Jax + Gammapy

Hacking / co-working session

Motivation

- Currently Gammapy analyses can still run on a single laptop

- However in future we will get:

  - Deep observations (e.g. O(100) observations for Galactic Center with CTA). A joint fit with a few hundred parameters (multiple sources, background norms, other systematics)

  - Event classes, so joint fits scale by the factor of event classes

  - Asymmetric PSF, where a kernel map can be >1GB

  - PSF changing in the FoV: 3d convolution with a kernel that depends on position. Requires GPUs for efficient computations.

- Also the number of parameters grow, so we shook profit from the experience in Machine learning and Deep learning

- Parallelization, GPU support and differentiable programming

- This is the only way to get **scalable analyses**
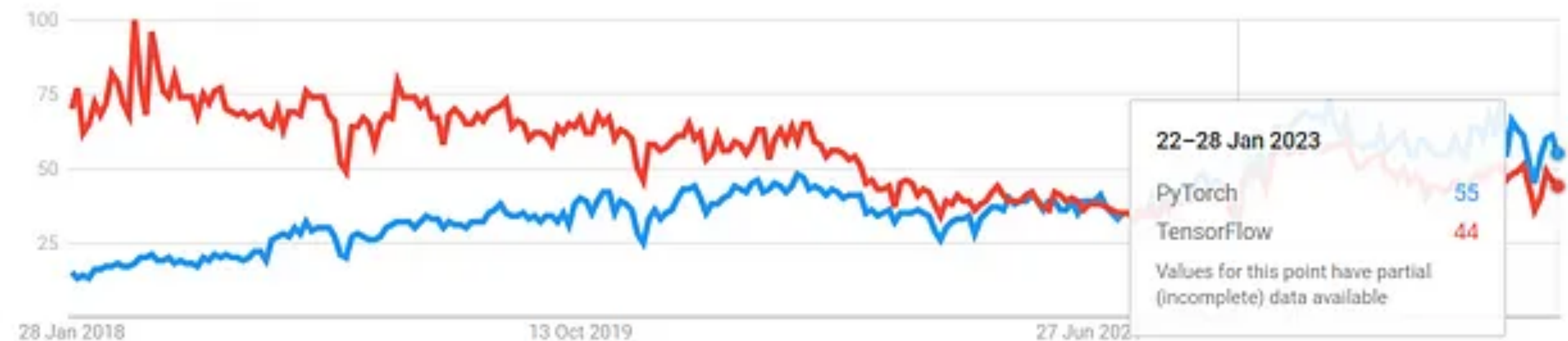
Deep learning libraries

PyTorch
- Nicer API for deep learning
- Easier to debug
- By now the most popular
- Support by the Python Software Foundation

JAX
- No high level API
- Composable function transforms, pmap, vmap, jit, etc
- Very popular in science, became very popular in past 5 years
- Best performance…
- Supported by Google, but could be https://killedbygoogle.com/

TensorFlow
- First large project
- Used to be very popular, but PyTorch won in the past 5 yrs

22–28 Jan 2023
PyTorch   55
TensorFlow   44
Values for this point have partial (incomplete) data available

28 Jan 2018   13 Oct 2019   27 Jun 202

Python Array API

- Standard for array APIs in Python. Allows to write array based code, that is agnostic to the backend

- Numpy 2.0, PyTorch, Jax, TensorFlow, Dask, xarray, Cupy, etc. agreed to conform to the standard

- Would be good intermediate step for Gammapy. Then we could experiment with different backends and see which one works best.

- However the array API has limited functionality.

- It supports devices, but in practice one would need more specific code that might not be agnostic to the backend anymore. So I think the idea is in practice less attractive…

- https://data-apis.org/array-api/latest/

Some Reasons for Jax…

- Most popular in the scientific community (tinygp, equinox, lineax, …)

- Numpy and Scipy compatible API, "drop in" replacement.

- Has limitations: no in-place modifications of arrays, more strict typing, limited type promotion for certain operations. **But:** this is required to support parallelization and best performance.

- Steep learning curve: many advanced concepts such as composable functions transform, Pytrees, custom GPU kernels in Pallas, array sharding, but beginning is drop in replacement for Numpy

- Overall best performance…

- Suggest a hacking session for Gammapy with Jax

	Batch size	Keras 2 (TensorFlow)	Keras 3 (TensorFlow)	Keras 3 (JAX)	Keras 3 (PyTorch) (eager)	Keras 3 (best)
SegmentAnything (fit)	1	386.93	**355.25**	361.69	1,388.87	**355.25**
SegmentAnything (predict)	4	1,859.27	438.50	**376.34**	1,720.96	**376.34**
Stable Diffusion (fit)	8	1,023.21	392.24	**391.21**	823.44	**391.21**
Stable Diffusion (predict)	13	649.71	**616.04**	627.27	1,337.17	**616.04**
BERT (fit)	32	486.00	**214.49**	222.37	808.68	**214.49**
BERT (predict)	256	470.12	466.01	**418.72**	1,865.98	**418.72**
Gemma (fit)	8	NA	232.52	273.67	525.15	**232.52**
Gemma (generate)	32	NA	1,134.91	**1,128.21**	7,952.67*	**1,128.21**
Gemma (generate)	1	NA	758.57	**703.46**	7,649.40*	**703.46**
Mistral (fit)	8	NA	**185.92**	213.22	452.12	**185.92**
Mistral (generate)	32	NA	966.06	**957.25**	10,932.59*	**957.25**
Mistral (generate)	1	NA	743.28	**679.30**	11,054.67*	**679.30**

Taken from: https://keras.io/getting_started/benchmarks/

Jax Hacking Session

- Two possible approaches:

  - Make a Gammapy branch and literally change to `import jax.numpy as xnp`

  - Start from scratch and try to get a minimal Gammapy analysis running in Jax, with some use go pmap and jit

- Suggest Tuesday 2pm: possibly two groups. I will do 10 - 15 min more detailed introduction…