Revista de Matemática: Teoría y Aplicaciones 2016 **23**(1) : 41–61
cimpa – ucr issn: 1409-2433 (Print), 2215-3373 (Online)

MAGNETOHYDRODYNAMIC EQUATIONS (MHD) GENERATION CODE

CÓDIGO DE GENERACIÓN DE ECUACIONES MAGNETOHIDRODINÁMICAS (MHD)

Francisco Frutos Alfaro* Rodrigo Carboni Méndez†

Received: 2 Mar 2015; Revised: 28 Aug 2015;
Accepted: 30 Sep 2015

---

*School of Physics, University of Costa Rica, San José, Costa Rica. E-Mail: frutos@fisica.ucr.ac.cr
†Same address as/*Misma dirección que*: F. Frutos. E-Mail: rcarboni@fisica.ucr.ac.cr

Abstract

A program to generate codes in Fortran and C of the full magnetohydrodynamic equations is shown. The program uses the free computer algebra system software REDUCE. This software has a package called EXCALC, which is an exterior calculus program. The advantage of this program is that it can be modified to include another complex metric or spacetime. The output of this program is modified by means of a LINUX script which creates a new REDUCE program to manipulate the magnetohydrodynamic equations to obtain a code that can be used as a seed for a magnetohydrodynamic code for numerical applications. As an example, we present part of the output of our programs for Cartesian coordinates and how to do the discretization.

Keywords: magnetohydrodynamic; Cartan's exterior calculus; numerical discretization.

Resumen

Se presenta un programa para generar códigos en Fortran y C de las ecuaciones magnetohidrodinámicas. El programa utiliza el software libre de álgebra computacional REDUCE. Este software tiene un paquete llamado EXCALC, que es un programa de cálculo exterior. La ventaja de este programa es que puede ser modificado para incluir otra métrica compleja o espacio-tiempo complejo. La salida de este programa es modificada por medio de una secuencia de comandos LINUX que crea un nuevo programa en REDUCE para manipular las ecuaciones magnetohidrodinámicas para obtener un código que puede ser utilizado como una semilla para un código de magnetohidrodinámica para aplicaciones numéricas. A modo de ejemplo, se presenta parte de la salida de nuestros programas en coordenadas cartesianas y como hacer la discretización.

Palabras clave: magnetohidrodinámica; cálculo exterior de Cartan; discretización numérica.

Mathematics Subject Classification: 58A15, 65D25, 76W05.

1 Introduction

Nowadays, there are programs that generate codes for a given mesh arrangement, but there is no a program to generate the differential equation in a given spacetime. Now, we show how it is possible by means of a EXCALC package [16] of REDUCE [8] and a LINUX script. REDUCE is a free computer algebra system (CAS) software intended for algebraic manipulation. In that sense, it is similar to Mathematica or Maple. The EXCALC package solves problems using the Cartan formalism or differential forms [1, 7].

Due to the advent of new technology, there is an interest in plasma simulations and visualizations e.g. [4, 6, 14]. There are a lot of information on these subjects in the Internet [2, 12, 9, 10, 17]. Moreover, the mesh or grid generation is important in designing complex engineering structures or complex nonsymmetric systems [15].

We wrote a REDUCE program (`MHD.red`) and a long UNIX script (`discretized-mhd`) mainly for magnetohydrodynamics (MHD), but the program and script we present here are easy to modify for other purposes. Our program could be used together with a mesh generation code to solve complex problems. At the moment, our UNIX script was written for a equally spaced grid, but it could be modified for other type of grids. The results provided by a new script written by the user, could be sensitive to different numerical methods or numerical techniques, therefore the user should be careful to modify it.

The paper is organized as follows. In section 2, a brief description of MHD is presented. A short summary of EXCALC is given in section 3. The explanation of the program is described in section 4. Moreover, we present the results of the discretization to Cartesian coordinates as an example. The conclusions and future work is discussed in section 5.

2 MHD

MHD is the study of the dynamics of electrically conducting fluids. Examples of such fluids include plasmas, liquid metals, and salt water or electrolytes. Plasmas can be regarded as fluids since the mean free paths for collisions between the electrons and ions are macroscopically long. Thus, collective interactions between large numbers of plasma particles can isotropize the particle velocity distributions in some local mean reference frame, thereby making it sensible to describe the plasma macroscopically by a mean density, velocity, and pressure. These mean quantities can then be shown to obey the same conservation laws of mass, momentum and energy for fluids.

The fundamental concept behind MHD is that magnetic fields can induce currents in a moving conductive fluid, which in turn creates forces on the fluid and also changes the magnetic field itself. The set of equations which describe MHD are a combination of the Navier-Stokes equations of fluid dynamics and Maxwell's equations of electromagnetism.

An important physical parameter that defines the way MHD treats the systems is resistivity. When the resistivity is very low plasmas can be treated as perfect conductors (ideal MHD in the infinite magnetic Reynolds number limit) and the lines of force appear to be dragged along with the conductor.

When the resistivity cannot be neglected the magnetic field can generally move through the fluid following a diffusion law with the resistivity of the plasma serving as a diffusion constant. According to this the solutions to the ideal MHD equations are only applicable for a limited time before diffusion becomes too important to ignore. The diffusion time across a solar active region is hundreds to thousands of years. The interested reader may consult the references [3, 5, 11, 13].

There are many applications of MHD but in this article we are interested in those related to astrophysics and cosmology. Plasma is the constituent of many astrophysical objects of the Universe: stars, nebulae, interplanetary, interstellar and intergalactic media.

Many astrophysical problems require the treatment of magnetized fluids in dynamical strongly curved spacetimes. Such problems include the origin of gamma-ray bursts, magnetic braking of differential rotation in incipient neutron stars arising from stellar core collapse or binary neutron star merger, the formation of jets and magnetized disks around newborn black holes and many others [18].

Now, we summarize the set of MHD equations, with which one has to treat.

MHD Equations:

Continuity Equation

The conservation of mass in a fluid tells us that

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \mathbf{J}_s = 0, \tag{1}$$

where ρ is the density, $\mathbf{v}$ is the fluid velocity and $\mathbf{J}_s = \rho \mathbf{v}$.

Maxwell Equations

The electric Gauß law is

$$\epsilon_0 \nabla \cdot \mathbf{E} = \rho_e, \tag{2}$$

where $\mathbf{E}$ is the electric field, ρ_e is the electric charge density and ϵ_0 is the vacuum permittivity.

The magnetic Gauß law is

$$\nabla \cdot \mathbf{B} = 0, \tag{3}$$

where $\mathbf{B}$ is the magnetic field.

The Ampère law is

$$\nabla \times \mathbf{B} = \mu_0 \mathbf{J} + \mu_0 \epsilon_0 \frac{\partial \mathbf{E}}{\partial t}, \tag{4}$$

where $\mathbf{J}$ is the electric current density and μ_0 is the vacuum permeability.
The Faraday law is

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}. \tag{5}$$

Equation of Motion

The equation of motion of a fluid or the Navier-Stokes equation with electromagnetic fields is

$$\frac{\partial \mathbf{J}_s}{\partial t} = \mathbf{F}_L + \nabla \cdot [\mathbf{P} - \rho \mathbf{v}\mathbf{v}], \tag{6}$$

where $\mathbf{P}$ is the stress tensor with components

$$P_{ij} = -P\delta_{ij} + \mathcal{T}_{ij}, \tag{7}$$

here P is the pressure and $\mathcal{T}_{ij}$ represents the viscosity stress tensor components, which are given by

$$\mathcal{T}_{ij} = \mu \left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3} \nabla \cdot \mathbf{v} \delta_{ij} \right) + \nu \nabla \cdot \mathbf{v} \delta_{ij}, \tag{8}$$

with μ and ν are the first and second viscosity coeficients.

$\mathbf{F}_L$ is the volume Lorentz force given by

$$\begin{aligned} \mathbf{F}_L &= \rho_e \mathbf{E} + \mathbf{J} \times \mathbf{B} \\ &= \nabla \cdot \mathbf{T} - \epsilon_0 \mu_0 \frac{\partial \mathbf{S}}{\partial t}, \end{aligned} \tag{9}$$

where $\mathbf{S}$ is the Poynting vector

$$\mathbf{S} = \frac{1}{\mu_0} \mathbf{E} \times \mathbf{B}, \tag{10}$$

and $\mathbf{T}$ is the electromagnetic tensor with following components

$$T_{ij} = \epsilon_0 E_i E_j + \frac{1}{\mu_0} B_i B_j - \left(\epsilon_0 \frac{E^2}{2} + \frac{B^2}{2\mu_0} \right) \delta_{ij}. \tag{11}$$

The Ohm Law
The Ohm law is

$$\mathbf{E} = \eta\mathbf{J} - \mathbf{v} \times \mathbf{B}, \tag{12}$$

where η is the resistivity.

Equation of State
The equation of state for an ideal gas is

$$P = \rho RT, \tag{13}$$

where T is the temperature and R is the gas constant.

Equation of energy conservation
The equation of energy conservation is

$$\frac{\partial}{\partial t}\left[\rho\left(\epsilon + \frac{v^2}{2}\right)\right] = -\left[\nabla \cdot \mathbf{Q} - \mathbf{J} \cdot \mathbf{E}\right], \tag{14}$$

where

$$\mathbf{Q} = \left(\epsilon + \frac{v^2}{2}\right)\mathbf{J}_s + \mathbf{q} - \mathbf{P} \cdot \mathbf{v} \tag{15}$$

and ϵ is the internal energy

$$\epsilon = C_V T = \frac{P}{(\gamma - 1)\rho} \tag{16}$$

here γ is the adiabatic exponent

$$\gamma = \frac{C_P}{C_V} = \frac{R + C_V}{C_V} \tag{17}$$

and $\mathbf{q}$ is the heat flux vector

$$\mathbf{q} = -k\nabla T, \tag{18}$$

with k is the thermal conductivity.

3 EXCALC

EXCALC is a package under REDUCE intended to solve problems with Cartan exterior calculus. The advantage of using it is that given equations can be written independent of the coordinate system. The outcomes of the calculations from these equations are expressed in the chosen coordinate system. Here, it is a list of the most used commands:

^	Exterior multiplication
`@`	Partial differentiation
#	Hodge $\star$ operator
_|	Inner product
|_	Lie derivative
`COFRAME`	Declaration of a coframe
`d`	Exterior differentiation
`FDOMAIN`	Declaration of implicit dependencies
`FRAME`	Declares the frame dual to the coframe
`METRIC`	Clause of `COFRAME` to specify a metric
`PFORM`	Declaration of exterior forms

The principal operations with 1-forms or vectors are listed below

Cross product	#(`v^B`) (`B` and `v` are vectors or 1-forms)
Nabla	`d P` (`P` is a function or scalar)
Divergence	#`d`# `F` (`F` is a vector or 1-form)
Curl	#`d F` (`F` is a vector or 1-form)

For more information about this REDUCE package, the interested reader may consult the references (for instance, [8]).

4 The program

The REDUCE program `MHD.red` to transform the MHD equations into Cartesian coordinates and the script `discretized-mhd` are included as an appendix. They can be emailed by the authors if requested, and in a near future they will be available at our *Space Research Center* Webpage `http://cinespa.ucr.ac.cr/software/`

4.1 Example of the script for Cartesian coordinates

Now, we will present the modified output of the programs for a given MHD equation using an UNIX script. Let us consider the Ampère law (4), the Faraday law (5) and the Ohm law (12). If we substitute the first one and the third one into the second one, we get the induction equation:

$$\frac{\partial \mathbf{B}}{\partial t} + \frac{1}{\mu_0}\nabla \times [\eta\nabla \times \mathbf{B} - \mathbf{v} \times \mathbf{B}] = 0, \tag{19}$$

where we have not taken into account the time variation of the electric field, and $\eta = \eta(x_1, x_2, x_3)$.

For example, the outcome from our program for the y component of the induction equation (Cartesian coordinates) is

```
induct( - x2) := @(b( - x2),t)
- b( - x1)*@(v( - x2),x1) + b( - x2)*@(v( - x1),x1)
+ b( - x2)*@(v( - x3),x3) - b( - x3)*@(v( - x2),x3)
- @(b( - x1),x1)*v( - x2) + @(b( - x2),x1)*v( - x1)
+ @(b( - x2),x3)*v( - x3) - @(b( - x3),x3)*v( - x2)
+ (@(b( - x1),x1,x2)*eta - @(b( - x2),x1,x1)*eta
+ @(b( - x1),x2)*@(eta,x1) - @(b( - x2),x1)*@(eta,x1)
- @(b( - x2),x3)*@(eta,x3) + @(b( - x3),x2)*@(eta,x3)
- @(b( - x2),x3,x3)*eta + @(b( - x3),x2,x3)*eta)/mu_0
```

As this output is not easy to handle, we translate such equation to a FORTRAN or C code, then we wrote a script that modify this outcome into equations with a prescribed discretization method. To this aim, we employ the UNIX command `sed` [19]. This command change the output using the prescribed discretization method. For instance, the following commands

```
sed -e 's:@(b( - x1),t):(bx(n+1,i,j,k)-bx(n,i,j,k))/dt:g'
mhdeqs r mhdeqsnew
sed -e 's:@(v( - x2),x1):(vy(n,i+1,j,k)-vy(n,i-1,j,k)
)/(2*dx):g' mhdeqs r mhdeqsnew
```

produce the translation

$$\frac{\partial b_x}{\partial t} \longrightarrow \frac{b_x(n+1,\, i,\, j,\, k) - b_x(n,\, i,\, j,\, k)}{\Delta t}$$

$$\frac{\partial v_y}{\partial x} \longrightarrow \frac{v_y(n,\, i+1,\, j,\, k) - v_y(n,\, i-1,\, j,\, k)}{2\Delta x}.$$

The output of our REDUCE program is `mhdeqs` (see Appendix) and the `mhdeqsnew`, which should be created before one runs the script, will contain all modifications of the entire system of equations.

4.2 Example of the code generation

Here, we show a part of the output after the script is run. In this case, the y component of the induction equation is converted into

```
induct_y := (by(n+1,i,j,k)-by(n,i,j,k))/dt
+ ((bx(n,i+1,j+1,k)-bx(n,i+1,j-1,k)
-bx(n,i-1,j+1,k)+bx(n,i-1,j-1,k))/(4*dx*dy)*eta(n,i,j,k)
+ (bx(n,i,j+1,k)-bx(n,i,j-1,k))/(2*dy)*(eta(n,i+1,j,k)
-eta(n,i-1,j,k))/(2*dx)
- (by(n,i+2,j,k)-2*by(n,i,j,k)+by(n,i-2,j,k))/(4*dx*dx)*
eta(n,i,j,k)
- (by(n,i+1,j,k)-by(n,i-1,j,k))/(2*dx)*(eta(n,i+1,j,k)
```

```
-eta(n,i-1,j,k))/(2*dx)
- (by(n,i,j,k+2)-2*by(n,i,j,k)+by(n,i,j,k-2))/(4*dz*dz)*
eta(n,i,j,k)
- (by(n,i,j,k+1)-by(n,i,j,k-1))/(2*dz)*(eta(n,i,j,k+1)
-eta(n,i,j,k-1))/(2*dz)
+ (bz(n,i,j+1,k+1)-bz(n,i,j+1,k-1)
-bz(n,i,j-1,k+1)+bz(n,i,j-1,k+1))/(4*dy*dz)*eta(n,i,j,k)
+ (bz(n,i,j+1,k)-bz(n,i,j-1,k))/(2*dy)*(eta(n,i,j,k+1)
-eta(n,i,j,k-1))/(2*dz))/mu_0
```

Thus, it is possible to use this output as a FORTRAN or C code.

Now, let us see how to modify the source code for spherical and cylindrical coordinate systems.

For a spherical coordinate system, you have to modify the first lines of the REDUCE code as follows

```
% Calculate in spherical coordinates
system the MHD
% equations.
coframe e r = d r, e theta = r*d theta,
e phi = r*sin(theta)*d phi$
frame x$
fdomain v = v(t,r,theta,phi), p = p(t,r,theta,phi)$
fdomain rho=rho(t,r,theta,phi), je=je(t,r,theta,phi)$
fdomain pstar=pstar(t,r,theta,phi), b=b(t,r,theta,phi)$
fdomain eta=eta(t,r,theta,phi),
energy=energy(t,r,theta,phi)$
fdomain jm=jm(t,r,theta,phi), jmp=jmp(t,r,theta,phi)$
fdomain ee=ee(t,r,theta,phi), a=a(t,r,theta,phi)$
```

For a cylindrical coordinate system, you have to modify the first lines of the REDUCE code as follows

```
% Calculate in cylindrical coordinates
system the MHD
% equations.
coframe e r = d r, e theta = r*d theta, e z = d z$
frame x$
fdomain v = v(t,r,theta,z), p = p(t,r,theta,z)$
fdomain rho=rho(t,r,theta,z), je=je(t,r,theta,z)$
fdomain pstar=pstar(t,r,theta,z), b=b(t,r,theta,z)$
fdomain eta=eta(t,r,theta,z), energy=energy(t,r,theta,z)$
fdomain jm=jm(t,r,theta,z), jmp=jmp(t,r,theta,z)$
fdomain ee=ee(t,r,theta,z), a=a(t,r,theta,z)$
```

5 Conclusion

In this paper, we show how to produce quickly a computer code useful to implement simulations or visualizations. The automatic generation of the computer code is possible by means of REDUCE programs and UNIX scripts that modify and manipulate the output of the programs and generate computer codes with a given discretization method. The programs shown here are flexible and easy to modify for other purposes.

Due to the rapid advance of technology, there is an interest in computer simulations and visualizations, in particular in plasma physics. Our program together with recent developed grid generation codes helps solve complex plasma phenomena.

To implement a computer program for solving a given plasma problem the programmer would have to include the code generated with our software as a solver code for the MHD differential equations. The boundary conditions and other features such as visualization subroutines should be included by the programmer.

There are softwares in the Internet for solving plasma physics problems, but sometimes one needs to solve a problem in a special coordinate systems and these programs are fixed to a given coordinate system and there is no way to modify them. In this case our software can be useful.

6 Appendix

A. The REDUCE Program

```
% REDUCE Program MHD.red
% It is based on an example of E. Schruefer
out mhdeqs$
load excalc$
off nat$
% Problem:
%
% Calculate in spherical coordinates system the MHD
% equations.
% coframe e r = d r, e theta = r*d theta,
% e phi = r*sin(theta)*d phi$
% frame x$
% fdomain v = v(t, r, theta, phi),
% p = p(t, r, theta, phi)$
%
```

```
% Calculate in cartesian coordinates system the MHD
% equations.
coframe e x1 = d x1, e x2 = d x2, e x3 = d x3$
frame x$
fdomain v=v(t,x1,x2,x3), p=p(t,x1,x2,x3)$
fdomain rho=rho(t,x1,x2,x3), je=je(t,x1,x2,x3)$
fdomain pstar=pstar(t,x1,x2,x3), b=b(t,x1,x2,x3)$
fdomain eta=eta(t,x1,x2,x3), energy=energy(t,x1,x2,x3)$
fdomain jm=jm(t,x1,x2,x3), jmp=jmp(t,x1,x2,x3)$
fdomain ee=ee(t,x1,x2,x3), a=a(t,x1,x2,x3)$
pform v(k)=0, je(k)=0, p=0, rho=0$
v := v(-k) * e(k);
% je := rho * v;
je := je(-k) * e(k);
%factor e, @$
factor e$
on rat$
%
% First we calculate the continuity equation.
% drho/dt + nabla.je = 0
pform conteq=0$
conteq := @(rho,t) + #d# je;
%
% Next we calculate the equation of motion.
% d je/dt + nabla pstar + (nabla.je) v + je.nabla v
% + (nabla.b) b + b.nabla b = 0
% nabla.b = 0
pform moveq=1, moveq(-k)=0, pstar=0, b(-k)=0$
b := b(-k) * e(k);
% pstar := p+(b(k) * b(-k))/(2*mu_0);
% moveq := @(je,t) + d pstar + (#d# je)*v
% + rho*((v(k) * x(-k)) |_ v - (1/2)*d (v(k) * v(-k)))
% + (#d# b)*b + ((b(k) * x(-k)) |_ b
% - (1/2)*d (b(k) * b(-k)));
moveq := @(je,t) + d pstar + (#d# je)*v
+ rho*((v(k) * x(-k)) |_ v - (1/2)*d (v(k) * v(-k)))
+ ((b(k) * x(-k)) |_ b - (1/2)*d (b(k) * b(-k)));
moveq(-k) := x(-k) _| moveq;
%
% Next we calculate Gauss equation for the magnetic
% field.
% nabla.b = 0
pform gauss=0;
gauss := #d# b;
%
```

```
% Next we calculate the induction equation.
pform induct=1, induct(-k)=0, ee=1, ee(-k)=0,
jm=1, jm(-k)=0, eta=0$
pform jmp=1, jmp(-k)=0, jmpp=1, jmpp(-k)=0,
a=1, a(-k)=0, bb=1, bb(-k)=0$
pform induct2=1, induct2(-k)=0, ee2, ee2(-k)=0$
jm := (#d b)/mu_0;
jm(-k) := x(-k) _| jm;
% b := #d a;
a := a(-k) * e(k);
jmp:= jmp(-k) * e(k);
bb := #d a;
jmpp := (#d bb)/mu_0 - jmp;
jmpp(-k) := x(-k) _| jmpp;
ee := eta*jm - #(v^b);
ee(-k) := x(-k) _| ee;
ee2 := eta*jm - #(v^bb);
ee2(-k) := x(-k) _| ee2;
induct := @(b,t) + #d ee;
induct(-k) := x(-k) _| induct;
induct2 := @(a,t) + eta*(#d bb)/mu_0 - #(v^bb);
induct2(-k) := x(-k) _| induct2;
%
% Finally we calculate the energy conservation.
% energy := rho * v**2/2 + p/(gamma-1) + b**2/(2*mu_0);
pform energy=0, enconser=0$
% energy := rho * (v(k) * v(-k))/2 + p/(gamma-1)
% + (b(k) * b(-k))/(2*mu_0);
enconser := @(energy,t) + #d# ((energy+pstar)*v
- ((v(k) * x(-k)) _| b)*b);
%
clear v, x, b, p, pstar, eta, conteq, moveq, gauss,
ee, energy, enconser$
remfac e, @$
remfdomain p, v, b, energy, eta, pstar, rho, je$
shut mhdeqs$
bye;
```

B. The UNIX Script

```
cp MHDEQS-Dif.red MHDEQS-Dif-old.red
sed -e 's:@(v( - x1),t):(vx(n+1,i,j,k)-vx(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x1),x1):dfx(i,j,k,vx,dvxx):g'
mhdeqs r MHDEQS-Dif.red
```

```
sed -e 's:@(v( - x1),x2):dfy(i,j,k,vx,dvxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x1),x3):dfz(i,j,k,vx,dvxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x2),t):(vy(n+1,i,j,k)-vy(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x2),x1):dfx(i,j,k,vy,dvyx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x2),x2):dfy(i,j,k,vy,dvyy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x2),x3):dfz(i,j,k,vy,dvyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x3),t):(vz(n+1,i,j,k)-vz(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x3),x1):dfx(i,j,k,vy,dvzx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x3),x2):dfy(i,j,k,vy,dvzy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x3),x3):dfz(i,j,k,vy,dvzz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x1),x1,x1):ddfxx(i,j,k,vx,ddvxxx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x1),x1,x2):ddfxy(i,j,k,vx,ddvxxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x1),x1,x3):ddfxz(i,j,k,vx,ddvxxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x1),x2,x1):ddfxy(i,j,k,vx,ddvxxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x1),x2,x2):ddfyy(i,j,k,vx,ddvxyy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x1),x2,x3):ddfyz(i,j,k,vx,ddvxyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x1),x3,x1):ddfxz(i,j,k,vx,ddvxxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x1),x3,x2):ddfyz(i,j,k,vx,ddvxyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x1),x3,x3):ddfzz(i,j,k,vx,ddvxzz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x2),x1,x1):ddfxx(i,j,k,vy,ddvyxx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x2),x1,x2):ddfxy(i,j,k,vy,ddvyxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x2),x1,x3):ddfxz(i,j,k,vy,ddvyxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x2),x2,x1):ddfxy(i,j,k,vy,ddvyxy):g'
```

```
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x2),x2,x2):ddfyy(i,j,k,vy,ddvyyy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x2),x2,x3):ddfyz(i,j,k,vy,ddvyyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x2),x3,x1):ddfxz(i,j,k,vy,ddvyxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x2),x3,x2):ddfyz(i,j,k,vy,ddvyyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x2),x3,x3):ddfzz(i,j,k,vy,ddvyzz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x3),x1,x1):ddfxx(i,j,k,vz,ddvzxx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x3),x1,x2):ddfxy(i,j,k,vz,ddvzxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x3),x1,x3):ddfxz(i,j,k,vz,ddvzxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x3),x2,x1):ddfxy(i,j,k,vz,ddvzxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x3),x2,x2):ddfyy(i,j,k,vz,ddvzyy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x3),x2,x3):ddfyz(i,j,k,vz,ddvzyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x3),x3,x1):ddfxz(i,j,k,vz,ddvzxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x3),x3,x2):ddfyz(i,j,k,vz,ddvzyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(v( - x3),x3,x3):ddfzz(i,j,k,vz,ddvzzz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x1),t):(bx(n+1,i,j,k)-bx(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x1),x1):dfx(i,j,k,bx,dbxx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x1),x2):dfy(i,j,k,bx,dbxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x1),x3):dfz(i,j,k,bx,dbxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x2),t):(by(n+1,i,j,k)-by(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x2),x1):dfx(i,j,k,by,dbyx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x2),x2):dfy(i,j,k,by,dbyy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x2),x3):dfz(i,j,k,by,dbyz):g'
mhdeqs r MHDEQS-Dif.red
```

```
sed -e 's:@(b( - x3),t):(bz(n+1,i,j,k)-bz(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x3),x1):dfx(i,j,k,bz,dbzx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x3),x2):dfy(i,j,k,bz,dbzy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x3),x3):dfz(i,j,k,bz,dbzz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x1),x1,x1):ddfxx(i,j,k,bx,ddbxxx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x1),x1,x2):ddfxy(i,j,k,bx,ddbxxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x1),x1,x3):ddfxz(i,j,k,bx,ddbxxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x1),x2,x1):ddfxy(i,j,k,bx,ddbxxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x1),x2,x2):ddfyy(i,j,k,bx,ddbxyy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x1),x2,x3):ddfyz(i,j,k,bx,ddbxyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x1),x3,x1):ddfxz(i,j,k,bx,ddbxxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x1),x3,x2):ddfyz(i,j,k,bx,ddbxyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x1),x3,x3):ddfzz(i,j,k,bx,ddbxzz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x2),x1,x1):ddfxx(i,j,k,by,ddbyxx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x2),x1,x2):ddfxy(i,j,k,by,ddbyxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x2),x1,x3):ddfxz(i,j,k,by,ddbyxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x2),x2,x1):ddfxy(i,j,k,by,ddbyxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x2),x2,x2):ddfyy(i,j,k,by,ddbyyy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x2),x2,x3):ddfyz(i,j,k,by,ddbyyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x2),x3,x1):ddfxz(i,j,k,by,ddbyxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x2),x3,x2):ddfyz(i,j,k,by,ddbyyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x2),x3,x3):ddfzz(i,j,k,by,ddbyzz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x3),x1,x1):ddfxx(i,j,k,bz,ddbzxx):g'
```

```
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x3),x1,x2):ddfxy(i,j,k,bz,ddbzxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x3),x1,x3):ddfxz(i,j,k,bz,ddbzxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x3),x2,x1):ddfxy(i,j,k,bz,ddbzxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x3),x2,x2):ddfyy(i,j,k,bz,ddbzyy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x3),x2,x3):ddfyz(i,j,k,bz,ddbzyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x3),x3,x1):ddfxz(i,j,k,bz,ddbzxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x3),x3,x2):ddfyz(i,j,k,bz,ddbzyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(b( - x3),x3,x3):ddfzz(i,j,k,bz,ddbzzz):g'
mhdeqs r MHDEQS-Dif.red
sed -e
's:@(je( - x1),t):(jex(n+1,i,j,k)-jex(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(je( - x1),x1):dfx(i,j,k,jex,djexx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(je( - x1),x2):dfy(i,j,k,jex,djexy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(je( - x1),x3):dfz(i,j,k,jex,djexz):g'
mhdeqs r MHDEQS-Dif.red
sed -e
's:@(je( - x2),t):(jey(n+1,i,j,k)-jey(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(je( - x2),x1):dfx(i,j,k,jey,djeyx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(je( - x2),x2):dfy(i,j,k,jey,djeyy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(je( - x2),x3):dfz(i,j,k,jey,djeyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e
's:@(je( - x3),t):(jez(n+1,i,j,k)-jez(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(je( - x3),x1):dfx(i,j,k,jez,djezx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(je( - x3),x2):dfy(i,j,k,jez,djezy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(je( - x3),x3):dfz(i,j,k,jez,djezz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:v( - x1):vx(n,i,j,k):g' mhdeqs r MHDEQS-Dif.red
```

```
sed -e 's:v( - x2):vy(n,i,j,k):g' mhdeqs r MHDEQS-Dif.red
sed -e 's:v( - x3):vz(n,i,j,k):g' mhdeqs r MHDEQS-Dif.red
sed -e 's:b( - x1):bx(n,i,j,k):g' mhdeqs r MHDEQS-Dif.red
sed -e 's:b( - x2):by(n,i,j,k):g' mhdeqs r MHDEQS-Dif.red
sed -e 's:b( - x3):bz(n,i,j,k):g' mhdeqs r MHDEQS-Dif.red
sed -e 's:je( - x1):jex(n,i,j,k):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:je( - x2):jey(n,i,j,k):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:je( - x3):jez(n,i,j,k):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(p,t):(p(n+1,i,j,k)-p(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(p,x1):dfx(i,j,k,p,dpx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(p,x2):dfy(i,j,k,p,dpy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(p,x3):dfz(i,j,k,p,dpz):g'
mhdeqs r MHDEQS-Dif.red
sed -e
's:@(pstar,t):(pstar(n+1,i,j,k)-pstar(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(pstar,x1):dfx(i,j,k,pstar,dpstarx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(pstar,x2):dfy(i,j,k,pstar,dpstary):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(pstar,x3):dfz(i,j,k,pstar,dpstarz):g'
mhdeqs r MHDEQS-Dif.red
sed -e
's:@(energy,t):(energy(n+1,i,j,k)-energy(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(energy,x1):dfx(i,j,k,energy,denergyx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(energy,x2):dfy(i,j,k,energy,denergyy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(energy,x3):dfz(i,j,k,energy,denergyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(eta,t):(eta(n+1,i,j,k)-eta(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(eta,x1):dfx(i,j,k,eta,detax):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(eta,x2):dfy(i,j,k,eta,detay):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(eta,x3):dfz(i,j,k,eta,detaz):g'
mhdeqs r MHDEQS-Dif.red
```

```
sed -e 's:@(rho,t):(rho(n+1,i,j,k)-rho(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x1),x1,x1):ddfxx(i,j,k,ax,ddaxxx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x1),x1,x2):ddfxy(i,j,k,ax,ddaxxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x1),x1,x3):ddfxz(i,j,k,ax,ddaxxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x1),x2,x1):ddfxy(i,j,k,ax,ddaxxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x1),x2,x2):ddfyy(i,j,k,ax,ddaxyy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x1),x2,x3):ddfyz(i,j,k,ax,ddaxyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x1),x3,x1):ddfxz(i,j,k,ax,ddaxxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x1),x3,x2):ddfyz(i,j,k,ax,ddaxyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x1),x3,x3):ddfzz(i,j,k,ax,ddaxzz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x2),x1,x1):ddfxx(i,j,k,ay,ddayxx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x2),x1,x2):ddfxy(i,j,k,ay,ddayxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x2),x1,x3):ddfxz(i,j,k,ay,ddayxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x2),x2,x1):ddfxy(i,j,k,ay,ddayxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x2),x2,x2):ddfyy(i,j,k,ay,ddayyy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x2),x2,x3):ddfyz(i,j,k,ay,ddayyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x2),x3,x1):ddfxz(i,j,k,ay,ddayxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x2),x3,x2):ddfyz(i,j,k,ay,ddayyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x2),x3,x3):ddfzz(i,j,k,ay,ddayzz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x3),x1,x1):ddfxx(i,j,k,az,ddazxx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x3),x1,x2):ddfxy(i,j,k,az,ddazxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x3),x1,x3):ddfxz(i,j,k,az,ddazxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x3),x2,x1):ddfxy(i,j,k,az,ddazxy):g'
```

```
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x3),x2,x2):ddfyy(i,j,k,az,ddazyy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x3),x2,x3):ddfyz(i,j,k,az,ddazyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x3),x3,x1):ddfxz(i,j,k,az,ddazxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x3),x3,x2):ddfyz(i,j,k,az,ddazyz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x3),x3,x3):ddfzz(i,j,k,az,ddazzz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x1),t):(ax(n+1,i,j,k)-ax(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x1),x1):dfx(i,j,k,ax,daxx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x1),x2):dfy(i,j,k,ax,daxy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x1),x3):dfz(i,j,k,ax,daxz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x2),t):(ay(n+1,i,j,k)-ay(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x2),x1):dfx(i,j,k,ay,dayx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x2),x2):dfy(i,j,k,ay,dayy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x2),x3):dfz(i,j,k,ay,dayz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x3),t):(az(n+1,i,j,k)-az(n,i,j,k))/dt:g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x3),x1):dfx(i,j,k,az,dazx):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x3),x2):dfy(i,j,k,az,dazy):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:@(a( - x3),x3):dfz(i,j,k,az,dazz):g'
mhdeqs r MHDEQS-Dif.red
sed -e 's:a( - x1):a(n,i,j,k):g' mhdeqs r MHDEQS-Dif.red
sed -e 's:a( - x2):a(n,i,j,k):g' mhdeqs r MHDEQS-Dif.red
sed -e 's:a( - x3):a(n,i,j,k):g' mhdeqs r MHDEQS-Dif.red
```

References

[1] Arfken, G.B.; Weber, H.J.; Harris, F.E. (2013) *Mathematical Methods for Physicists*. Elsevier, Amsterdam.

[2] Birdsall, C.K.; Langdon, A.B. (1991) *Plasma Physics via Computer*. Adam Hilger, Bristol.

[3] Cap, F. (1994) *Lehrbuch der Plasmaphysik und Magnetohydrodynamik*. Springer, Wien.

[4] Carboni-Méndez, R.; Frutos-Alfaro, F. (2005) "Computer simulation of convective plasma cells", *Journal of Atmospheric and Solar-Terrestrial Physics*, **67**(17-18): 1809–1814.

[5] Dendy, R. (1995) *Plasma Physics*. Cambridge University Press, Cambridge.

[6] Germaschewski, K.; Fox, W.; Ahmadi, N.; Wang, L.; Abbott, S.; Ruhl, H.; Bhattacharjee, A. (2013) "The plasma simulation code: A modern particle-in-cell code with load-balancing and GPU support", `http://arxiv.org/abs/1310.7866`

[7] Hassani, S. (2000) *Mathematical Physics. A Modern Introduction to Its Foundations*. Springer, New York.

[8] Hearn, A.C. (2004) *REDUCE* (User's and Contributed Packages Manual). Konrad-Zuse-Zentrum für Informationstechnik, Berlin.

[9] Hockney, R.W.; Eastwood, J.W. (1988) *Computer Simulation Using Particle*. Adam Hilger, Bristol.

[10] Hsu, J.J.Y. (2014) *Visual and Computational Plasma Physics*. World Scientific, Hackensack NJ.

[11] Jackson, J.D. (1975) *Classical Electrodynamics*. Wiley, New York.

[12] Jardin, S. (2010) *Computational Methods in Plasma Physics*. CRC Press, Boca Raton.

[13] Kulikovskiy, A.G.; Lyubimov, G.A. (1965) *Magnetohydrodynamics*. Addison-Wesley, Palo Alto.

[14] Pen, U.L.; Arras, P.; Wong, S.K. (2003) "A free, fast, simple, and efficient total variation diminishing magnetohydrodynamic code", *The Astrophysical Journal Supplement Series* **149**: 447–455.

[15] Persson, P.O.; Strang, G. (2004) "A simple mesh generator in Matlab", *SIAM Review* **46**(2): 329–345.

[16] Schrüfer, E. (2004) "EXCALC: A System for Doing Calculations in the Calculus of Modern Differential Geometry", in A.C. Hearn (Ed.) *REDUCE*, (see reference).

[17] Tajima, T. (2004) *Computational Plasma Physics*. Westview Press, Boulder.

[18] Tajima, T.; Shibata, K. (2002) *Plasma Astrophysics*. Westview Press, Boulder.

[19] `http://unixhelp.ed.ac.uk/alphabetical/`