백그라운드 도메인

Table of contents

0.1 백그라운드 도메인

0.1.1 1. UV 좌표계

전파 천문학에서의 u-v 좌표계는 매우 중요한 개념입니다.

- 이 좌표계는 전파 망원경이나 전파 간섭계를 사용하여 하늘의 천체를 관측할 때, 관측된 데이터의 공간 주파수 분포를 표현하는 데 사용됩니다.
- u-v 좌표계는 전파 천문학의 핵심적인 데이터 분석 도구로, 천체 이미징 및 데이터 해석에 널리 쓰입니다.

u-v 좌표계의 정의와 사용:

1. **(기본 개념)** u-v 좌표계는 기본적으로 전파 간섭계의 다양한 안테나 쌍 사이의 상대적인 위치에 기반합니다. 이 좌표계는 지구상의 안테나 쌍 사이의 거리를 기준으로 하는데, 이 거리는 천체로부터 오는 전파의 도달 시간 차이로 표현됩니다.
2. **(공간 주파수)** u-v 좌표계에서 각 좌표점은 관측된 전파 신호의 공간 주파수 성분을 나타냅니다. 이 공간 주파수는 신호의 푸리에 변환을 통해 얻어지며, 천체의 이미지를 복원하는 데 필수적입니다.
3. **(간섭계(Interferometer))** 전파 간섭계는 여러 개의 전파 망원경을 사용하여 하나의 큰 망원경과 같은 효과를 내는 장치입니다. 각 망원경 쌍 사이의 상대적인 위치는 u-v 평면상의 한 점으로 표현됩니다. 망원경들이 지구의 자전으로 인해 하늘을 따라 움직이면서, u-v 평면상에 여러 점들을 그리게 됩니다.
4. **(u-v 커버리지)** u-v 평면상에 망원경들이 그리는 점들의 집합을 u-v 커버리지라고 합니다. 이 커버리지는 관측된 천체의 이미지를 복원할 때 중요한 역할을 합니다. 좀 더 많은 u-v 커버리지는 이미지의 분해능과 품질을 향상시키며, 따라서 가능한 한 많은 u-v 커버리지를 얻는 것이 중요합니다.
5. **(천체 이미징)** u-v 데이터는 푸리에 변환을 거쳐 천체의 이미지로 변환됩니다. 이 과정에서 u-v 평면상의 데이터는 실제 하늘의 이미지로 변환되며, 이를 통해 천체의 구조와 성질을 분석할 수 있습니다.

6. **(한계와 도전)** 모든 간섭계에는 u-v 커버리지의 불완전성이 존재합니다. 이는 관측된 이미지가 완벽하지 않음을 의미하며, 천문학자들은 이러한 한계를 극복하기 위해 다양한 알고리즘과 기술을 개발하고 있습니다.

요약하자면, u-v 좌표계는 전파 천문학에서 관측된 데이터를 공간 주파수 영역에서 분석하고, 이를 통해 천체의 이미지를 복원하는 데 필수적인 도구입니다. 이 좌표계는 전파 간섭계의 기본 작동 원리를 기반으로 하며, 천문학적 데이터의 해석과 천체 이미징에 광범위하게 사용됩니다.

0.1.2 2. phase 위상

전파 천문학에서의 "baseline phase"와 "closure phase"는 간섭계 관측에서 중요한 개념들입니다. 이들은 간섭계로 관측된 데이터를 이해하고 처리하는 데 필수적인 요소들입니다.

Baseline Phase (베이스라인 위상) 1. **(정의)** Baseline phase는 특정 시간에 특정 주파수에서 두 전파 망원경(또는 안테나) 사이의 신호의 위상 차이를 의미합니다. 이 위상 차이는 두 망원경 사이의 상대적인 거리와 관측 대상까지의 거리에 의해 영향을 받습니다.

2. **(중요성)** 위상 정보는 천체의 이미지를 복원하는 데 사용됩니다. 특히, 위상은 천체의 구조와 위치 정보를 담고 있어, 정확한 위상 측정은 고해상도 천문학적 이미징을 가능하게 합니다.

3. **(도전)** 대기의 불규칙성, 기기의 지연, 그리고 기타 요인들로 인해 위상은 종종 변동될 수 있습니다. 이러한 위상 변동을 정확히 측정하고 보정하는 것이 전파 천문학에서의 중요한 과제 중 하나입니다.

Closure Phase (클로저 위상) 1. **(정의)** Closure phase는 세 개 이상의 전파 망원경을 사용하여 얻은 위상 정보의 합입니다. 특히, 세 망원경 A, B, C가 있다면, closure phase는 A-B, B-C, C-A 간의 위상 차이의 합으로 정의됩니다. 수학적으로는 아래와 같이 표현됩니다.

$$\phi_{AB} + \phi_{BC} + \phi_{CA}$$

2. **(중요성)** Closure phase는 대기나 기기로 인한 위상 오류를 상당 부분 제거할 수 있는 특성을 가지고 있습니다. 이는 각 망원경 쌍 사이의 위상 오류가 클로저 위상 계산에서 상쇄되기 때문입니다. 따라서, 클로저 위상은 상대적으로 안정적인 천체의 구조 정보를 제공하며, 이는 복잡하거나 확장된 천체의 고해상도 이미징에 특히 유용합니다.

3. **(응용)** Closure phase는 특히 긴 베이스라인을 가지는 전파 간섭계에서 중요합니다. 이를 통해 천체의 정밀한 구조를 파악하고, 대기 및 기기에 의한 오류의 영향을 줄일 수 있습니다.

종합

Baseline phase와 closure phase는 전파 간섭계를 통한 천문학적 관측에서 필수적인 개념들입니다. Baseline phase는 개별 망원경 쌍 사이의 위상 정보를 제공하는 반면, closure phase는 여러 망원경을 통한 복합적인 위상 정보를 제공하여 보다 정확한 천체 이미지 복원을 가능하게 합니다. 이러한 위상 정보는 천체의 정밀한 구조와 특성을 이해하는 데 필수적이며, 전파 천문학의 다양한 연구와 응용에 사용됩니다.

0.1.3 3. 율리우스 일

MJD: Modified Julian Date

- 율리우스 일(Julian Date, JD)을 변형한 시간 표현 방식
- 율리우스 일은 기원전 4713년 1월 1일 정오부터 계산되는 연속된 일 수로, 천문학에서 널리 사용되는 시간 체계
- 그러나 율리우스 일은 수가 매우 크기 때문에, 작은 시간 단위의 계산에는 불편함이 있음 -> 이를 해결하기 위해 MJD가 도입

  MJD는 다음과 같이 정의됩니다:

  MJD = JD - 2400000.5

  (MJD 의의) 1858년 11월 17일을 0으로 하는 날짜 체계, 이 날짜를 MJD의 기준점으로 삼음
  MJD의 도입으로 천문학자들은 더 작고 관리하기 쉬운 숫자로 날짜와 시간을 표현할 수 있게 되었습니다. MJD는 특히 천문학적 데이터의 시간을 기록하고 계산할 때 유용하며, 천체의 위치, 관측 데이터의 시간 태깅, 시간에 따른 천체 변화의 모니터링 등 다양한 분야에서 사용됩니다.

기원전 4713년 1월 1일 세계시 정오에서 현재까지 통산한 일수로서, 예를 들면 1991년 1월 1일 세계시 0시는 율리우스 통일의 2448257.5일에 해당한다. 율리우스 통일은 사용에 편리하여 착오를 피할 수 있으나, 이대로는 숫자가 방대하게 되며, 또 세계시 0시에서 시작하는 일자로 바뀌기 때문에 이것에서 2400000.5일을 뺀 일부가 제안되었다. 이것이 수정 율리우스 통일(MJD)로, 천문이나 표준 전파 등에서 편리하게 사용되고 있다. 위에서 예로 든 1991년 1월 1일 세계시 0시의 MJD는 48257.0이다. 또한 MJD는 1858년 11월 17일 세계시 0시를 기점으로 하여 통산한 일부라고 할 수 있다.

Time.now().mjd에서 Time.now()는 현재 시간을 가져오는 함수이며, .mjd는 그 시간을 MJD 형식으로 변환하여 반환합니다. 이는 현재 시간을 MJD 형식으로 표현하고자 할 때 사용됩니다.

- [참고] https://core2.gsfc.nasa.gov/time/

```
import os
os.getcwd()
```

'/Users/younghokim/astro/prj4_eht'

```
import ehtim as eh
from astropy.time import Time
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")
```

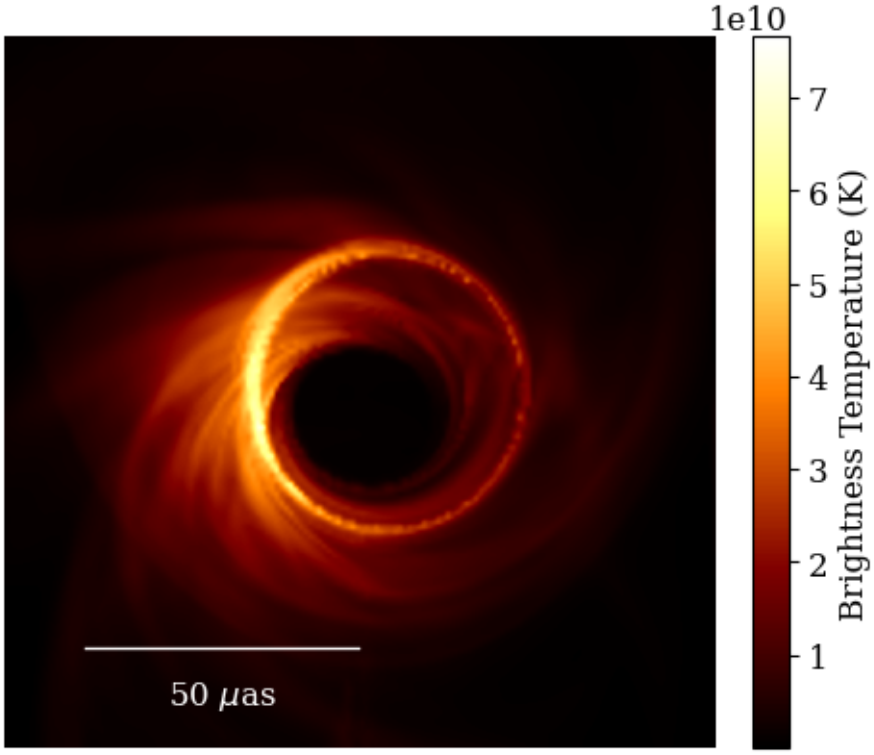
Welcome to eht-imaging! v 1.2.7

```
# mjd_now = Time.now().mjd
mjd_now = 60083.827770977245
print(mjd_now)
```

60083.827770977245

```
# load an image
im = eh.image.load_txt('eht-imaging/models/jason_mad_eofn.txt')
im.mjd = mjd_now
im.display(has_title=False, cbar_unit=['Tb'], label_type='scale');
im.ra = 89.
print(im.ra)
```

Loading text image: eht-imaging/models/jason_mad_eofn.txt
89.0



im.display(has_title=False, cbar_unit=['Tb'], label_type='scale');는 이미지 객체 im의 내용을 시각적으로 표시하는 데 사용됩니다. 이 코드는 천문학이나 관련 분야에서 이미지 데이터를 분석하고 시각화할 때 사용되는 방식을 반영하고 있습니다.

함수: - **display(…):** im 객체의 시각적 표현을 생성합니다. 이는 일반적으로 이미지 데이터를 처리하고 분석하는 데 사용되는 객체의 메소드입니다.

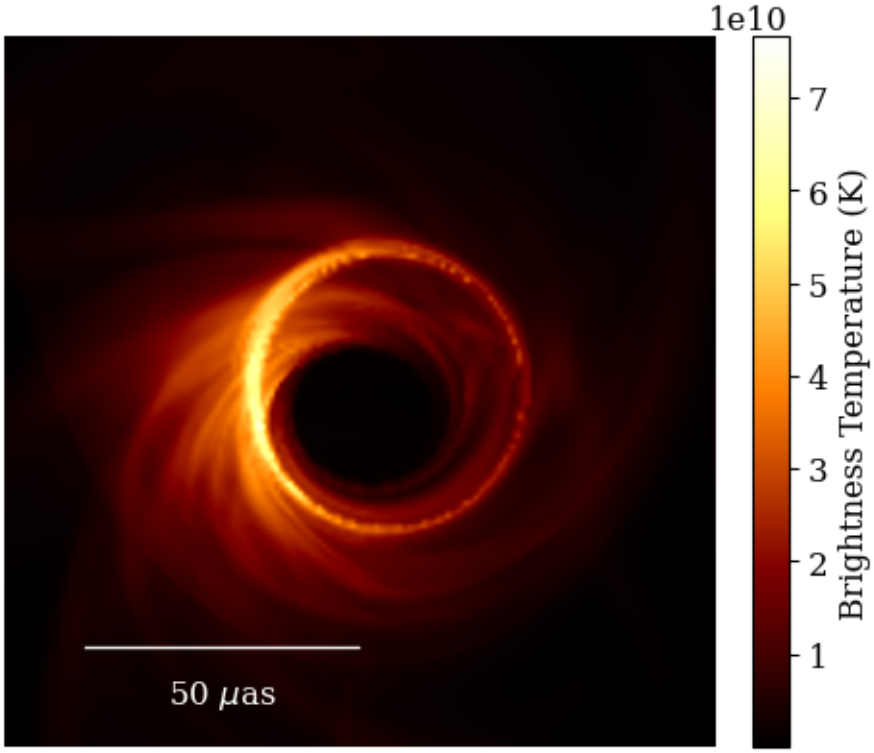
파라미터:

- **has_title=False:** 이 옵션은 이미지에 제목을 표시하지 않도록 설정합니다. 기본적으로 이미지에는 제목이 포함되어 있을 수 있지만, False로 설정하면 제목 없이 이미지만 표시됩니다.
- **cbar_unit=['Tb']:** cbar_unit는 색상 막대(color bar)의 단위를 설정합니다. 여기서 ['Tb']는 테라헤르츠(Terahertz) 또는 테슬라(Tesla)가 아니라 'K 밝기 온도'를 나타내는 'Tb'를 의미합니다. 'Tb'는 천문학에서 사용되는 밝기 온도(brightness temperature)의 단위로, 천체의 방출 또는 방사능을 온도로 표현한 것입니다.
- **label_type='scale':** 이 옵션은 축 라벨의 유형을 지정합니다. 'scale'로 설정되면, 이미지의 축에는 해당하는 척도(scale) 또는 크기 정보가 표시됩니다. 이는 이미지의 각 부분이 실제로 얼마나 큰지

또는 어떤 거리에 해당하는지를 보여주는 데 유용합니다.

```
# 오늘 날짜의 수정 율리우스 통일(통산일수) 적용
im2 = eh.image.load_txt('eht-imaging/models/jason_mad_eofn.txt')
im2.mjd = Time.now().mjd
im2.display(has_title=False,cbar_unit=['Tb'],label_type='scale');
im2.ra = 89.
print(im2.ra)
```

```
Loading text image:  eht-imaging/models/jason_mad_eofn.txt
89.0
```




```
# load an array
arr = eh.array.load_txt('eht-imaging/arrays/EHT2025wTESS.txt')
```

```
loaded spacecraft ephemeris eht-imaging/arrays/ephemeris/TESS
```

```
type(arr)
```

```
ehtim.array.Array
```

```
# look at the sites
arr.tarr['site']
```

```
array(['PDB', 'PV', 'SMT', 'SMA', 'LMT', 'ALMA', 'SPT', 'APEX', 'JCMT',
       'CARMA', 'KP', 'GLT', 'TESS'], dtype='<U32')
```

```
# look at the satellites
arr.ephem
```

```
{'TESS': array(['TESS',
        '1 43435U 18038A   23096.90060626 -.00000924  00000-0  00000-0 0  9994',
        '2 43435  37.5590  76.7615 4777727 169.3023   2.8332  0.07302504   994'],
       dtype='<U69')}
```

ephemeris (ephem) 이란?

Ephemeris는 천문학에서 특정 시간에 관측되거나 예측된 천체의 위치와 운동에 대한 정보를 제공하는 표나 데이터셋을 말합니다. 이 정보는 행성, 위성, 별, 우주선 등의 위치, 밝기, 다른 천체와의 상대적 위치 등을 포함할 수 있습니다. 천문학, 항해, 우주 탐사 등에서 중요한 역할을 합니다.

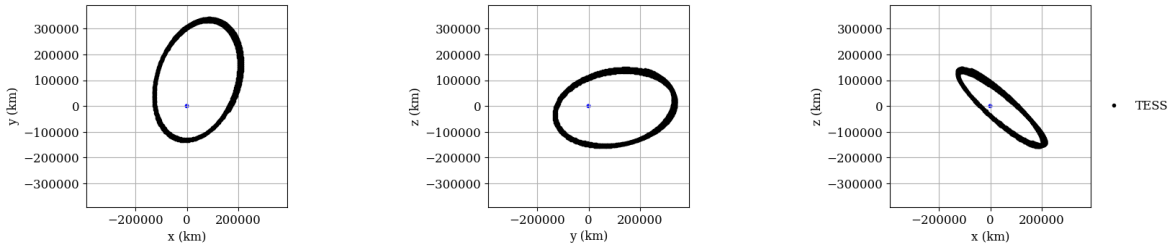
사용하는 이유:

- (위치 및 운동 정보 조회) 천체의 위치나 운동을 조회하고 분석하여, 예측 모델을 만들거나, 관측 계획을 수립하고, 천체의 특성을 연구합니다.
- (항해 및 우주 임무) 위성의 정확한 위치를 알아야 하는 경우, ephemeris 데이터는 위성과의 통신, 궤도 조정, 충돌 회피 등에 필요합니다.
- (과학 연구) 천체의 운동을 연구하여 물리적 특성, 중력 영향, 상호 작용 등을 이해하고, 우주의 기본 법칙을 탐구합니다.

```
arr.ephem['TESS'][1]
```

```
'1 43435U 18038A   23096.90060626 -.00000924  00000-0  00000-0 0  9994'
```

```
# plot the orbit over a 1 day period
arr.plot_satellite_orbits(tstart_mjd = mjd_now,
                          tstop_mjd = mjd_now + 50,
                          npoints = 1000);
```



제공된 코드는 arr 객체의 위성 궤도를 1일 동안의 기간에 대해 플롯하는 작업을 수행하는 것으로 보입니다. arr.plot_satellite_orbits 메소드는 위성의 궤도를 시각적으로 나타내기 위해 사용되며, 특정 시간 기간 동안의 궤도 경로를 플롯합니다. 여기서 사용된 파라미터들은 다음과 같은 의미를 가집니다:

```
# Copy code
arr.plot_satellite_orbits(tstart_mjd = mjd_now,
                          tstop_mjd = mjd_now + 50,
```

```
                    npoints = 1000);
```

파라미터 설명:

- **tstart_mjd:** 플롯을 시작할 시간을 Modified Julian Date(MJD) 형식으로 지정합니다. mjd_now는 현재 시간을 MJD 형식으로 나타낸 것을 가정합니다. 즉, 궤도 플롯의 시작 시간을 현재로 설정합니다.
- **tstop_mjd:** 플롯을 종료할 시간을 MJD 형식으로 지정합니다. 여기서는 시작 시간인 mjd_now에 50을 더하여 50일 후를 플롯의 종료 시간으로 지정합니다. 이는 1일보다 긴 기간을 나타내는 것 같으니, 주석이나 문제의 요구 조건과 다를 수 있습니다.
- **npoints:** 플롯에 사용할 포인트의 수를 지정합니다. 여기서 1000은 궤도를 나타내는 데 사용될 포인트의 총 수를 의미합니다. 이 값이 크면 클수록 더 세밀한 궤도가 표현됩니다.

전체적인 의미:

이 코드는 arr 객체의 plot_satellite_orbits 메소드를 사용하여, 특정 위성의 궤도를 특정 기간 동안 시각화하기 위한 것입니다. 시작 시간 tstart_mjd부터 종료 시간 tstop_mjd까지의 궤도를 npoints만큼의 포인트를 사용하여 그립니다. 이러한 플롯은 위성의 운동을 이해하고, 특정 시간대에 위성이 천체나 지구와 어떤 상대적 위치에 있을지 예측하는 데 유용합니다. 천문학, 위성 통신, 지구 관측 등 다양한 분야에서 이러한 정보는 매우 중요할 수 있으며, 궤도의 시각화는 이러한 정보를 이해하기 쉽게 만들어줍니다.

```
# generate a synthetic observation with no noise
tstart = 0 # hr relative to im.mjd
tstop = 24*50 # hr relative to im.mjd
tadv = 1200 # s
tint = 1200
bw = 4.e9

obs = im.observe(arr,
            tint,
            tadv,
            tstart,
            tstop,
            bw,
            no_elevcut_space=True,
            dcal=True, ampcal=True,
            phasecal=True,
            add_th_noise=False)
```

```
Generating empty observation file . . .
Producing clean visibilities from image with nfft FT . . .
Adding gain + phase errors to data and applying a priori calibration . . .
```

제공된 코드는 im.observe 메소드를 사용하여 합성 관측(synthetic observation)을 생성하는 과정을 수행하며, 여기서 "합성"은 실제 관측 데이터 대신 모의(simulated) 데이터를 생성하는 것을 의미합니다. 생성된 관측은 잡음이 없는(noiseless) 상태입니다. 각 파라미터의 역할과 의미를 자세히 살펴보겠습니다:

```
obs = im.observe(arr,
            tint,
```

```
            tadv,
            tstart,
            tstop,
            bw,
            no_elevcut_space=True,
            dcal=True, ampcal=True,
            phasecal=True,
            add_th_noise=False)
```

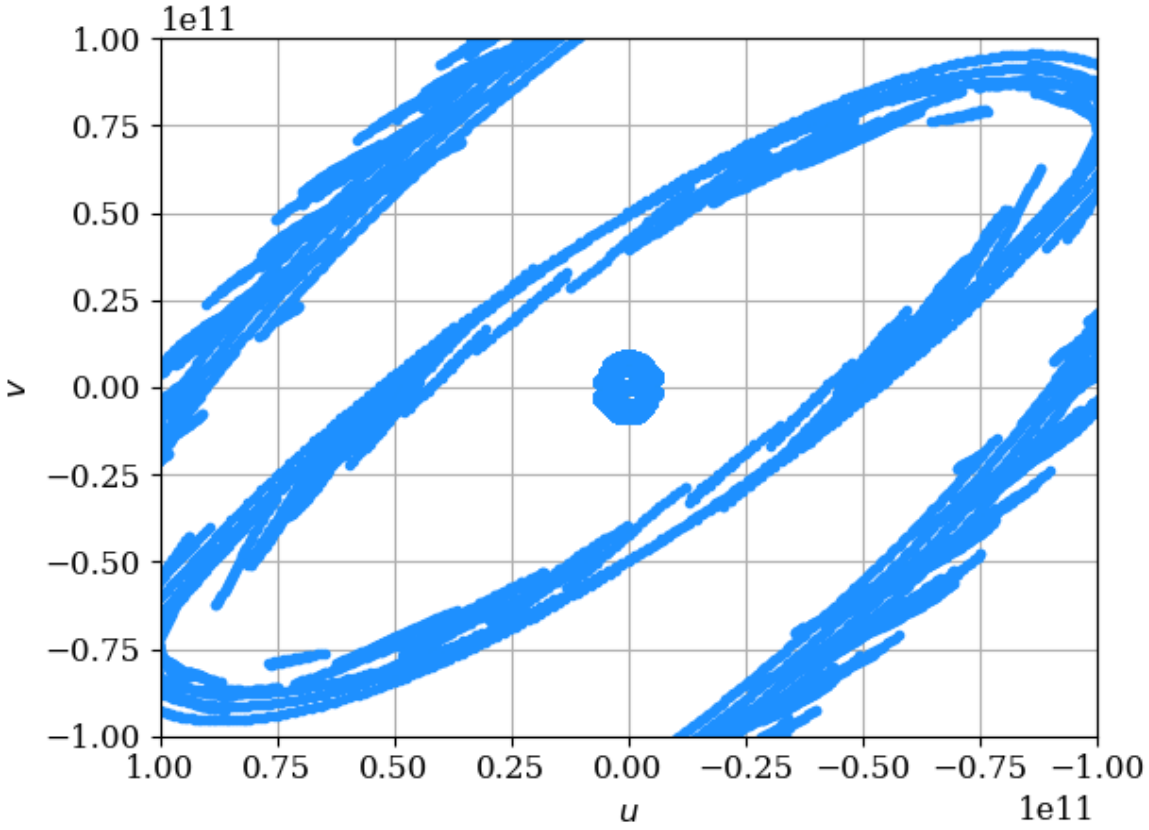
파라미터 설명

- **arr:** 관측 배열을 나타냅니다. 이는 전파 망원경 또는 다른 관측 장비의 배열을 의미할 수 있습니다.
- **tint (int):** 각 관측의 통합 시간(integration time)을 나타냅니다. 여기서는 1200초로 설정되어 있습니다. 통합 시간은 각 관측 데이터 포인트를 얻기 위해 통합하는 시간을 말합니다.
- **tadv (int):** 관측 시간을 진행시키는 단계(advance time)입니다. 여기서는 1200초 간격으로 설정되어 있습니다.
- **tstart (int):** 관측 시작 시간입니다. 여기서 0은 im.mjd 기준으로 상대적인 시작 시간을 의미합니다.
- **tstop (int):** 관측 종료 시간입니다. 여기서는 24*50시간, 즉 50일 후를 나타냅니다.
- **bw (float):** 관측의 대역폭(bandwidth)입니다. 여기서 4.e9 Hz (4 GHz)로 설정되어 있습니다.
- **no_elevcut_space (bool):** 이 옵션은 관측을 수행할 때 고도 제한(elevation cut)을 무시할지 여부를 설정합니다. True로 설정되면, 모든 고도에서 관측이 수행됩니다.
- **dcal (bool):** 지연 보정(delay calibration)을 수행할지 여부를 나타냅니다.
- **ampcal (bool):** 진폭 보정(amplitude calibration)을 수행할지 여부를 나타냅니다.
- **phasecal (bool):** 위상 보정(phase calibration)을 수행할지 여부를 나타냅니다.
- **add_th_noise (bool):** 관측에 열잡음(thermal noise)을 추가할지 여부를 설정합니다. 여기서는 False로 설정되어 있어 잡음을 추가하지 않습니다.

전체적인 의미:

이 코드는 im.observe 메소드를 통해 모의 관측을 생성합니다. 관측은 지정된 배열, 시간, 대역폭, 보정 옵션 등을 기반으로 수행됩니다. 생성된 관측 데이터는 실제 관측을 모방한 것으로, 다양한 변수와 조건을 실험하거나 관측 전략을 평가하는 데 사용될 수 있습니다. 이 과정에서 잡음을 추가하지 않음으로써, 관측 결과의 이상적인 상태를 분석하거나 다른 변수들의 영향을 명확하게 이해할 수 있도록 합니다

```
# plot uv coverage and visamps
obs.plotall('u','v',conj=True, rangex=[1.e11,-1.e11],rangey=[-1.e11,1.e11]);
```



제공된 코드 obs.plotall('u','v',conj=True, rangex=[1.e11,-1.e11],rangey=[-1.e11,1.e11]);는 obs 객체의 u-v 커버리지와 가시성 진폭(visibility amplitudes)을 플롯하는데 사용됩니다. 각 파라미터가 의미하는 바를 설명하겠습니다:

함수: plotall(…): 이것은 obs 객체의 메소드로, 다양한 데이터 포인트를 시각화하는데 사용됩니다. 여기서는 u-v 평면상의 데이터 포인트를 플롯합니다. 파라미터: 'u','v': 플롯할 데이터의 축을 지정합니다. 여기서 'u'와 'v'는 u-v 평면의 좌표를 나타내며, 이는 전파 간섭계의 각 안테나 쌍 사이의 기하학적 배열에 대응하는 공간 주파수를 의미합니다.

conj=True: 이 옵션은 복소 공액(complex conjugate) 포인트를 포함시킬지 여부를 결정합니다. 복소 공액 포인트를 포함시키면 u-v 평면의 대칭성을 높일 수 있으며, 이는 전체적인 데이터의 완전성을 향상시킵니다.

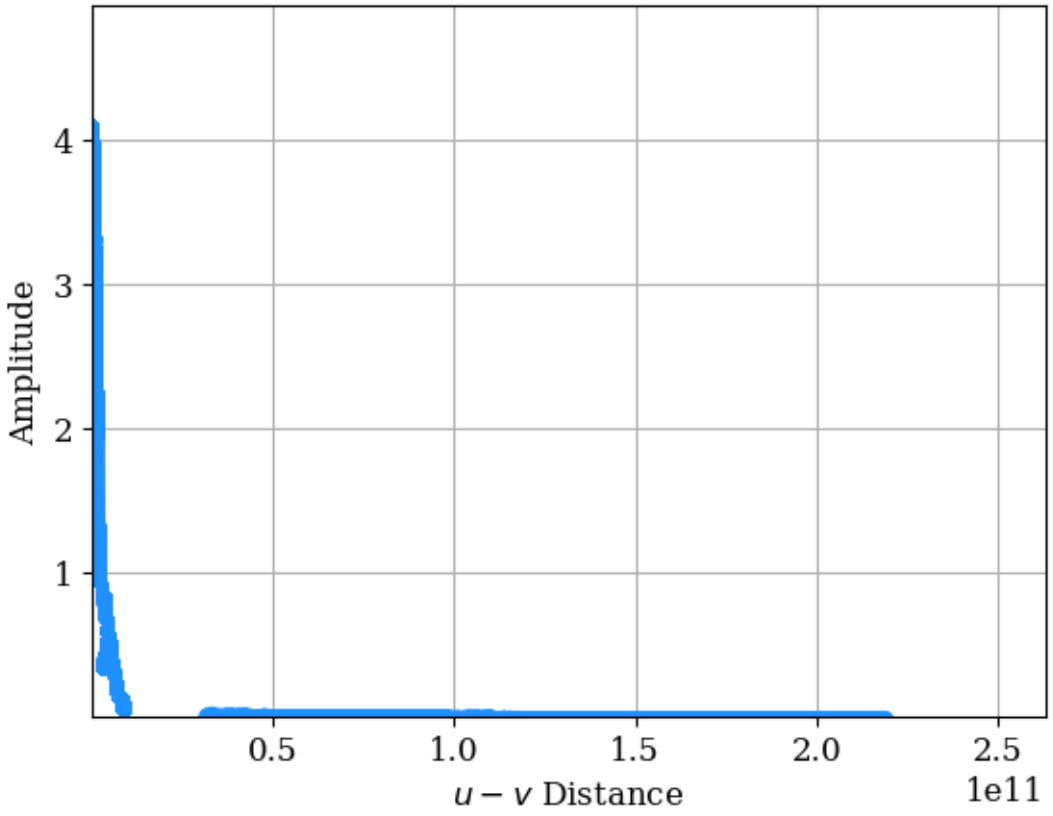
rangex=[1.e11,-1.e11]: x축(여기서는 'u' 좌표)의 범위를 설정합니다. 이 범위는 플롯에 표시되는 u 좌표의 최소값과 최대값을 지정합니다.

rangey=[-1.e11,1.e11]: y축(여기서는 'v' 좌표)의 범위를 설정합니다. 이 범위는 플롯에 표시되는 v 좌표의 최소값과 최대값을 지정합니다.

전체적인 의미: obs.plotall 함수 호출은 u-v 커버리지를 시각화합니다. u-v 커버리지는 전파 간섭계로부터 얻은 데이터가 u-v 평면에 어떻게 분포하는지 보여주는 중요한 정보입니다. 이 분포는 관측 가능한 천체의 크기와 구조에 대한 정보를 제공하며, 간섭계의 설계와 관측 전략에 중요한 영향을 미칩니다. conj=True

옵션은 u-v 평면의 대칭성을 높여 전체적인 데이터의 완전성을 향상시키는데 도움을 줍니다. rangex와 rangey는 플롯의 범위를 설정하여, 관심 있는 u-v 영역을 집중적으로 살펴볼 수 있게 해줍니다. 이러한 시각화는 관측 데이터의 품질을 평가하고, 향후 관측 계획을 수립하는데 중요한 도구로 사용됩니다.

```
obs.plotall('uvdist','amp');
```



제공된 코드 obs.plotall('uvdist','amp');는 obs 객체의 메소드 plotall을 사용하여, u-v 거리와 진폭(amp)에 대한 플롯을 생성합니다. 이는 전파 간섭계로부터 얻은 데이터의 특성을 시각적으로 분석하는 데 사용됩니다. 각 파라미터에 대해 설명하겠습니다:

함수: plotall(…): 이 메소드는 obs 객체의 다양한 데이터 포인트를 시각화하는 데 사용됩니다. 여기서는 u-v 거리와 진폭 데이터를 플롯합니다. 파라미터: 'uvdist': 첫 번째 파라미터는 플롯할 데이터의 첫 번째 축을 지정합니다. 여기서 'uvdist'는 u-v 거리를 의미합니다. u-v 거리는 간섭계의 각 안테나 쌍 사이의 거리를 공간 주파수 단위로 나타내며, 이는 해당 데이터 포인트가 관측된 천체의 구조에 대한 고해상도 정보를 제공합니다.

'amp': 두 번째 파라미터는 플롯할 데이터의 두 번째 축을 지정합니다. 'amp'는 진폭(Amplitude)을 의미하며, 이는 관측된 신호의 강도나 밝기를 나타냅니다. 진폭은 천체의 방사성 및 구조적 특성에 대한 중요한 정보를 포함하고 있습니다.

전체적인 의미: obs.plotall('uvdist','amp'); 코드는 u-v 거리와 진폭 데이터를 이용하여 플롯을 생성합니다.

이 플롯은 관측된 데이터의 공간 주파수 분포와 강도를 시각적으로 표현합니다. u-v 거리에 따른 진폭의 분포는 천체의 구조, 밝기 분포, 관측의 해상도 등에 대한 통찰을 제공할 수 있습니다. 특히, 다양한 u-v 거리에서 얻어진 데이터 포인트들은 천체의 이미지를 복원하는 데 필수적이며, 이러한 플롯을 통해 데이터의 품질과 관측 전략을 평가할 수 있습니다. 천문학자들은 이러한 시각화를 사용하여 관측 데이터를 분석하고, 천체의 특성을 이해하며, 향후 관측 계획을 수립하는 데 중요한 정보를 얻습니다.—

```
# replace TESS with a geosynchronous orbiter
arr2 = arr.remove_site('TESS')
arr2 = arr2.add_satellite_elements('GEO',
                           perigee_mjd=mjd_now,
                           period_days=1,
                           eccentricity=0,
                           inclination=0,
                           arg_perigee=0,
                           long_ascending=0,
                           sefd=10000)
arr2.ephem
```

{'GEO': [60083.827770977245, 1, 0, 0, 0, 0]}

제공된 코드는 우선 'TESS'라는 사이트 또는 위성을 arr라는 배열 또는 네트워크에서 제거하고, 그 자리에 새로운 정지궤도(geosynchronous) 위성을 추가하는 과정을 수행하고 있습니다. 그 후에는 이 변경된 배열의 천체 역학적 정보(ephemeris)를 조회합니다. 각 단계를 자세히 설명하겠습니다:

```
# Copy code
arr2 = arr.remove_site('TESS')
arr2 = arr2.add_satellite_elements('GEO',
                           perigee_mjd=mjd_now,
                           period_days=1,
                           eccentricity=0,
                           inclination=0,
                           arg_perigee=0,
                           long_ascending=0,
                           sefd=10000)
arr2.ephem
```

각 단계 설명:

1. arr.remove_site('TESS'):

- remove_site 메소드는 'TESS'라는 사이트(또는 위성)를 arr 배열에서 제거합니다. TESS(Transiting Exoplanet Survey Satellite)는 실제로는 외계 행성 탐색 위성이지만, 여기서는 배열 내의 한 요소를 지칭할 수 있습니다. 이 단계 후에, 'TESS'는 더 이상 배열에 포함되지 않습니다.

2. arr2.add_satellite_elements('GEO', ...):

- Add an earth-orbiting satellite to the array from simple keplerian elements perfect keplerian orbit is assumed, no derivatives
- add_satellite_elements 메소드는 새로운 위성을 배열에 추가합니다. 여기서 추가되는 위성은

'GEO'라고 명명되며, 그 특성은 정지궤도 위성의 것으로 설정됩니다. 각 파라미터는 다음을 의미합니다:

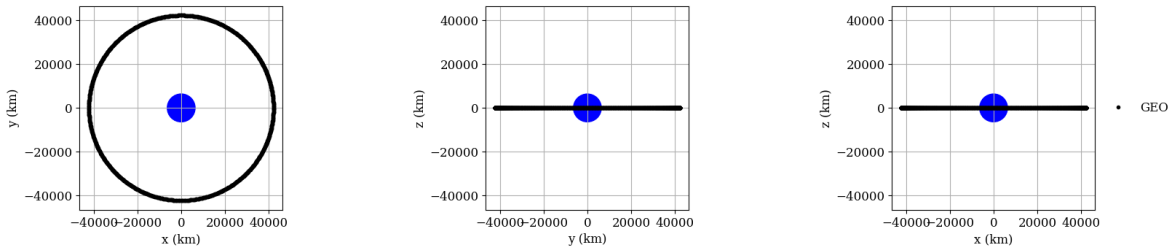
- perigee_mjd=mjd_now: 가장 가까운 지점의 시간을 MJD 형식으로 설정합니다. 여기서는 현재 시간을 나타냅니다.
- period_days=1: 위성의 궤도 주기를 1일로 설정합니다. 이는 정지궤도 위성의 특성을 반영합니다.
- eccentricity=0: 궤도의 이심률을 0으로 설정하여, 완벽한 원형 궤도를 의미합니다.
- inclination=0: 궤도의 경사도를 0으로 설정하여, 적도면에 대해 정지해 있는 것을 의미합니다.
- arg_perigee=0: 근지점의 인자를 0으로 설정합니다.
- long_ascending=0: 승교점의 경도를 0으로 설정합니다.
- sefd=10000: 시스템 등가 플럭스 밀도(System Equivalent Flux Density)를 10,000으로 설정합니다. 이는 위성의 관측 효율성을 나타내는 값입니다.

3. arr2.ephem:

- 마지막으로, 변경된 배열 arr2의 천체 역학적 정보(ephemeris)를 조회합니다. ephem 속성은 배열 내의 각 요소(여기서는 위성이나 사이트)의 위치, 속도, 경로 등의 정보를 포함할 수 있습니다. 이 정보는 위성의 궤도 계획, 관측 스케줄링, 데이터 분석 등에 사용됩니다.

전체적인 의미: 이 코드는 천문학적 배열에서 특정 요소를 제거하고, 새로운 정지궤도 위성을 추가한 다음, 결과적으로 변경된 배열의 천체 역학적 정보를 조회하는 과정을 수행합니다. 이러한 조작은 배열의 구성을 최적화하거나 특정 관측 목표를 달성하기 위해 필요할 수 있으며, 배열의 성능과 관측 능력에 중요한 영향을 미칩니다. 변경된 배열의 ephem 정보는 위성의 정확한 위치와 운동을 파악하는 데 필수적이며, 이는 향후 관측 계획과 데이터 처리에 중요한 기초 정보를 제공합니다.

```
# plot orbit
arr2.plot_satellite_orbits(tstart_mjd=mjd_now, tstop_mjd=mjd_now+1, npoints=1000);
```



제공된 코드 arr2.plot_satellite_orbits(tstart_mjd=mjd_now, tstop_mjd=mjd_now+1, npoints=1000);는 arr2 객체의 위성 궤도를 시각화하는 작업을 수행합니다. 이 함수는 특정 시간 기간 동안 위성의 궤도를 플로팅하는데 사용되며, 여기서는 mjd_now로부터 하루 동안의 궤도를 시각화합니다. 각 파라미터의 의미는 다음과 같습니다:

```
# Copy code
arr2.plot_satellite_orbits(tstart_mjd=mjd_now, tstop_mjd=mjd_now+1, npoints=1000);
```

파라미터 설명: - tstart_mjd: 궤도를 플로팅하기 시작할 시간을 Modified Julian Date(MJD) 형식으로 지정합니다. 여기서 mjd_now는 현재 시간을 MJD 형식으로 나타냅니다.

- tstop_mjd: 궤도 플로팅을 종료할 시간을 MJD 형식으로 지정합니다. 여기서 mjd_now+1은 현재 시간으로부터 하루 후를 의미합니다.
- npoints: 플롯에 사용될 포인트의 수를 지정합니다. 여기서 1000은 궤도를 나타내는데 사용될 포인트의 총 수를 의미합니다. 이 값이 클수록 더 세밀한 궤도가 표현됩니다.

전체적인 의미:

arr2.plot_satellite_orbits 메소드는 주어진 시간 동안 위성의 궤도를 시각화합니다. 이 시각화는 위성의 궤도 경로, 위치, 운동 등을 이해하는데 도움을 줄 수 있으며, 특히 위성이나 천체 관측 계획을 수립할 때 유용할 수 있습니다. 위성의 궤도를 시각화함으로써, 해당 위성이 지정된 기간 동안 어떤 경로를 따라 움직일지, 특정 시간에 어느 위치에 있을지 등의 정보를 얻을 수 있습니다. 이러한 정보는 위성의 통신, 관측, 우주 환경 모니터링 등 다양한 목적으로 활용될 수 있습니다.

```
# generate observation and plot
# tstart = 0 # hr relative to im.mjd
# tstop = 24*50 # hr relative to im.mjd
# tadv = 1200 # s
# tint = 1200
# bw = 4.e9

obs2 = im.observe(arr2,
                  tint,
                  tadv,
                  tstart,t
                  stop,
                  bw,
                  no_elevcut_space=True,
                  dcal=True,
                  ampcal=True,
                  phasecal=True,
                  add_th_noise=False)
```

```
Generating empty observation file . . .
Producing clean visibilities from image with nfft FT . . .
Adding gain + phase errors to data and applying a priori calibration . . .
```

제공된 코드는 im.observe 메소드를 사용하여 arr2 객체에 대한 합성 관측을 생성하고, 이를 obs2라는 변수에 할당합니다. 이는 전에 주어진 파라미터들을 사용하여 특정 시간 동안의 관측을 모의하는 과정입니다. 잡음을 추가하지 않는 등의 특정 옵션들을 설정하여 관측을 생성합니다. 각 파라미터의 의미와 기능을 자세히 살펴보겠습니다:

```
python
# Copy code
obs2 = im.observe(arr2,
                  tint,
                  tadv,
                  tstart,
                  tstop,
```

```
    bw,
    no_elevcut_space=True,
    dcal=True,
    ampcal=True,
    phasecal=True,
    add_th_noise=False)
```

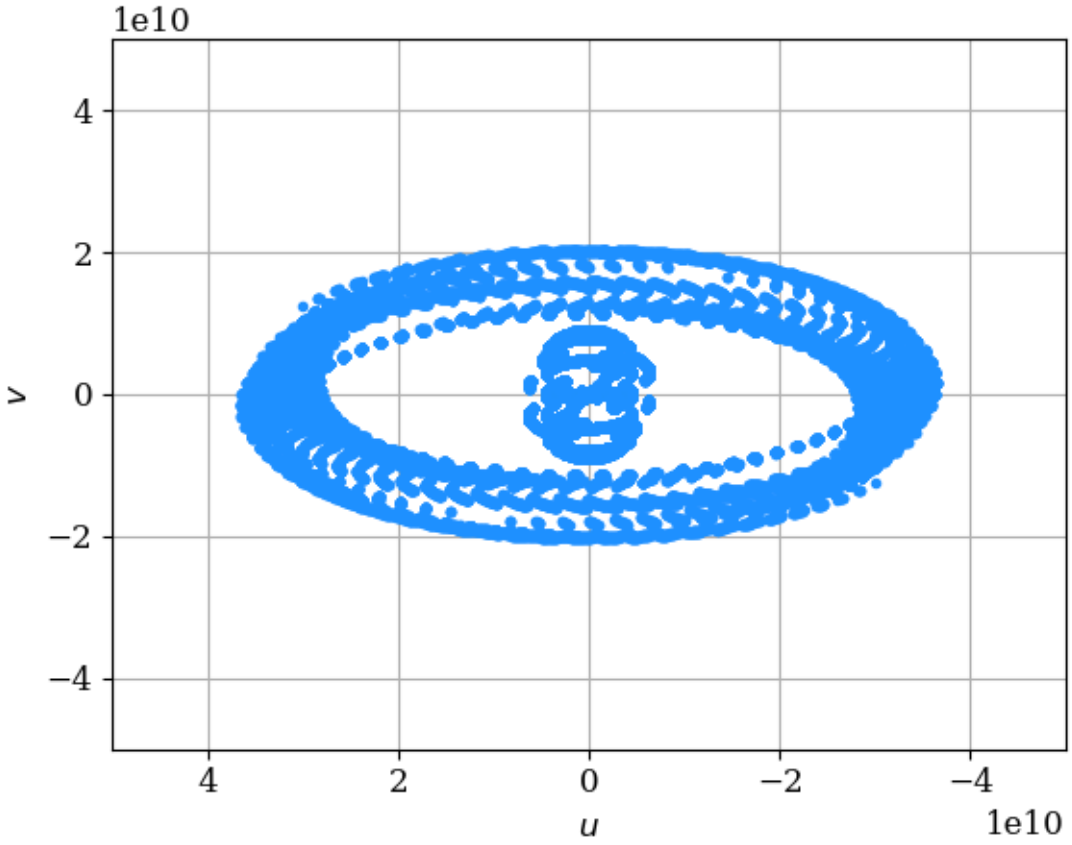
파라미터 설명:

- arr2: 관측 배열을 나타냅니다. 이전 단계에서 'TESS'를 제거하고 정지궤도 위성 'GEO'를 추가하여 수정된 배열입니다.
- tint: 각 관측의 통합 시간(integration time)을 나타냅니다. 여기서는 1200초로 설정됩니다.
- tadv: 관측 시간을 진행시키는 단계(advance time)입니다. 여기서는 1200초 간격으로 설정됩니다.
- tstart: 관측 시작 시간입니다. 여기서 0은 im.mjd 기준으로 상대적인 시작 시간을 의미합니다.
- tstop: 관측 종료 시간입니다. 여기서는 24*50시간, 즉 50일 후를 나타냅니다.
- bw: 관측의 대역폭(bandwidth)입니다. 여기서 4.e9 Hz (4 GHz)로 설정됩니다.
- no_elevcut_space: 고도 제한을 무시할지 여부를 결정합니다. True로 설정되면, 모든 고도에서 관측이 수행됩니다.
- dcal: 지연 보정을 수행할지 여부입니다.
- ampcal: 진폭 보정을 수행할지 여부입니다.
- phasecal: 위상 보정을 수행할지 여부입니다.
- add_th_noise: 관측에 열잡음(thermal noise)을 추가할지 여부를 설정합니다. 여기서는 False로 설정되어 잡음이 추가되지 않습니다.

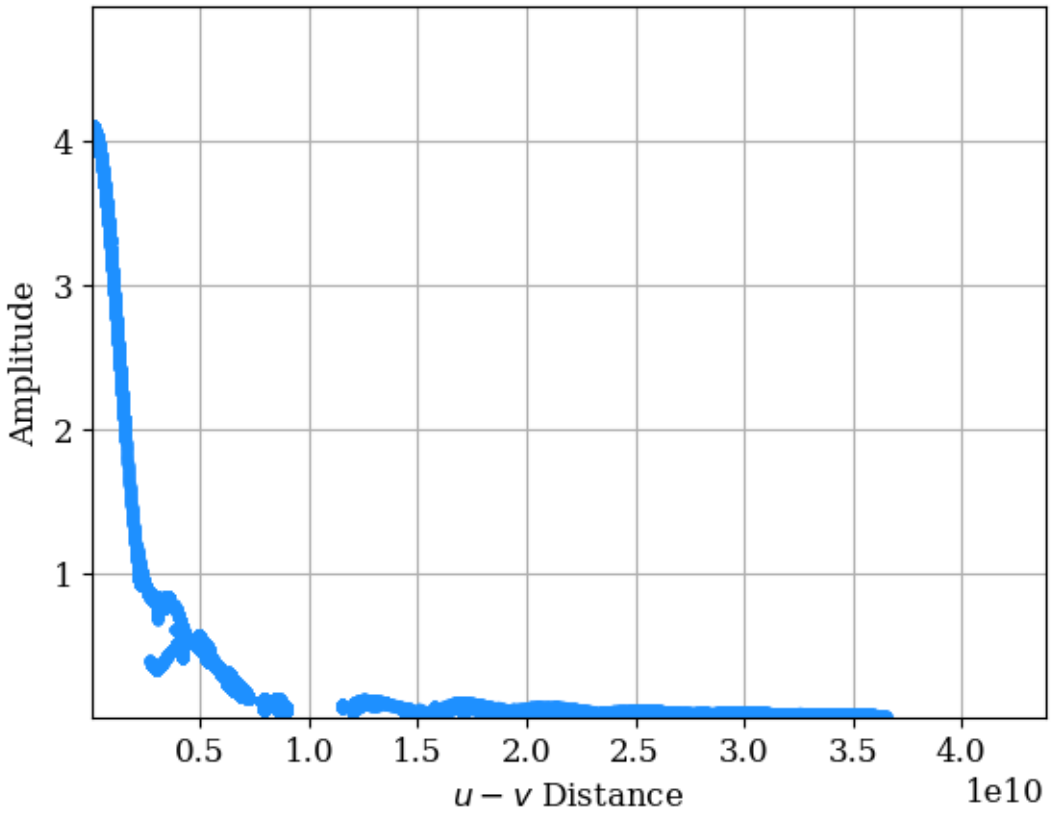
전체적인 의미:

im.observe 메소드는 관측 배열 arr2를 사용하여 지정된 시간 동안의 관측을 생성합니다. 생성된 관측은 obs2 변수에 저장되며, 이 데이터는 실제 관측을 모방한 모의 데이터입니다. 설정된 옵션들에 따라, 관측은 특정 통합 시간, 진행 시간, 대역폭 등을 가지며, 고도 제한, 지연 보정, 진폭 보정, 위상 보정 및 열잡음 추가 등의 여부에 따라 다른 특성을 가질 수 있습니다. 이러한 관측 데이터는 천문학적 연구, 관측 계획 평가, 데이터 처리 알고리즘의 개발 및 테스트 등에 사용될 수 있으며, 실제 환경에서의 관측과 유사한 조건을 제공하여 연구자들이 다양한 시나리오를 모의하고 결과를 예측할 수 있게 합니다.

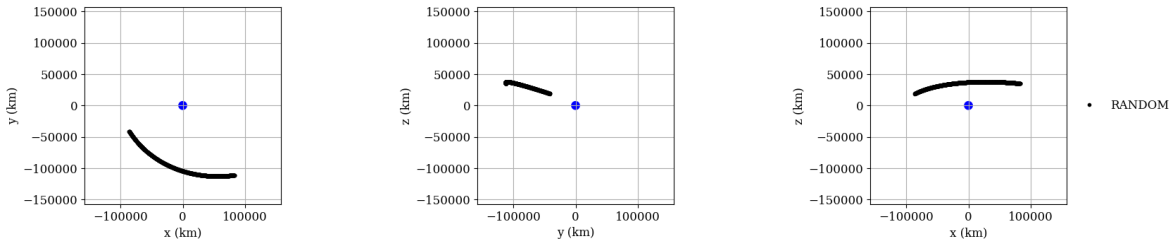
```
# plot uv coverage and visamps
obs2.plotall('u','v',conj=True, rangex=[5.e10,-5.e10],rangey=[-5.e10,5.e10]);
```




```
obs2.plotall('uvdist','amp');
```




```
# replace with a random elliptical orbiter
arr3 = arr2.remove_site('GEO')
arr3 = arr3.add_satellite_elements('RANDOM',
                                   perigee_mjd=mjd_now-2,
                                   period_days=10.2,
                                   eccentricity=0.5,
                                   inclination=20.,
                                   arg_perigee=44.,
                                   long_ascending=172.,
                                   sefd=10000)
# plot orbit
arr3.plot_satellite_orbits(tstart_mjd=mjd_now, tstop_mjd=mjd_now+1, npoints=1000);
```



제공된 코드는 먼저 arr2 배열에서 'GEO'라고 명명된 정지궤도 위성을 제거하고, 그 자리에 'RANDOM'이라 명명된 임의의 타원 궤도 위성을 추가합니다. 이후 이 변경된 배열 arr3의 위성 궤도를 시각화합니다. 각 단계별로 코드를 분석해 보겠습니다:

```
# Copy code
arr3 = arr2.remove_site('GEO')
arr3 = arr3.add_satellite_elements('RANDOM',
                           perigee_mjd=mjd_now-2,
                           period_days=10.2,
                           eccentricity=0.5,
                           inclination=20.,
                           arg_perigee=44.,
                           long_ascending=172.,
                           sefd=10000)

arr3.plot_satellite_orbits(tstart_mjd=mjd_now, tstop_mjd=mjd_now+1, npoints=1000);
```

단계별 설명:

1. arr2.remove_site('GEO'):

- 이 메소드는 arr2 배열에서 'GEO'라 명명된 사이트(또는 위성)를 제거합니다. 'GEO'는 이전 단계에서 추가된 정지궤도 위성을 나타냅니다.

2. arr3.add_satellite_elements('RANDOM', …):

- add_satellite_elements 메소드는 새로운 위성을 배열에 추가합니다. 여기서 추가되는 위성은 'RANDOM'이라고 명명되며, 그 특성은 임의의 타원 궤도 위성의 것으로 설정됩니다. 각 파라미터는 다음을 의미합니다:
- perigee_mjd=mjd_now-2: 가장 가까운 지점의 시간을 MJD 형식으로 설정합니다. 여기서는 현재 시간에서 2일 전을 나타냅니다.
- period_days=10.2: 위성의 궤도 주기를 10.2일로 설정합니다.
- eccentricity=0.5: 궤도의 이심률을 0.5로 설정하여, 타원형 궤도를 나타냅니다.
- inclination=20.: 궤도의 경사도를 20도로 설정합니다.
- arg_perigee=44.: 근지점의 인자를 44도로 설정합니다.
- long_ascending=172.: 승교점의 경도를 172도로 설정합니다.
- sefd=10000: 시스템 등가 플럭스 밀도(System Equivalent Flux Density)를 10,000으로 설정합니다.

3. arr3.plot_satellite_orbits(…):

- plot_satellite_orbits 메소드는 주어진 시간 동안 위성의 궤도를 시각화합니다. 여기서는 mjd_now로부터 하루 동안의 궤도를 시각화하며, 총 1000개의 포인트를 사용하여 궤도를 표현합니다.

전체적인 의미:

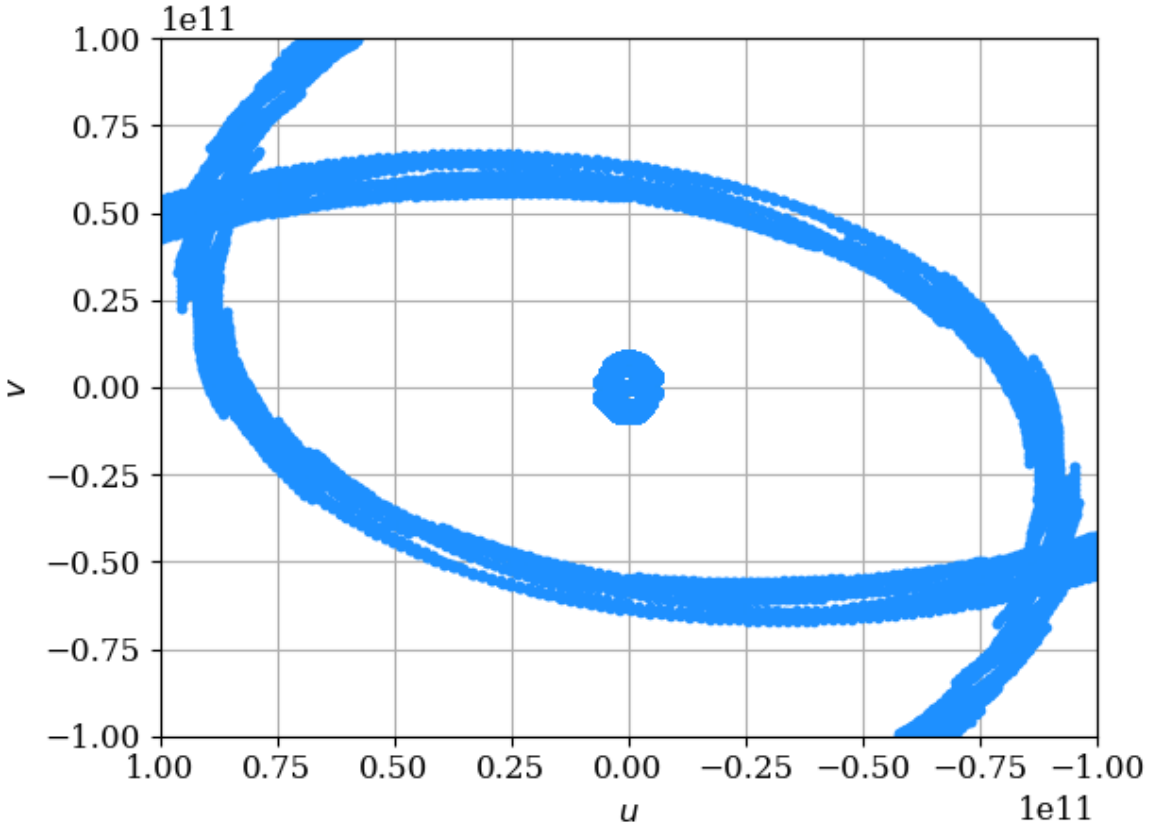
이 코드는 천문학적 배열에서 특정 위성을 제거하고, 새로운 임의의 타원 궤도 위성을 추가한 후, 결과적으로 변경된 배열의 위성 궤도를 시각화하는 과정을 수행합니다. 이러한 조작은 배열의 구성을 최적화하거나 특정 관측 목표를 달성하기 위해 필요할 수 있으며, 위성의 특성(궤도 주기, 이심률, 경사도 등)을 정확히 설정하여 특정 임무나 연구 요구사항에 맞춤화할 수 있습니다. 변경된 배열의 위성 궤도를 시각화하는 것은 위성의 경로와 위치를 이해하고, 향후 관측이나 통신 계획을 수립하는데 중요한 정보를 제공합니다.

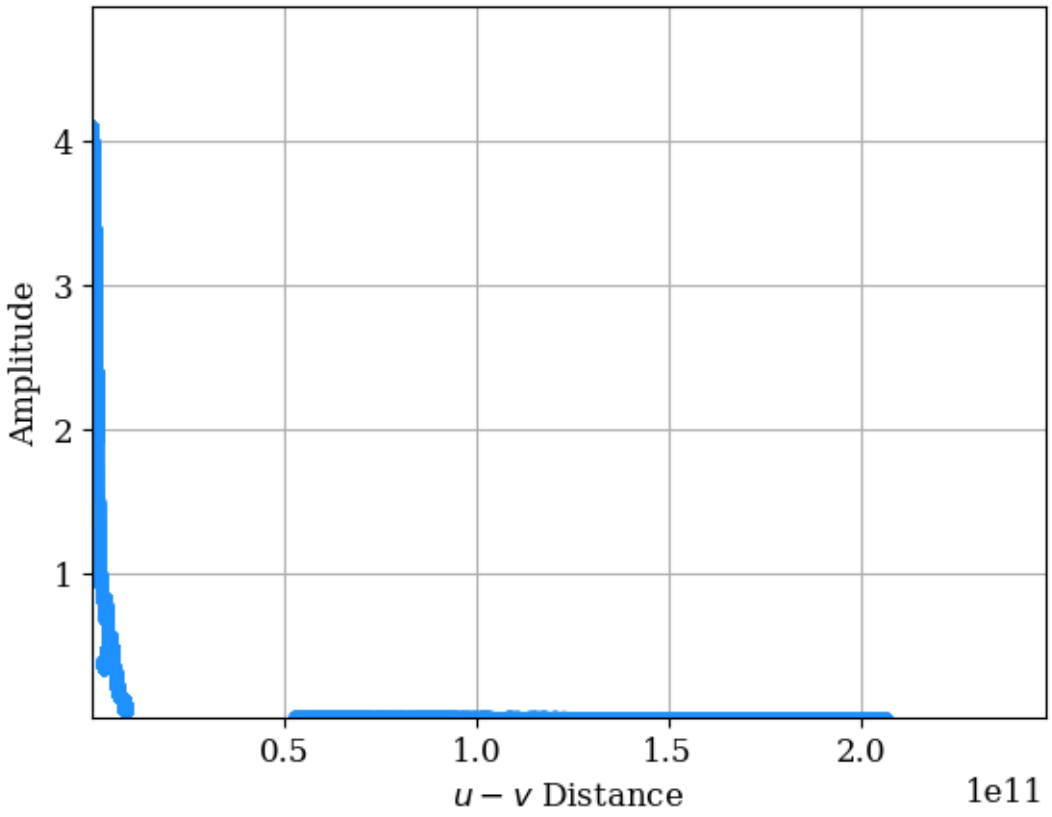
```
# generate observation and plot

# tstart = 0 # hr relative to im.mjd
# tstop = 24*50 # hr relative to im.mjd
# tadv = 1200 # s
# tint = 1200
# bw = 4.e9

obs3 = im.observe(arr3,
                  tint,
                  tadv,
                  tstart,
                  tstop,
                  bw,
                  no_elevcut_space=True,
                  dcal=True,
                  ampcal=True,
                  phasecal=True,
                  add_th_noise=False)
obs3.plotall('u','v',conj=True, rangex=[1.e11,-1.e11],rangey=[-1.e11,1.e11]);
obs3.plotall('uvdist','amp');
```

```
Generating empty observation file . . .
Producing clean visibilities from image with nfft FT . . .
Adding gain + phase errors to data and applying a priori calibration . . .
```


1e11
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
v
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
u
1e11



observe(array, tint, tadv, tstart, tstop, bw, mjd=None, timetype='UTC', polrep_obs=None, elevmin=10.0, elevmax=85.0, no_elevcut_space=False, ttype='nfft', fft_pad_factor=2, fix_theta_GMST=False, sgrscat=False, add_th_noise=True, jones=False, inv_jones=False, opacitycal=True, ampcal=True, phasecal=True, frcal=True, dcal=True, rlgaincal=True, stabilize_scan_phase=False, stabilize_scan_amp=False, neggains=False, tau=0.1, taup=0.1, gain_offset=0.1, gainp=0.1, phase_std=- 1, dterm_offset=0.05, rlratio_std=0.0, rlphase_std=0.0, sigmat=None, phasesigmat=None, rlgsigmat=None, rlpsigmat=None, caltable_path=None, seed=False, verbose=True)

Generate baselines from an array object and observe the image.

Parameters array (Array) - an array object containing sites with which to generate baselines

tint (float) - the scan integration time in seconds

tadv (float) - the uniform cadence between scans in seconds

tstart (float) - the start time of the observation in hours

tstop (float) - the end time of the observation in hours

bw (float) - the observing bandwidth in Hz

mjd (int) - the mjd of the observation, if set as different from the image mjd

timetype (str) - how to interpret tstart and tstop; either ‘GMST’ or ‘UTC’

polrep_obs (str) - ‘stokes’ or ‘circ’ sets the data polarimetric representation

elevmin (float) - station minimum elevation in degrees

elevmax (float) - station maximum elevation in degrees

no_elevcut_space (bool) - if True, do not apply elevation cut to orbiters

ttype (str) - “fast”, “nfft” or “dtft”

fft_pad_factor (float) - zero pad the image to fft_pad_factor * image size in the FFT

fix_theta_GMST (bool) - if True, stops earth rotation to sample fixed u,v

sgrscat (bool) - if True, the visibilites will be blurred by the Sgr A* kernel

add_th_noise (bool) - if True, baseline-dependent thermal noise is added

jones (bool) - if True, uses Jones matrix to apply mis-calibration effects otherwise uses old formalism without D-terms

inv_jones (bool) - if True, applies estimated inverse Jones matrix (not including random terms) to calibrate data

opacitycal (bool) - if False, time-dependent gaussian errors are added to opacities

ampcal (bool) - if False, time-dependent gaussian errors are added to station gains

phasecal (bool) - if False, time-dependent station-based random phases are added

frcal (bool) - if False, feed rotation angle terms are added to Jones matrix.

dcal (bool) - if False, time-dependent gaussian errors added to Jones matrix D-terms.

rlgaincal (bool) - if False, time-dependent gains are not equal for R and L pol

stabilize_scan_phase (bool) - if True, random phase errors are constant over scans

stabilize_scan_amp (bool) - if True, random amplitude errors are constant over scans

neggains (bool) - if True, force the applied gains to be <1

taup (float) - the fractional std. dev. of the random error on the opacities

gainp (float) - the fractional std. dev. of the random error on the gains or a dict giving one std. dev. per site

gain_offset (float) - the base gain offset at all sites, or a dict giving one gain offset per site

phase_std (float) - std. dev. of LCP phase, or a dict giving one std. dev. per site a negative value samples from uniform

dterm_offset (float) - the base std. dev. of random additive error at all sites, or a dict giving one std. dev. per site

rlratio_std (float) - the fractional std. dev. of the R/L gain offset or a dict giving one std. dev. per site

rlphase_std (float) - std. dev. of R/L phase offset, or a dict giving one std. dev. per site a negative value samples from uniform

sigmat (float) - temporal std for a Gaussian Process used to generate gains. If sigmat=None then an iid gain noise is applied.

phasesigmat (float) - temporal std for a Gaussian Process used to generate phases. If phasesigmat=None then an iid gain noise is applied.

rlgsigmat (float) - temporal std deviation for a Gaussian Process used to generate R/L gain ratios. If rlgsigmat=None then an iid gain noise is applied.

rlpsigmat (float) - temporal std deviation for a Gaussian Process used to generate R/L phase diff. If rlpsigmat=None then an iid gain noise is applied.

caltable_path (string) - If not None, path and prefix for saving the applied caltable

seed (int) - seeds the random component of the noise terms. DO NOT set to 0!

verbose (bool) - print updates and warnings