

Easy4PPP

User Manual

Zhuojun Jiang^{1,2}, Zeen Yang^{1,2}, Wenjing Huang^{1,2},
Chuang Qian^{1,2}

¹Intelligent Transportation System Research Center, Wuhan
University of Technology

²School of Navigation, Wuhan University of Technology

Email: zhuojun_jiang@whut.edu.cn&1162110359@qq.com

Content

1. Introduction	2
2. Downloading and Preparations	3
2.1 Downloading the Easy4PPP	3
2.2 Unzip the Sample Data	4
2.3 Python and Environment Preparations	6
3. Running the Easy4PPP	8
3.1 Running Easy4PPP in sPPP.py	8
3.2 Running Easy4PPP in ppp.ipynb	10
3.3 Visualizing Easy4PPP products in nav_result.ipynb	11
4. The GUI software of Easy4PPP	13

1. Introduction

Easy4PPP is an open-source Python toolbox about precise point positioning (PPP). With object-oriented syntax and multi-platform adaptability in programming language level, Easy4PPP is friendly to both the Global Navigation Satellite System (GNSS) scholars and the Positioning, Navigation, Timing (PNT) related researchers.

Easy4PPP mainly consist of 4 parts: the **Python-coded core computation library**, the **PPP and products visualization Jupyter Notebook tutorials**, the **User manual** and the **Graphical User Interface (GUI) software**. Easy4PPP aims to lowering the barrier of using PPP for positioning and derived products generation. All the Input/Output path is open for users, both experts well-versed in PPP principles and researchers from related fields who only need to use PPP-derived products can easily utilize Easy4PPP and engage in secondary development.

In this released version of Easy4PPP, we mainly focus on basic PNT computation and dual-frequency undifferenced and uncombined PPP. A data example is provided in the released package. The Section 2 is about downloading and preparation for running Easy4PPP. The Section 3 is about running Easy4PPP in Python environment of both Windows and Linux. The Section 4 is about running Easy4PPP_GUI in Windows. Any suggestions or bug reports are welcomed by sending email to authors.

2. Downloading and Preparations

Easy4PPP make sure that all the dependencies are originally included in the source package so that **git clone is not necessary** for using it. Easy4PPP is now available at <https://github.com/alxanderjiang/Easy4PPP>. Ensure that your Internet connection is stable and capable of accessing GitHub. In some trouble, please sending email to zhuojun_jiang@whut.edu.cn or 1162110359@qq.com for latest update.

2.1 Downloading the Easy4PPP

We highly recommend that download Easy4PPP in zip format on our Github home page (<https://github.com/alxanderjiang/Easy4PPP>). Just like it shown in Figure 2.1.

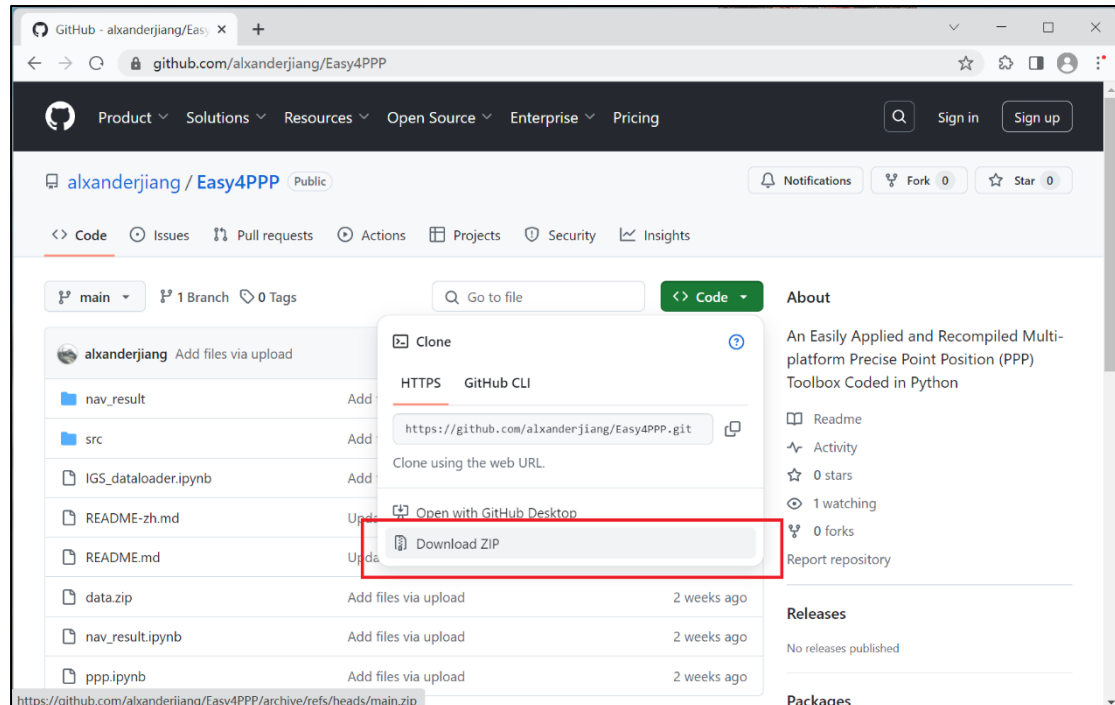


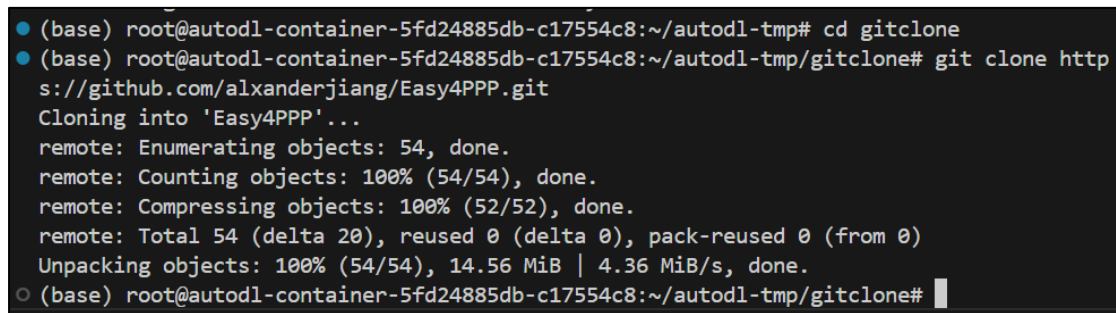
Figure 2.1 Downloading Easy4PPP on home page

No matter Windows or Linux platform, this zip downloading way is

available. If your device or system only supports terminal operations, such as some Linux workstations without a graphical desktop environment installed, Git clone will be your only way to obtain Easy4PPP. Please open a new terminal on your device and run the following command:

```
git clone https://github.com/alxanderjiang/Easy4PPP.git
```

Figure 2.2 is an example of downloading Easy4PPP through git clone.



```
(base) root@autodl-container-5fd24885db-c17554c8:~/autodl-tmp# cd gitclone
(base) root@autodl-container-5fd24885db-c17554c8:~/autodl-tmp/gitclone# git clone https://github.com/alxanderjiang/Easy4PPP.git
Cloning into 'Easy4PPP'...
remote: Enumerating objects: 54, done.
remote: Counting objects: 100% (54/54), done.
remote: Compressing objects: 100% (52/52), done.
remote: Total 54 (delta 20), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (54/54), 14.56 MiB | 4.36 MiB/s, done.
(base) root@autodl-container-5fd24885db-c17554c8:~/autodl-tmp/gitclone#
```

Figure 2.2 Downloading Easy4PPP through git clone

2.2 Unzip the Sample Data

Due to GitHub's file size limit for individual uploads (lower than 25MB), we are unable to directly upload the observation files, precise ephemeris, precise clock products, antenna files, and other data required for the examples to the GitHub repository. Therefore, the Easy4PPP project includes a `data.zip` file. Before using Easy4PPP, this folder must be extracted into the Easy4PPP project directory. **It is worth noting that Easy4PPP does not require a standard file structure;** the file structure after extracting `data.zip` is provided only for reference and to drive the example programs. Figure 2.3 shows how to unzip the sample data on Windows.

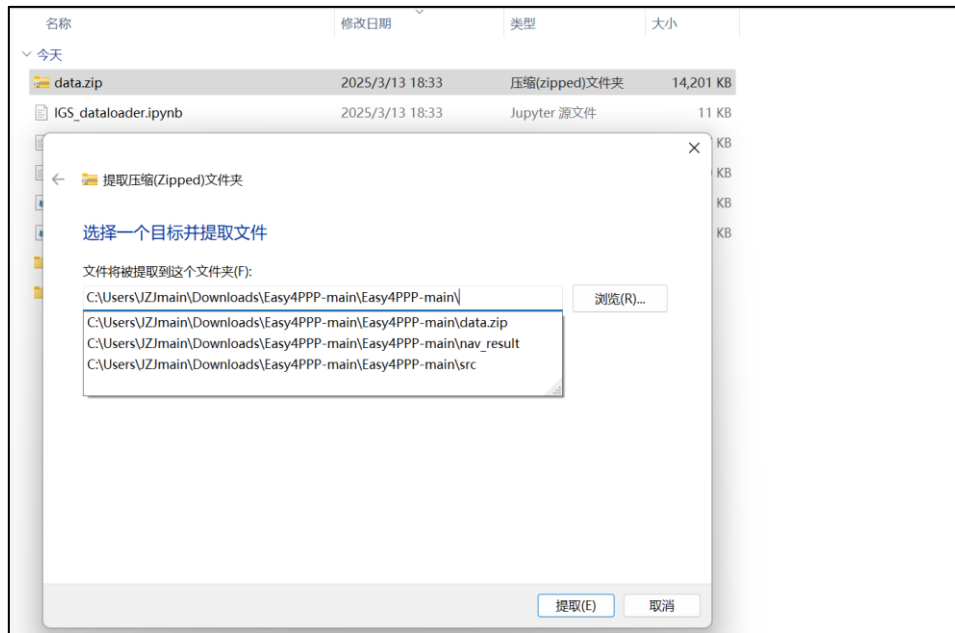


Figure 2.3 Unzip the sample data on Windows

For Linux but not git clone ways, run the following commands:

```
unzip Easy4PPP-main
cd Easy4PPP-main
unzip data.zip
```

For Linux by git clone ways, run the following commands:

```
cd Easy4PPP
unzip data.zip
```

Figure 2.4 shows an example of unzipping the sample data by Linux terminal.

```
(base) root@autodl-container-5fd24885db-c17554c8:~/autodl-tmp/gitclone# cd Easy4PPP
(base) root@autodl-container-5fd24885db-c17554c8:~/autodl-tmp/gitclone/Easy4PPP# unzip data.zip
Archive: data.zip
  creating: data/
  creating: data/ATX/
  inflating: data/ATX/igs14.atx
  inflating: data/ATX/igs20.atx
  creating: data/brdc/
  inflating: data/brdc/BRDC00IGS_R_20250350000_01D_MN.rnx
  creating: data/DCB/
  inflating: data/DCB/CAS10PSRAP_20250350000_01D_01D_DCB.BIA
  creating: data/OBS/
  creating: data/OBS/WUH2/
  inflating: data/OBS/WUH2/WUH200CHN_R_20250350700_01H_30S_MO.25o
  creating: data/Peph_clk/
  creating: data/Peph_clk/20250350/
  inflating: data/Peph_clk/20250350/WUM0MGXFIN_20250350000_01D_05M_ORB.SP3
  inflating: data/Peph_clk/20250350/WUM0MGXFIN_20250350000_01D_30S_CLK.CLK
  creating: data/SNX/
  inflating: data/SNX/IGS00PSSNX_20250350000_01D_01D_CRD.SNX
(base) root@autodl-container-5fd24885db-c17554c8:~/autodl-tmp/gitclone/Easy4PPP#
```

Figure 2.4 Unzip the sample data on Linux

The sample file structure of Easy4PPP after downloading and unzipping can be shown as follows:

```
Easy4PPP
.....data
.....ATX
.....igs20.atx
.....brdc
.....BRDC00IGS_R_20241320000_01D_MN.rnx
.....DCB
.....CAS1OPSRAP_20241320000_01D_01D_DCB.BIA
.....OBS
.....JFNG
.....jfng1320.24o
.....Peph_clk
.....wum23136.sp3
.....wum23136.clk
.....SNX
.....IGS0OPSSNX_20241320000_01D_01D_CRD.SNX
.....nav_result
.....jfng1320.24o.out.npy
.....src
.....RINEX.py
.....satpos.py
.....sppp.py
.....ppp.ipynb
.....nav_result.ipynb
.....ppp_test.ipynb
.....README.md
.....README-zh.md
.....UserManual.pdf
```

2.3 Python and Environment Preparations

Easy4PPP is coded in pure Python so that a suitable Python environment is required unless you only need to use the compiled Easy4PPP_GUI.exe. We've tested various Python versions and ensure that Easy4PPP can run successfully from version 3.7 to version 3.13. As for virtual environment, the “venv” provided by Python is OK and the tutorial can be found at <https://docs.python.org/3/tutorial/venv.html>. However, We

recommend using Anaconda3 for create an virtual environment as follows.

No matter Windows or Linux, your conda environment needs to have the following packages: **numpy** and **tqdm** for core computation, **matplotlib** for visualization and **ipykernel** for running Jupyter Notebook tutorials. To avoid potential package version conflicts in the environment, we strongly recommend creating a clean virtual environment by running the following commends in terminal:

```
conda create --name Py39 python==3.9
```

```
pip install numpy
```

```
pip install tqdm
```

```
pip install matplotlib
```

```
pip install ipykernel
```

Easy4PPP does not require specific versions of third-party libraries, as long as they are compatible with the Python version. At this point, all preparations for running Easy4PPP have been completed, including the project clone, sample data downloading and environment preparations.

3. Running the Easy4PPP

Easy4PPP provide varies ways to run the code in order to meet the requirements of diverse users. For PPP experts, we recommend using the main function of `sppp.py` or construct your own main function. For PPP beginners, we recommend using the Jupyter Notebook `ppp.ipynb` to get block to block solutions. For PPP developers or GNSS students, we recommend debug the Jupyter Notebook `ppp_test.ipynb`. For users from related aspects who needs PPP products, we recommend running the GUI software: `Easy4PPP_GUI.exe` directly.

3.1 Running Easy4PPP in sppp.py

`sppp.py` is a pure Python core source file for Single Point Position (SPP) and Precise Point Position (PPP) computation. We added a main function in the bottom of `sppp.py` to show how to use `sppp.py` to conduct a console PPP software. No matter Windows or Linux, running the following commends in terminal to run `sppp.py`:

```
python src/sppp.py
```

Before running this commends make sure that your terminal is in the main path of Easy4PPP (Easy4PPP-main for github zip and Easy4PPP for git clone) or you will get an error of `"No module named satpos.py"`. Figure 3.1 shows the output in terminal when running the main function of `sppp.py`. It is worth noticing that the main function of `sppp.py` is used to compute for the sample data. If you wants to use your own data, please change the

path and configs variables of the main function as Table 3.1, Table 3.2 shows.

```

(base) root@autodl-container-5fd24885db-c17554c8:~/autodl-tmp# cd Easy4PPP
(base) root@autodl-container-5fd24885db-c17554c8:~/autodl-tmp/Easy4PPP# python src/sppp.py
End index not set, solve all the observations. Total: 2881
Satellites outside for no precise eph ['C01', 'C12', 'C15', 'C17', 'C18', 'C25', 'C31', 'G01']
32%|██████████| 912/2881 [02:08<05:20, 6.14it/s]
G17 发生周跳 GF:22.96768195927143->-9.767793253064156 Mw:-74.53959484025836->59.1043582893908 p1:23596234.348709233 1
1:123998993.786 p2:23596248.210140567 12:96622632.558 dN1:3219.513825213606 dN2:3085.8698723917114
34%|██████████| 992/2881 [02:20<05:25, 5.80it/s]
G25 发生周跳 GF:-3.438590805977583->-8.593529056757689 Mw:23.01287765055895->44.89031923562288 p1:23548357.55562323 1
1:123747428.914 p2:23548365.669370595 12:96426603.174 dN1:523.2519645400165 dN2:501.37452295495257
53%|██████████| 1530/2881 [03:40<02:04, 10.81it/s]
G13 发生周跳 GF:-17.507985219359398->-0.10978041961789131 Mw:12.413233514875174->2.2873761132359505 p1:23303812.30892
464 11:122462353.876 p2:23303813.162306167 12:95425211.262 dN1:-520.6197036630602 dN2:-510.493846261421
56%|██████████| 1600/2881 [03:47<01:52, 11.36it/s]
G10 发生周跳 GF:1.0477696508169174->0.2892463840544224 Mw:-8.4066460467875->-2.6285147480666637 p1:24064932.530926894
11:126462083.805 p2:24064933.949711546 12:98541882.3 dN1:127.01466631310795 dN2:121.23653501438712
84%|██████████| 2429/2881 [05:17<00:44, 10.14it/s]
sat_num<4, pass epoch.
No valid observations, Pass epoch: Week: 2313, sec: 591300.0.
100%|██████████| 2881/2881 [06:12<00:00, 7.74it/s]
(base) root@autodl-container-5fd24885db-c17554c8:~/autodl-tmp/Easy4PPP#

```

Figure 3.1 Output in terminal when running the sppp.py directly

Table 3.1 The variables about paths in sppp.py

Variables	Description
obs_file	The path of observation data file in format >= RINEX v3.0
obs_type	The sign codes of selected dual-frequency observations. eg: ['C1C','L1C','D1C','S1C','C2W','L2W','D2W','S2W']
SP3_file	IGS precise orbit products in format of sp3
CLK_file	IGS precise clock products in format of clk
ATX_file	IGS antenna products in format of atx
BRDC_file	Broadcast ephemeris in format >=RINEXv2.0
ion_param	Klobuchar Ionospheric model parameters. If not set, read from BRDC_file
out_path	The output path of solution logs. If not set, “nav_result” automatically.
dcb_correction	The sign of DCB correction option. 0 for DCB correction off while 1 for on.
dcb_products	The origin of DCB products. “CAS” for Chinese Academy of

	Sciences while “CODE” for Center for Orbit Determination in Europe
dcb_file_0	The file of difference code bias between two frequencies.
dcb_file_1	The file of difference code bias between two codes in the first frequency. If dcb_products==“CAS”, this variable does not make sense.
dcb_file_2	The file of difference code bias between two codes in the second frequency. If dcb_products==“CAS”, this variable does not make sense.

Table 3.2 The variables about configs in spps.py

Variables	Description
obs_start	The first index of the epochs need to be processed.
obs_ephch	The total length of the epochs need to be processed.
out_age	The threshold of when to justice the loss of lock, in seconds.
sys_index	The systems that need to be read from observation file.
sys	The system that used in PPP.
dy_mode	The dynamic mode of PPP, static or dynamic.
f1	The first frequency, in MHz.
f2	The second frequency, in MHz.
el_threthod	The threshold of satellite elevation.
ex_threshold_v	The threshold of prior residuals.
ex_threshold_v_sigma	The threshold of posteriori residual multiple.
Mw_threshold	The threshold of phase slip detection in MW combination.
GF_threshold	The threshold of phase slip detection in GF combination.

3.2 Running Easy4PPP in ppp.ipynb

If you are familiar with Jupyter Notebook, we highly recommend you

try the Jupyter Notebook tutorials provided in Easy4PPP. Before running the `ppp.ipynb`, ensure that `ipykernel` is available in your virtual environment. If not, please run the following command:

```
pip install ipykernel
```

Open the `ppp.ipynb` in Jupyter Notebook or another code editors (for example, Microsoft Visual Studio Code) and run all the blocks. You will get a solution log of the sample data in the folder `nav_result`. The remain details of `ppp.ipynb` is annotated in the original Notebook file. Please check them to get point to point explanation.

3.3 Visualizing Easy4PPP products in `nav_result.ipynb`

If users get the solution logs from `sppp.py` or `ppp.ipynb` or anywhere else, it is capable for users to visualize them through `matplotlib`. Ensure that `matplotlib` is available in your virtual environment. If not, please run the following command:

```
pip install matplotlib
```

Open the `nav_result.ipynb` in Jupyter Notebook or another code editors (for example, Microsoft Visual Studio Code) and run all the blocks. Users will get the PPP convergence curve, receiver clocks, zenith tropospheric delay, slant total electron content and satellite elevation distribution of the sample data as shown in Figure 3.2. The remain details of `nav_result.ipynb` is annotated in the original Notebook file. Please check them to get point to point explanation.

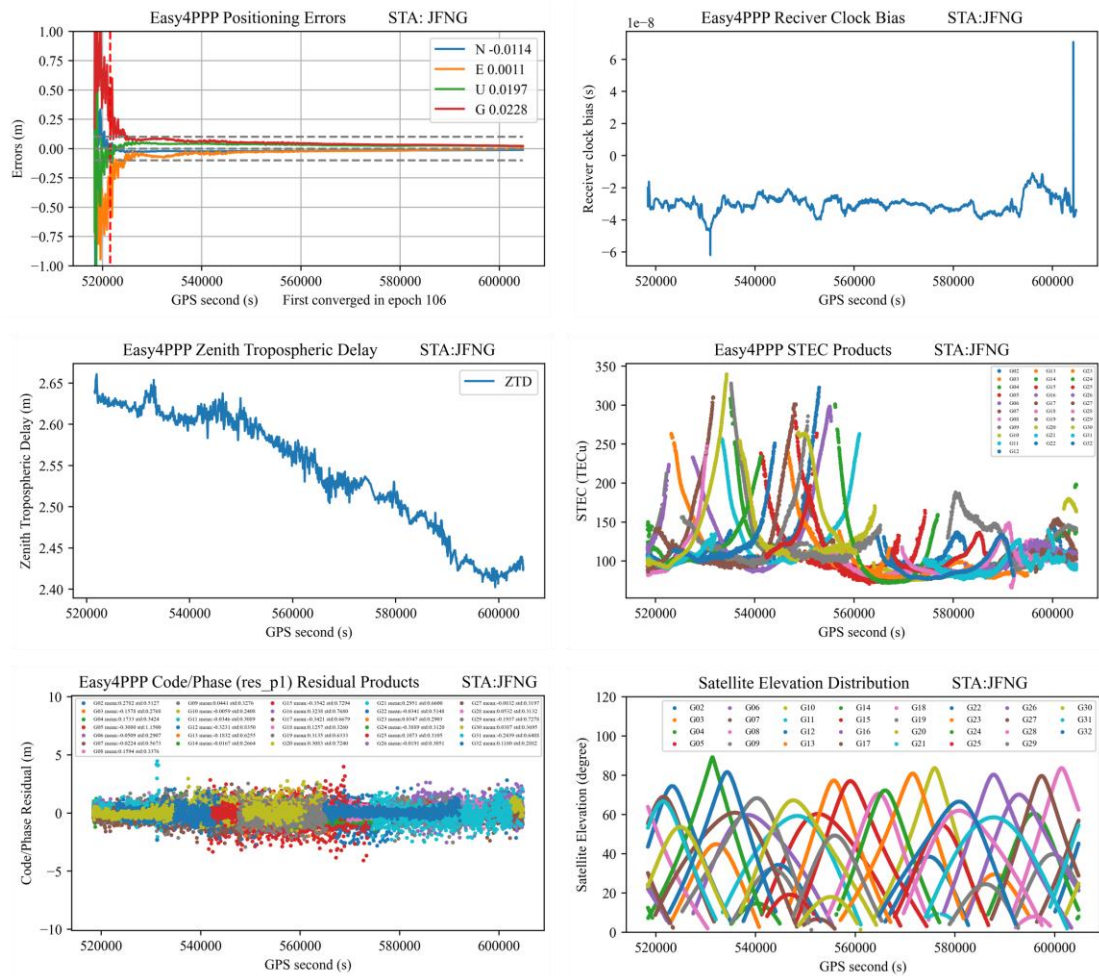


Figure 3.2 Visualization of the sample data by nav_result.iptnb

4. The GUI software of Easy4PPP

Easy4PPP provides a simple Graphical User Interface (GUI) software demo based on its core computation library and PyQt. If Users from non-GNSS background and only need to use PPP positioning results or products such as ZTD or STEC, we highly recommend the Easy4PPP_GUI. **Easy4PPP_GUI is an executable software in Windows and can be run directly without any Python dependencies.** The only operations required are using the keyboard to input or the mouse to select data file paths and PPP configurations. Easy4PPP_GUI.exe is now available at

<https://github.com/alexanderjiang/Easy4PPP/releases/tag/V0.1.0>.

Two versions are provided for different debug information levels. One is Easy4PPP_GUI with console and another is Easy4PPP_GUI without console.

After downloading from github, you will get a single executable program file (Easy4PPP_GUI.exe or Easy4PPP_GCUI.exe). Double-click the program, and the mainframe will appear as shown in Figure 4.1.

First, Select observation and IGS product files by clicking or keyboard inputting. Second, check the solution configs according to your requirements. At last, click the “solve” button, Easy4PPP_GUI or Easy4PPP_CGUI will start computation automatically as shown in Figure 4.2. The “Example” button is used to add the sample data quickly while the “data” folder is in the same path of software. The “reset” button is used to

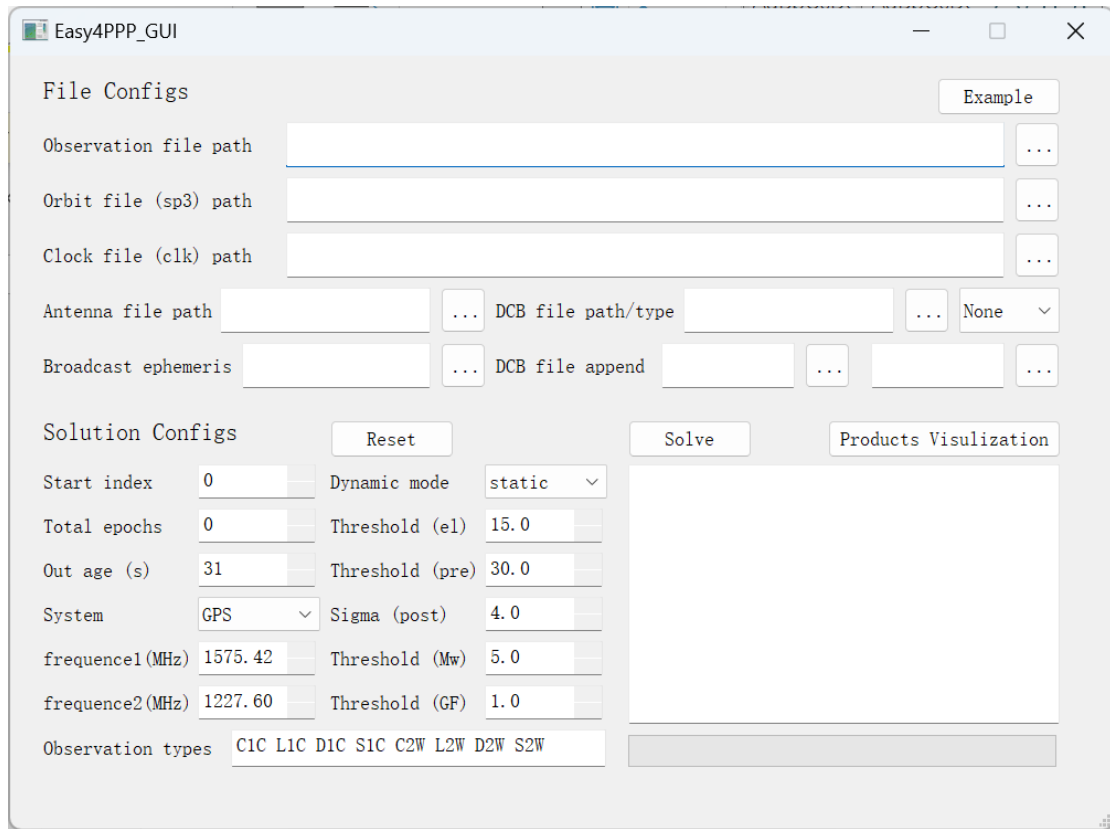


Figure 4.1 The mainfram of Easy4PPP_GUI

initial all the solution configs. The exact meaning of those file configs and solution configs is as same as that mentioned in Table 3.1 and 3.2, Section 3.1. The “Products Visualization” button is used to plot PPP results and products as shown in Figure 4.3. Make sure that the IGS SNX is selected correctly before get the final visualization figures. If users choose Easy4PPP_CGUI.exe, a console will be shown in the meantime and debug info during the computation will be displayed on it. Figure 4.4 shows how Easy4PPP_CGUI works.

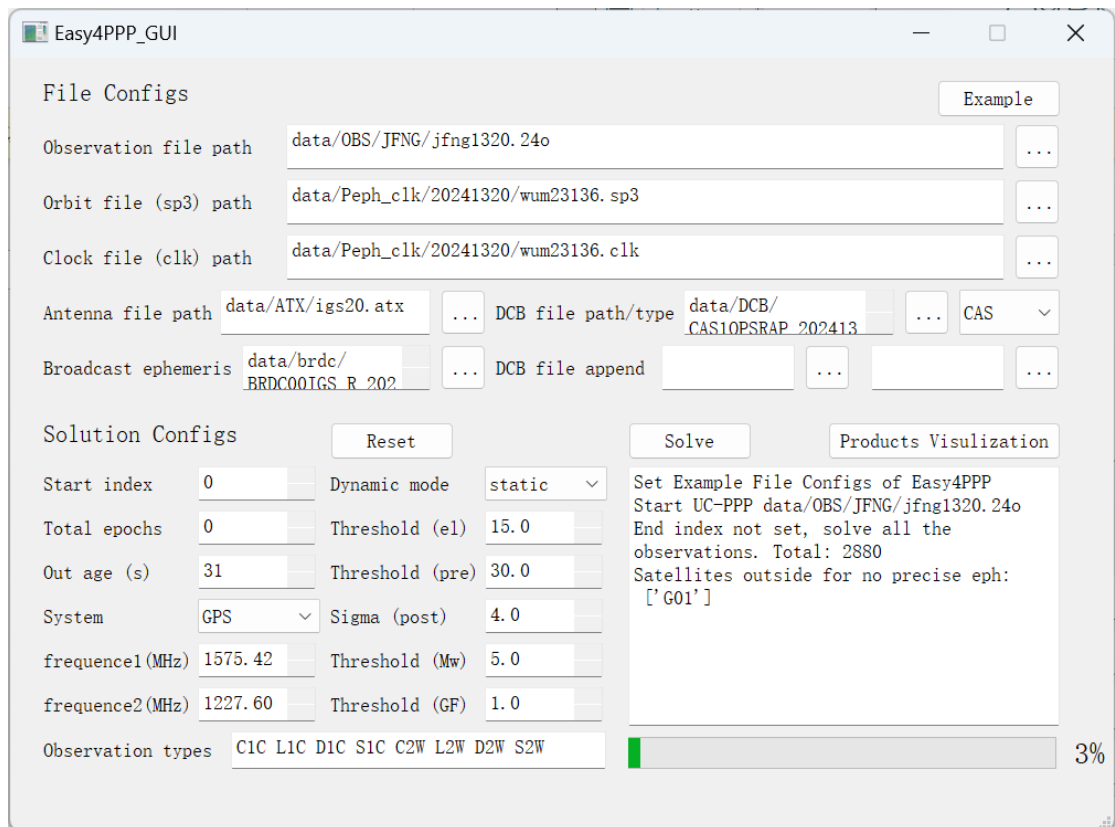


Figure 4.2 The computation stage of Easy4PPP

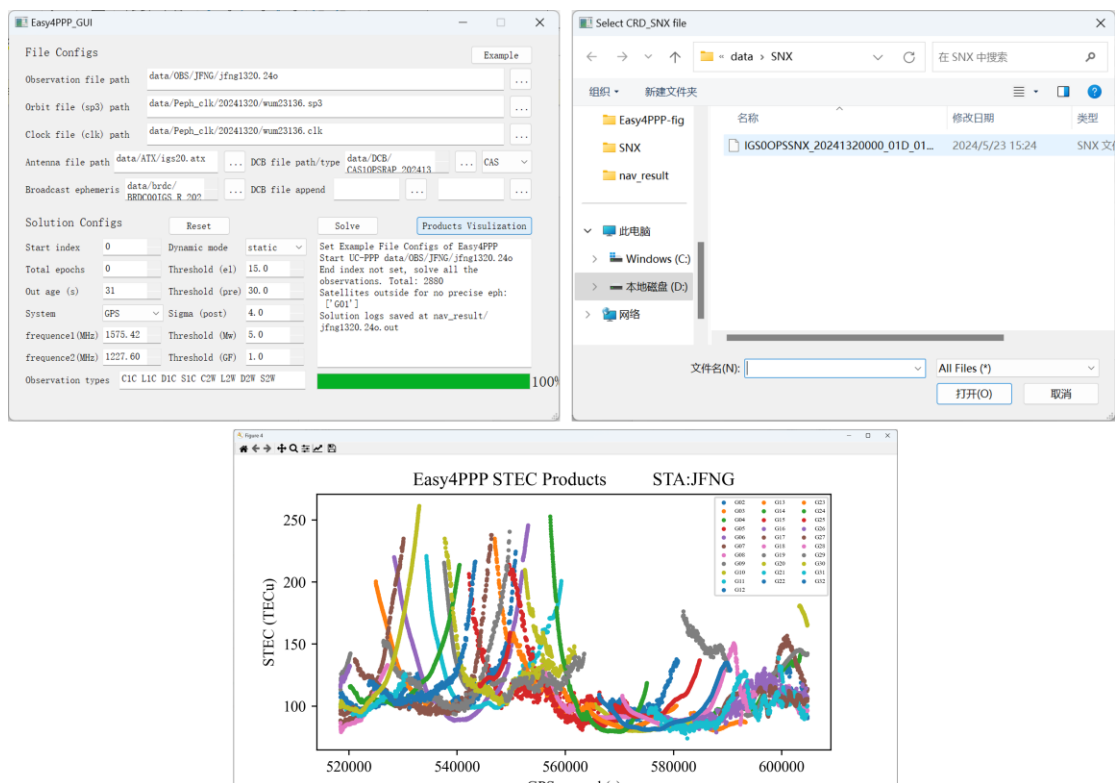


Figure 4.3 The results of clicked “Products Visualization”

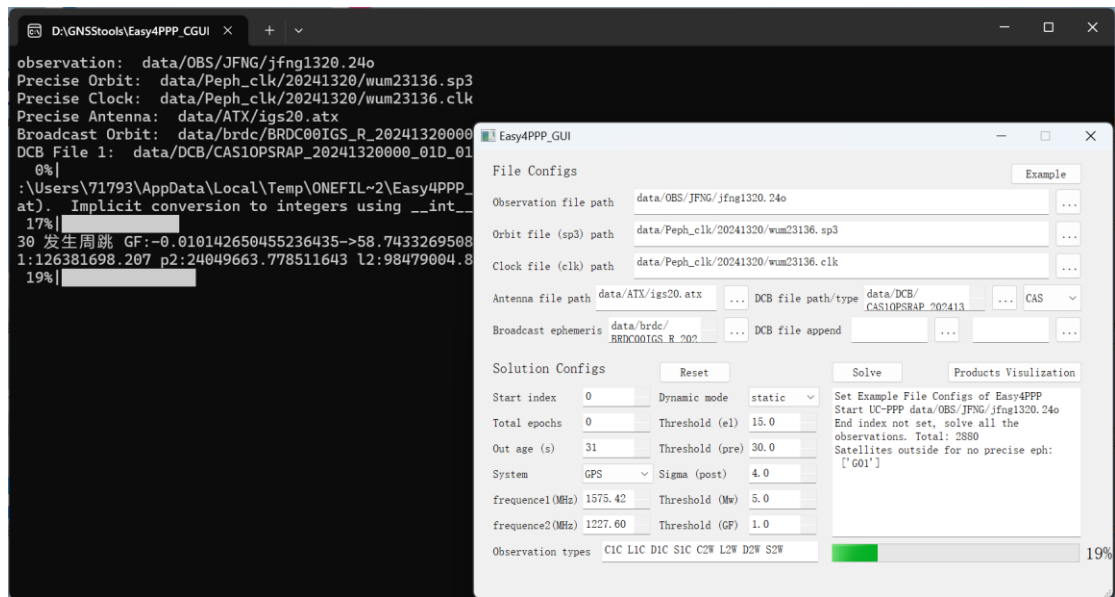


Figure 4.4 Running the Easy4PPP_CGUI with console and debug information