CosmoSIS: Modular cosmological parameter estimation

https://bitbucket.org/joezuntz/cosmosis/wiki/Home

Alessandro Manzotti
Sarah Bridle, Scott Dodelson, Elise Jennings,
Jim Kowalkowski, Marc Paterno,
Doug Rudd, Saba Sehrish, Joe Zuntz

CHEP 2015
Cosmology in the Era of Big Data

**BIG DATASET**

Past

15,000 galaxies

Present-future

> 1 Million

**BIG COLLABORATION**

Past

Present-future

**SOFTWARE**

Codes developed individually, in different languages. The output is shared.
**Collaborative Development Differences**

**Cosmologists** work on their own much more often

- much important code developed by very small groups
- **each** individual or **group** chooses **programming language**, tools, etc. (Python, Fortran, C are most common)
- no central management of software is possible
- collaboration is often informal

**HEP collaborations** have strong control over their process

- define **choice** of **programming language** (almost all C++)
- single framework used for most development
- centrally managed software
- requires strong control over member of the collaboration

CosmoSIS has to live **within** the **demands** of the **cosmology** community
CosmoSIS: Modular framework for parameter estimation.

Example: turn supernovae brightness into constraints on cosmological model parameters.

Consolidate and **connect** together existing codes

Enhance **collaboration**, and development of new algorithm in different languages

Make it **easy** to deploy, configure, and run **across** all supported platforms

**Fast** and able to run on HPC cluster

See:
Physics Analysis Software Framework for Belle II
M. STARIĆ et al.
CosmoSIS:

**Modular** framework for parameter estimation.

Example: turn supernovae brightness into constraints on cosmological model parameters.

Consolidate and **connect** together **existing codes**

Enhance **collaboration**, and development of new algorithm in different languages

Make it **easy** to deploy, configure, and run across **all supported platforms**

**Fast** and able to run on HPC **cluster**

**Nothing** about the framework is **cosmology** - specific

See: Physics Analysis Software Framework for Belle II M. STARIC et al.
**OUR MODULAR SOLUTION: COSMOSIS**

- **APIs:** Define module configuration, how to obtain their input data, how they interact with the framework, how outputs are organized, and how new modular components can be added.

- **Runtime environment:** To reliably configure and run programs that use these modules.

- **Development environment:** To write, build, and test analysis software, and which makes it easy to share what they have developed.

- **Distribution system:** To make it easy to install the code and ensure compatibility.

  Registration of attribution information (required citations, etc.) for use of any contributed code.
MODULARITY IS THE KEY

We provide commonly used sampler and modules but it is oriented to users contribution.

CosmoSIS main

Likelihood of model given data

Physics module A → Physics module B → Physics module C

Likelihood Calculator

Observables

CosmoSIS Framework
Standard library
Control relationship
Flow of data
COSMOSIS STRUCTURE

Software tools: e.g. gcc, g++, gfortran, Python, SciPy, fftw, gsl, NumPy, cfitsio,

External contributions are welcomed. We are happy to include them after internal review.

Publicly available
CosmoSIS

Private or collaboration libraries

Dependencies

CosmoSIS:
- Core libraries
- Infrastructure for modules
- Samplers

Dark Energy Survey specific modules

Collaboration modules

Other user’s physics

Standard Library
CAMB, Planck, WMAP, BICEP2, CFHTLens, BOSS
DataBlock passed to each module (similar to a HEP event)

Modules

Note: the result from event $n$ (the likelihood) influences what happens in event $n+1$ (MCMC process)

MATTER content of universe

Output

Clustering

Likelihood

also B.A.T talk by F. BEAUJEAN
Looking for a small Gaussian signal on top of a falling exponential background. We use a binned likelihood, Poisson statistics, and we integrate the cross section dependency across each mass bin.
COSMOSIS: A COMPLETE TOOLKIT

- CosmoSIS parallelism with OpenMP and MPI
  - develop a program on your laptop
  - without change, run using thousands of cores on an HPC cluster

- Tools for diagnosis of convergence, thinning, etc.
  - Gelman-Rubin statistic, auto-correlation length test
  - Continue sampling from a previous chain

- Tools for analysis of posterior densities
  - single parameters and two-parameter posterior density plots
  - Basic statistic of the chains and covariances

- Integration with diverge community supported codes
def execute(block, cfg):
    # Read this sample's parameters from the block
    lum = block['params', 'lum']
xsecbg = block['params', 'xsecbg']
beta = block['params', 'beta']
xsecsig = block['params', 'xsecsig']
mu = block['params', 'mu']
sd = block['params', 'sd']

    # Calculate the expected counts in each bin corresponding to this
    # sample's parameters
    lows = cfg.loewedges
    highs = cfg.loewedges + cfg.binwidth
    f1 = np.exp(-1.0 * lows / beta)
f2 = np.exp(-1.0 * highs / beta)
expected_bkg = lum * xsecbg * (f1 - f2)

    sqrt2sigma = np.sqrt(2.0)*sd
    g1 = special.erf( (mu-lows)/sqrt2sigma )
g2 = special.erf( (mu-highs)/sqrt2sigma )
expected_signal = lum * xsecsig * (g1 - g2) / 2.0

    expected_counts = expected_signal + expected_bkg

    # Now calculate the log-likelihood for our data, given the
    # expectation for this sample
    loglike = np.sum(-expected_counts + cfg.counts + np.log(expected_counts) - cfg.infactcounts)

    block['likes', 'BUMP_HUNT_LIKE'] = loglike
    return 0
**SOME LESSONS LEARNED (OR RE-LEARNED)**

**Cosmologists are learning.** We see the value and feasibility of well-controlled software, with strong control over versioning and binary compatibility.

**Contribution and sharing increases.** Open-source model for contribution of modules (with attribution for work) has helped attract interest in sharing code.

**HEP might consider** adopting a similar attribution concept to help encourage sharing (and rewarding the developers of) useful software. Multi-language systems lower the bar on programming expertise for contribution.

https://bitbucket.org/joezuntz/cosmosis/wiki/Home
KEEP CALM and BE MODULAR

https://bitbucket.org/joezuntz/cosmosis/wiki/Home
Additional temp Slides.