# User Manual

## NSRS ALIGN (Beta)
## (aka ORGN Align)

**Development Team:**

William Ohene, Chase Simpson , Brian Weaver, Hunter Mitchell

# Disclaimer

"THIS SOFTWARE IS PROVIDED 'AS IS', AND THE AUTHOR DISCLAIMS ALL WARRANTIES, INCLUDING IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER, RESULTING FROM LOSS OF USE, DATA, OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE, OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE."

# 1. Introduction

NSRS Align is a user-friendly desktop application designed to streamline the process of aligning the Oregon Real-Time GNSS Network (ORGN) with the National Spatial Reference System (NSRS).The application provides a structured, automated workflow that replaces manual procedures and simplifies interactions with RINEX data sources and OPUS Projects. It is designed specifically for use by RTN managers, with a focus on reliability, clarity, and operational efficiency.

The application is divided into three core modules:

1. Baseline Generator
2. RINEX Download
3. OPUS Project Adjustment

Each module corresponds to a specific stage of the workflow and is accessible through a dedicated tab within the graphical user interface. This manual provides detailed instructions for using each module, including file requirements, parameter configurations, and expected outputs.

# 2. Getting Started

Follow the steps below to launch NSRS Align:

1. **Open the Application Directory:**
   Navigate to the ***nsrs_align*** directory on your computer. This folder contains approximately 131 items, including scripts, dependencies, and support files required by the application.

2. **Locate the Executable File:**
   Inside the folder, find the file named **nsrs_align.exe**.

3. **Launch the Application:**
   Double-click nsrs_align.exe to start the program. After a few seconds, the graphical user interface (GUI) will appear, presenting three main tabs:
   - Baseline Generator
   - RINEX Download
   - OPUS Project Adjustment.

   Once the application is open, you can proceed to work within any of the core modules described in the following sections.

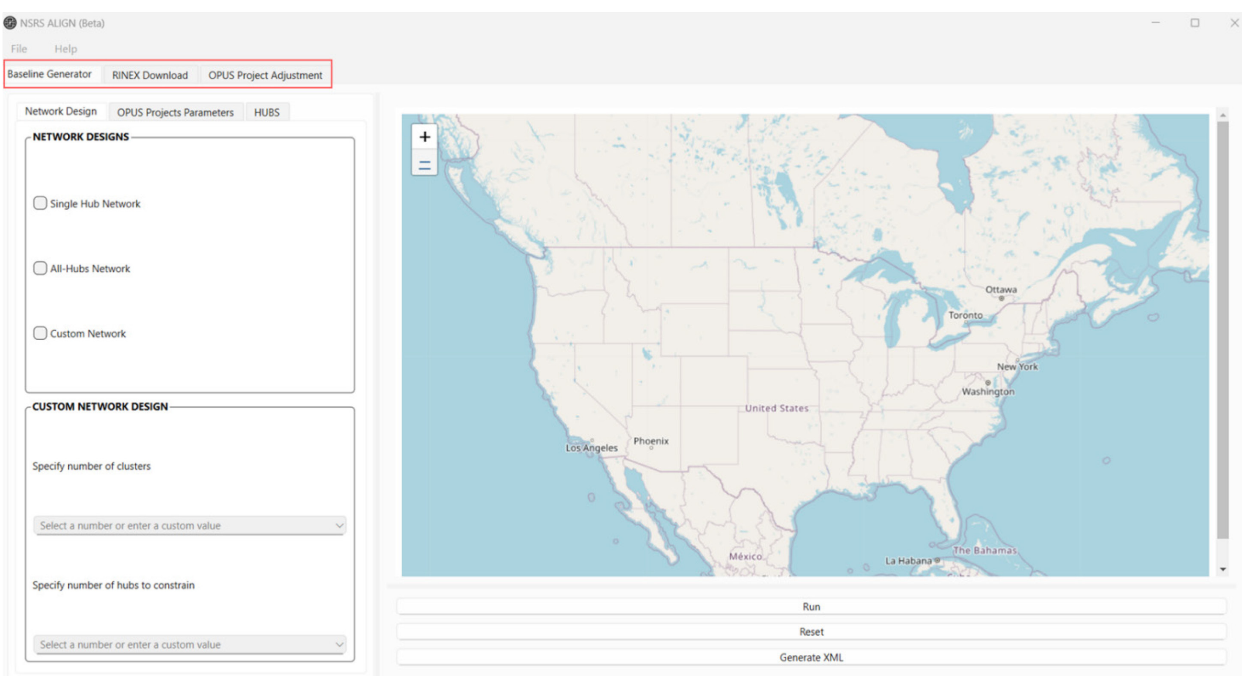
# 3. Module Overview

NSRS Align is organized into three primary modules, each responsible for a specific stage in the workflow for aligning the Oregon Real-Time GNSS Network (ORGN) to the National Spatial Reference System (NSRS). This section provides a high-level summary of each module's function and purpose.

1. **Baseline Generator:**
   This module allows users to define and visualize GNSS baseline configurations outside the OPUS Projects interface. It supports multiple network baseline designs, such as single hub, custom hubs, and all hubs. Users can review and modify the design before adjustment. The selected configuration is used during session and network processing.

2. **RINEX Download:**
   This section provides an interface for acquiring RINEX observation files required for GNSS processing. Users can manually enter station codes or load them from a predefined CSV file. It supports different data sources for stations in the Oregon Real-Time GNSS Network (ORGN) and allows users to download daily data or data across a custom date range.

3. **OPUS Project Adjustment:**
   This module automates the alignment of GNSS stations with the National Spatial Reference System using OPUS Projects. It replaces manual use of the OPUS Projects web interface by handling project creation, RINEX upload, session processing, and network adjustments programmatically.



# 4. General Workflow

NSRS Align follows a clear three-step process that guides users through defining their network, downloading required RINEX data, and performing automated or manual OPUS adjustments. This section provides a high-level overview of how the system works and what each module does. Detailed instructions and examples will follow in later sections.

**Overview of the Workflow**

The workflow consists of three main steps:

1. **Configure your network design:**
   Performed in the **Baseline Generator** tab.

   This step involves entering approximate coordinates for stations in the network and designating which are NGS CORS and which are RTN stations. A baseline configuration is then defined by selecting from several available network designs. OPUS Projects processing parameters and the associated user email are also specified here.

   Baseline configuration is typically a one-time setup and only needs to be updated if the network design changes, stations are added or excluded, or OPUS processing settings are modified.

2. **Download RINEX data:**
   Performed in the **RINEX Download** tab**.**

   An input file listing the network stations and their RINEX data sources is provided. This file enables the system to perform daily automated downloads of observation files required for processing.

   In addition to automated downloads, the module supports manual retrieval of data for specific dates or custom ranges. This is useful for reprocessing historical data. Manual downloads can be applied to the full network or to specific stations, either by uploading a file or by entering station codes and data sources directly into the interface.

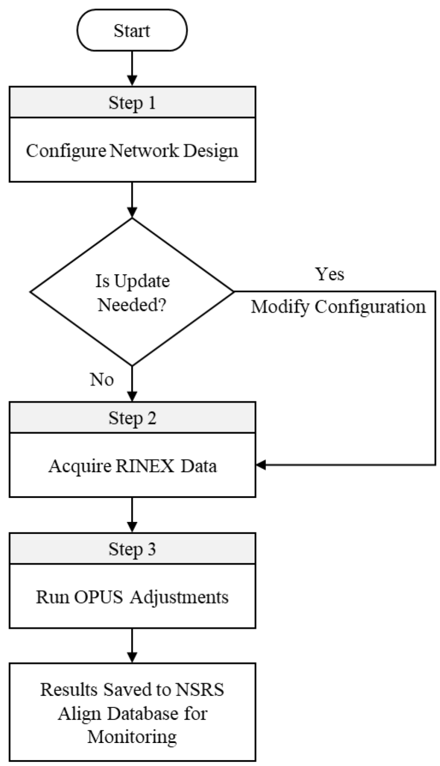
3. **Run OPUS adjustments:**
   Performed in the **OPUS Project Adjustment** tab.

   This step uses the baseline configuration from Step 1 and the downloaded RINEX data from Step 2 to run OPUS Projects adjustments. The system automates project creation, data upload, session processing, and network adjustment using the current configuration and available data.

Adjustments are executed automatically each day, but manual processing is also supported. Manual runs must match the established baseline design and require that all necessary RINEX data is already downloaded and correctly structured. If data is sourced outside the application, the directory format must match the system's expected layout for the adjustment to proceed.

Each step corresponds to one of the three modules in the NSRS Align interface. The steps are connected and must follow a specific order, especially if you're performing manual operations outside the automated workflow.

**NSRS Align Workflow Diagram**



# 5. Interface Overview

## A. Baseline Generator

The Baseline Generator tab is where users define the GNSS baseline design for their network and specify OPUS Projects processing parameters. This replaces the need to manually configure baseline designs and project settings through the OPUS Projects web interface.

The configuration produced here serves as the foundation for all OPUS adjustments, both automated and manual. Users can select from multiple baseline network designs, configure GNSS processing parameters, and manage which NGS CORS stations are used as hubs in the adjustment.

This setup typically only needs to be done once, unless the network layout changes, stations are added or removed, or the OPUS Projects configuration needs to be updated.

**Input File Format**

To begin building a baseline design, load an Excel file that includes all ORGN and NGS CORS stations. The file must contain the following columns:

| Site Code | Lat dec | Long dec | Elip ht | NGS CORS |
|---|---|---|---|---|
| ADEL | 42.17650546 | -119.8958509 | 1386.097 | FALSE |
| CABL | 42.83609529 | -124.5633295 | 38.207 | TRUE |

- Latitude and longitude must be in decimal degrees.
- Elevation must be in ellipsoidal height (meters).
- The NGS CORS column should use TRUE or FALSE to indicate control stations.

*To support map visualization using Folium, all longitudes must include the correct sign. Since the network is located in Oregon, longitudes should be entered as negative values to correctly represent the western hemisphere (e.g., –119.8958509).*

*Approximate coordinates are sufficient. These values do not need to be updated regularly and are used mainly for map rendering. Station roles (NGS CORS vs. RTN) and inclusion in adjustments can be modified at any time without updating coordinates.*

To load the input file:

- Click File in the top menu bar.
- Select Open.
- Choose the Excel file in the required format.

**Baseline Generator Panel Overview**

The Baseline Generator interface is divided into four panels. Each panel corresponds to a stage in the configuration process.

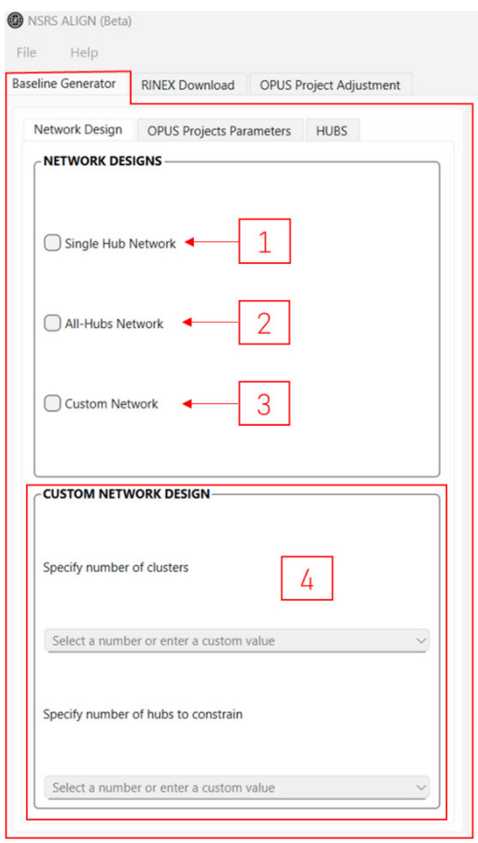
1. **Network Design**

This panel allows the user to select from a list of predefined baseline network designs or build a custom layout. Each design type defines how RTN CORS will be connected to NGS CORS hubs.

Table 3: Network Design Options

| No. | Baseline Design | Description |
|---|---|---|
| 1 | **Single Hub Network** | All RTN CORS connect to a single NGS CORS |
| 2 | **All Hubs Network** | RTN CORS connect to all available NGS CORS |
| 3 | **Custom Network** | Users manually define which NGS CORS to connect RTN CORS to. |
| 4 | **Custom Network Design** | Allows the user to define the number of network clusters and number of NGS CORS to be constrained within each cluster. |

When a design is selected, a preview will be displayed on the map.

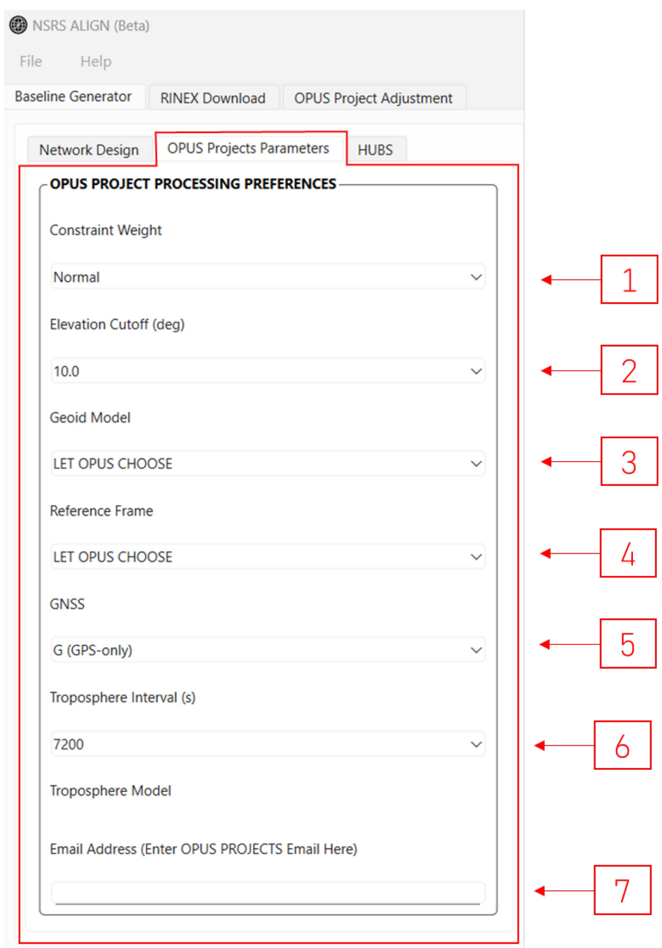


## 2. OPUS Project Parameters

This panel allows configuration of GNSS processing parameters that would otherwise be set manually within the OPUS Projects web interface. These values are embedded directly into the output XML used for processing.

| No. | Parameter | Description |
|---|---|---|
| 1 | **Constraint Weight** | Controls how tightly stations are constrained. Default: Normal. |
| 2 | **Elevation Cutoff (deg)** | Minimum satellite elevation for inclusion. Default: 10°. |
| 3 | **Geoid Model** | Vertical model used. Default: Let OPUS choose. |
| 4 | **Reference Frame** | Horizontal datum (e.g., NAD83). Default: Let OPUS choose. |
| 5 | **GNSS** | System used in processing. Default: GPS. |
| 6 | **Troposphere Interval** | Interval for estimating atmospheric delay. Default: 7200s. |
| 7 | **Email Address** | Required email address for OPUS Project submissions. |

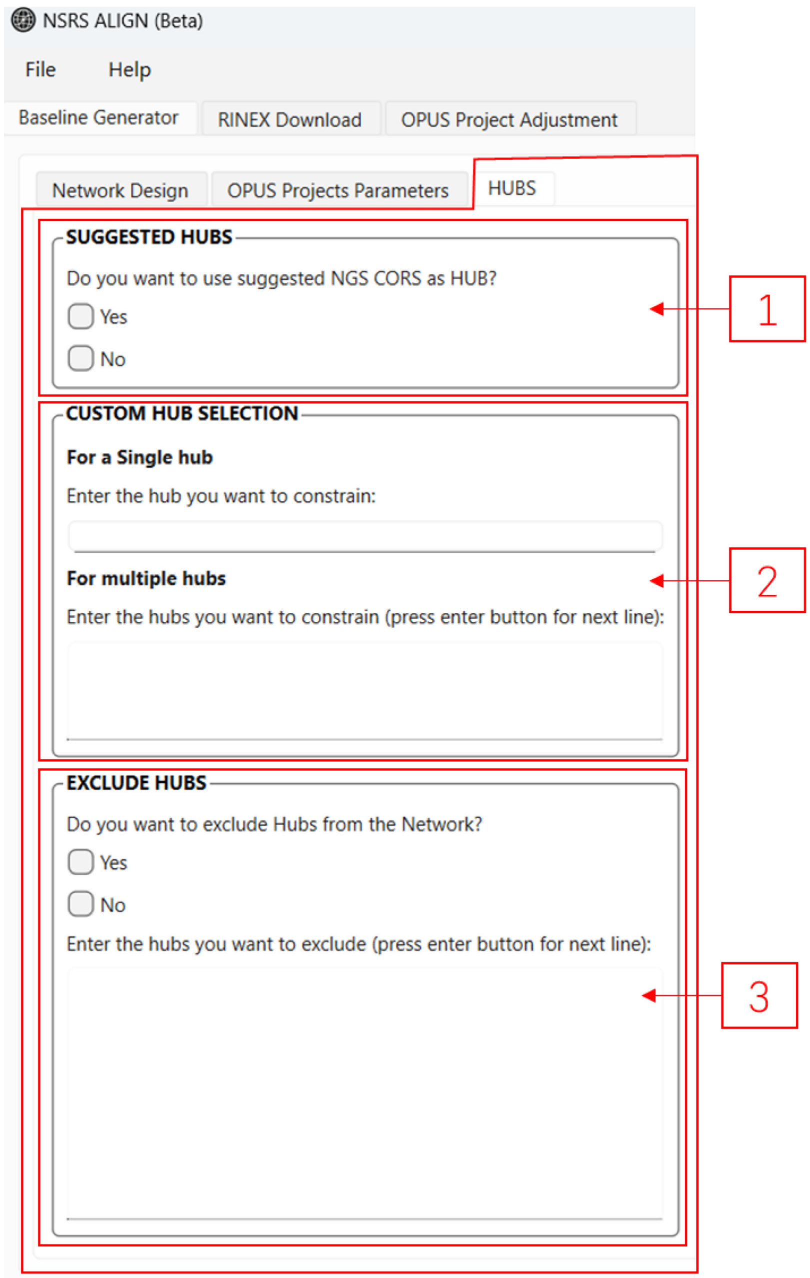
Default values are provided and can be edited as needed.



**3. HUBS Selection**

This panel allows users to manage which NGS CORS stations are included as hubs in the network baseline design.

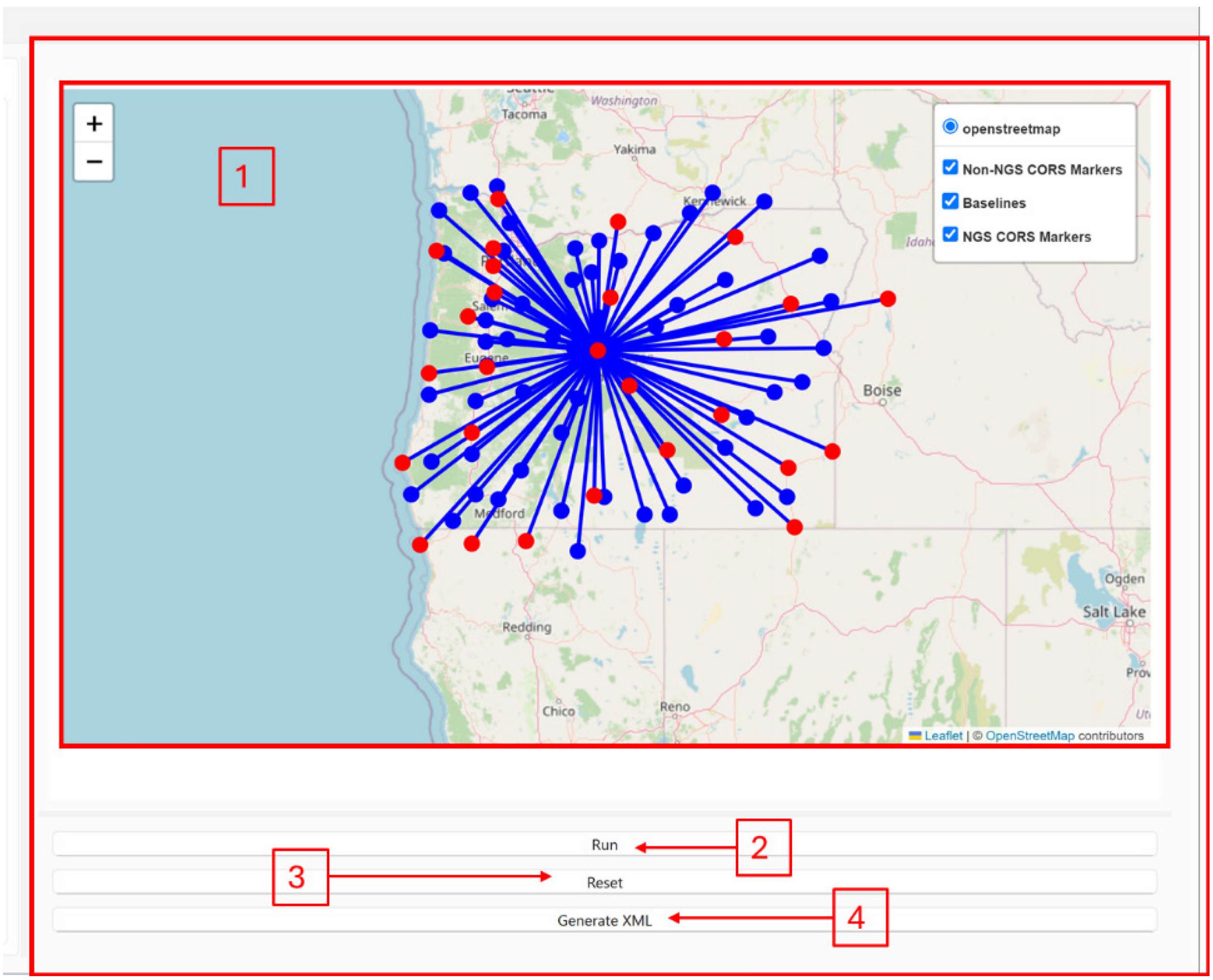
| No. | Option | Description |
|---|---|---|
| 1 | **Suggested CORS** | Uses most central NGS CORS as hubs. |
| 2 | **Custom Hubs Selection** | Manually specify which NGS CORS to use as hubs. |
| 3 | **Exclude Hubs** | Optionally remove any included NGS CORS from the design. |



### 4. Finalization and Visualization:

Once all design settings have been specified across the configuration panels, a set of tools is available for finalizing and visualizing the baseline network. Use these tools to update and verify your design before exporting the finalized configuration as an XML file for later processing.

| No. | Tool | Function |
|---|---|---|
| 1 | **Map** | Displays the current network baseline layout |
| 2 | **Run** | Generates or updates the map after design changes |
| 3 | **Reset** | Clears all inputs and prepares for a new configuration |
| 4 | **Generate XML** | Exports the complete configuration for later use in OPUS adjustment processing |



## Output XML File

After the baseline design has been finalized, , click **Generate XML** to export a configuration file. This file contains all the network design baselines and configuration parameters required for downstream OPUS Projects adjustment processing. It is automatically formatted and saved to the appropriate output directory. No manual editing is required. An example of a generated XML file structure is shown in below.

```xml
<?xml version="1.0" encoding="utf-8"?>
<OPTIONS>
  <BASELINES>
    <DISTANCE><FROM>REDM</FROM><TO>ADEL</TO></DISTANCE>
    <DISTANCE><FROM>REDM</FROM><TO>basq</TO></DISTANCE>
    <!-- Additional baseline connections -->
  </BASELINES>
  <CONSTRAINT_WEIGHT>Tight</CONSTRAINT_WEIGHT>
  <ELEVATION_CUTOFF>15.0</ELEVATION_CUTOFF>
  <EMAIL_ADDRESS>tuffourp@oregonstate.edu</EMAIL_ADDRESS>
  <GEOID_MODEL>LET OPUS CHOOSE</GEOID_MODEL>
  <REFERENCE_FRAME>NAD_83(2011)</REFERENCE_FRAME>
  <GNSS>GNSS</GNSS>
  <TROPO_INTERVAL>7200</TROPO_INTERVAL>
  <TROPO_MODEL>Piecewise Linear</TROPO_MODEL>
  <CORS>
    <HUB>REDM<FIX>3-D</FIX></HUB>
    <HUB>JIME<FIX>NONE</FIX></HUB>
    <!-- Additional hubs -->
  </CORS>
</OPTIONS>
```

## B. RINEX Download

The RINEX Download tab provides functionality for downloading RINEX files for stations in the ORGN. Users can specify date ranges and select stations manually or through a predefined CSV file. The module supports multiple data sources relevant to the Oregon Real-Time GNSS Network.

To enable automated daily downloads, users must first upload a station list in the required CSV format through the interface. After upload, the file is saved to the stations directory and used as the default input for scheduled processing. Manual one-time downloads are also supported through direct CSV upload or by entering stations in the interface.

All RINEX files saved in the processed directory structure are automatically included in OPUS automated adjustments. The station list does not need to be re-uploaded unless stations are added or removed or if the data source assignments need to be changed.

**Input File Format**

The input is a CSV file that specifies the list of ORGN stations and their associated RINEX data sources. The required format depends on whether the network is processed as a single region or as multiple regions.

For a single unified network, the CSV must contain two columns:

| Station | Source |
|---|---|
| AGNS | gage |
| BURN | gage |
| ANAT | geodesy |
| ADEL | ftp |

If the real-time network is processed as separate regions, a third column must be included to assign each station to a region. The required format in that case is:

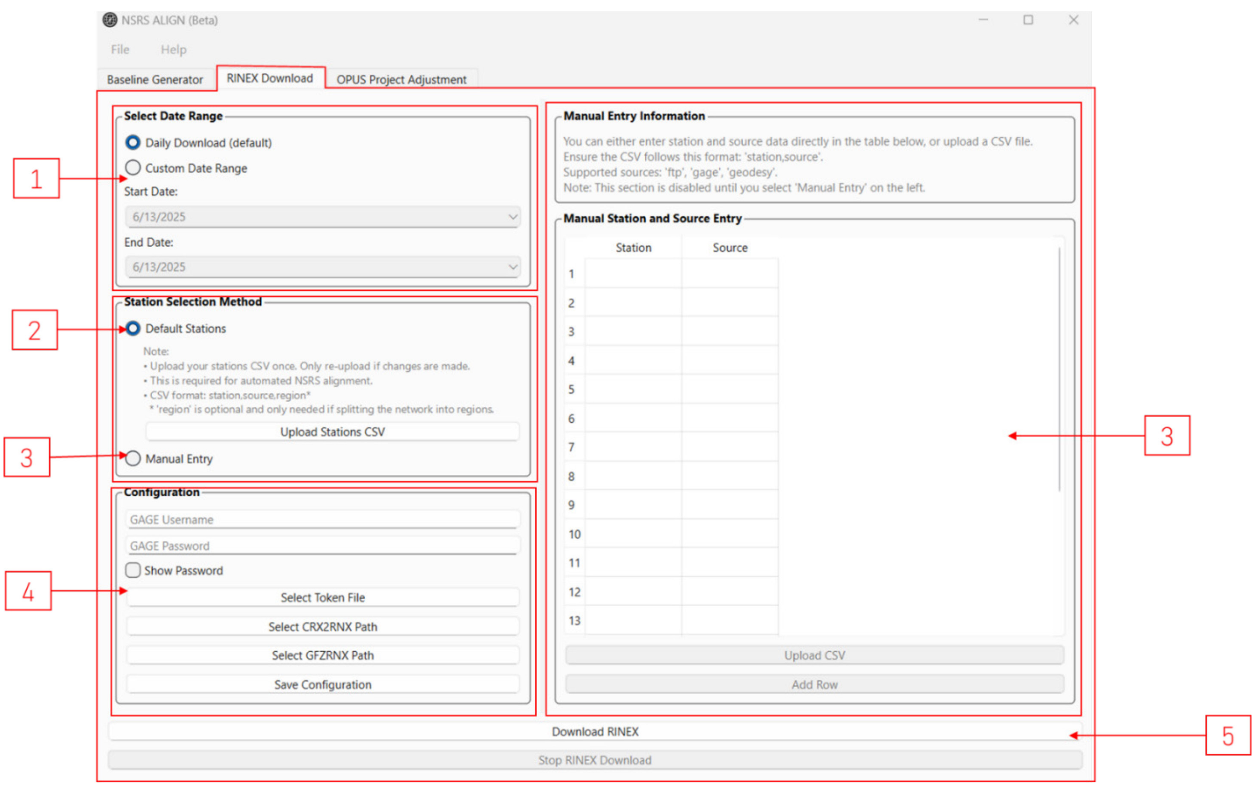
| Station | Source | Region |
|---|---|---|
| ADEL | ftp | E |
| BASQ | ftp | E |
| COBO | ftp | W |
| CRAN | ftp | E |

The following data sources are supported:

- **ftp** for the Oregon Department of Transportation FTP server.
- **gage** for the EarthScope GAGE data archive.
- **geodesy** for the Pacific Northwest Geodetic Array (PANGA) archive.

**Functional Components:**

| No. | Component | Description |
|---|---|---|
| 1 | **Date Selection** | Specify a date range for download. For a single day, the start and end dates must be the same. |
| 2 | **Station Selection Method** | Upload a CSV to define the station list used for automated daily downloads. This file is saved to the stations directory and remains active until replaced. |
| 3 | **Manual Entry** | Used for one-time downloads outside the automated workflow. Allows direct entry or CSV upload of a specific subset of stations and data sources. |
| 4 | **Configuration** | Provide GAGE token credentials and specify the full paths to the crx2rnx and GFZRNX executables. |
| 5 | **Download RINEX** | Start the download using the selected station list, date range, and source definitions. |



**GAGE Download Token Setup**

Some RINEX files, particularly from the EarthScope GAGE archive, require authenticated access. NSRS Align supports this using an OAuth 2.0 Device Flow. This setup is a one-time process per user. Once complete, the generated token will be automatically used for future downloads.

1. **Requirements**

- A valid EarthScope GAGE account
- Python installed on your system

2. **Account Setup**

If you already have a GAGE account:

- Go to https://www.earthscope.org
- Click Login in the top right
- Sign in with your credentials

To create a new account:

- Visit https://www.earthscope.org
- Click Login then Sign up
- Complete the registration process

3**. Installation**

Install the required packages using pip:

```
pip install earthscope-sdk requests
```

4. **Generate Token**

Create a script named ***generate_gage_token.py*** with the following content:

```
"""
EarthScope (GAGE) Authentication Token Generator
Implements OAuth 2.0 Device Flow for persistent GAGE data access.
"""

from pathlib import Path
from earthscope_sdk.auth.device_code_flow import DeviceCodeFlowSimple

TOKEN_PATH = Path.cwd() / "sso_tokens.json"

def run_device_login():
    print("\n=== EarthScope Token Setup ===")
    print("Generating secure token for GAGE account.")
    print("Browser authentication required (one-time).\n")
```

```
    try:
        device_flow = DeviceCodeFlowSimple(TOKEN_PATH)
        device_flow.do_flow()
        print(f"\n Success! Token saved to: {TOKEN_PATH}")
        print("GNSS download tools are now ready.")
    except Exception as e:
        print(f"\n Authentication failed: {e}")
        print("Check connection and try again.")

if __name__ == "__main__":
    run_device_login()
```

Then run the script:

```
python generate_gage_token.py
```

Steps:

- The script will display a URL and code
- Open the URL in your browser
- Enter the code and sign in
- A confirmation will appear in the terminal

5. **Token Storage**

- The token is saved to sso_tokens.json in the script directory
- This token is automatically used by NSRS Align for GAGE downloads
- If deleted, the setup must be repeated

For support: contact support@earthscope.org

**RINEX Download Output**

The download process produces two sets of output files, stored in a standardized directory structure.

1. **Original Files**
   - Directory:
     **data/original/<year>/<doy>/<station_id>/**

   - Description:
     Contains the raw RINEX files as retrieved from each source. Files may be in compressed formats such as gz or bz2, or in archive-specific formats such as crx or d.

2. **Processed Files**
   - Directory:
     **data/processed/<year>/<doy>/<station_id>/**

- Description:
  Files are automatically decompressed, converted to standard RINEX format, and resampled to 30 second intervals. These processed files are used directly in OPUS processing.

**All files present in the processed directory are automatically included in OPUS processing.**

**Manual Data Placement Requirements**

If the RINEX Download tab is not used, the required RINEX observation files must be placed manually into the processed directory structure. Files must meet all of the following conditions:

- Located in **data/processed/<year>/<doy>/<station_id>/**
- Contain 24 hour observation data.
- Sampled at 30 second intervals.
- Named and structured consistently with expected formats.

Files that do not meet OPUS input requirements or are not placed in the correct directory structure and format may fail to be processed or may be skipped entirely during the automated adjustment workflow.

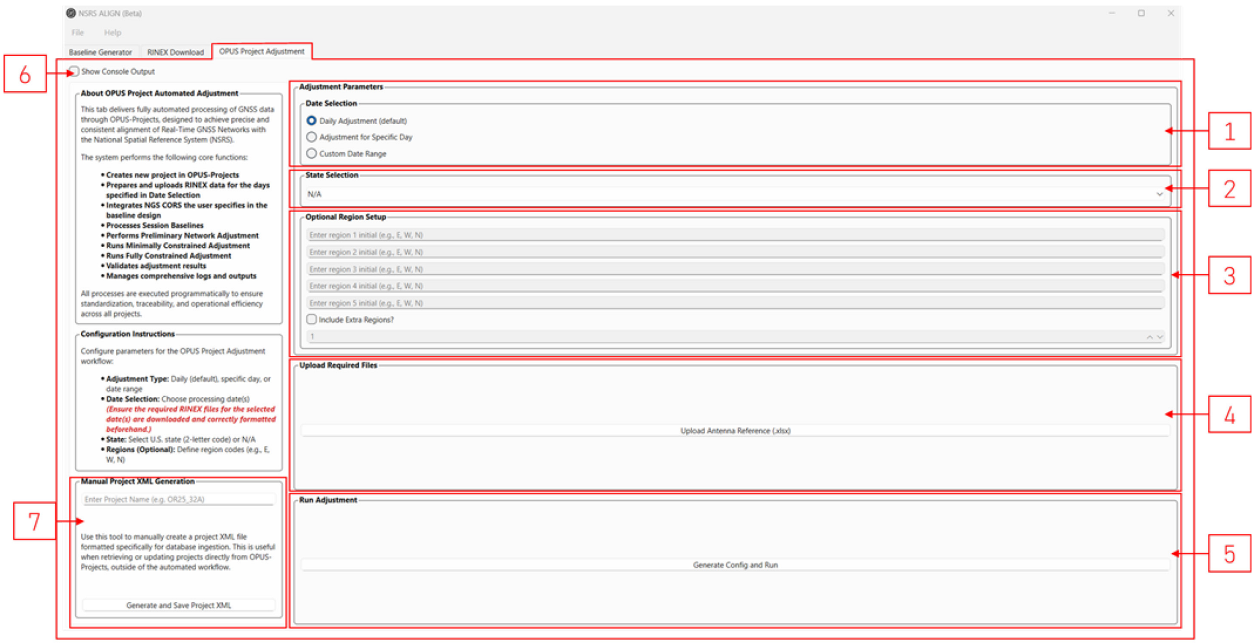
## C. OPUS Projects Adjustment Tab

The OPUS Project Adjustment tab automates the alignment of Oregon Real-Time GNSS Network (ORGN) stations with the National Spatial Reference System (NSRS) using OPUS Projects. It bypasses the manual OPUS Projects web interface by programmatically handling key workflow steps. These include:

- Creating new projects within OPUS Projects.
- Uploading RINEX observation files via OPUS Static.
- Ensuring user-specified NGS CORS are used during adjustment.
- Executing session and network adjustments.

The following components and steps guide users through the full adjustment workflow:

| No. | Component | Description |
|---|---|---|
| 1 | Date Selection | Select daily run, specific day, or custom date range. |
| 2 | State Selection | Select the two-letter abbreviation for the U.S. state. |
| 3 | Regional Setup | Specify custom region codes if processing multiple regions. |
| 4 | Upload Files | Upload required supporting files, including the Antenna Reference File. |
| 5 | Generate config and run | Generate configuration file and start processing. |
| 6 | Show Console Output | Display logs and monitor processing status. |
| 7 | Manual Project XML Generation | Generate XML files for database ingestion after manual OPUS adjustments. |




## 1. Adjustment Parameters Setup

This section allows users to define all configuration parameters necessary for OPUS processing.

### a. Date Selection

Users can choose one of the following modes:

- Daily Adjustment (default): Uses the current date.
- Specific Day: Allows selection of a single day.
- Custom Date Range: Enables selection of a start and end date.

### b. State Selection

Choose the two-letter abbreviation of the U.S. state corresponding to the network. If not selected, a default of "US" is used.

### c. Region Configuration

If the network is divided into regions, users may enter up to five custom region codes (such as E, W, N). This field is disabled unless the “Include Extra Regions?” option is checked.

## 2. Required File Uploads

Certain components of the backend workflow require external files. These must be uploaded before adjustment.

**Antenna Reference File:**

This file contains station-specific antenna types and antenna heights. These values are used during RINEX file uploads to OPUS Static and are treated as the authoritative reference, overriding any incorrect metadata found in the original RINEX files. Users are expected to maintain this file with accurate and up-to-date values. It only needs to be re-uploaded when antenna type or height changes for one or more stations.

**Antenna Reference File Format:**

| Station | Antenna Type | Antenna Height |
|---|---|---|
| ADEL | LEIAT504GG LEIS | 0 |
| AGNS | TWIVC6050 SCIT | 0.0083 |
| ANAT | TRM115000.00 | 0 |
| ARLN | SEPCHOKE_B3E6 | 0 |

The uploaded file is retained and used for all automated processing until replaced.

**3. Running the Adjustment Workflow**

Once parameters are defined and required files are uploaded:

1) Confirm all necessary RINEX files are downloaded and structured correctly under **data/processed/<year>/<doy>/<station_id>/**

2) Click **Generate Config and Run** to create a configuration file and start processing.

3) Select **Show Console Output** to monitor the progress and view real-time processing logs.

**4. Output from Adjustment Processing**

When processing completes, an OPUS Project XML file is automatically generated. This file contains:

- OPUS Static solutions.
- OPUS Projects Session Solutions.
- OPUS Projects Network Solutions.

The XML file produced is specifically formatted for direct ingestion into the *NSRS Align Database*, supporting residual monitoring and time series analyses.

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <PROJECT_INFO VERSION="2.3" MTIME="1733129448" SOURCE="project_info.xml">
    <USER ISID=",,,1" ESID="bcut">
      <PREFERENCES>
        <STATE>CA</STATE>
        <GEOID_MODEL>LET OPUS CHOOSE</GEOID_MODEL>
        <REFERENCE_FRAME>USE PREFERENCES</REFERENCE_FRAME>
        <SPCS_UNIT>USE PREFERENCES</SPCS_UNIT>
        <SPCS_ZONE>USE PREFERENCES</SPCS_ZONE>
      </PREFERENCES>
      <POSITION>
        <EPOCH SYSTEM="GPS">2022.84789999</EPOCH>
        <REF_FRAME>ITRF2014</REF_FRAME>
        <COORD_SET>
          <RECT_COORD>
            <COORDINATE UNIT="m" UNCERTAINTY="0.008" AXIS="X">-2708381.439</COORDINATE>
            <COORDINATE UNIT="m" UNCERTAINTY="0.007" AXIS="Y">-4002879.226</COORDINATE>
            <COORDINATE UNIT="m" UNCERTAINTY="0.003" AXIS="Z">4148018.068</COORDINATE>
            <UNCERTAINTY UNIT="m" AXIS="X">0.008</UNCERTAINTY>
            <UNCERTAINTY UNIT="m" AXIS="Y">0.007</UNCERTAINTY>
            <UNCERTAINTY UNIT="m" AXIS="Z">0.003</UNCERTAINTY>
            <VELOCITY UNIT="m/yr" UNCERTAINTY="0.01" AXIS="X">-0.0070</VELOCITY>
            <VELOCITY UNIT="m/yr" UNCERTAINTY="0.01" AXIS="Y">0.0091</VELOCITY>
            <VELOCITY UNIT="m/yr" UNCERTAINTY="0.01" AXIS="Z">0.0040</VELOCITY>
          </RECT_COORD>
          <ELLIP_COORD>
            <COORDINATE UNIT="deg" UNCERTAINTY="0.006" AXIS="LAT">
              <DEGREES>40</DEGREES>
              <MINUTES>49</MINUTES>
              <SECONDS>42.59569</SECONDS>
            </COORDINATE>
            <COORDINATE UNIT="deg" UNCERTAINTY="0.006" AXIS="EAST_LONG">
              <DEGREES>235</DEGREES>
              <MINUTES>55</MINUTES>
              <SECONDS>2.60620</SECONDS>
            </COORDINATE>
            <COORDINATE UNIT="m" UNCERTAINTY="0.006" AXIS="EL_HEIGHT">-19.213</COORDINATE>
            <UNCERTAINTY UNIT="m" AXIS="LAT">0.006</UNCERTAINTY>
            <UNCERTAINTY UNIT="m" AXIS="EAST_LONG">0.006</UNCERTAINTY>
            <UNCERTAINTY UNIT="m" AXIS="EL_HEIGHT">0.006</UNCERTAINTY>
          </ELLIP_COORD>
        </COORD_SET>
        <SOURCE MTIME="1733122055">opusReports/bcut3100.22o.txt coordinates with HTDP(3.5.0) ITRF2014 velocity</SOURCE>
      </POSITION>
      <POSITION>
        <EPOCH SYSTEM="GPS">2010.00000000</EPOCH>
        <REF_FRAME>NAD_83(2011)</REF_FRAME>
        <COORD_SET>
```


## 5. Manual XML Generation (Fallback Option)

If automated adjustment processing is incomplete or unsuccessful, the manual XML generation option ensures that manually adjusted data can still be integrated into the NSRS Align database. Follow these steps:

- Open the OPUS Projects web interface and log in using the automated project name as both the Project Identifier and Manager Keyword. This allows direct access to the correct project.
- Complete the required session and network adjustments manually within the OPUS Projects interface.
- Return to the NSRS Align application and enter the completed project's name into the **Manual Project XML Generation** field.
- Click **Generate and Save XML** to produce an XML file formatted for direct ingestion into the NSRS Align database.

This method guarantees that adjustments performed manually remain fully compatible and integrated within the automated NSRS Align workflow.

# 6. Workflow Walkthrough Using Example Dataset

This section guides users through the complete NSRS Align workflow using a provided example dataset located in the directory named **nsrs align test data**.

Two processing scenarios are possible:

- Unified Network
- Split Network by Region

The unified network scenario is described first. Each step is explained clearly to help users follow along effectively. Ensure the nsrs align test data directory is accessible before proceeding.

### Case 1 : Unified Network Workflow

The unified network workflow involves three sequential steps, in line with the general workflow explained in section 4 of this manual :

- Network Configuration
- RINEX Data Download
- OPUS Adjustments

### Step 1. Configure Network Design

To begin configuring your network:

1) Launch the **NSRS Align** application.

2) In the **Baseline Generator** tab, click File, then Open.

3) Navigate to the **nsrs align test data** directory, open the **single network** subdirectory, and select the file named **orgn_test_data.xlsx**.

This Excel file contains the approximate coordinates and station designations (NGS CORS or RTN stations) required by the application. Refer to Section 5 of this manual for detailed information about the required format.

After loading the input file:

4) In the **Network Design** panel, select a baseline network design (for example, Single Hub Network). Choose the design appropriate for your needs or preferences.

5) Open the **OPUS Project Parameters** panel and review the available settings.
   a. Default values match OPUS Projects requirements; typically, no changes are needed.
   b. Enter your authorized OPUS Projects email address.
      i. **Note**: If you have not completed OPUS Projects training and do not have an authorized email address, refer to the NGS website for training details. You cannot proceed without proper authorization.

6) Click Run to visualize the baseline connections on the map interface.
   a. NGS CORS stations appear in red, while RTN stations appear in blue.

7) Inspect the design. If modifications to the selection of NGS CORS hubs are needed:
   a. Navigate to the **HUBS** panel.
   b. If the suggested NGS CORS hubs are acceptable, leave the default "Yes" selected.
   c. To manually select hubs, choose "No" in response to the prompt asking whether to use suggested hubs.
   d. Specify desired hubs manually or exclude certain hubs by entering their station identifiers.

8) After finalizing the design, click Generate XML.
   a. When prompted with the dialog "Are you splitting your project into regions?", select No since this walkthrough describes the unified network scenario.
   b. Save the resulting XML file in the default directory location:
      *nsrs_align\opus_processing\gui_xml*
   c. The filename can be edited if desired, but the directory must remain unchanged.

**Note**: his XML configuration does not need to be regenerated unless the network design change, stations are added or removed, or the OPUS Projects configuration needs to be updated.

## Step 2. Download RINEX Data

### Automated Workflow Setup

To configure automated daily downloads:

1) In the **RINEX Download tab**, locate the **Station Selection Method** section.

2) With "**Default Stations**" selected (this is the default option), click Upload Stations CSV.

3) Navigate to the **nsrs align test data** directory, enter the single network subdirectory, and open the **stations** folder.

4) Select the provided file named **stations.csv**. After selection, the notification "Stations CSV uploaded and saved" confirms the successful upload.

5) In the Configuration panel, enter your EarthScope (GAGE) username and password. Also, specify the path to your GAGE download token (.Json) file.
    a. If you have not set up a GAGE download token, follow the detailed instructions provided in Section 5B (RINEX Download Tab) of this manual.

6) Specify the full paths for the required executables:
    a. **crx2rnx**: Converts Compact RINEX files to standard RINEX format.
    b. **gfzrnx**: Used for splicing and resampling RINEX.
    c. Click Save Configuration.
    d. These paths need only be configured once unless their file locations change.

7) After completing the above steps, click **Download RINEX** to initiate the download process.

**Manual Method**

For downloading specific historical datasets not included in automated processing:

1) In the **Date Selection** section, select "Custom Date Range" and enter the desired start and end dates.
    a. **Important**: Verify that the start date precedes the end date to avoid errors.

2) Under **Station Selection Method**, choose "**Manual Entry**".

3) Either upload a CSV file containing the stations and data sources or manually enter them directly in the provided interface fields.

4) The interface displays 20 rows by default; use the **Add Row** button if more rows are needed.

5) Complete the **Configuration** panel setup as described in the automated workflow section above.

6) Click **Download RINEX** to initiate the manual download process.

**Important:** If you have already downloaded RINEX files independently and prefer to place them manually instead of using the automated download interface, you must follow specific folder and naming conventions for the system to recognize them correctly.

Refer to Section 5 B for complete instructions on manual data placement, including required directory structure, expected filenames, and how the system processes manually placed data.

**Logs and Reporting**

The RINEX download process generates the following logs for monitoring and troubleshooting:

- download_log.csv (records download details including timestamp, station, source, file name, and status)
- download_log.log (detailed log of file processing including unzipping and conversion operations)
- station_download_summary.csv (concise summary indicating download status per station and date)

These log files are located within **nsrs_align\rinex_download_log** , providing useful diagnostics if issues arise.

## Step 3. Run OPUS Adjustments

This step initiates the automated or manual OPUS Projects adjustment process, based on the configuration and RINEX data prepared in the previous steps. The resulting project name is automatically generated from the parameters selected in this step. Refer to Appendix A for the full project naming logic and how this affects project tracking and file retrieval.

**Automated Processing**

This option supports continuous daily alignment of the RTN stations by automatically uploading the latest RINEX files and performing OPUS Projects adjustments.

1. **Adjustment Parameters Setup**

Begin by setting the adjustment parameters. This is the first panel shown in the OPUS Project Adjustment tab. The following must be configured:

- Date Selection
    - Set to **Daily Adjustment** (this is the default).
    - This mode is required for automated processing. The system will upload one day’s RINEX data per run into OPUS Static.
    - After 7 consecutive daily uploads, session and network processing are automatically triggered.

- State Selection
    - Choose the two-letter abbreviation of the U.S. state corresponding to the RTN.
    - If processing a nationwide network or default setup, enter N/A to represent the entire U.S.

2. **Upload Required Files**

- Navigate to the test dataset directory.
- Inside the **nsrs align test data** folder, open the **antenna reference file** directory.
- Select **test_orgn_antenna_reference_list.xlsx** to upload.
- This file provides verified antenna types and heights used during OPUS Static uploads. It overrides any incorrect metadata in the RINEX headers.
- This file only needs to be re-uploaded if antenna information changes for any station.

3. **Start Processing**

- Click **Generate Config and Run** to begin the adjustment process.
- The system will:
    - Upload RINEX data for the current day into OPUS Static.
    - Repeat this for 7 days (one per run).
    - After 7 successful uploads, perform session and network adjustments using OPUS Projects.

4. **Automation Tip**

To run this workflow without manual intervention, a cron job or scheduled task can be configured to trigger the process daily.

**Manual Processing**

This option is used when processing historical data or re-running adjustments with a custom date range.

1. **Date Selection**

- Set Adjustment for Specific Day if you need to run for one day.
- Set Custom Date Range to define a start and end date.
- Ensure the selected dates are valid:
    - The start date must not be later than the end date.
    - Dates must match available processed RINEX data (see Step 2 for expected directory structure).

2. **State Selection**

- As with automated mode, enter the appropriate 2-letter state abbreviation.
- For default processing across the U.S., enter N/A.

3. **Upload Required Files**

- Use the same procedure described in automated processing.
- Navigate to the test dataset directory and select the antenna reference file: **test_orgn_antenna_reference_list.xlsx**

4. **Start Processing**

- Click **Generate Config and Run** to initiate the adjustment.
- The system will use your selected date range and process the corresponding RINEX files through OPUS Static, followed by session and network adjustment.

Once adjustments are completed, the system generates an OPUS Project XML file formatted for ingestion into the NSRS Align database.

### Case 2: Split Network by Region

The split network workflow allows users to divide their RTN stations into distinct regions for separate OPUS Projects adjustments. This configuration is required when the total number of stations exceeds the OPUS Projects platform limit of 99 stations per project. All regions share the same set of NGS CORS, but RTN stations must be distributed uniquely across regions with no duplication.

The process follows the same three main steps described in Section 4:

- Network Configuration
- RINEX Data Download
- OPUS Adjustments

Each step is described in full detail below using the example dataset provided in the nsrs align test data directory.

## Step 1. Configure Network Design

When implementing a regional split, each region must have its own input file that defines only the RTN stations belonging to that region. RTN stations must be uniquely assigned to a single region and must not appear in multiple files. NGS CORS should be included in all regional input files and must be same across them. In the provided test dataset, two regions are defined:

- east_orgn_test_data.xlsx
- west_orgn_test_data.xlsx

To configure the split network design:

1) Launch the **NSRS Align** application and navigate to the Baseline Generator tab.

2) Click File > Open and navigate to the **nsrs align test data\regions** directory.
    a. Select the first regional file, for example **east_orgn_test_data.xlsx**.

3) After loading the file, select a **baseline network design** in the Network Design panel. You may choose any of the available options, such as Single Hub Network.

4) Open the **OPUS Project Parameters** panel and review the available fields:
    a. Default parameters meet OPUS Projects requirements and usually do not require modification.
    b. Enter your authorized OPUS Projects email address to proceed.
        i. If you do not have this, visit the NGS website to register for OPUS Projects training.
5) Click Run to visualize the network.
    a. NGS CORS stations appear in red.
    b. RTN stations for this region appear in blue.
6) If you want to change the selection of hubs:
    a. Open the HUBS panel.
    b. If you want to use the suggested NGS CORS as hubs, leave the default "Yes" selected.
    c. To override, choose "No" and manually enter or exclude hubs using the interface provided.

7) After finalizing the network:
    a. Click Generate XML.
    b. When prompted with "**Are you splitting your project into regions**?", select Yes and click OK.
    c. A second prompt will ask for a **one-letter region identifier**.
        i. For example, if you are configuring the east region, enter **E** and click OK.

8) Save the XML file in the directory prompted.
    a. The path will be nsrs_align\opus_processing\gui_xml\region_e
    b. You may rename the file, but do not change the folder location.

9) Repeat the above steps for the next region.
    a. For west_orgn_test_data.xlsx, repeat the same process and assign the region letter **W**.
    b. The corresponding directory will be nsrs_align\opus_processing\gui_xml\region_w.

## Step 2. Download RINEX Data

This step prepares the required observation data for each regional OPUS Project defined in the previous configuration step. The process supports both automated and manual download modes. However, manual entry through the interface is not supported when using regional splitting. All station information must be provided through a CSV upload.

This step differs from the unified network case in the following ways:

- You must upload a station CSV file containing a region column that assigns each station to a specific processing region.
- Manual entry via the interface is disabled for region-split workflows.

- Even if no download is required, this file is still needed to link RINEX data to the correct regional project.

Refer to Section 5B. RINEX Download of this manual for the required CSV format and description.

**Automated Workflow Setup**

To configure daily automated downloads for a split network:

1) In the **RINEX Download** tab, locate the Station Selection Method section.

2) Leave Default Stations selected (this is the default), then click Upload Stations CSV.

**3)** Navigate to **nsrs align test data\regions\stations**

**4)** Select the file **stations.csv**. A confirmation message ("Stations CSV uploaded and saved") will appear once the file is successfully uploaded.

    a. This file **must** include the additional region column to correctly assign each station to a regional OPUS Project during upload. For format details, see Section 5B.

5) In the Configuration panel:
    a. Enter your EarthScope (GAGE) username and password
    b. Specify the path to your GAGE token (.Json)
    c. Set full paths for the required executables:
        i. crx2rnx (RINEX conversion)
        ii. gfzrnx (RINEX splicing/resampling)
    d. Click Save Configuration
    e. These paths need only be configured once unless their file locations change or credentials change.

6) Click Download RINEX to initiate the automated daily download process.

**Manual Workflow Setup**

For downloading specific historical datasets not included in automated processing:

1) In the **Date Selection** section, choose Custom Date Range and enter your desired start and end dates.

2) Leave Default Stations selected under **Station Selection Method** and click Upload Stations CSV as described above.

3) Complete the **Configuration** panel setup as in the automated workflow.

4) Click **Download RINEX** to begin the download for the selected date range.

- **Important:**
  Manual entry of stations using the "**Manual Entry**" option in the RINEX Download tab is not supported when processing a split network by region. In this workflow, stations must be provided through a CSV file that includes the associated region for each station.

- Even if RINEX files have already been manually placed and no new downloads are needed, you **must still upload the stations CSV file**. This ensures each station is correctly mapped to its assigned region, which is required for assigning RINEX files to the appropriate OPUS Project during processing. For detailed requirements on manual RINEX data placement, refer to Section 5B.

**Logs and Reporting**

The RINEX download process generates the following logs for monitoring and troubleshooting:

- download_log.csv (records download details including timestamp, station, source, file name, and status)
- download_log.log (detailed log of file processing including unzipping and conversion operations)
- station_download_summary.csv (concise summary indicating download status per station and date)

These log files are located within **nsrs_align\rinex_download_log** , providing useful diagnostics if issues arise.

## Step 3. Run OPUS Adjustments

This step executes the OPUS Projects adjustments for each region individually, using the network design and RINEX data configured in the earlier steps. It supports both automated and manual processing modes. The only addition in this workflow is the optional region configuration step, which must be completed when using multiple regions.Refer to Appendix A for the full project naming logic and how this affects project tracking and file retrieval.

**Automated Processing**

This mode is designed for scheduled, daily adjustments. It automatically uploads one day’s RINEX data into OPUS Static for each region and performs OPUS Projects adjustments when the 7-day session window is complete.

**1. Adjustment Parameters Setup**

Begin by setting the adjustment parameters. This is the first panel shown in the OPUS Project Adjustment tab. The following must be configured:

- Date Selection
  - Set to **Daily Adjustment** (this is the default).
  - This mode is required for automated processing. The system will upload one day's RINEX data per run into OPUS Static.
  - After 7 consecutive daily uploads, session and network processing are automatically triggered.

- State Selection
  - Choose the two-letter abbreviation of the U.S. state corresponding to the RTN.
  - If processing a nationwide network or default setup, enter N/A to represent the entire U.S.

2. **Optional Region Setup**

In the same panel, enable region-based processing:

- Check the box labeled **Include Extra Regions?**
- In the **Number of Regions** field, enter the total number of regions used in your project.
- Based on this number, many **Enter Region Initials** boxes will activate.
- In each box, type the exact region letter assigned during XML generation in Step 1. For the test dataset, enter **E** and **W**.

**3. Upload Required Files**

- Navigate to the test dataset directory.
- Inside the **nsrs align test data** folder, open the **antenna reference file** directory.
- Select **test_orgn_antenna_reference_list.xlsx** to upload.
- This file provides verified antenna types and heights used during OPUS Static uploads. It overrides any incorrect metadata in the RINEX headers.
- This file only needs to be re-uploaded if antenna information changes for any station.

**4. Start Processing**

- Click **Generate Config and Run** to begin the adjustment process.
- The system will:
  - Upload RINEX data for the current day into OPUS Static.
  - Repeat this for 7 days (one per run).
  - After 7 successful uploads, perform session and network adjustments using OPUS Projects.

**5. Automation Tip**

To run this workflow without manual intervention, a cron job or scheduled task can be configured to trigger the process daily.

### Manual Processing

This option is used when processing historical data or re-running adjustments with a custom date range.

1. **Date Selection**
   - Set Adjustment for Specific Day if you need to run for one day.
   - Set Custom Date Range to define a start and end date.
   - Ensure the selected dates are valid:
     - The start date must not be later than the end date.
     - Dates must match available processed RINEX data (see Step 2 for expected directory structure).

2. **State Selection**
   - As with automated mode, enter the appropriate 2-letter state abbreviation.
   - For default processing across the U.S., enter N/A.

3. **Optional Region Setup**

   1) Check **Include Extra Regions?**
   2) Enter the number of regions and supply the region initials as described above.

4. **Upload Required Files**
   - Use the same procedure described in automated processing.
   - Navigate to the test dataset directory and select the antenna reference file: **test_orgn_antenna_reference_list.xlsx**

5. **Start Processing**
   - Click **Generate Config and Run** to initiate the adjustment.
   - The system will use your selected date range and process the corresponding RINEX files through OPUS Static, followed by session and network adjustment.

Once adjustments are completed, the system generates an OPUS Project XML file formatted for ingestion into the NSRS Align database.

# 7. Logs and Diagnostics

NSRS Align writes every significant action to structured logs so that users can verify progress, troubleshoot problems, or audit results afterwards. Logs are grouped by the two processing stages and are saved under the installation's nsrs_align working directory.

**RINEX-Download Logs (nsrs_align\rinex_download_log\)**

| **Log file** | **Purpose** | **Typical questions it answers** |
|---|---|---|
| download_log.csv | Row-by-row summary of every attempted download (time-stamp, station, source, year, DOY, filename, status, file-size). | Did a file reach my computer? How large was it? |
| download_log.log | Verbose chronological record including decompression, CRX→RNX conversion and resampling steps. | Where did the conversion fail? Which physical path was used? |
| station_download_summary.csv | One-line status (SUCCESS / ERROR) for each station-DOY pair. | Which stations or days are still missing? |

**OPUS-Adjustment Logs (nsrs_align\opus_processing\)**

| Location / File | Contents | Why you might open it |
|---|---|---|
| `project_summaries\<project>.Json` | High-level JSON timeline for each project (creation, uploads, adjustments, final status). | First-stop "executive summary". |
| `opuspy_automation\project_xml_info\YYYY\<project>\` | • Live `<project>_data.xml` (OPUS Project XML)<br>• parsed_outputs\ – three CSVs listing NGS CORS, RTN CORS and session IDs inside the project. | Inspect exactly what OPUS contains; compare expected vs. actual content. |
| `opuspy_automation\logs\project_summary.csv` | One line per project (state, GPS week, region, creation date). | Inventory of everything NSRS Align has created. |
| `opuspy_automation\logs\upload_status_log.csv` | Tracks the daily RINEX-upload CSV sent to OPUS Static (success/error, path, DOY). | Confirm that each day's files were accepted by OPUS Static. |
| `opuspy_automation\logs\rinex_upload_csvs\YYYY\<project>\DOY\` | The exact CSV submitted for that DOY. | Re-upload, manual inspection. |
| `opuspy_automation\logs\antenna_conflicts.csv` | Compares antenna type / height found in RINEX headers against your Antenna Reference File. | Identify stations with incorrect metadata. |
| `opuspy_automation\logs\ephemeris_rerun.log` | Flags projects run with broadcast ephemeris; automation re-queues them once precise ephemeris is released. | Check why a project was re-processed two weeks later. |
| `opuspy_automation\logs\projects_info\YYYY\<project>\central_logs\<project>.log` | Timestamped narrative of everything that happened to this project: creation, CSV uploads, session monitoring, network adjustments, retries. | **Primary troubleshooting file** for an individual project. |

| | | |
|---|---|---|
| **projects_info\YYYY\<project>\ session_solution_statistics\ <project>_peak_to_peak_summary.csv** | Peak-to-peak scatter (cm) for North, East, Up across all sessions. | Quick quality check after solutions post. |
| **Template folders inside opuspy_automation\logs\projects_info \YYYY\<project>** | JSON templates for preliminary, minimally-constrained, fully-constrained and retry adjustments. | Advanced users can re-submit or edit templates manually. |
| **add_remove_cors.csv** (same folder) | List of NGS CORS to add/remove so that the project matches your Baseline design exactly. | Verifies that only the intended control stations remain. |

Supporting sub-directories such as grouped_sessions, marks_info, match_logs, uploads, and status are maintained internally by the automation engine; direct user interaction is rarely required.

**For most troubleshooting tasks open**
**opuspy_automation\logs\projects_info\YYYY\<project>\central_logs\<project>.log**

It provides a full, time-ordered record of the project's processing sequence.

# Appendix A — Back-End Workflow Architecture

## A-1 Automated Project Creation & Naming OVERVIEW

This document outlines the configuration structure used to drive the automated project creation process. It explains:

- Supported config fields
- Default behaviors
- How different configurations influence the final project names created

| Field | Required | Default | Purpose |
|---|---|---|---|
| state_code | No | "US" | Two-letter code for project prefix |
| date | No | Today’s date | Start date for determining GPS week(s) |
| end_date | No | Same as date | Enables multi-week ranges |
| regions | No | ["A"] | List of regions, one project created per region |

**NAMING**

Projects are named using the following format:

{state_code}{year_suffix}_{gps_week: 02d}{region}

**Components:**

| Component | Derivation | Example (state_code="CA", date=2024-03-01) |
|---|---|---|
| state_code | Upper-case first two characters | CA |
| YY | Last two digits of calendar year | 24 |
| week | Contiguous 7-day block index starting from January 1 (DOY 001–007 = week 01, 008–014 = week 02, etc.).<br>The final week of the year includes any remaining days (8 or 9 days depending on leap year) and is still treated as a single full week. | 09 |
| region | Upper-case entry from regions array | A |

**BEHAVIOR MATRIX**

| Scenario Description | Example Config | Resulting Project Names |
|---|---|---|
| All fields present | { "state_code": "OG", "date": "12/15/2024", "end_date": "12/21/2024", "regions": ["E", "W"] } | OG24_50E, OG24_50W |
| Single day, multiple regions | { "state_code": "OG", "date": "12/15/2024", "regions": ["E", "W"] } | OG24_50E, OG24_50W |
| Single day, one region | { "state_code": "TX", "date": "01/10/2024", "regions": ["C"] } | TX24_02C |
| Minimal config (only date) | { "date": "03/01/2024" } | US24_09A |
| Empty config (all defaults applied) | {} | e.g., US25_13A (based on today's date) |
| Cross-week range, multi-region | { "state_code": "CA", "date": "03/01/2024", "end_date": "03/14/2024", "regions": ["A", "B"] } | CA24_09A, CA24_09B, CA24_10A, |

**FIELD EFFECTS EXPLAINED**

- **state_code**
  - Affects the prefix of the project name
  - Defaults to "US" if missing

- **date**
  - Used to determine:
    - Year suffix
    - Day of Year (DOY)
    - GPS week
  - Required for all logic

- **end_date**
  - Enables multi-week range if provided
  - If omitted → treated as single-day = single-week project

- **regions**
  - Each region generates its own project
  - Defaults to ["A"] if not provided

### A-2 SCAN AND UPLOAD RINEX FILES

#### OVERVIEW

This module handles:

- Scanning a directory of RINEX files (per day/region)
- Extracting antenna metadata
- Cross-checking with external reference antenna data
- Generating upload CSVs for OPUS
- Uploading the CSVs to OPUS projects
- Logging everything for monitoring and threshold control

#### SCANNING + METADATA EXTRACTION

Recursively scans for RINEX files by filename patterns.

Reads the header using georinex and pulls:

- Antenna type
- Antenna height

#### REFERENCE METADATA

The user is expected to provide a reference antenna CSV (typically uploaded before processing), which contains:

- Station ID (or filename)
- Antenna Type
- Antenna Height

#### METADATA VALIDATION

The extracted metadata is compared against the reference metadata.

Reference metadata takes precedence for uploading.

If there is a mismatch between extracted and reference metadata:

- The reference is used for the upload.
- The mismatch is logged in metadata_conflict_log.csv.

#### CSV GENERATION

The script generates a structured CSV (per DOY + region) with the required fields for OPUS Project upload.

**UPLOAD TO OPUS**

Once CSVs are generated:

- For each DOY and region, the corresponding CSV is passed to opus_cgi.upload_rinex().
- Upload status is returned, and the CSV is updated from PENDING → SUCCESS or FAILURE.

**WORKFLOW**

| **Step** | **What it Does** |
|---|---|
| get_date_range(config) | Gets start and end DOY from config |
| For each DOY and region | Constructs rinex_dir based on D:/.../processed/{year}/{doy} |
| process_rinex_files() | For each dir. → filters files by region → generates upload CSV |
| Wait 50 seconds | Give time between scan and upload |
| For each DOY + region | Computes project_name, loads CSV |
| If not uploaded yet | Uploads via opus_cgi.upload_rinex() |
| Logs result to upload_status_log.csv | Logs status as Success or Failure |

### A-3 Automated CORS Management (Add/Remove CORS)

#### OVERVIEW

This module handles the automated addition and removal of CORS stations to an OPUS Project after sufficient user RINEX files have been uploaded and processed. It includes:

- Monitoring when OPUS solutions for user files have been successfully processed.
- Using a per-day threshold to determine when CORS can be added or removed.
- Fetching CORS metadata from both the OPUS Project and the GUI input.
- Submitting a CORS configuration update via CGI.

#### MONITORING OPUS SOLUTIONS

The system scans the OPUS Project XML for processed user solutions per DOY (Day of Year). Once at least 90% of expected RINEX stations for a given DOY have solutions, that day is marked COMPLETE.

If a day fails to reach 90%, fallback rules apply:

- If any future DOY is marked complete, earlier incomplete DOYs are downgraded to FALLBACK.
- If it is the last DOY in the range, and 24 hours have passed since initial RINEX upload, it is also marked FALLBACK.

Once all DOYs are marked COMPLETE or FALLBACK, the CORS trigger is activated.

#### INPUT SOURCES

| Source | Description |
|---|---|
| Project XML | Parsed to identify already-included CORS |
| GUI CORS CSV | Parsed to get desired CORS and 3-D settings |

#### ADD/REMOVE CORS CSV GENERATION

Once the project is ready, we generate a CORS update CSV using both the GUI and Project CORS metadata.

#### CGI Request

The opus_cgi.add_cors() function is called with the contents of the generated CORS CSV, which sends a single add/remove request to OPUS.

## A-4 Session Template Generation & Submission

### OVERVIEW

This module is responsible for identifying sessions in a project, generating structured JSON templates for each session, and sending CGI requests to queue them for processing via OPUS.

It leverages session schedules, user and CORS metadata, GUI-defined parameters, and baseline configurations to generate each session's template accurately.

### FUNCTIONALITY

This module handles:

- Reading project session (session_ids_<project>.csv)
- Mapping users and CORS stations to their RINEX files and metadata
- Applying GUI-defined processing parameters
- Building a JSON session template per session
- Submitting each template to OPUS via opus_cgi.process_session()
- Logging expected station outcomes for later session solution verification

### TEMPLATE CONTENT

- User Block
- CORS Block
- Baseline
- GUI Parameters
- Session Processing keys

## A-5 PRELIMINARY NETWORK ADJUSTMENT

### OVERVIEW

This module monitors when session solutions are available in the project and once confirmed (or fallback threshold is reached), it triggers a preliminary network adjustment by submitting a JSON request to OPUS. The output from session solutions is used to populate the network adjustment template.

### FUNCTIONALITY

The module performs the following:

- Periodically checks for session solutions in project_info_<project>.xml
- Validates whether each session has a complete or fallback solution based on expected users
- Once all sessions pass, it:
  - Builds a preliminary network adjustment JSON template
  - Sends a CGI request to OPUS using the JSON template

### SESSION SOLUTION MONITORING

Session completeness is determined as follows:

- Each session has a CSV (<project>_<session>_expected.csv) listing the expected ESIDs.
- From the project XML, the actual solved ESIDs are parsed per session (only those with NAD_83(2011) positions).
- If ≥80% of expected ESIDs are solved → session marked as COMPLETE.
- If not, but 24 hours have passed since the session template was submitted → session marked as FALLBACK
- If neither is true → session is INCOMPLETE
- A match log (<project>_<session>_match_log.csv) is written for each session

Once all sessions for the project are either COMPLETE or FALLBACK → the module proceeds to trigger the preliminary adjustment.

### TEMPLATE CONTENT

- User Block
- CORS Block
- Baseline
- GUI Parameters
- Preliminary Network Adjustment Processing keys

## A-6 MINIMALLY CONSTRAINED NETWORK ADJUSTMENT

### OVERVIEW

This module triggers a minimally constrained network adjustment, where only one CORS station is constrained (3-D). It builds a new JSON template and submits the network adjustment request to OPUS.

### FUNCTIONALITY

The module performs the following:

- Builds a new adjustment JSON template
- Sends a CGI request for OPUS to process the adjustment

### TEMPLATE CONTENT

- User Block
- CORS Block
- Baseline
- GUI Parameters
- Minimally Constrained Network Adjustment Processing keys

## A-7 FULLY CONSTRAINED ADJUSTMENT

### OVERVIEW

This module triggers a fully constrained network adjustment, where all NGS CORS stations are constrained (3-D). It constructs the appropriate network adjustment JSON template and submits the CGI request to OPUS. After processing, it verifies the quality of the solution based on constraint ratios and F-statistics test.

### FUNCTIONALITY

The module performs the following:

- Builds a fully constrained adjustment JSON template
- Sends a CGI request for OPUS to process the adjustment
- After processing, checks the solution statistics to verify that the F-test passed
- Checks that all constraint ratios are less than 3
- If any station exceeds a constraint ratio of 3, it is unconstrained, and the network is reprocessed
- Repeats this loop until the F-test passes and all stations meet the constraint threshold

### TEMPLATE CONTENT

- User Block
- CORS Block
- GUI Parameters
- Fully Constrained Network Adjustment Processing Keys

## A-6 Central Log Hierarchy

```
opus_processing
├─ project_summary.csv
├─ logs
│  ├─ upload_status_log.csv
│  ├─ antenna_conflicts.csv
│  ├─ ephemeris_rerun.log
│  └─ projects_info
│     └─ YYYY
│        └─ <project>
│           ├─ central_logs\<project>.log
│           ├─ session_templates\
│           ├─ preliminary_network_adjustment_template\
│           ├─ minimally_constrained_network_adjustment_template\
│           ├─ fully_constrained_network_adjustment_template\
│           ├─ retry\fully_constrained_network_adjustment_template\
│           ├─ session_solution_statistics\
│           ├─ add_remove_cors.csv
│           └─ … (status / grouped_sessions / marks_info / uploads)
└─ project_xml_info
   └─ YYYY\<project>\projectname_data.xml
      └─ parsed_outputs\ (ngs_cors.csv, rtn_cors.csv, sessions.csv)
```

# Appendix B Log & File Reference

(Detailed field definitions for every diagnostic file mentioned in Section 7)

The tables below document the exact structure, purpose, and storage path of each log or data file generated by NSRS Align. Use these definitions for automated parsing, ad-hoc filtering, or advanced troubleshooting.

### B-1 RINEX Download Diagnostics

| File Name & Location | Description | Key Fields (Type / Example) |
|---|---|---|
| rinex_download_log \download_log.csv | Machine-readable summary of every individual download attempt. | **timestamp** (ISO 8601 / 2025-06-02T12:14:54.0752) · **station** (text / ADEL) · **source** (enum {ftp, gage, geodesy}) · **year** (int / 2025) · **doy** (int / 1) · filename (text) · **status** (enum {SUCCESS, ERROR}) · **file_size** (int bytes) |
| rinex_download_log \download_log.log | Verbose, line-by-line narrative of the entire download pipeline: transfer, unzip, conversion, resample. Generated with Python logging at INFO level. | Free-form text. Each entry begins with ***YYYY-MM-DD HH:MM:SS,ms – LEVEL – Message*** (e.g., "Converted: …\ADEL00USA…rnx") |
| rinex_download_log\ station_download_summary.csv | One-row status per station/DOY for rapid completeness checks. | **station** · **year** · **doy** · **status** (SUCCESS / ERROR) |

### B-2 Project-Level Overview

| File Name & Location | Description | Key Fields |
|---|---|---|
| opus_processing\project_summary.csv | Snapshot of every OPUS project created by NSRS Align. | **Project Name** (text) · **State** (2-letter) · **Year** (int) · **Week** · **Region** (char or blank) · **Creation Date** (local date) |

### B-3 RINEX Upload Tracking

| File Name & Location | Description | Key Fields |
|---|---|---|
| opus_processing\logs \upload_status_log.csv | One record for every daily RINEX CSV sent to OPUS Static. | **csv_filepath** (path) · **project_name** · **status** (Success / Error) · **year** · **doy** · **region** · **timestamp** · **error_msg** (blank on success) |
| opus_processing\logs \rinex_upload_csvs\YYYY \<project>\DOY\.csv | The actual CSVs posted to OPUS Static (auto-generated, read-only). | **file_id** (int) · **path** (full .rnx) · **height** (m) · **ant_type** (text) |

### B-4 Antenna Consistency Audit

| File Name & Location | Description | Key Fields |
|---|---|---|
| opus_processing\logs \antenna_conflicts.csv | Records any mismatch between RINEX header values and the Antenna Reference File. | **timestamp** · **year** · **doy** · **region** · **station** · **rinex_ant_type** · **ref_ant_type** · **ant_type_match** (bool) · **rinex_height** · **ref_height** · **height_match** (bool) · **notes** (path to offending file, remarks) |

### B-5 Ephemeris Rerun Flag

| File & Location | Purpose | Key Fields |
|---|---|---|
| opus_processing\logs \ephemeris_rerun\ephemeris_rerun.log | Records projects that are automatically re-processed once precise ephemerides are available (≥ 14 days after broadcast processing). A companion file <project>_rerun.status prevents duplicate reruns. | **timestamp** · **level** · **project_name** · **message** |

### B-6 Central Workflow Log — one per project

| File & Location | Purpose | Sample Tagged Messages |
|---|---|---|
| opus_processing\logs \projects_info\YYYY\<project> \central_logs\<project>.log | Chronological master log tracing every automation step (project creation, daily uploads, template generation, retries, validation). | • Created project<br>• Uploaded RINEX CSV to project: daily data accepted<br>• Session match monitoring<br>• Generating preliminary / minimally / fully constrained template<br>• Reprocessing with bad stations excluded |

### B-7 Project XML & Parsed Outputs

| Directory | Contents |
|---|---|
| opus_processing\project_xml_info\YYYY\<project>\ | projectname_data.xml — constantly updated master XML returned by OPUS. |
| opus_processing\project_xml_info\YYYY\<project>\parsed_outputs\ | Helper CSVs generated from the XML |

### B-8 Adjustment Templates & Statistics

| Directory / File | Purpose |
|---|---|
| …\fully_constrained_network_adjustment_template*.Json | CGI payload for the final constrained network adjustment. |
| …\minimally_constrained_network_adjustment_template*.Json | CGI payload for the minimally constrained adjustment. |
| …\preliminary_network_adjustment_template*.Json | Initial quality-gate adjustment. |
| …\retry\fully_constrained_network_adjustment_template*.Json | Regenerated template excluding failed stations. |
| session_solution_statistics\<project>_peak_to_peak_summary.csv | Mark-by-mark peak-to-peak scatter (cm N / E / U). |

### B-9 Add / Remove CORS

| File & Location | Purpose |
|---|---|
| opus_processing\logs\projects_info \YYYY\<project>\add_remove_cors.csv | Ensures the NGS CORS set in Baseline Generator exactly matches the OPUS Project. Each row is flagged Add or Remove with justification text. |

### Quick-Reference Troubleshooting Matrix

| Task | Primary Log(s) |
|---|---|
| Verify all daily RINEX files downloaded | station_download_summary.csv |
| Investigate a failed or missing file (RINEX download) | download_log.log (filter by station / DOY) |

| | |
|---|---|
| Confirm daily upload reached OPUS Static | upload_status_log.csv (status = SUCCESS) |
| Diagnose an adjustment that stalled | central_logs\\<project>.log + template JSONs |
| Check antenna discrepancies | antenna_conflicts.csv |
| Review network stability after session processing | session_solution_statistics*peak_to_peak_summary.csv |

# Appendix C Baseline-Design Algorithms

(Referenced by §5 Baseline Generator)

All three strategies below create a connected network with exactly ( n – 1 ) baselines, where n is the total number of stations.

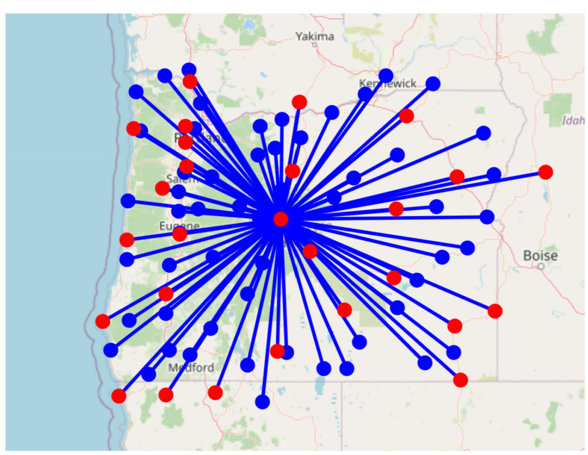
| Design | Hub Selection | Network Design |
|---|---|---|
| Single Hub | One user-chosen hub | Star (hub-and-spoke) |
| All Hubs | Every station flagged NGS CORS = TRUE | Star for RTN stations + lean hub-to-hub backbone |
| Custom Hubs | 1 – 5 hubs per user-defined cluster | Multi-star inside clusters with cross-cluster hub links |

### C-1 Single Hub Design

The Single Hub design is the simplest and focuses on creating baselines that connect all other stations directly to a single central hub.

Sequence of Steps:

1. Hub Selection:
   - The user selects one central station to serve as the hub. This hub will be the reference point for all baselines in the network.
2. Baseline Generation:
   - For each station in the network (excluding the hub), a baseline is created between the hub and the station.
3. Final Output:
   - The network consists of (n-1) baselines (where n is the total number of stations). Each baseline directly connects the hub to another station, creating a star-shaped network with the hub at the center.

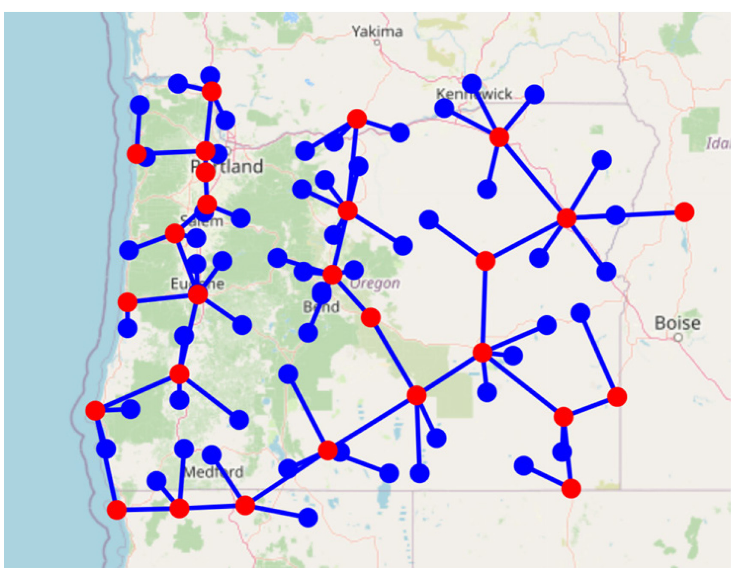


## C-2 All-Hubs Design

The All-Hubs design is more complex and focuses on creating a fully connected network where each NGS CORS (hub) is connected to every other station.

Sequence of Steps:

1. <u>Identify Hubs:</u>
   - The script identifies all NGS CORS stations in the network. These stations serve as the hubs.
2. <u>Connecting Non-Hubs to Nearest Hub:</u>
   - For each non-NGS CORS station (non-hub), the script finds the nearest hub based on geographic distance.
   - It then creates a baseline between the non-hub and its nearest hub.
3. <u>Connect Unconnected Hubs:</u>
   - The script ensures that each hub is connected to at least one other hub by finding the nearest hub and creating a baseline between them.
4. <u>Create Additional Hub-to-Hub Connections:</u>
   - If necessary, the script creates additional baselines between hubs to ensure that the number of baselines equals (n-1).
   - The connections are made by prioritizing the shortest possible distances and ensuring no loops are formed, thus maintaining an efficient network structure.
5. <u>Final Output:</u>
   - The network will have (n-1) baselines, ensuring that all stations are connected to the hubs, with direct connections between hubs.

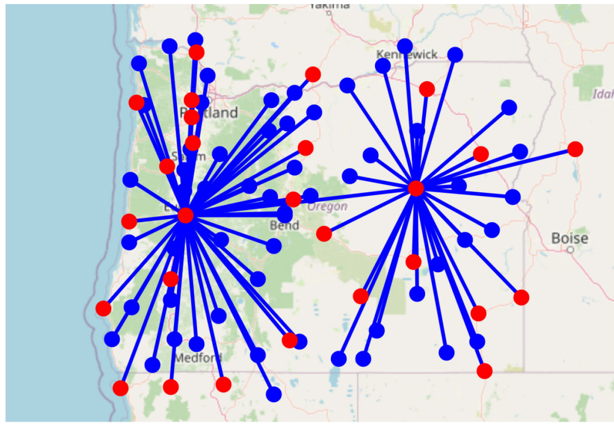


### C-2 All-Hubs Design

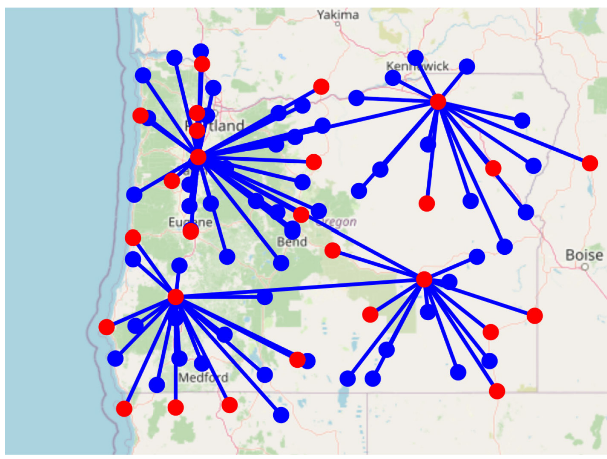
The Custom Hubs design is the most flexible and is tailored to user specifications. It allows the user to define the number of clusters and hubs, and it generates baselines accordingly.

Sequence of Steps:

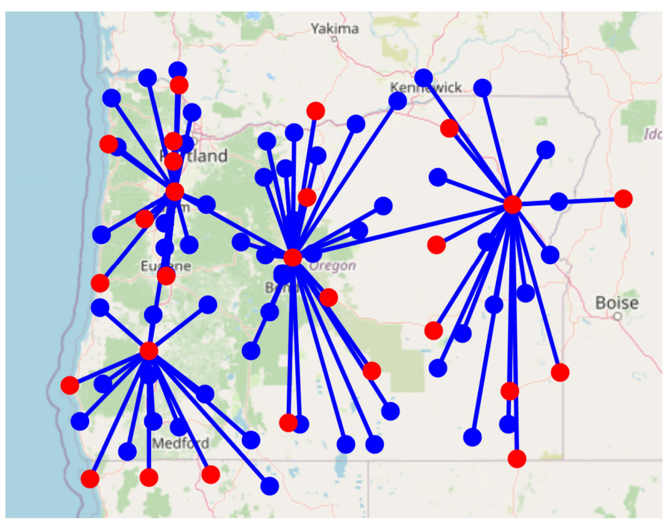
1. Cluster the Stations:
   - Use clustering algorithms (e.g., KMeans) to group stations into clusters based on their geographic coordinates. The number of main clusters is defined by the user.
2. Sub-Clustering (if Multiple Hubs are Specified):
   - If the user specifies more than one hub per cluster, each main cluster is further divided into sub-clusters. The number of sub-clusters within each main cluster matches the number of hubs specified by the user.
3. Select Hubs Within Each (Sub-)Cluster:
   - Within each main or sub-cluster, select the hub(s) based on centrality (i.e., the NGS CORS station with the smallest average distance to all other stations within the (sub-)cluster). Each (sub-)cluster will have one central hub.
4. Connect Stations Within (Sub-)Clusters:
   - Connect each non-hub station (both non-NGS CORS and non-selected NGS CORS stations) within a (sub-)cluster to its designated hub. This step ensures all stations are connected to their nearest hub, while minimizing the number of baselines.
5. Connect Hubs Across Clusters:
   - Create baselines between hubs in different clusters or sub-clusters. The connections are made by prioritizing the shortest possible distances and ensuring no loops are formed, thus maintaining an efficient network structure.
6. Ensure (n-1) Baselines:
   - Check the total number of baselines and ensure that it equals (n-1), where n is the total number of stations. If the number of baselines is less than (n-1), additional connections are made between hubs across clusters or sub-clusters, again prioritizing shorter distances and avoiding loops.
7. Final Output:
   - The network will consist of (n-1) baselines, forming star-shaped sub-networks within each (sub-)cluster, with hubs connected to stations and direct connections between hubs across clusters.



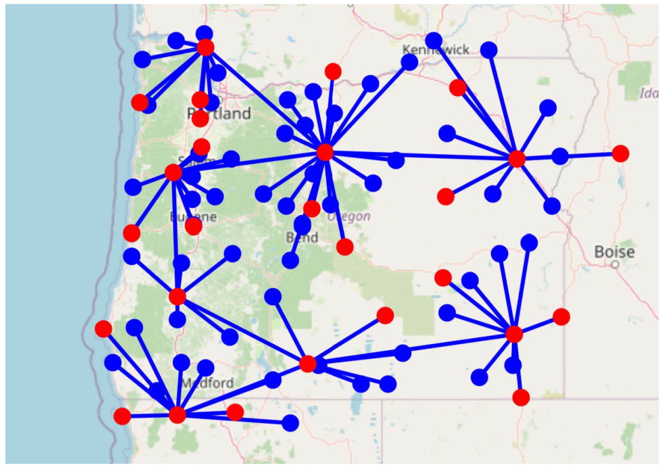

2 clusters 1 hub




2 clusters 2 hubs




4 clusters 1 hub




4 clusters 2 hubs