

This directory contains files adapted from ZEUS-3D for calculating accelerated particle spectra at a CME or interplanetary shock.

In terms of functionality, the code performs the following in sequence.

- 0) pre-task: set up a steady solar wind profile for the simulation.
- 1) When the steady solar wind is in stage, generating a CME-driven shocks by perturbing the inner boundary (located at 0.1 AU) condition. Shocks with different shock strength correspond to different factors of increase of the pressure, solar wind speed, etc. for a duration of 1 hour. parameter `i_shock_strength` in `gen_bndy.f` is used to control the shock strength.
 - `i_shock_strength = 1` ==> ultra-strong shock (increase by a factor of 6).
 - `i_shock_strength = 2` ==> strong shock (increase by a factor of 5).
 - `i_shock_strength = 3` ==> intermediate shock (increase by a factor of 4).
 - `i_shock_strength = 4` ==> weak shock (increase by a factor of 3).
 - `i_shock_strength = 5` ==> ultra-weak shock (increase by a factor of 2).
- 2) Follow the shock propagation: decide the shock parameter (such as compression ratio, shock speed) at every time step.
- 3) After every 150 time steps, perform the following central tasks:
 - a) calculate the wave intensities (Alfvén waves) due to streaming protons at the shock front.
 - b) expand all shells behind the shock; update the momentum of the particles in every shells if necessary. [momentum change depends on the time lag.]
 - c) generate a new shell and add accelerated particles at this new shell behind the shock.
 - d) calculate particle diffusion, re-allocate particles among all shells based on the transport equation. Also, calculate particles that leak from shock front and the back; write them out to data file together with time and leakage location. These particles will escape the turbulent shock complex into the interplanetary medium and gyrating with Parker Spiral Interplanetary magnetic field (IMF). The motion of these particles will be followed using a Monte-Carlo code, which is the second part of the program.
 - e) increase the total shell number by 1.
- 4) Terminate the calculation after enough time steps to insure shock passes 1 AU.

About the program:

Units:

- 1) The velocity unit of the code is $V_0 = 52.483 \text{ km/s}$
- 2) The distance unit of the code is $L_0 = 1 \text{ AU} = 1.5 \times 10^8 \text{ km}$
- 3) From 1) and 2), the time unit is $t_0 = L_0/V_0 = 28580683 \text{ second}$
- 4) The density unit is $\rho_0 = 1.67 \times 10^{-24} \text{ g/cm}^3$ ==> a number density of $d = 1 \text{ nucleon/cm}^3$.
- 5) From 1) and 4), one can define the unit of B through $B_0^2 = \rho_0 V_0^2$ since $[B^2] = [\rho V^2]$. Thus $B_0 = 6.78 \times 10^{-6} \text{ Gauss}$.
(We note here that $1 \text{ Gauss}^2 = 1 \text{ erg/cm}^3$. $1 \text{ Tesla} = 10^4 \text{ Gauss}$. $1 \text{ J} = 10^7 \text{ erg}$)

To perform the above functionality, the following files are added apart from those contained in ZEUS-3D package.

----- FILES -----
`add_acc_ptcls.f:c` written by: Gang Li

diff_distr_fn_posn.f:c written by: Ken Rice
 diffuse_distr.f:c written by: Ken Rice
 modified by: Gang Li 02/21/2003
 modified by: Gang Li 03/11/2003: Add heavy ions

diffuse_distr_integ2.f:c written by: Ken Rice
 diffuse_distr_integ.f:c written by: Ken Rice
 distr_1AU_time.f:c written by: Ken Rice
 distr_fn_out.f:c written by: Ken Rice
 expand_distr.f:c written by: Ken Rice
 modified by: Gang Li
 find_forw_shock_2.f:c written by: Ken Rice
 modified 1: by Gang Li

momenta_calc_bohm.f:c written by: Ken Rice
 momenta_calc_gorlee.f:c written by: Gang Li
 observed_ptcls.f:c written by: Ken Rice
 parker_field.f:c written by: Ken Rice
 posn_distr.f:c written by: Ken Rice
 shell_obs.f:c written by: Ken Rice
 shell_track.f:c written by: Ken Rice
 solar.f:c written by: Ken Rice
 time_distr.f:c written by: Ken Rice
 wave_intens.f:c written by : Ken Rice

SECTION 2,

We now describe each code; its function and its input and output.

----- CODES DESCRIPTION -----

The main code is zeus3d.f:

Firstly, "zeus3d.f" opens the following files for data output:
 (the first letter at the beginning of the following lines denotes in which file the data-writing are performed.)

[z] ==> zeus3d.f
 [w] ==> wave_intens.f
 [d] ==> diffuse_distr.f
 [e] ==> expand_distr.f
 [a] ==> add_acc_ptcls.f

[d] open(13,file='ptcls_in_shell.dat', form='formatted',status='unknown')
 [d] open(14,file='distr_in_shell.dat', form='formatted',status='unknown')
 [?] open(35,file='obs_ptcls.dat', form='formatted',status='unknown')
 [?] open(36,file='posn_distr.dat', form='formatted',status='unknown')
 [?] open(37,file='time_distr.dat', form='formatted',status='unknown')
 [d] open(38,file='esc_distr.dat', form='formatted',status='unknown')
 [?] open(39,file='distr_1AU_time.dat', form='formatted',status='unknown')
 [w] open(43,file='waveintens.dat', form='formatted',status='unknown')

```
[w] open(44,file='kappa.dat', form='formatted',status='unknown')
[z] open(45,file='shock_momenta.dat', form='formatted',status='unknown')
[?] open(47,file='distr_fn_out.dat', form='formatted',status='unknown')
[z] open(48,file='shock_posn_comp.dat',form='formatted',status='unknown')
[z] open(55,file='momenta-hi.dat', form='formatted',status='unknown')
[d] open(58,file='esc_distr-hi.dat', form='formatted',status='unknown')
[?] open(66,file='f_acc-hi.dat', form='formatted',status='unknown')
[e] open(67,file='kappa-wave-I.dat', form='formatted',status='unknown')
[d] open(68,file='shell-location.dat', form='formatted',status='unknown')
[d] open(78,file='dist_shock.dat', form='formatted',status='unknown')
[d] open(79,file='dist_shock_hi.dat', form='formatted',status='unknown')
[d] open(88,file='esc_num_up.dat', form='formatted',status='unknown')
[d] open(89,file='esc_num_up_hi.dat', form='formatted',status='unknown')
[z] open(90, file ='zeus.log', status='unknown')
[z] open(91, file ="gl.input", status='unknown')
[z] open(92, file ="gl.output", status='unknown')
```

In the code, parameter "i_shock_strength" is used to control which shock profile is to be used. This parameter is passed to " gen_bndy.f " from zeus3d.f. All information passed to the code are written to file zeus.log(90).

```
write(90,*) " i_shock_strength is: ", i_shock_strength
write(90,*) " (1: very strong shock ) "
write(90,*) " (2: strong shock ) "
write(90,*) " (3: mediate shock ) "
write(90,*) " (4: weak shock ) "
write(90,*) " (5: ultra-weak shock ) "
```

The code starts with no_shells = 1. After every 150 time steps, a new shell is constructed. Once a new shell is constructed, the above mentioned steps a) - e) are performed:

These corresponds to the following code segment:

```
call wave_intens    ! Get the wave intensity and the maximum momentum of proton and heavy ion.
call expand_distr   ! Calculate the simultaneous expansion of the shells with the solar wind.
call add_acc_ptcls  ! Add the newly generated particles to the newly generated shell.
call diffuse_distr  ! Calculate the change of distribution among all shells due to diffusion.
```

1) wave_intens.f:c written by : Ken Rice
modified by: Gang Li

Calculate the wave intensity, generated by streaming protons. Along the way, calculate the maximum proton energy and heavy ion energy.

Data are written to following files:

```
open(43,file='waveintens.dat', form='formatted',status='unknown')
open(44,file='kappa.dat', form='formatted',status='unknown')

Write(43,*) k*Vsw/2.0/pi, energy,
* 2.0E18*Ipluso(i)*2.0*pi/Vsw,
* 2.0e18*Iplus(i,no_shells)*2.0*pi/Vsw,
* 2.0E18*Ibohm(i)*2.0*pi/Vsw
```

```
Write(44,111) k*Vsw/2.0/pi, energy, kappa(i,no_shells),  
* kappa_hi(i,no_shells),4.0D0/3.0D0/pi*rg*vel,  
* x1a(shock_begin)
```

In waveintens.dat(43), we have in sequence:

- 1) frequency f in solar wind frame, $f = k * V_{sw} / (2 * \pi)$,
- 2) energy,
- 3) solar wind power spectra P in unit (nT^2/Hz), satisfying $P df = 2 * I dk$
- 4) turbulence power spectra P in unit (nT^2/Hz), satisfying $P df = 2 * I dk$
- 5) turbulence power spectra P in accordance with Bohm approximation.

In kappa.dat(44) we have in sequence:

- 1) frequency f in solar wind frame, $f = k * V_{sw} / (2 * \pi)$,
- 2) energy,
- 3) diffusion coefficient for proton,
- 4) diffusion coefficient for heavy ion,
- 5) diffusion coefficient for Bohm approximation.
- 6) shock location.

Plotting procedure:

```
split -250 kappa.dat KAPPA
```

Then plot KAPPA?? ($x=2, y = 3,4,5$) for kappa_p, kappa_hi, kappa_Bohm

```
split -250 waveintensity.dat WI
```

Then plot WI?? ($x=1, y = 3,4$) for I_{total} , I_o

- 2) expand_distr.f:c written by : Ken Rice
modified by: Gang Li

Follow the expansion of each shell in the solar wind. No diffusion in this step.

The diffusion coeff. and wave intensity in these shells are recorded in file (67).

```
write(67,*) "--- time is ---", time  
do i = 1, no_shells  
  write(67,*) i,temp, Energy, Iplus(j,i),kappa(j,i), kappa_hi(j,i)  
enddo
```

- 3) add_acc_ptcls.f written by: Gang Li

This subroutine add the injected protons and heavy ions at the first shell.

- 4) diffuse_distr.f:c written by: Ken Rice
modified by: Gang Li 02/21/2003
modified by: Gang Li 03/11/2003: Add heavy ions

This file follows the diffusion of the particles between all shells.
It solves the diffusion equation for the current time step for all the shells.
Within this file, two subroutines are called as shown in follows.

call diffuse_distr_integ(num_ptcls,fi,shell_posn, R_shell_x(l), R_shell_x(l+1), temp)

call diff_distr_fn_posn(dens_ptcls,fi,shell_posn, x1a(shock_begin)+esc_length, temp)

3.1) diffuse_distr_integ(num_ptcls,fo, ri, r1, r2, temp)

c written by: Ken Rice

c modified by: Gang Li

The parameters are:

c----- num_ptcls is the output,

c----- fo: input number density, shell_posn: input shell position,

c----- ri = $0.5*(r1+r2)$ is the shell center location.

c----- r1: beginning loc. of the shell.

c----- r2: ending loc. of the shell.

c----- temp: the length scale (the width of the Gaussian distribution.)

This subroutine gives the number of particles in the lth shell, i.e. between R_shell_x(l) and R_shell_x(l+1).

3.2) diff_distr_fn_posn(dens_ptcls, fo, ri, r, temp)

c written by: Ken Rice

c modified by: Gang Li

The parameters are:

c----- dens_ptcls is the output.

c----- fo: input number density,

c----- ri is the shell center location.

c----- the location of where the dens_ptcl is calculated.

c----- temp: the length scale (the width of the Gaussian distribution.)

This subroutine calculate the number density at r due to the shell located at ri.

The data output from this file are:

A1) ptcls_in_shell.dat (13)

This file contains particle number in every ith shell and for every jth momentum bin.

```
write(13,*) Energy,distr_array(i,j)* factor_num,
*      distr_array_hi(i,j)*factor_num
```

To plot:

1) csplit ptcls_in_shell.dat -f PTCL /time/ {*}

choose the time period for your interest, for example, PTCLae then

2) csplit PTCLae -f SHELL /shell/ {*}

Then files of SHELL(??) with ?? = 00, 01, 02, ... etc will contain the data in each shell.

plot (x=1, y = 2) for protons in each shell.

plot (x=1, y = 3) for hvyions in each shell.

A2) distr_in_shell.dat (14)

This file contains particle number density in every ith shell and for every jth momentum bin. (number / d^3p)

```
write(14,*) Energy,distr_array(i,j),distr_array_hi(i,j),time
```

Note, distr_array(i,j) has the unit of (number / d^3p).

To plot:

1) csplit distr_in_shell.dat -f DST-SHELL /time/ {*}

choose the time period for your interest, for example, DST-SHELL06 then

2) csplit DST-SHELL06 -f SHELL /shell/ {*}

Then files of SHELL(??) with ?? = 00, 01, 02, ... etc will contain the data in each shell.

plot (x=1, y = 2) for proton deptsity in each shell.

plot (x=1, y = 3) for hvyion density in each shell.

A3) shell-location.dat (68)

The location of the shell boundaries, as well as total number of particles in each shell are recorded in (68).

These corresponds from i = 1, 2, 3, no_shells + 1

and the number are between 1-to-2, 2-to-3, ,no_shells+1-to-infinity,

```
write(68,*) "--- time is ---", time
do i = 1, no_shells + 1
  write(68,*) i, R_shell_i(i), R_shell_x(i),total_num_p(i),
*      total_num_hi(i)
enddo
```

where total_num_p(i) is for proton, total_num_hi(i) is for hvy ion.

```
| | | | |
| | | | |
| | | | |
```

i= 1 2 no_shells no_shells+1 (=shock_begin)

To plot: csplit shell-location.dat -f SHELL /time/ {*}

then choose SHELL?? (eg. SHELL03)

and plot (x=3,y=4,dy=4)

B1) esc_distr.dat(38) and esc_distr-hi.dat(58)

```
write(38,*) Energy, esc_dens_up(p_int), x1a(shock_begin)+4.0D0*temp, time
write(58,*) Energy, esc_dens_up_hi(p_int), x1a(shock_begin)+4.0D0*temp_hi, time
```

These two files contain proton and heavy ion density at location
x1a(shock_begin)+4.0D0*temp.

Note: esc_dens_up(p_int) is the phase density and has unit $\#/(d^3r*d^3p)$.

B2) esc_num_up.dat (88) and esc_num_up_hi.dat (89).

```
If (Exp(p) .LE. p_cut ) Then
  write(88,*) Energy, esc_number_up(p_int),
*      esc_number_up(p_int), temp_counter1,
*      x1a(shock_begin)+escape_length(p_int)
ELSE
  write(88,*) Energy, esc_number_up(p_int), 0.,
```

```

*      temp_counter1,
*      x1a(shock_begin)+escape_length(p_int)
ENDIF

If (Exp(p) .GE. p_hi_cut ) Then
  write(89,*) Energy, esc_number_up_hi(p_int), 0.,
*   temp_counter2,
*   x1a(shock_begin)+escape_length_hi(p_int)
ELSE
  write(89,*) Energy, esc_number_up_hi(p_int),
*   esc_number_up_hi(p_int), temp_counter2,
*   x1a(shock_begin)+escape_length_hi(p_int)
ENDIF

```

Note: esc_number_up(p_int) has unit of $\#/(d^3p)$.

split -250 esc_num_up.dat ESC (split -250 esc_num_up_hi.dat ESC-HI)
 plot (x =1, y =2) for (energy, esc_num)
 or plot (x =1, y =4) for (energy, temp_counter1) where
 temp_counter is the time-integrated escaped particle number.

C) current_at_shock.dat (78) and distr_at_shock.dat (79)

```

write(78,*) Energy, den_sh_j, den_sh_hi_j,x1a(shock_begin), time

write(79,*) Energy, distr_array(no_shells, p_int),
*   distr_array_hi(no_shells, p_int), time

```

These file contains particle current (proton and heavy ions)
 and particle distribution at the shock location.

Note, den_sh_j and den_sh_hi_j have the unit of $\#/(cm^2 s MeV)$.
 and distr_array(i,j) has the unit of (number / d^3p).

the plotting procedure:

```

split -250 current_at_shock.dat CRT
split -250 dist_at_shock.dat DIST

```

plot (x=1,y=2) for energy-current and (x=1,y=3) for hvy-ion current.

5) Back to file zeus3d.f. File "shock_momenta.dat", which is opened through,

```

open(45,file='shock_momenta.dat',form='formatted')

write(45,2110)x1a(shock_begin),(time-1.00003167)*to/3600.,
*   p_sml,p_big,energy_sml, energy_big

```

contains the location of the shock, maximum and minimum proton energy.

Similarly, file "momenta-hi.dat" opened through

```
open(55,file='momenta-hi.dat',form='formatted',  
*      status='unknown')
```

```
write(55,2110)x1a(shock_begin),(time-1.00003167)*to/3600.,  
*      p_sml,p_big,energy_sml, energy_big
```

contains the location of the shock, maximum and minimum heavy ion energy.

Also, file shock_posn_comp.dat opened through

```
open(48,file='shock_posn_comp.dat',form='formatted',  
*      status='unknown')
```

```
write(48,2100)x1a(shock_begin),comp_rat,  
*      shock_speed*vo/1000.,v1(shock_end+2,1,5)*vo/1000.,  
*      (time - 1.00003167)*to/3600.
```

contains the location, comp_rat, speed, time of the shock.

plot (x = 1, y = 2) for comp. ratio

plot (x = 1, y = 3) for shock speed.

plot (x = 1, y = 4) for upstream solar wind speed.