DIANA-HEP

Advisory Board Meeting - Princeton

Peter Elmer, David Lange, Jim Pivarski

January 26, 2017

Peter Elmer

David Lange

Jim Pivarski

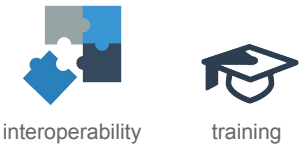
Broad range of related software projects

- In the scientific Python ecosystem
  - Scikit-HEP
  - Python toolkit in CMS
  - pip-based package management

- In the Spark ecosystem
  - Porting analyses to Apache Spark
  - Spark-ROOT

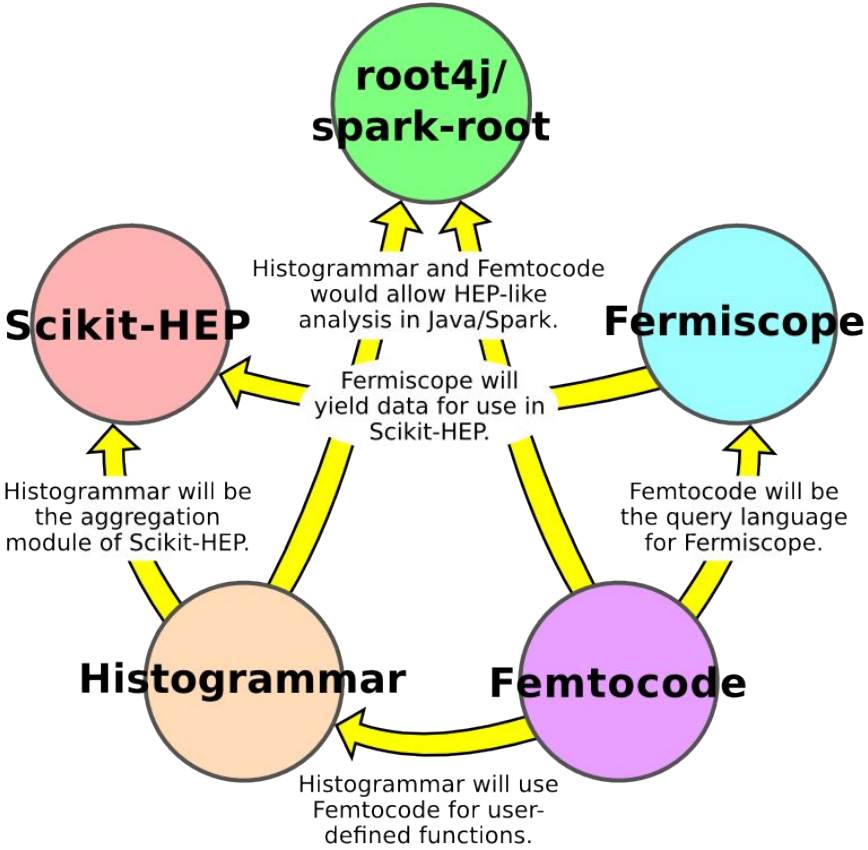
- Novel analysis tools
  - Histogrammar
  - Femtocode

- Training
  - Evangelizing use of Python, Pandas, Scikit-Learn, Jupyter (David)
  - Talks/tutorials on Spark, TensorFlow, and “Big Data” (Jim)



Distribution of interests

- Many different projects, all related and reinforcing one another.
- Diverse research directions, following best practices for R&D.
- Team emphasis on outside connections. Want to establish software development *communities*.



Collaboration within and outside of DIANA-HEP

DIANA-HEP collaborators

- **Scikit-HEP:** Eduardo (Cincinnati)
- **Analysis language paper:** Brian (Nebraska)
- **HistFitter/Histogrammar:** Kyle, Lukas (NYU)

Outside collaborators

- **Scikit-HEP:** Noel Dawe (Melbourne), Vanya Belyaev (ITEP), Sasha Mazurov (Birmingham)
- **CMS Big Data:** Oliver Gutsche, Matteo Cremonesi, Nhan Tran, Jim Kowalkowski, Saba Sehrish (Fermilab)
- **Spark-ROOT:** Viktor Khristenko (Iowa), ROOT Team, CERN IT
- **Histogrammar:** Alexey Svyatkovskiy (Princeton)
- **Femtocode/Fermiscope:** Jin Chang, Igor Mandrichenko (Fermilab) and Peter Hansen (Minnesota)
- **Presentations in industry:** Strange Loop, KDD, CHUG **and interactions with:** Wes McKinney (Pandas), Julien Le Dem (Parquet), Michael Armbrust (SparkSQL)

Descriptions of each project

Scikit-HEP

- Collaboration with the authors of rootpy (Noel Dawe) and Ostap (Vanya Belyaev, Sasha Mazurov), two widely used Pythonic interfaces to ROOT, RooFit, and RooStats.
- Intended to be a general bridge between HEP software products and the scientific Python ecosystem: Numpy, Scipy, Scikit-Learn, Keras, etc.
- Not starting from scratch: combining the strengths of rootpy and Ostap with a unified software design.
  - Generator, Dataset, Aggregation, Modeling (fitting and machine learning), Visualization.
- Will draw in other packages later, following the model of Astropy.
  - Common definitions and APIs so that existing code can be made to talk to each other.

Spark-ROOT

- Bridge between ROOT and Spark: collections of ROOT files become a Spark DataFrame.
  - Directly connects HEP data with the Big Data world— everything that has been wired into Spark becomes available to physicists: machine learning, in-memory analytics, file formats.
  - Nice match between ROOT concepts and Spark concepts (such as columnar reading).
- Working closely with physicist users, CERN's IT team (setting up a Spark cluster), and the ROOT team.
- Pure-Java reader based on Tony Johnson's FreeHEP (SLAC) and intensive development and testing by Viktor Khristenko (Iowa).
- Nearing maturity: reading complex events from HDFS and EOS, in parallel on the CERN-Spark cluster, including real data for two CMS analyses.

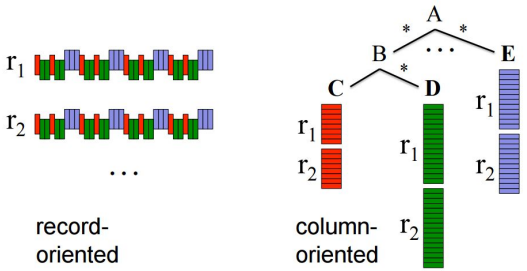
Histogrammar

histo·*grammar*
/histōˌˈgɹæm.ər/

- Started as a way to match Spark's functional programming interface (for parallelism) with HEP-style histograms.
- Became a generalization of the concept of histogramming as structured aggregation, a language of aggregation primitives that can be combined for an open-ended set of tasks.
  - Many analysis tasks that are currently being forced into 1, 2, and 3-dimensional histograms can be simplified using Histogrammar.
  - Data collection and plotting in ROOT, Matplotlib, Spark, and GPU environments.
  - Used in two CMS analyses and a political science data mining project (and possibly others).
- Working with Kyle and Lukas to see if Histogrammar could power HistFactory, generalizing HistFactory's abilities and broadening Histogrammar's impact.

Femtocode query language and Fermiscope service

- Ambitious project to replace private skims with a queryable server.
  - Private skims introduce provenance, version control, and resource use problems.
  - Use of private skims is partly cultural: other fields query from terabytes of data in real time.
  - Orders of magnitude speedup is possible, but requires fundamentally new techniques.
- Inspired by Dremel/Drill, Ibis, Impala, Kudu: fast SQL engines based on columnar data, extended for the HEP use case by allowing complex, structured events.



- Femtocode converts object-oriented queries into vectorized kernel functions, Fermiscope is a database/cache to keep the most popular columns in memory.
- Fermiscope development: Jin Chang and Igor Mandrichenko (FNAL LDRD).

Engaging user communities

Developing an “evangelization plan”

- **Scikit-HEP:** rootpy and Ostap have active communities; provide tutorials for *transition*. (Original packages will be frozen at v1.0.)
- **Spark-ROOT:** infrastructure component; the problem will be introducing physicists to Spark and its advantages.
- **Histogrammar:** hard to interest physicists as a standalone tool; integrating into HistFactory, Scikit-HEP, and Femtocode.
- **Femtocode/Fermiscope:** fundamentally new way to get data, but simplicity and performance should be good selling points.

Engaging users is an important problem and the development of an evangelization plan will be a major focus in the next six months.

Plans for the next year

Six and twelve month plans

SIX MONTHS

- Crystalize the design of Scikit-HEP.
- Finalize Spark-ROOT with documentation.
- Attract more HEP analyses to use Spark-ROOT and Histogrammar.
- Investigate inclusion of Histogrammar into HistFactory and therefore RooFit.
- Develop “demo quality” Femtocode/Fermiscope with an emphasis on research.
- Develop a concrete plan for software evangelism.

TWELVE MONTHS

- Evangelize Scikit-HEP and Spark-ROOT as a starting point for HEP analysis in Python and Spark, respectively.
- Fully integrate Histogrammar into HistFactory.
- Develop Femtocode/Fermiscope as a usable tool with a trial set of users. Further development into a production-ready system would follow next year (2018).

Backup slides

astropy

A Community Python Library for Astronomy

The Astropy Project is a community effort to develop a single core package for Astronomy in Python and foster interoperability between Python astronomy packages.

Current Documentation | Other Docs ▾

Current Version: 1.3

Please remember to acknowledge the use of Astropy!

Install Astropy

OS X | Linux | Windows | </> Source | Developer

There are a number of options for installing the astropy package on MacOS X. Astropy can be installed using the MacPorts or Fink package managers, and is also included by default in the Anaconda Python Distribution (more details here), Enthought Canopy, and AstroConda, which provide an easy way to get set up with a scientific Python distribution. MacPorts usually includes new releases almost immediately, but Anaconda and Canopy may not always include the latest version.