o}$$

$$\frac{\partial \mathbf{B}}{\partial t} + (\nabla \times \mathbf{E}) + \xi \nabla \varphi = 0$$

$$\frac{\partial \varphi}{\partial t} + \xi \nabla \cdot \mathbf{B} = 0$$

$$\frac{\partial \mathbf{u}_s}{\partial t} + (\mathbf{u}_s \cdot \nabla)\mathbf{u}_s + \frac{m_i}{m_s} \frac{\nabla (n_s T_s)}{n_s} - \frac{r_0 B_0}{c} \frac{e_s}{m_s} (\mathbf{E} + \mathbf{u}_s \times \mathbf{B}) + \frac{1}{c r_o} C_{visc} \nabla^2 \mathbf{u}_s = \frac{r_o}{c} \left(\frac{\omega_{pe}}{c}\right)^2 C_{elecd} n_{e_o} \frac{n_e}{n_s} (\mathbf{u}_t - \mathbf{u}_s)$$

$$\frac{\partial n_s}{\partial t} + \nabla \cdot n_s \mathbf{u}_s + \frac{1}{c r_o} D_{visc} \nabla^2 n_s = 0$$

$$\frac{\partial T_s}{\partial t} + (\mathbf{u}_s \cdot \nabla) T_s + \frac{2}{3} T_s (\nabla \cdot \mathbf{u}_s) = 0$$

NIMROD Implementation

Just to show a very brief look at some of the lines of coding, this is part of the $\mathbf{\Delta u} \cdot \nabla \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{\Delta u}$ term in the dot routine.

```
c-----------------------------------------------------------------------
c     construct V.grad(V)
c-----------------------------------------------------------------------
      CALL math_grad(nmodes,keff,3_i4,geom,ve,ver,vez,gveten,bigr)
      CALL math_grad(nmodes,keff,3_i4,geom,vee,veer,veez,gveeten,bigr)
      CALL fft_nim('inverse',ncx*ncy,mpseudo,lphi,9_i4,
     $             gveten,real_gveten,dealiase)
      CALL fft_nim('inverse',ncx*ncy,mpseudo,lphi,9_i4,
     $             gveeten,real_gveeten,dealiase)
      CALL fft_nim('inverse',ncx*ncy,mpseudo,lphi,3_i4,
     $             force_ve,real_force1,dealiase)
      CALL fft_nim('inverse',ncx*ncy,mpseudo,lphi,3_i4,
     $             force_vee,real_force2,dealiase)
        DO ip=1,nphi
          DO ix=1,mpseudo
            tmp1=real_ve(:,ix,ip)
            tmp2=real_veptr(:,ix,ip)
            real_force1(1,ix,ip)=real_force1(1,ix,ip)-
     $              tmp1(1)*real_gveptr(1,ix,ip)-
     $              tmp1(2)*real_gveptr(2,ix,ip)-
     $              tmp1(3)*real_gveptr(3,ix,ip)-
     $              tmp2(1)*real_gveten(1,ix,ip)-
     $              tmp2(2)*real_gveten(2,ix,ip)-
     $              tmp2(3)*real_gveten(3,ix,ip)
```

And from the operator routine, this shows the coupling of velocity in radial direction for the resistivity term, eta J

```
c-----------------------------------------------------------------------
c       now for extra part of ion velocities resistivity
c       associated with ja
c-----------------------------------------------------------------------
        IF (elecd>0.) THEN
          int(12,9,:,jv,iv) =SUM(
     $                    -dt*feflow*alpha(:,:,iv)*alpha(:,:,jv)*
     $                   f_resi*ne*ne/nd,1)
          int(15,9,:,jv,iv) =SUM(
     $                    dt*feflow*alpha(:,:,iv)*alpha(:,:,jv)*
     $                   f_resi*(ne*ne/(nd*nd))*
     $                                (vee(1,:,:)-ve(1,:,:)),1)
          int(16,9,:,jv,iv) =SUM(
     $                    -dt*feflow*alpha(:,:,iv)*alpha(:,:,jv)*
     $                  f_resi*2*(ne/nd)*(vee(1,:,:)-ve(1,:,:)),1)
```

NIMROD Implementation

Here are other parts for the radial component of the ion velocity in the operator routine

```
c-----------------------------------------------------------------------
c         now for ion velocities
c-----------------------------------------------------------------------
          alprs(:,:)=g*alpha(:,:,iv)/bigr(:,:)
          int(9,9,:,jv,iv)=SUM(alpha(:,:,iv)*alpha(:,:,jv)
     $             +dt*feflow*alpha(:,:,iv)*alpha(:,:,jv)*
     $              (ver(1,:,:)+ve(3,:,:)*(0,1)*keff(jmode)/bigr
     $               +f_resi*ne*ne/nd) ! resistive term
     $             +dt*feflow*alpha(:,:,iv)*
     $              (ve(1,:,:)*dalpdr(:,:,jv)+ve(2,:,:)*dalpdz(:,:,jv))
     $             +vidt*feflow*(gr_alpha(1,:,:,iv)*piten(1,1,:,:,jv)+
     $                          gr_alpha(2,:,:,iv)*piten(2,1,:,:,jv)+
     $                          gr_alpha(3,:,:,iv)*piten(3,1,:,:,jv)+
     $                          alprs*piten(9,1,:,:,jv)/bigr),1) ! visc
          int(10,10,:,jv,iv)=SUM(alpha(:,:,iv)*alpha(:,:,jv)
```

```
          IF (beta>0) THEN
            int(15,9,:,jv,iv) =SUM(dd_ve1*dt*feflow*alpha(:,:,iv)*
     $                          (dalpdr(:,:,jv)*ti/nd
     $                           -alpha(:,:,jv)*ti*ndr/(nd*nd)),1)
            int(17,9,:,jv,iv) =SUM(dd_ve1*dt*feflow*alpha(:,:,iv)*
     $                          (dalpdr(:,:,jv)
     $                           +alpha(:,:,jv)*ndr/nd),1)
```

And from the operator routine, this shows the coupling of velocity in radial direction

```
          int(10,9,:,jv,iv)=SUM(
     $                        dt*feflow*alpha(:,:,iv)*alpha(:,:,jv)*
     $                    (vez(1,:,:)
     $                     +dd_ve2*(qs(2)/ms(2))*be(3,:,:)),1)
          int(11,9,:,jv,iv)=SUM(
     $                       -dt*feflow*alpha(:,:,iv)*alpha(:,:,jv)*
     $                    (g*ve(3,:,:)/bigr
     $                     +dd_ve2*(qs(2)/ms(2))*be(2,:,:))+
     $                    (gr_alpha(3,:,:,iv)*piten(3,3,:,:,jv)+
     $                     alprs*piten(9,3,:,:,jv))*vidt*feflow,1)
          int(1,9,:,jv,iv)=SUM(
     $                       -dt*feflow*alpha(:,:,iv)*alpha(:,:,jv)*
     $                            dd_ve2*(qs(2)/ms(2)),1)
          int(6,9,:,jv,iv)=SUM(
     $                       -dt*feflow*alpha(:,:,iv)*alpha(:,:,jv)*
     $                            dd_ve2*(qs(2)/ms(2))*ve(3,:,:),1)
          int(7,9,:,jv,iv)=SUM(
     $                        dt*feflow*alpha(:,:,iv)*alpha(:,:,jv)*
     $                            dd_ve2*(qs(2)/ms(2))*ve(2,:,:),1)
```

Plasma Frequency

Plasma frequency

$$\omega_{ps} = \sqrt{\frac{n_s e^2}{\varepsilon_0 m_s}}$$

Which if you are using two species, ion and electrons, is calculated as

$$\omega_p = \sqrt{\omega_{pi}^2 + \omega_{pe}^2}$$

To give an idea of the value of the ion, electron, and total plasma frequency we will use $n_s = 1 \times 10^{20}$. (The simulation has been run between $10^{12} - 10^{30}$ with similar results in low beta cases)

$\omega_{pi} = 1.48200647 \times 10^9$ $\omega_{pe} = 8.97866371 \times 10^{10}$ $\omega_p = 8.97988672 \times 10^{10}$

Numerical results give

$$\omega_p = 8.97988644272 \times 10^{10}$$

Finite Pressure and Frequency

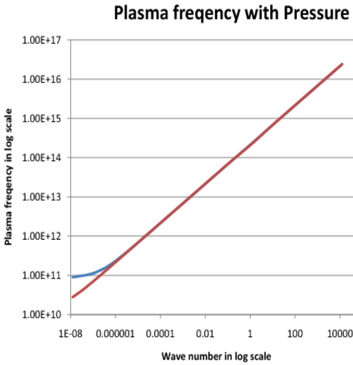
The dispersion relationship with temperature gives

$$\omega^2 = \omega_{pe}^2(1 + \gamma_e \lambda_{De}^2 k^2)$$

Which if the temperature is very low, the Debye wavelength is small and the frequency follows the plasma frequency.

Contrary if the temperature gets very large it follows the thermal velocity as

$$\frac{\omega}{k} \simeq \sqrt{\frac{\gamma_e \kappa T_e}{m_e}}$$

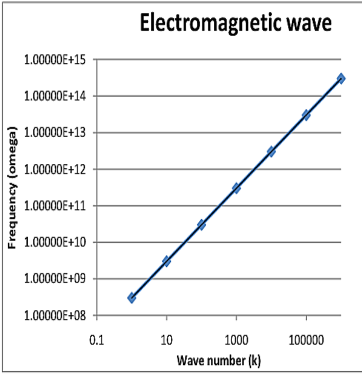


Vacuum and Frequency

Going the other way, dropping the number density and pressure to a vacuum

$$\omega = ck$$

Shown here are a spread of a few wave numbers graphed against obtained frequency, the slope being the speed of light.

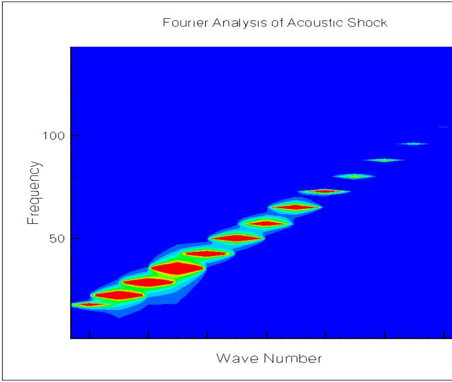


Fourier Analysis

- The trick with these simulations is to get the correct start up conditions, and then let the wave run at different wave numbers, or temperatures and then graph them. This is a bit tricky, especially with breaking down MHD terms into 2 separate flow velocities.
- As a different way to test multiple types of waves, a single shock is initiated then many waves would be produced. If a Fourier analysis is taken then the corresponding wave numbers and frequencies could be graphically generated.

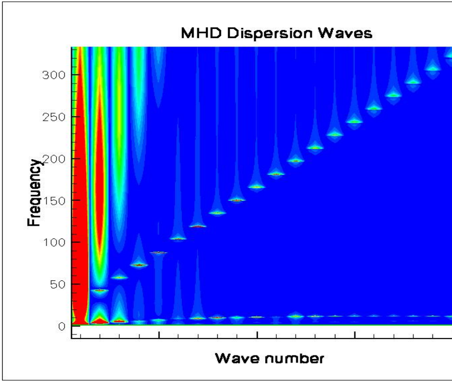
Fourier Analysis

- Previous dispersion curves were made by changing wave number, or temperature, or density to produce different frequencies.
- This Fourier method is a real 2-dimensional transfor taken from numerical recipes.
- This plot was produced on my lap top, and is a serial computation of a shock to ion and electron flow velocites in the z direction.
- It is a run of 8000 steps, and 64 grid points in the direction of the shock.



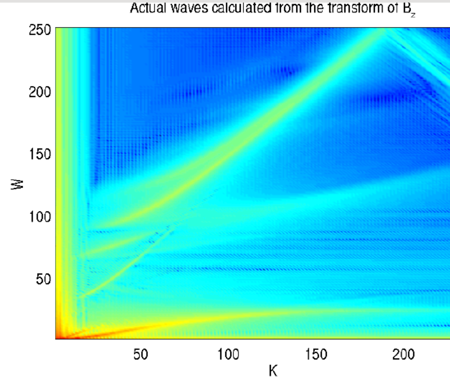
Fourier Analysis

- This plot was produced on Franklin using 8 nodes running 8000 steps and 256 grid points in the direction of the shock.
- The trick here was how to write out the complete variable, each node had its respective R block.
- To solve this rb_cel structures where used, this what node 0 pulled in all the information and the following plot was produced



Fourier Analysis

- Compare the previous to this plot by J. Loveric ‘03.



Summary

- A two fluid advance with displacement current and separate number densities and flows for each species is feasable, and produces good results for dispersion relationships with initial waves tested.
- A 2-D Fourier analysis has produced similar dispersion relations for Aucoustic waves, and shows promise for other MHD waves.
- Continued work will be to improve 2-D Fourier analysis on Franklin, to produce larger and better plots, and to initialize other MHD waves.