

## 引き継ぎ資料

# Simultaneous Faint Meteor Observations using MU Radar Head Echo and 105-cm Schmidt Telescope equipped with Tomo-e Gozen Camera

Akira Hirota

March 30, 2019

### NOTE

引き継ぎ用のレポジトリ (<https://github.com/akira-hirota1994/handover>). 中身は前回の引き継ぎ会のとおり (容量の大きいファイルはアップロードできないので除く).

# Contents

<b>I</b>	<b>はじめに</b>	<b>2</b>
<b>1</b>	<b>使用するプログラム, ソフトウェア等</b>	<b>2</b>
1.1	Python . . . . .	2
1.1.1	Astropy . . . . .	2
1.1.2	Jupyter Notebook . . . . .	2
1.2	Matlab . . . . .	3
1.3	TOPCAT . . . . .	3
1.4	SAO Image DS9 . . . . .	3
<b>2</b>	<b>ディレクトリ</b>	<b>4</b>
<b>II</b>	<b>マニュアル</b>	<b>6</b>
<b>1</b>	<b>解析の全体的な流れの確認</b>	<b>6</b>
1.1	手順 . . . . .	6
1.2	shell スクリプトに関して . . . . .	10
1.3	各プログラムの簡単な説明 . . . . .	11
1.3.1	<a href="#">mu-tomoe_analysis.ipynb-セクション 1</a> . . . . .	11
1.3.2	<a href="#">mu-tomoe_analysis.ipynb-セクション 2</a> . . . . .	11
1.3.3	<a href="#">mu-tomoe_analysis2.ipynb-セクション 1</a> . . . . .	11
1.3.4	<a href="#">mu-tomoe_analysis2.ipynb-セクション 2</a> . . . . .	11
1.3.5	<a href="#">mu-tomoe_analysis2.ipynb-セクション 3</a> . . . . .	12
1.3.6	<a href="#">photometry_meteor.m</a> . . . . .	12
1.3.7	<a href="#">xml_gaia.ipynb, xml_ucac.ipynb</a> . . . . .	13
1.3.8	<a href="#">photometry_gaia.ipynb, photometry_ucac.ipynb</a> . . . . .	13
1.3.9	<a href="#">gband2vband.ipynb</a> . . . . .	14
1.3.10	<a href="#">mu-tomoe_analysis3.ipynb-セクション 1</a> . . . . .	14
1.3.11	<a href="#">mu-tomoe_analysis3.ipynb-セクション 2</a> . . . . .	14
<b>2</b>	<b>解析以外のプログラム</b>	<b>15</b>
2.1	<a href="#">/iccd_2018/population_index/population_index.ipynb-セクション 1</a> . . . . .	15
2.2	<a href="#">/iccd_2018/population_index/population_index.ipynb-セクション 2</a> . . . . .	15
2.3	<a href="#">/iccd_2018/population_index/mass_size_distribution_function.ipynb</a> . . . . .	15
<b>III</b>	<b>TOPCAT のグラフ</b>	<b>17</b>
<b>1</b>	<b>プロジェクションや散布図</b>	<b>17</b>
<b>2</b>	<b>ヒストグラム</b>	<b>20</b>

## Part I

# はじめに

## 1 使用するプログラム,ソフトウェア等

### 1.1 Python

主にカラムの演算や座標変換に使用. バージョンは 2.7

通常 python スクリプトの拡張子は.py ですが, 次のエディターをしているため.ipynb となっています. 中身は普通の python と同じ.

#### 1.1.1 Astropy

天文系に特化した python ライブラリで非常に出番が多い.fits ファイルを numpy 配列に変換, 作成することも可能. 単位や座標変換, 拡張ライブラリをインポートすることで測光 (アパーチャー測光) も可能.

#### 1.1.2 Jupyter Notebook

python スクリプトの作成に使用したエディター.

python の対話型という特徴を残しつつ普通のエディターのように, ある程度の長さのプログラムも書くことができます. 簡単な例を以下に示すと, 最上段のように一つのセクションでプログラムが終了している場合も可能ですが, jupyter notebook は現在開いているノート内で変数が保存されるため最下段に記してあるように手前のセクションで定義した変数を使うことができます. また同様に幾つもあるスクリプトなどを一つのノートブックにまとめることができるのが最大の利点です.



```
In [4]: import numpy as np

        test1 = np.array([1,2,3])

In [5]: test2 = "This is test"

In [6]: print test1,test2

[1 2 3] This is test
```

## 1.2 Matlab

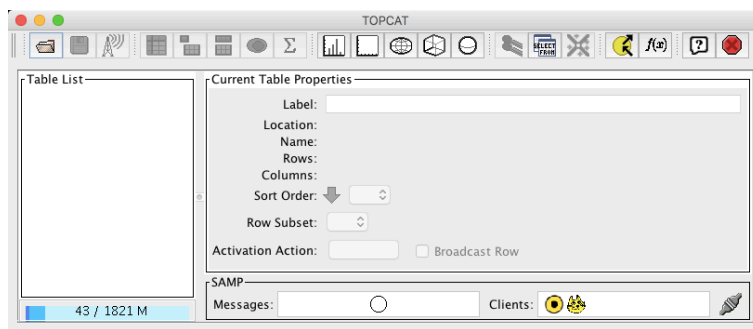
画像処理に使用. 特に説明は不要かと思います.

## 1.3 TOPCAT

(Tool for OPERations on Catalogues And Tables ; TOPCAT)

数百 MB ぐらいまでの大きいデータテーブルまで扱うことができ, 以下のような GUI ソフトウェア. カラム間の演算やデータの可視化が可能です. グラフの作成は後述.

2次元, 3次元だけでなく極座標系, 天球座標などの散布図, ベクトルマップ, ヒストグラム, ヒートマップなどプロットの幅はかなり広いです.



## 1.4 SAO Image DS9

GUI ベースの画像処理ソフト.XPA(X Public Acces) をインストールすることでコマンドラインから ds9 を操作できるようになるので恒星のカタログ等級の取得の際に使用.

## 2 ディレクトリ

ホームディレクトリ (handover) より下のディレクトリ構造は以下に示す通り.  
いくつかのプログラムは別のディレクトリ内にあるデータをインポートしているので移動したり, 名前を変更する際は要注意.

Listing 1: ディレクトリ樹形図

```
1 handover/  
2 +-- Kiso_observation  
3 | +-- Fujiwara@MU  
4 | +-- Fujiwara@Osaka  
5 | +-- Ueda@Osaka  
6 +-- Tomo-e_pipeline  
7 | +-- 201804  
8 |     +-- d_csv_gaia  
9 |     +-- d_csv_gaia_used  
10 |     +-- d_csv_ucac  
11 |     +-- d_csv_ucac_modified  
12 |     +-- d_csv_ucac_used  
13 |     +-- d_fits  
14 |     +-- d_lightcurve  
15 |     +-- d_photometry_gaia  
16 |     +-- d_photometry_meteor  
17 |     +-- d_photometry_ucac  
18 |     +-- d_xml_gaia  
19 |     +-- d_xml_ucac  
20 +-- d_criterion  
21 | +-- d_planet  
22 | +-- d_txt  
23 +-- iccd_2018  
24 | +-- population_index  
25 | | +-- d_data  
26 | | +-- d_img  
27 | +-- ufo_xml  
28 +--mu_tomoe_scripts-master
```

### handover/Kiso-Observation/

2018.4.18-22 のカメラ観測の UFOAnalyzer 解析の csv ファイル. 阿部研究室の  $\alpha$  7S と上田さん, 藤原さんから頂いたデータ.

### handover/Tomo-e\_pipeline/

Tomo-e データの解析用ディレクトリ, データリダクション以降 測光解析までのスクリプトやグラフ, データが諸々入っています.

### handover/d-criterion/

d-criteiron を使って流星群軌道を調べるスクリプトと gnuplot のスクリプトが入っています.

### handover/iccd\_2018/

卒論のデータ (iccd+MU) や 2009-2015 年までの MU のデータ, フラックス, population index, 分布関数のグラフを作ったスクリプトなどが入っています.

**handover/mu\_tomoe\_scripts-master**

daniel のスクリプト, ビームモデルや collection area の計算スクリプトが入っています.  
Github からダウンロード可能([https://github.com/danielk333/mu\\_tomoe\\_scripts](https://github.com/danielk333/mu_tomoe_scripts)).

## Part II

# マニュアル

### 1 解析の全体的な流れの確認

基本的にカラム演算 測光までは全て/Tomo-e\_pipeline/201804/ディレクトリ内で作業を行います。

はじめに単純なベクトル計算,RaDec から AzEl への座標変換,MU と Tomo-e の検出のオーバーラップなどの計算を行います。

データリダクション済みのデータ (simultaneous\_event\_candidates\_table\_MU+Tomo-e3.csv) から平面内での流星のベクトルを求め,MU レーダーでディテクトされた流星との距離, 内積の計算を行います。

続いて測光.shell スクリプトを用いて恒星カタログ (Gaia DR2) 情報を抜き出し,Matlab スクリプトで流星の画像処理と線密度 (Line Intensity), フラックスの計算を行います。

最後に恒星のアパーチャー測光値と流星のフラックスから等級を決定します。

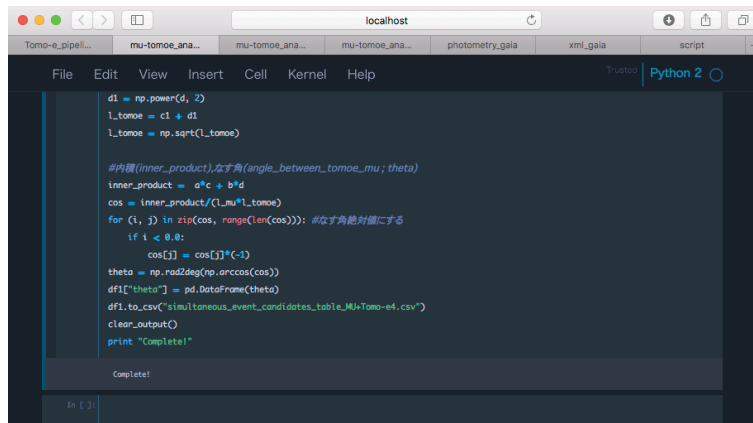
#### 1.1 手順

データのリダクションを行なったあと。

simultaneous\_event\_candidates\_table\_MU+Tomo-e3.csv が手元にある状態から最終的に欲しい流星の等級が得られるまでの手順を一通りなぞっていきます。

1. mu-tomoe\_analysis.ipynb を実行。

セクション1で中間ファイル(simultaneous\_event\_candidates\_table\_MU+Tomo-e4.csv) が出力され, セクション2では sensor.csv が出力されます。jupyter の画面上で実行した後”complete!”と表示されれば動いています。また, 基本的な操作は調べた方が早いと思いますが,Shift+Enter で選択しているセクションのプログラムを実行します。



```

d1 = np.power(d, 2)
l_tomoe = c1 + d1
l_tomoe = np.sqrt(l_tomoe)

#内積(inner_product),なす角(angle_between_tomoe_mu; theta)
inner_product = a*c + b*d
cos = inner_product/(l_mu*l_tomoe)
for (i, j) in zip(cos, range(len(cos))): #なす角絶対値にする
    if i < 0.0:
        cos[j] = cos[j]*(-1)
theta = np.rad2deg(np.arccos(cos))
df1["theta"] = pd.DataFrame(theta)
df1.to_csv("simultaneous_event_candidates_table_MU+Tomo-e4.csv")
clear_output()
print "Complete!"

```

Complete!

2. mu-tomoe\_analysis2.ipynb を実行。  
セクション 1,2 と連続で実行すると中間ファイル

(simultaneous\_event\_candidates\_table\_MU+Tomo-e5.csv) が出力されます。  
 セクション3を実行すると中間ファイル  
 (simultaneous\_event\_candidates\_table\_MU+Tomo-e6.csv) が出力されます。

### 3. photometry\_meteor.m を実行.

2つの中間ファイル(simultaneous\_event\_candidates\_table\_MU+Tomo-e7.csv と simultaneous\_event\_candidates\_table\_MU+Tomo-e8.csv) が出力されます。  
 テーブル7はチェック用のファイルとして作ったのでいいです。  
 ソースファイルの先頭に setting が何行か書いてあるので、初回は以下のようにする

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% section : SETTING

%TF = readtable("recheck_photometry.csv");
%name = TF(:, "ID"); name = table2array(name);
%for k = 1:size(name)
    %ID_number = name(k,1);
    %ID_number = string(ID_number);

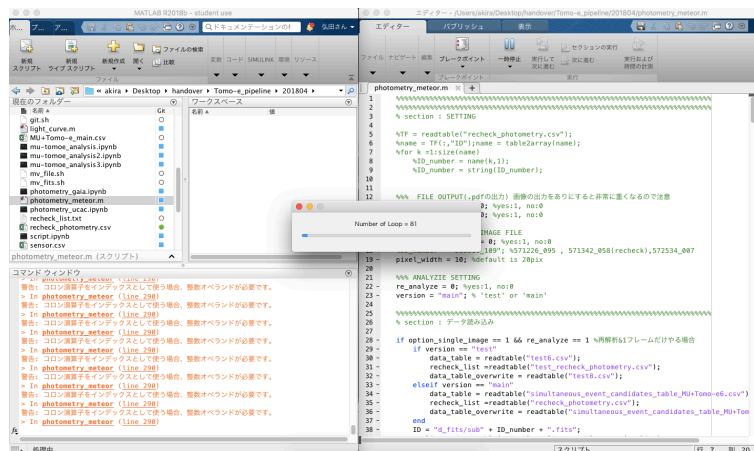
%% FILE OUTPUT(.pdfの出力) 画像の出力をありにすると非常に重くなるので注意
option_show_image = 0; %yes:1, no:0
option_save_image = 0; %yes:1, no:0

%% ANALYZE SINGLE IMAGE FILE
option_single_image = 0; %yes:1, no:0
%ID_number = "586316_109"; %571226_095 , 571342_058(recheck), 572534_007
pixel_width = 10; %default is 20pix

%% ANALYZE SETTING
re_analyze = 0; %yes:1, no:0
version = "main"; % 'test' or 'main'

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

実行中は進行バーがあるのでしばらく待機. たまにエラーメッセージで "This frame is out of range." と出るがとりあえず無視しても問題ない. ちゃんと再解析できるようになっている.



### 4. ds9 を使用して各フレームごとに恒星カタログを抜き出してくる.

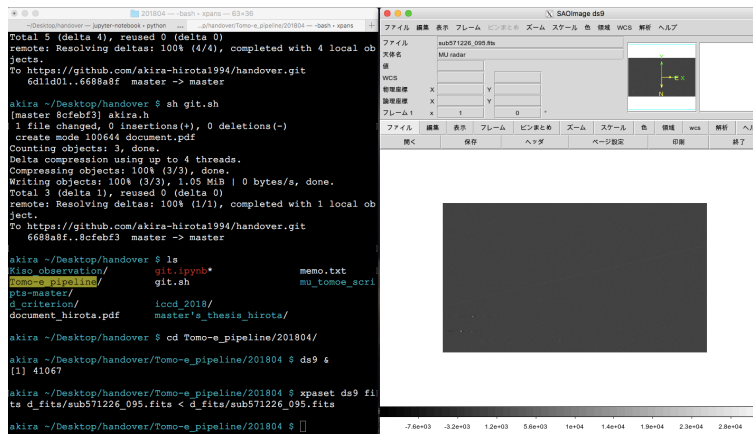
ターミナルから ds9 を起動し, 画像 (fits ファイル) を開き, 恒星カタログと参照しデータの



入った xml ファイルを出力するという流れを全フレームに対して行います。  
 コマンドライン上での ds9 の操作は xpa が必要になるのでインストールが必要です。  
 画像を開く (試しに sub571226\_095.fits を開いて見ます) にはコマンドラインで以下を打ち込みます (いま, いる場所は Tomo-e\_pipeline/201804/ です.)

```
akira ~/Desktop/handover/Tomo-e_pipeline/201804 $ xpa ds9 fi
ts d_fits/sub571226_095.fits < d_fits/sub571226_095.fits
```

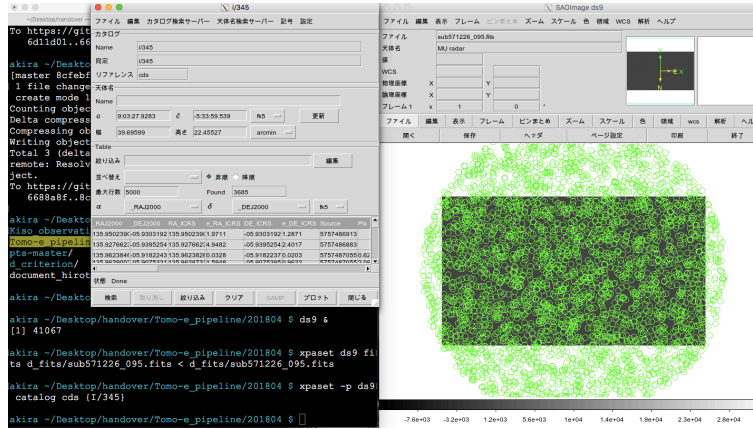
そうすると, ds9 で画像が表示されます。



続いてカタログをダウンロードしてみます。

```
akira ~/Desktop/handover/Tomo-e_pipeline/201804 $ xpa ds9 -p
catalog cds {I/345}
```

ds9 のカタログ検索ウィンドウが開いて指定したカタログからフレームの視野が該当するところを抜き出してきてくれます (おそらくこれは大澤さんが既に WCS 校正をしていたため, 勝手に抜き出してくれているのだと思います). このウィンドウには恒星の ID やら天球座標やら等級などが入っています。



最後にこれを xml ファイルにまとめて出力します. コマンドラインでは次のように打ち込みます.

```
akira ~/Desktop/handover/Tomo-e_pipeline/201804 $ xpaset -p ds9
catalog save sub571226_095_gaia.xml
```

手作業でもできますが, shell スクリプト (ds9\_9\_catalog\_gaia.sh と ds9\_9\_catalog\_ucac.sh) にコマンド文が全て書かれているのでそれを実行しても同じ結果が得られます.

```
akira ~/Desktop/handover/Tomo-e_pipeline/201804 $ ds9 &
[1] 27002
akira ~/Desktop/handover/Tomo-e_pipeline/201804 $ sh ds9_catalog_gaia.sh
```

5. xml\_gaia.ipynb を実行.
6. xml\_ucac.ipynb を実行.  
カタログから測光に必要なパラメーター (等級, radec) を抜き出して csv ファイル ID\_number\_star.csv に出力します.
7. photometry\_gaia.ipynb を実行.  
中間ファイル  
(simultaneous\_event\_candidates\_table\_MU+Tomo-e9\_gaia.csv) が出力されます.
8. gband2vband.ipynb を実行.  
フレーム内に写っている同一の恒星のカタログ等級 (G バンド, V バンド) を用いて変換係数を求めます.  
csv ファイル (gband2vband.csv) が出力されます.
9. mu-tomoe\_analysis3.ipynb を実行.  
セクション 1 で 3 つのファイル  
(1): simultaneous\_event\_candidates\_table\_MU+Tomo-e10.csv

(2):MU+Tomo-e\_main.csv

(3):simultaneous\_event\_candidates\_table\_MU+Tomo-e10\_full.csv が出力されます.

(1) は全ての candidates を含んだテーブル,(2) は MU レーダーと Tomo-e Gozen で検出された距離が中点 (middle point) から  $0.25[\text{deg}]$  以下かつベクトルのなす角が  $2.5$  度以下のもののみをピックアップしたテーブル,(3) は (2) のテーブルに含まれているイベントかつ検出した飛跡が MU レーダーでの検出に対して Tomo-e の検出が完全にオーバーラップしているイベントのテーブルです.

セクション 2 では RCS から等級, 質量, サイズの焼き直しをしており,

simultaneous\_event\_candidates\_table\_MU+Tomo-e10.csv,

/iccd\_2018/MU\_data\_2009-2015.csv

の二つのファイルにカラムが書き加えられます.

## 1.2 shell スクリプトに関して

handover/Tomo-e\_pipeline/201804/ にはいくつか shell スクリプト (.sh ファイル) が入っていますが,script.ipynb で作ることができます. 必ず必要なものというわけではないので,適せん使用するといい感じだと思います.

## 1.3 各プログラムの簡単な説明

### 1.3.1 [mu-tomoe\\_analysis.ipynb-セクション 1](#)

#### Input

simultaneous\_event\_candidates\_table\_MU+Tomo-e3.csv  
d\_fits/の fits ファイル

#### Output

simultaneous\_event\_candidates\_table\_MU+Tomo-e4.csv  
Tomo-e の露光開始時間 (exposure time) Tomo-e で検出した流星の Az,El, ベクトルの中点 (modified midpoint)  
ベクトルの中点から MU レーダーで検出した流星のベクトルへの法線の長さ [deg]  
二つのベクトルのなす角の絶対値 [deg]

fits ファイルのヘッダーから Tomo-e の露光開始時間を抜き出します (使わなかった...).  
データリダクションで求めた Ra,Dec を Az,El に変換します.  
ベクトルの中点 (modified midpoint) から MU レーダーで検出した流星のベクトルへの法線の長さ (最短距離) となす角を計算してカラムを付け足してファイルに出力します.

### 1.3.2 [mu-tomoe\\_analysis.ipynb-セクション 2](#)

#### Input

simultaneous\_event\_candidates\_table\_MU+Tomo-e4.csv

#### Output

sensor.csv  
Tomo-e のセンサーの視野

センサの四角の Az,El を求めています. プロジェクションのグラフの作成に使いましたが, それ以外特に用途はありません..

### 1.3.3 [mu-tomoe\\_analysis2.ipynb-セクション 1](#)

#### Input

simultaneous\_event\_candidates\_table\_MU+Tomo-e4.csv

#### Output

simultaneous\_event\_candidates\_table\_MU+Tomo-e4.csv

データリダクションの際, ピクセル座標平面で y 軸に近い方を始点 (Az1,El1) としていたため,MU レーダーのデータと比べると向きが逆になっているものがあつた. なので MU レーダーの starting point と end point の方向に Tomo-e サイドのベクトルの向きを揃えている.

### 1.3.4 [mu-tomoe\\_analysis2.ipynb-セクション 2](#)

#### Input

simultaneous\_event\_candidates\_table\_MU+Tomo-e4.csv

#### Output

simultaneous\_event\_candidates\_table\_MU+Tomo-e5.csv

MU レーダーと Tomo-e で検出した流星の飛跡が被っている割合 (オーバーラップ) を計算している. この割合は MU レーダー側で検出した流星のベクトル内に Tomo-e 側で検出した流星のベクトルが完全に内側に存在する場合を 100%, 外側にある場合を 0% としている.

### 1.3.5 [mu-tomoe\\_analysis2.ipynb-セクション 3](#)

```
simultaneous_event_candidates_table_MU+Tomo-e5.csv  
../MU_orbit_data_20181023.csv
```

#### Output

simultaneous\_event\_candidates\_table\_MU+Tomo-e6.csv MU レーダーを中心とした 3 次元直行座標系の流星の座標

MU レーダーで得られた流星の軌道要素などが入ったテーブル (MU\_orbit\_data\_20181023.csv) と合併する. 何かに使うと思った starting point と end point の三次元座標も計算している.

### 1.3.6 [photometry\\_meteor.m](#)

#### Input

```
simultaneous_event_candidates_table_MU+Tomo-e6.csv
```

#### Output

```
simultaneous_event_candidates_table_MU+Tomo-e7.csv  
simultaneous_event_candidates_table_MU+Tomo-e8.csv
```

流星のフラックスを求めるプログラム. 角度を持った流星の飛跡がピクセル座標に直行するように画像を回転させて矩形領域でカウント値の平均値をとり, ガウシアンフィッティングの結果を積分して, 1 ピクセルあたりの平均フラックス (Line Intensity) を求めます. MU レーダーのデータから角速度を計算して 0.5 秒 (Tomo-e の露光時間) で通過するであろうピクセル数 (travel distance) を求め, 二つの積が流星の 0.5 秒間でのフラックスとして計算しています. 図は d\_photometry\_meteor/に入っています

プログラムの最初に以下の setting が書いてあります. デフォルトはいくつかコメントアウトがしてあり, FILE OUTPUT(113,14 は off (0), Single Image(117) は off (0), re.analyze(122) は off(0), version(123) が”main”になっています.

FILE OUTPUT は解析の画像 (矩形領域, ガウシアンフィット) を出力するかどうか.

Single Image は特定の ID の画像を一枚, 再度解析 (矩形領域幅の変更) する場合に使用します. そのときは 118 の ID\_number のコメントアウトをはずし, ID を入れると該当する画像だけ pixel.width に入力された矩形の幅 (デフォルトは 20 ピクセル) で再解析をします.

re.analyze は初回以外は on (1) にしておきます (off のままにするとなんかエラーが出た記憶があります).

最後に version は”main”のままで大丈夫です.”test”があるのは当初いちいち全部の ID に対してプログラムを回すのが面倒くさかったからです (おそらくもう変えることはないでしょう).

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% section : SETTING

%TF = readtable("recheck_photometry.csv");
%name = TF(:, "ID"); name = table2array(name);
%for k = 1:size(name)
    %ID_number = name(k,1);
    %ID_number = string(ID_number);

%%% FILE OUTPUT(.pdfの出力) 画像の出力をありにすると非常に重くなるので注意
option_show_image = 0; %yes:1, no:0
option_save_image = 0; %yes:1, no:0

%%% ANALYZE SINGLE IMAGE FILE
option_single_image = 0; %yes:1, no:0
%ID_number = "586316_109"; %571226_095 , 571342_058(recheck), 572534_007
pixel_width = 10; %default is 20pix

%%% ANALYZIE SETTING
re_analyze = 0; %yes:1, no:0
version = "main"; % 'test' or 'main'

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

エラーの出たIDはrecheck\_photometry.csvというファイルに出力されます(このときoriginalのデータのほうはフラックスが0になります). ここでいうエラーは矩形の頂点がフレーム外に出てしまったものが大半です. この場合re\_analyze(l22)をon (1) にして,pixel\_width(l19)を小さくすることで解決できます.

### 1.3.7 [xml\\_gaia.ipynb](#), [xml\\_ucac.ipynb](#)

#### Input

simultaneous\_event\_candidates\_table\_MU+Tomo-e8.csv  
d\_xml/のxmlファイル  
d\_fits/のfitsファイル

#### Output

xmlファイルからカタログ等級と座標を抜き出したcsvファイル(ID\_number\_star.csv)

ds9で出力した各フレームごとのカタログ恒星のxmlファイルからphotometryで使用するカタログ等級,座標を抜き出してcsvファイルに出力するプログラム.

ds9で一度カタログ恒星をプロットしてみるとわかりますが, フレーム外に存在する恒星や非常に暗いもの(とにかく数が多い)もあるため, 多少間引くことをしています. アパーチャー径を考慮してフレームの端から10ピクセル内側の恒星のみを使用しています. またastropyのstarfinderである程度の明るさの恒星のみを事前にピックアップしておき, それと同一の恒星(中心間距離が2ピクセル)のみを使用しています. ファイルはd\_csv\_gaia/とd\_csv\_ucac/に入っています.

### 1.3.8 [photometry\\_gaia.ipynb](#), [photometry\\_ucac.ipynb](#)

#### Input

simultaneous\_event\_candidates\_table\_MU+Tomo-e8.csv  
d\_csv\_gaia/のxmlファイル  
d\_fits/のfitsファイル

#### Output

simultaneous\_event\_candidates\_table\_MU+Tomo-e9\_gaia.csv

simultaneous\_event\_candidates\_table\_MU+Tomo-e9\_ucac.csv  
ポグソンの式のフィッティンググラフと恒星のフォトメトリーイメージ  
等級原点を求めるのに使用したカタログ恒星のリスト (ID\_number\_star\_used.csv)

恒星のアーチャー測光を行いポグソンの式にフィッティング, 等級原点を求めて, 流星のフラックスから等級を計算する. また, 距離の補正 (絶対等級) も同様に計算. フォトメトリーのグラフとイメージを出力する場合はコメントアウトしてある部分を消して実行するとたくさんグラフが出てきます (図は d\_photometry\_gaia/に入っています).

### 1.3.9 gband2vband.ipynb

#### Input

simultaneous\_event\_candidates\_table\_MU+Tomo-e10.csv  
d\_xml/の xml ファイル  
d\_fits/の fits ファイル

#### Output

同一恒星の G バンド等級と V バンド等級をまとめたリスト (gband2vband.csv)  
同一恒星の G バンド等級と V バンド等級の散布図とフィッティンググラフ (gband2vband.png)

ds9 で G バンド等級カタログ (gaia dr2) と V バンド等級のカタログ (ucac) を抜き出してきたので二つのカタログで同一の恒星を抜き出して G バンドから V バンドへの変換係数を求めています. Input ファイルが 10 番目の csv ファイルになっているのは, このプログラム自体が測光解析を一度行なった後から作ったものだからです (過去の同時観測の結果と比較するために V バンド等級が必要になったため). ID\_number を参照しているだけなので 9 番目のものでも問題はないです.

### 1.3.10 mu-tomoe.analysis3.ipynb-セクション 1

#### Input

simultaneous\_event\_candidates\_table\_MU+Tomo-e9\_gaia.csv  
simultaneous\_event\_candidates\_table\_MU+Tomo-e9\_ucac.csv

#### Output

simultaneous\_event\_candidates\_table\_MU+Tomo-e10.csv  
MU+Tomo-e\_main.csv  
simultaneous\_event\_candidates\_table\_MU+Tomo-e10\_full.csv

データの整理と G バンド等級から V バンド等級への変換をしています. 10 番目のファイルは全ての candidates を含んだファイル, MU+Tomo-e\_main はベクトルの距離が 0.25[deg] 以下かつなす角が 2.5[deg] 以下の同時流星のみのファイル, 10\_full は main 内にあるもので飛跡が完全にオーバーラップしているものです. まあ Line Intensity が非常に小さい (14 等級よりも暗いもの) は測光が上手くできてない可能性が濃厚だったのでここでは削除しました.

### 1.3.11 mu-tomoe.analysis3.ipynb-セクション 2

#### Input

simultaneous\_event\_candidates\_table\_MU+Tomo-e10.csv  
../../iccd\_2018/MU\_data\_2009-2015.csv

#### Output



```
simultaneous_event_candidates_table_MU+Tomo-e10.csv  
../../iccd_2018/MU_data_2009-2015.csv
```

RCS から等級, 質量, サイズへの変換を行なっています.

## 2 解析以外のプログラム

### 2.1 [/iccd\\_2018/population\\_index/population\\_index.ipynb-セクション 1](#)

#### Input

```
../MU_data_2009-2015.csv
```

#### Output

```
../MU_data_2009-2015.csv
```

次のセクションでフラックスを計算するために, 2009-2015 年のデータの total の観測時間を計算しています. 理由は不明確ですが, 観測期間中微妙に空いている時間があったため, 約 30 分以上の空白がある場合はその時間を除いて観測時間を求めています.

### 2.2 [/iccd\\_2018/population\\_index/population\\_index.ipynb-セクション 2](#)

#### Input

```
../MU_data_2009-2015.csv  
collection_area.csv
```

#### Output

population index を計算したグラフ (d.img/population\_index.pdf)

population index の計算.collection\_area は daniel の Git のレポジトリ内にある calc\_collection\_area.py から引っ張ってきたデータです.

RCS<sub>i</sub>20dBsm, 天頂角が 90 度以上のデータは間引きしました.

### 2.3 [/iccd\\_2018/population\\_index/mass\\_size\\_distribution\\_function.ipynb](#)

#### Input

```
../MU_data_2009-2015.csv  
collection_area.csv  
d_data/visual.csv  
d_data/radio.csv  
d_data/satelite.csv  
d_data/fireball.csv  
d_data/Gruen.csv
```

#### Output

mass distribution function, size distribution function を計算したグラフ 3 枚 d.img/mass\_and\_size.index.png  
d.img/mass\_distribution.png  
d.img/size\_distribution.png

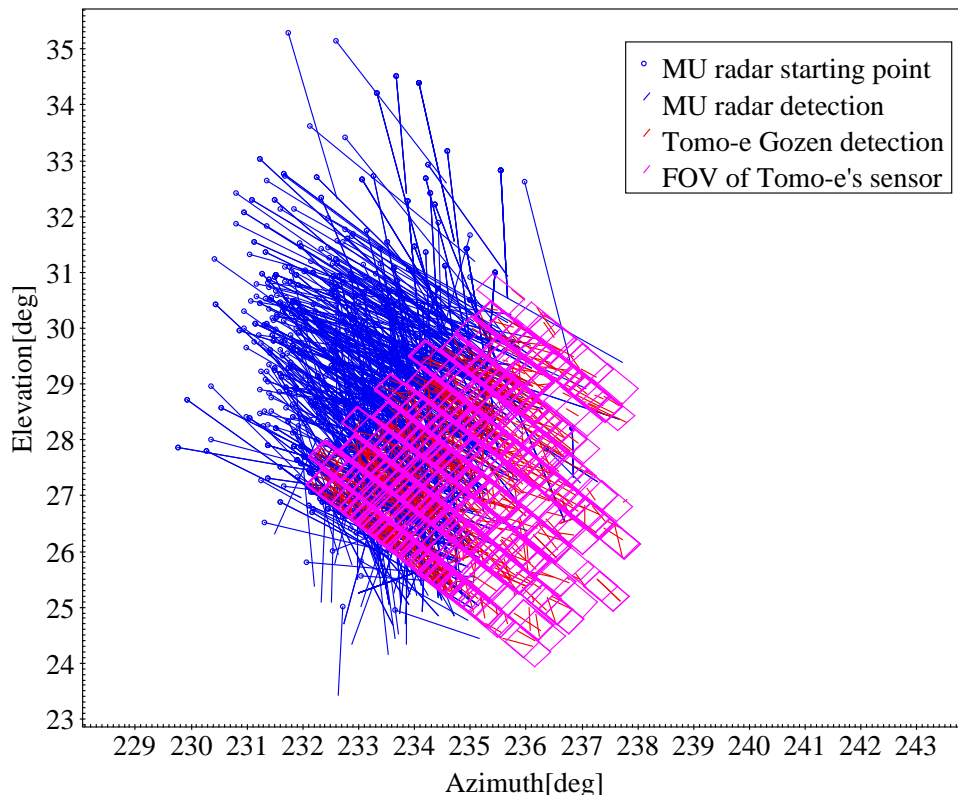


mass distribution function, size distribution function の計算と比較. 使っているデータは Huges, gruen, brown のもので本論の参考文献を参照してください.

## Part III

## TOPCATのグラフ

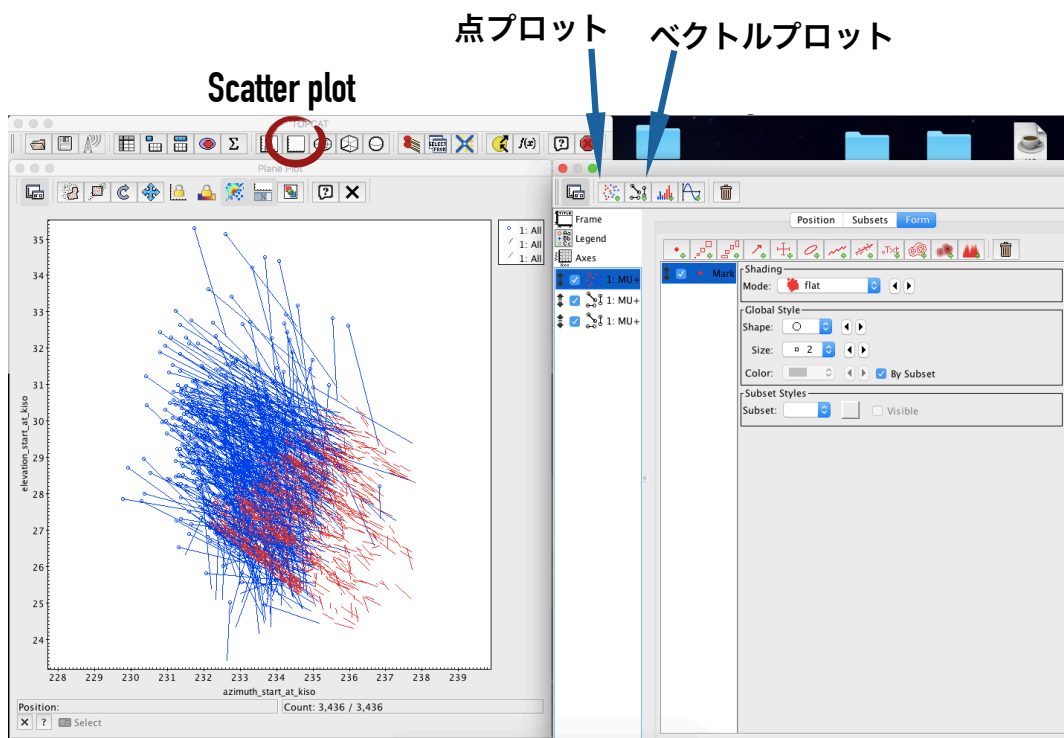
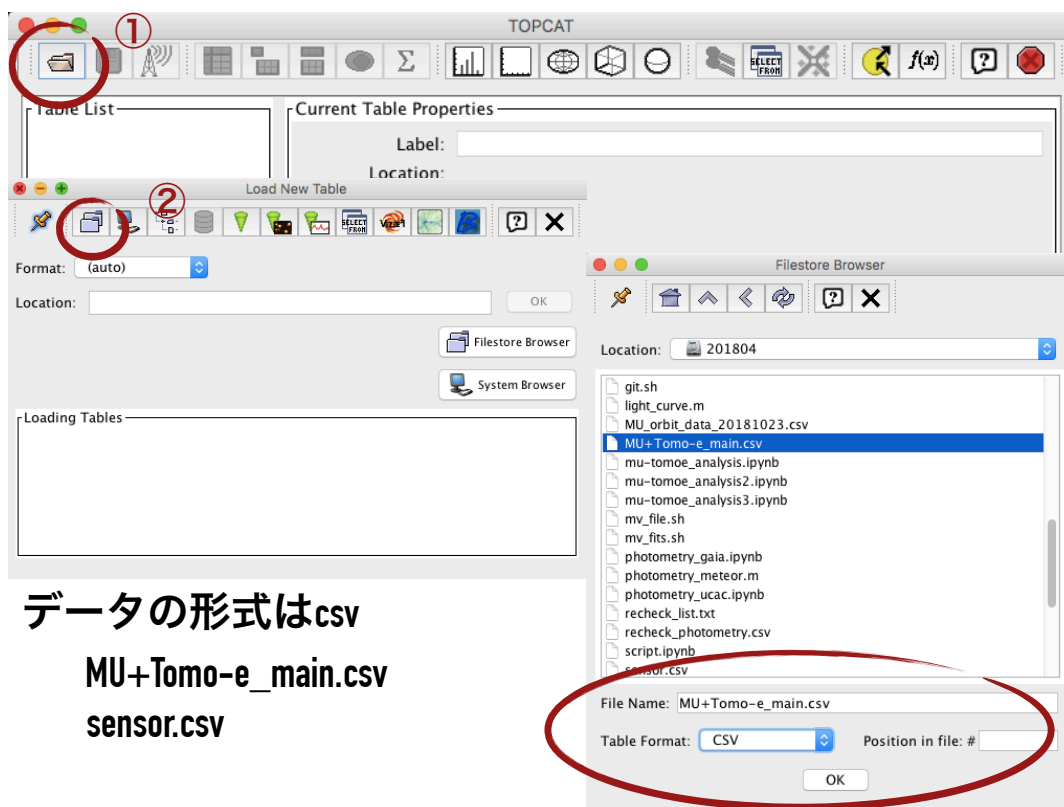
## 1 プロジェクションや散布図



この作成手順

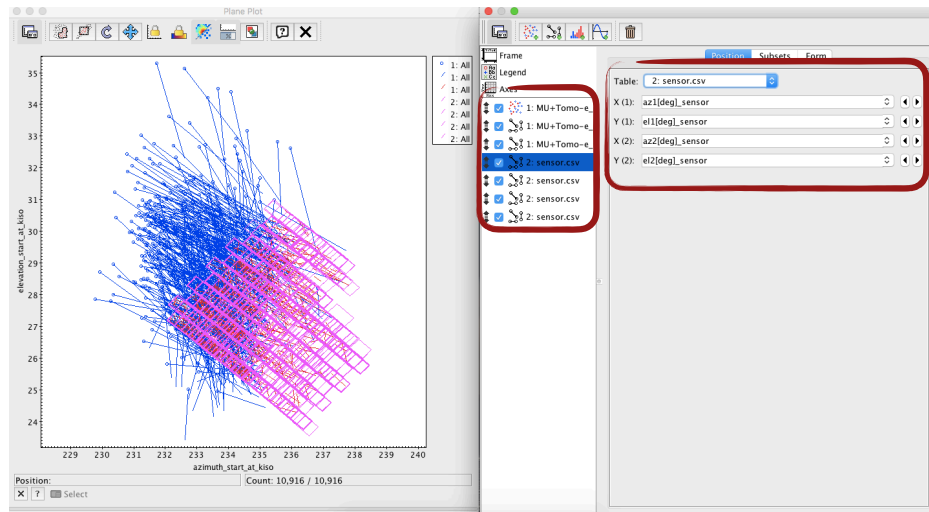
はじめにテーブルの読み込みを行います.topcat 上部タブのディレクトリボタンを押してテーブルを選択します.フォーマットはcsvで読み込ませます.上図の作成にはMU+Tomo-e\_main.csvとsensor.csvを読み込ませます.

テーブルを読み込んだら topcat 上部タブにある散布図(グラフみたいなボタン)を選択するとグラフウィンドウが出てきます.散布図とベクトルのプロットが可能でデータに合わせて選択してください.



今回の場合は散布図 (starting point) が一つとベクトルプロットが6つ選び、データの内訳はMUレーダーで検出された流星のベクトル, Tomo-eで検出された流星のベクトル, センサの視野が4つです。それぞれベクトルの始点と終点のデータをテーブルとカラムを選択することでベクトルが描画されます。

グラフの体裁は左側のタブ内の Legend で凡例の設定, Axis で軸の書式設定, フォントの設定が可能です。また Form を選択するとプロットの形や色なんかを変更することができ、近似直線の追加やエラーバーの追加ができます。



テーブルを選択した後、データをプロット  
ベクトルの場合は始点と終点のx,y座標  
FormのタブやAxisなどでFontなどで  
図の体裁を整えることが可能

## 2 ヒストグラム

topcat 上部タブ内のヒストグラム (棒グラフみたいなやつ) を押すとグラフウィンドウが開きます. 散布図と同様に使用するテーブルとデータを選ぶとヒストグラムが作成されます.

左側にある Bins からビンサイズの設定や累積ヒストグラムにすることができます. 軸やフォントの設定は同様に Axis から選ぶことができます.

