Extension of PRIDE PPP-AR Kinematic Orbit Determination to Other LEO Satellites

1 Related Scripts

1.1 lat2obx.py

Reads LEO satellite attitude files (quaternions) and converts them into ORBEX format. The version released by the PRIDE group currently supports only GRACE/GRACE-FO satellites.

It is important to verify the definition of the attitude product—specifically, whether the transformation is between the satellite body frame and the inertial frame, or between the body frame and the Earth-fixed frame.

The script adds a FRAME_TYPE keyword to the header of the generated file to indicate the reference frame: ECI for the inertial frame (e.g., GRACE/GRACE-FO satellites) and ECEF for the Earth-fixed frame (e.g., Swarm satellites).

```
82 def wr_obxh(wr, leocode):
83     ref_fra = 'ECEF'
84     if leocode[0:3] == 'gra':
85         ref_fra = 'ECI'
86     wr.write("%=ORBEX  0.09\n")
87     wr.write("%%\n")
88     wr.write("+FILE/DESCRIPTION\n")
89     wr.write("%s%s\n" % (" DESCRIPTION".ljust(21, ' '), "LEO satellite attitude quaternions "))
90     wr.write("%s%s\n" % (" CONVERTED_BY".ljust(21, ' '), "PRIDE Lab, GNSS Research Center, Wuhan University"))
91     wr.write("%s%s\n" % (" CONTACT".ljust(21, ' '), "pride@whu.edu.cn"))
92     wr.write("%s%s\n" % (" LEO_SATELLITE".ljust(21, ' '), leocode))
93     wr.write("%s%s\n" % (" FRAME_TYPE".ljust(21, ' '), ref_fra))
94     wr.write("-FILE/DESCRIPTION\n")
95     wr.write("+EPHEMERIS/DATA\n")
```

Figure 1 Screenshot of the lat2obx.py Script

1.2 leoatx.py

Adds LEO satellite PCOs (Phase Center Offsets) to an ANTEX file by frequency. The version released by the PRIDE group currently supports only GRACE/GRACE-FO satellites.

In the ATX class, PCO values are stored in a nested dictionary with the structure:

{satellite_id1: (start_time, end_time): {frequency1: PCO_value, frequency2: PCO_value}, satellite_id2: …}.

The satellite ID consists of four lowercase characters. Frequency identifiers follow the ANTEX convention, and the PCO unit is millimeters (mm).

```
class ATX:
    def __init__(self):
        self.PCO = {'graa': {(52368, 58058): {'G01': "-0.4 -0.4 -451.142", 'G02': "-0.4 -0.4 -475.65"}},
                    'grab': {(52368, 58058): {'G01': "0.6 0.754 -451.73", 'G02': "0.6 0.754 -475.96"}},
                    'grac': {(58260, 99999): {'G01': "-1.28 260.23 -486.24", 'G02': "-1.28 260.23 -492.24"}},
                    'grad': {(58260, 99999): {'G01': "-1.07 260.04 -485.45", 'G02': "-1.07 260.04 -491.45"}},
                    }
```

Figure 2 Screenshot of the leoatx.py Script

1.3 plotkin2pso.py

Plots time series of orbit determination errors using the precise science orbit (PSO) as reference. The version released by the PRIDE group currently supports only GRACE/GRACE-FO satellites.

1.4 pso2kin.py

Converts precise science orbits into the PRIDE PPP-AR kin file format, which is used as the initial values for position parameters. The version released by the PRIDE group currently supports only GRACE/GRACE-FO satellites. **If no corresponding precise science orbit is available, the SPP solution is used as the initial value for PPP position parameters.**

1.5 prepare_leodata.sh

Used to download GRACE/GRACE-FO onboard observation files, attitude files, and precise science orbits. The observation files are stored in the "${year}/data/" directory, while the latter two are stored in the "${year}/product/leo" directory, with naming formats "lat_${year}${doy}${sat}" and "pso${year}${doy}_${sat}", respectively. Here, ${year}, ${doy}, and ${sat} denote the year, day of year, and four-character satellite ID. **This directory structure and naming convention are designed to facilitate software recognition and simplify operations (the pdp3 script reads files by default using this structure and naming format).**

```
case $prefix in
"GPS1B" )
    local rnx1="grac${ydoy[1]}0.${yy}o"
    local rnx2="grad${ydoy[1]}0.${yy}o"
    mv $file1 data/${ymd[0]}/$rnx1
    mv $file2 data/${ymd[0]}/$rnx2
    ;;
"SCA1B" )
    local lat1="lat_${ydoy[0]}${ydoy[1]}_grac"
    local lat2="lat_${ydoy[0]}${ydoy[1]}_grad"
    lat2obx.py $file1 grac && mv $lat1 ${ymd[0]}/product/leo/
    lat2obx.py $file2 grad && mv $lat2 ${ymd[0]}/product/leo/
    rm -f $file1 $file2
    ;;
"GNV1B" )
    local pso1="pso_${ydoy[0]}${ydoy[1]}_grac"
    local pso2="pso_${ydoy[0]}${ydoy[1]}_grad"
    mv $file1 ${ymd[0]}/product/leo/$pso1
    mv $file2 ${ymd[0]}/product/leo/$pso2
```

Figure 3 Screenshot of the prepare_leodata.py Script

1.6 pdp3.sh

1.6.1 LEO Satellite ID Matching

When calling the pdp3 script without specifying the station name via the -n option, the four-character

satellite ID is extracted from the “MARKER NAME” field in the observation file header (e.g., graa/grab/grac/grad). **This satellite ID corresponds to those used in leoatx.py and in the attitude file names.**

When adding other satellites, this part needs to be updated accordingly, or the satellite ID can be specified explicitly each time by using the -n option when running pdp3.

```
623     # Get LEO satellie name form observation file and set site name
624     if [ "$mode" == "L" -a -z "$site" ]; then
625         site=$(grep "MARKER NAME" "$rnxo_path" | awk -v sep='-' '{print $1sep$2}')
626         [ -n "$site" ] && site="${site:0:3}${site: -1}"
627     fi
```

Figure 4 LEO Satellite ID Matching in the pdp3 Script

1.6.2 Note on Attitude Definition

When the LEO satellite attitude is defined as a transformation between the satellite body frame and the inertial frame, ERP parameters for the previous day and the following day are required.

1.6.3 Adding LEO PCO to ANTEX

Use **leoatx.py** to add LEO satellite PCOs to the ANTEX file.

```
2767     ### LEO ANTEX
2768     if [ "$mode" == "L" ]; then
2769         grep -q "^${site}  .*TYPE / SERIAL NO$" abs_igs.atx || leoatx.py abs_igs.atx ${site}
2770     fi
2771
2772     echo -e "$MSGINF Prepare IGS ANTEX file: $abs_atx done"
2773
```

Figure 5 Adding LEO Satellite PCO Using leoatx.py in the pdp3 Script

2 Code-Related Modifications

2.1 tedit

For LEO onboard GNSS data, the functions **lfbmwa.f90** (based on the Melbourne–Wübbena combination) and **mstpir.f90** (based on the geometry-free combination) have been added.

Unlike ground-based receivers, LEO satellites have short observation arcs, experience complex and rapidly changing space environments, and are subject to fast ionospheric delay variations. Therefore, it is necessary to verify whether the empirical thresholds used in these two functions are suitable for the onboard data of the target LEO satellite.

Reference: Zeng, Jing, et al. Improving cycle slip detection in ambiguity-fixed precise point positioning for kinematic LEO orbit determination. GPS Solutions, 28(3), 2024.

2.2 lsq

Subroutines such as **gpsmod.f90** call the **read_lat** subroutine to read the attitude file "lat_${year}${doy}_${sat}" and compute the rotation matrix **rot_l2j** from the satellite body frame to the inertial frame.

Based on **rot_l2j**, the PCO correction for the onboard receiver and the phase wind-up effect (subroutine **lphase_windup.f90**) are computed.

Note that no nominal attitude model is implemented. Therefore, an external attitude file is required, or a nominal attitude model must be added separately.

2.3 lib / Related Library Functions

2.3.1 read_lat.f90

Reads LEO attitude files in ORBEX format and calls **leoqua2mat.f90** to compute the rotation matrix **rot_l2j** from the satellite body frame to the inertial frame.

2.3.2 leoqua2mat.f90

Special attention should be paid to the method used to convert quaternions into rotation matrices. For example, the code includes specific implementations for GRACE and GRACE-FO satellites.

```
  if (index(leoid, 'gra') .ne. 0) then       ! GRACE & GRACE-FO
    xmat(1,1)=q00+q11-q22-q33
    xmat(1,2)=2.d0*(q12+q03)
    xmat(1,3)=2.d0*(q13-q02)
    xmat(2,1)=2.d0*(q12-q03)
    xmat(2,2)=q00-q11+q22-q33
    xmat(2,3)=2.d0*(q23+q01)
    xmat(3,1)=2.d0*(q13+q02)
    xmat(3,2)=2.d0*(q23-q01)
    xmat(3,3)=q00-q11-q22+q33
  else
    xmat(1,1)=1.d0-2.d0*q11-2*q22
    xmat(2,1)=2.d0*(q01+q23)
    xmat(3,1)=2.d0*(q02-q13)
    xmat(1,2)=2.d0*(q01-q23)
    xmat(2,2)=1.d0-2.d0*q00-2.d0*q22
    xmat(3,2)=2.d0*(q12+q03)
    xmat(1,3)=2.d0*(q02+q13)
    xmat(2,3)=2.d0*(q03-q12)
    xmat(3,3)=1.d0-2.d0*q00-2.d0*q11
  endif
```

Figure 6 Screenshot of the leoqua2mat.f90 Code