pyGnssToolbox - Package pour l'enseignement des GNSS

Généralités

Import des modules

```
# Gestions des ephemerides
import gnsstoolbox.orbits as orb

# Donnees GNSS (fichiers rinex "o")
import gnsstoolbox.rinex_o as rx

# Outils de calcul (rotations,
    conversions de coordonnees...)
import gnsstoolbox.gnsstools as tools

# Calcul sur le code
import gnsstoolbox.gnss_process as proc

# Constantes utiles
import gnsstoolbox.gnss_const as const

# Corrections (troposphere, antennes...)
import gnsstoolbox.gnss_corr as corr
```

Module gnss_const

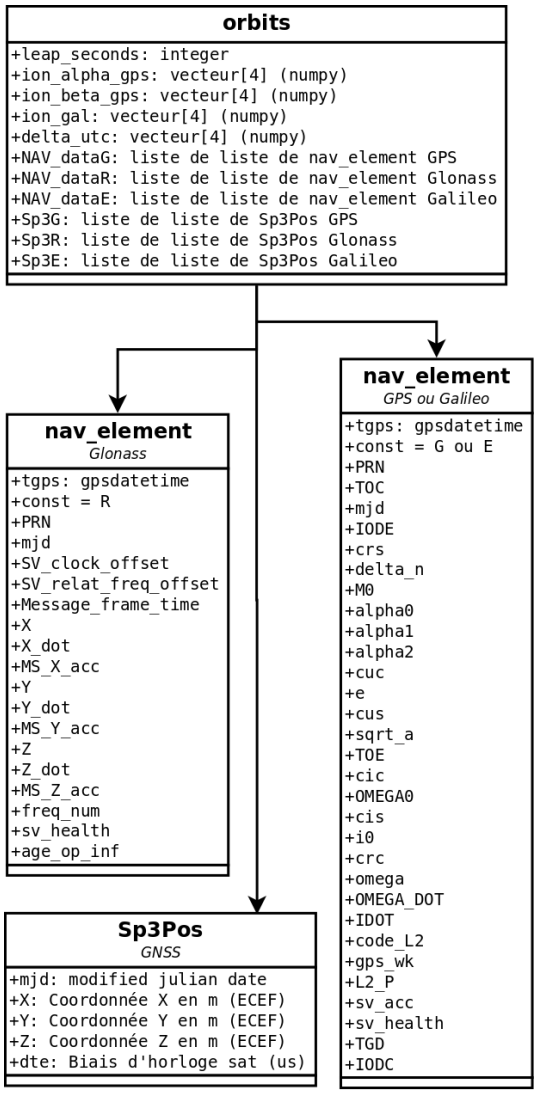
Constante	Description	Unité
c	célérité	299792458.0 $\mathrm{m.s^{-1}}$
f1	fréquence L1 gps	1.57542e9 Hz
f2	fréquence L2 gps	1.22760e9 Hz
f5	fréquence L5 gps	1.17645e9 Hz

Module Orbits

Structure

La classe orbit contient toutes les informations d'orbites :

- NAV_dataG : message de navigation GPS (liste par satellite de liste par époque)
- NAV_dataE : message de navigation Galileo
- NAV_dataR : message de navigation Glonass
- G : orbite précise GPS
- E : orbite précise Galileo
- R : orbite précise Glonass



Chargement

- Chargement d'un fichier d'éphémérides radiodiffusées (*.yyn ou *.yyg)

```
Orb = orbits.orbit()
Orb.loadRinexN('brdm1500.13p')
```

- Chargement d'un fichier d'éphémérides précises (*.sp3)

```
Orb = orbits.orbit()
Orb.loadSp3(['igs17424.sp3','
    igl17424.sp3','grm17424.sp3'])

ou
Orb.loadSp3('igs17424.sp3')
```

Accès aux données

- Récupération des informations d'un fichier d'éphémérides radiodiffusées

```
Eph = Orb.getEphemeris(
    constellation,PRN,mjd)
try:
  print("TOC : ",Eph.tgps.
    st_iso_epoch())
except:
  print("Unable to find satellite"
    )
```

Exemple :

```
Eph = Orb.getEphemeris('G'
    ,14,55480.265)
sqrt_a = Eph.sqrt_a
e = Eph.e
```

- Récupération des informations d'un fichier d'éphémérides précises

```
(orb,nl) = Orb.getSp3('G',5) #
    Satellite GPS, PRN 5
# 'G' : GPS, 'E' : Galileo, 'R' :
    Glonass
X = orb[:,1] # les coordonnees X a
     tous les instants du fichier
    sp3
Y = orb[:,2]
Z = orb[:,3]
```

Calcul de la position d'un satellite

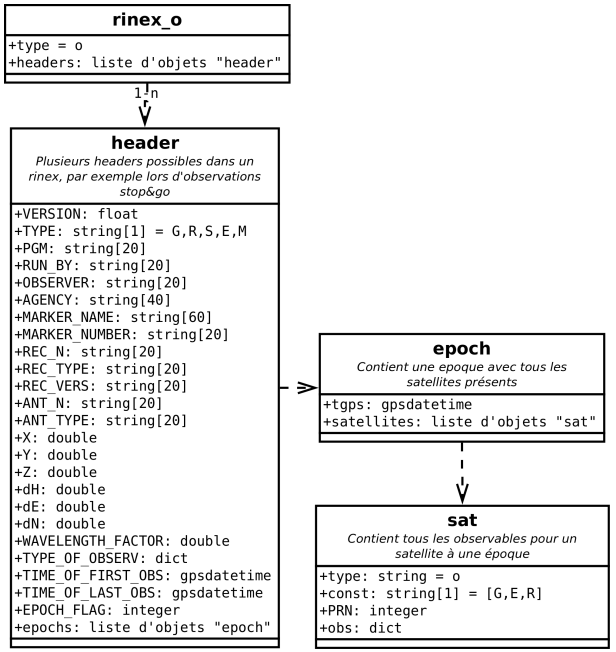
Calcul de la position ECEF et du dte d'un satellite donné par sa constellation ('G','E','R'), son PRN, un instant mjd et éventuellement un degré dans le cas d'orbites précises calculées avec un polynôme de Lagrange.

```
X, Y, Z, dte = Orb.calcSatCoord(const,
    PRN,mjd,degree)
```

Un objet de la classe debug est ajouté lors du calcul à l'objet de la classe Orbit. Il contient les résultats des calculs intermédiaires. Ses attributs correspondent aux étapes intermédiaires du calcul.
Pour y accéder :

```
print(Orb.debug.__dict__)
```

Module Rinex_o



Chargement

Chargement d'un fichier rinex d'observation (*.yyo)

```
Rnx = rx.rinex_o()
Rnx.loadRinexO('smne1500.13o')
```

Accès aux données

— Récupération d'un objet **epoch** pour une date donnée en MJD :

```
t=gpst.gpsdatetime()
t.rinex_t('13  5 30  1  0
    30.0000000')
Ep = Rnx.getEpochByMjd(t.mjd)
```

— Récupération d'un objet **header** pour une date donnée en MJD :

```
Hd = Rnx.getHeaderByMjd(t.mjd)
```

— Récupération d'un attribut d'un **header**

```
X0 = Hd.X
```

— Impression écran d'un objet **header** ou **epoch**

```
print(Hd)
print(Ep)
```

— Accès à un observable dans une **epoch** :

```
C1 = Ep.getObs("C1","G", 31)
```

ou

```
S = Ep.getSat("G", 31)
C1 = S.getObs("C1")
```

ou (valable uniquement pour le C/A code)

```
C1 = Ep.getSat("G", 31).C1
```

— Récupération des observations communes pour une même époque dans 2 rinex :

```
  Ep_base,Ep_rover = rinex_base.
    getCommonEpochs(rinex_rover,
    56442)
```

Module gnsstools

— toolGeoCartGRS80 : conversion de coordonnées géographiques vers des coordonnées cartésiennes. Les angles en entrée sont en radians.

```
X,Y,Z = tools.toolGeoCartGRS80(lon
    ,lat,h)
```

— toolCartGeoGRS80 : conversion de coordonnées cartésiennes vers des coordonnées géographiques. Les angles sont en radians.

```
lon,lat,h = tools.toolCartGeoGRS80
    (X,Y,Z)
```

— toolCartLocGRS80 : conversion de coordonnées cartésiennes vers des coordonnées locales référencées au point de coordonnées X0,Y0,Z0.

```
x, y, z = tools.toolCartLocGRS80(
    X0,Y0,Z0,X,Y,Z)
```

— toolAzEle : calcul de l'azimut et l'élévation (radians) d'un ou plusieurs satellites de coordonnées Xs,Ys,Zs (scalaire ou vecteur) vu(s) depuis un point de coordonnées X,Y,Z.

```
Az, Ele = tools.toolAzEle(X,Y,Z,Xs
    ,Ys,Zs)
```

— toolRotX, toolRotY, toolRotZ : calcul d'une rotation d'angle α radians autour de l'axe X, Y ou Z d'un point de coordonnées X,Y,Z.

```
X,Y,Z =  tools.toolRotX(X,Y,Z,
    alpha)
```

Module gnss_process

— TrilatGps : trilatération à 4 paramètres (X,Y,Z,cdtr)

```
X,Y,Z,cdtr,sigma0_2,V,Qxx =
    trilatGps(PosSat,Dobs,X0)
```

— TrilatGnss : trilatération à 4,5 ou 6 paramètres (X,Y,Z,cdtr,[cGGTO,cGPGL])

```
X,Y,Z,cdtr,cGGTO,cGPGL,sigma0_2,V,
    Qxx =  trilatGnss(PosSat,Dobs,
    X0,sat_index)
```

— TrilatGnssPonderationElev : trilatération à 4,5 ou 6 paramètres (X,Y,Z,cdtr,[cGGTO,cGPGL]) avec pondération sur l'élévation.

```
X,Y,Z,cdtr,cGGTO,cGPGL,sigma0_2,V,
    Qxx =
    trilatGnssPonderationElev(
    PosSat,Dobs,X0,sat_index,
    ElevSat)
```