

GIZMO で遊んでみる

岡本 崇 (北海道大学)

December 23, 2019

GIZMO Users Guide

- 以下の URL

▶ http://www.tapir.caltech.edu/~phopkins/Site/GIZMO_files/gizmo_documentation.html

- 実習するテスト問題用の初期条件やパラメータファイルもここから入手可能
- `/home/hydro00/GIZMO_hydro2019.tar.gz` を解凍してできる
`GIZMO_hydro2019/scripts` にも演習用の初期条件やパラメータファイルが入っている
- コードにも同じ **User Guide** が付属 (`scripts` ディレクトリの中)
- `scripts/test_problems/` の中にテスト問題のパラメータファイルがある

準備と確認 (1)

- 実習は Intel 環境で行います. XC50 のコマンドラインで

`module list`

を実行し,

- ▶ `PrgEnv-intel/6.0.4`
- ▶ `gsl/2.40_intel-18.0`
- ▶ `cray-hdf5/1.10.1.1`
- ▶ `fftw/2.1.5.9`

が表示されることを確認.

- 表示されないものがあつた場合はそれを

`module load (or add)`

しておく.

準備と確認 (2)

- `gizmo` のディレクトトに入り, `Config.sh` が存在することを確認.
- 無ければ `cp Template-Config.sh Config.sh` する.
- 好みの `editor` で (`vim` のことです) `Makefile.systype` を開き,
`SYSTYPE="XC-intel"` がコメントアウトされていないことを確認

準備と確認（この実習限定の話）

- 必要な module は自動的に load されているはずです.
- 各自の work 領域 (/work/hydro[01-20]) に /home/hydro00/GIZMO_hydro2019.tar.gz をコピーして展開します.
- 解凍してできた GIZMO_hydro2019/scripts/test_problems/ の中に今回用いるテスト問題が入っています.

実習 1: 点源爆発

- 冷たい (10 K) 一様密度 $n = 0.5 \text{ cm}^{-3}$ のガス分布の中心に $6.78 \times 10^{46} \text{ erg}$ のエネルギーを注入。
→ $M \sim 1000$ の衝撃波と Users Guide には書いてありますが, 解析してみた結果, $n = 5 \times 10^9 \text{ cm}^{-3}$, $E = 6.78 \times 10^{56} \text{ J}$. なんだこれ.

Config.sh

```
BOX_PERIODIC
```

```
SELFGRAVITY_OFF
```

```
EOS_GAMMA=(5.0/3.0)
```

以外をコメントアウト. 特に指定しない場合流体法は MFM (HYDRO_MESHLESS_FINITE_MASS) が選ばれる.

- make する. Config.sh を変更した場合は必ず make し直すこと.

実習 1: 点源爆発-初期条件とパラメータファイル

- 実行ディレクトリに移動. 今回の場合, GIZMO_hydro2019/scripts/test_problems/sedov に全部入ってます.
- 実行ファイル GIZMO を実行ディレクトリにコピー (今回はコピーしなくて良い)
- sedov.param の編集 (そのままでも動きます)

sedov.param

```
MaxSizeTimestep 0.0001
```

を

```
MaxSizeTimestep 0.03
```

に変更. そのままだと最大のタイムステップが小さすぎて独立時間刻みにならない.

実習 1: 点源爆発-実行

- 1 ノードを使って MPI 並列 (40 並列) で実行
- 以下の話は一般論なので皆さんは
GIZMO_hydro2019/scripts/test_problems/sedov の下にある batch.sh を使っ
て下さい.
- ファイル名を仮に batch.sh としておく

```
batch.sh
```

```
#PBS -N sedov
```

```
#PBS -l nodes=1
```

```
#PBS -q test-bp ← 自分の使うキュー名
```

```
cd ${PBS_O_WORKDIR}
```

```
aprun -n 40 ./GIZMO sedov.params >& log
```

実習 1: 点源爆発-実行 (続き)

- 実習用 batch.sh の編集

batch.sh

aprun -n 40 ../../GIZMO sedov.params を
aprun -n 40 ../../GIZMO sedov.params >& log に変更
後で tail -f log で進捗を見ることができて便利

- qsub batch.sh でジョブをサブミット
- 実行されると OutputDir で指定したディレクトリに (今回は ./output) snapshto_???.hdf5 が出力される (000 から 011 まで 011 はゴミ).

実習 1: 点源爆発-可視化

- とりあえずの可視化には yt が手軽

▶ 詳しくはここ

- 解析サーバーにログイン
- /work/hydro?*/ 以下に適当なディレクトリを作成
- そのディレクトリで ipython を起動

```
ipython (or projection.py)
```

```
import yt
```

```
ds = yt.load('/xc-work/okamtotk/hydro/tests/sedov/snapshot_010.hdf5') ← 読み込む
```

```
snapshot
```

```
plot = yt.ProjectionPlot(ds, "z", fields=[('gas','density')])
```

```
plot.save()
```

- 以上で密度場の projection plot ができる。
display snapshot_011_Projection_z_density.png で表示できる。
- plot = yt.ProjectionPlot(ds, 'z', fields=[('gas','temperature')]), weight_field=[('gas','density')] とすると密度で重みをつけた温度の projection map が得られる。

実習 1: 点源爆発—解析

- 物理量の半径依存性を見てみたい
- ちょっと凝った解析をしようとする `yt` では大変
- 直接 `python` で

```
ipython (or sedov_density.py)
```

```
import h5py
import numpy as np

ds = h5py.File('/xc-work/okamtotk/hydro/tests/sedov/snapshot_010.hdf5', 'r')
# ここでデータの構造をちょっと見てみる

list(ds.keys())

# Header と PartType0 が表示されたはず. さらに PartType0 (gas) の中にどんな情報があるかは
list(ds['PartType0'].keys()) # で確認できる. Header の中身は
list(ds['Header'].attrs)

# simulation box の大きさは例えば
boxlen = ds['Header'].attrs['BoxSize']
```

実習 1: 点源爆発–matplotlib

- データ構造がわかったところで plot を作ってみる

```
ipython (or sedov_density.py)
import matplotlib.pyplot as plt
c = np.full(3, 0.5*boxlen) #中心の座標
pos = np.array(ds['PartType0/Coordinates'])
r = np.sqrt( (pos[:,0] - c[0])**2 + (pos[:,1] - c[1])**2 + (pos[:,2] - c[2])**2)
# ↑多分もっと賢いやり方がある知らない
plt.plot(r, ds['PartType0/Density'], ',', rasterized=True)
plt.savefig('density.png')
```

- exit で ipython を抜けて図を確認。

実習 2: SPH との比較

- PSPH (Saito & Makino の DISPH とほぼ同じ) にしてみる
- Config.sh の編集
HYDRO_PRESSURE_SPH のコメントアウトを外す
デフォルトで Cullen & Dehnen 2010 の人工粘性や人工熱伝導が入る
- make
- GIZMO を実行ディレクトリにコピー
- 実行ディレクトリに移動し, `cp sedov.params sedov_pspsh.params` と新しいパラメータファイルを作る
- `sedov_pspsh.params` を編集して `SnapshotFileBase snapshot` を `snapshot_pspsh` に
- バッチスクリプト `run.sh` を編集し `sedov.params` を `sedov_pspsh.params` に
- `qsub run.sh`

実習 2: 結果の解析と作図

- 解析サーバーにログイン
- 作業を行うディレクトリに移動
- 作図に必要なファイル `sedov.txt` と `sedov.py` を `/home/okamtotk/hydro` から入手
- `sedov.py` のファイルを読み込んでいる行を適切に編集
- `python sedov.py`
- `sedov.png` というファイルが出来るので表示
- 皆, 密度しか見せないが圧力の振る舞いの方がシビアなことが分かる

注意: 単位系について

- シミュレーションで使われる単位系はパラメータファイル内の
 - ▶ UnitLength_in_cm
 - ▶ UnitMass_in_g
 - ▶ UnitVelocity_in_cm_per_s

によって指定される. それぞれ $3.08568e+21$, $1.989e+43$, 100000 になっていた場合, kpc, $10^{10} M_{\odot}$, km/s が code units であることが分かる. その他の物理量の単位はこれらの組み合わせから計算できる.

実習 3: その他のテスト問題

- Config.sh の HYDRO_PRESSURE_SPH をコメントアウトし直すことを忘れないように
- おすすめのテスト問題は
 - ▶ The Noh (Spherical Implosion) Problem (noh.params)
BOX_PERIODIC
SELFGRAVITY_OFF
EOS_GAMMA=(5.0/3.0)
 - ▶ Kelvin Helmholtz Instabilities (kh_mcnally_2d.params)
BOX_PERIODIC
BOX_SPATIAL_DIMENSION=2
PREVENT_PARTICLE_MERGE_SPLIT
SELFGRAVITY_OFF
KERNEL_FUNCTION=3
EOS_GAMMA=(5.0/3.0)

実習 3: その他のテスト問題 (続き)

- The Blob Test (blob.params)

BOX_PERIODIC

BOX_LONG_X=1

BOX_LONG_Y=1

BOX_LONG_Z=3

SELFGRAVITY_OFF

EOS_GAMMA=(5.0/3.0)

- The Evrard (Spherical Collapse) Test (evrard.params)

EOS_GAMMA=(5.0/3.0)

- ▶ ADAPTIVE_GRAVSOFT_FORGAS をアクティブにし、blob.params の SofteningGas と SofteningGasMaxPhys の値をずっと小さくして結果を比較してみましょう
- ▶ 余裕があれば SPH との比較も

実習 4: SPH 近似の確認

- 適当な 3 次元の問題のスナップショットを使って、SPH 近似で

$$\sum_j \frac{m_j}{\rho_j} W(r_{ij}, h_i)$$

がどのくらい 1 からずれるか、いくつかの i 粒子に対して計算してみる

- ここで $W(r_{ij}, h_i)$ は

$$W(r_{ij}, h_i) = \begin{cases} \frac{8}{\pi h_i^3} \left\{ 1.0 + 6.0(u - 1.0)u^2 \right\} & \text{for } u = r_{ij}/h_i \leq 0.5 \\ \frac{8}{\pi h_i^3} \left\{ 2.0(1.0 - u)^3 \right\} & \text{for } 0.5 \leq u \leq 1.0 \\ 0 & \text{for } u > 1.0 \end{cases}$$

- また、 h_i は SmoothingLength という名前でスナップショットに入っている

実習 5: 初期条件の作成

- SPH や Mesh-free 法では初期条件の作成が結構面倒
- これは粒子分布によって密度が与えられるから.
- GIZMO_hydro2019/scripts/make_IC.py にサンプルがあるので python に慣れてる人は挑戦してみてください.
 - ① 流体要素を grid 上に置かずに、一様密度場を作るにはどうすればいいか?
 - ② 中心からの距離 r に対して $\rho r \propto r^{-2}$ の密度構造をもつ等温球を作るには?