iSALE 講習会
配布テキスト
-実践編-

iSALE 講習会実行委員会

講師一覧

2023 年度

黒澤 耕介	千葉工業大学 惑星探査研究センター
黒崎 健二	神戸大学
松本 侑士	国立天文台天文シミュレーションプロジェクト
伊藤 孝士	国立天文台天文シミュレーションプロジェクト

2022 年度

黒澤 耕介	千葉工業大学 惑星探査研究センター
嶌生 有理	宇宙航空研究開発機構
伊藤 孝士	国立天文台天文シミュレーションプロジェクト

2021 年度

黒澤 耕介	千葉工業大学 惑星探査研究センター
末次 竜	大島商船高等専門学校
嶌生 有理	宇宙航空研究開発機構
伊藤 孝士	国立天文台天文シミュレーションプロジェクト

2020 年度

黒澤 耕介	千葉工業大学 惑星探査研究センター
末次 竜	大島商船高等専門学校
脇田 茂	Purdue University
伊藤 孝士	国立天文台天文シミュレーションプロジェクト

編集履歴

2020 5/30	第 1 版作成	黒澤耕介
2020 7/13	2020 年度版確定	黒澤耕介
2021 5/27	2021 年度版作成	黒澤耕介
2021 6/2	目次の自動生成	嶌生有理
2021 9/9	2021 年度版確定	黒澤耕介
2022 6/4	2022 年度版作成	黒澤耕介
2022 8/29	2021 年度版確定	黒澤耕介
2023 5/29	2023 年度版作成	黒澤耕介

まえがき

本テキストは iSALE 講習会において参加者本人が iSALE 計算を実践するための手引として作成されたものである．座学講義編と同様に iSALE-Dellen manual [Collins et al., 2016]を補足するという指針で作成されているので, iSALE-Dellen manual と一緒に読み進めることをおすすめする．前半は iSALE の入力ファイルの内容と基本操作方法について解説する．後半は今年度の講習会の課題として設定した，いくつかの課題について解説する．これらの課題をこなせるようになれば，自身で iSALE を用いた研究を進めていくことができるようになるであろう．大部分の内容は 2020–2022 年度テキストと同様であるが，よりわかりやすい記述になるように微修正している．誤植など発見された場合は(isale-developers-jp@perc.it-chiba.ac.jp)まで連絡をいただけると幸いである．

2023 年 5 月 31 日

黒澤 耕介

目次

1. 国立天文台 CfCA の共同利用計算機への接続方法

まずは自然科学研究機構国立天文台天文シミュレーションプロジェクト(Center for Computational Astrophysics, 以下では CfCA と略記する)の共同利用計算機にログインしよう．今回の講習会では「`grd0.cfca.nao.ac.jp`」という計算サーバと「`an00.cfca.nao.ac.jp`」，「`an01.cfca.nao.ac.jp`」という解析サーバを使用する(これらはいずれも今回の講習会向けに準備された専用の機材であり，通常の利用者はログインできないことに注意されたい.)．国立天文台 CfCA の共同利用計算機の利用には CfCA が運用する「HPC ネットワーク」に VPN (Virtual Private Network, ただし SSL-VPN のみ)を用いて接続する必要がある．まずは VPN 接続のためのソフトウェア「Cisco Anyconnect Secure Mobility Client」を整備しよう．各環境(Mac OS X, Windows, Linux)における詳細手順は

https://www.cfca.nao.ac.jp/node/78

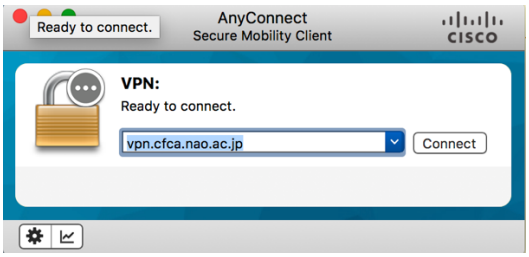
に記述されている．上記のページに記された内容と重なる部分が多いが，以下では本講習会の受講者向けに改めて接続方法をまとめる．この先の作業では以下 2 種類のパスワードが必要になるので，手元に準備しておこう．

1. VPN 接続用パスワード
   CfCA のウェブアカウントにログインした際に現れるページに表示されている．なおこの VPN 接続用パスワードは固定されており，受講者を含むユーザはこれを変更できない．

2. NIS の初期パスワード

CfCA の機材運用担当者から届いた Slack のダイレクトメッセージまたは電子メールに記載されているパスワードである. VPN 接続の確立後、実際に iSALE を動かす機材(`grd0, an00, an01`)に ssh ログインする際に必要となる。なおこれは初回ログイン時に必ず変更すること（後述）.

以下では macOS 上の実行例を示すが, 他の OS でも同様である. まず自分の PC 上で Cisco Anyconnect Secure Mobility Client を立ち上げると以下のウィンドウが現れるだろう.



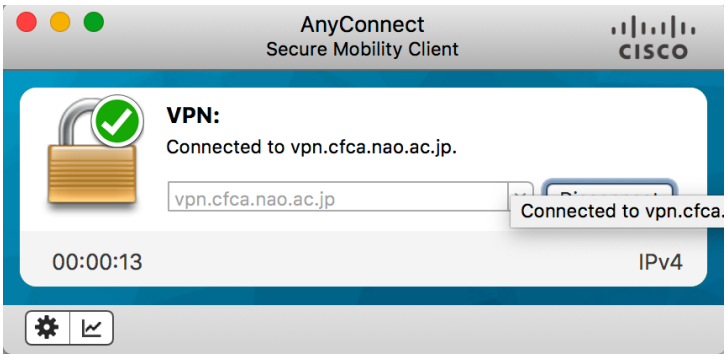
ここには CfCA の VPN サーバ（入り口）となる機材の名前

`vpn.cfca.nao.ac.jp`

を入力し,「Connect」をクリックする. するともう一つのウィンドウが現れる。ここでは Username と Password の入力を求められる。

- Username には各人に配布されたアカウント名 isaleXX (XX は 2 桁の数字. `isale01, isale02, ..., isale22` など)を入力
- Password には各人に配布された VPN パスワード（12 桁の文字列）を入力

Username と Password を入力したら、OK をクリックする。そして下記する画面に変われば，国立天文台 CfCA への VPN 接続は確立したことになる.



VPN 接続が確立したら，引き続いて今回の講習で用いる計算サーバ「grd0」にログインする．まずは何らかの端末アプリを立ち上げる。macOS であれば「アプリケーション」→「その他」→「ターミナル」などである。その上で,

```
$ ssh -CY isale00@grd0.cfca.nao.ac.jp
```

を入力する[1]．上記の例にあるアカウント名 isale00 は各自の ID で置き換えること．すると，

```
$ isale00@grd0.cfca.nao.ac.jp's password:
```

とパスワードを求められるので，各人へ Slack のダイレクトメッセージまたは電子メールで届いた NIS の初期パスワードを入力する．

```
Last login: Fri Jun 4 12:25:48 2021 from an00.cfca.nao.ac.jp
--- Maintenance schedule (date/time in JST)
--- XC50, Analysis servers, fileservers, GRAPE and GPU:
---             2021/06/07 09:00 --> 17:00.
--- General-purpose PCs will remain in operation
--- Web service:
---             2021/06/14 09:00 --> 2021/06/15 17:00.
[isale00@grd0 ~]$
```

上記のように表示されれば grd0 へのログインは成功している．初回ログイン時には

```
$ yppasswd
```

と入力し，与えられた初期の NIS パスワードを直ちに変更すること[2]．

続いて図の描画に必要な環境が整っていることを確認しよう．gnuplot を load し簡単な図を描画してみる．なおこのためには各自の PC が X11 に対応した環境を備えていることが必須である（macOS なら XQuartz, Windows なら MobaXterm, Cygwin/X など）．

---

[1] $はコマンドプロンプトのつもりです。この後の文字列(ssh ...以降)を入力してください．以下同様です．なおオプション -CY の意味についてはご自分で man ssh などを行なって調べてください． -C -Y と書いても同じ結果となります．

[2] 言うまでも無いですが，パスワード管理は個人で責任を持ってお願いします．

```
[isale00@grd0 ~]$ module load gnuplot
[isale00@grd0 ~]$ gnuplot
```

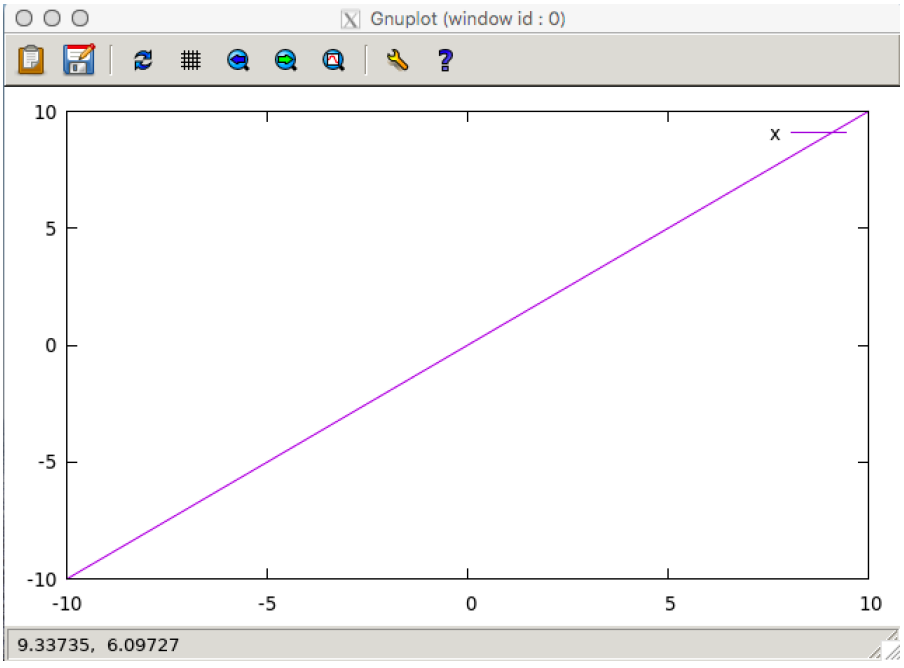
G N U P L O T
Version 5.2 patchlevel 5 last modified 2018-10-06

Copyright (C) 1986-1993, 1998, 2004, 2007-2018
Thomas Williams, Colin Kelley and many others

gnuplot home: http://www.gnuplot.info
faq, bugs, etc: type "help FAQ"
immediate help: type "help" (plot window: hit 'h')

Terminal type is now 'wxt'

```
gnuplot> plot x
```



このような画面が表示されれば, X11 に対応した描画環境が整っている.

以上で，計算サーバへのログインが完了し，iSALE 計算を行う準備が完了した．

なお，各利用者がホームディレクトリ上で可能なディスク容量には上限(1 TB)がある．リミット超過しそうな段階で発出される警告メールを受け取るため，/home/iSALEXX/ 下に .forward ファイルを作成し，自身のメールアドレスを記載しておこう．好みのエディタを開きメールアドレスを入力し，「.forward」という名前で保存すれば OK である．

emacs の場合:

```
$ emacs -nw .forward
```

vi の場合:

```
$ vi .forward
```

CfCA の共同利用計算機のディスク環境に関する詳しい説明は以下を参照．
https://www.cfca.nao.ac.jp/node/22#disk

2. iSALE 関連ファイル，実行方法，サーバ利用規則

本章でははじめにログインに成功したら iSALE 関連ファイルを準備する手順を説明する．ログインしたディレクトリで「ls」コマンドを打ってみても初期状態ではファイルは何も置かれていない．iSALE 関連ファイルは ~isale00/school2023/の中に置かれている．まずは iSALE 関連ファイルを現在のディレクトリに展開しよう．なおファイルの展開先としては各人のホームディレクトリ，もしくは計算サーバの作業領域(/mwork1, /mwork2 など)という選択肢がある．後者の方がより大きな容量があり，読み書きも高速に出来るので，推奨される方法である（但し作業領域上では最終アクセス日時から 120 日間を過ぎたファイルは逐次削除される．詳細は計算サーバの利用方法ウェブページを参照のこと）．以下ではまずホームディレクトリ上にファイルを展開する方法を示し(2.1)、次に作業領域上にファイルを展開する方法を示す(2.2)．講習会は後者の方法で進める．

2.1. ホームディレクトリ上でのファイルの展開

コマンドラインに以下を入力する．

```
$ cd
$ tar xvpf ~isale00/school2023/isale-work-2023.tar.gz
$ chgrp -R isale ./isale-work
```

すると，~isale00/school2023/に置かれているisale-work-2023.tar.gzが解凍される．~1秒ほどで完了するはずである．chgrpコマンドで上記ファイルをグループ「isale」に属するユーザのみが閲覧できるようにしている．

```
$ ls
```
isale-work

となればOKである．ディレクトリisale-work中にiSALE関連ファイルが納められている．これはiSALE [Amsden et al., 1980; Ivanov et al., 1997; Wünnemann et al., 2006]の最新版[3]である「iSALE-Dellen」の実行ファイルと入力ファイルをまとめたものである．***なお，isale-work.tar.gz及びその中身の2次配布は禁止である[4]．*** 続いて作られたisale-workのアクセス権限を設定する．

```
$ chmod 750 isale-work
```

とすることで，自分とグループ「isale」のメンバがアクセスできる設定となる．

2.2. 作業領域上でのファイルの展開（推奨）

2023 年 6 月現在，計算サーバには二つの作業領域がある（/mwork1 と /mwork2）.このうち/mwork2 がより新しい機材上にあり，読み書きの速度も速いので，以下ではここを使う例を示す。/mwork1 を使う場合も同様である．

今回の講習会に際して受講者向けアカウント(isaleXX)向けには作業領域 /mwork2 の下に既にサブディレクトリ(/mwork2/isaleXX)が作成されている

[3] Stable release の最新版です．開発版ではありません．
[4] 計算結果はその限りではありません．

ので、まずはそこへ移動する．以下の例は受講者 isale13 での例だが、自分のアカウント名に読み替えること．

```
$ cd /mwork2/isale13
```

もしも/mwork1で作業を行いたい場合には，このディレクトリを自分で作る(mkdir /mwork1/isale13 など)．次に2.1節で行なったのと同様にファイルを展開し，グループを変更し，ディレクトリのアクセス権を制限する．

```
$ tar xvpf ~isale00/school2023/isale-work-2023.tar.gz
$ chgrp -R isale ./isale-work
$ chmod 750 isale-work
```

そして，上記に引き続いて以下のコマンドを実行しておくのが良い．

```
$ find isale-work -print -exec touch -a {} ¥;
```

これはディレクトリ isale-work/ 以下のファイルの最終アクセス時刻(atime)を手動で更新するコマンである．作業領域(/mwork1, /mwork2)下では最終アクセス時刻が一定期間(120 日間)を超えたファイルは逐次削除される．一方で，iSALE の配布ファイル(tar.gz)内には最終アクセス時刻が 120 日前より古いものが含まれことが普通である．よって tar.gz ファイルの展開後に上記の find コマンドを一度は実施しておかないと，tar.gz ファイル展開した日の深夜に古いファイルが自動的に削除される可能性がある（この現象は数年前の iSALE 講習会で実際に発生した）．なおファイルの最終アクセス時刻は ls -luR コマンド等で見ることができる．詳細は man ls を参照していただきたい．

Tar.gz ファイル展開と最終アクセス日時の変更が済めば，あとはホームディレクトリ上に於ける場合と同様に作業を進めることができる．

本章の残りの部分では iSALE 計算に必要なファイル群について解説し，iSALE の実行方法を解説する．2.1 節ではとりあえず iSALE を走らせてみることにしよう．続く 2.2 節で配布された iSALE-Dellen の中身について解説する．2.3 節では CfCA 共同利用計算機の構成について解説する．

2.3. iSALE の実行

まずは iSALE 開発者らによってデモ用に用意されている例題「demo2D」を走らせてみよう.

```
$ cd isale-work/demo2D
```

でディレクトリ demo2D へ移動する. ディレクトリの中身を確認してみよう.

```
$ ls -F
```

Plotting/ asteroid.inp eos@ iSALE2D@ iSALEMat@ iSALEPar@ isale_run.sh isale_run.sh.sample@ material.inp parameters.db

となる. なお上記の例では ls にオプション -F を付け, ディレクトリ名がスラッシュ / 付きで, symbolic link は@付きで表示されるようにしている.

```
$ qsub -N ${PWD##*/} isale_run.sh
```

と入力すればプログラム iSALE2D が実行される. 終了まで数分かかる. この間に「qstat」コマンドを入力すると自身が投入した Job が走っているかどうかを確認できる. ここで-N ${PWD##*/}は Job 名を現在のディレクトリに指定するオプションである.

```
$ qstat
```

Job id Name User Time Use S Queue

---------- ---------- ---------- -------- - -----

66306.grd0 ldemo2D isale01 00:00:00 R long

このように自分の講習会 ID に Job ID(この例では 66306)が付されていれば, Job が流れているということになる.「qstat」コマンドで上記の Job ID が消えたときは計算が終了したことを示している. qstat コマンドの詳細については CfCA が提供する計算サーバの利用手引書にまとめられているので, 参照のこと.

https://www.cfca.nao.ac.jp/node/165#management

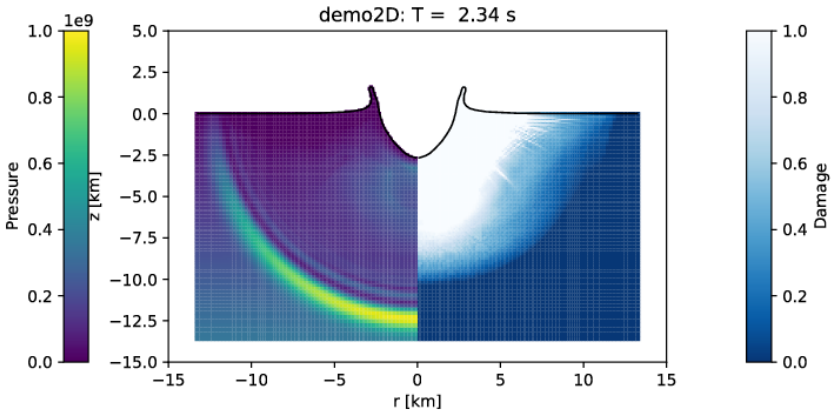
この状態で「ls」で確認すると「Plots」と「demo2D」という新たなディレクトリが作成されているはずである．「demo2D」には今走らせた iSALE の計算出力が，「Plots」には計算結果が描画された画像ファイルが格納されている．「Plots」の中身を確認しよう．

```
$ ls ./Plots
```

と打つと，DamPre-XXXXX.png and .eps という画像が全部で 40 枚できているはずである．試しに 1 枚の図を開いてみよう．

```
$ display -resize 20% ./Plots/DamPre-00190.png
```

と入力し，



のような図が出てくれば計算&描画が正しく行われている[5]．

2.4. 配布ファイル iSALE-work/demo2D の中身

前節で iSALE2D を走らせることができるようになった．本節では iSALE-work ディレクトリの中身の中で iSALE 計算に特に必要なものを説明する．前節で iSALE2D を実行するために移動した「demo2D」の中身がエンドユーザにとって

[5] -resize オプションの後に置く数値を増やせば表示される画像の大きさも大きくなるが (例. -resize 50%)，データの転送量も増えるので，描画に長い時間が掛かるようになる．

必要なファイル群である．以下では「demo2D」の中身を説明する．なお，「demo2D」は./share/examples/demo2D と同一である．

2.4.1. isale_run.sh (ファイル)

計算サーバの計算ノードへ Job を投げるための命令ファイルである．

```
$ less isale_run.sh
```

と入力すると，ファイルの中を読むことができる．この中で，

```
time ./iSALE2D –i asteroid.inp –M material.inp
```

とある部分が iSALE2D の実行命令である．-i, -M オプションによって 2 つの入力ファイル「asteroid.inp」，「material.inp」を読み込んでいる．冒頭の time は iSALE2D の実行時間を測定・出力するためのコマンドである．この 2 つの入力ファイルを編集することによって好みの計算を実施することができる．この 2 つのファイルの内容については 3 章で解説する．iSALE 実行命令に続く

```
python ./Plotting/plot.py
```

は Python スクリプト plot.py の実行命令である．Python で書かれた「plot.py」の中で pySALEPlot を読み込んでいる．

q を入力して「isale_run.sh」を閉じ，再び「ls」コマンドで「demo2D」ディレクトリに置かれたファイル群をみると，

iSALE 実行ファイル:	iSALE2D
iSALE 入力ファイル:	asteroid.inp, material.inp
計算結果(ディレクトリ):	demo2D, Plots
python スクリプト群(ディレクトリ)	Plotting

が含まれていることを確認できる.「isale_run.sh」を編集して，読み込ませる 2 つの iSALE 入力ファイルと実行する python スクリプトを指定すること，が iSALE で計算を実行することの中身である.

2.4.2. Log.out, Log.err (ファイル)

Log.out は先程実行した iSALE2D の標準出力を格納したテキストファイルである．Log.err には標準エラー出力が書かれている．「qsub」コマンドで「isale_run.sh」を実行した後には両ファイルを確認する癖をつけるとよい．ファイル「Log.out」の中に

SETTINGS FINISHED..... START JOB

という行があれば入力ファイルが正常に処理され，計算が始まったことを意味する．入力ファイルの編集に失敗すると，計算は実行されない．この場合は「qstat」で確認しても自分の Job ID が見つからない．ファイル Log.out, Log.err の中にエラーが書き込まれているので，内容を確認し入力ファイル「asteroid.inp」，「material.inp」を適切に修正する必要がある．pySALEPlot による解析でも実行エラーは Log.out, Log.err に書き込まれる．この場合は Python スクリプトの py ファイルを修正する．計算が正常に終了している場合は計算にかかった時間が記録されている．この情報は入力ファイルを編集し，異なる計算を実施するときに計算終了までにかかる実時間を推定するのに役立つ.

2.4.3. eos (ディレクトリ)

ここには状態方程式への入力ファイル群が格納されている.「ls」コマンドで中身をみてみるとよいだろう．ファイルの名前が物質を表す．入力ファイルの名前は 7 文字でなければいけないという規則がある．ファイル名の拡張子.tillo は Tillotson EOS への入力パラメータ, .aneos は ANEOS で計算された EOS table, .input は ANEOS への入力パラメータを意味する．自身で新たに状態方程式の入力ファイルを作成した場合はこのディレクトリに置くことによって,「material.inp」から呼び出すことができる. 3 章(3.2 節)で解説する.

2.4.4. parameters.db (ファイル)

iSALE の入力ファイルのパラメータの許容値を規定するファイル．すべての入力パラメータの説明と設定可能範囲がこの中に記入されているので、manual の一部と考えるとよい．

本章の最後に講習会では事前に講師側で設定してある PYTHONPATH の設定について説明しておく．今回の講習会期間中は必要ではないが，終了後に CfCA の利用申請を行い，iSALE 計算を継続していく際には必要となるので覚えておこう．座学講義編で解説したように iSALE の計算出力は pySALEPlot を用いて解析，描画を行う．Python で pySALEPlot を実行する際に必要なファイル群は isale-work/lib に置かれているため PYTHONPATH をこのディレクトリに指定しておく必要がある．

```
$ pushd isale-work/lib
```

で isale-work/lib に移動し，

```
$ export PYTHONPATH="`pwd`"
```

とすることでパスを通すことができる．

```
$ echo ${PYTHONPATH}
/home/isaleXX/isale-work/lib
```

と応答があれば成功である．上記の echo コマンドで何も応答がない，もしくは違うディレクトリが返って来た場合には PYTHONPATH が正しく通っておらず，pySALEPlot による解析&描画を行うことができないため，原因を追求し，解決すること．この設定は CfCA 計算機からログアウトすると失われてしまうため，毎回行う必要がある．もし CfCA の計算サーバ，解析サーバで iSALE 計算以外で python を使用しない場合は，環境変数を設定し，ログイン時に自動で PYTHONPATH が通るようにしておくと便利である．環境変数は自身のホームデ

ィレクトリの.bash_profile に書かれている．お好みのエディタで.bash_profile を開き以下の内容を入力し，保存しておく．

```
if [ "${PYTHONPATH:-}" = "" ]; then
  PYTHONPATH="$HOME/isale-work/lib"
  export PYTHONPATH
else
  PYTHONPATH="$HOME/isale-work/lib:$PYTHONPATH"
  export PYTHONPATH
fi
```

これで毎回のログイン時に自動で PYTHONPATH が HOME/isale-work/lib に通るようになる．

作業領域 /mwork2 上で作業を行うことを決めた場合には，上記の $HOME を /mwork2/isaleXX に変更する．利用者 isale13 を例に取れば以下となる．

```
if [ "${PYTHONPATH:-}" = "" ]; then
  PYTHONPATH="/mwork2/isale13/isale-work/lib"
  export PYTHONPATH
else
  PYTHONPATH="/mwork2/isale13/isale-work/lib:$PYTHONPATH"
  export PYTHONPATH
fi
```

2.5. CfCA 共同利用計算機の構成

本章ではこの講習会で受講者が使用する 2 種類のサーバとその用途について解説する．1 つ目が計算サーバ，2 つ目が解析サーバである．両者の使い分けとしては、まず利用者にとって比較的に自由度の高い解析サーバで pySALEPlot 用の python スクリプトを試行錯誤を行いながら完成させ，その後に計算サーバで Job を投入するという手順がおすすめである．

なお*今回の講習会アカウント(isaleXX)で計算サーバ・解析サーバが利用できるのは 2023 年 7 月 7 日(金)まで*に限られる．講習で使ったファイルや計算結果はこの期日よりも前にどこかに退避する必要がある．

この講習会が終了した後にも iSALE を使いたい場合には，一般利用者として CfCA 共同利用計算機の利用申請を行う必要がある．申請書はその内容の学術性や実行可能性を審査され，採択されれば晴れて一般利用者として登録される．なお一般利用者のアカウント有効期限は当該年度末なので，利用申請は毎年度行う必要がある．

2.5.1. 計算サーバ

これは最初にログインした機材のことである．今回の講習会では専用の計算サーバである「grd0」にログインする[6]．計算サーバは「ログインノード」と「計算ノード群」から構成されている．1章でログインしたのは計算サーバのログインノードである．ユーザはログインノードにログインし，

```
$ qsub isale_run.sh
```

によって計算ノード群へ Job を投入するという流れになっている．isale_run.sh の冒頭のヘッダ 10 数行は計算ノード群へ投入する際の命令である．なお，ユーザは計算ノード群にはログインできないし，ログインする必要もない．

計算サーバ(のログインノード。この講習会では grdo のこと)で行えるのは主に以下の操作である．

(a). qsub isale_run.sh による Job 投入 (iSALE2D, pySALEPlot)
(b). 好みのエディタによるファイル編集
(c). 他サイトへのファイルの送受信 (scp などを利用)

*なお計算サーバのコマンドライン上で iSALE2D や python を直接実行することは禁止されているので絶対に行わないこと．*これらの実行は必ず qsub コマン

[6] 講習会終了後に一般利用者として CfCA に利用申請を提出し，審査を通過すると別機材「m000」にログインできるようになります．m000 はホームディレクトリなどを grd0 と共有していますが，m000 が"本物"の計算サーバです．使える資源の質・量とも本講習会で使われるものより上です．なお一般利用者は講習専用機材 grd0 にはログインできません．

ドを使った PBS ジョブとして投入することが定められている[7]. 図 2.3.1.1 に上記説明を図示する.

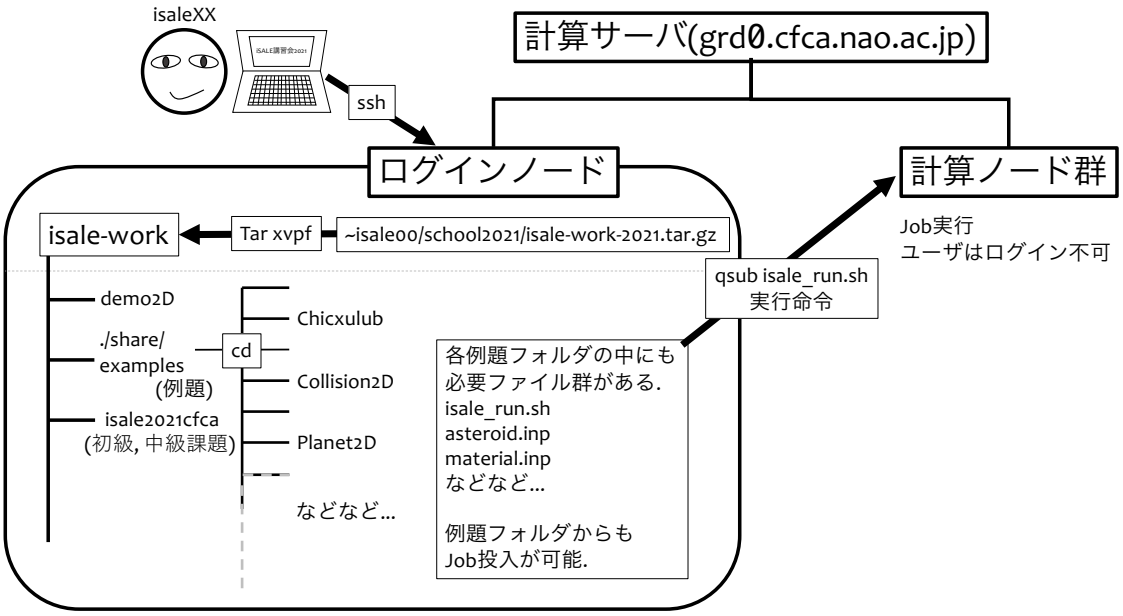


図 2.3.1.1. 本講習会における計算サーバの中の概念図.

計算サーバの詳細, 及び利用規約は以下のページを参照.

https://www.cfca.nao.ac.jp/node/157

https://www.cfca.nao.ac.jp/node/165#regulation

2.5.2. 解析サーバ

解析サーバは計算結果を解析するためのサーバ群として用意されている. CfCA の共同利用計算機のアカウントを持つ全てのユーザが使用でき, アカウント取得と同時に解析サーバのホーム領域も提供される. なお先述したようにこの解析サーバについても今回の講習会用に専用の機材が準備されているので[8]、受講者は以下の機材にログインすることになる. 新たに端末アプリを立ち上げ,

```
$ ssh -CY isale00@an00.cfca.nao.ac.jp
```

[7] 今回の講習会専用機 grd0 にはこの規約は適用されないものの, 将来的に一般利用者として計算サーバを使う日のことを考えると今からこの規約に慣れておくことが望ましいです.

[8] an00, an01 には一般の共同利用者はログインできません. 一般の共同利用者が使う解析サーバには別のノード名が与えられています.

もしくは

```
$ ssh -CY isale00@an01.cfca.nao.ac.jp
```

(`isale00` は自身の講習会 ID に置き換える.)

と入力する. すると, 再びパスワードを要求されるので, grdo へのログインに使ったものと同一の NIS パスワードを入力すれば解析サーバにログインできる. また計算サーバと解析サーバのホームディレクトリは共有されており, 片方でのファイル複製, 削除, 編集といった作業はもう片方へ自動的に反映される[9].

CfCA の規約では, 解析サーバでは計算資源の 50%を超えない限りは自由度の高い利用が可能である[10]. Python には iPython という対話機能が実装されており, 解析用 Python スクリプト作成時に有用である. iPython の利用方法については 4 章(4.3 節)で解説する. 計算サーバではログインノードのコマンドライン上での iPython の利用は禁止されている. よって受講者は解析サーバで iPython を活用しつつ, 解析用の Python スクリプトを作成するのがよいであろう. また JupyterLab, Jupyter Notebook のような対話型開発環境も利用可能である.

解析サーバの詳細, 及び利用規約は以下のページを参照のこと.

https://www.cfca.nao.ac.jp/node/22

https://www.cfca.nao.ac.jp/node/22#policy

[9] 従ってバックアップとしての利用はできないことに注意しましょう. 解析サーバで消してしまったファイルは計算サーバからも消えます. 必要な場合は

```
$ cp -pr hoge hoge_bk
```

など適当に複製をとり, 復元できるようにしておきましょう.

[10] 今回の講習会専用機 `an00, an01` にはこの規約は適用されないものの, 将来的に一般利用者として解析サーバを使う日を考えると今からこの規約に慣れておくことが望ましいです.

3. iSALE の 2 つの入力ファイル

2.2.1 項で iSALE2D の実行ファイルに読み込ませる 2 つの入力ファイル，「asteroid.inp」，「material.inp」があることを述べた．この 2 つのファイルを編集することが，自身で iSALE 研究を進めることである，と言える．本章ではこの 2 つの入力ファイルの読み方を解説する．3.1 節で asteroid.inp，3.2 節で material.inp について述べる．変数については iSALE-Dellen manual [Collins et al., 2016, Section 3]，もしくは parameters.db に詳細が記載されているので，そちらも合わせて熟読することを薦める．3.3 節では「examples」ディレクトリの中身の活用法について述べる．数値は全て MKS 単位系である．なお***入力ファイルの編集時に tab を挿入するとエラーとなり，計算は実行されない．スペースを挿入すること．***

3.1. asteroid.inp

これは一言でいうと数値計算条件を設定するファイルである．

- 座標系選択(デカルト or 円柱座標)
- 計算領域，格子サイズ
- 計算終了時間，ファイルへの書き出し間隔
- グローバル変数(重力加速度，地表面温度)
- 衝突天体(形状，サイズ， 速度，温度構造)
- 標的天体(形状，サイズ， 速度，温度構造)
- 書き出す物理量
- トレーサ粒子(挿入の有無，挿入間隔，書き出す物理量)

などを入力する．asteroid.inp の中身を読んでいこう．ディレクトリ~/isale-work/demo2D に移動し，

```
$ less asteroid.inp
```

で asteroid.inp の中身[11]を読むことができる．冒頭に 10 数行のヘッダがあり，7 つのブロックがある．各ブロックは 3 列の構成になっている．左列は iSALE に渡される変数名，中列はその変数の一言説明，右列が入力する値(文字列も含む)である．以下，ヘッダと 7 つのブロックについてみていこう．なお，本節では円柱座標系を選択していることを前提として説明する[12]．また Ac. Fluid. Parameters のブロックは音響流動モデル(Acoustic fluidization)に関するパラメータとなっているが，本講習会では触れない．

3.1.1. ヘッダ

ヘッダの 10 数行にはコメントアウトの方法が書かれている．各行の先頭に「-」もしくは「!」と入力すると，その行はコメントアウトされ，実行ファイル(iSALE2D)に無視される．両者の違いは backup ファイルへの書き込みの有無である．iSALE では計算を実行すると自動的に入力ファイルが複製され「INFO」ディレクトリに格納される(4.2 節参照)．「-」でコメントアウトしたときは backup ファイルにも同様に書き込まれるが，「!」でコメントアウトすると backup ファイルからその行が削除されるという違いがある．

3.1.2. General Model Info

計算出力の格納先(ディレクトリ)を指定する．

```
VERSION                 __DO NOT MODIFY__       : 4.1
DIMENSION               dimension of input file : 2
PATH                    Data file path          : ./
MODEL                   Modelname               : demo2D
```

VERSION と DIMENSION は編集する必要はない．下 2 つの変数 PATH がディレクトリのパスを指定し，MODEL で名前を指定する．この例だと，現在いるディレ

[11] ~/isale-work/demo2D に置かれている asteroid.inp は~/isale-work/share/examples/demo2D/asteroid.inp と同一です. iSALE の典型的な計算例として様々な物理モデルを同時に使用する計算となっています．デモ用に計算が数分で終わるようになっています.

[12]円柱座標系では定式上，r~dr の計算領域の結果の正しさに気をつける必要があります.

クトリの中に「demo2D」というディレクトリが生成され，その中に iSALE の結果群が格納される．

3.1.3. Mesh Geometry Parameters

計算格子(計算領域，サイズ)を設定する．iSALE では高解像度計算用の格子(High-resolution zone)と計算領域拡張用の格子(Extension zone)の 2 種類の格子を配置できる．数値衝突計算は多くの場合，点源を起点とする運動を解くので，流体計算で頻繁に活用される周期境界条件を採用することができない．計算領域の端で物理量をどのように計算するか，は悩ましい問題となる．境界条件をどのようにとっても現実では起こり得ない反射波(圧縮波，膨張波のどちらもあり得る)が，流体運動に影響を与えてしまう可能性がある．ところが全格子数 N_{grid} を増やすと計算にかかる時間は$>O(N_{grid}^2)$で増えてしまう．これを解決し，少ない格子数で計算領域を拡張するのが Extension zone である．

GRIDH	horizontal cells	: 0	: 80	: 32
GRIDV	vertical cells	: 35	: 90	: 15

GRIDH, GRIDV はそれぞれ動径方向(対称軸に垂直な方向)と鉛直方向(対称軸に平行な方向)に張る格子数である．3 つの値を入力する．真ん中の数字が High-rezolution zone，左右の数字はそれぞれ負の向き，正の向きに配置する Extension zone の格子数に対応する．GRIDH の左側の数字(0)はデカルト座標を選択しているときのみ有効で，円柱座標系では 0 にしなければならない．

GRIDEXT	ext. factor	: 1.03d0
GRIDSPC	grid spacing	: 100.D0
GRIDSPCM	max. grid spacing	: -20.D0

GRIDSPC は High-resolution zone に配置する格子の実空間における長さ(1 格子が何 m に対応するか)である．Extension zone に配置する格子のサイズ $L_{ext,i}$ は

$$L_{ext,i} = \min(\text{GRIDEXT} \times L_{ext,i-1}, \text{GRIDSPCM}) \quad (3.1.3.1)$$

となる．つまり，公比 GRIDEXT の等比数列の格子サイズ，ただし上限の格子サイズは GRIDSPCM となる．上記の例では GRIDSPCM は–20 となっている．負の値は GRIDSPC で規格化した値に対応しており，ここでは上限格子サイズを 2 km を設定していることになる．

3.1.4. Global Setup Parameters

このブロックでは iSALE の数値解法や重力の扱いや初期温度構造を決定するパラメータを入力する．

S_TYPE	setup type	: DEFAULT

iSALE では用途に合わせて様々な計算(地すべりや，変形実験の模擬など)を行うことができる．ここでは文字列で指定する(LANDSLIDE, DEFORM など)ことによってそれに適した計算セットアップが自動的に行われる．本講習会では一様重力場中の衝突に適した設定となる「DEFAULT」のみを使用する．その他の計算に興味のある読者は parameters.db を参考にしてほしい．

ALE_MODE	ALE modus	: EULER

計算中の数値解法を選択する．EULER, LAGRANGE, ALE を選択可能であるが，講義編で述べたとおり，iSALE では EULER 法を中心として整備が進められている．本講習会では ALE_MODE = EULER のみを使用する．

T_SURF	Surface temp	: 293.D0
DTDZSURF	Temp. grad. surf.	: 10.D-3
D_LITH	Lithosp. thickness	: 80.D3
R_PLANET	Planet radius	: 6370.D0[13]

これらは設定した Projectile と Target 中の初期温度構造に関するパラメータである．T_SURF は表面温度，DTDZSURF は岩石圏(Lithosphere)中の温度勾配，D_LITH は岩石圏の厚み，R_PLANET は想定している惑星半径である．初期温度構造

[13] この例では半径 6,370 m の惑星を想定しています．地球より 3 桁小さい惑星です．

(LAYTPROF, 後述)を等温にする場合は T_SURF のみを入力すればよい. 残りの 3 つは伝導や対流によって決定される温度構造を簡易的に入力するためのパラメータである[14].

GRAV_V gravity : -9.81D0
GRAD_TYPE gradient type : DEFAULT
GRAD_DIM gradient dimension : 2

これらは重力場を記述する. GRAV_V は重力加速度である. 負の値は鉛直下向きを意味する. GRAD_TYPE は重力場の設定を変える.

GRAD_TYPE には

DEFAULT (時間一定)
NONE (無重力)
CENTRAL (標的天体中心)
SELF (自己重力)

のいずれかを入力する. 本講習会では DEFAULT のみを使用する. GRAD_DIM は重力加速度勾配を設定する. 格子サイズを大きくとった計算では深さ方向に重力加速度が変化することを考慮する必要がある. GRAD_DIM = 1 では鉛直方向, 2 では動径方向, 鉛直方向ともに重力加速度勾配を計算する.

3.1.5. Projectile & Target Parameters

この 2 つのブロックでは“Projectile”と“Target”に関する設定を行う. 2 つを合わせて解説する. iSALE における“Target”は設定した表面の位置より下部の計算領域全てを埋める円柱として設定される. それ以外は全て“Projectile”である.

OBJNUM number of proj. : 1
LAYNUM number of layers : 1

[14] iSALE の中で熱伝導や対流熱輸送を解くわけではないことに注意してください. 均質媒体中で熱伝導, 対流熱輸送で決定される温度構造の解析解から温度構造を与えます.

OBJNUM は“Projectile”の数を設定する．図 3.1.5.1, 3.1.5.2 に設定例を示す．例えば金属核をもつ球形状微惑星同士の衝突を計算したい場合は LAYNUM=0, ONJNUM=4 と設定することになる．

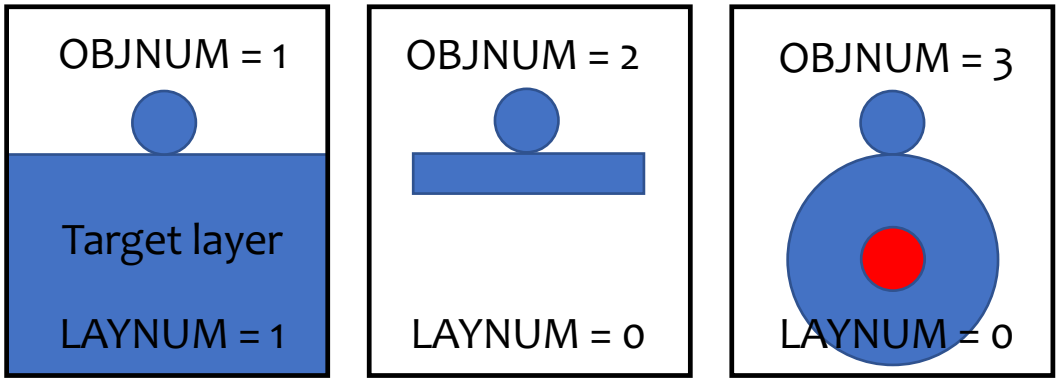


図 3.1.5.1. “Projectile”設定例と OBJNUM の関係.

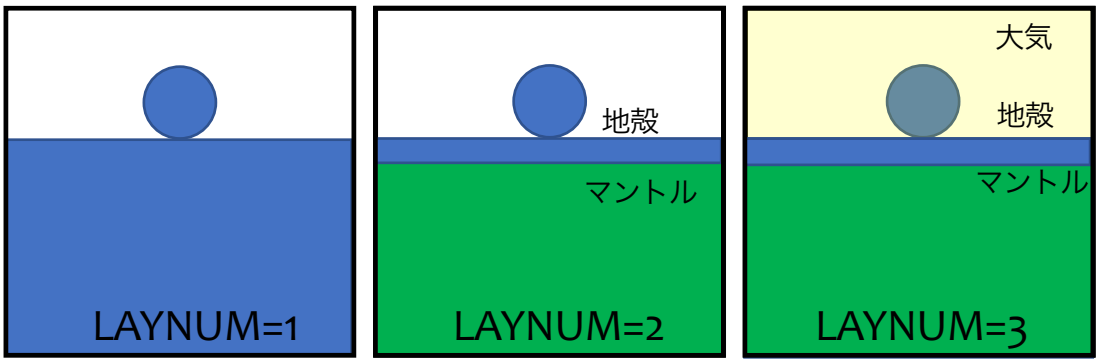


図 3.1.5.2. “Target”設定例と LAYNUM の関係.

OBJRESH[15]	CPPR horizontal	: 8
OBJTYPE	object type	: SPHEROID

OBJRESH は Projectile の動径方向の半径を何格子で分割するかを規定するパラメータである．数値衝突計算の空間解像度はこの値の大小で決まる．CPPR は Cells per projectile radius の略である．OBJTYPE は Projectile の形状を規定する．

[15]この例では計算時間を短くするため，CPPR の値が過剰に小さくなっています．

SPHEROID (球)
CUBOID (立方体)
BITMAP (ユーザ定義)
PLATE (動径方向全てを占める円板)
CYLINDER (円柱)

のいずれかを入力する．使用頻度が高いのは SPHEROID と CYLINDER であろう．この例(demo2D)では動径方向の格子数しか設定していないため自動的にアスペクト比が 1 に設定される．

OBJRESV CPPR vertical : 16

のように OBJRESV というパラメータを追加する[16]と鉛直方向の格子数を OBJRESH と独立に指定することも可能である．

OBJVEL object velocity : -6.5D3

OBJVEL で Projectile の初期速度を設定する．負の値は鉛直下向きを意味する．従ってこの例では Projectile の初期粒子速度が鉛直下向きに 6.5 km s^{-1} となる．

OBJMAT object material : mygrani
LAYMAT layer material : mygrani

“Projectile”と“Target”の物質を選択する．次節で説明する material.inp でつけた名前と正確に合わせることで 2 つの入力ファイルを紐付ける．

OBJTPROF object temp prof : CONST
LAYTPROF layer therm. prof : COND

OBJTPROF は Projectile の LAYTPROF は Target の初期温度構造を決める．

[16] OBJRESH の上か下あたりに挿入するとわかりやすいですが, asteroid.inp 中の変数の入力順序は自由です．

CONST	(等温)
COND	(熱伝導)
CONDCONV	(Lithosphere は熱伝導，それ以深は対流熱輸送)
CONDCONVCAP	(基本は CONDCONV，ただし，融点[17]を超えない)

のいずれかを入力する.

LAYPOS	layer position	: -85

LAYPOS で計算領域中の Target 表面位置を設定する．正の値は計算領域下端から数えた鉛直方向の格子数，負の値は計算領域下端から数えた鉛直方向格子数と鉛直方向の計算領域全体の格子数の比である．例えば鉛直方向に 200 格子を設定していて，下から 140 格子目に標的表面を配置したいときは LAYPOS = 140 or -70 と設定すればよい.

3.1.6. Time Parameters

DT	initial time increment	: 1.0D-3
DTMAX	maximum timestep	: 5.D-2

iSALE は陽解法で数値積分を実施する．時間刻み Δt を十分に細かくとる必要がある. DT で初期のタイムステップ, DTMAX でタイムステップの最大値を設定する．このとき Courant-Friedrichs-Lewy (CFL)条件を満たしていないと十分な精度をだすことができない. CFL 条件は音速 C_s，格子サイズ Δx，時間刻み Δt_{CFL} を使って

$$C_s \frac{\Delta t_{CFL}}{\Delta x} < 1 \qquad (3.1.6.1)$$

で与えられる. iSALE では計算の各タイムステップで EOS から音速の最大値を計算しており，実際の時間刻み Δt は

[17] 座学講義編テキストの 4.2 節も参照.

$$\Delta t = \min\left(\frac{\Delta t_{CFL}}{4},\ \mathrm{DTMAX}\right) \qquad (3.1.6.2)$$

となる．つまり時間刻み Δt は毎ステップで更新され，数値積分が行われる．

TEND　　end time　　: -10.D0

TEND で計算終了時刻(計算中の時刻)を設定する．負の値は Projectile の貫入特徴時間 t_s で規格化した値に対応する．貫入特徴時間 t_s は Projectile 半径 R_p と衝突速度 v_{imp} から

$$t_s = \frac{2R_p}{v_{imp}} = \frac{2 \times \mathrm{OBJRESH} \times \mathrm{GRIDSPC}}{\mathrm{OBJVEL}} \qquad (3.1.6.3)$$

と計算される．この計算では R_p (OBJRESH×GRIDSPC)= 800 m, v_{imp} (OBJVEL) = 6.5 km s^{-1} なので, t_s = 0.25 s である．計算中の時刻がその 10 倍(2.5 s)になるまで計算を行う，ということになる．従ってこの例では TEND = 2.5 = -10 である[18].

DTSAVE　　save interval　　: -0.05D0

DTSAVE でデータ出力間隔を指定する. TEND と同様に負の値は t_s で規格化した値を表す．書き出す総ステップ数 n_{out} は

$$n_{out} = \frac{\mathrm{TEND}}{\mathrm{DTSAVE}} \qquad (3.1.6.4)$$

となる. iSALE では n_{out} の上限が 4000 に設定[19]されているので注意する必要がある．それ以上に細かい出力をさせたい場合は Restart 機能を活用することになる．

[18] t_s は OBJRESH と OBJVEL の値に応じて変化するので気をつけましょう.
[19] 不用意に巨大なファイルを生成してしまわないようにする安全措置であると想像できますが, なぜこのような設定になっているのか真相は不明です.

Restart に関しては本講習会では扱わない．必要になった場合は iSALE-Dellen manual [Collins et al., 2016, pp.18]で DUMP パラメータを調べるとよいだろう．

3.1.7. Boundary Conditions

格子法の計算では計算領域の端での差分処理をどうするか(境界条件と呼ぶ)が問題となる．このブロックでは 4 つの領域での境界条件を設定する．

BND_L	left	: FREESLIP
BND_R	right	: OUTFLOW
BND_B	bottom	: NOSLIP
BND_T	top	: OUTFLOW

境界条件には

OUTFLOW	(物質の粒子速度を保ったまま，計算から取り除く)
FREESLIP	(物質の粒子速度の接線成分は保存，直行成分を 0 にする)
NOSLIP	(物質の粒子速度を境界で 0 にする)

のいずれかを選択できる．円柱座標を選択している場合，BND_L は必ず FREESLIP に設定しなければならない．FREESLIP, NOSLIP では計算領域の端から圧縮波が反射してくる場合があり，注意が必要である．OUTFLOW では圧縮波の反射は起こらないが膨張波が伝播する．こちらも注意が必要である．また OUTFLOW では計算中の質量保存が成立しない，という問題もある．どの境界条件がよいか?については一般的な正解がないので，基本的には反射波の影響がでないように，Extension zone を広く設定するのがよい．しかし，クレータ形成完了まで計算を実行しようとすると，それが難しい場合もある．そのときは計算領域を系統的に変化させ反射波が結果にどの程度影響しているか，を評価する必要があるであろう．

3.1.8. Numerical Stability Parameters

人工粘性係数の大きさを設定する．人工粘性については講義編 3.2.3 項を参照してほしい．

AVIS	art. visc. linear	: 0.24D0
AVIS2	art. visc. quad.	: 1.2D0

AVIS が 1 次, AVIS2 は 2 次の項の大きさを表す．ここで入力されている値は iSALE 開発チームの推奨値であるので，特別な理由がない限りは変更しないことをおすすめする．

3.1.9. Data Saving Parameters

ここでは出力する物理量を指定する．

QUALITY Compression rate : -50

VARLIST List of variables : #Den-Pre-Tmp-Yld-Dam-Ert-Vib-YAc-PVb-VEL#

QUALITY は出力データの圧縮方法を規定する(らしい)．変更しないことが推奨[20]されている．VARLIST は出力する物理量を決定する．それぞれの物理量が 3 文字で定義されており，間を“-”(マイナス)でつなぎ，両端を“#”(シャープ)で囲まなければならない．

Den	Density
Tmp	Temperature
Pre	Pressure
Sie	Specific Internal Energy
Dam	Damage i.e. Total Plastic (shear) Strain
VSt	Volume strain
Alp	Distension
Yld	Yield Strength
YAc	Strength of weakening due to Acoustic Fluidization
VEL	Velocity components (horz. and vert.; cell-centered)

[20] parameters.db に「Please do not change this value unless you know what you do. Changing compression algorithm or density might result in compression artifacts and, thus, affect the simulation results and post-processing quality.」と記載されています．筆者も詳細は理解していません...

Dm1　Dummy field number 1
Dm2　Dummy field number 2
C_x　Horizontal co-ordinate (needed for 2D Lagrangian calcs.)
C_y　Vertical co-ordinate (needed for 2D Lagrangian calcs.)

などを選択可能である. VARLIST に含まれていない物理量を計算後に調べたくなった場合には再計算を行うしかないので, 計算前によく検討しておく必要がある.

3.1.10. Control Parameters

このブロックでは様々なオプションを設定する.

STRESS　calc_stress　: 1

STRESS は偏差応力テンソルを計算するか否かを 0 か 1 の値で指定する. STRESS=0 のときは強制的にポワソン比＝0.5 となる. 従って material.inp で降伏応力モデルを選択していたとしても偏差応力テンソルの計算値が 0 になるため, 完全流体と等しくなる. material.inp を編集することなく, 強度のありなしの効果を調べることができるので, 便利である. ただし, VARLIST の出力の中には STRESS=0 ではエラーを出すもの(Dam, Vst, Yld, YAc など)がある. エラーが出たときはエラーファイルを確認し, VARLIST を適切に修正する必要がする. 実はここには顕に含まれていないが自動的に入力されている Control parameters も多くある. 本講習会では触れないが, 興味ある方は iSALE-Dellen manual, 及び parameters.db を参考にしてほしい.

3.1.11. Tracer Particle Parameters

これまで`./isale-work/demo2D/asteroid.inp` の中身を解説した. この入力ファイルではトレーサ粒子は挿入されない. iSALE には Lagrangian tracer particles = 格子の流速に従って運動する質量ゼロの粒子を挿入することができる. トレーサ粒子には格子に記録された物理量だけでは追うことができない, 物質の実空間, 運動量空間, 及び相図の上での軌跡を記録できる. 研究を進める上で使わない手はないだろう. そこで以下では asteroid.inp を編集してトレーサ

粒子を挿入する方法を述べる. iSALE 開発チームが用意した例題~/isale-work/share/examples/Chicxulub ではトレーサ粒子が活用されている.

```
$ less ./examples/Chicxulub/asteroid.inp
```

と入力し，例題「Chicxulub」の中の入力ファイルを覗いてみよう.

Numerical Stability Parameters と Control parameters (global)の間に「Tracer Particle Parameters」のブロックがある. トレーサ粒子を挿入するには，このブロックをコピーして自身で編集している asteroid.inp にペーストすればよい. 以下ではパラメータについて簡単に説明する.

```
TR_QUAL                    integration qual.                    : 1
```

トレーサ粒子を挿入するか否かを設定する. 0 ならば挿入しない設定になる.

```
TR_SPCH    tracer spacing X    : -2.D0          : -2.D0          : -2.D0
TR_SPCV    tracer spacing Y    : -2.D0          : -2.D0          : -2.D0
```

トレーサ粒子を挿入する間隔を指定する. TR_SPCH は動径方向, TR_SPCV は鉛直方向である. 正の値は実距離，負の値は格子数を表す. この例では 2 格子毎にトレーサ粒子を挿入するということになる. なおここで数字が 3 列になっているのは「Chicxulub」は 3 種類の物質を扱う例題だからである.

```
TR_VAR                     add. tracer fiels[21]                : #TrP-TrT#
```

トレーサ粒子に記録する物理量を指定する.「3.1.9. Data Saving Parameters」の VARLIST と同様に，それぞれの物理量が 3 文字で定義されており，間を“-”でつなぎ，両端を“#”で囲まなければならない.

```
TrP          Peak pressure
```

[21] 綴りが誤っているように見えるが，原文ママ

TrT	Peak temperature
TrE	Peak specific internal energy
Trp	Pressure
Trt	Temperature
Tre	Specific internal energy
Trd	Density
TrM	Material
TrA	Distension (alpha)
TrV	Volume strain
TrS	XX and YY components of the dev. stress tensor
TrX	XX component of the dev. stress tensor
TrY	YY component of the dev. stress tensor

を選択できる.

3.2. material.inp

material.inp は使用する物質モデル(状態方程式, 降伏応力, 空隙圧密, 熱弱化など)を設定するファイルである. material.inp の中身を読んでいこう. ディレクトリ~/isale-work/demo2D に移動し,

```
$ less material.inp
```

で material.inp[22]を読むことができる. material.inp は 2 つのブロックで構成されている. 上半分は使用するモデルの選択, 下半分はそれぞれのモデルへの入力パラメータである. 本節ではモデルの選択部(上ブロック)について解説する. ここで選択したモデルにはそれにあったパラメータを下ブロックに入れる必要がある. モデルとパラメータの組み合わせの詳細は iSALE-Dellen manual [Collins et al., 2016, Section 4]に書かれているので, そちらを参考にしてほしい.

MATNAME Material name : mygrani

[22] ~/isale-work/share/examples/demo2D/material.inp と同一ファイルです.

計算中の物質名である．ユーザ自身が自由に設定可能だが，半角[23]英数字でぴったり 7 文字でなければならない．material.inp の MATNAME でつけた名前を asteroid.inp の OBJMAT 及び LAYMAT と対応させることによって 2 つの入力ファイルを紐付けることができる．

EOSNAME EOS name : granit1
EOSTYPE EOS type : aneos

状態方程式モデルを選択する．EOSTYPE には

tillo Tillotson EOS
aneos ANEOS

のどちらかを選択することができる．それぞれの EOS の特徴については講義編の 3.2.2 項を参照してほしい．EOSNAME は~/isale-work/share/eos の中から選択することができる．このとき拡張子に気をつける必要がある．Tillotson EOS のパラメータは拡張子.tillo, ANEOS table は拡張子.aneos である．

STRMOD Strength model : ROCK

降伏応力モデルを選択する．降伏応力モデルについては講義編の 4.1 節を参照してほしい．STRMOD には

ROCK - Pressure- and damage-dependent strength model for rock-like materials.
DRPR - Drucker-Prager: Linear pressure-dependent strength model for granular materials.
LUNDI - Lundborg intact: Non-linear pressure-dependent strength model for intact rock.
LUNDD - Lundborg damaged: Non-linear pressure-dependent strength model for damaged rock.
VNMS - Von Mises: Constant yield-strength model for ductile materials.
JNCK - Johnson and Cook: Strain and strain-rate dependent strength model for metals.
LIQU - Liquid: Newtonian fluid model
HYDRO - Hydrodynamic: Inviscid fluid model

[23] 1 バイトの ASCII

を選択することができる．使用頻度が高いと思われる，ROCK, DRPR, JNCK, HYDRO については講義編 4.1 節で言及した．ここではそれ以外の物質モデルについて簡単に言及する. LUNDI, LUNDD は ROCK model と完全互換であるため，ほぼ使う機会はないと思われる. VNMS model は降伏応力を定数として与えるモデルであり，延性物質の降伏挙動の近似として使われることがある．またいわゆる□-group scaling law [e.g., Holsapple and Schmidt, 1982]の□$_3$ 中の強度 Y は定数として扱われることが多い [e.g., Suzuki et al., 2012]. LIQU は粘性が効く流体(Newtonian fluid)を表すモデルである．粘性流体の偏差応力テンソル S_{ij} は歪速度 $\dot{\varepsilon}_{ij}$を用いて，

$$S_{ij} = 2\eta\dot{\varepsilon}_{ij} \quad (3.2.1)$$

となる．ここでηは粘性率である．

DAMMOD Damage model : COLLINS

DAMMOD は STRMOD=ROCK のときに有効になる. Damage parameter *D* の計算法を選択する．

NONE - No damage model; material remains intact.
SIMPLE - Shear failure model with constant failure strain.
IVANOV - Shear failure model with pressure-dependent failure strain.
COLLINS - Combined shear and tensile failure model with brittle,
semi-brittle and ductile shear failure regimes.

のいずれかを選択する. DAMMOD=NONE とした場合は塑性歪が蓄積して強度が下がる効果を取り入れられないので注意が必要である．なお DAMMOD=NONE のとき ROCK model は LUNDI, LUNDD と同一である．

ACFL Acoustic fluidisation : BLOCK

音響流動モデル(Acoustic fluidization model)を選択する．このモデルは一言で言うならば，応力依存の粘性を導入するモデルである．もともと月のクレータでよく知られていた Simple crater から Complex crater への遷移を説明するために導入された [e.g., Melosh, 1979].

NONE	No acoustic fluidization of the material.
BLOCK	Simple block-oscillation model.
MELOSH	Full Melosh (1979) model of acoustic fluidization.

のいずれかを選択できる．本講習会では扱わない．

PORMOD Porosity model : NONE

無限小空隙圧密モデルを導入するか否かを選択する．

NONE	No microscopic porosity
WUNNEMA	with microscopic porosity

のどちらかを選択する．

THSOFT Thermal softening : OHNAKA

熱弱化モデルを選択する．講義編 4.1.4 項，及び 4.2 節を参照して欲しい．

NONE	No thermal softening.
OHNAKA	Smooth hyperbolic tangent function of temperature. For use with all strength models except JNCK.
JNCK	Polynomial function of temperature. For use with Johnson and Cook (JNCK) strength model.

のいずれかを選択可能である．

LDWEAK Low density weakening : POLY

流体計算では蒸発するほどには温度が上がっていないにも関わらず，標準密度よりも密度が低くなることがある．これは設定した格子サイズに対して細かい構造の物質(jetting や spallation で生じた高速放出物ではよくみられる)が格子間を移動すると，周囲の空隙と平均化されて人工的に低密度になってしまう場合があることが原因である．実在物質であれば破壊，あるいは液滴分裂が起こっていることに相当するが，現時点の iSALE ではそのような物理を解いていない．このような物質は強度が下がるであろう．その効果を密度の多項式の形で与えるのが Low density weakening model である．ここでは

NONE　No low-density weakening.
POLY　Polynomial function of density.

のいずれかを選択できる．

3.3.　iSALE 開発チームによる例題

~/isale-work/share/examples ディレクトリには iSALE 開発チームが用意した様々な例題が格納されている．asteroid.inp や material.inp を編集する際に参考になるであろう．ただし，***material.inp に入力されているパラメータにはそれぞれの物質によって異なる値が入っているものの，信頼できる値ではない***ことを注意しておく．多くの場合，参照文献も明記されていない．実際に使用する値はユーザ自身が文献調査をするか，実験によって決定する必要がある．本節では各例題の内容を簡単に紹介する．3.3.1 項以降の表題の括弧内には，以前に筆者が計算を実行した際に計算終了までにかかった実時間の情報を付しておく(MacBook Pro, Mid 2014, OS X 10.10.5, 3 GHz Intel Core i7, 16 GB 1600 MHz, DDR3)．計算終了までにかかる時間を見積もるのに使えるだろう．

3.3.1.　Airburst (26 min)

大気中の爆発現象を模擬した計算．気体を計算に導入する際の参考になるであろう．asteroid.inp に初期内部エネルギーを指定する OBJENER が使われている．標準状態ではない初期条件を設定したい場合にはこのようにすればよい．

3.3.2. aluminum_1100_2D (60 min) & aluminum_6061_2D (62 min)

金属板を用いた室内衝突実験を模擬する計算．金属を計算に導入したいときに参考になるであろう．またクレータ径，深さを解析するための解析スクリプト(cratergrowth.py)が同梱されており，iSALE を使ってクレータ形成過程を研究したい読者には参考になるだろう．

3.3.3. Chicxulub (359 min = 6.0 hours)

6600 万年前に起きた生物大量絶滅事件(K/Pg 衝突)を引き起こした天体衝突[e.g., Schulte et al., 2010; Collins et al., 2020]の模擬計算．堆積岩，地殻，マントルという多層構造標的となっている．層構造を持つ天体への衝突計算を行いたい場合に参考になるだろう．

3.3.4. Collision2D (19 min)

空隙を持つ楕円球形状微惑星が空隙含有マントル，空隙無し岩石核を持つ真球形状微惑星に衝突する計算．空隙圧密モデルを計算に導入したい場合，衝突体形状を変更したい場合，天体中へ核を導入したい場合などに参考になる．

3.3.5. demo2D (31 sec)

本章で説明してきた asteroid.inp, material.inp はこの例題のものである．

3.3.6. dilatancy (310 min = 5.2 hours)

剪断歪がかかった際の粉体の挙動を記述する dilatancy model が使用されている．dilatancy model は Collins (2014)によって導入された．一般に砂のような粉粒体に剪断歪をかけると硬くなることが知られている．このとき粉粒体のバルク密度は変化しないが，比体積は大きくなる，つまり空隙率が上昇する．地球，月，火星などに観られるクレータには Free air 重力異常が観られることがあるが，これは地下構造が空隙を含むためである．標的地殻が dilatant fluid であると考えるとこのような重力異常を説明できる(Collins, 2014)．計算で形成されたクレータの重力異常を計算する解析スクリプト(porosity_gravity.py)も同梱されている．

3.3.7. Ice (>7 hours, TEND を小さくすることを推奨)

氷微惑星が内部海を持つ氷天体へ衝突する計算．水氷を計算に導入したいときに参考になるだろう．

3.3.8. landslide (204 min =3.4 hours)

S_TYPE = LANDSLIDE とする計算．地すべりを計算するのに適した計算セットアップになっている．2 次元デカルト座標系の計算例としても参考になる．なおこの例題には解析•描画用の python スクリプトが用意されていない．

3.3.9. MesoParticle2D (25 min)

S_TYPE = MESO_PART とする計算．Macro porosity の圧密を計算可能である[e.g., Davison et al., 2017]． 隕石組織との比較などを行いたい場合には有用である．この例題では 9 種類の物質を扱っており，複数物質を計算に取り入れたい際にも参考になる．なおこのモデルには additional.inp (asteroid.inp への追加ファイル)という入力ファイルが追加されている．isale.pbs の中を

```
time ./iSALE2D -i asteroid.inp -a additional.inp -M material.inp
```

と書き換えることによって additional.inp も実行ファイル(iSALE2D)に読み込ませている．

3.3.10. mesoscale2D (255 min = 4.3 hours)

MesoParticle2D と同様に mesoscale 計算を実施する．この例題では標的中に巨視的な pore(空隙)が整列している際の衝撃波伝播を計算する．空隙を含む物質に衝撃波が伝播する際の素過程を調べる際に有用である[e.g., Güidemeister et al., 2013].

3.3.11. planar_eulerian_2D (5.8 min) & planar_lagrangian_2D (2.3 sec)

いわゆる 1 次元平板衝突実験を模擬する計算．衝撃物理の勉強にも有用である．planar_lagrangian_2D では ALE_MODE=LAGRANGE が採用されている．

3.3.12. Planet2D (42 min)

直径 200 km の小惑星が月へ衝突する計算. S_TYPE = PLANET, GRAD_TYPE = CENTRAL が採用され，月中心重力場中の衝突計算を実施可能である.

3.3.13. Sand2D (433 min = 7.2 hours)

石英砂を用いた衝突実験の模擬計算. 数値計算の利点を活かし，重力加速度が 500 G (= 4,905 m s^{-2})に設定されている.

3.3.14. SelfGravity2D (21 min)

直径 4,800 km の原始惑星が原始地球に衝突する計算. S_TYPE = PLANET, GRAD_TYPE = SELF が採用されており，例題中で唯一自己重力を考慮した流体計算を実施可能である.

4. iSALE 計算実践編 -初級-

本章からは実際に入力ファイル asteroid.inp, material.inp を編集し, pySALEPlot で作図をしていく. 初級編は配布した asteroid.inp, material.inp から出発し, 衝突速度や EOS モデルを変更して計算を実施することにする. 合わせて pySALEPlot の基本操作を解説する. 4.1 節で必要ファイルを準備する. 4.2 節では iSALE の出力ファイル群について述べる. 4.3 節では pySALEPlot に関して対話式の iPython を使用して解説する. 4.4 節では読者自身で入力ファイル編集と作図を行う.

4.1. Elementary2D 準備

初級として「Elementary2D」という例題を準備した. `~/isale-work/isale2023cfca/Elementary2D` に移動しよう.

```
$ cd ~/isale-work/isale2023cfca/Elementary2D
```

続いて計算を実行する.

```
$ qsub isale_run.sh
```

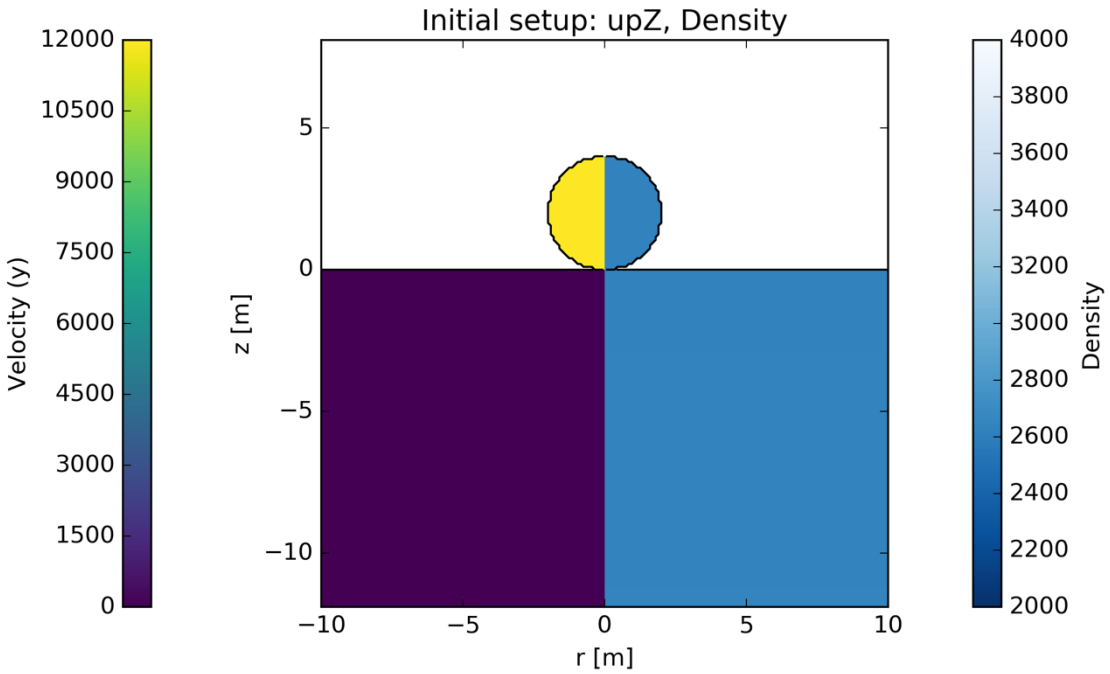
で計算を実行する.「qstat」で自身の Job が正常に流れているか確認しよう. 計算は数分で終了するはずである. その間にここで用意した入力ファイル, asteroid.inp と material.inp について簡単に解説する.

基本は「demo2D」と似ているが, material.inp を書き換え, Projectile と Target を独立に扱えるようにした. また簡単のために物質モデルをすべて OFF にして完全流体としてある. EOS には ANEOS granite(granit2)を選択している. asteroid.inp は計算領域を少し拡張し, Projectile 半径を 20 格子で分割するように変更した. またトレーサ粒子を各格子に 1 個挿入している. トレーサ粒子を使った解析については次章で取り扱うことにする.

計算が終了すると「Initial_UpZDen」ディレクトリが生成され, 初期条件が描画されているはずである.

```
$ display -resize 20% ./Initial_UpZDen/upZDen.png
```

で画像を開いて確認してみよう.



のような図が描けていれば準備完了である. Initial.py は入力ファイルを変更したときに自分の意図通りに変更されているかを確認する際に有用であろう. なお各自のネットワーク環境によってはこの動画の表示に長い時間がかかる可能性がある. その時にはひたすら待つか, 以下のコマンドで画像ファイル(*.png)を自分の手元の PC へ転送(scp)してそこで画像を見る方法がある.

```
$ scp -p isale00@grd0.cfca.nao.ac.jp:iSALE-work/isale2023cfca/Elementary2D/Initial_UpZDen/upZDen.png ./
```

4.2. iSALE の出力ファイル

これまでに iSALE を走らせると asteroid.inp で指定したディレクトリに計算結果群が格納され, Python スクリプトを走らせると計算結果を描画した画像が生成されることを学んできた. 本節ではここまで系統的には説明してこなかった

iSALE の計算出力の中身について特に筆者が有用であると考えるものを中心に解説する.

4.1 節の準備によって Elementary2D の下のディレクトリに Elementary2D というディレクトリが生成されているはずである. この中をみてみよう.

```
$ ls ./Elementary2D
```

```
INFO            SETUP           grid.txt   jdata.dat.jcerr   restart.dump.gz
PROGRESS        errors.txt0     jdata.dat  output.txt0
```

「jdata.dat」が iSALE の計算出力である. しかし, 通常のエディタでは開いても内容は理解できないであろう. このファイルを適切に解読し, フィルタを掛けてユーザが欲しい情報に加工するのが pySALEPlot である. pySALEPlot については次節で説明する. output.txt0 には計算が正常に実行されているかどうかの諸量が記録される. 現時点で計算終了までの達成度も確認できる.

```
$ tail -20 ./Elementary2D/output.txt0
```

などと入力してみよう. tail コマンドはファイルの末端から指定した行数を端末に表示させるコマンドである. 上記の例ではファイル末端の 20 行を表示する.

```
PERCENT COMPLETE:     XXX%
```

の部分が現時点までの達成度である. 計算開始からの経過時間とこの値から計算終了まであとどのくらいの実時間がかかるか概算できる.

「INFO」ディレクトリの中をみてみよう.

```
$ ls ./Elementary2D/INFO
```

```
asteroid.inp            material.inp            modelsetup.tex
setup_report.txt
```

「INFO」には計算に使用した入力ファイル asteroid.inp と material.inp がコピーされ格納されている. modelsetup.tex はこれらが LaTeX フォーマットで保存されて

いる[24]. setup_report.txt には入力ファイルをもとに構成した計算条件が全てまとめて記入されている．具体的には asteroid.inp や material.inp で顕に記入していないパラメータ群の値[25], EOS から計算された標準状態における諸量，計算領域の実空間距離，挿入したトレーサ粒子の位置及び総数，「Projectile」の諸量(半径，質量，運動エネルギー, ...)などが記入されており，意図した通りの計算設定が実現されているかどうかを確認する際に有用である．

続いて「SETUP」ディレクトリの中をみてみよう．

```
$ ls Elementary2D/SETUP
```

Tref_isotherm_granit2.txt energy.txt melttemp.txt temperature.txt
alpha.txt gravity.txt pressure.txt yield.txt
alpha_volstr_proj___.txt hugoniot-proj___-granit2-aneos.txt
pressure_yield_proj___.txt alpha_volstr_target_.txt
hugoniot-target_-granit2-aneos.txt pressure_yield_target_.txt
density.txt matdist.txt sound.txt

のように様々なファイルが格納されている．初期条件として計算された値がテキストファイルで保存されている．例えば pressure.txt には初期条件における深さ方向の静水圧構造, temperature.txt には初期温度構造が記述されている．また特に有用なのは EOS から計算された Hugoniot 曲線である．ここでは hugoniot-proj___-granit2-aneos.txt, hugoniot-target_-granit2-aneos.txt である．material.inp で入力した物質モデルに従って計算された Hugoniot 曲線(講義編テキストの図 3.1.1 参照)が格納されている．物質ごとの Hugoniot 曲線が出力されるので，この結果と 1 次元インピーダンス・マッチング法[e.g., Melosh, 1989, pp54–56]を用いて衝突直下点の最大圧力を求めることも可能である．

「PROGRESS」ディレクトリには計算中のタイムステップと，質量保存状況，エネルギー保存状況を記述したテキストファイルが格納されている．

[24] LaTeX ユーザにとっては論文などを書く際に有用でしょう．
[25] これらのパラメータには iSALE 開発チームの推奨値が自動的に入力されます. iSALE にはこのような「隠しパラメータ」が結構多いので，確認することをおすすめします．

4.3. pySALEPlot 概要

本節では iSALE の計算出力の解析に利用する pySALEPlot について解説する. pySALEPlot の本体は Python で書かれた pySALEPlot.py である. pySALEPlot.py は iSALE の計算出力である jdata.dat に適当なフィルタをかけて読み込むために使用できる他, 衝突計算に適したいくつかの関数(例えばクレータ直径, 深さの時間変化の抽出, ある位置にあるトレーサ番号の特定など)が定義されている. Python ベースとすることで Numpy(演算), Matplotlib(描画)といった優れた module(package)を呼び出して使用することができるという利点がある. iPython という対話型の User Interface が準備されており, 自身の解析スクリプトを作成する際に便利である. なお pySALEPlot について開発者の Tom Davison 博士が書いた簡易マニュアルも用意されている. そちらも熟読することをおすすめする.

pySALEPlot manual
http://isale-code.de/redmine/projects/isale/wiki/PySALEPlot_manual

以下では, iPython を使って pySALEPlot でデータを読み込み, Matplotlib で描画を行う一連の流れをみていこう. ***なお計算サーバのコマンドライン上での iPython の起動は禁止されているので絶対に行わないこと.*** 計算サーバで行う処理は必ず qsub コマンドを使った PBS ジョブとして投入することが定められている[26]。

1. まず解析サーバにログインする.

```
$ ssh -CY isale00@an00.cfca.nao.ac.jp
```
もしくは
```
$ ssh -CY isale00@an01.cfca.nao.ac.jp
```
(`isale00` は自身の講習会 ID に書き換える, 以下も全て同様)

2. Python で使用可能な package を読み込む[27].

[26] 今回の講習会専用機 `grd0` にはこの規約は適用されないものの, 将来的に一般利用者として計算サーバを使う日を考えると今からこの規約に慣れておくことが望ましいです.
[27] anaconda には上述の Numpy, Matplotlib といった module が含まれています.

```
$ module load anaconda/2 intel
```

3. ディレクトリ~/isale-work/demo2D/Plotting に移動する.

```
$ cd ~/isale-work/demo2D/Plotting
```

4. python と ipython の version を確認する.

```
$ python --version
```
Python 2.7.15 :: Anaconda, Inc.
```
$ ipython --version
```
5.8.0

となっていれば OK である.

5. iPython を起動する.

```
$ ipython --matplotlib
```
Python 2.7.15 |Anaconda, Inc.| (default, May 1 2018, 23:32:55)
Type "copyright", "credits" or "license" for more information.

IPython 5.8.0 -- An enhanced Interactive Python.
?   -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help   -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.
Using matplotlib backend: Qt5Agg

```
In [1]:
```

iPython で対話モードに入るとこのように In [X]: と表示される．やりとりの度に X が 1 つ増える．テキストに示されている番号や，講師が実演しているときの番号と，自分の PC 画面上の番号が一致していなくても問題はない．

6.「Import」コマンドで必要な module を読み込む．

```
In [2]: import sys
```

7. version を確認する．

```
In [3]: print sys.version
```

```
2.7.15 |Anaconda, Inc.| (default, May    1 2018, 23:32:55)
[GCC 7.2.0]
```

8. pySALEPlot を import する．

```
In [4]: import pySALEPlot as psp
```[28]

```
============================
        - pySALEPlot -
        by Tom Davison
============================
```

9. jdata.dat を Python の変数に受け渡す．

```
In [5]: model = psp.opendatfile('../demo2D/jdata.dat')
```

```
Opened iSALE data file './demo2D/jdata.dat', with 201 time steps
```

[28] import X as Y で，それ以降は Y と入力すると X が読み込まれます．

これで「model」という変数の中に./demo2D/jdata.dat を格納したことになる．もちろん変数名は変更できる[29]．

10．タブ補完機能を使い「model」に含まれている内容をみてみる．

```
In [6]: model.[TAB]
```

([TAB]はタブキーを押すという意味)

補完候補が複数でてくる．これらが変数 model に格納されている情報である．このように利用できる機能をタブ補完で調べることができるのが iPython を使うことの利点である．

11．試しに「model.inputDict」に何が書かれているか確認してみよう．

```
In [7]: print model.inputDict
```

{'DAMMOD': 'COLLINS', 'GAMETA': 0.008, 'DTDZSURF': 0.01, 'D_LITH': 80000.0, 'YLIMDAM': 2000000000.0, 'R_PLANET': 6370.0, 'BPTPRES': -1.0, 'YINT0': 10000000.0, 'YDAM0': 10000.0, 'TFRAC': 1.2, 'PATH': './', 'MODEL': 'demo2D', 'TEND': -10.0, 'DT': 0.001, 'ALE_MODE': 'EULER', 'BND_T': 'OUTFLOW', 'STRESS': 1, 'GRIDV': [35, 90, 15], 'OBJTPROF': 'CONST', 'GRIDEXT': 1.03, 'BDTPRES': -1.0, 'BND_B': 'NOSLIP', 'LAYTPROF': 'COND', 'OBJVEL': -6500.0, 'CSIMON': 3.0, 'DTSAVE': -0.05, 'LAYPOS': -85, 'STRMOD': 'ROCK', 'FRICINT': 1.1, 'PORMOD': 'NONE', 'GRAD_DIM': 2, 'GRIDH': [0, 80, 32], 'AVIS': 0.24, 'YLIMINT': 2500000000.0, 'BND_R': 'OUTFLOW', 'OBJNUM': 1, 'VARLIST': '#Den-Pre-Tmp-Yld-Dam-Ert-Vib-YAc-PVb-VEL#', 'OBJMAT': 'mygrani', 'POIS': 0.3, 'TMELT0': 1673.0, 'VERSION': 4.1, 'OBJTYPE': 'SPHEROID', 'GAMBETA': 115.0, 'ACFL': 'BLOCK', 'EOSTYPE': 'aneos', 'DIMENSION': 2, 'TOFF': 16.0, 'EOSNAME': 'granit1', 'VIB_MAX': 200.0, 'AVIS2': 1.2, 'GRAD_TYPE': 'DEFAULT', 'OBJRESH': 8, 'BND_L': 'FREESLIP', 'T_SURF': 293.0, 'LDWEAK': 'POLY', 'GRIDSPC': 100.0, 'S_TYPE': 'DEFAULT', 'THSOFT': 'OHNAKA', 'GRAV_V': -9.81, 'MATNAME': 'mygrani', 'LAYMAT':

[29] 異なる asteroid.inp, material.inp による計算出力の jdata.dat を別の変数に受け渡し，同時に解析を行うことも可能です．例えば衝突速度を変化させた計算を実施したときに両者を比較するなど．

'mygrani', 'DTMAX': 0.05, 'LAYNUM': 1, 'FRICDAM': 0.8, 'CVIB': 0.1, 'ASIMON': 6000000000.0, 'QUALITY': -50, 'GRIDSPCM': -20.0}

これらは asteroid.inp, material.inp に入力した内容である．例えば

```
In [8]: print model.inputDict['R_PLANET']
6370.0
```

とすれば asteroid.inp 中に記載した R_PLANET の値を呼び出すことができる．

12. pySALEPlot の「readStep」関数で任意のタイムステップの計算出力を呼び出せる．まずは初期条件を「step0」に格納してみよう．

```
In [9]: step0=model.readStep(0)
Read in ['Den'] for timestep -1 (0.000e+00 s)
```

変数を指定しない場合は密度(Den)の値が読み込まれる．別の物理量を読み込みたいときは

```
In [10]: step0=model.readStep(['Den', 'Pre'],0)
Read in ['Den', 'Pre'] for timestep 0 (0.000e+00 s)
```

のように指定すればよい．なお asteroid.inp の VARLIST に記載した物理量しか読み込めない．

13. step0 の中を覗いてみる．`step0.[VAR]`，もしくは `step.data[]` でその物理量の情報を呼び出せる．ここで[VAR]には物理量の名前(Den, Pre など), `data[]` には readStep 関数で読み込んだときの順番に整数を入れる．つまりこの例では `step0.Den = step0.data[0]` である．

```
In [11]: print step0.Den
[[2636.44921875 2636.33203125 2636.218017578125 ... -- -- --]
 [2636.44921875 2636.33203125 2636.218017578125 ... -- -- --]
```

```
 [2636.44921875 2636.33203125 2636.218017578125 ... -- -- --]
 ...
 [2636.44921875 2636.33203125 2636.218017578125 ... -- -- --]
 [2636.44921875 2636.33203125 2636.218017578125 ... -- -- --]
```

花崗岩(granit2)の標準密度が並んでいるはずである.

ここで

```
In [12]: print step0.Den[0,0]
```

2636.4492

などとすれば格子[0,0]に格納されている値を読むこともできる.

格子[0,0]の原点から測った実距離は model.x, model.y に格納されている.

```
In [13]: print model.x[0,0]
```

0.0

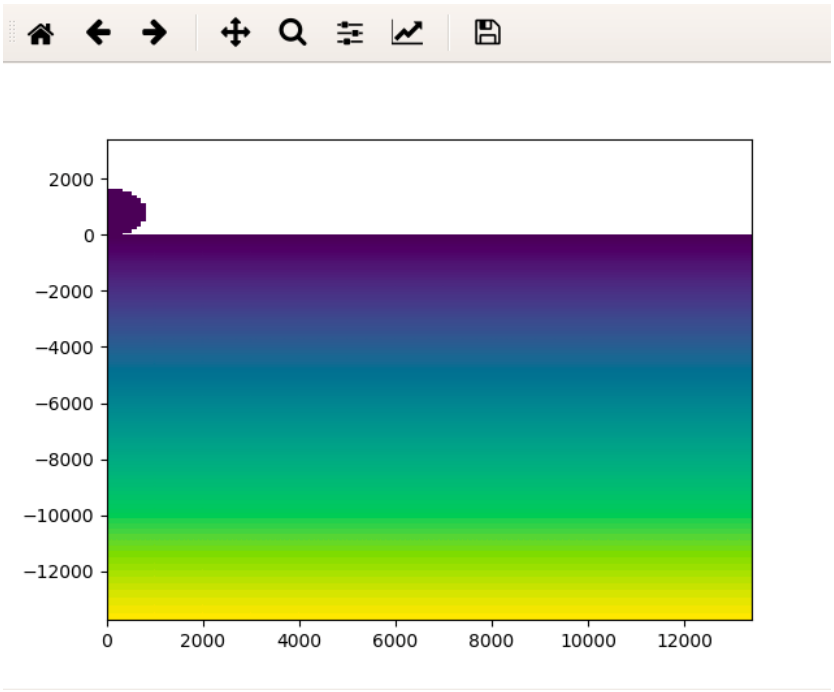
```
In [14]: print model.y[0,0]
```

-13727.5947265625

つまり[R, Z] = [0, -13.7 km]の位置における密度は 2,636 kg m^{-3} であることがわかる.

14. matplotlib を import し，描画をしてみよう.

```
In [15]: import matplotlib.pyplot as plt
In [16]: plt.pcolormesh(model.x,model.y,step0.Den)
```



上記のような図が現れれば成功である．以上で pySALEPlot による計算出力 jdata.dat の読み込み，フィルタによる特定の物理量の Python への受け渡し，任意のタイムステップのデータの取り出し，Matplotlib による描画の流れが完了する．これらを一つのファイルにまとめ，まとめて実行できるようにしたものが plot.py などの Python スクリプトである[30]．

ここで作成した図は論文に載せたり，研究発表の場に耐える質とは言い難いが，自分好みの図に改良していくところは本講習会の範疇ではないので，基本的には取り扱わない．Matplotlib による美しい図の作り方は web 上に山程の情報が蓄積されているので，そちらを参照してほしい．

4.4. asteroid.inp & material.inp の編集 & 描画

本節からはいよいよ読者自身が asteroid.inp, material.inp を編集し，自分好みの入力ファイルを生成していくことにしよう．作図用の Python スクリプト 3 種が Plotting ディレクトリ中に用意してある．

Initial.py

初期条件描画用．鉛直方向の粒子速度と密度を描画する．衝突速度や EOS モデルの変更が意図通りに行われているか確認する際に使えるだろう．

[30] gnuplot のユーザでしたら plt file の編集と同じ要領です．

PreDen.py

本スクリプトは Python の for 文を用いて，ループを生成し図を複数枚生成する．現象を時系列で調べる際に有用であろう．用意した asteroid.inp は t = 10 t_s [31]まで計算し，0.1 t_s ごとに計算結果を書き出す設定になっている．本スクリプトでは 1 t_s ごとの図を出力する．圧力と密度の時間変化を描画する設定になっているが，asteroid.inp の VARLIST で指定した中の任意の物理量に変更可能である．

Hugoniot_up-P-Den.py

./Elementary2D/SETUP 中の Projectile と Target の Hugoniot 曲線データを粒子速度–圧力平面上に描画する．Projectile の Hugoniot 曲線については反転 Hugoniot 曲線となっており，1 次元インピーダンス・マッチング法を可視化することができる．

適宜活用してほしい．

以下，自身の好きなように進めて構わないが，指示がないとわかりにくいという読者もいるかもしれないので，以下ではいくつかの方針を述べる．

(a). ファイル編集時には元ファイルを残しておく．

編集作業を重ねてエラーが出た場合に，元ファイルがないと原因を特定できないことがある．編集の際にはバックアップを残しておこう．これは iSALE 計算を進める際のデータ管理方法にも関係してくる．ここでは 2 通りの方法を紹介しておこう．1 つ目は「Elementary2D」をディレクトリごとコピーすることである．

```
$ cp –pr Elementary2D [HOGE]
```

([HOGE]には任意の名前をつける)

と cp コマンドに-r オプションをつけることで Elementary2D をコピーし，その中身を任意の名前のディレクトリに格納することができる．[HOGE]の中の

[31] Projectile の貫入特徴時間．式(3.1.6.3)

asteroid.inp, material.inp を編集することにすれば元ファイルとの比較も容易である.

2 つ目は Elementary2D の中で asteroid.inp, material.inp を別の名前をつけて複製し, iSALE2D に読み込ませることである. この場合, Elementary2D に移動し, 例えば

```
$ cp ./asteroid.inp ./ asteroid_[HOGE].inp
$ cp ./material.inp ./ material_[HOGE].inp
```

([HOGE]には任意の名前をつける)

などとし, さらに isale_run.sh の中の該当行を編集する.

```
time ./iSALE2D –i asteroid_[HOGE].inp –M material_[HOGE].inp
```

とすればよい. ただし, asteroid_[HOGE].inp 内の PATH と MODEL を元ファイルの asteroid.inp で指定したそれらとは変更しておく必要がある. これを忘れると計算結果が上書き保存されてしまう.

(b). 衝突速度を変更してみよう.

元の asteroid.inp では衝突速度が 12 km s^{-1} と設定されている. asteroid.inp の OBJVEL を好きな値に変更し, Initial.py で描画してみると, 速度が変更されていることを確認できるだろう. ここで isale_run.sh の中で

```
time ./iSALE2D –i asteroid.inp –M material.inp –C
```

のように-C オプションつけると, 初期条件の構成のみを行うことができる. 入力ファイルを編集する段階では有用である.

(c). EOS model を変更してみよう.

「eos」ディレクトリには使用可能な EOS model が格納されている. EOS model には標準密度の情報も含まれている. 例えば granite2 から iron___に変更すれば,

Initial.py で密度が変わることが明確にわかるであろう．また proj___と target_に異なる物質を割り当てることも可能である．

(d). Projectile size，形状を変更してみよう．

元の asteroid.inp では 20 CPPR が入力されている．OBJRESH を変更すると Projectile のサイズが変わる．

OBJRESH	CPPR horizontal	: 20
OBJRESV	CPPR vertical	: 40

などと入力すれば楕円球形状の Projectile を作成することができる．

ここまでできるようになれば，iSALE の基本的な操作は習得できているであろう．続いて PreDen.py を例に取って時系列変化を描画するスクリプトの内容を簡単に解説する．

1–15 行
必要な module を import する．最初の 3 行は計算サーバで python スクリプトを実行する際に必要である．

17–30 行
出力グラフの見栄えを変更するためのヒントを入れておいたが，デフォルトではコメントアウトされている．Python ではシングルクォーテーションを 3 つ並べて複数行をはさむと，複数行コメントアウトが可能である．なお単一の該当行のコメントアウトは文頭に“#”(シャープ)をつければよい．

32–34 行
解析結果を格納するディレクトリを指定する．

```
dirname = 'PreDen'
psp.mkdir_p(dirname)
```

任意の名前に変更可能である．好みの PATH 指定も可能である．

36–37 行
jdata.dat の中身を model に受け渡す.

```
model = psp.opendatfile('./Elementary2D/jdata.dat')
```

40–41 行
計算条件を確認する.

```
print model.modelInfo()
print model.tracerInfo()
```

./Elementary2D/INFO/setup-report.txt だけでなく pySALEplot からも計算条件を確認することができる. 必要なければコメントアウトすればよい.

43–44 行
空間スケールの単位を指定する.

```
model.setScale('m')
```

ここでは um, mm, cm, m, km を選択できる. 変更すると,「model」の中の実空間距離の単位が全てその単位に変換される.

例をあげると

model.x = 10,000 (for model.setScale('m'))
model.x = 10 (for model.setScale('km'))

となる.

46–53 行
「model」に格納された情報から Projectile 半径 R_p, 衝突速度 v_{imp}, 貫入特徴時間 t_s を計算する.

```
Grid_size      = model.inputDict['GRIDSPC']
CPPR           = model.inputDict['OBJRESH']
Rp             = Grid_size*CPPR
vimp           = -model.inputDict['OBJVEL']
```

```
ts                = 2.*Rp/vimp
```

このようにすると入力ファイルの情報を読み取って自動的に計算することができる. Projectile のサイズや衝突速度といった計算条件を変更した際に解析スクリプトを編集する必要がなくなる. また手入力にありがちな入力間違いを防止できる.

55–57 行
Matplotlib で図を作ることを宣言する.

```
fig = plt.figure(figsize=(12, 8))
ax = fig.add_subplot(111, aspect='equal')
```

figsize を変更すれば, 作成される画像サイズの大きさを変更できる.

59–71 行
図の左右に 2 つの独立したカラーバーを作成する関数を定義する.

```
def make_colorbars(ax, p, f, units):
    # Create axes either side of the plot to place the colorbars
    divider = make_axes_locatable(ax)
    cx1 = divider.append_axes("right", size="5%",pad=0.3)
    cx2 = divider.append_axes("left", size="5%", pad=0.8)
    cb1 = fig.colorbar(p[0], cax=cx1)
    cb1.set_label(psp.longFieldName(f[0])+units[0])
    cb2 = fig.colorbar(p[1], cax=cx2)
    cb2.set_label(psp.longFieldName(f[1])+units[1])
    # Need to set the labels on the left for this colorbar
    cx2.yaxis.tick_left()
    cx2.yaxis.set_label_position('left')
```

Python で関数定義を行うときの参考にもなるであろう.

73–76 行
時刻(ステップ)に対するループを宣言する.

```
iStart = 0
iLast    = model.laststep
for i in arange(iStart, iLast, 10):
```

今回の asteroid.inp では model.laststep = 100 である. arange(iStart, iLast, 10)は iStart から iLast までの等差数列を用意する関数である. 数列の値を順次整数 i に代入して処理を繰り返すという宣言をしている[32]. なお Python では for 文のループ範囲はインデント(半角スペース 4 つ)によって判断される. これ以降繰り返し処理させる部分は全てインデントをつけて記述する.

78–84 行
X 軸, Y 軸の値域, と軸ラベルの指定する.

```
    # Set the axis labels
    ax.set_xlabel('Radial distance (projectile radius)')
    ax.set_ylabel('Height (projectile radius)')

    # Set the axis limits
    ax.set_xlim([-7, 7])
    ax.set_ylim([-8, 6])
```

計算領域を変更したときは適宜変更する.

86–87 行
時刻ステップ i のデータを変数 step に読み込む.

```
    # Read the time step 'i' from the datafile
    step = model.readStep(['Pre', 'Den'],i)
```

89–93 行
読み込んだデータをプロットする.

[32] i = [0, 10, 20, 30, 40, 50, 60, 70, 80, 90]の 10 要素になります.

```
    # Plot the density field
    p1 = ax.pcolormesh(model.x/Rp, model.y/Rp, step.Pre*1e-9,
                cmap='magma', vmin=0, vmax=10)
    p2 = ax.pcolormesh(-model.x/Rp, model.y/Rp, step.Den*1e-3,
                cmap='YlGnBu_r', vmin=1.5, vmax=4)
```

model.x, model.y は各格子点の原点からの動径方向距離，鉛直方向距離である．ここでは衝突天体半径 R_p で規格化してあるが，実距離に変更したければ/R_p を削除すればよい．pcolormesh はこのような均一格子点の物理量を描画するのに適している．

95–100 行
物質境界線を追加する．

```
    [ax.contour(model.xc/Rp, model.yc/Rp, step.cmc[mat], 1, colors='k',
linewidths=2) for mat in [1, 2]]
    [ax.contour(-model.xc/Rp, model.yc/Rp, step.cmc[mat], 1, colors='k',
linewidths=2) for mat in [1, 2]]
```

model.xc, model.yc は原点からの測った各格子点中心の動径方向距離，鉛直方向距離である．step.cmc は物質インデックスであり，model.readStep で指定していなくても自動的に受け渡されている情報である．

102–105 行
カラーバーを設定する．さきほど定義した make_colorbar 関数を呼び出す．

```
    # Add a colourbar, but only need to do this once
    if i == 0:
        #Call the function
        make_colorbars(ax, [p1,p2], ['Pressure','Density'], [' (GPa)','(g/cc)'])
```

全ての図で共通のカラーバーを定義しておくと時間変化を読み取りやすい．この場合は i=0 で 1 度定義しておけばよいので If 文を使っている．Python で If 文を使った条件分岐をさせたいときはこのようにすればよい．

107 行
グラフタイトルを指定する.

```
ax.set_title('Scaled time t/ts = {: 5.2f} '.format(step.time/ts))
```

ここでは貫入特徴時間で規格化した時刻を指定している. もちろん, 衝突速度も合わせて記入するなど自由に変更可能である.

109–110 行
図を書き出す.

```
fig.savefig('{}/PreDen-step{:05d}.png'.format(dirname,i), dpi=100)
```

32–34 行で指定したディレクトリの中に PreDen-step[i]という名前で png ファイルが生成される. dpi の値も指定できる.

112–113 行
軸をリセットする.

```
# Clear the axis
ax.cla()
```

以上が PreDen.py で行わせている処理の流れである. ここまでを理解した上で./examples 中に含まれている iSALE 開発チームが作成した python スクリプトを読むとその内容がわかってくるであろう.

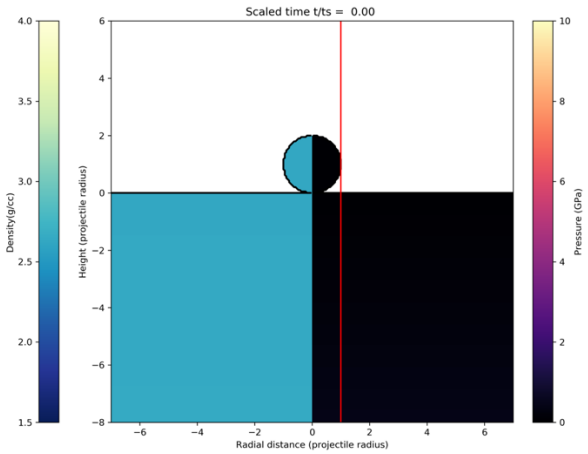
5. iSALE 計算実践編 -中級-

前章で計算結果のスナップショットを可視化する方法を習得できた．本章では pySALEPlot を用いて計算結果を解析していく方法を学ぶ．トレーサ粒子を活用して特定の条件を満たす物質の動きを可視化したり，粒子速度を定量化してみよう．また，トレーサ粒子の質量を計算し，重心位置を算出する方法を学んでみよう．中級課題として「Intermediate2D_1」と[Intermediate2D_2]を Slack で配布した．asteroid.inp, material.inp は「Elementary2D」で使用したものと概ね同じであるが, Intermediate2D_2 の asteroid.inp は球と球が衝突するように変更を加えている．初級編で用いた PreDen.py を元にして解析を行ってみよう．いち早く全ての課題をこなしてしまった場合は自分の思うような解析スクリプトの作成を進めるとよい．もしくは過去の講習会の課題に取り組んでもよいかもしれない．今年度には取り扱っていないいくつかの課題(経験した温度と累積質量分布，クレータ直径深さの時間変化，衝撃波面・膨張波面の可視化，最大衝撃圧力分布など)と解答用の python スクリプトが用意されている．

5.1.1 特定の座標の圧力分布の可視化

解答例スクリプト: TrP_fix_r.py

トレーサ粒子を活用することで，任意の場所にある物質の情報の可視化を行う．例えば，ある測線 $r = R_p$ (r は動径方向距離, R_p は衝突体半径)での圧力分布の描画を考える．



これは PreDen.py で描画した図上で，赤線上の情報を描画することに対応している．この線付近にあるトレーサ粒子を取得する．このときに便利なのが，Numpy に用意されている where 関数である．この関数を利用するためにまず

```
import numpy as np
```

を冒頭に記述し，numpy の関数を np.[関数名]として呼び出せるように設定する．where 関数は配列等(array_like)から条件に該当するものがどこ(where)にあるのかを取得する関数にあたる．トレーサ粒子が初期に全メッシュに載っているような状況であれば，例えば，

```
r_cut = 1.0
tracer_id = np.where( (step.xmark/Rp >r_cut*(1.0-2.5/CPPR) ) & (step.xmark/Rp <r_cut*(1.0+2.5/CPPR) ) )
```

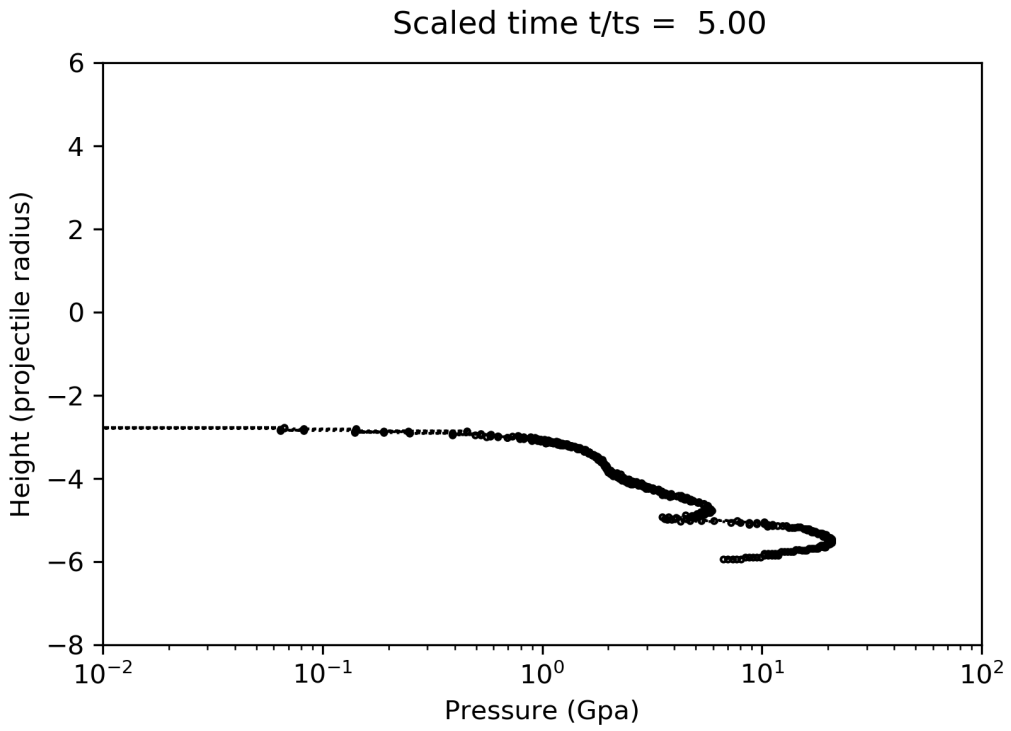
と，R_p で規格化したトレーサ粒子の位置(step.xmark/Rp)が，r_cut から 2.5 メッシュ以内にあるものの番号を取得することができる．例えば，

```
print(step.xmark[tracer_id]/Rp)
```

で，実際に r_cut 周りの粒子が取得されていることが確認できる．あとはこれらのトレーサ粒子の圧力情報(Trp)を

```
ax.plot(step.Trp[tracer_id]*1.e-9, step.ymark[tracer_id]/Rp, 'o', mec='k', c='none', ms=2 )
```

のようにして描画をしてやれば良い．



5.1.2　特定の座標の圧力分布と全体圧力分布との同時可視化

解答例スクリプト: Pre_Trp_fix_r.py

課題 5.1 では特定の座標の圧力分布を可視化した．本課題ではこれを全体の圧力分布と合わせた可視化を行う．このような図を並べることで，衝突後の状態と特定座標の圧力分布が比較され，よりわかりやすい可視化が可能となる．

PreDen.py の右図(圧力分布)と 5.1 の図を並列するため，まず図を分割する．

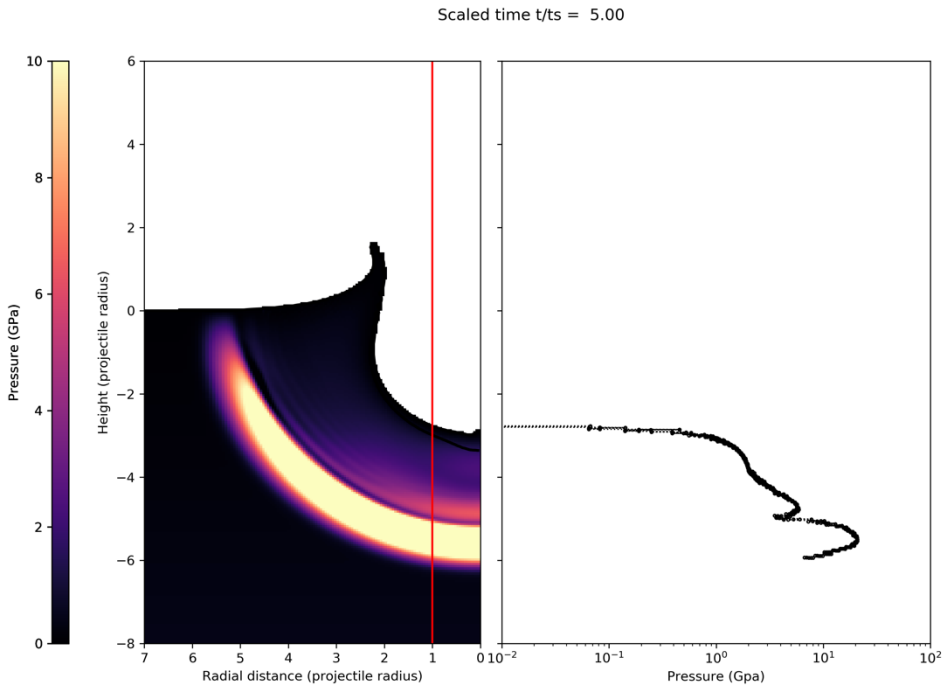
```
fig = plt.figure(figsize=(12, 8))
fig.subplots_adjust(wspace=0.05)
ax = fig.add_subplot(121)
ax2 = fig.add_subplot(122)
```

ここでは add_subplot を用いて，2 つ目の数字 2 で 2 列に分割し，3 つ目数字で ax を 1 列目，ax2 を 2 列目に分けた．1 列目の ax に関しては，PreDen.py をコピーし，Pre を可視化する部分だけ残す．このとき

```
ax.set_xlim([7, 0])
```

とするちょっとした工夫で，そのまま右側にあった圧力分布を左側に変え，細かな変更の必要なく流用できる．

次に ax2 として 5.1 のスクリプトを流用する．こちらは ax を ax2 に置換する必要がある．これを描画することで，下の図が得られる．



左図がつくことで，右図が衝突のどの波を記述したのかがより明解になっていることが見て取れる．

5.2.1 2 つの球の衝突，重心位置の可視化

解答例スクリプト: PreDen_CoM.py (CoM = Center of mass)

2 つの球形状天体が衝突することを考えよう．Intermediate2D_2 中の asteroid.inp は同一物質の 20 CPPR の球が 40 CPPR の球に 12 km s^{-1} で衝突するように設定した．従って質量比は 1:8 である．計算時間中に物質が計算領域外に出ないように計算領域は少し広げてある．本項では PreDen.py をもとにそれぞれの球の重心位置を計算し，描画できるようにしよう．

重心位置を計算するには個々のトレーサ粒子が担う質量を計算しておく必要がある． iSALE2D で円柱座標を選択している場合，個々のトレーサ粒子の質量は初期の動径方向距離に依存することに注意しよう．各時刻の図を出力するループに入る前に

```
step0=model.readStep('Trd',0)
```

```
Tracer_mass = np.zeros(num_total)
Tracer_mass = 2.*3.141592*step0.Trd*step0.xmark*Grid_size**2
```

とすればTracer_massに個々のトレーサ粒子が担う質量が格納される．この質量を以下のように足し上げれば Projectile, Target のそれぞれの球の質量を計算できる．

```
num_projectile = model.tru[0].end
num_total      = model.tracer_num
M_proj_0 = 0.0
for i in np.arange(0, num_projectile+1, 1):
    M_proj_0    += Tracer_mass[i]
M_target_0 = 0.0
for i in np.arange(num_projectile+1, num_total, 1):
    M_target_0 += Tracer_mass[i]
```

ここでpySALEPlotの関数であるmodel.truとmodel.tracer_numを用い，Projectile球中に配置された最後のトレーサ粒子の番号とトレーサ粒子の総数を抽出し，それぞれ num_projectile, num_total に格納している．トレーサ粒子の 0 から num_projectile のトレーサ質量を足し上げれば，Projectile球の質量となる．Target球についても同様に num_projectile+1 から num_total までを足し上げればよい．

続いてループの中で定義に従って重心位置を計算しよう．

```
Mass_weighted_Z_proj    = 0.0
    for Tracer_id in np.arange(0, num_projectile+1, 1):
        Mass_weighted_Z_proj
        += Tracer_mass[Tracer_id]*step.ymark[Tracer_id]
Z_CoM_proj = Mass_weighted_Z_proj/M_proj_0
R_CoM_proj = 0.0
```

のようにすればよい．標的についても同様である．なお，2 次元円柱座標系での計算の場合，重心位置の R 座標は常に 0 である．ここで一点注意を述べておく．

iSALE では数値誤差の伝播を抑えるため，計算中で高速になりすぎた物質を取り除いたり，低密度と判定された物質を取り除く処理を行っている．そのような格子に位置していたトレーサ粒子の R, Z 座標には異常に大きな値が挿入され，他の粒子と区別される．従って質量平均値が異常値に引きづられてしまい，重心位置を正しく計算することができない．またそもそものトレーサ位置自体の計算精度は high-resolution zone のみで保証される．そこで，質量平均値を求める際に if 文を挿入し，フィルタをかけるとよい．pySALEPlot に用意されている model.xhires, model.yhires 関数は high-resolution zone の境界位置を返してくれる．時間ループに入る前に

```
Rmax=model.xhires[1]
Zmin=model.yhires[0]
Zmax=model.yhires[1]
```

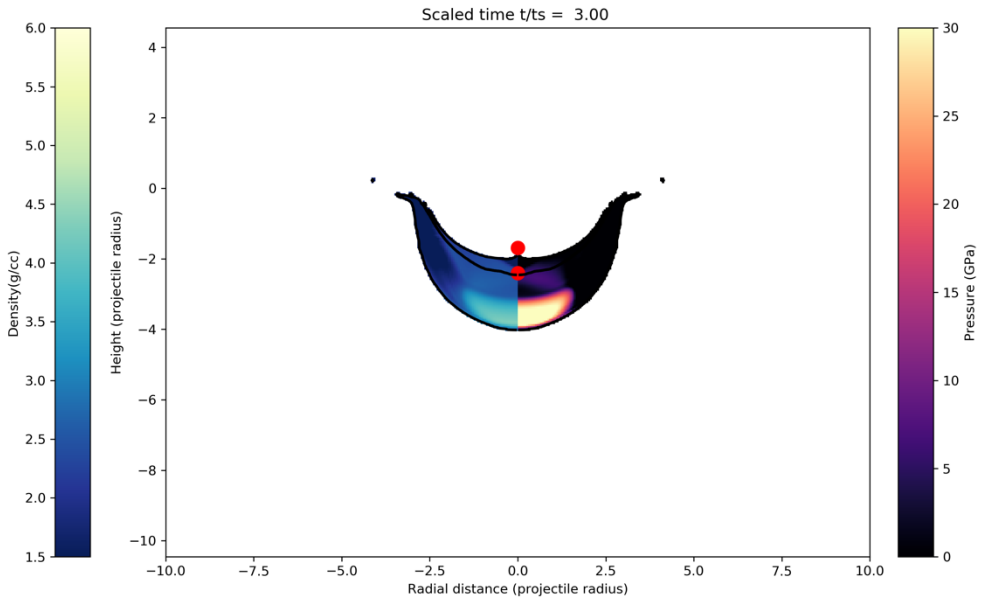
を追記し，high-resolution zone の境界位置を取得しておこう．その上で質量平均値を取るループ中に以下の if 文を追加しよう．

```
if((step.xmark[Tracer_id] <= Rmax)
& (step.ymark[Tracer_id] > Zmin)
& (step.ymark[Tracer_id] < Zmax)):
```

あとは

```
ax.plot(R_CoM_proj/Rp, Z_CoM_proj/Rp, 'ro', markersize=10, zorder=1)
ax.plot(R_CoM_target/Rp,Z_CoM_target/Rp, 'ro', markersize=10, zorder=1)
```

などとすれば，それぞれの球の重心位置を可視化することができる．



このような図が書ければ OK である.

5.2.2　2 つの球の衝突, 重心速度の定量化

解答例スクリプト: PreDen_2fields_Time-V.py

ここでは重心の移動速度を求めてみよう. 課題 5.1.2 と同様に課題 5.2.1 に 2 つ目のパネルを追加し, ax2 に時刻に対する重心の移動速度をプロットする. 図をつくる宣言をする箇所を

```
fig=plt.figure(figsize=(28,14))
fig.subplots_adjust(left=0.1,right=0.93,wspace=0.35)
ax=fig.add_subplot(121, aspect="equal")
ax2=fig.add_subplot(122, aspect="0.25")
```

のように改訂しておこう. 重心の移動速度を得るには先程と同様に個々のトレーサ粒子の Z 方向速度の質量平均を求めればよい. トレーサ粒子の粒子速度は TR_VAR で指定できる計算出力ではないので, pySALEPlot で計算する必要がある. 個々のトレーサ粒子の Z 方向速度 u_{pZ} は時間ループの中で

$$u_{pZ} = \frac{Z(t+\Delta t)-Z(t)}{\Delta t}, \quad \text{もしくは} \quad u_{pZ} = \frac{Z(t)-Z(t-\Delta t)}{\Delta t}$$

を逐一計算すれば求めることができる．Δt は asteroid.inp で設定した DTSAVE の値にとるのが最も精度よくトレーサ粒子の速度を計算できる．従ってトレーサ粒子の粒子速度が重要となる問題の場合は DTSAVE の値を小さくとっておくのが好ましい．時間ループに入る前のどこかに

```
dtsave          = model.inputDict['DTSAVE']
```

を追加する．DTSAVE に負の値を入れた場合(弾丸貫入特徴時間で規格化)にも対応できるように以下のような行を追加しておこう．

```
delta_time = dtsave
if(dtsave < 0):
    delta_time = -ts*dtsave
```

このようにすると Δt を自動的に計算することができる．続いてトレーサ粒子の Z 方向速度 u_{pZ} を計算しよう．例えば以下のような行を時間ループに入ったあとに追加する．

```
step_forward = model.readStep(['Pre', 'Den', 'Trd', 'Trp'],i+1)
Tracer_up_Z      = (step_forward.ymark - step.ymark)/delta_time
```

step_forward に i+1 step における計算出力ステップの諸量を格納することで，$Z(t+\Delta t)$を読み出し，u_{pZ} を計算している．あとは先程と同様に u_{pZ} の質量平均値を取ればよいが，計算から取り除かれたトレーサ粒子の u_{pZ} の計算値は異常値を取る場合があるので，if 文のフィルタに

```
if(Tracer_up[Tracer_id] < 2.0*vimp):
```

を追加しておこう．

時間に対する重心の移動速度を時々刻々追加するには，時刻ごとにそれぞれを配列に格納していくとよい．時間ループに入る前に

```
Time_in_calc                = []
up_CoM_proj_in_calc     = []
up_CoM_target_in_calc = []
```

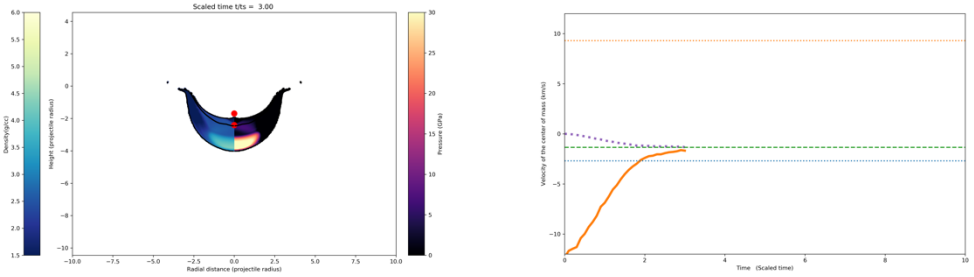
のような空の配列を定義する．そして，時間ループ中で

```
Time_in_calc.append(step.time/ts)
up_CoM_proj_in_calc.append(up_CoM_proj*1.0e-3)
up_CoM_target_in_calc.append(up_CoM_target*1.0e-3)
```

のように append コマンドを使用し，規格化時間，Projectile 球の重心移動速度，Target 球の重心移動速度を各計算出力ステップで格納していく．最後に

```
ax2.plot(Time_in_calc, up_CoM_proj_in_calc,      color='C1', linestyle='-',
    linewidth = 4.0, zorder=1)
ax2.plot(Time_in_calc, up_CoM_target_in_calc, color='C4', linestyle=':',
    linewidth = 4.0, zorder=1)
```

などとすれば右パネルに時間に対する重心移動速度の図が描かれる．



このような図を書ければ OK である．

5.2.3　2 つの球の衝突，密度–圧力平面上での挙動

数値衝突計算を行う場合，物質が相図上でどのような値を持っているのか調べたい場面が頻繁にある．ここでは課題 5.2.2 で作成したスクリプトを微修正し，

ax2 に個々のトレーサ粒子の密度–圧力平面上での動きを可視化してみることにしよう．相図上の動きを調べるときに Hugoniot 曲線は一つの基準になる．opendatfile 関数で jdata.dat を python に渡すところで合わせて SETUP フォルダ中の Hugoniot data も読み込んでおこう．

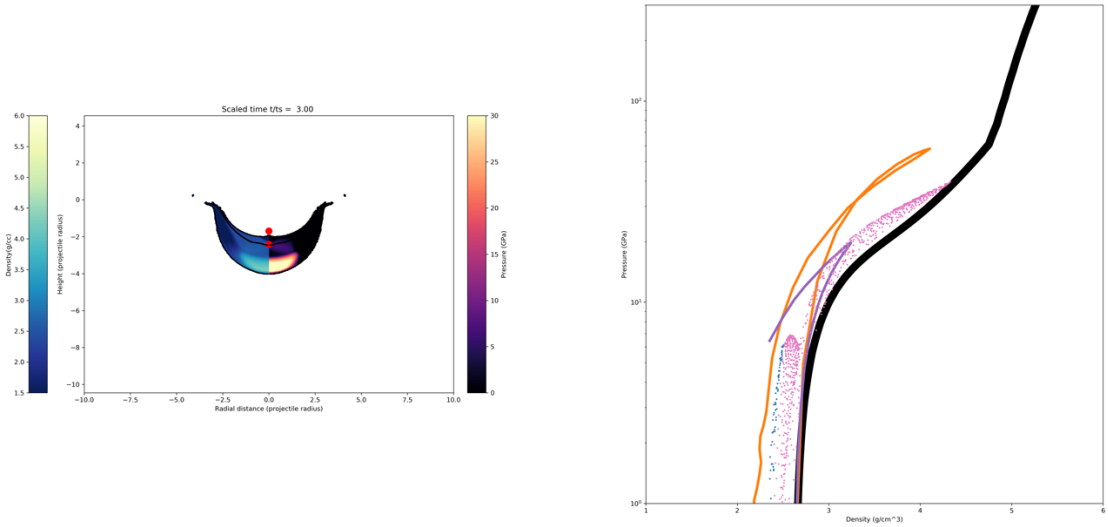
```
hugo = np.genfromtxt('./Intermediate2D/SETUP/hugoniot-target_-granit2-aneos.txt',usecols=(0,3,4))
```

とすると配列 hugo 中に Target 球の Hugoniot 曲線データが格納される．ここで列 0, 3, 4 は元ファイル中でそれぞれ密度，圧力，温度に対応している．hugo 中ではそれぞれ 0, 1, 2 列に密度，圧力，温度が記録される．例えば

```
print(hugo[:,2])
```

などとすれば Hugoniot 曲線の温度データを見ることができる．あとは ax2 に密度に対する圧力を描画すればよい．トレーサ粒子群を点としてプロットする際は scatter 関数を使うのがよい．

```
ax2.plot(hugot[:,0]*1.0e-3, hugot[:,1]*1.e-9, color='black', linestyle='-', linewidth = 12.0, zorder=1)
ax2.scatter(step.Trd[0:num_projectile]*1.0e-3,  step.Trp[0:num_projectile]*1.e-9, s=2, color='C0', zorder=2)
ax2.scatter(step.Trd[num_projectile+1:num_total]*1.0e-3,  step.Trp[num_projectile+1:num_total]*1.e-9, s=2, color='C6', zorder=2)
```



右パネルの黒線は granite の Hugoniot 曲線，青点が Projectile 球，ピンク点が Target 球のトレーサ粒子の密度–圧力である．ここでは課題 5.2.2 を参考に Projectile 球, Target 球それぞれの密度と圧力質量平均値の軌跡も描画している．

6. 宿題

6.1. 実践編 第 1 回目 宿題

1. examples 中の例題を実行せよ.

それぞれのディレクトリ中に isale_run.sh が置かれているので, 各ディレクトリまで移動し, qsub isale_run.sh で Job を投入可能である. examples/HOGE/Plotting (HOGE は各例題の名称)中に解析用 python スクリプト(拡張子.py)が格納されているので, 好きなものを選択し, isale_run.sh を適切に書き換えてから Job を投入すること.

grdo で各人が同時に走らせることができる Job 数の上限は 6 である. qstat で Job の状態を確認し, Job を投入してほしい.

作った図を自身の PC に転送したい場合は「scp」コマンドを使用する. 端末を新しく立ち上げ,

```
$ scp -pr isale00@grd0.cfca.nao.ac.jp:'~/iSALE-work/[HOGE]' ./
```

もしくは

```
$ scp -pr isale00@grd0.cfca.nao.ac.jp:iSALE-work/[HOGE] ./
```

とすると, 自身の PC の [HOGE]が ダウンロードされる. 適切にパス指定をする必要がある. ここでオプション-r をつけることでディレクトリごと再帰的に転送できる. オプション -p は転送元ファイルの更新時間やパーミションなどの情報を維持する. また emacs や vi ではなく手元の好みのエディタでファイルを編集したいときは, 上記のコマンドで手元にファイルをダウンロードしたあとに編集すればよい. 編集後に計算サーバへ転送するには

```
$ scp -pr ./[FILE] isale00@grd0.cfca.nao.ac.jp:~/iSALE-work/
```

とすれば, ./iSALE-work の中に[FILE]がディレクトリ構造も含めて転送される. ここも適当にパス指定をして欲しい. なお, 自分が今いるディレクトリのパス名を調べたいときは「pwd」コマンドを使えばよい.

6.2. 実践編 第 2 回目 宿題

1. /mwork2/isaleXX/isale-work/share/examples/demo2D 中の asteroid.inp を編集し，衝突天体形状や速度を変えて描画せよ．

ヒント: 計算の前に

```
$ cp -r demo2D demo2D_[HOGE]
```

などとして demo2D を demo2D_[HOGE]として複製しておくとよいだろう．

6.3. 実践編 第 3 回目 宿題

1. Projectile が金属核を持つように asteroid.inp, material.inp を変更し，計算結果を適切に描画せよ．

ヒント: ./examples/Collision2D が参考になるであろう．

2. Projectile と Target が金属核を持つ球形状天体となるように asteroid.inp, material.inp を変更し，計算結果を適切に描画せよ．

ヒント: 宿題 1 を元に標的天体にも金属核を挿入し，衝突するように OBJVEL を適切に設定する．COL_SITE と OBJOFF_V を適切に選ぶことで 2 球の衝突位置を任意の Z 座標に指定することができる．

3. 標的が異なる物質の多層構造になるように asteroid.inp, material.inp を変更し，計算結果を適切に描画せよ．

ヒント: ./examples/Chicxulub が参考になるであろう．

4. Projectile が半球殻形状になるように asteroid.inp を変更し，計算結果を適切に描画せよ．ここでははやぶさ 2 が実施した Small Carry-on Impactor による衝

突実験[Arakawa et al., 2020]を想定している．余裕があれば material.inp も適切に変更せよ．

ヒント：asteroid.inp 中で OBJMAT=VOID___と設定すると，何もない「空間」を配置できる．このとき material.inp は編集しなくてよい．真球を「空間」で「上書き」することによって SCI 衝突体のような半球殻形状を作ることができる．

6.4. 実践編 第 4 回目 宿題

1. /mwork2/isaleXX/isale-work/isale2023cfca/Elementary2D/Elementary2D/SETUP フォルダに格納されている Projectile と Target の Hugoniot data を用いて衝突直下点における最大衝撃圧力を推定せよ．

ヒント：1 次元インピーダンス・マッチング法 [Melosh, 1989, pp.54–57]を用いるとよい．横軸に粒子速度，縦軸に圧力を描画する．この際 Projectile の Hugoniot data の粒子速度は衝突速度からの差分とり，反転 Hugoniot 曲線を描画する．Projectile と Target の曲線の交点の Y 座標が最大衝撃圧力である．

解答例は Hugoniot_up-P-Den.py である．
なお物質を変えた場合, Hugoniot_up-P-Den.py 中の

```
hugop = np.genfromtxt('./Elementary2D/SETUP/hugoniot-proj___-granit2-aneos.txt',usecols=(0, 3, 7))
hugot = np.genfromtxt('./Elementary2D/SETUP/hugoniot-target_-granit2-aneos.txt',usecols=(0, 3, 7))
```

の行を適切に編集する(granite2 -> iron___など)必要がある．

2. 2020 年度中級課題を参考に各トレーサ粒子の初期位置における最大衝撃圧力を描画せよ．

ヒント：時刻 0 におけるトレーサ位置座標に対して，最終時刻における TrP(トレーサが経験した最大圧力)を描画する．

7. 謝辞

iSALE の主要開発メンバである, Gareth Collins, Kai Wünnemann, Boris Ivanov, H. Jay Melosh, Dirk Elbeshausen の各氏に深謝致します．また pySALEPlot を開発した Tom Davison 氏に御礼申し上げます．本テキストは過去の講習会及び, iSALE users group in Japan のみなさまとのやりとりで得られた知見，過去の講習会における講習内容への参加者の反応などを元にして執筆されました．いくつかの pySALEPlot スクリプトは脇田茂氏に作成していただいたサンプルを元にしました. iSALE 講習会 2023 参加者用 ML は千秋博紀氏に作成して頂きました．皆様に謝意を表します．最後に講習会開催に向けご尽力頂いた国立天文台天文シミュレーションプロジェクトの皆様，特に講習会専用サーバ群の準備をしていただいた波々伯部広隆氏，受講者への事細かな対応を行っていただいた加納香織氏に感謝申し上げます．

補遺

A. ファイル転送ソフトウェア Cyberduck

CfCA の計算&解析サーバと手元の PC の間のファイル転送は scp コマンドで実行する．しかし，計算機に不慣れな読者の場合はファイル転送アプリを使用したほうが簡便であるかもしれない．ここではその 1 例として Windows, Mac ともに対応している「Cyberduck」を紹介する．

Cyberduck
https://cyberduck.io

以下，導入と使用手順を述べる．

1. Cyberduck を DL して，自身の PC にインストールする．

2. 国立天文台に VPN 接続する．

3. Cyberduck を立ち上げ，
   接続方法: SFTP 接続
   サーバ: grdo.cfca.nao.ac.jp
   ユーザ名: isaleXX [XX は自身の講習会 ID に変更]
   パスワード: NIS パスワード
   ポート番号: 22
   と入力し，「接続」をクリック．

以上で，自身の PC 上でファイルやディレクトリをマウスポインタでつまんでやり取りするのと同じ感覚でファイル転送を行うことができる．

B. Python 用対話型開発環境 JupyterLab

JupyterLab はブラウザ内でスクリプトの編集，実行，描画を全て行うことができる Python 用対話型開発環境のことである．読み込んだ module 類の機能を tab 補間で探すことができ，開発に適している．中級課題では講師が JupyterLab を使用して実演するので，各自で環境を整えておくこと．以下に環境整備の手順を述べる．

1. 国立天文台に VPN 接続する．

2. 以下のコマンドで「解析サーバ」にログインする．

```
$ ssh -L 9XXY:localhost:9XXY isaleXX@an00.cfca.nao.ac.jp
```

or

```
$ ssh -L 9XXY:localhost:9XXY isaleXX@an01.cfca.nao.ac.jp
```

ここで XX は自分の講習会 ID，Y は 0 から 9 までの任意の数字である．

3. 解析サーバにログイン後、以下を打ち込み，実行

```
$ module load intel anaconda/2
```

4. 以下のコマンドを実行し，JupyterLab を起動

```
$ jupyter lab --no-browser --port=9XXY
```

ここで 9XXY は手順 2 の ssh コマンドで使用した番号と合わせること．

5. 表示される以下のコメントに続く URL をブラウザ(Google Chrome を推奨)に貼り付ける．

Copy/paste this URL into your browser when you connect for the first time,
to login with a token:
http://localhost:9XXY/?token=6c17bcfd5c6169f371ccb28cf9a25e924dd29e8d945bd4c2

図 A1 のような画面が表示されれば JupyterLab が正常に起動している.

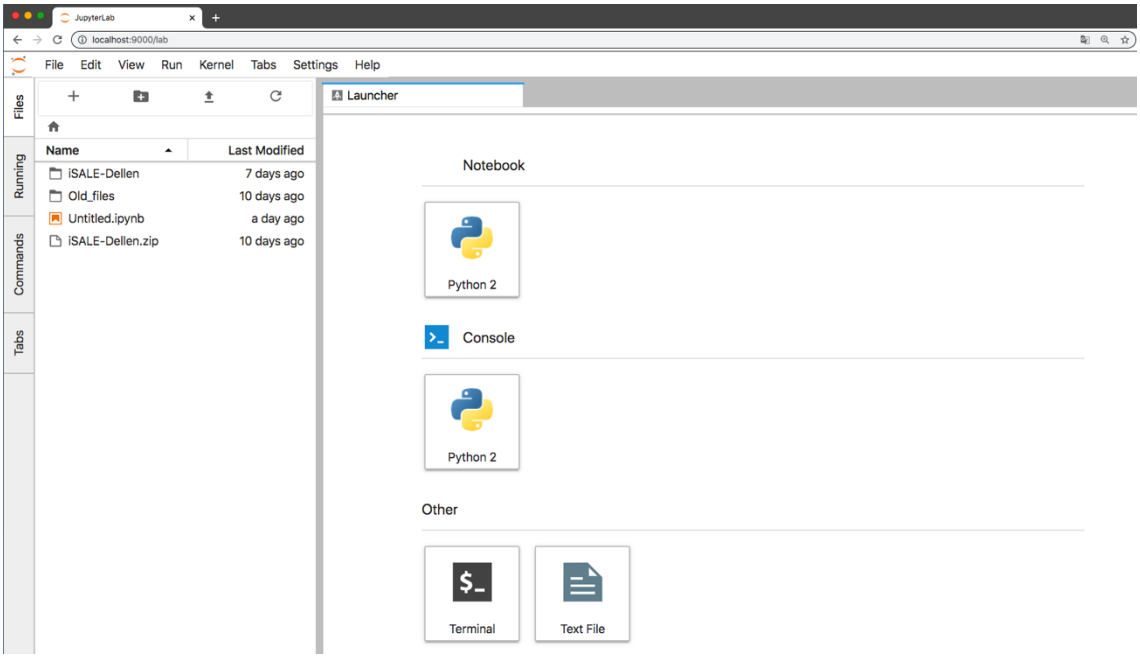


図 A1. JupyterLab の起動画面. このような画面が現れれば正常に起動している. このような画面が現れない場合は原因を追求し, 解決すること.

以下, よくある 2 つのトラブルとその解決方法を述べる.

a. URL で表示される番号(localhost:9XXY)が ssh の時の番号と異なっている. 一度解析サーバからログアウトして、表示された番号を使って再度 ssh(手順 2 に戻る)する. ssh の番号と URL の番号が一致するまで繰り返す. 例えば 9XXY = 9000 のときに localhost:9001 などと表示された場合は一度解析サーバからログアウト(exit コマンド)し,

```
$ ssh -L 9001:localhost:9001 isaleXX@an00.cfca.nao.ac.jp
$ module load anaconda/2 intel
$ jupyter lab --no-browser --port=9001
```

とする.

b. ブラウザでうまく表示されなかった場合,
　–ssh で使用した番号と URL の番号が一致しているか確認する.
　–ブラウザを変えてみる(Safari や Edge).
　–プライベートモードで試してみる.

上記でうまくいかない場合, 手順 4 で以下のコマンドを実行してみよう.

```
$ jupyter notebook --no-browser --port=9XXY
```

JupyterLab ではなくその 1 世代前の Jupyter Notebook が起動する. Tab 補完などの機能は同様に使用することができる.

以上でうまくいかない場合は ML, もしくは Slack にて講師に連絡すること. その際, 自身で打ち込んだコマンドとエラーメッセージをそのままコピペして付すこと.

参考になる websites

CfCA HP

https://www.cfca.nao.ac.jp

CfCA 共同利用計算機の利用資格・利用申請資格

https://www.cfca.nao.ac.jp/node/2

iSALE HP (本家)

https://isale-code.github.io

iSALE users group in Japan wiki

https://www.wakusei.jp/~impact/wiki/iSALE/

pySALEPlot manual(再掲)

http://isale-code.de/redmine/projects/isale/wiki/PySALEPlot_manual

計算サーバの詳細，及び利用規約(再掲)

https://www.cfca.nao.ac.jp/node/165#regulation

解析サーバの詳細，及び利用規約(再掲)

https://www.cfca.nao.ac.jp/node/22#policy

参考文献

Amsden, A.A., Ruppel, H.M., Hirt, C.W., 1980. A simplified ALE computer program for fluid flow at all speeds. *Los Alamos National Laboratories Report LA-8095*. Los Alamos, New Mexico. 101 pp.

Collins, G.S., 2014. Numerical simulations of impact crater formation with dilatancy. *Journal of Geophysical Research - Planets* **119**, 2600–2619.

Collins, G.S., Elbeshausen, D., Davison, T.M., Wünnemann, K., Ivanov, B., Melosh, H.J., 2016. iSALE-Dellen manual. *figshare*, https://doi.org/10.6084/m9.figshare.3473690.v2.

Collins, G.S., Patel, N., Davison, T.M., Rae, A.S.P., Morgan, J.V., Gulick, S.P.S., IODP-ICDP Expedition 364 Science Party, 2020. A steeply-inclined trajectory for the Chicxulub impact. Nature Communications 11:1480, https://doi.org/10.1038/s41467-020-15269-x

Davison, T.M., Derrick, J.G., Collins, G.S., Bland, P.A., Rutherford, M.E., Chapman, D.J., Eakins, D.E., 2017. Impact-induced compaction of primitive solar system solid: The need for mesoscale modelling and experiments. Procedia Engineering 204, 405-412, 10.1016/j.proeng.2017.09.801

Güldemeister, N., Wünnemann, K., Durr, N., Hiermaier, S., 2013. Propagation of impact-induced shock waves in porous sandstone using mesoscale modeling. *Meteoritics & Planetary Science*, 48(1): 115–133.

Holsapple, K.A., Schmidt, R.M., 1982. On the scaling of crater dimensions: 2. impact processes. J. Geophys. Res. 87, 1849–1870. https://doi.org/10.1029/JB087iB03p01849.

Ivanov, B.A., de Niem, D., Neukum, G., 1997. Implementation of dynamic strength models into 2D hydrocodes: application for atmospheric break-up and impact cratering. *Int. J. Impact Eng.* **17**, 375–386.

Melosh, H. J. 1979. Acoustic fluidization: A new geologic process? *J. Geophys. Res.* **84**, 7513–7520.

Melosh, H.J., 1989. *Impact Cratering — A Geologic Process*. Oxford University Press, New York, pp. 112–125. Chap. VII.

Schulte, P. et al. The Chicxulub asteroid impact and mass extinction at the Cretaceous-Paleogene boundary. Science 327, 1214–1218 (2010).

Suzuki, A.I., et al., 2012. Laboratory experiments on crater scaling-law for sedimentary rocks in the strength regime. J. Geophys. Res. 117, E08012. https://doi.org/10.1029/2012JE004064.

Wünnemann, K., Collins, G.S., Melosh, H.J., 2006. A strain-based porosity model for use in hydrocode simulations of impacts and implications for transient crater growth in porous targets. *Icarus* **180**, 514–527.