Универсальный пакет для синтеза изображений srhimages

srhimages

Сибирский радиогелиограф (СРГ) каждый день записывает огромные объёмы данных. Чтобы получить готовые радиокарты Солнца в радиодиапазоне, сырые данные нужно откалибровать, собрать воедино из функций видности на UV-плоскости, «нарисовать» — синтезировать изображение с помощью обратного Фурье-преобразования и вычистить боковые лепестки диаграммы направленности алгоритмом CLEAN.

Пакет `srhimages` берёт на себя всю эту рутину: организует список расчётных заданий и доводит их до готовых FITS-файлов и красивых картинок. Нужно только указать требуемый интервал времени, список частот и параметры синтеза.

Что умеет пакет

- **Создаёт расчётные задания по вашему запросу.** По запросам вроде "мне нужно Солнце 5 марта с 02:00 до 04:00, каждые 15 минут, на 3000 МГц" — и получите список задач в JSON.
- **Самостоятельно скачивает данные.** Сырые файлы подтянутся с сервера в Иркутске (или с Бадар, если очень надо), ничего вручную искать не нужно.
- **Калибрует.** Коэффициенты усиления антенн берутся с сервера, а если их нет — считаются с нуля. Пакет сам выравнивает и сглаживает их суточный ход, чтобы изображение было качественным.
- **Синтезирует изображения.** На выходе — FITS-файлы в поляризациях I, V (или R, L). При необходимости FITS можно пересохранить в HDF5, в том числе со сжатием
- **Считает где угодно.** Локально на вашей машине, на кластере в Бадарах или на удалённом сервере.

- **Помогает анализировать результат.** Радиокарты можно открыть, обрезать, получить PSF, отобразить, пересчитать яркостные температуры в SFU и собрать видео из серии кадров.

Установка

Понадобится Python 3.12, `conda` и `uv`:

```
conda create -c conda-forge -n srhsynth python=3.12 uv git
conda activate srhsynth

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

Дополнительные пакеты `anfinogentov` и `globa` — это два расчётных ядра (два подхода к синтезу, разработанные в ИСЗФ). Если нужно только одно, поставьте `[anfinogentov]` или `[globa]`. На данный момент рекомендуется пользоваться кодом Сергея Анфиногентова

Если вы ставите ядро `globa` впервые, обновите данные CASA:

```
python3 -m casaconfig --update-all
```

Первый шаг - синтез 1 изображения

```
from srhimages import create_synth_tasks, run_computation

if __name__ == "__main__":
    tasks = create_synth_tasks(
        "2024-03-02 01:00:00",
        frequencies=[21000],
        average_width=1,
    )

    for result in run_computation(tasks, algorithm="anfinogentov"):
        result.to_json_file("progress.json")
```

Через несколько минут в папке `images/out` появятся готовые изображения.

> Если запускаете код из обычного `.py`-файла, оборачивайте вычисления в `if __name__ == "__main__":`. Это особенность параллельных расчётов в Python; в Jupyter это обычно не нужно.

Как это устроено

Расчётные задания

Вся работа в `srhimages` построена вокруг **расчётных заданий**. Расчётное задание — это маленький «рецепт» получения одного изображения. В нём записано:

- какие сырые файлы телескопа взять и какие сканы внутри них усреднить;
- на какой частоте и в каких поляризациях строить картинку;
- с какими параметрами: размер изображения, масштаб пикселя, нужен ли CLEAN.

Задание — обычный Python-овский словарь, который можно сохранить в JSON и посмотреть глазами:

```
{
  "task": [
    {
      "file_path": "/SRH/SRH0306/2024/03/02/file.fits",
      "scans": [0, 1, 2],
      "frequency_index": 2
    }
  ],
  "params": {
    "array": "SRH0306",
    "frequency": 3000,
    "average_mode": "visibilities",
    "output_polarizations": "IV",
    "cdelt": 4.9,
    "clean_disk": true,
    "naxis": 512,
    "compressed": true
```

```
  }
}
```

Зачем они нужны? Задание полностью описывает, что нужно посчитать, — поэтому вычисления становятся воспроизводимыми.

- список заданий можно проверить до запуска;
- сохранить в файл и перезапустить позже, на другой машине или на кластере;
- одно и то же задание прогнать разными алгоритмами синтеза;
- по ходу расчёта видеть, какие задания готовы, а какие упали с ошибкой.

Вручную их писать не нужно: `create_synth_tasks` построит задания по вашему запросу — времени, частоте и шагу по времени (cadence).

Что происходит с заданием дальше

Каждое задание проходит цепочку этапов:

1. **Калибровка** — подбор коэффициентов усиления антенн. Готовые можно взять с сервера (srhimages занимается этим сам, но можно и вытащить их самостоятельно), а можно посчитать с нуля.
2. **Постобработка** — выравнивание и сглаживание суточного хода коэффициентов усиления, чтобы картинка получилась без артефактов.
3. **Синтез** — сам расчёт изображения (сборка радиокарты в яркостных температурах + CLEAN).

За пакетной обработкой следит менеджер очередей на основе Redis: задания распределяются между воркерами, а по ходу расчёта видно, какие расчётные задания уже готовы, а какие упали с ошибкой.

Примеры

Готовые примеры лежат в каталоге `examples/` репозитория:

Пример	Про что
`basic_srh_image.py`	Одно изображение «от начала до конца», плюс работа с радиокартой.
`solar_flare.py`	Обработка солнечной вспышки в два этапа: сначала калибровки на длинном интервале, потом серия изображений с максимальным временным разрешением.
`artifacts_filesystem.py`	Сохранение и получение файлов на локальном диске.

Пример	Про что
`artifacts_http.py`	То же самое, но через HTTP-сервер артефактов.

Полезные команды:

```
srhimages create_synth_tasks --help
srhimages run_computation --help
```

Ссылки

- Исходный код: https://git.iszf.irk.ru/fedenev/srhimages
- Техническое описание API — в `README.md` репозитория.
- WIKI и Reference: https://git.iszf.irk.ru/fedenev/srhimages/-/wikis/home
- База знаний отдела радиоастрофизики: страница о пакете.

Revision #37
Created 2024-08-16 08:47:46 UTC by Виктор Феденёв
Updated 2026-08-13 14:01:54 UTC by Виктор Феденёв