
GRID-pipeline Documentation

发布 *2.1.0*

GRID Team

2021 年 04 月 13 日

1 概览 Overview	1
2 文件组织结构 Documents Structure	3
2.1 配置文件 Configuration Files	3
2.2 输入输出文件 Input & Output	3
2.3 数据类型 Data type	6
3 观测计划 Observation Plan	7
3.1 原始数据 Original data	7
3.2 处理函数 Python functions	7
3.3 处理过程 Process	7
4 监测数据 Monitor Data	9
4.1 原始数据 Original data	9
4.2 处理函数 Python functions	10
4.3 处理过程 Process	10
5 科学数据 Time-Tagged Event	13
5.1 原始数据 Original data	13
5.2 处理函数 Python functions	14
5.3 处理过程 Process	14
6 时间轴重建 Timeline Reconstruction	17
6.1 原始数据 Original data	17
6.2 处理函数 Python functions	17
6.3 处理过程 Process	17
7 能量重建 Energy Reconstruction	21
7.1 原始数据 Original data	21
7.2 处理函数 Python functions	21
7.3 处理过程 Process	22
8 姿态数据 Attitude Data	25
8.1 原始数据 Original data	25
8.2 处理函数 Python functions	25
8.3 处理过程 Process	26
9 地理坐标 Geographic coordinate	27

9.1	原始数据 Original data	27
9.2	处理函数 Python functions	27
9.3	处理过程 Process	28
10	图片文件 Quicklook	29
10.1	原始数据 Original data	29
10.2	处理函数 Python functions	29
10.3	处理过程 Process	29
11	程序运行方法 Operation of pipeline	31
11.1	数据产品 Data products	31
11.2	观测计划检查 Check observation plan	32
11.3	运行状况检查 Check running status	32
11.4	增益刻度 Gain calibration	32
12	残余问题 Problems Remaining	33
12.1	数据重复 Data Repetition	33
13	gallery	35
13.1	Data in 01.py	35
13.2	Data in 02.py	36
14	make.py	37
14.1	Functions in make.py	37
14.2	Commandline interface in make.py	37
15	gridfun.py	39
15.1	Data in gridfun.py	39
15.2	Classes in gridfun.py	39
15.3	Functions in gridfun.py	41
16	gridpath.py	53
16.1	File path in gridpath.py	53
17	cali.py	55
17.1	Functions in cali.py	55
17.2	Commandline interface in cali.py	55
18	statuspic.py	59
18.1	Functions in statuspic.py	59
18.2	Commandline interface in statuspic.py	60
19	plancompare.py	61
19.1	Functions in plancompare.py	61
19.2	Commandline interface in plancompare.py	61
20	runtime.py	63
20.1	Functions in runtime.py	63
20.2	Commandline interface in runtime.py	63
21	更新日志 Change Log	65
21.1	v2.1.0	65
22	索引 Indices and tables	67
	Python 模块索引	69

CHAPTER 1

概览 Overview

本文档用于介绍 [GRID-pipeline](#) 的数据处理流程与数据格式，及其所使用的函数。[GRID-pipeline](#) 是 [GRID](#) 科学组 用于 [天格计划](#)（Gamma-Ray Integrated Detectors, GRID）项目中的数据处理流水线。流程介绍与函数注释均采用中英文混合编辑。

[GRID-pipeline](#) 的目标是由卫星下传数据、地面标定数据等生成可供科学数据分析的数据文件。[GRID-pipeline](#) 采用面向过程的程序设计方法（Procedure Oriented Programming），核心在于输出文件的自动生成过程，及其正确性。

This document is the introduction of [GRID-pipeline](#). [GRID-pipeline](#) is the data processing pipeline software of [GRID](#) (Gamma-Ray Integrated Detectors) developed by [GRID-datagroup](#). Data processing progress, data structure and python functions are listed here. The whole documentation is both in Chinese and English.

The target of [GRID-pipeline](#) is providing datasets for scientific data analysis based on transferred data from satellites and calibration data. [GRID-pipeline](#) is procedure-oriented programming, which focuses on the automatic generation of output files as well as data accuracy.

文件组织结构 Documents Structure

文件路径分两级，第一级为文件在服务器上的位置，由`gallery` 决定，第二级为文件目录树的分支路径，由`gridpath` 决定。代码程序运行中文件路径配置所有 `.fits` 文件中的时间均为 **UTC+0**。

There are 2 levels of directories. The first level is files in sever, which is determined by `gallery`. The second level is branch in directory tree, which is determined by `gridpath`. Time in all `.fits` file is in timezone **UTC+0**

2.1 配置文件 Configuration Files

配置文件为`gallery`，其中记录了每颗探测器的名称、服务器上文件位置和探测器法线等信息。

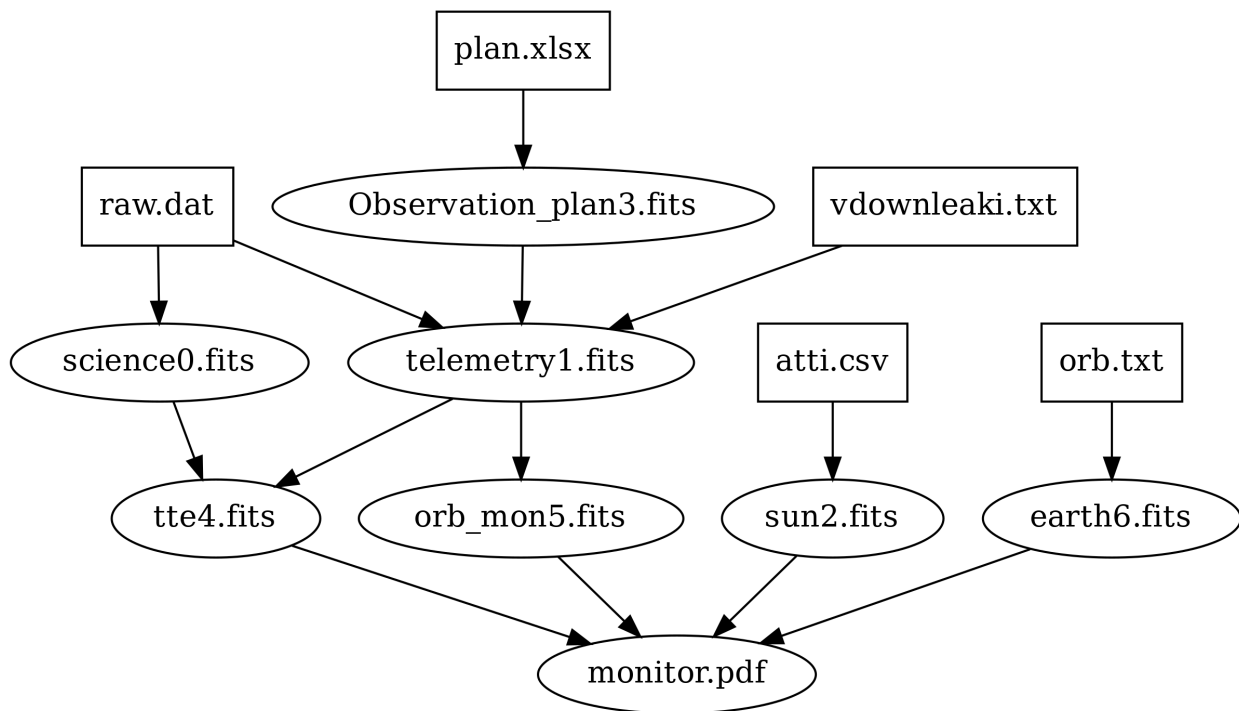
Configuration Files are `gallery`, which include name of detectors, direction of related files in sever and unitary normal of detectors.

2.2 输入输出文件 Input & Output

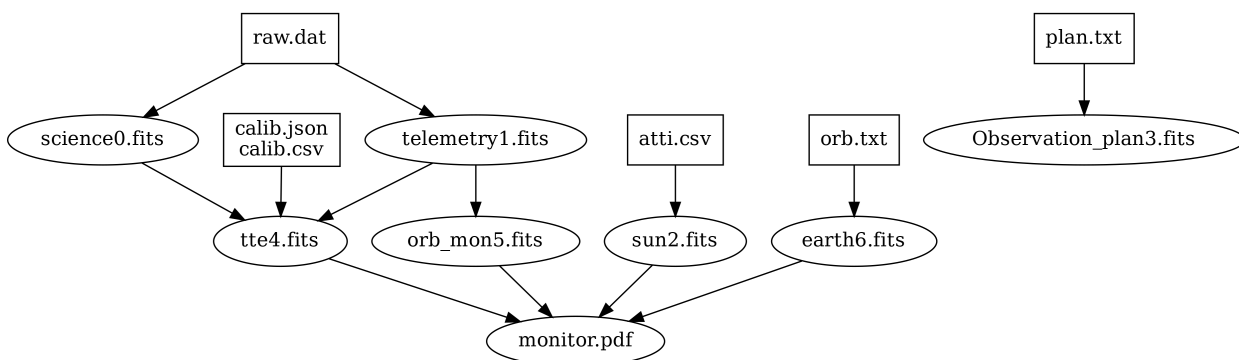
输入文件的组成比较复杂。输出文件大多为天文领域常用的 `.fits` 文件，以及日志文件。

There various kinds of input files. Output files are `.fits` which are widely used in the field of astronomy, and some log file.

2.2.1 GRID-01 文件依赖关系 File Dependencies



2.2.2 GRID-02 文件依赖关系 File Dependencies



生成图片方法为，在 [GRID-pipeline](#) 文件夹下输入：

```
dot -Tpng gallery/01.dot -o docs/source/_static/01.png && dot -Tpng gallery/02.dot -o docs/source/_static/02.png
```

2.2.3 输入文件 Input

02 星输入文件 Input files of GRID-02

位于 *02.sourprefix* 描述的路径中。

In directory described by *02.sourprefix*.

主要包括：

Mainly includes:

- *raw.dat* 卫星下传二进制文件，只包括科学数据包和监测数据包，已经经过了解扰等处理，类似于 GECAM 的 0A 级文件 Binary transferred data from satellite
- *atti.csv* 姿态控制文件，包括随时间变化的四元数 Attitude file
- *orb.txt* 地球轨道文件，包括随时间变化的卫星轨道的经纬度和高度 Earth orbit file
- *plan.txt* 观测计划文件 Observation plan file
- 相关标定文件 *calib.json* 和 *calib.csv* Calibration file
- 地球轨道辐射环境数据 Radiation environment dataset in near earth orbit

2.2.4 输出文件 Output

02 星输出文件 Output files of GRID-02

位于 *02.fitsprefix* 描述的路径中。

In directory described by *02.fitsprefix*.

主要包括：

Mainly includes:

- *science0.fits* 初步处理的科学数据 Preliminarily treated scientific data
- *telemetry1.fits* 初步处理的监测数据 Preliminarily treated telemetry data
- *sun2.fits* 探测器姿态、太阳-探测器相对位置数据 Attitude of detector & sun-detector relative position
- *Observation_plan3.fits* 观测计划记录文件 Observation plan
- *tte4.fits* Time Tagged Event (TTE)
- *orb_mon5.fits* 分轨监测数据 Telemetry data split by orbit
- *earth6.fits* 卫星地球轨道、地球-探测器相对位置数据 Earth orbit & earth-detector relative position
- 所有 *log.csv* 日志文件 All *log.csv* log file

2.2.5 图片文件 Quicklook

位于 `02.picsprefix` 描述的路径中。

In directory described by `02.picsprefix`.

2.3 数据类型 Data type

表 1: Data type

Data	struct format	FITS format	minimum format
utc	L	J	int32
pps	Q	J	int32
usc	Q	K	int64
channel	B	I	int16
adc_value	H	J	int32
valid_count	L	J	int32
lost_count	L	J	int32
sipm_temp	H	I	int16
adc_temp	H	I	int16
vmon	H	I	int16
imon	H	I	int16
mcu_temp	H	I	int16
position		J	int32
label		I	int16
utc_recon		J	float64
iop		D	float64
vop		D	float64
vout		D	float64
t		D	float64
E		D	float64
ra		D	float64
dec		D	float64
xyz		D	float64

观测计划 Observation Plan

观测计划 `Observation_plan3.fits` 的整理是科学数据处理中的第一步。观测计划记录了每一轨的计划时间戳起点和终点和定姿态观测中的计划四元数姿态。

Collection of observation plan is the first step in the data process. `Observation_plan3.fits` recorded planned start-end timestamp and quaternion of planned attitude.

3.1 原始数据 Original data

3.1.1 02 星原始数据 Original data of GRID-02

`plan.txt` 观测计划文件，由观测计划制定者生成。文件夹为 `gridpath.bin_source_H`。

`plan.txt`, directory is `gridpath.bin_source_H`.

3.2 处理函数 Python functions

`gridfun.makefits3()`

3.3 处理过程 Process

3.3.1 02 星处理过程 Process of GRID-02

1. 使用 `gridfun.makefits3()`，每次处理均遍历 `gridpath.bin_source_H` 下所有文件 Traverse all file in `gridpath.bin_source_H`
2. 找到所有关键行 Find all critical lines in `.txt` file:

1. 包含 ‘upload_quaternion’ ‘tg_PowerOn.bin’ ‘tg_PowerOnM.bin’ ‘tg_TXDataOn.bin’ ‘tg_PowerOff’ 的为关键行
3. 提取包含 ‘tg_PowerOn.bin’ ‘tg_PowerOnM.bin’ 的行中的起始时间信息 Extract start time of orbit
4. 提取包含 ‘tg_PowerOn.bin’ ‘tg_PowerOnM.bin’ 的行的下一行中的终止时间信息 Extract end time of orbit
5. 提取包含 ‘tg_PowerOn.bin’ ‘tg_PowerOnM.bin’ 的行的上一行中的四元数信息 Extract quaternion of orbit
6. 存储数据到 *gridpath.fits3p* Save data into *.fits*

监测数据 Monitor Data

监测数据 `telemetry1.fits` & `orb_mon5.fits` 包含了探测器及卫星的工作状态等信息，用于清洗和筛选科学数据，是科学数据分析的基础。在提取监测数据的过程中，进行了时间轴重建。

Monitor data `telemetry1.fits` & `orb_mon5.fits` contains working status of detectors and satellites. They are used in data cleaning and filtering of scientific data. Timeline reconstruction is performed when extracting monitor data.

4.1 原始数据 Original data

4.1.1 02 星原始数据 Original data of GRID-02

`telemetry1.fits` 的原始文件 Original data of `telemetry1.fits`

`raw.dat` 卫星下传二进制文件。文件夹为 `gridpath.dat_source_H`。

`raw.dat` Binary file transferred from satellites, directory is `gridpath.dat_source_H`.

`orb_mon5.fits` 的原始文件 Original data of `orb_mon5.fits`

`telemetry1.fits`, 文件夹为 `gridpath.tel_source_H`

`telemetry1.fits`, directory is `gridpath.tel_source_H`.

4.2 处理函数 Python functions

4.2.1 02 星处理函数 Python functions of GRID-02

telemetry1.fits (与 **science0.fits** 同时进行)

```
gridfun.makefits01()  
gridfun.extract_tel_events()  
gridfun.fits1_recon()  
gridfun.make02electronics()
```

orb_mon5.fits

```
gridfun.makefits5()
```

4.3 处理过程 Process

4.3.1 02 星处理过程 Process of GRID-02

telemetry1.fits (与 **science0.fits** 同时进行)

1. 使用`gridfun.makefits01()`, 查找`gridpath.dat_source_H` 下所有文件 Find all file in `gridpath.dat_source_H`
2. 不处理`gridpath.log_dat_path` 日志文件记录的 `raw.dat` 文件 Ignore `raw.dat` recorded in `gridpath.log_dat_path`
3. 使用函数`gridfun.extract_tel_events()` 提取 `raw.dat` 中的监测数据 (`raw.dat` 中监测数据和科学数据由不同的包头区分) Extract telemetry from `raw.dat` using functions `gridfun.extract_tel_events()`
4. 使用函数`gridfun.fits1_recon()` 进行时间轴重建 *Timeline Reconstruction* Timeline reconstruction
5. 时间轴重建后, 使用函数`gridfun.make02electronics()` 计算运行中的电压电流等监测数据 After timeline reconstruction, calculate monitor data such as current & voltage during operation
6. 按 `raw.dat` 文件名存储保存 Save `telemetry1.fits` data
7. 保存的 `telemetry1.fits` 的文件名, 是 `raw.dat` 修改扩展名后在前面加 `tel` Saved `telemetry1.fits` file name, is modified extension of `raw.dat` with `tel` prefix
8. 将处理过的 `raw.dat` 文件名保存到日志文件`gridpath.log_dat_path` Save file name of `raw.dat` into log file

orb_mon5.fits

1. 使用`gridfun.makefits5()`, 查找`gridpath.tel_source_H` 下所有文件 Find all file in `gridpath.tel_source_H`
2. 不处理`gridpath.log_tel_path` 日志文件记录的 raw.dat 文件 Ignore raw.dat recorded in `gridpath.log_tel_path`
3. 将初步处理的 `telemetry1.fits` 监测数据按轨分开, 按天存储在`gridpath.orbmon_source_H` 文件夹中, 文件夹名为 `nnnnn_yymmdd`, 其中 `nnnnn` 为自 GRID-01 入轨的天数, 文件名为 `yymmddHHMM_yymmddHHMMorb.fits` Divide preliminarily treated monitor data for each orbit, save in `gridpath.orbmon_source_H` directory, with directory name `nnnnn_yymmdd`, file name `yymmddHHMM_yymmddHHMMorb.fits`
4. 将每一天的 `orb_mon5.fits` 数据中每一轨的起始和终止时间的时间戳和格式化时间 (`yymmddHHMMSS`) 存储在`gridpath.log_orb_H` 中 Save the unix timestamp and formatted time of every orbit in each day `orb_mon5.fits` to `gridpath.log_orb_H`
5. 将处理过的 `telemetry1.fits` 文件名保存到日志文件`gridpath.log_tel_path` Save file name of `telemetry1.fits` into log file

科学数据 Time-Tagged Event

科学数据，狭义上指 TTE (Time Tagged Events) 文件，是时间 - 能量数组。是 GRB 数据分析的核心。
Scientific data, in a narrow sense, TTE, is the core of GRB data analysis.

5.1 原始数据 Original data

5.1.1 02 星原始数据 Original data of GRID-02

science0.fits 的原始文件 Original data of science0.fits

raw.dat 卫星下传二进制文件。文件夹为 *gridpath.dat_source_H*。

raw.dat Binary file transferred from satellites, directory is *gridpath.dat_source_H*.

tte4.fits 的原始文件 Original data of tte4.fits

science0.fits 初步处理的科学数据。文件夹为 *gridpath.sci_source_H*。

science0.fits preliminarily treated scientific data, directory is *gridpath.sci_source_H*.

telemetry1.fits 初步处理的监测数据。文件夹为 *gridpath.tel_source_H*。

telemetry1.fits preliminarily treated telemetry data, directory is *gridpath.tel_source_H*.

相关标定文件 calib.json 和 calib.csv

Calibration file

5.2 处理函数 Python functions

5.2.1 science0.fits (与 telemetry1.fits 同时进行)

```
gridfun.makefits01()  
gridfun.extract_sci_events()  
gridfun.fits0_save()
```

5.2.2 tte4.fits

```
gridfun.makefits4()  
gridfun.energy_recon()  
gridfun.fits4_make()
```

5.3 处理过程 Process

5.3.1 02 星处理过程 Process of GRID-02

science0.fits (与 telemetry1.fits 同时进行)

1. 使用`gridfun.makefits01()`, 查找`gridpath.dat_source_H` 下所有文件 Find all file in `gridpath.dat_source_H`
2. 不处理`gridpath.log_dat_path` 日志文件记录的 raw.dat 文件 Ignore raw.dat recorded in `gridpath.log_dat_path`
3. 使用函数`gridfun.extract_sci_events()` 提取 raw.dat 中的科学数据 (raw.dat 中监测数据和科学数据由不同的包头区分) Extract scientific data from raw.dat using functions `gridfun.extract_sci_events()`
4. 使用函数`gridfun.fits0_save()` , , 按 raw.dat 文件名存储保存 Save science0.fits data
5. 保存的 science0.fits 的文件名, 是 raw.dat 修改扩展名后在前面加 sci Saved science0.fits file name, is modified extension of raw.dat with sci prefix
6. 将处理过的 raw.dat 文件名保存到日志文件`gridpath.log_dat_path` Save file name of raw.dat into log file

tte4.fits

1. 使用`gridfun.makefits4()`, 查找`gridpath.sci_source_H` 下所有文件 Find all file in `gridpath.sci_source_H`
2. 不处理`gridpath.log_sci_path` 日志文件记录的 science0.fits 文件 Ignore science0.fits recorded in `gridpath.log_sci_path`
3. 对所有待处理的 science0.fits 文件, 查找`gridpath.tel_source_H` 下的对应 telemetry1.fits 文件 (直接用文件名对应即可) For each science0.fits, lookup for corresponding telemetry1.fits in directory `gridpath.tel_source_H`

4. 提取对应的 `telemetry1.fits` 中的重建过的时间轴信息 Extract reconstructed timeline in `telemetry1.fits`
5. 使用 `gridfun.energy_recon()`, 对待处理的 `science0.fits`, 使用时间轴进行 `usc` 的差值, 得到 TTE 对应的 `utc` Interpolate based on timeline in `science0.fits`, get `utc` time in TTE
6. 使用 `gridfun.energy_recon()`, 对其他监测数据 (电压电流等) 也进行差值 Interpolate to get corresponding monitor data
7. 使用 `gridfun.energy_recon()` 中定义的 `gridfun.Adc_mapping` 进行能量重建 *Energy Reconstruction* Energy reconstruction
8. 使用函数 `gridfun.fits4_make()` 保存 TTE Save TTE
9. 以每一轨的 **起始时间**确定日期, **分日期按轨储存** TTE 数据, `tte4.fits` 的文件名与 `orb_mon5.fits` 文件名对应 Filename of `tte4.fits` is corresponding to filename of `orb_mon5.fits`
10. 将处理过的 `science0.fits` 文件名保存到日志文件 `gridpath.log_sci_path` Save file name of `science0.fits` into log file

时间轴重建 Timeline Reconstruction

时间轴重建指的是，在分析提取监测数据时，建立 `usc-utc` 对应关系。`usc` 是晶振的计数，具有很高的时间精度，`utc` 指协调世界时，是常用的时间系统。只有建立 `usc-utc` 对应关系后，才可以确定科学数据（Time-Tagged Event, TTE）中的时间。

Timeline reconstruction is to find numerical relation between `usc` and `utc`. `usc` is the count of crystal oscillator which is highly precise. `utc` is Coordinated Universal Time which is commonly used. Only after establishing `usc-utc` relation, we can determine the time in TTE.

6.1 原始数据 Original data

6.1.1 02 星原始数据 Original data of GRID-02

`gridfun.extract_tel_events()` 读取出的监测数据

6.2 处理函数 Python functions

`gridfun.fits1_recon()`

6.3 处理过程 Process

6.3.1 02 星处理过程 Process of GRID-02

相关数据 Related data

表 1: Related data

描述 Description	简称 Abbr	更新间隔 Update Interval
uScount	usc	—
last PPS	pps	1s
last PPS 对应的 usc	usc_pps	—
last UTC	utc	10s
last UTC 对应的 PPS	pps_utc	—

- 正常工作状态下每一轨中的 usc 只会不断累加而不会清零 usc will accumulate consistently through the time of one orbit
- 时间信息的更新和记录不发生在同一进程中 Update and record of time data is not correlated in 1 progress
- utc 和 pps 并不同时更新 utc & pps do not update simultaneously
- 时间信息的记录间隔设定值为 1s，但星载 cpu 负载高的时间间隔可能会变长 The record interval of time data is set to be 1s, but will prolong when the load of cpu is high

重建过程 Reconstruction

准备工作

1. 依据监测数据中记录的 utc，utc 前后差值大于 600 秒会被认定为不同轨 Divide monitor data according to the time hopping of utc

基于 Hough transform 的数据清洗

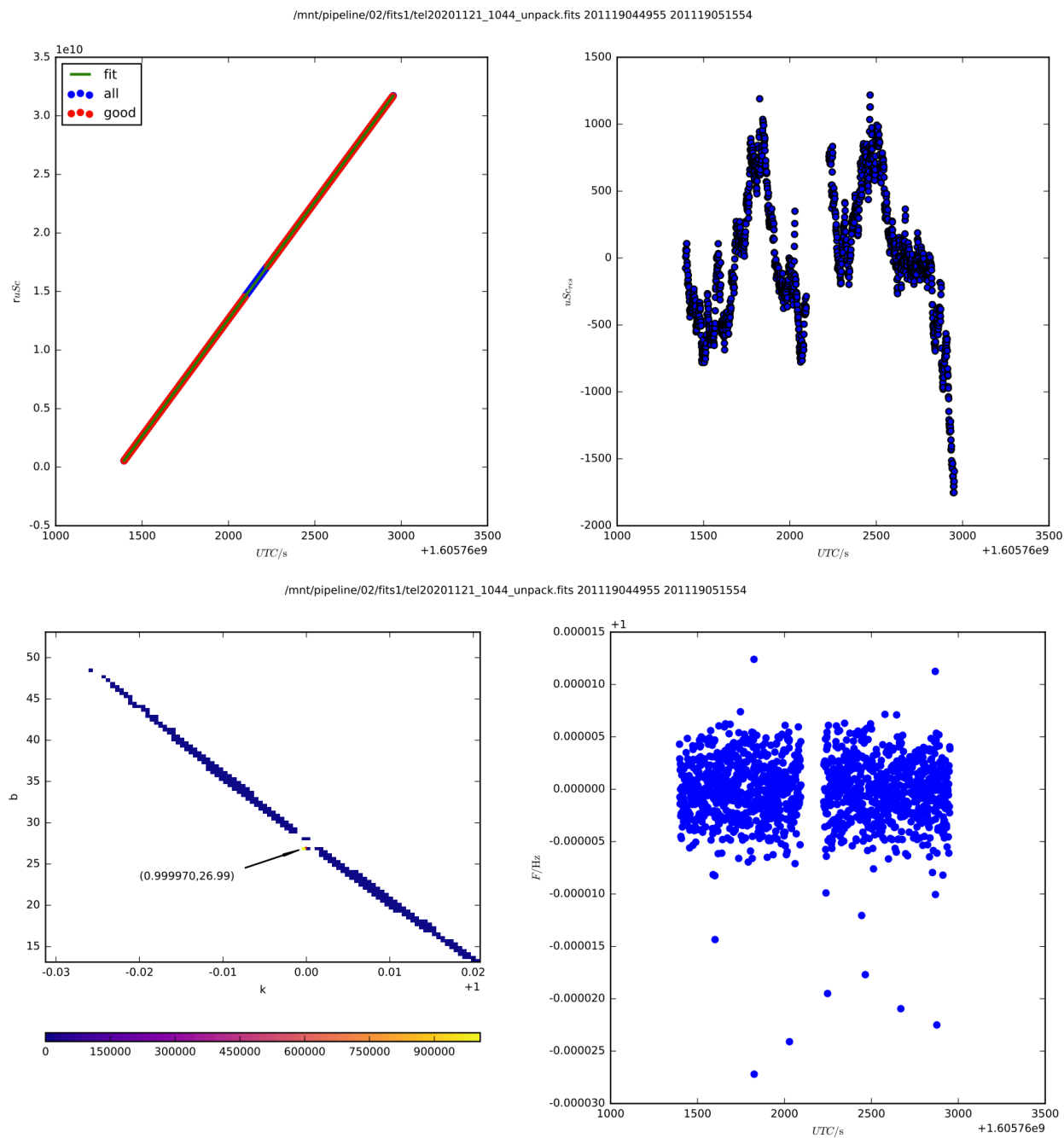
1. 去除¹: $utc == 0 \ || \ usc == 0$
2. 去除: pps 未更新的前后两个点
3. 计算: $x = utc + pps - pps_utc$
4. 计算: $y = usc_pps / 02.Freq$
5. 计算: Hough Transform ($x, y \rightarrow k, b$), $b_median = b$ 的中位数
6. 去除: $k == 0$
7. 去除: $k < 0.96 \ || \ k > 1.04$
8. 去除: $b < b_median - 20 \ || \ b > b_median + 20$
9. 选取: 做 (k, b) 的 2Dhistogram, 选取计数最大的 bin 为 ($k0, b0$)
10. 计算: $\delta = y - (k0 * x + b0)$
11. 去除: $abs(\delta) > 0.3$

¹ 去除和选取均指 usc pps utc 等数据组成的一组数

基于晶振频率计算的数据清洗

1. 计算: $f = \text{usc_pps} / \text{pps}$, $f_{\text{median}} = f$ 的中位数
2. 去除: $\text{abs}(f - f_{\text{median}}) > 0.1\% f_{\text{median}}$

重建结果 Result



能量重建 Energy Reconstruction

能量重建，指的是使用标定数据和 ADC 值计算 `tte4.fits` 中的能量。

Energy reconstruction is the calculation of energy in `tte4.fits` based on ADC value.

7.1 原始数据 Original data

7.1.1 02 星原始数据 Original data of GRID-02

`science0.fits` 初步处理的科学数据。文件夹为 `gridpath.sci_source_H`。

`science0.fits` preliminarily treated scientific data, directory is `gridpath.sci_source_H`.

`telemetry1.fits` 初步处理的监测数据。文件夹为 `gridpath.tel_source_H`。

`telemetry1.fits` preliminarily treated telemetry data, directory is `gridpath.tel_source_H`.

7.2 处理函数 Python functions

`gridfun.energy_recon()`

7.3 处理过程 Process

7.3.1 01 星处理过程 Process of GRID-01

1. 差值得到 V_{mon} 和 $LeakI$
2. 4 个通道对应的 breakdown voltage, $V_{br} = 01.V_{br}$
3. 根据观测计划判断期望的电压值, $V_{out} = 01.V_{out_dic}$
4. 计算每个通道的 over voltage, $V_{ov} = V_{mon} - 2000 * \text{Interpolate1D}[V_{down}, LeakI, V_{out} - V_{mon}] - V_{br}$
5. 计算每个通道的增益, $Gain = (V_{ov} + 0.4) * 1e6$
6. 计算地面实验中的增益, $V_{ovgnd} = 28 - 24.9$, $G_{aingnd} = (V_{ovgnd} + 0.4) * 1e6$
7. 计算能量, $Energy = ADCvalue * G_{aingnd} / Gain * 238.632 / 10197.246$

7.3.2 02 星处理过程 Process of GRID-02

相关数据 Related data

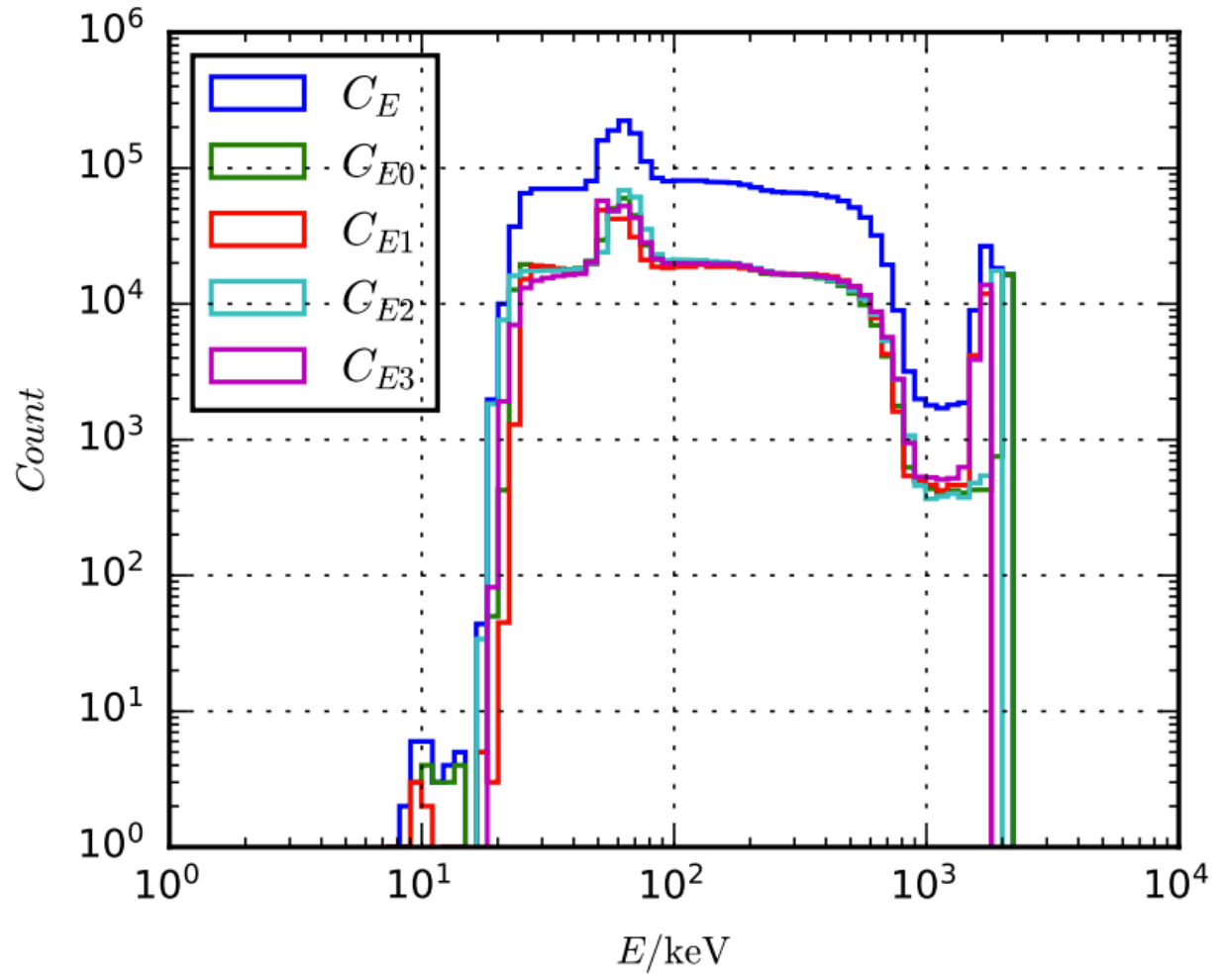
表 1: Related data

变量 Variables	简介 Introduction
ADC value	ADC 值, 来自初步处理的科学数据
temperature	温度, 来自初步处理的监测数据
voltage bias	偏压, 来自初步处理的监测数据
uScount	晶振计数, 来自初步处理的科学数据
UTC	计算得到的 UTC 时间
energy	计算得到的能量

重建过程 Reconstruction

1. 在 `gridfun.energy_recon()` 函数中, 因为监测数据和科学数据的 `usc` 以及 `utc` 不对应, 依据监测数据, 差值计算随时间变化的温度和偏压 `Interpolate & calculate temperature & voltage bias with respect to time in science0.fits`
2. 对 ADC 值, 使用函数 `gridfun.Adc_mapping.adcmap()` 进行温度 - 偏压矫正 Temperature-voltagebias correction of ADC value
3. 对矫正后的 ADC 值, 使用函数 `gridfun.Adc_mapping.ecmap()` 进行 ADC 值 - 能量转换 ADC value - energy convention

重建结果 Result



姿态数据 Attitude Data

姿态数据 `sun2.fits` 记录了探测器和卫星的姿态四元数随时间的变化，同时也用不同坐标系中探测器法线方向描述了探测器和太阳的相对位置关系。

Attitude data `sun2.fits` recorded the attitude of detectors and satellites with respect to time, while the relative position of detectors and sun can be described by unitary normal of detectors in various coordinates.

8.1 原始数据 Original data

8.1.1 02 星原始数据 Original data of GRID-02

`atti.csv` 姿态控制文件。文件夹为 `gridpath.atti_source_H`。

`atti.csv`, directory is `gridpath.atti_source_H`.

8.2 处理函数 Python functions

```
gridfun.makefits2()  
gridfun.extract_sun()  
gridfun.fits2_make()
```

8.3 处理过程 Process

8.3.1 02 星处理过程 Process of GRID-02

1. 使用`gridfun.makefits2()`, 查找`gridpath.attl_source_H` 下所有文件 Find all file in `gridpath.attl_source_H`
2. 不处理`gridpath.log_attl_path` 日志文件记录的 `attl.csv` 文件 Ignore `attl.csv` recorded in `gridpath.log_attl_path`
3. 使用函数`gridfun.extract_sun()` 提取 `attl.csv` 中的姿态数据 Extract attitude from `attl.csv` using functions `gridfun.extract_sun()`
4. 坐标系由 坐标系中心原点和 坐标轴指向决定。姿态数据按坐标系分为如下几种:
 1. 探测器为中心的天球坐标系 (坐标轴不随时间变化) `g_X_cel g_Y_cel g_Z_cel g_Ra_cel g_Dec_cel s_X_cel s_Y_cel s_Ra_cel s_Dec_cel`
 2. 探测器为中心的探测器坐标系 (坐标轴随探测器转动) `s_X_det s_Y_det s_Z_det s_Ra_det s_Dec_det`
5. 使用函数`gridfun.fits2_make()` 按天保存 `sun2.fits` 数据到`gridpath.sun_source_H`, 文件名为 `nnnnn_yymmddsun.fits` Save `sun2.fits` with repeat to date into directory `gridpath.sun_source_H`
6. 将处理过的 `attl.csv` 文件名保存到日志文件`gridpath.log_sun_path` Save file name of `attl.csv` into log file

地理坐标 Geographic coordinate

地理坐标数据 `earth6.fits` 记录了探测器随时间变化的经纬度和高度，以及地球在探测器坐标系内的坐标
Geographic coordinate data `earth6.fits` recorded the longitude, latitude and altitude of detectors with respect to time, as well as the earth orientation in detector-centered coordinate.

9.1 原始数据 Original data

9.1.1 02 星原始数据 Original data of GRID-02

`orb.txt` 姿态控制文件。文件夹为 `gridpath.earth_source_H`。
`orb.txt`, directory is `gridpath.earth_source_H`.

9.2 处理函数 Python functions

```
gridfun.makefits6()  
gridfun.extract_ear()  
gridfun.fits6_make()
```

9.3 处理过程 Process

9.3.1 02 星处理过程 Process of GRID-02

1. 使用`gridfun.makefits6()`, 查找`gridpath.earth_source_H` 下所有文件 Find all file in `gridpath.earth_source_H`
2. 不处理`gridpath.log_eart_path` 日志文件记录的 orb.txt 文件 Ignore orb.txt recorded in `gridpath.log_eart_path`
3. 使用函数`gridfun.extract_ear()` 提取 orb.txt 中的地理坐标信息 Geographic coordinate from orb.txt using functions `gridfun.extract_ear()`
4. 探测器为中心的探测器坐标系 (坐标轴随探测器转动) e_Ra_cel e_Dec_cel
5. 使用函数`gridfun.fits6_make()` 按天保存 earth6.fits 数据到`gridpath.earth_source_H`, 文件名为 `nnnnn_yymmddsun.fits` Save earth6.fits with repeat to date into directory `gridpath.earth_source_H`
6. 将处理过的 orb.txt 文件名保存到日志文件`gridpath.log_eart_path` Save file name of orb.txt into log file

图片文件的主要目的是直观地展示探测器运行状态和观测结果。

Main purpose of picture (quicklook) is to intuitively show the status of detector & observation results.

10.1 原始数据 Original data

10.1.1 02 星原始数据 Original data of GRID-02

10.2 处理函数 Python functions

```
statuspic.main()  
statuspic.draw_single_day()  
statuspic.chunk()
```

10.3 处理过程 Process

10.3.1 02 星处理过程 Process of GRID-02

1. 绘制过程有两种模式 Two modes of drawing

1. 对每一天绘图 Draw everyday pictures
2. 指定某一天进行绘图 Draw single day pictures

2. 读取 tte4.fits TTE 数据、sun2.fits 姿态数据、earth6.fits 地理坐标数据

3. 差值计算探测器坐标系内地球的方向，计算随时间变换的 AEM 粒子通量 Interpolate & calculate earth orientation in detector coordinate, calculate AEM flux with respect to time

4. 绘制一整轨的 .pdf 图片, 绘制按 5 min 分段的详细数据 Draw .pdf pictures for a whole orbit, draw picture for every 5 min
5. 同时保存与 .pdf 对应的 .png 文件

11.1 数据产品 Data products

11.1.1 02 星 GRID-02

.fits 文件 .fits files

GRID-pipeline 下, 运行

In GRID-pipeline, run

```
make sat=02 tte
```

绘图 Pictures

GRID-pipeline 下, 运行

In GRID-pipeline, run

```
make sat=02 pic
```

清除数据 Clean data

将 .fits 文件和绘图不包括原始数据, 一次性全部清除, GRID-pipeline 下, 运行

Clean .fits files & pictures, excluding source data. In directory GRID-pipeline, run

```
make sat=02 clean
```

11.2 观测计划检查 Check observation plan

```
python3 plancompare.py --cons 02
```

11.3 运行状况检查 Check running status

```
python3 runtime.py --cons 02
```

11.4 增益刻度 Gain calibration

```
python3 cali.py --mode eccoef && python3 cali.py --mode tempbias
```

残余问题 Problems Remaining

12.1 数据重复 Data Repetition

由于要保证数据传输的经济性，`raw.dat` 会有监测数据以及科学数据的重复。这种重复既可能发生在同一个 `raw.dat` 文件中，也可能发生在 `raw.dat` 文件之间。目前对这两种重复都没有主动去除。采用的方式是保存全部数据，即使有重复。所以在 `tte4.fits` 所属文件夹中会看到文件名类似（即包含的时间段类似）的文件。值得注意的是，这些文件对应的时间轴重建和能量重建都是自洽的，因为发生数据重复时一定会同时重复监测数据（用于重建时间轴）和科学数据（用于能量重建）。重复下传的文件中内容一般区别不大（但有存在一些小概率情况差异很大）。在完成 `tte4.fits` 后，我们可以对重复的数据进行 **筛选**，从所有数据中选取我们感兴趣的质量最好的数据。

The data repetition happens during data transfer from satellites. Data repetition may happens in one `raw.dat` itself, and also inter `.dat` files. Currently we only save them all without de-duplication. It is noted that repeated data will include monitor data and scientific data simultaneously, which results into the self-consistent of the timeline and energy reconstruction results.

13.1 Data in 01.py

```
name = '01'  
    name of detector 探测器名称  
Freq = 24050000.0  
    frequency of crystal oscillator 晶振频率  
sourprefix = '/srv/pipeline/01'  
    Input file directory 输入文件文件夹  
fitsprefix = '/srv/pipeline/01'  
    Output file directory 输出文件文件夹  
picsprefix = '/srv/pipeline/01'  
    Picture directory 图片文件文件夹  
O_A = [0, 0, -1]  
    Unitary normal of detector 探测器法线  
Vout_dic = {'_28_': 30.6, '_29_': 31.8, '_30_': 32.9, 'ivscan': -1}  
    Voltage set in observation plan 观测计划设定电压  
Vbr = [24.4497, 24.332, 24.3614, 23.7242]  
    Breakdown voltage 雪崩电压  
vdown_path = '/source_plan/Vdown.txt'  
    Vdown file Vdown 文件  
leaki_path = '/source_plan/LeakI.txt'  
    LeakI file LeakI 文件  
datestrcolumn = 16  
    Column number of time in att1.csv 姿态文件中的星上时间列数  
quatstrcolumn = 17  
    Column number of quaternion in att1.csv 姿态文件中的四元数列数
```

13.2 Data in 02.py

```
name = '02'
    name of detector 探测器名称
Freq = 20000000.0
    frequency of crystal oscillator 晶振频率
sourprefix = '/srv/gridftp/02'
    Input file directory 输入文件文件夹
fitsprefix = '/srv/pipeline/02'
    Output file directory 输出文件文件夹
picsprefix = '/srv/pipeline/02'
    Picture directory 图片文件文件夹
O_A = [0, 0, -1]
    Unitary normal of detector 探测器法线
datestrcolumn = 15
    Column number of time in atti.csv 姿态文件中的星上时间列数
quatstrcolumn = 24
    Column number of quaternion in atti.csv 姿态文件中的四元数列数
tempbias = '/srv/gridftp/02/source_plan/tempbias_ch'
    temperature-bias calibration path 温度-偏压矫正标定文件
ec_data = '/srv/gridftp/02/source_plan/ec_data_ch'
    energy-channel calibration data path 能量-通道关系标定数据文件
ec_coef = '/srv/gridftp/02/source_plan/ec_coef.json'
    energy-channel calibration path 能量-通道关系标定文件
newec_coef = '/srv/gridftp/02/source_plan/newec_coef.json'
    new energy-channel calibration path 新的能量-通道关系标定文件
newtempbias = '/srv/gridftp/02/source_plan/tempbias.json'
    new temperature-bias calibration path 新的温度-偏压矫正标定文件
```

14.1 Functions in make.py

main()

Pipeline main function GRID-pipeline 主函数，由输入文件得到输出文件

14.2 Commandline interface in make.py

14.2.1 make.py

```
usage: make.py [-h] [--cons CONS] [--delete]
```

-h, --help

show this help message and exit

--cons <cons>

gallery defined constants such as file path gallery 文件定义的文件路径等常数

--delete

whether delete .fits (default: False) 是否删除所有 .fits

15.1 Data in gridfun.py

```
DT_LAUNCH = datetime.datetime(2018, 10, 29, 0, 0, tzinfo=datetime.timezone.utc)
    GRID-01 launch date 2018-10-29 GRID-01 入轨时间 2018-10-29

UTC_LAUNCH = 1540771200.0
    GRID-01 launch date Unix timestamp GRID-01 入轨时间 Unix 时间戳

TZ_UTC_8 = datetime.timezone(datetime.timedelta(seconds=28800))
    timedelta 8 hours 8 小时时间差

N = 44
    N 科学数据包中，每个包有 N 个 usc-adc
```

15.2 Classes in gridfun.py

```
class Saa_mapping
```

```
    __init__()
        Initialization Saa_mapping instance 初始化 Saa_mapping 实例
```

```
    fun_saa()
        Generate coordinate - flux functions 生成坐标 - 粒子通量函数
```

返回

- **mapping** (*object*) –coordinate - flux function 某种坐标 - 粒子通量函数
- **indmapping** (*object*) –Nearest-neighbor interpolation 最近邻插值函数
- **fluxm** (*object*) –max AEM flux 最大 AEM 模型的粒子通量

```
readSAA(saacoord_path, saaAE8_path)
    Read AEM coordinate & flux 读取 AEM 模型中的坐标和粒子通量
```

参数

- **saacoord_path** (*str*) –path of AEM coordinate AEM 模型的坐标文件路径
- **saaAE8_path** (*str*) –path of AEM flux AEM 模型的粒子通量文件路径

返回

- **coord** (*array_like*) –AEM coordinate AEM 模型的坐标
- **flux** (*array_like*) –AEM flux AEM 模型的粒子通量

class Adc_mapping (*const*)**__init__** (*const*)

Initialization Adc_mapping instance 初始化 Adc_mapping 实例

参数 const (*dict*) –gallery defined constants such as file path gallery 文件定义的文件路径等
常数**adcmap** (*i, adcv, temp, bias*)

temperature-voltagebias correction of ADC value 对 ADC 值进行温度 - 偏压修正

参数

- **i** (*int*) –channel number 通道号
- **adc_v** (*int*) –uncorrected ADC value 未经修正的 ADC 值
- **temp** (*float*) –measured temperature 实测温度
- **bias** (*float*) –measured voltage bias 实测偏压

返回 adcv –corrected ADC value 修正过的 ADC 值**返回类型** float**ecmap** (*i, adcv_r, temp, bias*)ADC value - energy convention based on temperature-voltagebias correction 在温度 - 偏压修正后, 进行
ADC 值 - 能量转换**参数**

- **i** (*int*) –channel number 通道号
- **adc_v_r** (*int*) –uncorrected ADC value 未经修正的 ADC 值
- **temp** (*float*) –measured temperature 实测温度
- **bias** (*float*) –measured voltage bias 实测偏压

返回 E –energy 能量**返回类型** float**readcali** (*const*)

Read calibration .json files 读取标定数据 .json 文件

参数 const (*dict*) –gallery defined constants such as file path gallery 文件定义的文件路径等
常数**返回**

- **tempbiascoef** (*dict*) –coefficients of temperature-voltagebias correction 温度 - 偏压修正参
数

- **ecccoef** (*dict*) –channel-energy convention functions & coefficients ADC 值 - 能量转化函数和参数

15.3 Functions in gridfun.py

pol2d (*t, p22, p21, p20, p12, p11, p10, p02, p01, p00*)

Binary quadratic function 二元二次函数

参数

- **t** (*array_like*) –array of (*x*, *y*) (*x*, *y*) 序列
- **p** (*sequence of float*) –9 coefficient 各项参数, 共 9 个

返回 **z** –result 计算结果

返回类型 *array_like*

read_raw (*filename*)

Read raw.dat file, each position correspond to 1 byte 将 raw.dat 读取, 每个字符对应 1 byte

参数 **filename** (*bin*) –raw.dat filename raw.dat 文件名

返回 **raw_data** –rawdata 读出的字符串

返回类型 *bin*

find_pattern_pos (*sdata, pattern*)

Find the start positions of the pattern in data 在二进制数据中找到相应的 pattern 的起始位置

参数

- **sdata** (*bin*) –source data 二进制数据
- **pattern** (*bin*) –pattern

返回

- **pos** (*array_like*) –position index array pattern 的起始位置
- **length** (*array_like*) –length pattern 长度

check_crc16 (*sdata, src16*)

Cyclic redundancy check. Check the CRC16 value of the data with given CRC16 value 循环冗余校验

参数

- **sdata** (*bin*) –source data 原数据
- **src16** (*bin*) –CRC16 value in source data 校验码

返回 **bool** –True means correct and False means incorrect CRC16 校验是否正确

返回类型 *bool*

divide_u (*usc, utc=None*)

Store the uScout of new telemetry' s beginning, divide data respect to orbit 按照 *usc* 和 *utc* 的跳变将数据按轨分开

参数

- **usc** (*array_like*) –count of crystal oscillator 晶振计数
- **utc** (*array_like*) –UTC+0 time UTC 时间

返回 **pos** –divided position index 每轨起始索引

返回类型 `array_like`

rm_frag (*pos, l*)

Remove the tiny fragments 剔除序列中过小的数据段

参数

- **pos** (*array_like*) –divided position index 每轨起始索引
- **l** (*int*) –len threshold 长度阈值

返回 **pos** –washed divided position index 清洗过的每轨起始索引

返回类型 `array_like`

omi2pos (*vali*)

Convert label to position 将示性标签转化为索引序列

参数 **vali** (*array_like, bool*) –valid label 示性标签

返回 **pos** –divided position index 每轨起始索引

返回类型 `array_like`

pos2omi (*pos, len_n*)

Convert position to label 将索引序列转化为示性标签

参数

- **pos** (*array_like*) –divided position index 每轨起始索引
- **len_n** (*int*) –length of array 序列总长度

返回 **vali** –valid label 示性标签

返回类型 `array_like, bool`

find_best_part (*t, d=None*)

Find best part in sequence which contains duplicated data 在可能含有重复的数据的序列中找到最优的部分

参数

- **t** (*array_like*) –duplicated data 可能含有重复的数据
- **d** (*array_like*) –adjoint sequence 伴随序列

返回

- **b_t** (*array_like*) –divided position index of best part 最优组分对应的起始索引
- **best** (*array_like*) –best part 最优组分
- **n** (*int*) –length of best part 最优组分的长度
- **b_d** (*array_like*) –corresponding part of adjoint sequence 最优组分在伴随序列对应的部分

man_hough (*X, Y*)

Manual Hough transfer 手动 Hough 变换

参数

- **X** (*array_like*) –(*x, y*) 数据点中的 X
- **Y** (*array_like*) –(*x, y*) 数据点中的 Y

返回 **k & b** –all (*k, b*) Hough 变换得到的所有 (*k, b*)

返回类型 `array_like`

utc_to_days (*time_u*)

Convert UTC time to days from UTC_LAUNCH in UTC+0 将 **UTC+0** 时间戳转化为 GRID-01 自发射日 UTC_LAUNCH 起的天数

参数 **time_u** (*float*) –UTC timestamp in UTC+0 **UTC+0** 时间戳

返回

- **days_i** (*int*) –number of days since launch in UTC+0, int 自 GRID-01 自发射日 UTC_LAUNCH 起的天数
- **date_str** (*str*) –date str in UTC+0, example: 181029 自 GRID-01 自发射日 UTC_LAUNCH 起的格式化天数字符串 `yyymmdd`

utc_to_str (*time_u*)

Convert utc time to string 将 **UTC+0** 时间戳转化为字符串

参数 **time_u** (*float*) –UTC timestamp in UTC+0 **UTC+0** 时间戳

返回 **str_u** –time str in UTC+0 with accuracy of seconds 格式化的时间字符串 `yyymmddHHMMSS`

返回类型 `str`

days_to_date (*days*)

Output the date of number of days after launch 由 GRID-01 发射后的天数，输出格式化的日期

参数 **days** (*int*) –number of days in UTC+0 after launch GRID-01 发射后的天数，任务起始后的天数

返回 **date_str** –formatted date string 格式化的日期，`yyymmdd`

返回类型 `str`

date_to_days (*date_str*)

Output the number of days after launch of given date 由格式化的日期，输出 GRID-01 发射后的天数

参数 **date_str** (*str*) –formatted date in UTC+0 string 格式化的日期，`yyymmdd`

返回 **days** –number of days in UTC+0 after launch GRID-01 发射后的天数，任务起始后的天数

返回类型 `int`

del_hidden (*name_list*)

Delete hidden file and pdf file names in list of files 在文件路径列表中删除隐藏文件和 .pdf

参数 **name_list** (*list*) –file name list 文件名列表

返回 **name_re** –washed file name list 清洗后的文件名列表

返回类型 `list`

xyz_to_radec (*X_t*)

Transform xyz coordinate to Ra & Dec 笛卡尔坐标系坐标转天球坐标系坐标

参数 **X_t** (*array_like*) –xyz coordinate array(unnormlized array) in Cartesian coordinate system 笛卡尔坐标系中的坐标指向，未进行归一化

返回 **radec** –Ra & Dec (array) in Celestial coordinate system 天球坐标系中的坐标指向，Ra & Dec

返回类型 `array_like`

normal_array (*X_t*)

Normalize the array 将数组归一化

参数 **X_t** (*array_like*) –unnormlized array 未归一化数组

返回 **X** –normalized array 归一化数组

返回类型 `array_like`

radec_to_xyz (*X_t*)

Transform Ra & Dec to xyz coordinate 天球坐标系坐标转笛卡尔坐标系坐标

参数 **X_t** (*array_like*) -Ra & Dec (array) in Celestial coordinate system 天球坐标系中的坐标指向, Ra & Dec

返回 **xyz** -xyz coordinate array(unnormalized array) in Cartesian coordinate system 笛卡尔坐标系中的坐标指向, 未进行归一化

返回类型 `array_like`

intersection_angle (*x, y*)

Calculate intersection angle between two direction in celestial 计算天球中两个方向的夹角

参数

- **x** (*array_like*) -Ra
- **y** (*array_like*) -Dec

返回 **rad** -intersection angle in rad 以 rad 为单位的夹角

返回类型 `array_like`

my_cmap ()

Create specific cmap 自定义 color map

返回 DIY cmap 自定义的 color map

返回类型 `newcmp`

del_dir_tree (*path*)

Delete path recursively 递归地删除文件夹

参数 **path** (*str*) -directory 文件夹

delete (*paths*)

Delete directories permanently 永久递归删除文件夹

参数 **paths** (*array_like, str*) -directory list 文件夹路径列表

detectCode (*path*)

Get file encoding 提取文件 encoding

参数 **path** (*str*) -file path 文件路径

返回 encoding of file 文件的 encoding

返回类型 `encoding`

find_base (*fdat, inibsl=None, inistd=None, r=2, delta=0.01*)

Calculate baseline and the fluctuation of baseline

参数

- **fdat** -data.
- **inibsl** -initial baseline.
- **inistd** -initial standard deviation.
- **r** -fold of deviation.
- **delta** -threshold when stop iteration.

返回

- *bsl* –baseline.
- *std* –standard deviation.

extract_tel_events (*raw_data*)

Match and extract monitor data in raw data 匹配提取二进制数据中的监测数据, 组织成 numpy structured data

参数 **raw_data** (*str*) –raw data 原始数据

返回

- **tel** (*array_like*) –telemetry data 监测数据
- **crc_error_data** (*str*) –telemetry data with CRC error CRC 校验错误的监测数据

extract_sci_events (*raw_data*)

Match and extract science data in raw data 匹配提取二进制数据中的科学数据, 组织成 numpy structured data

参数 **raw_data** (*str*) –raw data 原始数据

返回

- **sci** (*dict*) –science data 科学数据
- **crc_error_data** (*str*) –telemetry data with CRC error CRC 校验错误的监测数据

makefits3 (*plan_source_H*, *fits3_H*, *fitspath*, *const*, *excel*)

Generate fits file containing observation plans 生成包含每一轨计划的起始终止时间、设定四元数、设定电压等数据的 Observation_plan3.fits 文件

参数

- **plan_source_H** (*str*) –source directory 原始文件的文件夹
- **fits3_H** (*str*) –Observation_plan3.fits directory Observation_plan3.fits 文件的文件夹
- **fitspath** (*str*) –Observation_plan3.fits path Observation_plan3.fits 路径
- **excel** (*str*) –Excel observation plan path for GRID-01 Excel 文件路径, GRID-01 专用

makefits01 (*dat_source_H*, *log_dat_path*, *fits0_H*, *fits1_H*, *fits3path*, *const*)

Extract science0.fits & telemetry1.fits 提取 science0.fits 初步处理的科学数据和 telemetry1.fits 初步处理的监测数据

参数

- **dat_source_H** (*str*) –raw.dat source path raw.dat 原始文件的文件夹
- **log_dat_path** (*str*) –raw.dat log path raw.dat 的日志文件路径
- **fits0_H** (*str*) –science0.fits directory science0.fits 文件夹
- **fits1_H** (*str*) –telemetry1.fits directory telemetry1.fits 文件夹
- **fits3path** (*str*) –Observation_plan3.fits path Observation_plan3.fits 路径
- **const** (*dict*) –gallery defined constants such as file path gallery 文件定义的文件路径等常数

fits0_save (*sci*, *fitspath*)

Save sci data to science0.fits file 将科学数据保存成 science0.fits

参数

- **sci** (*dict*) –science data 科学数据

- **fitspath** (*str*) –science0.fits 文件夹

make01electronics (*tel, s_pat, utc, pos, fits3p, const*)

monitor calculator for GRID-01 时间轴重建之后计算运行中的电压电流等监测数据, 仅对 GRID-01 有效

参数

- **tel** (*array_like*) –telemetry data 监测数据
- **s_pat** (*array_like, bool*) –valid part assigned by timeline reconstruction 监测数据中可用部分对应的布尔值
- **utc** (*array_like*) –reconstructed utc time 时间轴重建得到的 utc 时间
- **pos** (*array_like*) –indices of valid part 监测数据中可用部分对应的索引
- **fits3p** (*str*) –path of fits3
- **const** (*dict*) –gallery defined constants such as file path gallery 文件定义的文件路径等常数

返回

- **vout** (*array_like*) –vout set value vout 设定值
- **iop** (*array_like*) –operation current 运行电流, 单位为 uA
- **vop** (*array_like*) –operation voltage 运行电压, 单位为 V
- **label** (*array_like*) –label of valid part 重建结果对应的标签
- **vmon** (*array_like*) –original voltage 原始监测电压
- **imon** (*array_like*) –original current 原始监测电压

make02electronics (*tel, s_pat, pos, const*)

monitor calculator for GRID-02 时间轴重建之后计算运行中的电压电流等监测数据, 仅对 GRID-02 有效

参数

- **tel** (*array_like*) –telemetry data 监测数据
- **s_pat** (*array_like, bool*) –valid part assigned by timeline reconstruction 监测数据中可用部分对应的布尔值
- **pos** (*array_like*) –indices of valid part 监测数据中可用部分对应的索引
- **const** (*dict*) –gallery defined constants such as file path gallery 文件定义的文件路径等常数

返回

- **vout** (*array_like*) –vout set value vout 设定值, 对 GRID-02 来说是 NaN
- **iop** (*array_like*) –operation current 运行电流, 单位为 uA
- **vop** (*array_like*) –operation voltage 运行电压, 单位为 V
- **label** (*array_like*) –label of valid part 重建结果对应的标签, 对 GRID-02 来说均为 4
- **vmon** (*array_like*) –original voltage 原始监测电压
- **imon** (*array_like*) –original current 原始监测电压

fits1_recon (*tel_t, fits1path, fits3p, const*)

Save reconstructed data to telemetry1.fits file, save orbit division, usc-utc correspondence in every orbit 将监测数据按轨分开, 进行 usc-utc 对应关系重建, 将重建结果存入 telemetry1.fits 文件, 同时相关过程存入与 telemetry1.fits 文件相同的文件夹

参数

- **tel_t** (*array_like*) –telemetry data 监测数据
- **fits1path** (*str*) –telemetry1.fits path telemetry1.fits 路径
- **fits3p** (*str*) –Observation_plan3.fits path Observation_plan3.fits 路径
- **const** (*dict*) –gallery defined constants such as file path gallery 文件定义的文件路径等常数

makefits5 (*tel_source_H, log_tel_path, log_orb_H, fits5_H*)

Save orbits utc in tel 保存 orb_mon5.fits

参数

- **tel_source_H** –telemetry1.fits directory telemetry1.fits 文件夹
- **log_tel_path** –telemetry1.fits log path telemetry1.fits 的日志文件路径
- **log_orb_H** –orb_mon5.fits record path orb_mon5.fits 的记录文件路径，记录了每天的分轨起始时间信息
- **fits5_H** –orb_mon5.fits directory orb_mon5.fits 文件夹

energy_recon (*i, const, sci_fitsname, tel_fitsname*)

Reconstruction of energy 能量重建

参数

- **i** (*int*) –mark number in multiprocessing 并行计算中的标号
- **const** (*dict*) –gallery defined constants such as file path gallery 文件定义的文件路径等常数
- **sci_fitsname** (*str*) –science0.fits full path science0.fits 绝对路径
- **tel_fitsname** (*str*) –telemetry1.fits full path telemetry1.fits 绝对路径

返回

- **t** (*array_like*) –reconstructed utc time 时间轴重建得到的 utc 时间
- **E** (*array_like*) –reconstructed energy 重建得到的能量
- **ch** (*array_like*) –channel 通道号码，从 0 开始
- **adcv** (*array_like*) –adc value ADC 值
- **days_date** (*array_like, str*) –days & date string 上述数据对应的格式化日期 nnnnnn_yymmdd
- **orb_span** (*array_like, str*) –stating & ending time of each orbit TTE 数据按轨分开的起始时间
- **n** (*int*) –length of original TTE TTE 原始数据总长度

makefits4 (*sci_source_H, log_sci_path, tel_source_H, fits4_H, const*)

Reconstruction of Time Tagged Events & save tte4.fits 重建 TTE，并保存 tte4.fits

参数

- **sci_source_H** (*str*) –science0.fits directory science0.fits 文件夹
- **log_sci_path** (*str*) –science0.fits log path science0.fits 的日志文件路径

- **tel_source_H**(*str*)-telemetry1.fits directory telemetry1.fits 文件夹
- **fits4_H**(*str*)-tte4.fits directory tte4.fits 文件夹
- **const**(*dict*)-gallery defined constants such as file path gallery 文件定义的文件路径等常数

fits4_make(*t, E, ch, adcv, days_date, orb_span, fits4_H*)
Save tte4.fits file 保存 tte4.fits TTE 数据文件

参数

- **t**(*array_like*)-reconstructed utc time 时间轴重建得到的 utc 时间
- **E**(*array_like*)-reconstructed energy 重建得到的能量
- **ch**(*array_like*)-channel 通道号码, 从 0 开始
- **adcv**(*array_like*)-adc value ADC 值
- **days_date**(*array_like, str*)-days & date string 上述数据对应的格式化日期 nnnnn_yymmdd
- **orb_span**(*array_like, str*)-stating & ending time of each orbit TTE 数据按轨分开的起始时间
- **fits4_H**(*str*)-tte4.fits directory tte4.fits 文件夹

makefits2(*atti_source_H, log_atti_path, fits2_H, const*)
Extract sun2.fits attitude data 提取 sun2.fits 探测器姿态、太阳-探测器相对位置数据

参数

- **atti_source_H**(*str*)-atti.csv source path atti.csv 原始文件的文件夹
- **log_atti_path**(*str*)-atti.csv log path atti.csv 的日志文件路径
- **fits2_H**(*str*)-sun2.fits directory sun2.fits 文件夹
- **const**(*dict*)-gallery defined constants such as file path gallery 文件定义的文件路径等常数

extract_sun(*atti_file, const*)
Extract quaternion & timestamp in atti.csv, calculate related sun data using astropy.coordinates.solar_system_ephemeris 提取 atti.csv 中的四元素和时间戳, 并使用 astropy.coordinates.solar_system_ephemeris 计算相关的探测器-太阳相对位置数据

参数

- **atti_file**(*str*)-atti.csv file path atti.csv 原始文件的文件路径
- **const**(*dict*)-gallery defined constants such as file path gallery 文件定义的文件路径等常数

返回

- **time_orien**(*array_like*)-timestamp 时间戳
- **quat**(*array_like*)-quaternion of detector 探测器四元数
- **grid_orien**(*array_like*)-Unitary normal of detector in celestial coordinate 天球坐标系内的探测器法线指向, 分为探测器为中心的 Cartesian coordinate system (xyz) 和 Celestial coordinate system (RaDec)
- **sun_orien**(*array_like*)-Sun orientation in celestial coordinate and detector coordinate 天球坐标系内的太阳方向、探测器坐标系内的太阳方向, 探测器为中心的天球坐标系和探测

器为中心的探测器坐标系, 分为 Cartesian coordinate system (xyz) 和 Celestial coordinate system (RaDec)

- **days_date** (*array_like*) –days & date string 上述数据对应的格式化日期 nnnnnn_yymmdd

fits2_make (*time_orien, quaternion, grid_orien, sun_orien, days_date, fits2_H*)

Save attitude data to sun2.fits file 将姿态数据保存成 sun2.fits

参数

- **time_orien** (*array_like*) –timestamp 时间戳
- **quat** (*array_like*) –quaternion of detector 探测器四元数
- **grid_orien** (*array_like*) –Unitary normal of detector in celestial coordinate 天球坐标系内的探测器法线指向, 分为 Cartesian coordinate system (xyz) 和 Celestial coordinate system (RaDec)
- **sun_orien** (*array_like*) –Sun orientation in celestial coordinate and detector coordinate 天球坐标系内的太阳方向、探测器坐标系内的太阳方向, 分为 Cartesian coordinate system (xyz) 和 Celestial coordinate system (RaDec)
- **days_date** (*array_like*) –days & date string 上述数据对应的格式化日期 nnnnnn_yymmdd
- **fits2_H** (*str*) –sun2.fits directory sun2.fits 文件夹

makefits6 (*txt_source_H, log_txt_path, fits6_H*)

Extract earth6.fits geographical information 提取 earth6.fits 卫星地球轨道、地球-探测器相对位置数据

参数

- **txt_source_H** (*str*) –orb.txt source path orb.txt 原始文件的文件夹
- **log_txt_path** (*str*) –orb.txt log path orb.txt 的日志文件路径
- **fits6_H** (*str*) –earth6.fits directory earth6.fits 文件夹

extract_ear (*ear_file*)

Extract geographical information in orb.txt 提取 orb.txt 中的地理坐标数据, 包括经纬度、高度、探测器-地球相对位置等

参数 **ear_file** (*str*) –orb.txt file path orb.txt 原始文件的文件路径

返回

- **time_geo** (*array_like*) –timestamp 时间戳
- **grid_geo** (*array_like*) –Longitude, latitude & altitude of detector 探测器的经纬度和高度
- **ear_orien** (*array_like*) –Earth' s orientation in detector coordinate 探测器坐标系内的地球坐标
- **days_date** (*array_like*) –days & date string 上述数据对应的格式化日期 nnnnnn_yymmdd

fits6_make (*time_geo, grid_geo, ear_orien, days_date, fits6_H*)

Save geographical information to sun2.fits file 将地理坐标数据保存成 sun2.fits

参数

- **time_geo** –timestamp 时间戳
- **grid_geo** –Longitude, latitude & altitude of detector 探测器的经纬度和高度
- **ear_orien** –Earth' s orientation in detector coordinate 探测器坐标系内的地球坐标

- **days_date** –days & date string 上述数据对应的格式化日期 nnnnnn_yymmdd
- **fits6_H** –earth6.fits directory earth6.fits 文件夹

vop_normal (*fits5_H, r=2*)

Find over voltage baseline and standard deviation when working normally

参数

- **fits5_H** –fits5 head.
- **r** –fold of deviation.

返回

- *vop_bsl* –vop baseline.
- *vop_std* –vop standard deviation.

snip_baseline (*h, itera=0.03333333333333333*)

参数

- **h** –
- **itera** –

返回

返回类型 baseline

getHTMLText (*url, pagefile, force=False*)

参数

- **url** –
- **pagefile** –

返回

返回类型 force

extract_sun_ear (*t, sun_source_H, earth_source_H*)

参数

- **t** –
- **sun_source_H** –
- **earth_source_H** –

返回

- *sun_orien*
- *ear_orien*
- *sun_ear_t*

read_sun_ear (*days_date, sun_source_H, earth_source_H*)

参数

- **days_date** –
- **sun_source_H** –
- **earth_source_H** –

返回

- *sun_orien*
- *ear_orien*

16.1 File path in gridpath.py

```
log_path = '/log'
    path of log 日志文件文件夹

dat_source_H = '/source_dat'
    path of raw.dat transferred data from satellite .dat 卫星下传二进制数据文件夹

log_dat_path = '/log/log_dat.csv'
    log of raw.dat raw.dat 卫星下传二进制数据的日志文件

atti_source_H = '/source_atti'
    path of atti.csv file atti.csv 姿控数据文件夹

log_atti_path = '/log/log_atti.csv'
    log of atti.csv atti.csv 姿控数据的日志文件

txt_source_H = '/source_txt'
    path of earth orb.txt file orb.txt 地球轨道数据文件夹 https://www.space-track.org

log_txt_path = '/log/log_txt.csv'
    log of orb.txt orb.txt 地球轨道文件的日志文件

sci_source_H = '/fits0'
    path of science0.fits science0.fits 初步处理的科学数据文件夹

log_sci_path = '/log/log_sci.csv'
    log of science0.fits data science0.fits 初步处理的科学数据的日志文件

tel_source_H = '/fits1'
    path of telemetry1.fits telemetry1.fits 初步处理的监测数据的文件夹

log_tel_path = '/log/log_tel.csv'
    log of telemetry1.fits data telemetry1.fits 初步处理的监测数据的日志文件

sun_source_H = '/fits2'
    path of sun2.fits sun2.fits 探测器姿态、太阳-探测器相对位置数据的文件夹
```

```
log_sun_path = '/log/log_sun.csv'
    log of sun2.fits data sun2.fits 探测器姿态、太阳-探测器相对位置数据的日志文件

plan_excel = '/source_plan/天格观测记录.xlsx'
    path of observation plan for GRID-01 01 星手动制作的观测计划文件夹

bin_source_H = '/source_plan/cmdbin/cmd.txt'
    path of observation plan for GRID-02 02 星自动生成的.txt 观测计划文件夹 https://github.com/GRID-datagroup/GRID-plan

plan_source_H = '/fits3'
    path of Observation_plan3.fits Observation_plan3.fits 观测计划记录文件夹

fits3p = '/fits3/Observation_plan.fits'
    path of Observation_plan3.fits Observation_plan3.fits 观测计划记录文件

tEnergy_source_H = '/fits4'
    path of tte4.fits tte4.fits Time Tagged Event (TTE) 文件夹

log_tte_path = '/log/log_tte.csv'
    log of tte4.fits tte4.fits Time Tagged Event (TTE) 文件的日志文件

orbmon_source_H = '/fits5'
    path of orb_mon5.fits orb_mon5.fits 分轨监测数据的文件夹

log_orb_H = '/log/log_orb'
    log head of orb_mon5.fits data orb_mon5.fits 文件中各分轨时间起始点的记录文件

earth_source_H = '/fits6'
    path of earth6.fits earth6.fits 卫星地球轨道、地球-探测器相对位置数据的文件夹

log_eart_path = '/log/log_eart.csv'
    log of earth6.fits data earth6.fits 卫星地球轨道、地球-探测器相对位置数据的日志文件

pic_source_H = '/picture'
    path of saved Quicklook pictures Quicklook 图片文件夹

saacoord_path = '/srv/pipeline/saa/coordsAEM_MAX_0.04MeV.txt'
    path of AEM coordinate AEM 模型的坐标文件 https://www.spennis.oma.be/help/system/toc.html

saaAE8_path = '/srv/pipeline/saa/AEM_MAX_0.04MeV.txt'
    path of AEM flux AEM 模型的粒子通量文件
```

17.1 Functions in cali.py

main()

Calculate ADC value - energy conversion coefficients, temperature-voltagebias correction coefficients 增益刻度, 计算 ADC 值 - 能量转换系数, 温度 - 偏压修正系数

17.2 Commandline interface in cali.py

17.2.1 cali.py

```
usage: cali.py [-h] [--mode {eccoef,tempbias}]
```

-h, --help

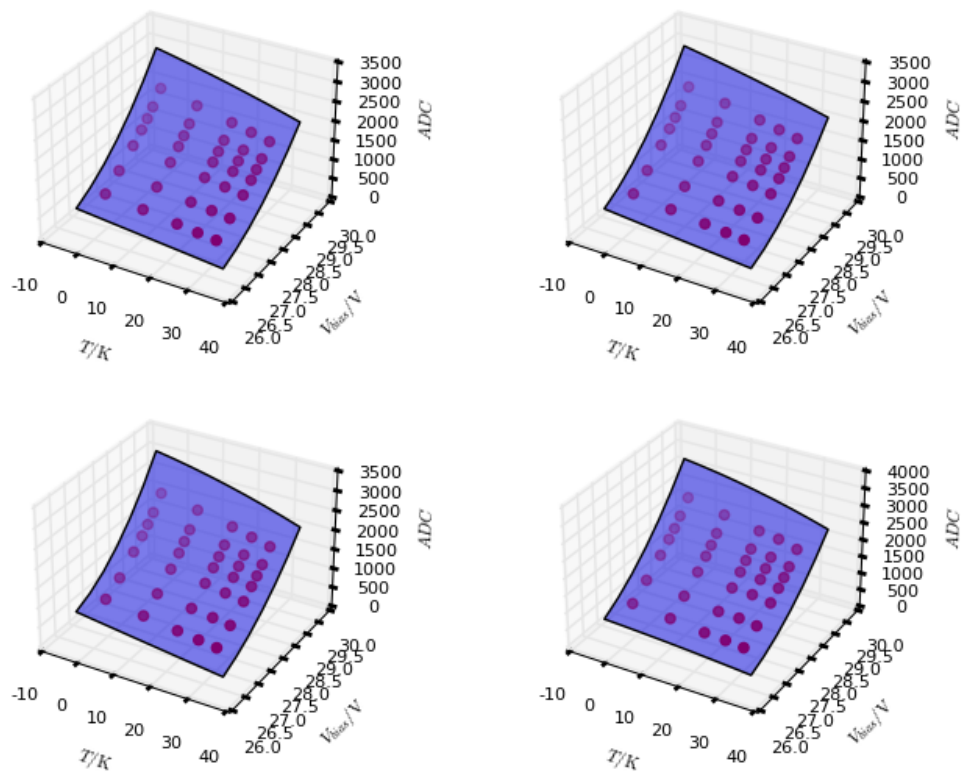
show this help message and exit

--mode {eccoef,tempbias}

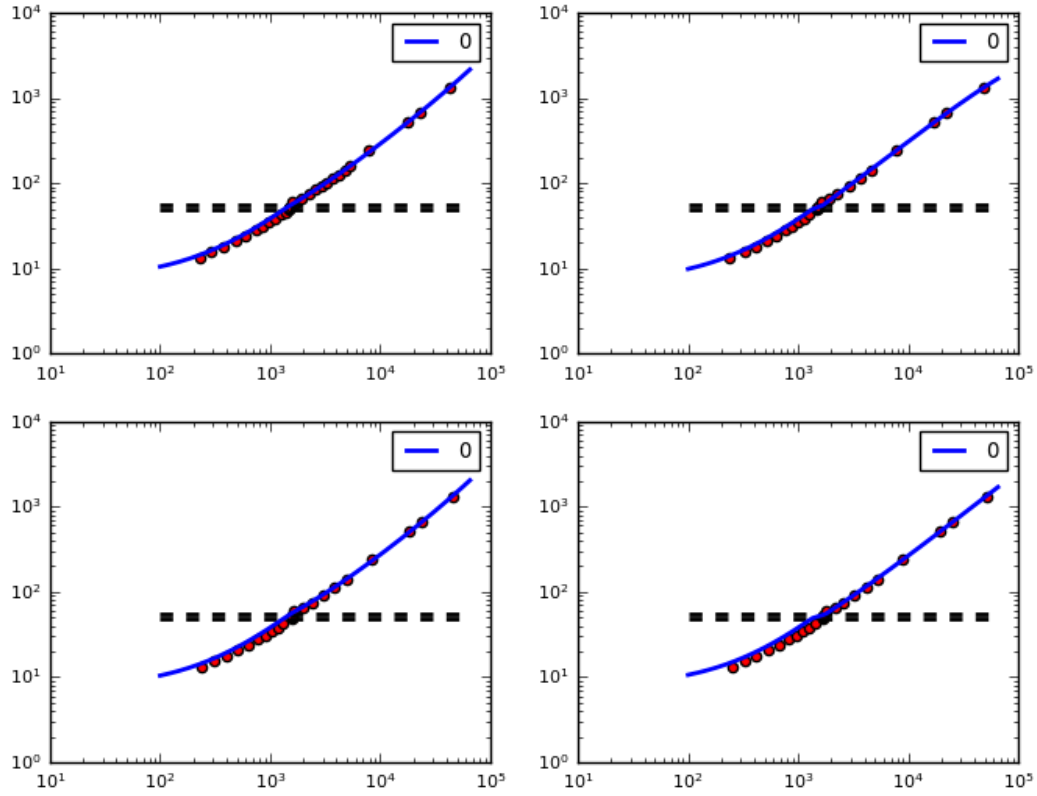
Mode of calibration 标定数据处理模式

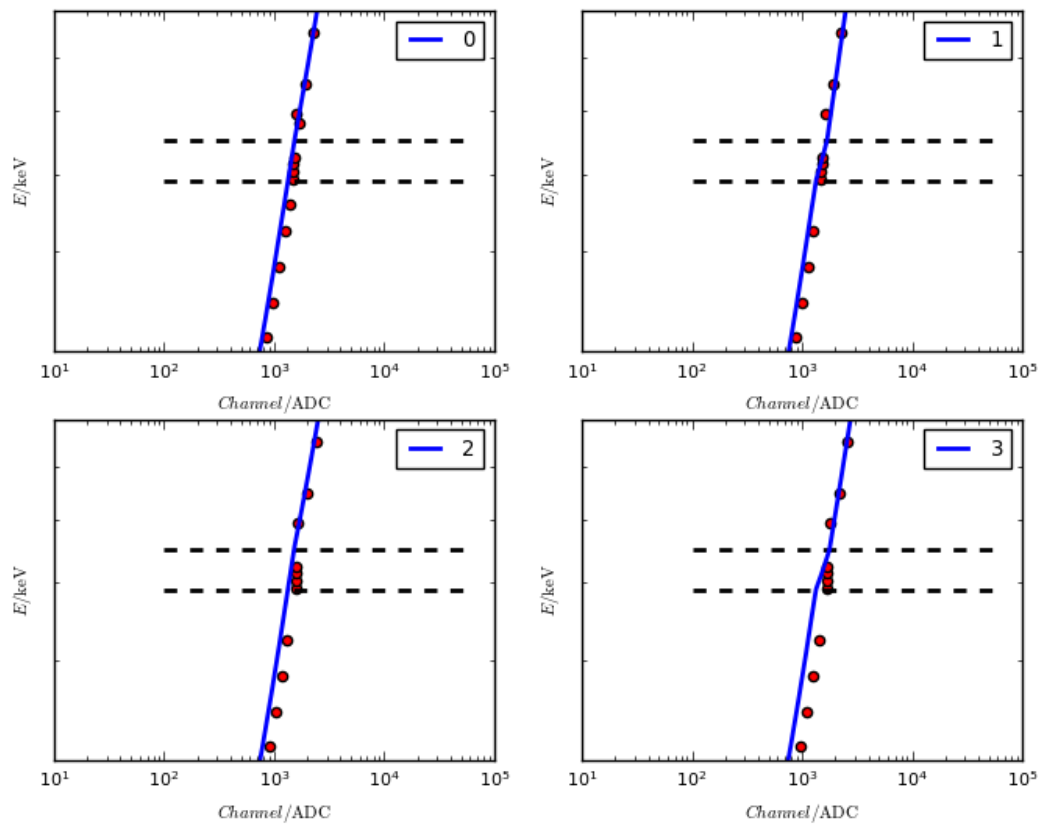
17.2.2 刻度结果 Calibration results

温度 - 偏压校正 Temperature-voltagebias correction coefficients (by Dacheng Xu)



ADC 值 - 能量转换 ADC value - energy convention coefficients (by Dacheng Xu)





18.1 Functions in statuspic.py

main()

Draw pictures iteratively 主函数，按日期绘制图片

draw_single_day (*date_str, tte_done, pic_path, f_saa*)

Draw pictures iteratively 主函数，按日期绘制图片

参数

- **date_str** (*str*) –formatted date string 格式化的日期, *yymmdd*
- **tte_done** (*list*) –已经处理过的 *tte4.fits* treated *tte4.fits*
- **pic_path** (*str*) –path of saved Quicklook pictures Quicklook 图片文件夹
- **f_saa** (*Saa_mapping*) –Saa_mapping instance

chunk (*tte_t, k, m, pdf, saa_edge, ind_intime, pngfold, ff=None*)

Draw picture for every 5 min 按 5 min 将每轨数据分开绘图

参数

- **tte_t** (*array_like*) –TTE
- **k** (*int*) –number of pic 图片标号
- **m** (*int*) –total number of pics 图片总数
- **pdf** (*PdfPages instance*) –
- **saa_edge** (*array_like*) –time for AEM flux calculation 用于计算 AEM 模型的粒子通量的时间数据
- **ind_intime** (*array_like*) –AEM flux AEM 模型的粒子通量
- **pngfold** (*str*) –directory of png files 存储 png 的文件夹
- **ff** (*Figure instances*) –定义的图片实例

返回 `ff` 定义的图片实例

返回类型 `Figure instances`

18.2 Commandline interface in `statuspic.py`

18.2.1 `statuspic.py`

```
usage: statuspic.py [-h] [-d DATE] [-p PATH] [--cons CONS]
```

-h, --help

show this help message and exit

-d <date>

Specific date to draw pictures 绘制图片的具体日期

-p <path>

Path to save pictures 保存图片的文件夹

--cons <cons>

gallery defined constants such as file path gallery 文件定义的文件路径等常数

19.1 Functions in plancompare.py

main()

Compare observation plan & actual observation time 观测计划检查，对比观测计划和实际观测时间

19.2 Commandline interface in plancompare.py

19.2.1 plancompare.py

```
usage: plancompare.py [-h] [--cons CONS]
```

-h, --help

show this help message and exit

--cons <cons>

gallery defined constants such as file path gallery 文件定义的文件路径等常数

20.1 Functions in runtime.py

main()

Show running status in a long time span 运行状况检查，查看长时间运行状态

20.2 Commandline interface in runtime.py

20.2.1 runtime.py

```
usage: runtime.py [-h] [--cons CONS]
```

-h, --help

show this help message and exit

--cons <cons>

gallery defined constants such as file path gallery 文件定义的文件路径等常数

CHAPTER 21

更新日志 Change Log

21.1 v2.1.0

CHAPTER 22

索引| Indices and tables

- `genindex`
- `search`

0

01, [35](#)

02, [36](#)

c

cali, [55](#)

g

gallery, [35](#)

gridfun, [39](#)

gridpath, [53](#)

m

make, [37](#)

p

plancompare, [61](#)

r

runtime, [63](#)

s

statuspic, [59](#)

符号

`__init__()` (*Adc_mapping* 方法), 40
`__init__()` (*Saa_mapping* 方法), 39
`--cons <cons>`
 `make.py` command line option, 37
 `plancompare.py` command line option, 61
 `runtime.py` command line option, 63
 `statuspic.py` command line option, 60
`--delete`
 `make.py` command line option, 37
`--help`
 `cali.py` command line option, 55
 `make.py` command line option, 37
 `plancompare.py` command line option, 61
 `runtime.py` command line option, 63
 `statuspic.py` command line option, 60
`--mode {ecccoef,tempbias}`
 `cali.py` command line option, 55
`-d <date>`
 `statuspic.py` command line option, 60
`-h`
 `cali.py` command line option, 55
 `make.py` command line option, 37
 `plancompare.py` command line option, 61
 `runtime.py` command line option, 63
 `statuspic.py` command line option, 60
`-p <path>`
 `statuspic.py` command line option, 60

数值

01
 模块, 35
 02
 模块, 36

A

`Adc_mapping` (*gridfun* 中的类), 40
`adcmap()` (*Adc_mapping* 方法), 40
`atti_source_H()` (在 *gridpath* 模块中), 53

B

`bin_source_H()` (在 *gridpath* 模块中), 54

C

`cali`
 模块, 55
`cali.py` command line option
 `--help`, 55
 `--mode {ecccoef,tempbias}`, 55
 `-h`, 55
`check_crc16()` (在 *gridfun* 模块中), 41
`chunk()` (在 *statuspic* 模块中), 59

D

`dat_source_H()` (在 *gridpath* 模块中), 53
`date_to_days()` (在 *gridfun* 模块中), 43
`datestrcolumn()` (在 *01* 模块中), 35
`datestrcolumn()` (在 *02* 模块中), 36
`days_to_date()` (在 *gridfun* 模块中), 43
`del_dir_tree()` (在 *gridfun* 模块中), 44
`del_hidden()` (在 *gridfun* 模块中), 43
`delete()` (在 *gridfun* 模块中), 44
`detectCode()` (在 *gridfun* 模块中), 44
`divide_u()` (在 *gridfun* 模块中), 41
`draw_single_day()` (在 *statuspic* 模块中), 59
`DT_LAUNCH()` (在 *gridfun* 模块中), 39

E

`earth_source_H()` (在 *gridpath* 模块中), 54
`ec_coef()` (在 *02* 模块中), 36
`ec_data()` (在 *02* 模块中), 36
`ecmap()` (*Adc_mapping* 方法), 40
`energy_recon()` (在 *gridfun* 模块中), 47
`extract_ear()` (在 *gridfun* 模块中), 49

`extract_sci_events()` (在 *gridfun* 模块中), 45
`extract_sun()` (在 *gridfun* 模块中), 48
`extract_sun_ear()` (在 *gridfun* 模块中), 50
`extract_tel_events()` (在 *gridfun* 模块中), 45

F

`find_base()` (在 *gridfun* 模块中), 44
`find_best_part()` (在 *gridfun* 模块中), 42
`find_pattern_pos()` (在 *gridfun* 模块中), 41
`fits0_save()` (在 *gridfun* 模块中), 45
`fits1_recon()` (在 *gridfun* 模块中), 46
`fits2_make()` (在 *gridfun* 模块中), 49
`fits3p()` (在 *gridpath* 模块中), 54
`fits4_make()` (在 *gridfun* 模块中), 48
`fits6_make()` (在 *gridfun* 模块中), 49
`fitsprefix()` (在 *01* 模块中), 35
`fitsprefix()` (在 *02* 模块中), 36
`Freq()` (在 *01* 模块中), 35
`Freq()` (在 *02* 模块中), 36
`fun_saa()` (*Saa_mapping* 方法), 39

G

`gallery`
 模块, 35
`getHTMLText()` (在 *gridfun* 模块中), 50
`gridfun`
 模块, 39
`gridpath`
 模块, 53

I

`intersection_angle()` (在 *gridfun* 模块中), 44

L

`leaki_path()` (在 *01* 模块中), 35
`log_atti_path()` (在 *gridpath* 模块中), 53
`log_dat_path()` (在 *gridpath* 模块中), 53
`log_eart_path()` (在 *gridpath* 模块中), 54
`log_orb_H()` (在 *gridpath* 模块中), 54
`log_path()` (在 *gridpath* 模块中), 53
`log_sci_path()` (在 *gridpath* 模块中), 53
`log_sun_path()` (在 *gridpath* 模块中), 54
`log_tel_path()` (在 *gridpath* 模块中), 53
`log_tte_path()` (在 *gridpath* 模块中), 54
`log_txt_path()` (在 *gridpath* 模块中), 53

M

`main()` (在 *cali* 模块中), 55
`main()` (在 *make* 模块中), 37
`main()` (在 *plancompare* 模块中), 61
`main()` (在 *runtime* 模块中), 63
`main()` (在 *statuspic* 模块中), 59
`make`

模块, 37

`make.py` command line option
`--cons <cons>`, 37
`--delete`, 37
`--help`, 37
`-h`, 37

`make01electronics()` (在 *gridfun* 模块中), 46
`make02electronics()` (在 *gridfun* 模块中), 46
`makefits01()` (在 *gridfun* 模块中), 45
`makefits2()` (在 *gridfun* 模块中), 48
`makefits3()` (在 *gridfun* 模块中), 45
`makefits4()` (在 *gridfun* 模块中), 47
`makefits5()` (在 *gridfun* 模块中), 47
`makefits6()` (在 *gridfun* 模块中), 49
`man_hough()` (在 *gridfun* 模块中), 42
`my_cmap()` (在 *gridfun* 模块中), 44

N

`N()` (在 *gridfun* 模块中), 39
`name()` (在 *01* 模块中), 35
`name()` (在 *02* 模块中), 36
`newec_coef()` (在 *02* 模块中), 36
`newtempbias()` (在 *02* 模块中), 36
`normal_array()` (在 *gridfun* 模块中), 43

O

`O_A()` (在 *01* 模块中), 35
`O_A()` (在 *02* 模块中), 36
`omi2pos()` (在 *gridfun* 模块中), 42
`orbmon_source_H()` (在 *gridpath* 模块中), 54

P

`pic_source_H()` (在 *gridpath* 模块中), 54
`picsprefix()` (在 *01* 模块中), 35
`picsprefix()` (在 *02* 模块中), 36
`plan_excel()` (在 *gridpath* 模块中), 54
`plan_source_H()` (在 *gridpath* 模块中), 54
`plancompare`
 模块, 61
`plancompare.py` command line option
`--cons <cons>`, 61
`--help`, 61
`-h`, 61
`pol2d()` (在 *gridfun* 模块中), 41
`pos2omi()` (在 *gridfun* 模块中), 42

Q

`quatstrcolumn()` (在 *01* 模块中), 35
`quatstrcolumn()` (在 *02* 模块中), 36

R

`radec_to_xyz()` (在 *gridfun* 模块中), 44
`read_raw()` (在 *gridfun* 模块中), 41

read_sun_ear() (在 *gridfun* 模块中), 50
 readcali() (*Adc_mapping* 方法), 40
 readSAA() (*Saa_mapping* 方法), 39
 rm_frag() (在 *gridfun* 模块中), 42
 runtime
 模块, 63
 runtime.py command line option
 --cons <cons>, 63
 --help, 63
 -h, 63

02, 36
 cali, 55
 gallery, 35
 gridfun, 39
 gridpath, 53
 make, 37
 plancompare, 61
 runtime, 63
 statuspic, 59

S

Saa_mapping(*gridfun* 中的类), 39
 saaAE8_path() (在 *gridpath* 模块中), 54
 saacoord_path() (在 *gridpath* 模块中), 54
 sci_source_H() (在 *gridpath* 模块中), 53
 snip_baseline() (在 *gridfun* 模块中), 50
 sourprefix() (在 01 模块中), 35
 sourprefix() (在 02 模块中), 36
 statuspic
 模块, 59
 statuspic.py command line option
 --cons <cons>, 60
 --help, 60
 -d <date>, 60
 -h, 60
 -p <path>, 60
 sun_source_H() (在 *gridpath* 模块中), 53

T

tel_source_H() (在 *gridpath* 模块中), 53
 tempbias() (在 02 模块中), 36
 tEnergy_source_H() (在 *gridpath* 模块中), 54
 txt_source_H() (在 *gridpath* 模块中), 53
 TZ.UTC_8() (在 *gridfun* 模块中), 39

U

UTC_LAUNCH() (在 *gridfun* 模块中), 39
 utc_to_days() (在 *gridfun* 模块中), 42
 utc_to_str() (在 *gridfun* 模块中), 43

V

Vbr() (在 01 模块中), 35
 vdown_path() (在 01 模块中), 35
 vop_normal() (在 *gridfun* 模块中), 50
 Vout_dic() (在 01 模块中), 35

X

xyz_to_radec() (在 *gridfun* 模块中), 43



模块
 01, 35