DESC BAYESIAN PIPELINE TOPICAL TEAM & JAX-COSMO

- Introduction to the DESC Bayesian pipeline topical team
- Introduction to the jax-cosmo library
- DES Y1 3x2pts exercise done with jax-cosmo

J.E Campagne

LSST-France May 2022

Thanks to A. Boucraud & F. Lanusse

DESC BAYESIAN PIPELINE TOPICAL TEAM

Goals: Foster *community-wide* momentum on Bayesian Forward Modelling, and develop an *open-source and extensible* end-to-end Bayesian pipeline for DESC and the wider community (ie. use in other exp.)

Technically hosted under the DESC WL Working Group but aims to be very transverse. Intersections with LSS, PZ, CL, MCP are natural, expected, and encouraged

on slack
#desc-bayesian-pipelines-tt

Bayesian Forward Modeling Seminars

As part of the recently created LSST DESC Bayesian Pipelines Topical Team, we are organizing a series of community-wide open seminars on topics related to Bayesian Forward Modeling for the analysis of LSST data. The goal of these seminars, *open to anyone*, is to foster discussions of various modeling and technical aspects of a full end-to-end Bayesian pipeline.

https://lsstdesc.org/bayesianpipelineseminars/

WHY BAYESIAN PIPELINES?

Precursor studies and proofs of concepts have demonstrated that in principle Bayesian modelling of various stages of the survey is possible.

if you can actually solve it, a Bayesian Forward Modelling approach is a good idea for a number of reasons:

- Avoid summary statistics to deal with (Shear) Map directly w/o loss of information:
  eg. Replace "Shear Map-> Pk/ξ -> Likelihood" by "Cosmo+Model-> shear map -> Likelihood"
- Introduction of systematics (eg. mask) directly at shear map level and avoid computation of systematics at summary statistics level
- Takes into account of covariance combining different probes naturally avoid covariance matrix computation for summary statistics
- As a by-product some intermediate variables are estimated: eg. 3D matter distribution along line-of-sight

WHY BAYESIAN PIPELINES?

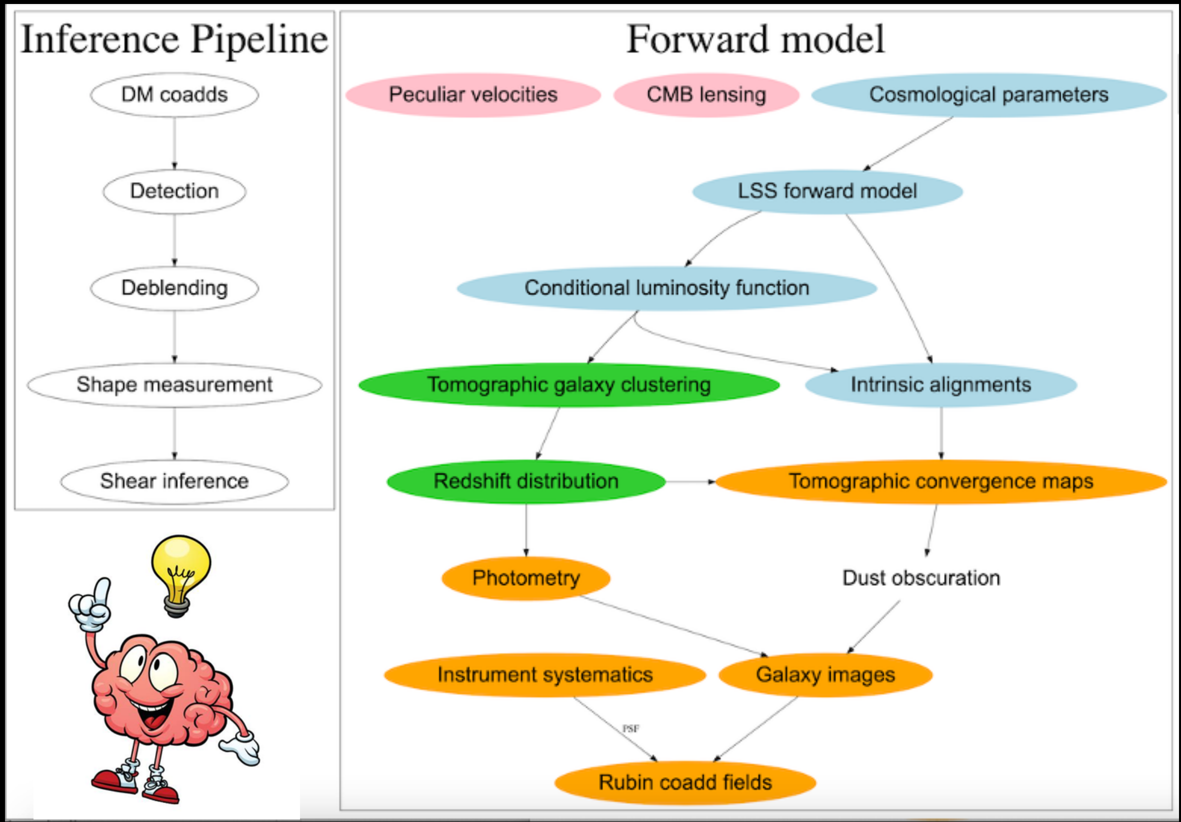
"if" you can actually solve it…

The counterpart is an extremely challenging task:

- High-dimensional inference over hierarchical models is just very hard
- You need fast, high accuracy and most likely differentiable forward models (which can include a full approximate N-body simulations)

Very few tools exist and even fewer actual analyses have been performed so far as people have mostly concentrated on trying to demonstrate that these hard problems can be solved.

PROJECTS/DREAMS



Cosmological parameter project

Pixel analysis project

Interfaces

ROAD MAP

In preparation

- **Pixels to shear**
- Demonstrate probabilistic analysis of simulated Rubin single-visit and coadd images to infer posterior weak lensing shear and convergence fields on the sky.
- Tested on forward simulated data and DC2 images
- Incorporating existing tools: BLISS, MADNESS, JIF, BFD, GalFlow, …
-

Shear & LSS to cosmological parameters

- Develop and apply a Bayesian Hierarchical Model to infer cosmological parameters from tomographic maps of galaxy density and shear.
- Apply to forward simulated data and CosmoDC2 catalogs
- Incorporating existing tools: FlowPM, BORG, KarMMA, …

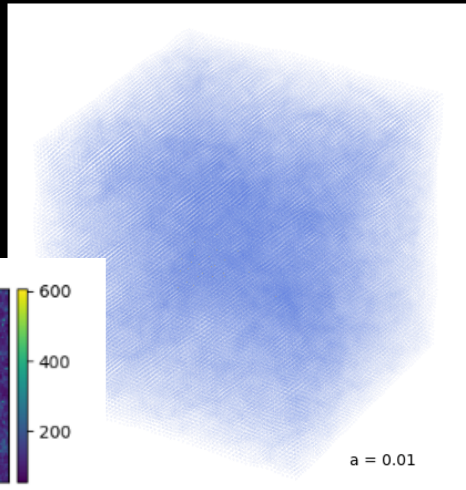
SOME REALISATIONS

ABSOLUTELY NON EXHAUSTIVE APOLOGIES

FlowPM: cosmological N-body code implemented in Mesh-TensorFlow for GPU-accelerated, distributed, and differentiable simulation

Denise Lanzieri et al. arXiv:2010.11847

LSS evolution



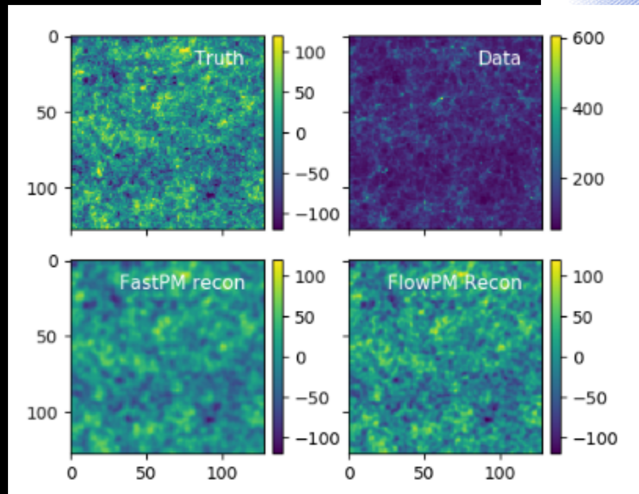
Ray-tracing performed by implementing the Born approximation in Tensorflow

Includes intrinsic alignments (NLA) and tomographic analysis

Current resolution target:

- 1 arcmin resolution, up to about l=3000
- Corresponds roughly to LSST Y1

Fully Automatically Differentiable: you can take derivatives with respect to initial conditions, cosmological parameters, and nuisance parameters



Reconstructing initial conditions from the dark matter field observable

BLISS: *Bayesian Light Source Separator*

Ismael Mendoza, Derek Hansen, Camille Avestruz, Jeffrey Regier, et al. *arXiv:2102.02409*

Galaxy/star **fully probabilistic** detection, deblending, and measurement **algorithm**.

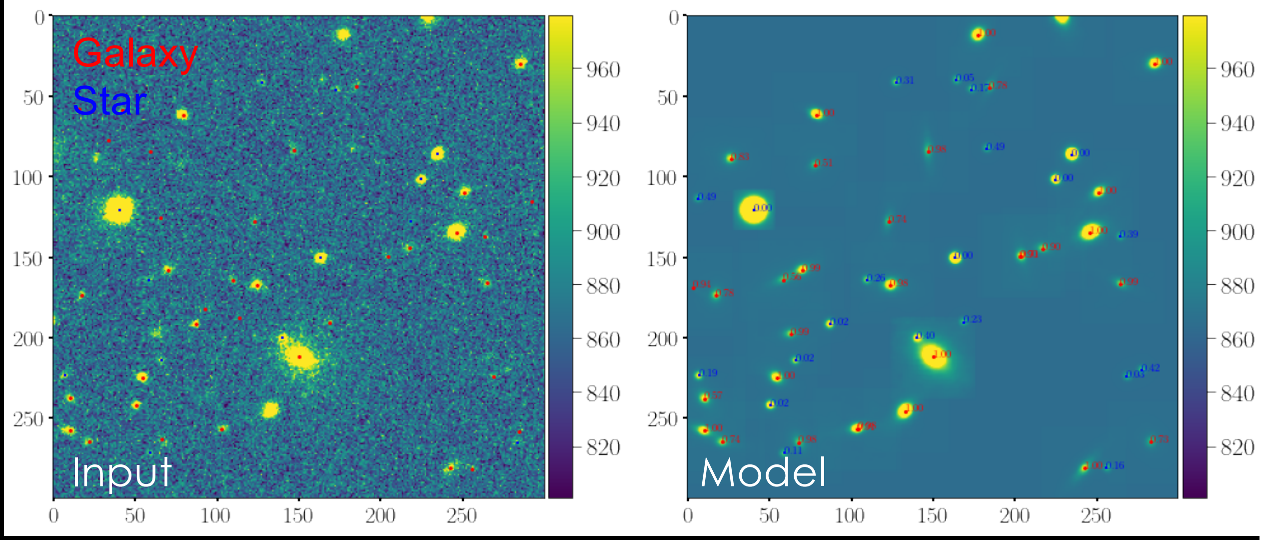
All measured quantities have an associated **probability distribution** from which samples/uncertainties can be obtained.

Output: number of sources, centroids, star vs. galaxy classification, star fluxes, and reconstruction of each individual galaxy.

Fast full posterior inference

Deep learning backbone built in *Pytorch.*

Full inference on 1500x2100 SDSS ~ 20sec (GPU)



Github page, Slack: #desc-bl-bliss

DESC paper describing BLISS and results on deblending Galsim and SDSS galaxies [Project 143].

MADNESS: *Maximum-A-posteriori solution with Deep generative NEtworks for Source Separation*

Biswajit Biswas, Eric Aubourg, Alexandre Boucaud, Axel Guinot, Junpeng Lao, Cécile Roucelle
DESC Project

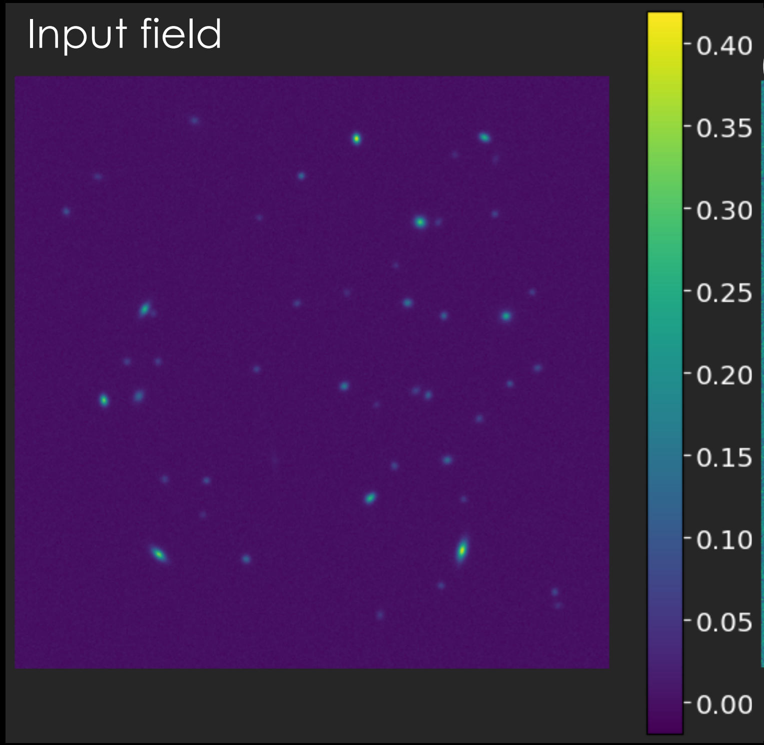
Biswajit Biswas's talk this morning

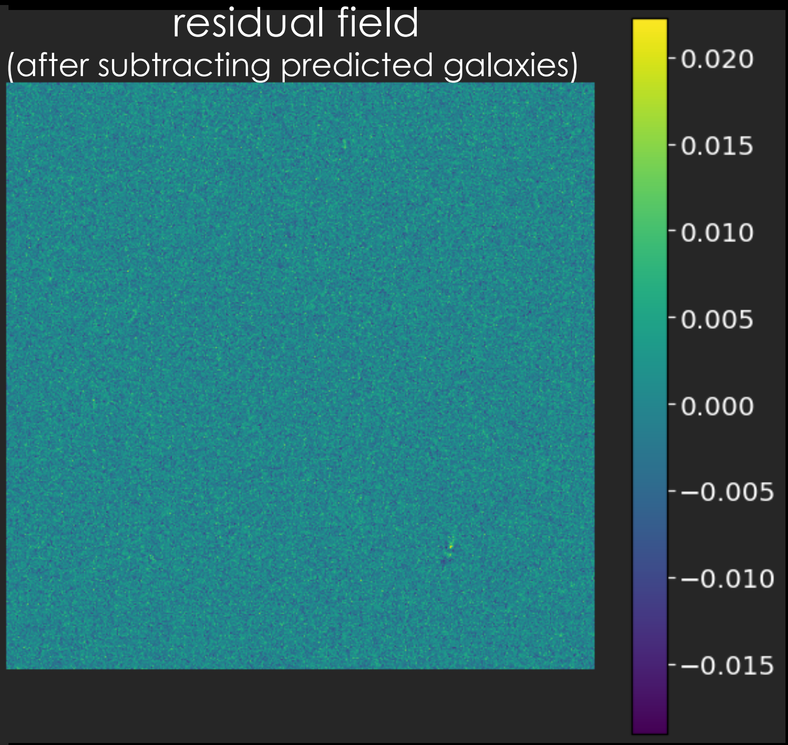
Algorithm based on deep generative models to deblend galaxies from a field.

Input: a field of galaxies and the positions of detected Galaxies.

Output: the maximum a posteriori solution for the reconstructed image of each galaxy.

Based on TensorFlow and TensorFlow-probability.





BORG*-WL: Bayesian Hierarchical Model from shear catalogue to cosmology

Natalia Porqueres et al. arXiv:2108.04825

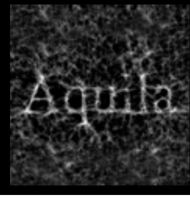
<u>input:</u> galaxy shear estimates, variances; samples of n(z)

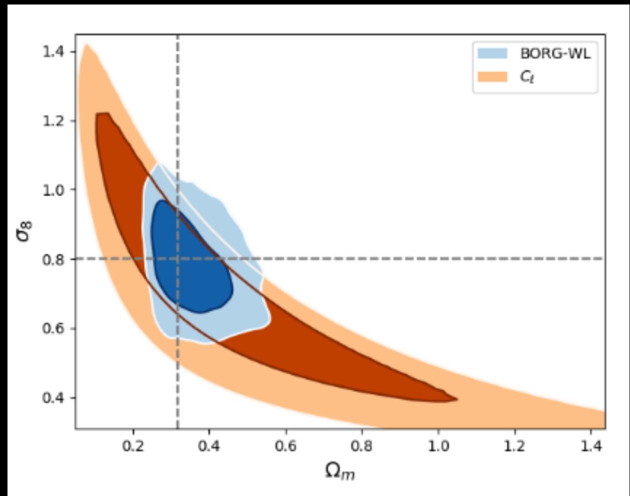
<u>Model:</u> Full forward gravity model with Lagrangian Perturbation Theory/ gravitational particle mesh.

<u>Output:</u> samples of: initial (gaussian) 3D density field; true shear/convergence 'mass' maps; cosmological parameters (~10^7 + 3 or more)

https://www.aquila-consortium.org/projects/

*) bayesian origin reconstruction from galaxie





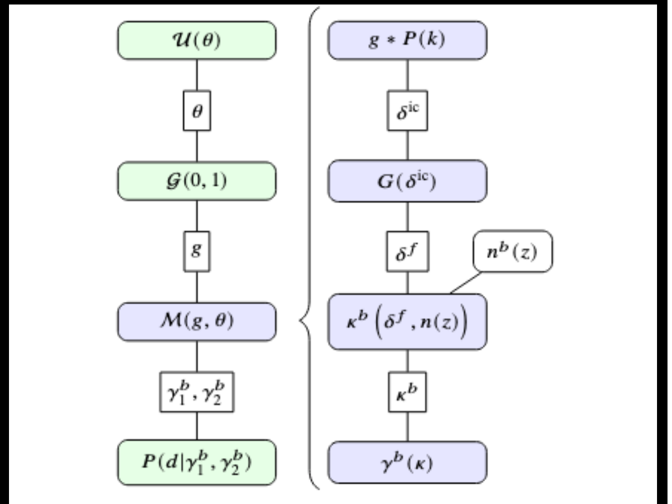


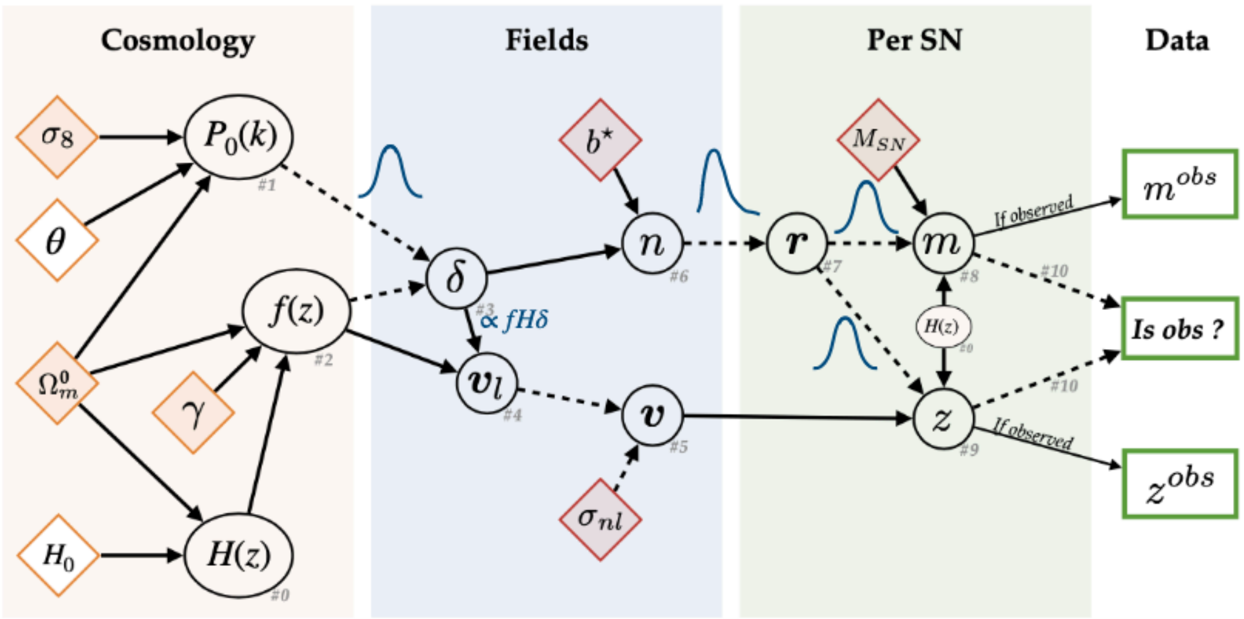
Figure 1. Representation of the Bayesian hierarchical model. Blue boxes indicate deterministic functions and green boxes represent probability distributions. θ are the cosmological parameters, g is a Gaussian field with zero mean and unit variance, $\mathcal{M}(g,\theta)$ is the forward model, which consists of a convolution of the Gaussian field with the initial matter power spectrum $P(k)$ to obtain the initial conditions $\delta^{\rm ic}$, a gravity and structure formation model $G(\delta^{\rm ic})$ and the data model to obtain the shear γ^b for each tomographic bin b. $P(d|\gamma_1^b,\gamma_2^b)$ is the likelihood with d being the data.

constraints on cosmology obtained by applying BORG-WL to the simulated forward model mock data

OTHERS

Tuesday (this meeting)

Goal: Bayesian Forward Modeling for $f\sigma_8$



Each node = conditional prob. of a parameter given other parameters connected to it by an arrow

OTHERS: CLUSTER

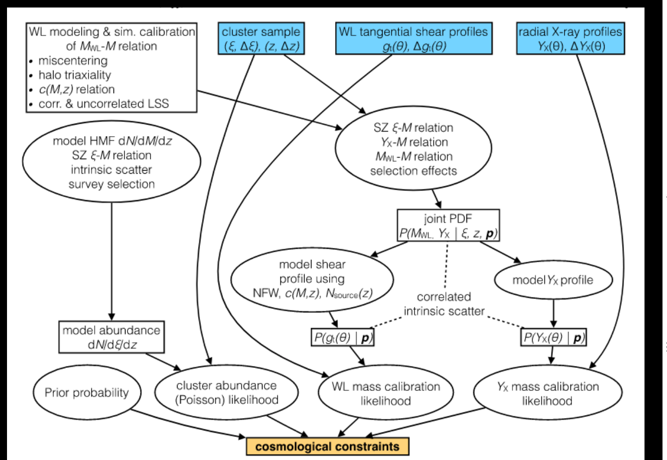
Hierarchical models for cluster mass-observable relation -> cosmology from X-ray selected clusters (Mantz et al. 2010)

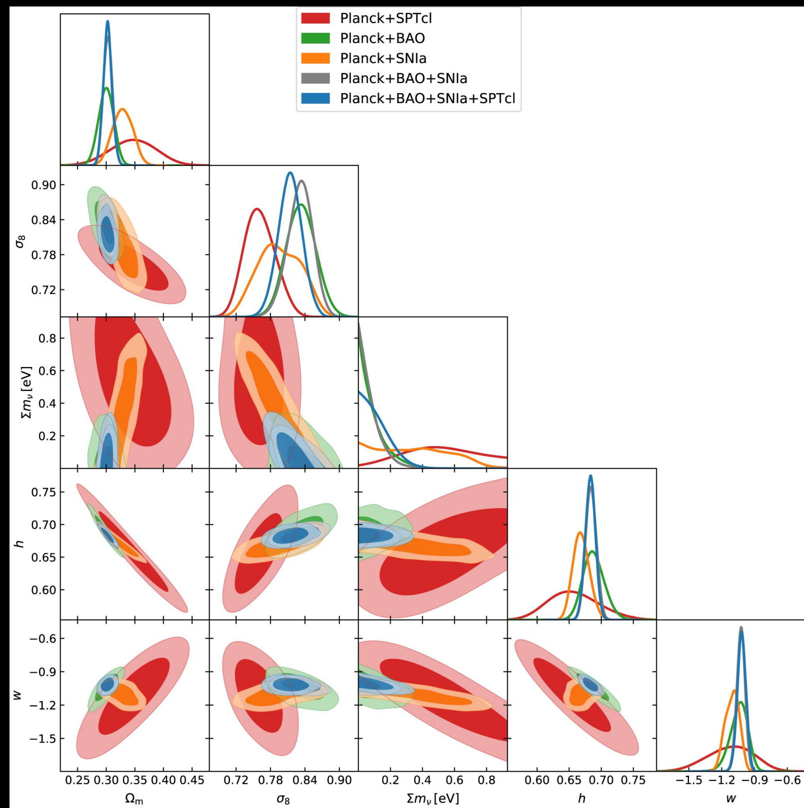
Includes modelling the cluster weak lensing signals (*Weighing the Giants*; von der Linden et al. 2014, Applegate et al. 2014, Mantz et al. 2015)

Also implemented for SZ-selected clusters

…

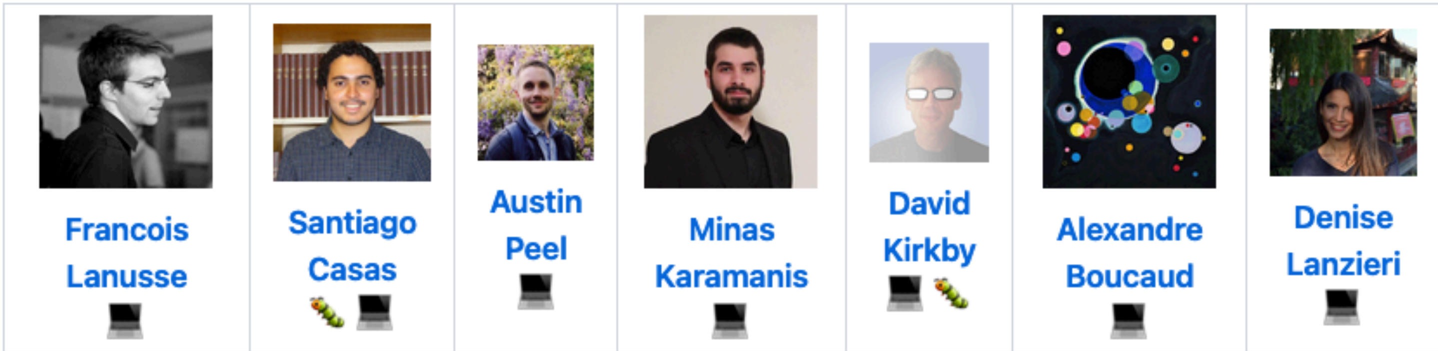
Bocquet et al. 2019

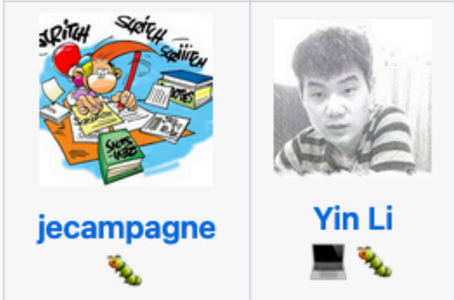




JAX-COSMO LIBRARY

Collaborators





& Joe Zuntz, Tilman Troester, Ben Horowitz

Part of the DifferentiableUniverseInitiative Github

JAX-COSMO LIBRARY

JAX: NumPy + Autograd + XLA

- uses the NumPy API
  => You can copy/paste existing code, works pretty much out of the box
- is a successor of autograd
  => You can *transform* any function to get forward or backward **automatic derivatives** (grad, jacobians, hessians, etc)
- **JIT: Just-In-Time compilation**: uses XLA as a backend
  => Same framework as used by TensorFlow (supports CPU, GPU, TPU execution)
- **Auto-vectorization:**
  => *vmap* transformation will add a batch dimension to the input/output of any function

Several libs are now written to JAX

Flax, Optax, JaxOpt, Numpyro/PyMC3, and TensorFlow Probability (TFP) now also works on JAX.

Many more here

Want to know more: https://github.com/jecampagne/JaxTutos

JAX-COSMO LIBRARY

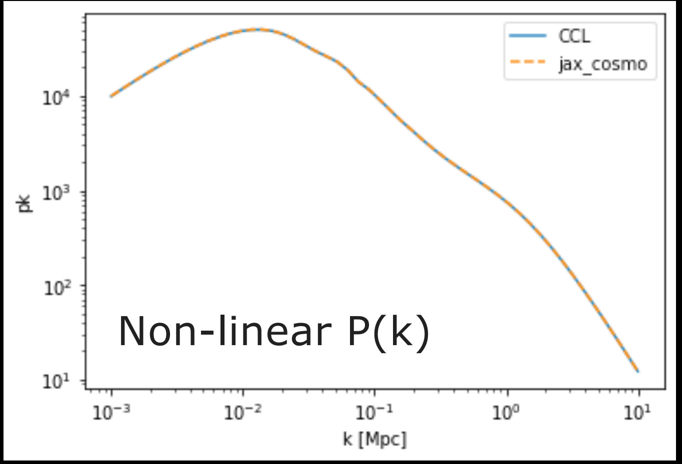
<u>Goal</u>: Cosmological library *fully differentiable* written in JAX (Autodiff, GPU acceleration, automated batching)

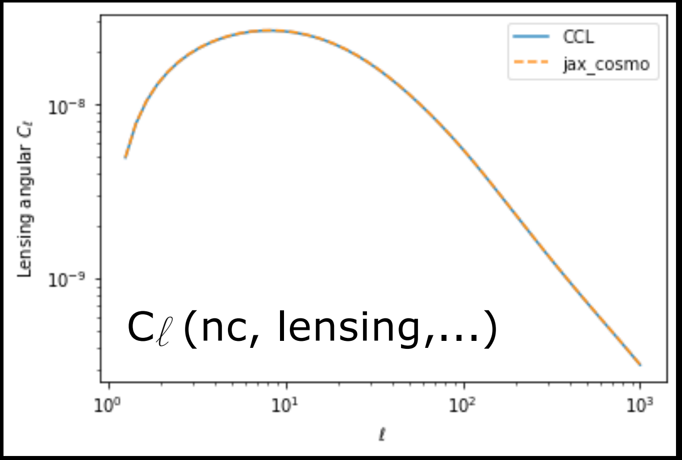
<u>Currently</u>:

- Follow CCL's tracer mechanism
- Unit tested against CCL *(Limber-only)*
- P(k) (Eisenstein-Hu, Halofit), Cls (Lensing including NLA, multiplicative bias, number counts)

JAX CLASS $P^{nl}_{,\delta}(k,z)$ emulator (*JEC- in preparation*)

MIT licensed, collaboratively developed





```python
# Create a cosmology with default parameters
cosmo = jc.Planck15()
```

Cosmo: JAX-PyTree structure for cosmology params.

nzs: 2 redshifts distrib.

```python
probes = [ jc.probes.WeakLensing(nzs, sigma_e=0.26),
           jc.probes.NumberCounts(nzs, jc.bias.constant_linear_bias(1.)) ]
```

```python
ell = np.logspace(1,3)
```

```python
mu, cov = jc.angular_cl.gaussian_cl_covariance_and_mean(cosmo, ell, probes,␣
↪sparse=True);
```

Probes, C_ℓ + cov. mtx.

```python
# We define a parameter dependent function that computes the mean
def mean_fn(p):
  cosmo = jc.Planck15(Omega_c=p[0], sigma8=p[1])
  # Compute signal vector
  m = jc.angular_cl.angular_cl(cosmo, ell, probes)
  return m.flatten() # We want it in 1d to operate against the covariance matrix
```

A function of cosmo parameters

```python
# We compute it's jacobian with JAX, and we JIT it for efficiency
jac_mean = jax.jit(jax.jacfwd(mean_fn))
```

```python
# Let's define a parameter vector for Omega_cdm, sigma8, which we initialize
# at the fiducial cosmology used to produce the data vector.
params = np.array([cosmo.Omega_c, cosmo.sigma8])
```

```python
# We can now evaluate the jacobian at the fiducial cosmology
dmu = jac_mean(params)
```

Its gradient +JIT

$$\frac{\partial \mu_i}{\partial \theta_\alpha}$$

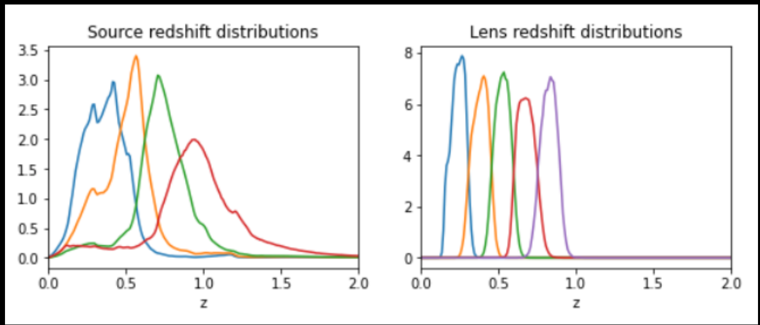
```python
# Now we can compose the Fisher matrix:
F_2 = jc.sparse.dot(dmu.T, jc.sparse.inv(cov), dmu)
```

Fisher…

$$\frac{\partial \mu_i}{\partial \theta_\alpha} C^{-1}_{ij} \frac{\partial \mu_j}{\partial \theta_\beta} = F_{\alpha\beta}$$

EX: DES Y1 3X2PTS WITH JAX-COSMO & NUMPYRO*

The n(z) from DES Public area

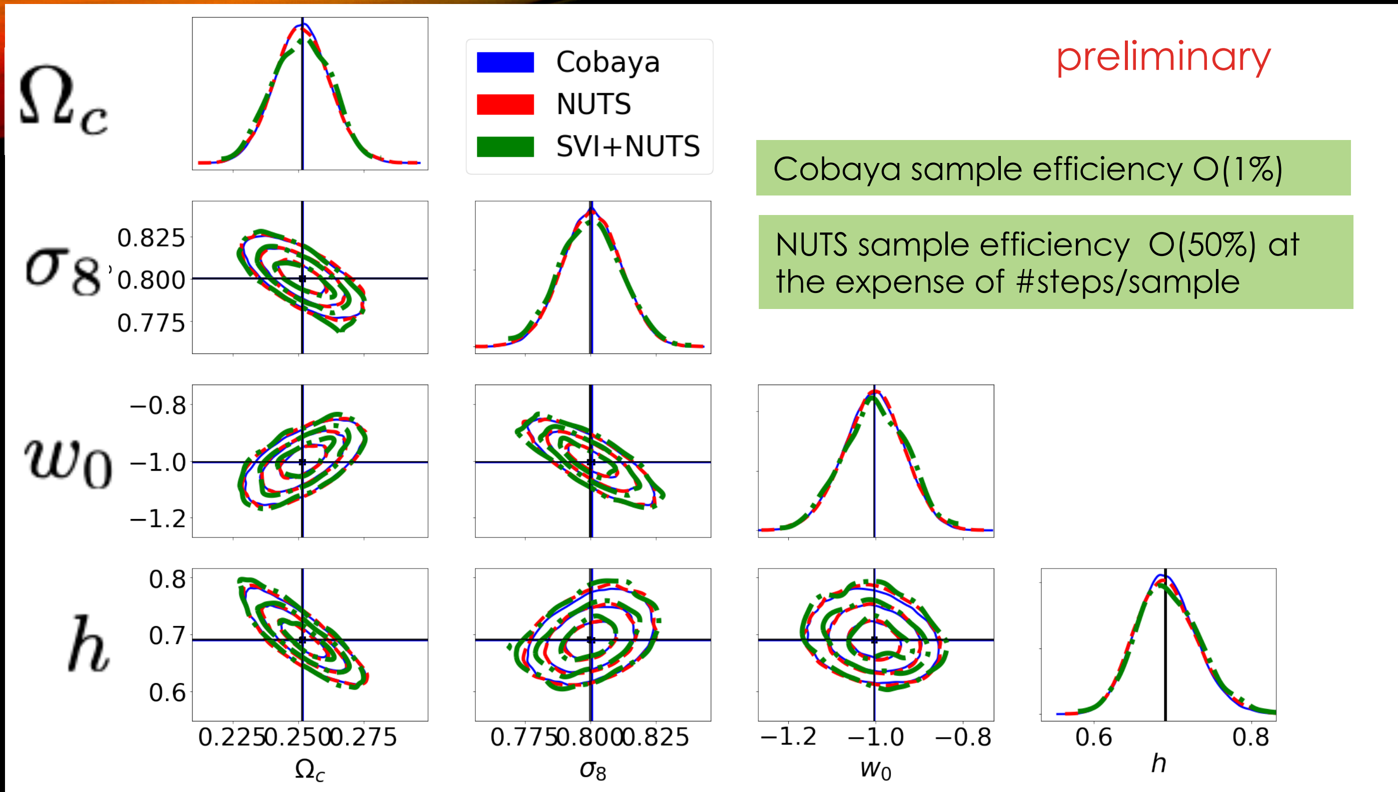


Forward model to generate 'mock' Cls + Cov. Mtx
(50 ells x 45 auto + cross)
21-param.: Cosmo. params (6) + Intrinsic Alignment (2) + (shear/photo-z) syst. param per tomog. bin (2x4) + linear gal. bias (5)

Numpyro model built from Parameter Priors + jax-cosmo Cls & fixed Cov. Mtx *(Following Abbot et al (DES) arXiv:1708.01530)*

Conditioned by mock data => Inference on the parameters

*) Numpyro: JAX implementation of Pyro probabilistic programming language (Pytorch)



Cobaya (J. Torrado, A. Lewis arXiv:2005.05290) 140,000 samples (200hrs CPU)
NUTS (No-U-Turns Hamiltonian Monte Carlo): 16,000 samples: 16 chains x 1,000 samples (20hrs @ CCIn2p3 GPU V100)
SVI+NUTS: 1,000 samples, Stochastic Variational Inference with MVN kernel (20,000 steps/2hrs; GPU V100) followed by NUTS 1 chain of 1000 samples (1hrs)

WANT TO GO FURTHER?

- On Bayesian Pipeline Topical Team: #desc-bayesian-pipelines-tt
- On jax/jax-cosmo: contact the team and ask for tutos
- On Bayesian methods related to astro-cosmology:

https://indico.in2p3.fr/e/bdlCosmo2022

Workshop sponsored by LSSTC which **provides grants** for young scientists to attend.

Deadline coming soon !

Please register ASAP