github.com/jax-ml/jax

docs.jax.dev

This tutorial

github.com/JAXtronomy/tutorials

Intro-to-JAX.ipynb

JAX is a general numerical computing library for Python

Combines a familiar Pythonic (numpy/scipy/etc.) interface with:

- Just-in-Time (JIT) compilation
- Auto-differentiation
- Support for multiple hardware devices (CPU/GPU/…PU)

Developed mainly in house at Google Research by a team of software engineers (primarily to support machine learning efforts)

Not a Google product, so support for the broader research community

JAX: Quick Demo

Notebook: Intro-to-JAX.ipynb

JAX Layers

`jax & jax.numpy:` high-level API with ducktyping, Pythonic interface

`jax.lax:` stricter, lower-level (still Python) interface; closer to XLA

XLA (Accelerated Linear Algebra): transforms operations into optimized low-level code to be performant on different hardware

GPU / CPU / TPU / etc.: hardware backends targeted by XLA

🗡️ A few sharp bits

JAX works best with functional programming:

- Pure functions: no side effects or changing persistent state, as often happens with OOP
- Immutable data structures: Bad to modify array elements in place (but it is possible)

Control flow (e.g., `if` statements) may need to be handled with lower level functions in `jax.lax` (vs. writing plain if statements)

Autodiff direction can matter for performance

- Forward mode: generally better when num inputs < num outputs
- Backward mode: use when num outputs < num inputs

🗡️ A few more sharp bits (especially for astronomers)

Astropy and much astronomy-specific code won’t work out of the box with JAX

OOP code that relies heavily on modifying state may be dangerous with JAX

Can’t use wrapped C/C++/Fortran code out of the box (but can connect them!)

I/O needs to be cleanly separated from JAX code

Random number generation is subtly different than Numpy

Some recommendations

I can be hard or tedious to refactor an existing project/package to use JAX

When you start a new project: Commit to try using JAX from the beginning!

Separate JAX components from Python-only (file I/O, astropy, …)

Other Resources

The JAX documentation is *excellent* and improves regularly

Some key pages:

- Thinking in JAX
- The Sharp Bits (a more exhaustive list than covered here)

Many active community projects: flax, equinox, optax, numpyro, diffrax, galax, …

Building some Astropy functionality: unxt, coordinax

Most are in rapid development phase — use with care!

Future topics

- Using GPUs with JAX
- More advanced autodiff / grad
- Using physical units in JAX code (with unxt)
- Writing probabilistic models with numpyro
- …