

第5章 実習の手引き (差分法の実習、磁気流体基本課題)

5.1 スカラー方程式の差分解法

スカラー方程式の差分解法パッケージの説明

スカラー方程式の差分解法を試してみましょう。以下のファイルが用意されています。

```
# ls scalar
Makefile  anime.pro  main.f      pldt.pro    pldtps.pro  rddt.pro
```

プログラムは Fortran 言語を用いて書かれています。上記のリストで main.f が Fortran プログラムファイルです。

5.1.1 プログラムのコンパイルと実行 (make)

次にプログラムをコンパイルする方法について説明します。ディレクトリ “scalar” に移動した後に **make** を実行します。するとプログラムがコンパイルされ、実行されます。正しくコンパイルされるとオブジェクトファイル **main.o** と実行オブジェクトファイル **a.out** を作成します。正しく実行されるとデータファイル **out.dat** を出力します。

```
# cd scalar
# make
gfortran -c -o main.o main.f
main.f:
  MAIN:
gfortran -o a.out main.o
./a.out
  write      step=          0 time= 0.000E+00
  write      step=         50 time= 0.125E+02
  write      step=        100 time= 0.250E+02
  ### normal stop ###
# ls
Makefile      anime.pro    main.o        pldt.pro      rddt.pro
a.out*        main.f      out.dat       pldtps.pro
```

5.1.2 出力ファイルの説明 (out.dat)

出力ファイル `out.dat` はデータの確認を容易にするためにアスキー形式でかかれており、ファイルを直接エディタで開いて見ることができます。ファイルをみてみましょう。第1行目に配列の大きさ (`jx`) と時間データの数 (`nx`) がそれぞれ書かれています。第2行目に始めの時間データの `time step` 数 (`ns`)、時刻 (`time`) が書かれています。第3行目から第102行目に渡って、始めの時間データの x 座標 (`x`)、そこでの変数の値 (`u`) が順にかかれています。第103行目移行に、次の時間データが書かれています。書式については、Fortran プログラム `main.f` の 53, 55, 59 行目に `Format` 文で指定されていますので、参考にして下さい。

```
# head out.dat
100,    3
    0,  0.00
1.0, 1.0000000
2.0, 1.0000000
3.0, 1.0000000
(後略)
```

5.1.3 結果の可視化表示

結果の表示には IDL といった可視化プログラムを利用します。IDL は数値シミュレーション結果を可視化をするのによく用いられます。

IDL の起動 (idl)

まずは idl を実行してみましょう。

```
# idl
```

すると以下のようになり、IDL が起動します。

```
IDL Version 8.6. (c) 2011, Research Systems, Inc.  
Installation number: XXXXX.  
Licensed for use by: XXXXX  
  
IDL>
```

データ読み込み (.r rddt)

データの読み込みには `rddt.pro` というプログラムを用います。以下のように入力してみてください。ファイル `out.dat` からデータが読み込まれ、idl でデータを利用できるようになります。

```
IDL> .r rddt
```

`.r` は run を意味します。

データ表示 (.r pldt)

データの表示には `pldt.pro` というプログラムを用います。以下のように入力してみてください。

```
IDL> .r pldt
```

IDL の終了 (exit)

`exit` を入力すると IDL を終了することができます。

```
IDL> exit
```

5.1.4 プログラムの変更について

計算エンジンの変更

サンプルプログラムの計算エンジンは FTCS になっています。以下の 72 行目から 88 行目の間で、計算エンジンを変更してください。

メッシュ数の設定 (jx)

メッシュ数を変更するには、5 行目の `parameter` 文にある `jx` の値を変更します。メッシュ数をかえることで数値計算の分解能をあげることができます。

```
parameter (jx=100)
```

最終ステップ数、出力の設定 (nstop, nskip)

最終ステップ数と出力ファイルの間隔は、14 行目の `nstop` と 15 行目の `nskip` の値をそれぞれ変更します。ファイルの出力間隔を短くすることで、アニメーションを滑らかに表示することができます。その一方、出力されるファイルのサイズは大きくなります。

```
c    time control parameters

      nstop=100
      nskip = 50
```

CFL 条件の変更 (safety)

CFL 条件は、68 行目の (`safety`) の値を変更します。出力されるデータファイルの時間間隔は、現在、`nskip` で制御しているため、`safety` の値を変更すると、出力される時間も変わることにご注意ください。

```
c    obtain time spacing
      safety=0.25
```

5.1.5 データのアニメーション表示 (.r anime)

データのアニメーション表示には `anime.pro` というプログラムを用います。IDL でデータを読み込んだ後に、以下のように入力してみてください。

```
IDL> .r anime
```

デフォルトのままでは、データのステップ間隔が大きく、きれいにみれません。`nskip` を 1 にしてどのように進化するか見てみましょう。`anime.pro` の window が開いている状態で、もう一度、`anime.pro` を実行するとエラーが出ますのでご注意ください。

付録

サンプルプログラム、main.f

```
c=====|
c   array definitions
c=====|
c   implicit real*8 (a-h,o-z)
c   parameter (jx=100)
c
c   dimension x(1:jx),u(1:jx),f(1:jx)
c
c=====|
c   prologue
c=====|
c   time control parameters
c
c   nstop=100
c   nskip = 50
c
c-----|
c   initialize counters
c
c   time  = 0.0
c   ns    = 0
c
c   nx = nstop/nskip+1
c-----|
c   Set initial condition
c-----|
c   pi=4.*atan(1.0)
c   grid
c   dx=1.0
c   x(1)=dx
c   do j=1,jx-1
c     x(j+1)=x(j)+dx
c   enddo
```

```

c
c  variable
    do j=1,jx/2
        u(j)= 1.0
    enddo
    do j=jx/2+1,jx
        u(j)= 0.0
    enddo
c
c  velocity
    cs=1.0

c-----|
c    Output initial condition
c
    write(6,103) ns,time
103  format (1x,' write      ','step=',i8,' time=',e10.3)
    open(unit=10,file='out.dat',form='formatted')
    write(10,100) jx,nx
100  format(i5,',',i5)
    write(10,101) ns,time
101  format (i5,',',f6.2)
    do j=1,jx
        write(10,102) x(j),u(j)
    enddo
102  format(f5.1,',',f10.7)

c=====|
c    time integration
c=====|
1000  continue
    ns = ns+1
c-----|
c    obtain time spacing
    safety=0.25
    dt=safety*dx/cs
    time=time+dt

```

```

c-----|
c      solve equation
c
c
c                                     ftcs - start >>>
c
c      do j=1,jx-1
c          f(j)=0.5*cs*(u(j+1)+u(j))
c      enddo
c
c      f(jx)=f(jx-1)
c
c      do j=2,jx-1
c          u(j)=u(j)-dt/dx*(f(j)-f(j-1))
c      enddo
c
c      u(1)=u(2)
c      u(jx)=u(jx-1)
c
c                                     ftcs - end   >>>
c-----|
c      data output
c      if (mod(ns,nskip).eq.0) then
c          write(6,103) ns,time
c          write(10,101) ns,time
c          do j=1,jx
c              write(10,102) x(j),u(j)
c          enddo
c      endif
c
c      if (ns .lt. nstop) goto 1000
c      close(10)
c
c=====|
c      write(6,*) '   ### normal stop   ###'
c      end

```

サンプルプログラム、rddt.pro

```
; rddt.pro
openr,1,'out.dat'
readf,1,jx,nx

; define array
ns=intarr(nx)
t=fltarr(nx)

x=fltarr(jx)
u=fltarr(jx,nx)

; temporary variables for read data
ns_and_t=fltarr(2,1)
x_and_u=fltarr(2,jx)

for n=0,nx-1 do begin
  readf,1,ns_and_t
  readf,1,x_and_u
  ns(n)=fix(ns_and_t(0,0))
  t(n)=ns_and_t(1,0)
  u(*,n)=x_and_u(1,*)
endfor

close,1
free_lun,1

x(*)=x_and_u(0,*)

delvar,ns_and_t,x_and_u

help
end
```

サンプルプログラム、pldt.pro

```
!x.style=1
!y.style=1
!p.charsize=1.4

plot,x,u(*,0),xtitle='x',ytitle='u',linest=1,yrange=[-1,3],xrange=[0,100]
for n=1,nx-1 do begin
  oplot,x,u(*,n)
  oplot,x,u(*,n),psym=4
endfor

end
```

サンプルプログラム、anime.pro

```
!x.style=1
!y.style=1
!p.charsize=1.4

window,xsize=480,ysize=480
xinteranimate,set=[480,480,nx]

for n=0,nx-1 do begin

  plot,x,u(*,n),xtitle='x',ytitle='u',yrange=[-1,3],xrange=[0,100]
  oplot,x,u(*,n),psym=4

  xinteranimate,frame=n>window=0

endfor

xinteranimate

end
```

5.2 差分法の実習課題

5.2.1 1次元波動方程式

スカラー方程式の差分解法のパッケージを動かさない。サンプルプログラムは、FTCS スキームを用いて、1次元波動方程式を計算するものである。初期条件は、 $j = 1, \dots, 50$ に対して $u_j = 1$ 、 $j = 51, \dots, 100$ に対して $u_j = 0$ 、クーラン数 $\nu = c\Delta t/\Delta x = 0.25$ として計算をおこなう。このプログラムを

1. Lax-Wendroff 法によるもの、
2. 空間1次精度の風上差分によるもの、
3. 流束制限関数として minmod 関数を用いたもの

に書きかえ、計算結果をグラフ表示し、テキストの結果と比較しなさい。

補足解説

1次元波動方程式

$$\frac{\partial u}{\partial t} + c \frac{\partial u}{\partial x} = 0 \quad (5.1)$$

を差分化すると

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{\Delta x} (f_{j+1/2}^n - f_{j-1/2}^n) \quad (5.2)$$

となる。ここで、FTCS スキームの数値流束を用いると

$$f_{j+1/2}^n = \frac{1}{2} (f_{j+1} + f_j) = \frac{1}{2} c (u_{j+1} + u_j) \quad (5.3)$$

となる。

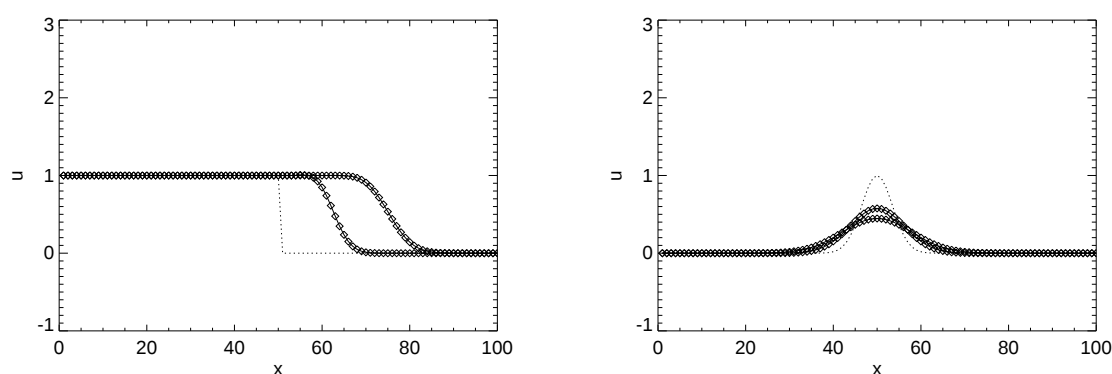


図 5.2: 計算結果。左：Lax-Wendroff 法+流束制限関数。右：1次元拡散方程式。

5.2.2 Burgers 方程式

1次元波動方程式のプログラムを参考にして Burgers 方程式

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} \left(\frac{u^2}{2} \right) = 0 \quad (5.4)$$

を1次精度風上差分法を用いて解くプログラムを作成し、結果をグラフ表示し、テキストの結果と比較しなさい。数値流束は

$$f_{j+1/2}^n = \frac{1}{2} \left\{ \left(\frac{u_{j+1}^2}{2} + \frac{u_j^2}{2} \right) - \frac{1}{2} |u_{j+1} + u_j| (u_{j+1} - u_j) \right\} \quad (5.5)$$

と書くこともできる。

5.2.3 1次元拡散方程式

1次元波動方程式のプログラムを参考にして 1次元拡散方程式

$$\frac{\partial u}{\partial t} = \kappa \frac{\partial^2 u}{\partial x^2} \quad (5.6)$$

を FTCS スキームを用いて解くプログラムを作成し、適当な初期条件を設定してシミュレーションを行い、結果をグラフ表示しなさい。

例えば、初期の分布にはガウス分布を仮定し、`cs` の代わりに `kappa` を定義しましょう。

```
c variable
  do j=1,jx
    u(j)= exp(-(((x(j)-x(jx/2))/5.))**2))
  enddo
c
c kappa
  kappa=1.0
```

5.3 CANS 基本課題

衝撃波管問題

CANS の 1 次元パッケージを使ってみなさい。

1. 基本課題「等温衝撃波管 (md_itshktb)」を実行し、IDL で rddt.pro と pldt.pro を用いて可視化せよ。
2. 基本課題「流体衝撃波管 (md_shktb)」を実行し、可視化せよ。
3. 基本課題「衝撃波生成 (md_shkform)」を実行し、anime.pro を用いて可視化せよ。
4. 基本課題「MHD 衝撃波管 (md_mhdshktb)」を実行し、可視化せよ。

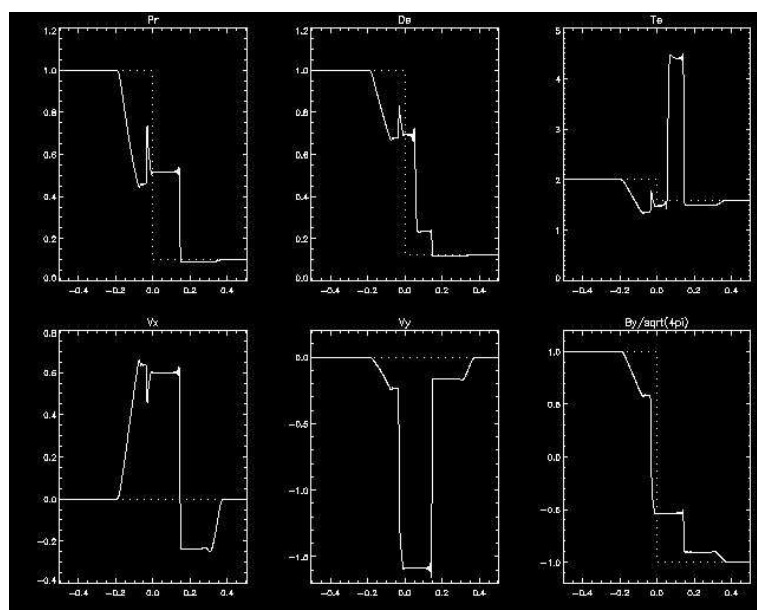


図 5.3: md_mhdshktb の結果

補足

- 基本課題を動かした後に Fortran プログラムを変更し、**make** すると、再コンパイルし、計算を実行します。この際、出力ファイル `params.txt` や `***.dac` を上書きします。必要な出力ファイルは、名前を変更したり、別ディレクトリに移してとっておきましょう。
- オブジェクトファイル、出力ファイルを消去したい場合は、**make clean** を実行してください。

重力成層流体中を伝わる磁気流体波動

太陽表面大気的光球・彩層では磁気流体波に、さまざまなおもしろい現象がおこる。重力的に成層しているに加えて、ちょうどガス圧優勢から磁気圧優勢に遷移することから、周波数によるカットオ

フ、突っ立ちによる衝撃波形成やモード変換などを起こす。またこうして伝わった波のエネルギーが、高密プラズマを押し上げることで、スピキュールと呼ばれるジェット現象を引き起こす。そして外部に伝わった波はやがて破碎・熱化してコロナ加熱に寄与するとも考えられている。

本演習では、2次元で太陽表面大気中を伝わる磁気流体波についてのシミュレーションを行う。まずは、一様大気を伝わる波動から始め、成層や一様磁場、さらには非一様磁場など一歩ずつ拡張して物理に対する理解を深めながら進める。

方程式系・計算モデル

2次元デカルト座標系で、3成分磁場が入った磁気流体方程式を解く。磁気拡散・熱伝導・放射冷却を考慮しない（演習の進み具合によってはこれらの効果も導入する）。波発生のために、運動方程式に加速源をソース項として加える。空間局所的（Gauss 関数）、時間周期的（正弦関数）。圧縮性波動・シア（Alfven 的）波動をそれぞれ試す。背景として、以下のような、大気・磁場構造を順に考える。以下で $\beta = p/p_m$ は、 p はガス圧、 p_m は磁気圧。

- (1) 一様背景中の MHD 波動（CANS のまま）
- (2) 重力成層・一様温度・一様磁場（ $\beta > 1$ ）中の MHD 波動（まずは、波動の突っ立ち）周波数を変えて、エヴァネッセントや重力波のケースも実施。
- (3) 同上（ $\beta < 1$ ）
- (4) 同上（下半分 $\beta > 1$ 、上半分 $\beta < 1$ ）モードコンバージョンがみえるか？
- (5) 重力無・膨らんだ磁力線（ $\beta > 1$ ）
- (6) 同上（ $\beta < 1$ ）
- (7) 同上（下半分 $\beta > 1$ 、上半分 $\beta < 1$ ）モードコンバージョンがみえるか？
- (8) 重力成層・膨らんだ磁力線
- (9) …

参考とする先行研究

重力的に成層した大気中（ただしコロナ含まず）を伝わる線形磁気流体波動の伝播については Bogdan et al. (2003) が2次元シミュレーションを実施している。波が $\beta = 1$ 境界面を通過する際のモード変換について詳しく調べている。いっぽう、波動注入によるスピキュール形成については、Kudoh & Shibata (1999) による1次元シミュレーションがある。

参考文献

- Bogdan et al. 2003, ApJ, 599, 626
 Kudoh et al. 1999 ApJ, 514, 493