UDAS egg manual

for UDAS egg 1.02

Dec 20, 2018

IUGONET project team

What is UDAS egg ?

UDAS egg is a template program for IDL/SPEDAS

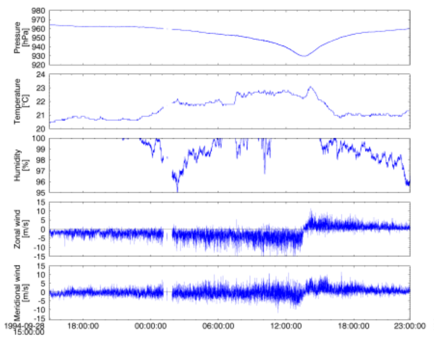
to read/analyze scientific data that are not supported by the original SPEDAS.

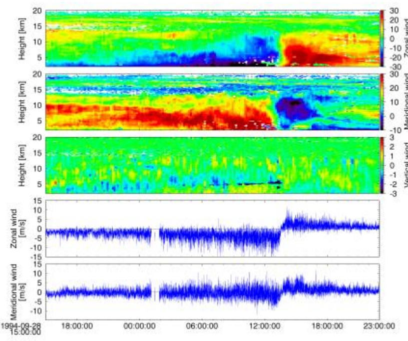
You can easily visualize/analyze your data by

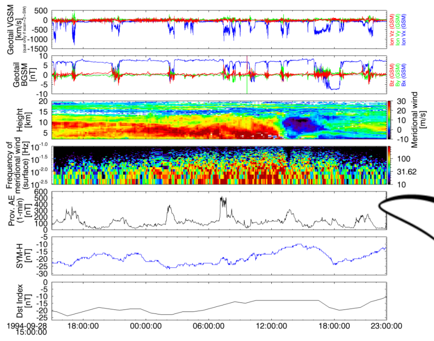
(1) modifying some parts marked in the code or

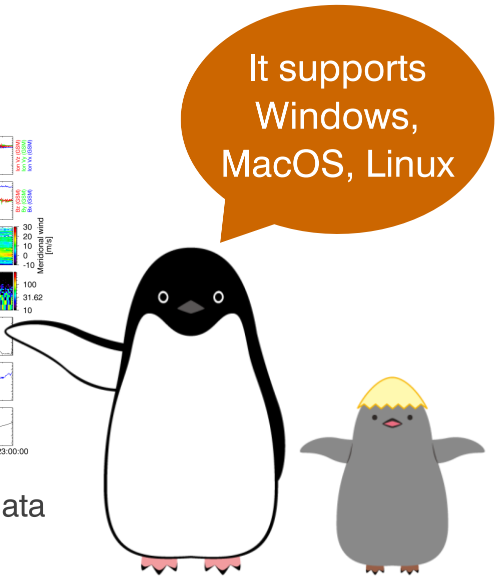
(2) setting how to read the data files

according to this manual.


Line plot


Color contour


Combination with other data



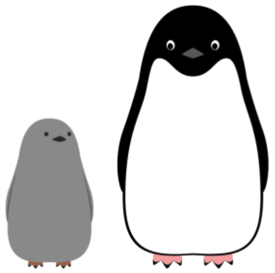
IDL (Interactive Data Language) is a programming language used for data visualization and analysis.
SPEDAS (Space Physics Environment Data Analysis Software) is a set of IDL library for visualizing and analyzing various scientific data obtained by satellite- (e.g., THEMIS, MMS, GOES, and ERG) and ground-based observations.
The details of these software are described at the websites as follows;
IDL: https://www.harrisgeospatial.com/ SPEDAS: http://spedas.org/

Comparison with SPEDAS (including UDAS)

	SPEDAS	UDAS EGG CUI	GUI
How to create load procedures	**Difficult** Need to create each procedure individually	**Easy** Have only to modify about 10 lines in the template	**Unnecessary** Generic routine has been implemented
Location of data files that you want to read/analyze	**On-line server** Download via the internet	**Directly from PC** Download via the internet or select files on your PC	**Directly from PC** Select files on your PC *remote files will be available in future

This is suitable for users **who need help creating analysis routine for your data,** and users **who want to spend more time for your research.**

You can download UDAS egg and use it with SPEDAS.

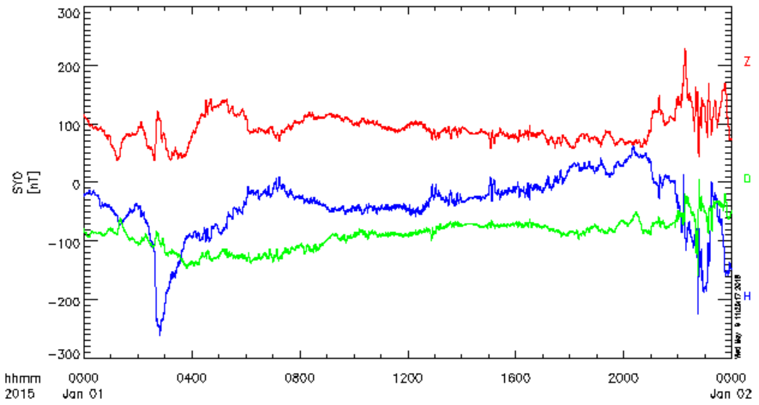


IUGONET

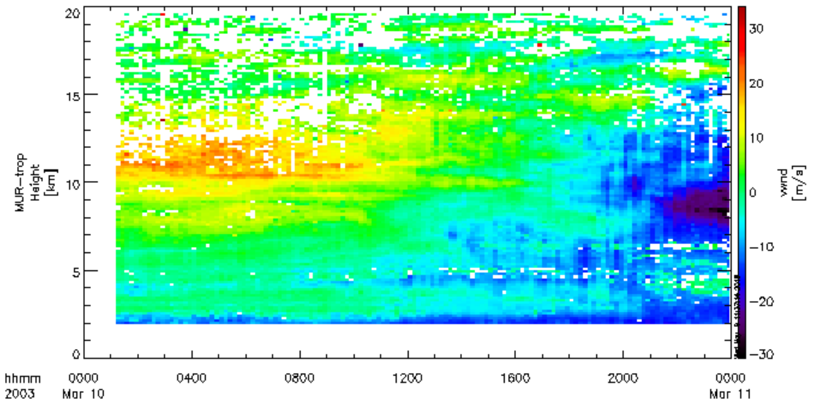
Readable file types and examples of visualization

1. CDF (Common Data Format)

CDF files



Line plot



Color contour

In case of CDF, SPEDAS reads the information of vertical axis (v data) and select line plot or color contour plot automatically.

It is unnecessary to select manually which of line plot or color contour plot before visualization.

Readable file types and examples of visualization

2. ASCII Format

0. Simple time-series data

Date & Time	other data 1	2	3	...
1994/09/28 15:00:07,	964.0,	20.4,	100.0,	...
1994/09/28 15:00:12,	964.0,	20.4,	100.0,	...
1994/09/28 15:00:17,	964.0,	20.4,	100.0,	...
1994/09/28 15:00:22,	964.0,	20.4,	100.0,	...
1994/09/28 15:00:27,	964.0,	20.4,	100.0,	...
1994/09/28 15:00:32,	964.0,	20.4,	100.0,	...
1994/09/28 15:00:37,	964.0,	20.4,	100.0,	...
1994/09/28 15:00:42,	964.0,	20.4,	100.0,	...
1994/09/28 15:00:47,	964.0,	20.4,	100.0,	...
1994/09/28 15:00:52,	964.0,	20.4,	100.0,	...
...				

The kind of data arranges horizontally, and time changes vertically.

1. Column data including v-data (y-axis information)

Date & Time	other data v	y1	y2 ...
1994/09/28 15:05:00,	1.998,	-12.0,	…
1994/09/28 15:05:00,	2.145,	-11.3,	…
1994/09/28 15:05:00,	2.293,	-9.7,	…
1994/09/28 15:05:00,	2.441,	-9.1,	…
1994/09/28 15:05:00,	2.589,	-8.0,	…
1994/09/28 15:05:00,	2.736,	-10.3,	…
1994/09/28 15:05:00,	2.884,	-9.4,	…
1994/09/28 15:05:00,	3.032,	-8.5,	…
1994/09/28 15:05:00,	3.179,	-6.9,	…
1994/09/28 15:05:00,	3.327,	-7.1,	…
…			

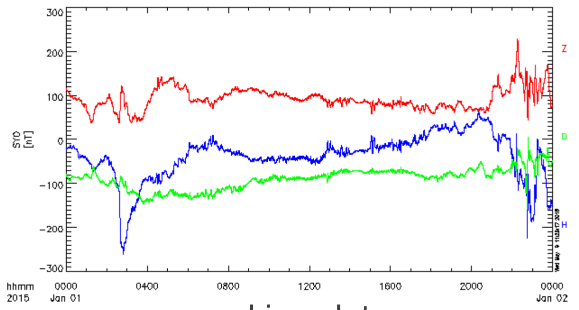
V-data corresponds to y-axis information for color contour plot. For example, altitude for atmospheric data, frequency for spectral data, range for radar data.

2. Raw data without v-data (y-axis information)

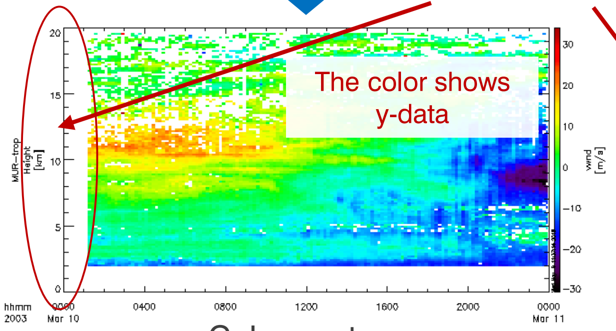
Date & Time	Other data y	y	y	...
1994/09/29 00:05,	-12.0,	-11.3,	-9.7,	-9.1, ...
1994/09/29 00:15,	-12.2,	-12.1,	-11.7,	-11.5, ...
1994/09/29 00:25,	-12.8,	-11.5,	-11.1,	-10.9, ...
1994/09/29 00:35,	-10.5,	-11.1,	-10.6,	-10.8, ...
1994/09/29 00:45,	-12.9,	-12.7,	-10.9,	-12.4, ...
1994/09/29 00:55,	-13.6,	-13.9,	-12.6,	-12.2, ...
1994/09/29 01:05,	-11.2,	-11.1,	-11.1,	-10.4, ...
1994/09/29 01:15,	-10.5,	-10.2,	-11.4,	-10.8, ...
1994/09/29 01:25,	-11.5,	-11.1,	-11.6,	-10.9, ...
1994/09/29 01:35,	-12.6,	-12.5,	-11.8,	-12.0, ...
…				

Y-data arranges horizontally. V-data is required as the input argument.

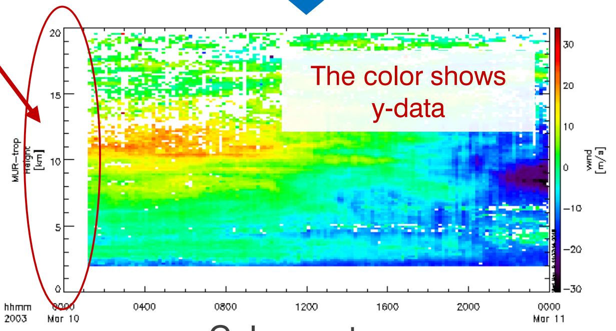
V-data is used for vertical axis.



Line plot



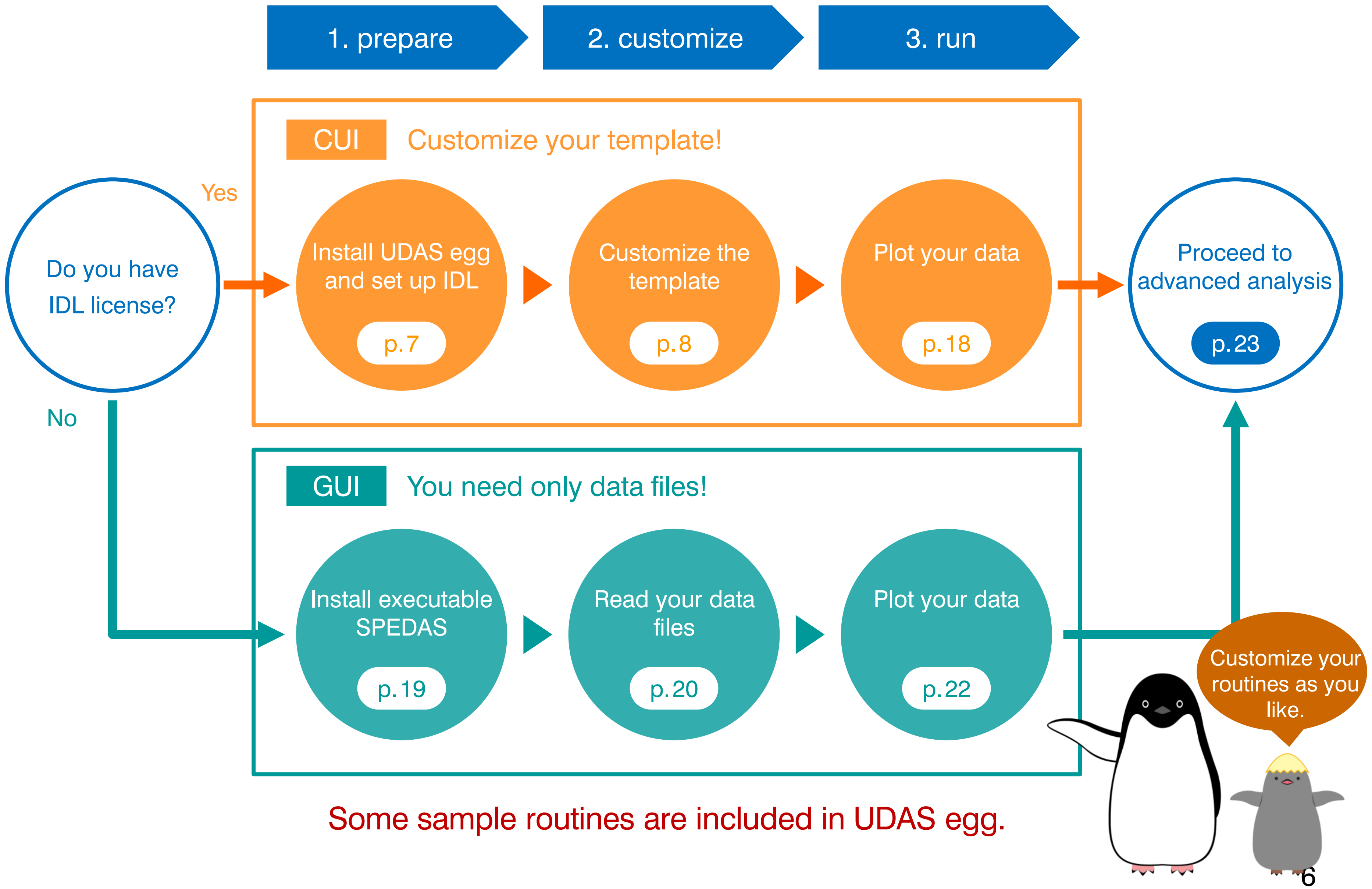
Color contour



Color contour

If you want to load binary-format or other original-format files, please convert them to ASCII format as shown above.
It is easy to read various kinds of data, such as scientific data, social data, and environmental data.

Let’s start



Some sample routines are included in UDAS egg.

CUI 1. Install UDAS egg and set up IDL

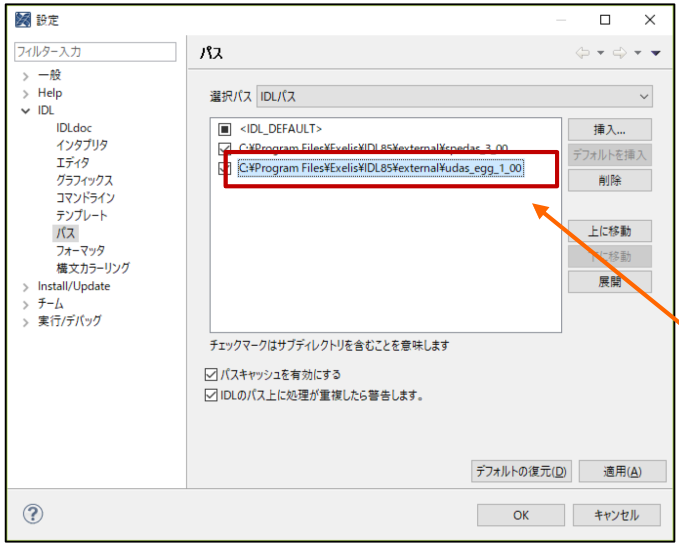
1. Download SPEDAS 3.1 source code and set up IDL

   (In case that SPEDAS has not been installed yet)

   How to download and install SPEDAS 3.1 : http://themis.ssl.berkeley.edu/software.shtml

2. Download UDAS egg and set up IDL

   UDAS egg ダウンロード: http://www.iugonet.org/product/analysis.jsp



選択パス IDLパス

- ■ <IDL_DEFAULT>
- ☑ C:¥Program Files¥Exelis¥IDL85¥external¥spedas_3_00
- ☑ C:¥Program Files¥Exelis¥IDL85¥external¥udas_egg_1_00

チェックマークはサブディレクトリを含むことを意味します

- ☑ パスキャッシュを有効にする
- ☑ IDLのパス上に処理が重複したら警告します。

1. Download UDAS egg (.zip).
2. Copy the unzipped file to a directory you like.

If you have no idea about the directory, please copy it to C:¥Program Files¥Exelis¥IDL8x¥external¥

3. Run IDL and select [settings][IDL/path] from the [window] tab, and set IDL path to both SPEDAS and UDAS egg.

3. Run the following command on the IDL command line to check the success of the installation.

```
IDL> .r loadproc_template
% Compiled module: LOADPROC_TEMPLATE.
IDL>
```

Installation succeeded.

CUI

2. Customize template

1. Copy and rename the template file.

   [Original file] loadproc_template.pro

   As an example, it is assumed that the template file was renamed to ”loadproc_mag.pro”.

2. Edit the new file.

*: required

```
pro loadproc_mag, site=site, datatype=datatype, $
      trange=trange, verbose=verbose, downloadonly=downloadonly, $
      no_download=no_download
```

*** pro loadxxxxx: Routine’s name**
Set the same name as the file name. This routine’s name corresponds to the command name.

CUI

2. Customize template

2. Edit the new file. (Step.1: Settings of general parameters)

* : required

```
;********************************;
;***** Step1: Set parameters *****;
;********************************;
file_format='cdf'   ; Choose  'cdf' or 'ascii'
url='http://www.iugonet.org/data/SITE/DATATYPE/YYYY/mag_SITE_DATATYPE_YYYYMMDD_v??.cdf'
acknowledgement = 'You can write the data use policy here. ' + $
   'This description is displayed when you use this load procedure. '
site_list='sta1 sta2 sta3'
datatype_list='1sec 1min 1hr'
```

*** file_format**

Choose ‘cdf’ or ‘asci’.

*** url: URL for data files that you want to read**

Set the URL for data files that you want to read. The following strings are automatically converted to specific values.

SITE: Site name (corresponds to the elements defined in site_list)

DATATYPE: Data type (corresponds to the elements defined in datatype_list)

Strings used for data and time

Year (4-digit)	Year (2-digit)	Month (01-12)	Day (01-31)	Hour (2-digit)
YYYY	yy	MM	DD	hh

CUI

2. Customize template

2. Edit the new file. (Step.1: Settings of general parameters)

```
;********************************;
;***** Step1: Set parameters *****;
;********************************;
file_format='cdf'   ; Choose  'cdf' or 'ascii'
url='http://www.iugonet.org/data/SITE/DATATYPE/YYYY/mag_SITE_DATATYPE_YYYYMMDD_v??.cdf'
acknowledgement = 'You can write the data use policy here. ' + $
   'This description is displayed when you use this load procedure. '
site_list='sta1 sta2 sta3'
datatype_list='1sec 1min 1hr'
```

【Example】

If you set the URL to

http://www.iugonet.org/data/SITE/DATATYPE/YYYY/mag_SITE_DATATYPE_YYYYMMDD_v??.cdf

the URL is automatically converted as follows:

http://www.iugonet.org/data/ath/1sec/2018/mag_ath_1sec_20181013_v01.cdf

http://www.iugonet.org/data/ath/1sec/2018/mag_ath_1sec_20181014_v01.cdf

……

These date and time are fixed by the “timespan” command, which is given before running this routine.

'??' means the wild card in SPEDAS.

CUI

2. Customize template

2. Edit the new file. (Step.1: Settings of general parameters)

```
;********************************;
;***** Step1: Set parameters *****;
;********************************;
file_format='cdf'  ; Choose 'cdf' or 'ascii'
url='http://www.iugonet.org/data/SITE/DATATYPE/YYYY/mag_SITE_DATATYPE_YYYYMMDD_v??.cdf'
acknowledgement = 'You can write the data use policy here. ' + $
   'This description is displayed when you use this load procedure. '
site_list='sta1 sta2 sta3'
datatype_list='1sec 1min 1hr'
```

acknowledgement: Data usage policy

Describe the data usage policy here. The description will be displayed on the window when you run this routine.

site_list: site list

Specify a list of observation sites, separated by spaces, for example, ‘stn1 stn2 stn3’.

If there is no site, you can set a dummy character string here.

This corresponds to site names, which are available for the keyword ‘site’ in this command.

datatype_list: data type list

Specify a list of data type, for example, sampling interval, wave length, observation mode, and file version.

If there is no data type, you can set a dummy character string here.

This corresponds to data types, which are available for the keyword ‘datatype’ in this command.

IUGONET CUI

2. Customize template

2. Edit the new file. (Step.2: Settings for reading files)

In case of CDF

```
;*********************************************;
;**** Step2: Load data into tplot variables ****;
;*********************************************;
;----- For CDF format files -----;
 prefix='test_'+site[isite]+'_'+datatype[idt]+'_'
;          suffix='_'+site[isite]+'_'+datatype[idt]
```

prefix: prefix for tplot variables

Specify a prefix that is added to loaded tplot variables. You can choose any prefix which is easy for you to understand. In the example above, 'test_stn_1min_' is prefixed to tplot variables.

suffix: suffix for tplot variables

Specify a suffix that is added to loaded tplot variables. (In most cases, this keyword is not set.)

IUGONET

CUI

2. Customize template

2. Edit the new file. (Step.2: Settings for reading files)

*: required

In case of ASCII

```
;*********************************************;
;**** Step2: Load data into tplot variables ****;
;*********************************************;

;----- For ASCII format files -----;

format_type=0     ; 0:xy, 1:xyv_1, 2:xyv_2

tformat='YYYY-MM-DD hh:mm:ss.fff'

tvar_column=[1, 2, 3, 4]

tvarnames='test_'+site[isite]+'_'+datatype[idt]

delimiter=','

data_start=0
```

*** format_type: type of ASCII format**

Specify the number (0-2), which corresponds to the type of ASCII formats shown on page 5.

*** tformat: time format**

Specify the format of date and time described in the data file.

*** tvar_column: column number**

Specify the column number that identifies which columns are loaded. It starts at 0 except for the columns of date and time.

*** tvarnames: tplot variable name**

Specify output tplot variable name. You can choose any name which is easy for you to understand.

CUI

2. Customize template

2. Edit the new file. (Step.2: Settings for reading files)

*: required

In case of ASCII

```
;*********************************************;
;**** Step2: Load data into tplot variables ****;
;*********************************************;

;----- For ASCII format files -----;

format_type=0      ; 0:xy, 1:xyv_1, 2:xyv_2

tformat='YYYY-MM-DD hh:mm:ss.fff'

tvar_column=[1, 2, 3, 4]

tvarnames='test_'+site[isite]+'_'+datatype[idt]

delimiter=','

data_start=0
```

* **delimiter: delimiter**

Specify the delimiter to split data.

data_start: row number for data start

Specify the number of header lines you want to skip. If not specified, it is set to 0.

IUGONET CUI

2. Customize template

2. Edit the new file. (Step.2: Settings for reading files)

In case of ASCII

```
;*********************************************;
;**** Step2: Load data into tplot variables ****;
;*********************************************;

;----- For ASCII format files -----;

comment_symbol=''
;          v_column=0
;          vvec=[60., 70., 80., 90., 100., 110., 120.]
;          time_column=[0, 0, 0, 1, 1, 1]
;          input_time=[2017, 1, 1, 0, 0, 0]
```

comment_symbol: comment out symbol

Specify the string used to delineate comments. If not specified, all rows will be read.

v_column: column number for v-data

This keyword is used when format_type=1. Specify the column number that identifies which column is used for v-data, except for the columns of date and time.

vvec: a vector of v-data

This keyword is used when format_type=2. Set a vector for v-data here (this is used for y-axis).

IUGONET CUI

2. Customize template

2. Edit the new file. (Step.2: Settings for reading files)

In case of ASCII

```
;*********************************************;
;**** Step2: Load data into tplot variables ****;
;*********************************************;

;----- For ASCII format files -----;

comment_symbol=''
;          v_column=0
;          vvec=[60., 70., 80., 90., 100., 110., 120.]
;          time_column=[0, 0, 0, 1, 1, 1]
;          input_time=[2017, 1, 1, 0, 0, 0]
```

time_column: flag for date and time (option) / input_time: input date and time (option)

If data files do not have all the information of date and time [year, month, day, hour, minute, second], these keywords supplement insufficient information.

As for "time_column", set the elements that are included (not included) in the data file to 1 (0).

As for "input_time", set the elements where "time_column" = 0 to concrete values for date and time.

For example, if data file has only the information of hh, mm, ss, each keyword is set as follows:

```
time_column = [0, 0, 0, 1, 1, 1]
input_time = [2018, 10, 13, 0, 0, 0]
```

IUGONET

CUI

2. Customize template

2. Edit the new file. (Step.3: Other customization)

```
;*********************************************;
;***** Step3: Options for tplot variables *****;
;*********************************************;
;----- Missing data --> NaN -----;
;       tclip, tvarnames, -1e+5, 1e+5, /overwrite
;----- Ylim -----;
;       ylim, tvarnames, -1000, 1000
;----- Labels -----;
;       options, /def, tvarnames, labels=['ch1','ch2',' ch3'], $
;           ytitle = 'Tatejiku', $
;           ysubtitle = '[Unit]', labflag=1, colors=[2,4,6]
```

tclip: range for replacing with NaN
If values of data are over this range, they are replaced with NAN. NaN is not plotted in IDL.

ylim: y-axis range
Specify the maximum and minimum values of y-axis. The y-axis range can also be set by the “ylim” command on the command line after running this routine.

options

labels	: specify labels displayed in the figure. A new line character is “!C”.
ytitle	: specifies a title for y-axis.
ysubtitle	: specifies a subtitle for y-axis.
labflag	: flag for labels. 0: non-display, 1: display
colors	: color of lines. 0: black, 1: magenta, 2: blue, 3: cyan, 4: green, 5: yellow, 6: red

CUI

3. Plot your data

1. Start IDL, and run the following commands.

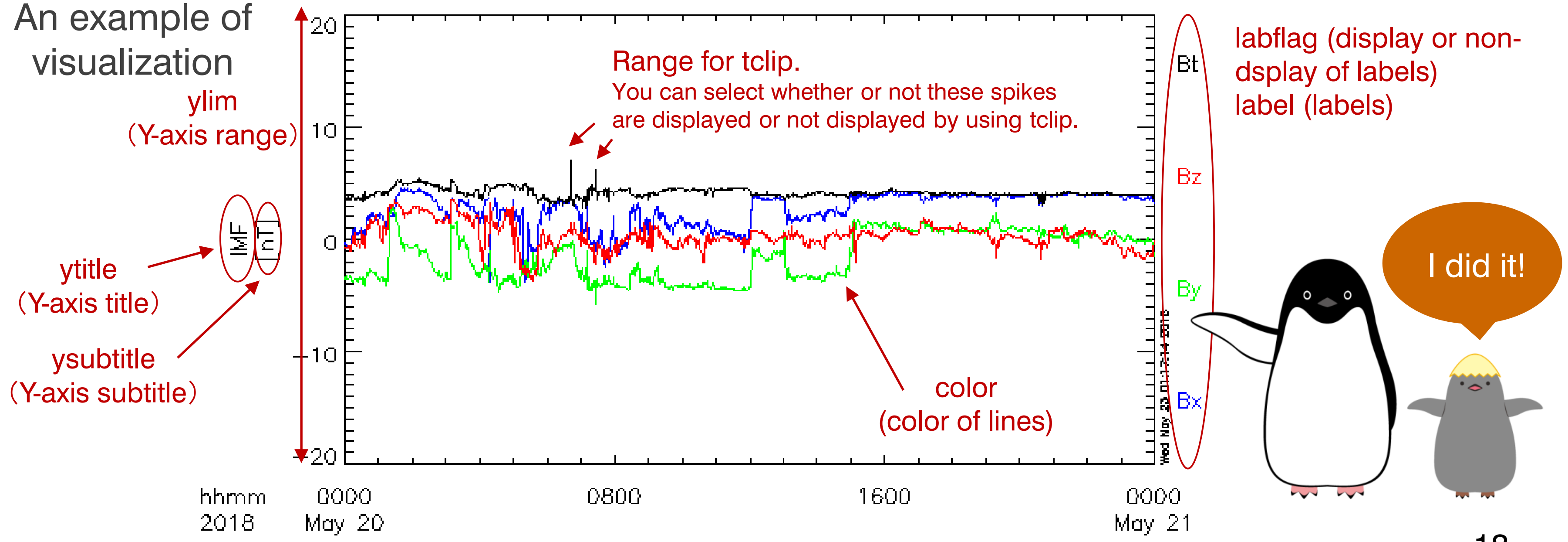
```
IDL> thm_init
THEMIS> timespan, '2018-05-20'
THEMIS> .r loadproc_mag
THEMIS> loadproc_mag, site= 'stn', datatype='1min'
THEMIS> tplot_names
  1 mag_stn_1min
THEMIS> tplot, 'mag_stn_1min'
```

Compile the created routine.

Run the created routine.
If you defined site_list and datatype_list, you can set the keywords 'site' and 'datatype' here.

The tplot variables defined in Step 3 are loaded.

An example of visualization



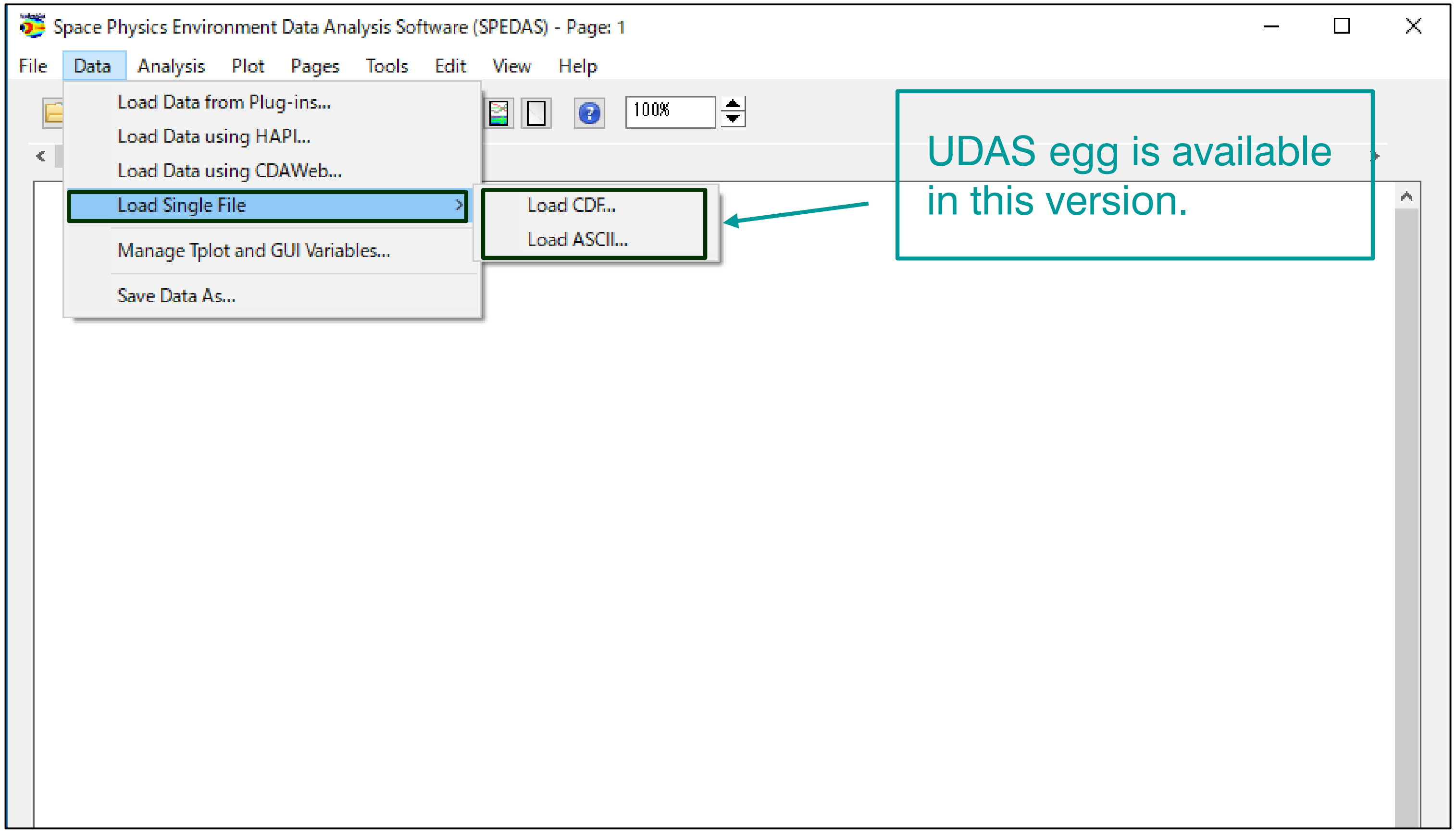
GUI

1. Install executable SPEDAS

1. Download executable SPEDAS (GUI) via internet.

   You can download it from http://themis.ssl.berkeley.edu/software.shtml

2. If there is [Load Single File] menu in the [Data] tab on the SPEDAS window, you can use UDAS egg.


Space Physics Environment Data Analysis Software (SPEDAS) - Page: 1
File
Data
Analysis
Plot
Pages
Tools
Edit
View
Help
Load Data from Plug-ins...
Load Data using HAPI...
Load Data using CDAWeb...
Load Single File
Manage Tplot and GUI Variables...
Save Data As...
Load CDF...
Load ASCII...
100%

UDAS egg is available in this version.

IUGONET GUI

2. Read data file

1. Select [Load ASCII…] in the [Load Single file] menu from [Data] tab.
2. Set parameters for reading your data file.

In case of CDF, select the CDF file you want to read and click OK afterward.

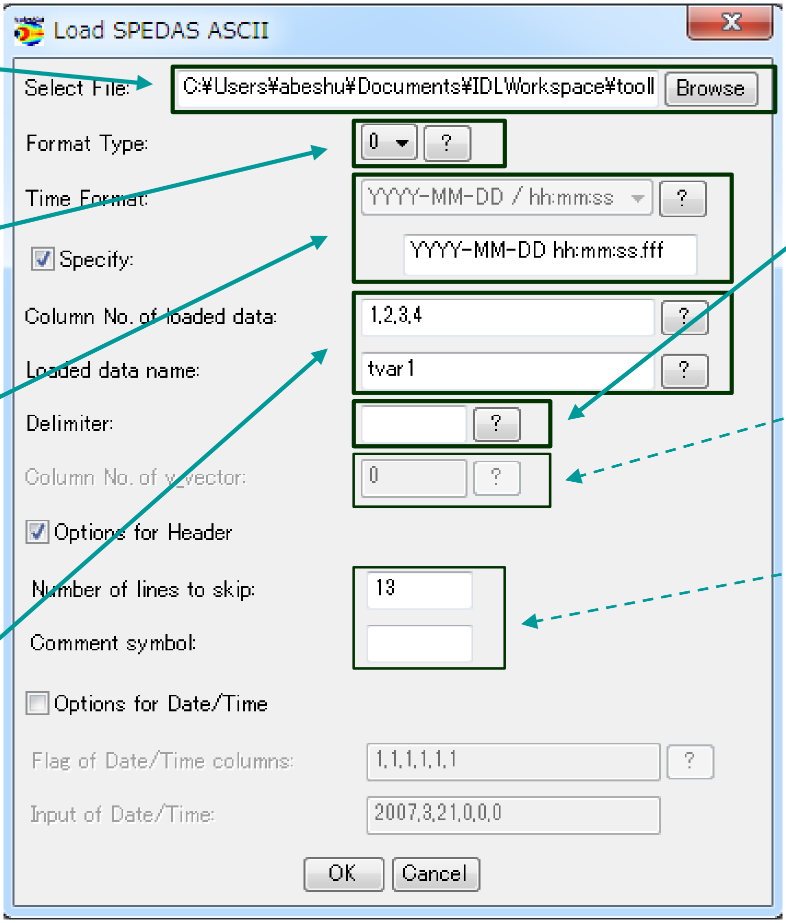
In case of ASCII, set parameters as shown below.

Specify a data file you want to read.

Specify format type (0-1) for ASCII file. (See page 5.)

Select a format string for date and time. If there is no format in the dropdown list, specify the string below.

Input numbers of column to be read. Set output tplot variable names for loaded data.



Load SPEDAS ASCII

Select File: C:¥Users¥abeshu¥Documents¥IDLWorkspace¥tooll [Browse]

Format Type: 0

Time Format: YYYY-MM-DD / hh:mm:ss

[✓] Specify: YYYY-MM-DD hh:mm:ss.fff

Column No. of loaded data: 1,2,3,4

Loaded data name: tvar1

Delimiter:

Column No. of v_vector: 0

[✓] Options for Header

Number of lines to skip: 13

Comment symbol:

[ ] Options for Date/Time

Flag of Date/Time columns: 1,1,1,1,1,1

Input of Date/Time: 2007,3,21,0,0,0

[OK] [Cancel]

Input delimiter to split data.

Specify column number of V-data (which starts at 0 except for date and time). (This is used only for Format Type =1)

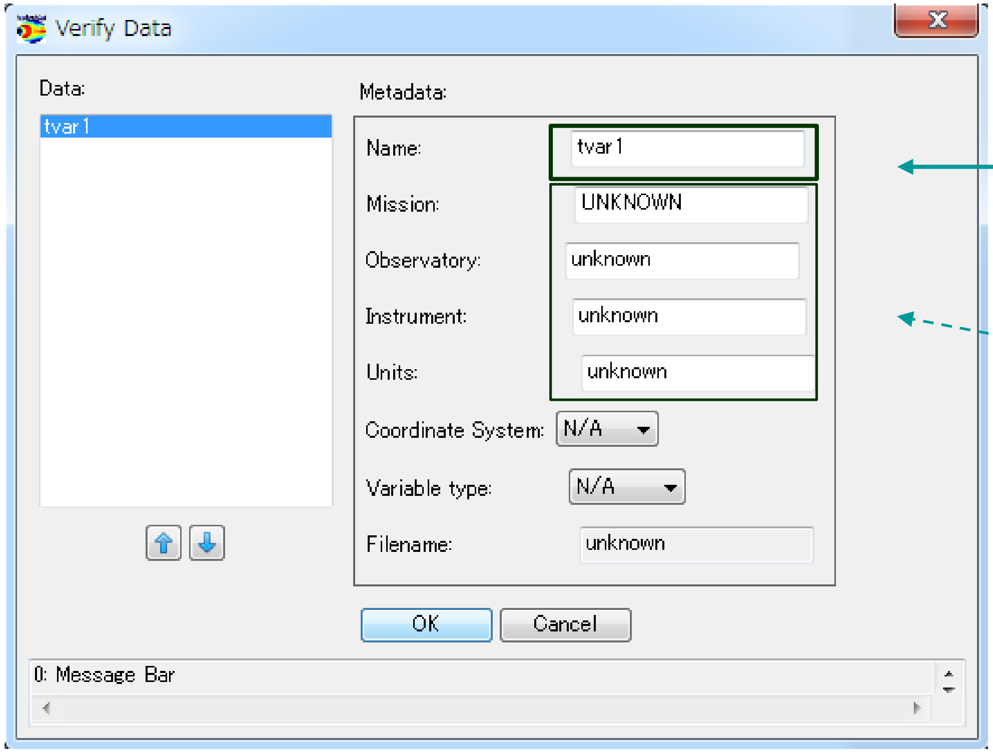
Specify number of header lines to skip and character strings to delineate comments.

Click OK button.

IUGONET GUI

2. Read data file

3. Set parameters for tree structure of tplot variable.



Name: tvar1
Mission: UNKNOWN
Observatory: unknown
Instrument: unknown
Units: unknown
Coordinate System: N/A
Variable type: N/A
Filename: unknown

Confirm the specified tplot variable name.

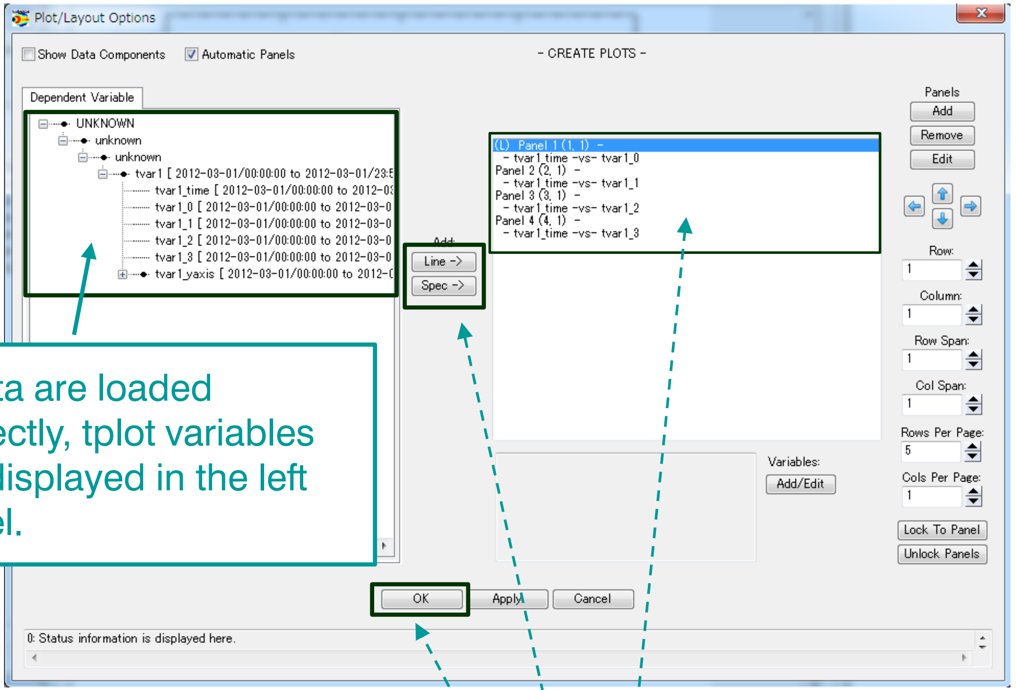
If necessary, modify names for tree structure of created tplot variable.
When you load other data together, the tree structure may help you to distinguish these data.

Click OK button.

IUGONET GUI

3. Plot your data

1. Select [Plot/Layout Options] in the [Plot] tab.



```
UNKNOWN
  unknown
    unknown
      tvar1 [ 2012-03-01/00:00:00 to 2012-03-01/23:5
        tvar1_time [ 2012-03-01/00:00:00 to 2012-03
        tvar1_0 [ 2012-03-01/00:00:00 to 2012-03-0
        tvar1_1 [ 2012-03-01/00:00:00 to 2012-03-0
        tvar1_2 [ 2012-03-01/00:00:00 to 2012-03-0
        tvar1_3 [ 2012-03-01/00:00:00 to 2012-03-0
        tvar1_yaxis [ 2012-03-01/00:00:00 to 2012-(
```

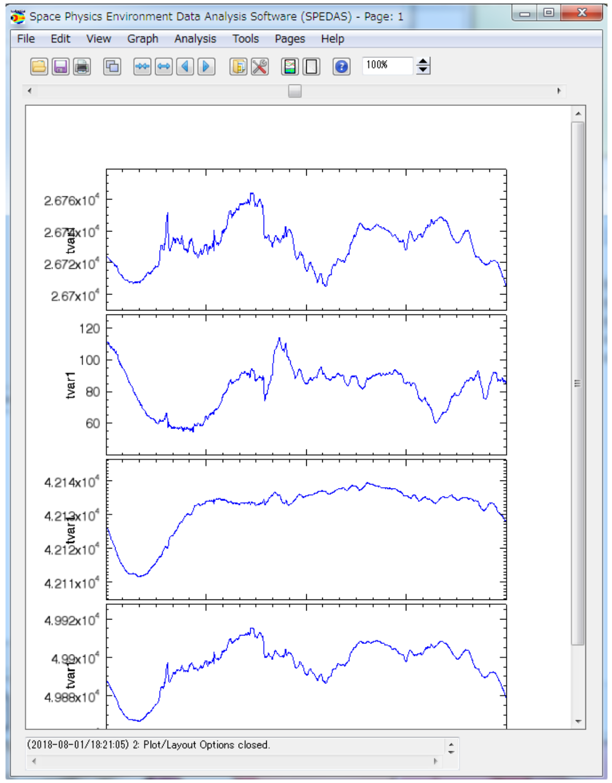
```
(L) Panel 1 (1, 1) -
  - tvar1_time -vs- tvar1_0
Panel 2 (2, 1) -
  - tvar1_time -vs- tvar1_1
Panel 3 (3, 1) -
  - tvar1_time -vs- tvar1_2
Panel 4 (4, 1) -
  - tvar1_time -vs- tvar1_3
```

Row: 1 · Column: 1 · Row Span: 1 · Col Span: 1 · Rows Per Page: 5 · Cols Per Page: 1

```
0: Status information is displayed here.
```

If data are loaded correctly, tplot variables are displayed in the left panel.

Select tplot variables you want to visualize and click OK.



```
(2018-08-01/18:21:05) 2: Plot/Layout Options closed.
```

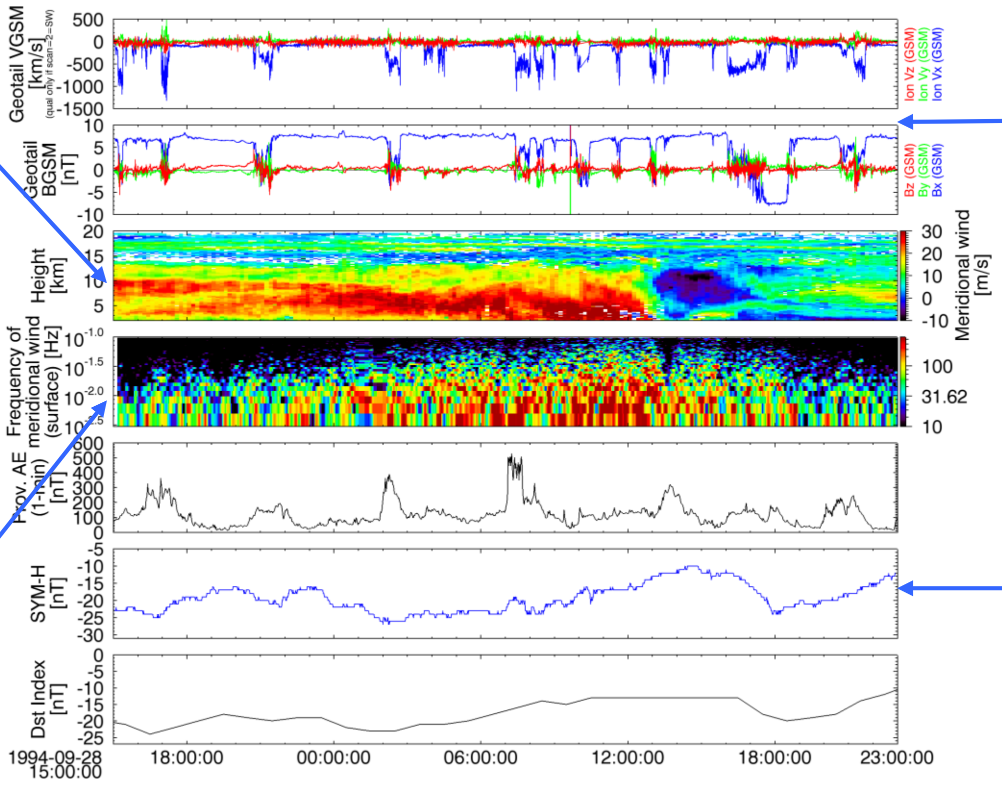
Proceed to more advanced analysis

Analyze more according to documents used in previous data analysis workshops.

An example of visualization

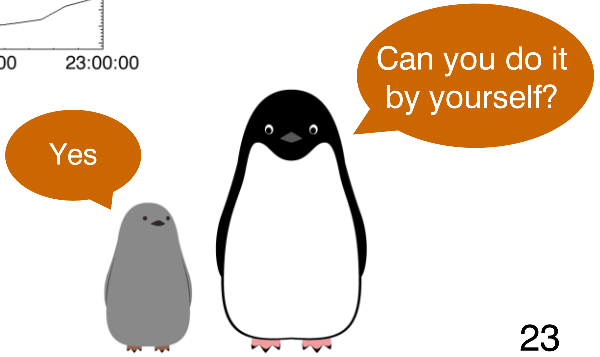
Troposphere/stratosphere wind data from the MU radar at Shigaraki.
1. Arrange data file by shell
2. Read it with UDAS egg
3. Apply FFT to the data.

Surface wind data (ASCII)
1. Read with UDAS egg
2. Apply FFT to meridional wind data.



Read solar wind data from Geotail satellite with SPEDAS.

Read geomagnetic indices with SPEDAS.



System requirements and notes

System requirements (as of Dec. 2018)

O S	Windows / MacOS / Linux
IDL	8.0 or higher. IDL license is unnecessary for GUI.
SPEDAS	2.0 or higher (3.0 or higher is recommended)
Supported file formats	CDF (Common Data Format) ASCII format

Notes

1. When you use the UDAS egg, please refer to the rules of the road of the IUGONET project.

   Rules of the road: http://www.iugonet.org/rules/
2. Please note that we do not take any responsibility or liability for any damage or loss caused by the UDAS egg.

IUGONET

Appendix 1. Program description

```
1   ;+
2   ; PROCEDURE:
3   ;   loadproc_template, site = site, $
4   ;                 datatype=datatype, $
5   ;                 trange=trange, $
6   ;                 verbose=verbose, $
7   ;                 downloadonly=downloadonly, $
8   ;                 no_download=no_download
9   ;
10  ; PURPOSE:
11  ;   This is an example showing how to create a load procedure for
12  ;   CDF or ASCII files.
13  ;
14  ; KEYWORDS:
15  ;   site  : site
16  ;   datatype : datatype
17  ;   trange : (Optional) Time range of interest  (2 element array).
18  ;   /verbose: set to output some useful info
19  ;   /downloadonly: if set, then only download the data, do not load it
20  ;          into variables.
21  ;   /no_download: use only files which are online locally.
22  ;
23  ; EXAMPLE:
24  ;   loadproc_template_cdf, site = 'aaa', datatype = 'bbb', $
25  ;           trange=['2003-11-20/00:00:00','2003-11-21/00:00:00']
26  ;
27  ; Written by Y.-M. Tanaka, March 13, 2018
28  ; Modified by Y.-M. Tanaka, September 5, 2018
29  ;-
30
```

We describe the template in detail in the appendix.

Line 1-29 header (lines commented out by “;”)

The header describes the summary of program, but it is not required.

If you want to share the program with other people or implement it into a package/large-scale software, the header description is helpful.

Appendix 1. Program description

```
31  pro loadproc_template, site=site, datatype=datatype, $
32          trange=trange, verbose=verbose, downloadonly=downloadonly, $
33                  no_download=no_download
34
35  ;*********************************;
36  ;***** Step1: Set parameters *****;
37  ;*********************************;
38  file_format='cdf'   ; Choose 'cdf' or 'ascii'
39  url='http://www.iugonet.org/data/SITE/DATATYPE/YYYY/mag_SITE_DATATYPE_YYYYMMDD_v??.cdf'
40  acknowledgement = 'You can write the data use policy here.' + $
41     'This description is displayed when you use this load procedure.'
42  site_list='sta1 sta2 sta3'
43  datatype_list='1sec 1min 1hr'
44
45  ;===== Split URL =====;
46  split_url, url=url, remote_data_dir=remote_data_dir, $
47     pathname_base=pathname_base, filename_base=filename_base
48  ipos_local=strpos(remote_data_dir, '://')+3
49  local_data_dir  = root_data_dir() + $
50     strmid(remote_data_dir, ipos_local, strlen(remote_data_dir)-ipos_local)   ; Base local directory
51  ; remote_data_dir='http://www.iugonet.org/data/'
52  ; local_data_dir  = root_data_dir() + 'tmp/'
53
```

--- Line 45～53: Find URL for data files and local directory for saving downloaded files ---

Line 46-47, split_url: Split url into URI, PATH, Filename

URL in line 39 is separated into 1. remote_data_dir: Base URL of remote data server, 2. pathname_base: Relative path of data files from the base URL, 3. filenames_base: filename. 1～3 are used later, so do not change the variable names.

Line 49-52, local_data_dir: Local directory for saving downloaded files

Base local directory for saving downloaded files when running spd_download (in line 126). root_data_dir() = "c:¥data", if OS is windows.

Line 51, remote_data_dir: If you want to customize it by yourself, you can modify it here.

Line 52, local_data_dir: If you want to customize it by yourself, you can modify it here.

Appendix 1. Program description

```
54  ;===== Keyword check =====;
55  ;----- default -----;
56  if ~keyword_set(verbose) then verbose=0
57  if ~keyword_set(downloadonly) then downloadonly=0
58  if ~keyword_set(no_download) then no_download=0
59  
60  ;----- remote_data_dir -----;
61  if remote_data_dir eq '' then remote_data_dir=' '
62  
63  ;----- site -----;
64  if n_elements(site_list) eq 0 then site_list='sta'
65  site_all = strsplit(site_list, /extract)
66  if(not keyword_set(site)) then site='all'
67  site = ssl_check_valid_name(site, site_all, /ignore_case, /include_all)
68  if site[0] eq '' then return
69  
70  ;----- datatype -----;
71  if n_elements(datatype_list) eq 0 then datatype_list='dt'
72  datatype_all=strsplit(datatype_list, /extract)
73  if(not keyword_set(datatype)) then datatype='all'
74  datatype=ssl_check_valid_name(datatype, datatype_all, /ignore_case, /include_all)
75  if datatype[0] eq '' then return
76  
```

--- Line 54~76: Check of keywords, parameters, variables and error handling ---

Line 56-58, ~keyword_set(kewords): *IDL function

Check if the input argument is defined. If not, set the keyword to the default values.

Line 61, [remote server URL] eq '': *IDL operator

Check if the remode_data_dir is defined. If not, set the remote_data_dir to ' '.

Line 64, n_elements(variable) eq 0: *IDL function

Check the number of elements of the variable. If no element, set the variable to default value.

Line 65, array = strsplit(string, /extract): *IDL function

Split a string into separate strings and return them as a string array.

Line 67, 値 = ssl_check_valid_name(string, string array, /ignore_case, /include_all): *SPEDAS function

Check if the string is included in the string array. The keyword "ignore_case" means that upper and lower cases are not distinguished. The keyword "include_all" allows users to set the string to "all".

Appendix 1. Program description

```
77  ;----- Make date & time string array -----;
78  ipos=strpos(url, 'hh')
79  if ipos lt 0 then hres=0 else hres=1
80  yyyy = file_dailynames(file_format='YYYY', trange=trange, hour_res=hres)
81  yy = file_dailynames(file_format='yy', trange=trange, hour_res=hres)
82  mm = file_dailynames(file_format='MM', trange=trange, hour_res=hres)
83  dd = file_dailynames(file_format='DD', trange=trange, hour_res=hres)
84  hh = file_dailynames(file_format='hh', trange=trange, hour_res=hres)
85  yyyymm = file_dailynames(file_format='YYYYMM', trange=trange, hour_res=hres)
86  yyyymmdd = file_dailynames(file_format='YYYYMMDD', trange=trange, hour_res=hres)
87  yyyymmddhh = file_dailynames(file_format='YYYYMMDDhh', trange=trange, hour_res=hres)
88
```

--- Line 77~88: Create string array for date and time (specialized process in UDAS egg) ---

Line 78, strops(string1, 'string2'): *IDL function

Find the first occurrence of the string2 within the string1. If found, it returns the position of the string2. If not found, it returns -1.

Line 80-87, file_dailynames(string for time format, time range, hour flag) *SPEDAS function

Set the defined string for time format (e.g., YYYY, yy, MM, DD, …) to concrete values within the time range. This is a specialized process in SPEDAS, so you don't have to modify it.

Appendix 1. Program description

```
89  ;===== Download files, read data, and create tplot vars at each site =====
90  ;----- Loop -----
91  for isite=0, n_elements(site)-1 do begin
92    for idt=0, n_elements(datatype)-1 do begin
93
94      ;----- Set parameters for spd_download -----;
95      source = file_retrieve(/struct)
96      source.verbose = verbose
97      source.local_data_dir  = local_data_dir
98      source.remote_data_dir = remote_data_dir
99      if keyword_set(no_download) then source.no_download = 1
100     if keyword_set(downloadonly) then source.downloadonly = 1
101
…     …

191   endfor
192 endfor
```

--- Line. 89-192: Create communication object for data download ---

Line 91, for [site array] do begin *IDL basic

Loop for site list

Line 92, for [datatype array] do begin *IDL basic

Loop for data type

Replace the strings “SITE” and “DATATYPE” with concrete values in these two loops.

Line 95, source = file_retrieve(/struct) *SPEDAS function

Create the communication object for data download.

Line 96, source.verbose = verbose *SPEDAS object

If verbose = 1, detailed information of process is displayed on the window.

Line 97, source.local_data_dir = local_data_dir *SPEDAS object

Set it to local_data_dir (base local directory for saving downloaded files)

Line 98, source.remote_data_dir = remote_data_dir *SPEDAS object

Set it to remote_server_dir (base remote server URL)

Line 99, source.no_download = 1 *SPEDAS object

If you do not want to download files and want to read files on PC, set it to 1.

Line 99, source.downloadlonly = 1 *SPEDAS object

If you want to download files only and do not want to read them, set it to 1.

IUGONET

Appendix 1. Program description

```
102  ;----- Make relpathnames -----;
103    replace_strings, pathname_base, 'SITE', site[isite], pathnames
104    replace_strings, pathnames, 'DATATYPE', datatype[idt], pathnames
105    replace_strings, pathnames, 'YYYYMMDDhh', yyyymmddhh, pathnames
106    replace_strings, pathnames, 'YYYYMMDD', yyyymmdd, pathnames
107    replace_strings, pathnames, 'YYYYMM', yyyymm, pathnames
108    replace_strings, pathnames, 'YYYY', yyyy, pathnames
109    replace_strings, pathnames, 'yy', yy, pathnames
110    replace_strings, pathnames, 'MM', mm, pathnames
111    replace_strings, pathnames, 'DD', dd, pathnames
112    replace_strings, pathnames, 'hh', hh, pathnames
113    replace_strings, filename_base, 'SITE', site[isite], filenames
114    replace_strings, filenames, 'DATATYPE', datatype[idt], filenames
115    replace_strings, filenames, 'YYYYMMDDhh', yyyymmddhh, filenames
116    replace_strings, filenames, 'YYYYMMDD', yyyymmdd, filenames
117    replace_strings, filenames, 'YYYYMM', yyyymm, filenames
118    replace_strings, filenames, 'YYYY', yyyy, filenames
119    replace_strings, filenames, 'yy', yy, filenames
120    replace_strings, filenames, 'MM', mm, filenames
121    replace_strings, filenames, 'DD', dd, filenames
122    replace_strings, filenames, 'hh', hh, filenames
123    relpathnames = pathnames + filenames
124
```

--- Line. 102-123: URL and filename (specialized process in UDAS egg) ---

Line 103-123, Create relative paths from base server URL and filenames.

Replace the defined strings within pathnames and filenames with concrete values and create relative paths from base server URL and filenames.

This is a specialized process in UDAS egg, so you don’t have to modify it.

Appendix 1. Program description

```
125  ;----- Download data files -----;
126    files = spd_download(remote_file=relpathnames, $
127            remote_path=source.remote_data_dir, $
128            local_path=source.local_data_dir, $
129            no_server=no_server, $
130            no_download=no_download, $
131            _extra=source, /last_version)
132
133    filestest=file_test(files)
134    if total(filestest) ge 1 then begin
135      files=files(where(filestest eq 1))
136    endif
137
```

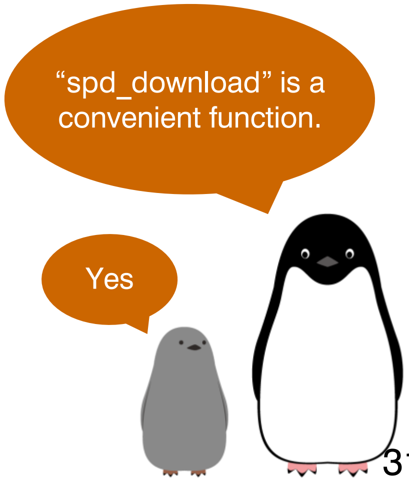
--- Line. 125-137: Download data files ---

Line 126-131, spd_download(.., ..) *SPEDAS fuction

Download data files from the remote server by using spd_download function included in SPEDAS.

Line 133-137, Check downloaded files *IDL function

Check whether or not the data files were downloaded to PC by using file_test (1: true, 0:fase).
Only downloaded files are set to a variable “files”.



Appendix 1. Program description

```
138  ;----- Load data into tplot variables -----;
139    if(downloadonly eq 0) then begin
140        ;************************************************;
141        ;***** Step2: Load data into tplot variables *****;
142        ;************************************************;
143        case file_format of
144          'cdf': begin
145            ;----- For CDF format files -----;
…            …  skip …
148            cdf2tplot, files=files, verbose=source.verbose, $
149               prefix=prefix, suffix=suffix
150          end
151          'ascii': begin
152            ;----- For ASCII format files -----;
…            … skip …
164            ascii2tplot, files, files=files2, format_type=format_type, $
165               tformat=tformat, tvar_column=tvar_column, $
166               tvarnames=tvarnames, delimiter=delimiter, $
167               data_start=data_start, comment_symbol=comment_symbol, $
168               v_column=v_column, vvec=vvec, $
169               time_column=time_column, input_time=input_time
170          end
171          else: begin
172            print, 'Not support the file format: '+file_format
173            return
174          end
175        endcase
…          … skip …
190    endif
```

--- Line. 138-190: Read data files and create tplot variables ---

Line 148-149, cdf2tplot *SPEDAS function

Read CDF files and create tplot variables by using cdf2tplot function included in SPEDAS.

Line 164-169, ascii2tplot *UDAS egg function

Read ASCII files and create tplot variables by using ascii2tplot function included in UDAS egg.

Appendix 1. Program description

```
194  ;----- Display data policy -----;
195  print_str_maxlet, acknowledgement
196
197  end
```

--- Line. 194-195: Display data usage policy---

Line 194-145, print_str_maxlet, acknowledgement *SPEDAS function

Display the strings defined in the acknowledgement on the window.

The process has been completed.
You can visualize and analyze the loaded tplot variables in the next step.