MEAKit – A simple toolkit for processing data from Multi-electrode arrays (MEA)

Another open-source toolbox for accessing and analyzing MEA data

A simple sketch in MATLAB of
loosely-coupled functions and scripts for
easily reusing, creating and sharing your own codes.

Pu Jiangbo
Britton Chance Center for Biomedical Photonics
Wuhan National Laboratory for Optoelectronics - Huazhong University of Science and Technology
Wuhan, Hubei, 430074, P. R. China
http://bmp.hust.edu.cn/
Jiangbo.pu@gmail.com

Version: 1.5.9 May 2010

Getting started

MEAKit is originally created for processing data acquired by Multi-channel Systems devices. Inspired by FIND Toolbox (http://find.bccn.uni-freiburg.de/), the later development of MEAKit embraces the light of Neuroshare (http://neuroshare.sourceforge.net/) which makes it compatible with data formats produced by other manufacturers.

MEAKit is not going to compete with other existing powerful toolboxes like FIND and MEABench. It is just another choice for new researchers to build up their own foundations. The design of MEAKit is based on dataflow. Assuming there is a data stream from the sources (i.e. a single or group of data files, and maybe the device itself), we shall name it as ***raw data***, and it contains the complete and continuous waveforms of the signals acquired from MEA. The structure contains this stream is labeled as ***rawif*** which is passing through scripts and functions. Sometimes, the data file contains a pre-processed data stream of spikes (***spike data***) instead of raw data (e.g. after the processing of spike sorter in MC-Rack).The structure in MEAKit that contains spike data is labeled as ***spif*** which is also been extensively used among MEAKit. The data file may also contain some important information that holds the timestamps of triggers (if you have recorded them in a stimulation experiment). The structure contains trigger stream is labeled as ***trif***. The notion of these three structures is borrowed from MEATools, if you have experience with MEATools, you will be familiar with it. MEAKit is currently not an object-oriented framework, but a bunch of very simple and goal-directed MATLAB scripts and functions based on the data flow in the form of ***rawif***, ***spif*** and ***trif***. Under the circumstances of reading data files produced by non-default manufacturers (the default one is Multi-channel Systems - *.mcd), data will be read and translated into ***raw/sp/trif*** with the help of Neuroshare.

Functions are organized by their objectives. In MEAKit, data must be first loaded from "IO" functions which will turn it into corresponding information structures. Then the structures can be sent to the "Calculation" functions as input parameters. Results from "Calculation" function can be illustrated by MATLAB built-in plotting functions or our home-made scripts in "Plotting". We deeply understand the need of extending more functions, it will be still neat and tidy if you are coding with the above rules kept in mind, which may also helps others to understand your goal and improve the reusing of your lovely codes.

Installation

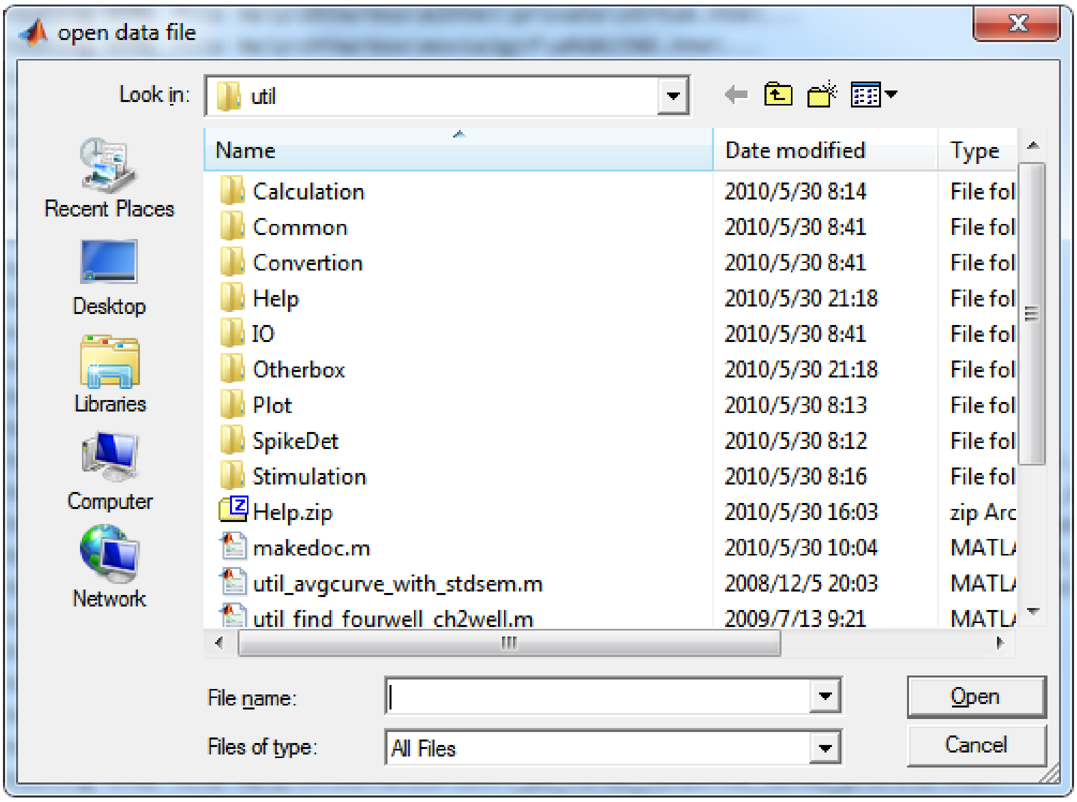
The installation of MEAKit is easy. Just add the directory of MEAKit (including subdirectories) into the path of MATLAB.

Load data

You can load the whole length of data or just a part of it from a single datafile or a group of datafile (automatically sorted by their recording time). For example, if you want to load the spike stream from a specific single MCD file, you will use:

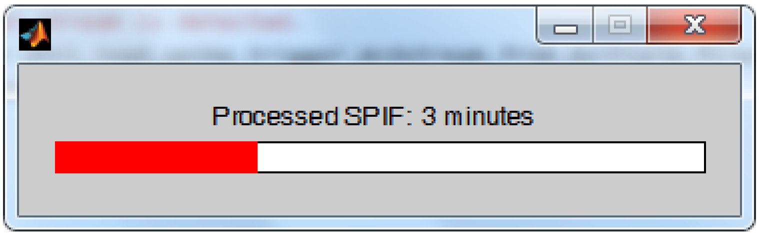
```
[d spif trif] = util_load_spike_trigger_mcdstream('isCompact',true);
```

Then a dialog will pop up for you to select a data file (Fig. 1):



Name	Date modified	Type
Calculation	2010/5/30 8:14	File fol
Common	2010/5/30 8:41	File fol
Convertion	2010/5/30 8:41	File fol
Help	2010/5/30 21:18	File fol
IO	2010/5/30 8:41	File fol
Otherbox	2010/5/30 21:18	File fol
Plot	2010/5/30 8:13	File fol
SpikeDet	2010/5/30 8:12	File fol
Stimulation	2010/5/30 8:16	File fol
Help.zip	2010/5/30 16:03	zip Arc
makedoc.m	2010/5/30 10:04	MATL/
util_avgcurve_with_stdsem.m	2008/12/5 20:03	MATL/
util_find_fourwell_ch2well.m	2009/7/13 9:21	MATL/

Fig.1 The open file dialog



Processed SPIF: 3 minutes

Fig.2 The progress bar when reading

The function will return three variables: d – the file object, spif – the spike information structure, trif – the trigger information structure (if any). In this example, the function just returns as:

```
Start:21-May-2010 02:07:24, Stop: 21-May-2010 02:37:23, Length: 1799900 ms.
Note: No trigger stream is detected.
```

This will help us know that the file contains the recording started from 21-May-2010 02:07:24 and ended at 02:37:23 with a total length of 1799900 ms. Besides, no trigger information is detected in that file.

This function takes other forms of input parameters. We can specify the filename without using File Open Dialog and the region of interest recordings. For example:

```
[d spif1 trif] = util_load_spike_trigger_mcdstream('filename', 'G:\test.mcd', 'isCompact',
true, 'startend',[0 100000]);
```

This will read from the file named “test.mcd” for the region of 0 – 100000 ms. You can go and check the documentation of **util_load_spike_trigger_mcdstream** for more details.

Sometimes, you may need to read from a group of data files sorted by their recording times. The function of **util_load_spike_trigger_mcdstream_from_multiple_files** will help you out. For example:

```
[spif trif] = util_load_spike_trigger_mcdstream_from_multiple_files('isCompact',true);
```

The same File Open Dialog will pop up. But this time, you can select multiple files. Don’t worry about the sequence of your clicks. The function will sort these data files internally in an ascending rule by their recording times. It will call **util_load_spike_trigger_mcdstream** and automatically connect the information structures into one. In this example, the function displays the following information while it is running:

```
The CONTINUOUS mode is on by default.


 ==============
   MCD Files
 ==============

[1] G:\MC_Rack Data\Stimulation\DualElec\74105-0316\20100512\74105-0316-0512 1 Baseline
715s SPK.mcd
    Start: 12-May-2010 07:38:47. Length:714900 ms.
[2] G:\MC_Rack Data\Stimulation\DualElec\74105-0316\20100512\74105-0316-0512 1 Baseline
715s0001 SPK.mcd
    Start: 12-May-2010 07:50:42. Length:715000 ms.
[3] G:\MC_Rack Data\Stimulation\DualElec\74105-0316\20100512\74105-0316-0512 1 Baseline
715s0002 SPK.mcd
    Start: 12-May-2010 08:02:37. Length:715000 ms.


[1] 74105-0316-0512 1 Baseline 715s SPK.mcd is been processing.
Start:12-May-2010 07:38:47, Stop: 12-May-2010 07:50:42, Length: 714900 ms.
 Note: No trigger stream is detected.
[2] 74105-0316-0512 1 Baseline 715s0001 SPK.mcd is been processing.
Start:12-May-2010 07:50:42, Stop: 12-May-2010 08:02:37, Length: 715000 ms.
 Note: No trigger stream is detected.
[3] 74105-0316-0512 1 Baseline 715s0002 SPK.mcd is been processing.
Start:12-May-2010 08:02:37, Stop: 12-May-2010 08:14:32, Length: 715000 ms.
```

```
Note: No trigger stream is detected.
```

You can check if the sequence of reading is wrong by the table of “MCD Files”. The information of each data file is displayed individually. For more information, please go and check the documentation.

Process the data – ASDR / SPB

After we get the spif, it can be sent to “Calculation” functions (built-in or home-made) for further processing. For example, the ASDR (arraywide spike detection rate) or SPB (spikes per bin) can be calculated using **util_calc_spb_avg**.

```
[spb] = util_calc_spb_avg('spif',spif,'gnd',15);
```

The function requires at least two inputs. ‘**spif**’ is the spike information structure, and ‘**gnd**’ is the grounding electrode which will be excluded from the following calculation. The default bin is 1000 ms. If you want to specific another bin width, you can set it like this:

```
[spb] = util_calc_spb_avg('spif',spif,'gnd',15,’bin’,2000);
```

Then the width of each individual bin is set to 2000 ms. After **spb** is generated, we can illustrate it using any plotting tools you prefer, for example, the built-in **plot** (Fig. 3).

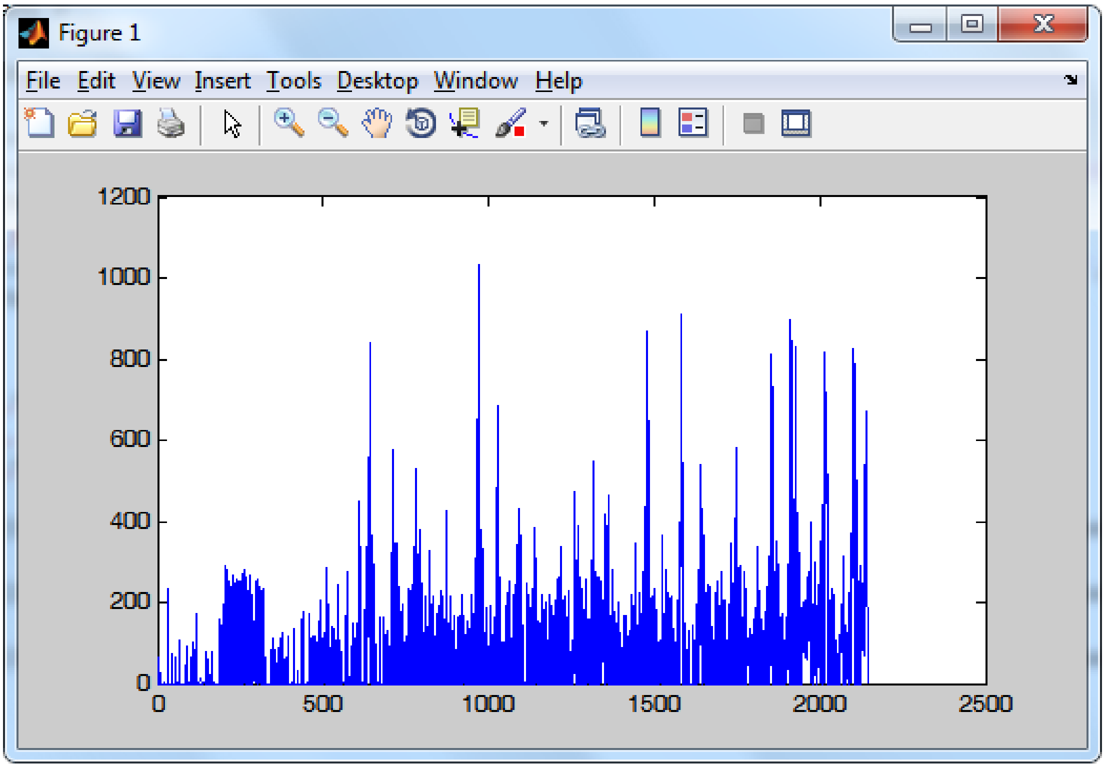


Fig.3 Plot result of the ASDR - using MATLAB command: **plot(spb)**

Process the data – Cross-correlogram

The function **util_calc_raw_spont_cch()** can be used to calculate the raw cross-correlogram for spontaneous recordings. “Raw” here emphasizes the cross-correlogram is based on the original algorithm instead of any other revised or modified (corrected / improved) ones.

```
>> [lags cch] = util_calc_raw_spont_cch('spif',spif,'ref',55,'obs',56);
```

```
bin(10ms), Ref(#55), Obs(#56)
```

This will generate the cross-correlogram using the parameters set as: Reference Electrode = #55; Observing Electrode = #56, Bin = 10 ms (default). We can use the built-in **bar** command to illustrate the result (Fig. 4).

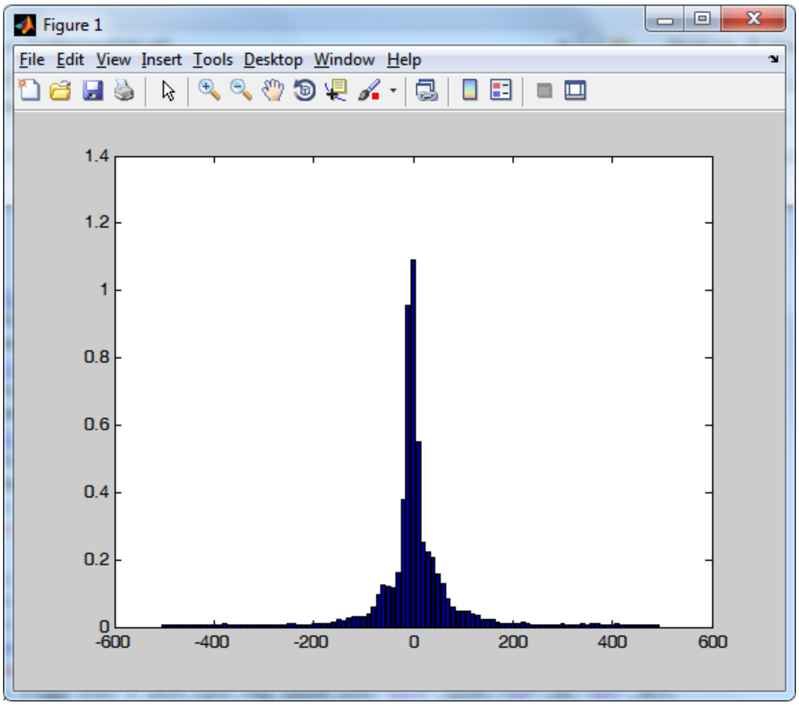


Fig.4 Result of cross-correlogram - using MATLAB command **bar(lags, cch)**

If we want to know more about the peak in cross-correlogram, we can add one more output parameter like this:

```
[lags cch peak] = util_calc_raw_spont_cch('spif',spif,'ref',55,'obs',56);
```

The **peak** contains:

```
peak =

       value: 1.0924
    position: 0
       width: 26.3383
```

It means the peak has a value of 1.0924 and located at Zero point. The width of this peak is 26.3383 ms. For more information, please refer to the documentation of **util_calc_raw_spont_cch()**.

Visualize the data – Raster plot

Raster plot is one of the most frequently used plot types in neuroscience. MEAKit provides the function of **util_plot_raster_simple** for you to make a raster very easily. Just type in:

```
util_plot_raster_simple(spif,15,0)
```

The raster will be plotted after several seconds (depends on the speed of your computer). It is based on the provided **spif**. The second parameter is the number of grounding electrode. The GND will not be plotted. The third parameter tells the function to draw it slow but more accurately (not in "quickmode", you can refer to the manual of this function). Run this command and we will get a figure like in Fig.5.

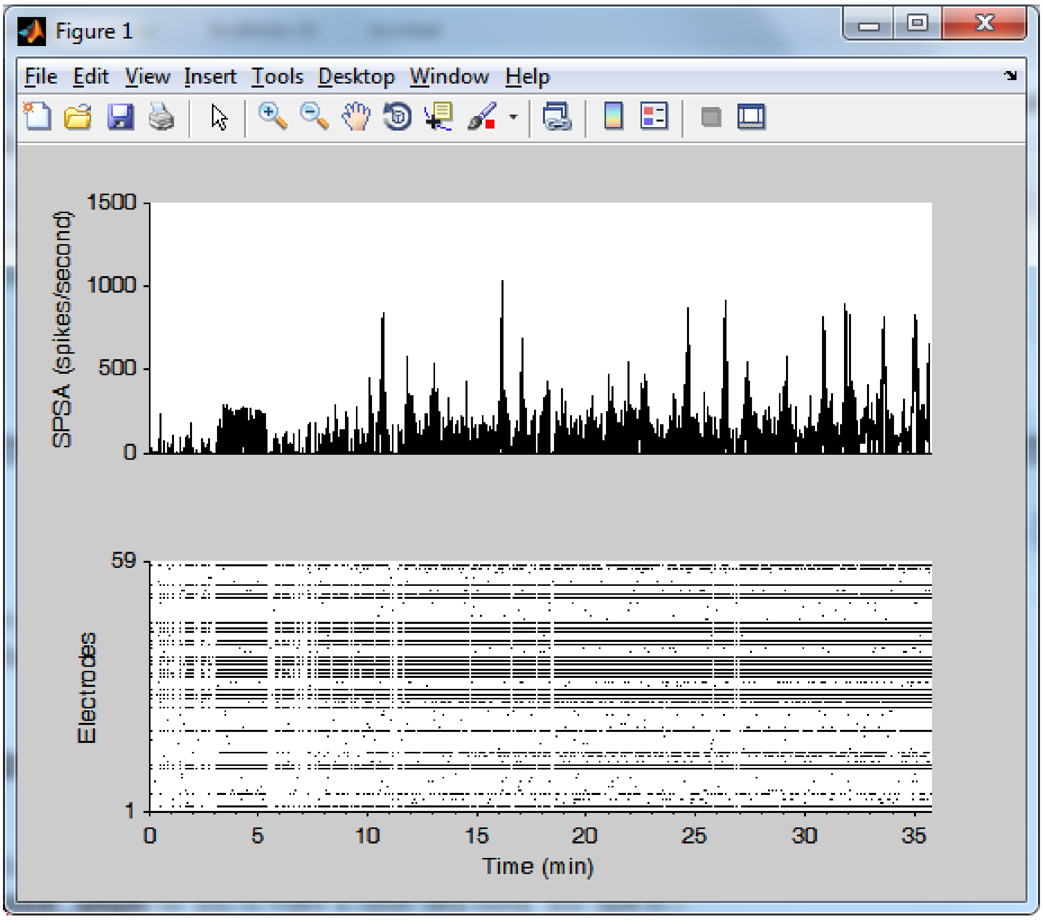


Fig.5 The rasterplot

Get help

Every function has been documented as possible as we can. You can get the online help by typing:

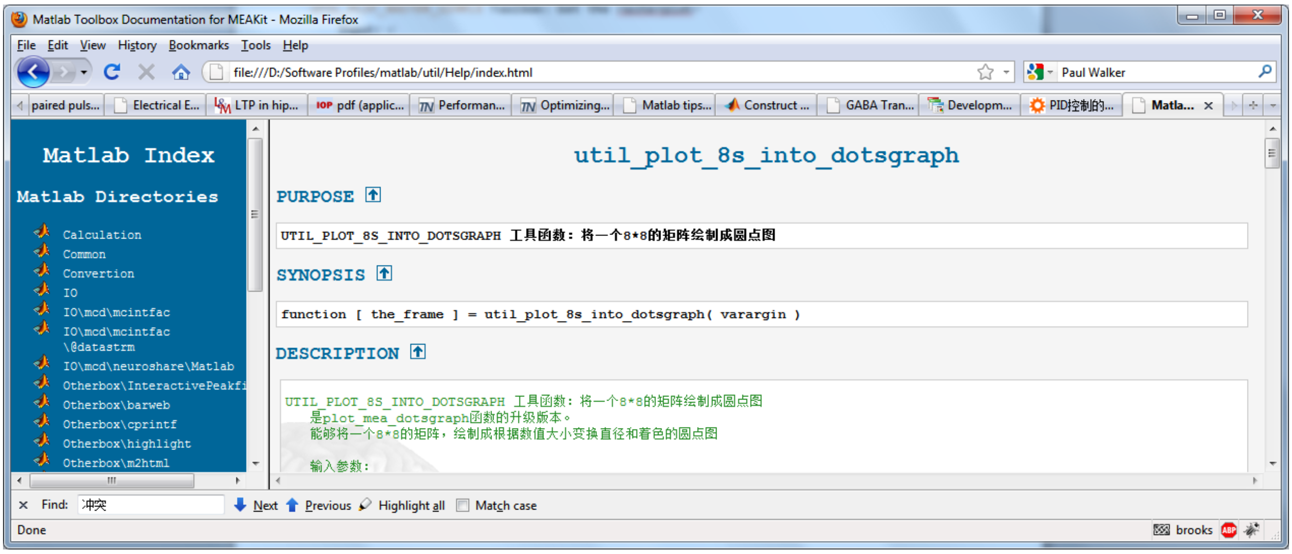
```
>> help util_plot_raster_simple
```

MATLAB will output the help message like the following:

```
UTIL_PLOT_RASTER_SIMPLE Toolbox: Get the rasterplot
    Input:
            'spif', spike information structure.
            'gnd', the number of grounding electrode.
            'quickmode', specific if is in the Quick Mode.

    Pu Jiangbo May-27-2010 (jiangbo.pu@gmail.com)
```

The output contains the basic usage of **util_plot_raster_simple()** and the author information (maybe versioning information). For the complete manual of MEAKit, you can go to the "Help" subfolder under MEAKit's root directory and open the "index.html" to launch a comprehensive help page (based on framed html pages, Fig. 6).



Matlab Index

Matlab Directories

- Calculation
- Common
- Convertion
- IO
- IO\mcd\mcintfac
- IO\mcd\mcintfac \@datastrm
- IO\mcd\neuroshare\Matlab
- Otherbox\InteractivePeakfi
- Otherbox\barweb
- Otherbox\cprintf
- Otherbox\highlight
- Otherbox\m2html

util_plot_8s_into_dotsgraph

PURPOSE

```
UTIL_PLOT_8S_INTO_DOTSGRAPH 工具函数：将一个8*8的矩阵绘制成圆点图
```

SYNOPSIS

```
function [ the_frame ] = util_plot_8s_into_dotsgraph( varargin )
```

DESCRIPTION

```
UTIL_PLOT_8S_INTO_DOTSGRAPH 工具函数：将一个8*8的矩阵绘制成圆点图
   是plot_mea_dotsgraph函数的升级版本。
   能够将一个8*8的矩阵，绘制成根据数值大小变换直径和着色的圆点图

   输入参数：
```

Fig.6 The help pages

Contact and Links

Don't feel hesitate to reach us if you have any problems.

Project organizer: Xiangning Li (lixiangning@mail.hust.edu.cn)

Program: Jiangbo Pu (jiangbo.pu@gmail.com)

Homepage of Britton Chance Center for Biomedical Photonics: http://bmp.hust.edu.cn/en/