Australian Government
Geoscience Australia

Ginan Workshop – Running Ginan

1. Introduction

This document runs through the steps that we will be taking to produce a static position based on a given RINEX file using Ginan – Geoscience Australia's (GA) in-house and open source GNSS processing software.

The data we will be using comes from one of the Continuously Operating Reference Stations (CORS) in the Geoscience Australia network – specifically the ALIC00AUS station in Alice Springs, NT.

A python script will be initially used to gather all necessary prerequisite reference models, precise orbital and clock information as well as the RINEX data itself (downloaded from GA's GNSS Data Repository) for the day of interest – in our case randomly chosen as 6 Jan 2024.

This will be followed by a run-through of the YAML configuration file, including a short explanation of the various blocks and a walk-through of the edits participants will need to make to this file in order to run our PPP example.

Lastly, we will run Ginan to process and determine its location and display the resultant GPX file via a free online viewer. We will also showcase some of the additional features of the Ginan toolkit but these will not be part of the this walkthrough.

2. Gathering Data

In this section, we will go through the steps necessary to download the prerequisite files for running Ginan as well as the RINEX data itself.

Luckily, Ginan includes a handy Python script that downloads all these files automatically, given a start and end date/time and a location to place the files in.

We assume that you have followed all steps in the Set-Up document that preceded this one – ***Ginan Workshop Set-Up Guide – IGNSS2024 – 6 Feb***, have your Terminal open and are in the *ginanworkshop* directory in your preferred location.

The next step is to run the *auto_download_PPP.py* python script to download the files. In your terminal window, run the following:

```
python3 auto_download_PPP.py --target-dir products --preset igs-station --station-list ALIC --start-datetime 2024-01-06_00:00:00 --end-datetime 2024-01-06_23:59:30 --product-dir products --rinex-data-dir data
```

This tells the script to download files to the target directory "products":

```
--target-dir products
```

Using a preset of the function called "igs-station" which grabs CORS station data for the station you specify:

```
--preset igs-station --station-list ALIC
```

For the time period from midnight 6 Jan 2024 to just before midnight the next day (effectively daily data):

```
--start-datetime 2024-01-06_00:00:00 --end-datetime 2024-01-06_23:59:30
```

An additional option has been used here to split up where RINEX data, and reference products are saved. Namely, we assign the directory "products" for all reference material (with the *--product-dir* flag / option) and "data" for RINEX data specifically (with the *--rinex-data-dir* flag / option):

```
--product-dir products --rinex-data-dir data
```

There are further options that can be manually assigned, but these are all that we will need for this workshop. For further details, you can read into the options via the "*--help*" flag / option.

3. Editing the Configuration YAML file

The next step will be to edit the configuration file we downloaded to match the data we want to process. This will involve editing certain fields within the "ppp_example.yaml" file in your preferred text editor. Therefore:

Step 1 is to Open *ppp_example.yaml* in your preferred text editor

The inputs in the first 16 lines are already assigned correctly (as we are using the same directory structure).

Step 2 is to comment out the inputs that do now align with our example here. This will include:

- the weekly SNX file under *snx_files*
- the CLK, BSX and SP3 files under *satellite_data*
- the RINEX files under *gnss_observations:rnx_inputs*

Step 3 is to Uncomment the wildcard inputs that we do want Ginan to process. Very similar to Step 2, this will be:

- the "*.SNX" wildcard under *snx_files*
- the "*.CLK", "*.BSX" and "*.SP3" wildcards as well as uncommented *nav_files* and its ".rnx" wildcard from the *satellite_data* field
- the ".rnx" wildcard for RINEX data files under *gnss_observations:rnx_inputs*

This covers the *inputs:* section of the file, which is followed by the *outputs:* This is where we tell Ginan what files to produce as outputs and where to place them.

Step 4 is to turn off Galileo constellation processing for now by setting "process:" to false, which is located under *processing_options: gnss_general: gal:*. In this step also, we can verify our code priorities.

As default, the *code_priorities* section for the GPS constellation is [L1W, L1C, L2W]. Ginan will attempt to process all these codes, so it is best to keep it short and process only those codes that are necessary.

- An important point is to verify that your RINEX file has the data for the codes you specified in your YAML.

We can see that in the ALIC00AUS RINEX data that we downloaded, it contains the L1C and L2W codes but not the L1W. Therefore, this will process the L1C and L2W codes in our run. When using your own RINEX files, please ensure that you list codes that are present in your file – if the code is not present, it will not be processed.

A note about processing positions as we are making our way down the file, under *estimation_parameters: receivers: global: pos:* we have the settings for the processed noise and sigma values for the Position. These can be altered to achieve better results when running with kinematic rather than static data. The process noise approximates the motion of the receiver and therefore higher values imply greater speeds.

The **Last Step** is to set *mongo: enable:* to "none" as we will not be using a Mongo Database to store our results in – rather they will only come out as files.

The Mongo Database is required if you wish to use the Ginan – Exploratory Data Analysis (EDA) utility. We will showcase this feature during the workshop but do not provide a walkthrough of getting it up and running. Please ask us for further information later.

Once the last step is complete, we are ready to run Ginan.

4. Running Ginan

Now that we have our input files and have edited the configuration YAML file to tell Ginan what to do, we can run the command for it to start processing our data. The command for this is simply:

```
./Ginan-x86_64.AppImage --config ppp_example.yaml &>stdout.txt
```

In the command above, we are using the Ginan AppImage to run Ginan, giving the option "--config" to specify the path to the configuration file it should be running with. Lastly, we are piping the results of what comes to screen to a file for our use later. This last option can be omitted if you want to read print statements straight on screen.

Once complete, all results of the run will be in the "outputs" directory – the one we specified in the YAML. If we go into that directory, we can see TRACE files and GPX files.

The GPX file format is a readily accessible way to share GNSS coordinate data. There are a number of online viewers for this format and that is what we will use here to see our results

5. Viewing Results

We have found the following website to provide adequate plotting and information on GPX files:

https://www.maplorer.com/view_gpx.html

We will use this to view our results.

- Click on "Choose File" and browse to the relevant location. Choose the .GPX_smoothed if possible.
- Click the "View Profile" button at the bottom.
- A new window will pop up with a map of the results.

We can zoom in and out on this map to get an idea how the processing job went. A bit of data processing is required before we zero in on an accurate result, but generally after a few hours worth of data, the results will settle down to a stable position.