Синтез радиоизображений с помощью пакета srh_synth

Установка пакета `srh_synth` на свой компьютер

ПО для синтеза радиоизображений СРГ написано на языке программирования `Python 3`, поэтому прежде чем его использовать, вам необходимо установить интерпретатор этого языка.

Если вы синтезируете изображения на одном из серверов отдела радиоастрофизики ИСЗФ СО РАН, то Продолжите чтение инструкции с раздела Синтез радиоизображений, так как у вас уже установлены все необходимые пакеты.

Установка Python 3 в Linux

В большинстве популярных дистрибутивов Linux Интерпретор `Python 3` обычно установлен по умолчанию. Проверить установлен ли Python на вашей машине можно, запустив следующую команду в терминале:

```
python --version
```

или

```
python3 --version
```

Если `Python` установлен, то вы команда выведет версию установленного Интерпретатора. Мы рекомендуем использовать `Python 3.10` или более свежую версию. Если же в вашей

системе Python 3 не установлен, то установите его с помощью менеджера пакетов вашего дистрибутива. Например, в Ubuntu для этого нужно в терминале выполнить следующую команду.

```
sudo apt install python3
```

Установка Python 3 в Windows

Интерпретатор языка `Python` не входит в состав операционной системы Windows, поэтому его предётся устанавливать отдельно. Это можно сделать двумя способами.

1. Установить "чистый" интерпретатор Python запустив инсталятор, скачанный с официального сайта Python .
2. Установить Python в составе менеджера пакетов `Anaconda` и `Miniconda`. В первом случае вместе с интерпретаром языка будут также установлен большой набор пакетов предназначенных для научных вычислений и визуализации данных. В случае же `Miniconda` будет установлен только интерпретатор языка Python и менеджер пакетов `conda`.

Создание виртуального окружения

Мы рекомендуем устанавливать ПО для синтеза радиоизображений в отдельном виртуальном окружении (virtual environment), чтобы избужать конфликта зависимостей. В случае, если вы планируете использовать Python только для синтеза радиоизображений, этот шаг можно пропустить.

TO BE DONE

Установка пакета `srh_synth`

Пакет `srh_synth` в текущей версии использует часть функций универсального пакета `srhimages` и устанавливается вместе с ним командой:

```
pip install -U "srhimages[anfinogentov] @ git+https://git.iszf.irk.ru/fedenev/srhimages"
```

Альтернативно можно установить пакет `srhimages` с полным набором опций.

```
pip install -U "srhimages[all] @ git+https://git.iszf.irk.ru/fedenev/srhimages"
```

Тогда вместе с пакетом srh_synth будет доступен также и код для синтеза изображений `srhdata`, разрабатываемый Марией Глоба. Если вы используете виртуальное окружение, то перед запуском команд установки его нужно активировать.

Синтез радиоизображений

Синтез радиоизображений состоит из двух этапов:

1. Калибровка коэффициентов усиления антенн - функция `srh_synth.srh_calib`
2. Синтез изображений - функция `srh_synth.srh_synth` В начале работы необходимо импортировать импортировать эти функции.

```
from srh_synth import srh_synth, srh_calib
```

Калибровка коэффициентов усиления антенн

Калибровка выполняется для заданного интервала времени. Приэтом калибруются не все изображения, а токлько некоторые. По умолчанию калибруются изображения с интервалом 5 минут. Затем полученные калибровочные коэффициенты подвергаются постобработке в ходе которой устраняются пространственные сдвиги между отдельными изображениями и выравниваются амплитуды коэффициентов усиления. Полученные калибровочные данные можно затем использовать для построения изображений внутри выбранного интервала на заданных частотах с произвольной скважностью и произвольным временем накопления.

Примеры команд для калибровки:

Калибровка для частот 6 и 6.4 ГГц для одного момента времени 2024-06-01 03:00:

```
calibrations = srh_calib("2024-06-01 03:00", frequency = [6000, 6400])
```

Калибровка наблюдений за весь день для частоты 6 ГГц:

```
calibrations = srh_calib("2024-06-01 00:00","2024-06-01 12:00", frequency = [6000])
```

Калибровка наблюдений за два часа для частоты 16 ГГц:

```
calibrations = srh_calib("2024-06-01 02:00","2024-06-01 04:00", frequency = [16000])
```

Калибровка наблюдений за весь день для частоты 3 ГГц с сохранением калибровок в файл 'calib.json':

```
calibrations = srh_calib("2024-06-01 00:00","2024-06-01 12:00", frequency = [3000], write_to=
'calib.json')
```

Калибровка данных решётки 6-12 ГГЦ для одного момента времени 2024-06-01 03:00:

```
calibrations = srh_calib("2024-06-01 03:00", frequency = 'SRH0612')
```

Полный список параметров функции `srh_calib`:

- `start_time` - начало интервала времени для калибровки, например '2024-05-14 00:30'
- `stop_time` - конец интервала времени для калибровки, например '2024-05-14 03:00'
- `frequency` - список частот, которые надо откалибровать, например [3200, 4000]. Значение по умолчанию 'all' -- калибровать все частоты, 'SRH0306', 'SRH0612', 'SRH1224' -- калибровать все частоты соответвующей решётки.
- `n_proc` - Количество параллельно обрабатываемых калибровок. По умолчанию 16. Если это число больше числа имеющихся в систме процессорных ядер, то этот параметр уменьшается до числа ядер в системе. При назначении числа параллельных процессов, нужно иметь ввиду, что каждый такой процесс потребляет около 1 Гб оперативной памяти.
- `cadence`- скважность калибровок в секундах. По умолчанию калибруются данные с интервалом в 300 с. Менять не рекомендуется.
- `int_time` - Время накопления в секцндах для каждого калибровочного изображения. По умолчанию 30 с. Менять не рекомендуется
- `parallel` - запускать калибровки параллельно в дочерних процессах. По цмолчанию True.В случае установки в False, калибровки будут выполняться последовательно в основном процессе.
- `post_process` - постобработка калибровок с целью устранения 'дрожания' и 'уезжания' изображений. Кроме того выравнивается амплитуда коэффициентов усиления антенн
- `raw_dir` - директория в которой код программа будет искать сырые данные СРГ и куда их будет скачивать при необходимости. Рекомендуется для всех своих задач использовать одну и ту же директорию на достаточно емком диске.
- `db_file` - имя файла с базой данных SQLite для сохранения калибровок. По умолчанию калибровки сохраняются в файл 'calibrations.db' в текущем каталоге. Обратите внимание, что в БД создайтся запись для каждой комбинации даты наблюдения и частоты наблюдения.
- `write_to` - имя файла для сохранения калибровок в формате JSON. По умолчанию калиброви сохраняются только в локальную базу данных SQLite.
- `recalibrate` - по умолчанию `False`: программа берёт готовые калибровки из локальной БД SQLite, если они уже делались ранее. Если вы хотите пересчитать

калибровки, то установите этот параметр в `True`

Возвращаемое значение: Функция `srh_calib` возвращает словарь, в котором находятся результаты калибровки разобранные по частотным каналам и отсорторированные по времени. Данную переменную нужно использовать для синтеза изображений.

Синтез радиозображений

После выполнения калибровки можно приступать непосредственно к построению изображений.

Примеры команд синтеза рад иоизображений

Синтез изображений на двух частототах для одного момента времени с использованием калибровок, полученных функцией `srh_calib`:

```
srh_synth("2024-06-01 03:00", frequency = [6000, 6400], calibrations=calibrations)
```

Синтез серии изображений на одной часоте для заданного интервала времени с временным разрешением 60 секунд, используя калибровки из файла 'calib.json':

```
srh_synth("2024-06-01 02:30", "2024-06-01 03:30", frequency=[3000], cadence=60,
calibrations='calib.json')
```

Синтез изображений для заданного интервала времени с накоплением 30 секунд и временным разрешением 300 секунд. Изображения сохраняться в папку "srh20240601", имена сохранённых файлов будут напечатаны в консоли:

```
files = srh_synth("2024-06-01 03:00", int_time=30, cadence=300, calibrations=calibrations,
out_dir='srh20240601')
print(files)
```

Полный список параметров функции srh_synth:

- `start_time` - начало интервала времени для калибровки, например '2024-05-14 00:30'
- `stop_time` - конец интервала времени для калибровки, например '2024-05-14 03:00'
- int_time - время интегрирования для каждого изображения в секундах. По умолчанию изображения строится из одной наблюдательной записи (скана).
- `cadence` - скважность (временное разрешение) синтезируемых изображений. По умолчанию 0 - изображения строятся с максимально возможной скважностью.
- `frequency` - список частот, на которых надо построить изображения, например [3200, 4000]. Значение по умолчанию 'all' -- строить изображения на всех частотах. Данные на выбранных частотах должны быть предварительно откалиброваны

функцией `srh_calib`

- `out_dir` - путить к каталогу, куда будут записаны построенные изображения. Если в данном каталоге уже имеются файлы изображений, они будут перезаписаны при совпадении имён. По умолчанию изображения записываются в текущий каталог
- `raw_dir` - директория в которой код программа будет искать сырые данные СРГ и куда их будет скачивать при необходимости. Рекомендуется для всех своих задач использовать одну и ту же директорию на достаточно емком диске.
- **`calibrations`** - калибровочные данные, возвращаемые функцией `srh_calib`, или путь к JSON файлу с сохранёнными калибровками.
- `naxis` - размер изображения в пикселях (по умолчанию 512)
- `cdelt` - размер одного пикселя в секундах дуги (по умолчанию 5.)
- `save_RL`- сохранять изображения в правой (R) и левой (L) круговой поляризации. По умолчанию False - сохраняется интенсивность (I) и поляризация (V).
- `no_channel_dirs` - не создавать подкаталоги для каждого частотного канала. По умолчанию False - подкаталоги создаются
- `save_psf` - сохранять диаграмму нааправленности для каждого изображения в отдельный FITS файл
- `n_proc` - Количество параллельно синтезируемых изображений. По умолчанию 16. Если это число больше числа имеющихся в систме процессорных ядер, то этот параметр уменьшается до числа ядер в системе.
- `round_beam` - использовать симметричную кругулую "чистую" диаграмму направленности при "чистке" изображений. Данная опция экспериментальная. По умолчанию False. Применения этой опции приводит к уменьшению пространственного разрешения изображения.
- `high_resolution` - Искусственно увеличить пространственое разрешения изображения за счёт уменьшения размера "чистой" диаграммы направленности. Данная опция эксперментальная. Ей использование имеет смысл только для заведомо компактных источников. А интерпретация полученных изображений возможна только с оговорками.
- `compressed` - сохранять изображения в сжатом формате FITS. По умолчанию True.
- `parallel` - запускать синтез изображений параллельно в дочерних процессах. По цмолчанию True.В случае установки в False, изображения будут строится последовательно в основном процессе.
- `clean_weights` - Путь к FITS файлу с радиомзображением Солнца на низкой часте, на которое нет перекрытия порядков или двухмерный numpy массив с весами, которые будут использоваться алгоритмом CLEAN при поиске точки с максимальной интенсивностью. Массив должен иметь те же размеры, что и изображение, то есть `[naxis, naxis]`. В случае, если в этом параметре передается путь к FITS файлу, изображеие из файла будет автоматически преобразовано в веса для CLEAN и отмасштабировано для требуемых значений NAXIS и CDELT.

Возвращаемое значение: Функция `srh_synth` возвращает список имен файлов синтезированных изображений.

Revision #37
Created 2024-05-05 15:49:54 UTC by Сергей Анфиногентов
Updated 2025-05-29 07:55:01 UTC by Сергей Анфиногентов