Sarracen Roadmap

Terrence Tricco
Memorial University of Newfoundland
Canada

INSERT SARRACEN LOGO HERE

5th Phantom + MCFOST Users Workshop

MEMORIAL UNIVERSITY

Sarracen

- Sarracen is a Python-based analysis and visualization package for SPH.

- Hosted on PyPi (`pip install sarracen`)
- GPL 3 licence.
- Documentation hosted on readthedocs (installation, examples, API).
- Comprehensive unit tests.
- Open source – contributions welcome!

Phantom File Reading

- Sarracen can read Phantom dump files (native binary format).
- Particle data is stored in a custom pandas dataframe (SarracenDataFrame).
- Global variables are stored in a dict accessible within the data frame.

```
import sarracen
```

```
sdf, sdf_sinks = sarracen.read_phantom('dustydisc_00250')
```

```
sdf
```

	itype	iorig	x	y	z	dustfrac	vx	vy	vz	h	divv
0	1	1	24.071157	34.700990	-9.799426	0.0	-0.122988	0.084390	0.001250	1.279343	0.000132
1	1	2	-16.809586	39.822064	10.325098	0.0	-0.135606	-0.058517	-0.001010	1.344829	-0.000487
2	1	3	-31.448560	143.168881	33.471512	0.0	-0.074776	-0.015854	0.000207	3.128934	-0.000085
3	1	4	-2.028744	-149.181369	-35.725110	0.0	0.074930	-0.001240	-0.001125	3.289851	0.000012
4	1	5	-93.489736	85.529352	2.238443	0.0	-0.055908	-0.062941	0.000656	1.965504	0.000022
...	...	...	...	...	...	...	...	...	...	...	...
1799995	7	1799996	17.313110	27.298153	0.093162	0.0	-0.147175	0.091926	-0.000311	0.193278	-0.005489
1799996	7	1799997	-62.532845	-41.111989	-0.020214	0.0	0.065162	-0.093672	-0.000124	0.388128	0.000159
1799997	7	1799998	49.120059	56.648474	0.057463	0.0	-0.088019	0.071694	-0.000016	0.310863	-0.000488
1799998	7	1799999	-72.487795	34.148759	0.021874	0.0	-0.043867	-0.101104	-0.000011	0.406251	-0.000011
1799999	7	1800000	78.909543	-0.012577	0.010850	0.0	-0.003001	0.110391	-0.000018	0.357142	-0.001144

1639072 rows × 13 columns

```
sdf.params
```

```
{'nparttot': 1800000.0,
 'ntypes': 28.0,
 'npartoftype': 1200000.0,
 'npartoftype_2': 0.0,
 'npartoftype_3': 0.0,
 'npartoftype_4': 0.0,
 'npartoftype_5': 0.0,
 'npartoftype_6': 0.0,
 'npartoftype_7': 600000.0,
 'npartoftype_8': 0.0,
 'npartoftype_9': 0.0,
```

Built upon pandas

- pandas give an intuitive, performant API for slicing, re-shaping, aggregating and transforming data.

- Data is stored in custom pandas DataFrames extended with several features specific to SPH data.
  - Detection of key particle properties (smoothing length, etc).
  - Density calculation from h and positions.
  - Visualization and interpolation.

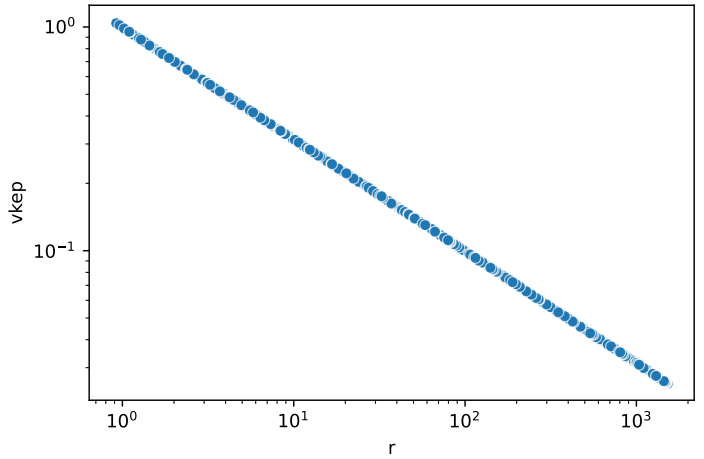
```
sdf, sdf_sinks = sarracen.read_phantom('dustydisc_00250')
```

```
central_star_mass = sdf_sinks.m[0]
```

```
sdf['r'] = np.sqrt(sdf['x']**2 + sdf['y']**2 + sdf['z']**2)
```

```
sdf['vkep'] = np.sqrt(central_star_mass / sdf['r'])
```

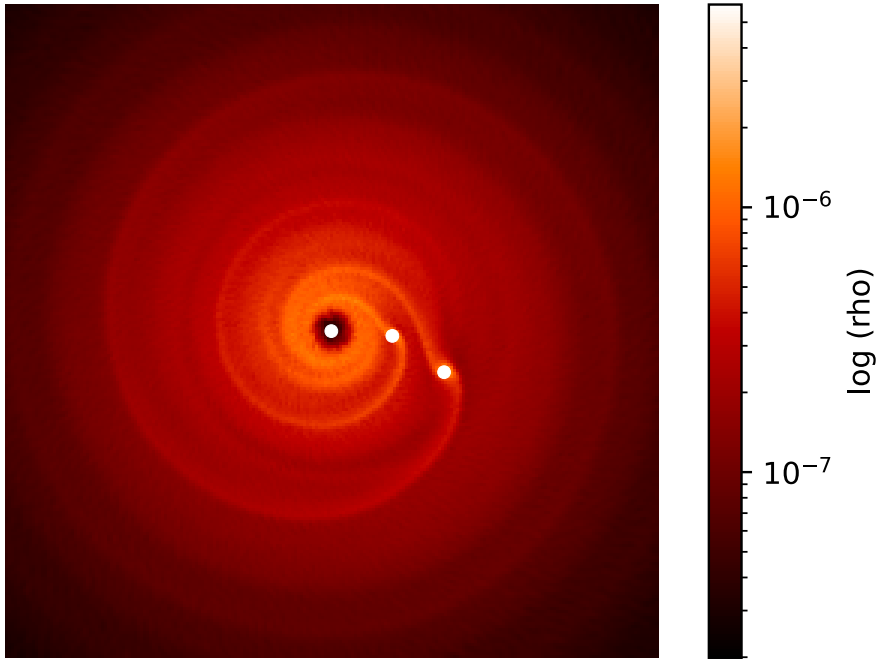
```
ax = sns.scatterplot(x='r', y='vkep', data=sdf)
ax.set_yscale('log')
ax.set_xscale('log')
```

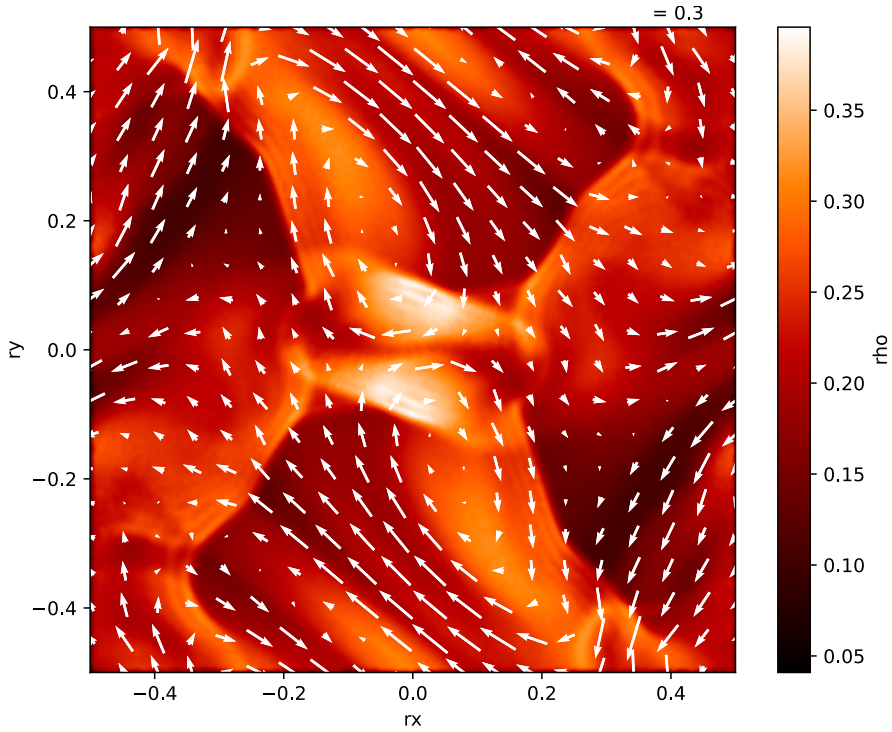


Visualization

- 4 main rendering functions:
  - `.render()`
  - `.lineplot()`
  - `.streamlines()`
  - `.arrowplot()`

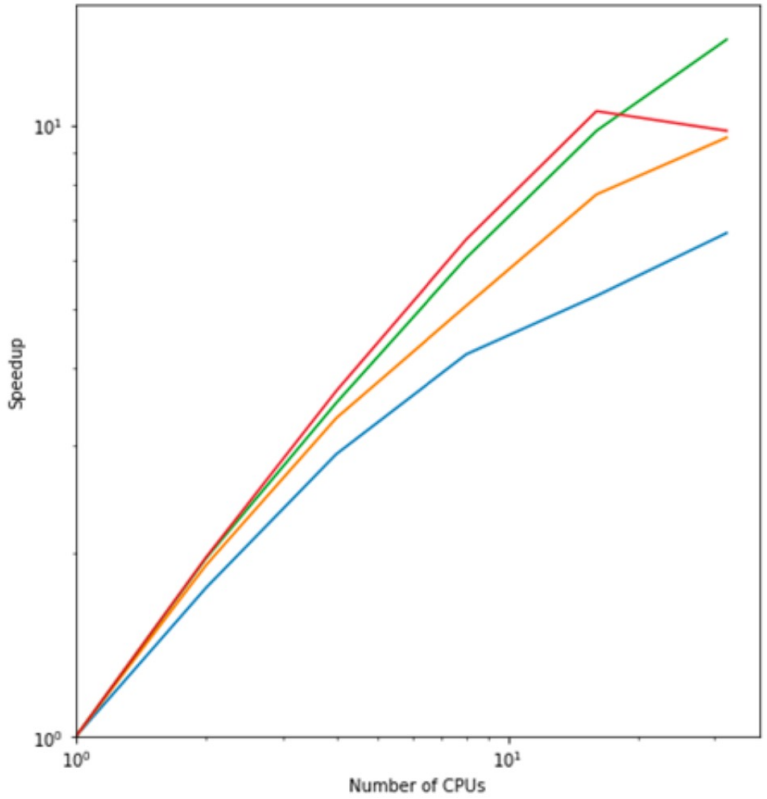
- Supports:
  - 2D/3D data,
  - planar cross-sections,
  - line of sight column-integrated views, and
  - the “exact” interpolation of Petkova+ 2018.





Performance

- Rendering functions:
  - Multi-threaded CPU or GPU.
  - Vectorized.
  - JIT compiled to machine code when first executed. (Means first time running may be a bit slower, but should be faster afterward.)



- Strong scaling measured up to ~20-30 CPU cores (particle count matters).

Current Development

- Current version is **1.2.3.**
- Fixed issues with 2-fluid dust/gas assigning correct particle masses. (*Thanks Jeremy Smallwood for the inspiration on how to fix this!*)

- **Version 1.3.0** will be a significant release with two primary features:

  1. Accretion disc analysis tools.
  2. Writing Phantom dump files.

- Targeting ~May release.

Accretion Disc Analysis

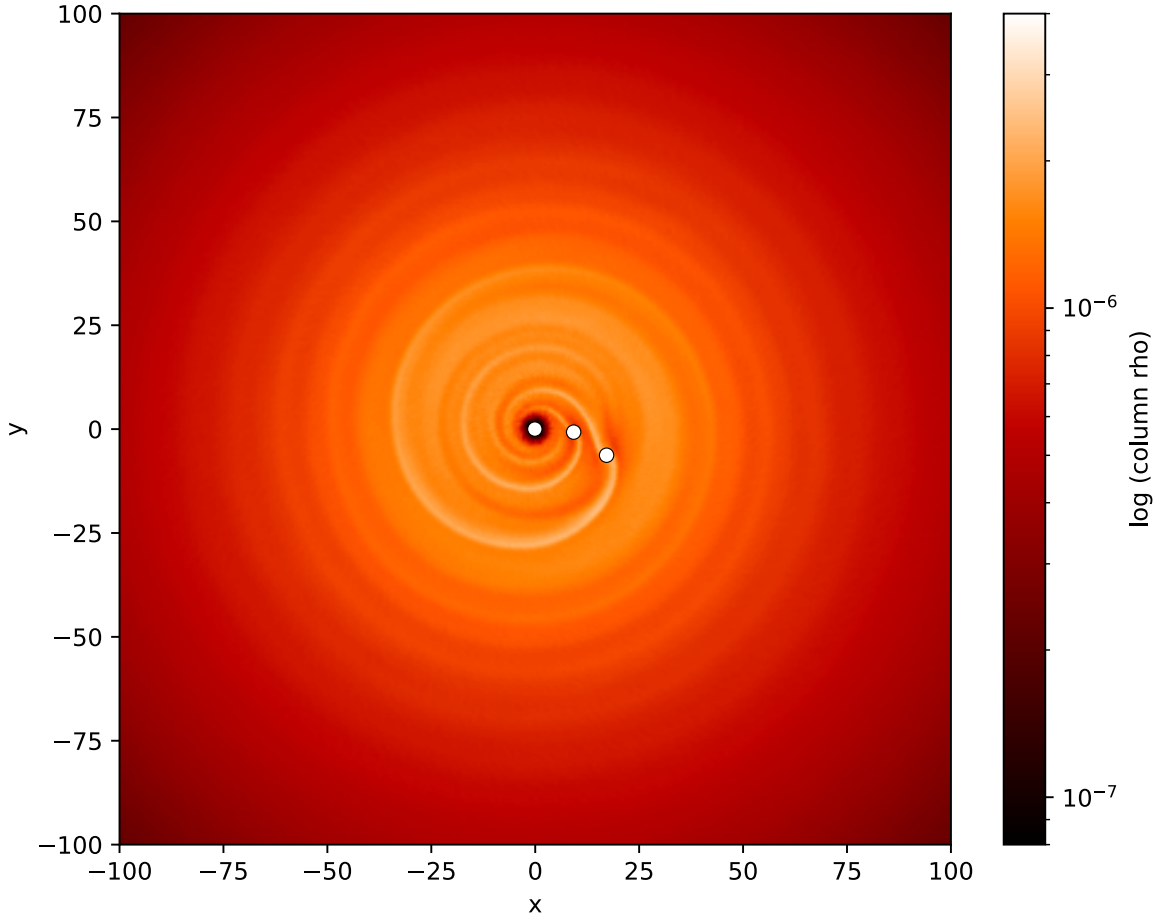
- Surface density profile
- Angular momenta profile
- Scale height, H/R
- <h> / H

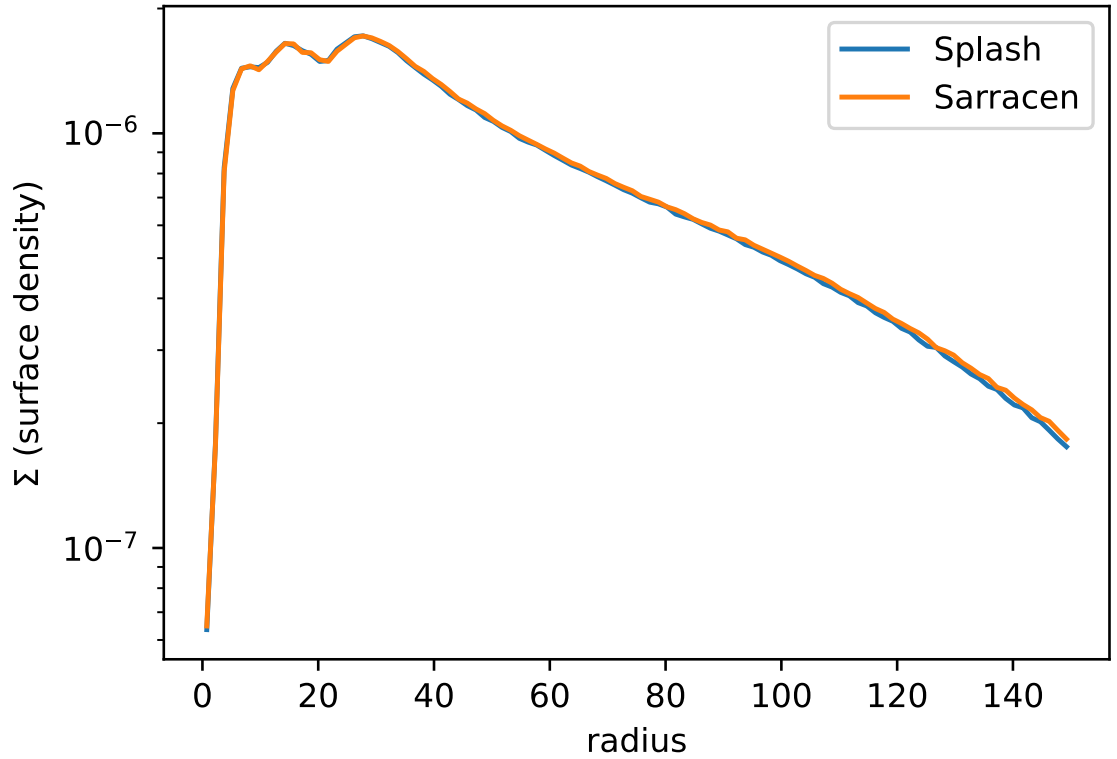
Accretion Disc Analysis

- Surface density profile — `sarracen.disc.surface_density()`
- Angular momenta profile — `sarracen.disc.angular_momentum()`
- Scale height, H/R — `sarracen.disc.scale_height()`
- <h> / H — `sarracen.disc.honH()`

- **Status:** 99% complete.
- Developed, tested, currently available on latest dev snapshot (github repo).
- May be further minor tweaks to the API, but nothing substantive.
- And need to implement proper unit tests.

Accretion Disc Analysis





Thank you Rebecca Nealon for providing prototyped code and testing the new disc analysis routines!

Writing Phantom Dump Files

- Prototype code has been developed to implement this.
- Two steps to productionize:
  1. Finalizing the API design.

     ```
     sarracen.write_phantom(data=, sinks=)
     sdf.to_phantom()
     ```

  2. Thoroughly testing that it works.

- **Status:** 10% complete.
- Would open the door for moddump and writing initial setups.

Future Roadmap

1. Physical unit conversion.
   - Have explored the package pint, but this massively slowed everything down.

Future Roadmap

1. Physical unit conversion.
   - Have explored the package pint, but this massively slowed everything down.

2. Periodic boundary support.

Future Roadmap

1. Physical unit conversion.
   - Have explored the package pint, but this massively slowed everything down.
2. Periodic boundary support.
3. Time-series analysis(?)
   - Currently can only load data sets individually.
   - Time-series analysis requires manually looping to do what you need.

Future Roadmap

1. Physical unit conversion.
   - Have explored the package pint, but this massively slowed everything down.
2. Periodic boundary support.
3. Time-series analysis(?)
   - Currently can only load data sets individually.
   - Time-series analysis requires manually looping to do what you need.
4. Automated code style error checking (linting).

Future Roadmap

1. Physical unit conversion.
   - Have explored the package pint, but this massively slowed everything down.
2. Periodic boundary support.
3. Time-series analysis(?)
   - Currently can only load data sets individually.
   - Time-series analysis requires manually looping to do what you need.
4. Automated code style error checking (linting).
5. Cool logo!

Long-term Vision

- Our goal is to implement and support commonly used analysis routines.

1. Achieve consistency of analysis across projects or between groups.
2. Reliability that analysis works correctly.
3. Provide a basis for specific, customized analyses.

- Issues requesting features are welcomed.

Summary

Harris & Tricco, *Journal of Open Source Software*, 2023.
https://github.com/ttricco/sarracen

INSERT SARRACEN LOGO HERE

- **Disc analysis tools are available now** in the dev version.
- **Version 1.3.0** will release disc analysis tools + **Phantom file writing**. Anticipating release in the next few months.

- Get involved by reporting bugs or suggesting new features.
- We use our **issue tracker** extensively as part of our development cycle.
- PRs for code or documentation submissions are welcome.

- Thanks to Andrew Harris and everyone who has contributed to Sarracen in some way!