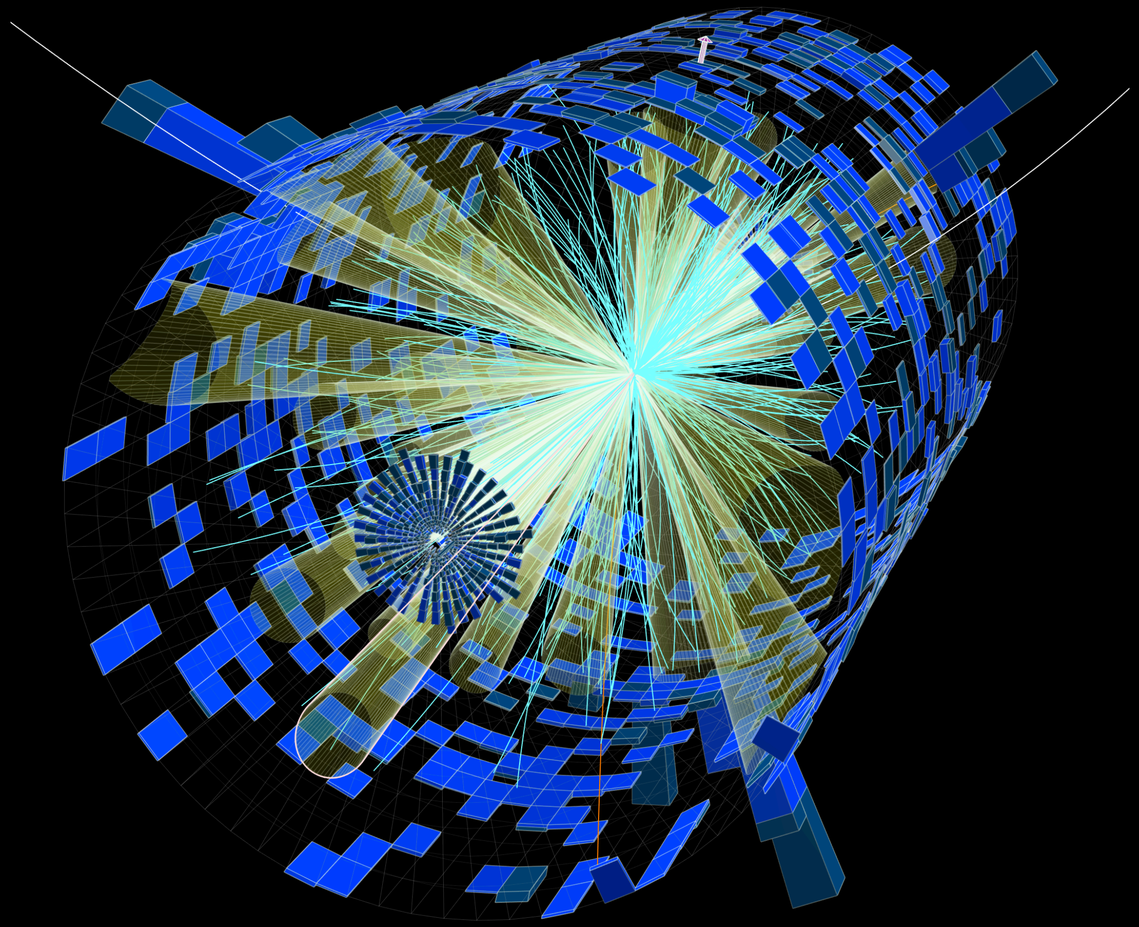
NYU Center for Data Science

Center for Cosmology and particle physics

MACHINE LEARNING

LECTURE 4

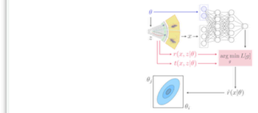


@KyleCranmer
New York University
Department of Physics
Center for Data Science
CILVR Lab

Hands-on Tutorial Yesterday

https://cranmer.github.io/madminer-tutorial/

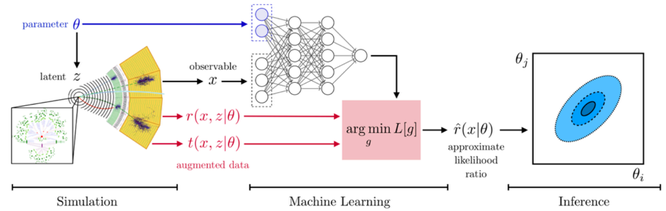
- We accomplished a lot!
- From scratch:
  - Generate simulated data for EFT with MadGraph
  - Smear (could have used Delphes)
  - Trained neural network to learn likelihood ratio
  - Trained neural network to learn Score (Optimal Observable)
  - Calculated expected limit for both approaches and compared to simple 1-d histogram approach
  - Calculated Fisher information matrix
- This is workflow for several published papers
  - To speed this up, working to streamline MadMiner with REANA



Introduction

MadMiner tutorial

This is a tutorial on MadMiner developed by Johann Brehmer, Felix Kling, Irina Espejo, and Kyle Cranmer. It is built using Jupyter Book.



Introduction to MadMiner

Particle physics processes are usually modelled with complex Monte-Carlo simulations of the hard process, parton shower, and detector interactions. These simulators typically do not admit a tractable likelihood function: given a (potentially high-dimensional) set of observables, it is usually not possible to calculate the probability of these observables for some model parameters. Particle physicisists usually tackle this problem of "likelihood-free inference" by hand-picking a few "good" observables or summary statistics and filling histograms of them. But this conventional approach discards the information in all other observables and often does not scale well to high-dimensional problems.

In the three publications "Constraining Effective Field Theories With Machine Learning", "A Guide to Constraining Effective Field Theories With Machine Learning", and "Mining gold from implicit models to improve likelihood-free inference", a new approach has been developed. In a nut shell, additional information is extracted from the simulations that is closely related to the matrix elements that determine the hard process. This "augmented data" can be used to train neural networks to efficiently approximate arbitrary likelihood ratios. We playfully call this process "mining gold" from the simulator, since this information may be hard to get, but turns out to be very valuable for inference.

http://reanahub.io

reana

Reproducible research data analysis platform

Flexible

Run many computational workflow engines.

COMMON WORKFLOW LANGUAGE

Scalable

Support for remote compute clouds.

kubernetes

Reusable

Containerise once, reuse elsewhere. Cloud-native.

Free

Free Software. MIT licence.
Made with ❤ at CERN.

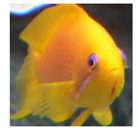
CERN

NSF

diana hep

iris hep
Institute for Research and Innovation in Software for High Energy Physics

The SCAILFIN Project
scailfin.github.io

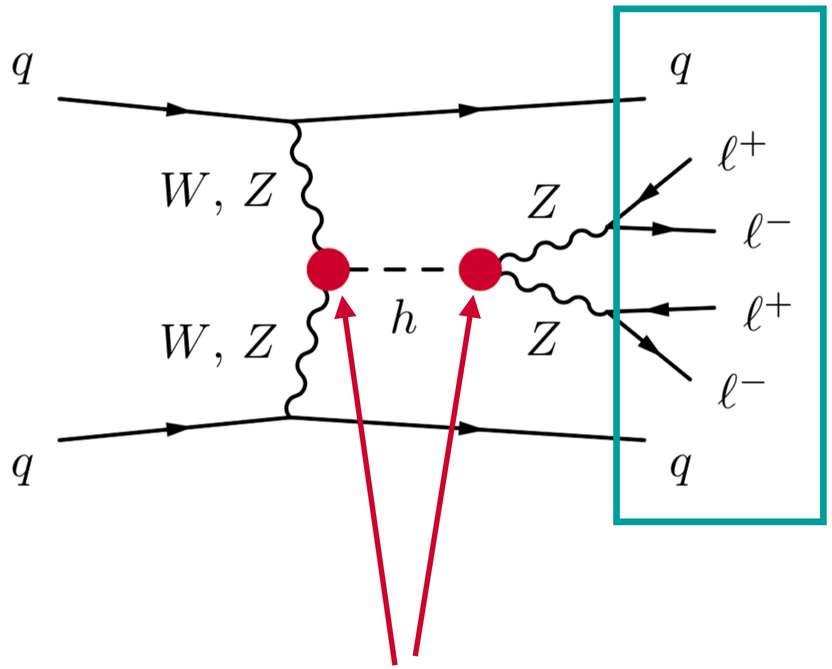


These techniques let us constrain effective theories more effectively.

Proof of concept: Higgs production in weak boson fusion

[JB, K. Cranmer, G. Louppe, J. Pavez
1805.00013, 1805.00020]

at least 16-dimensional
observable space



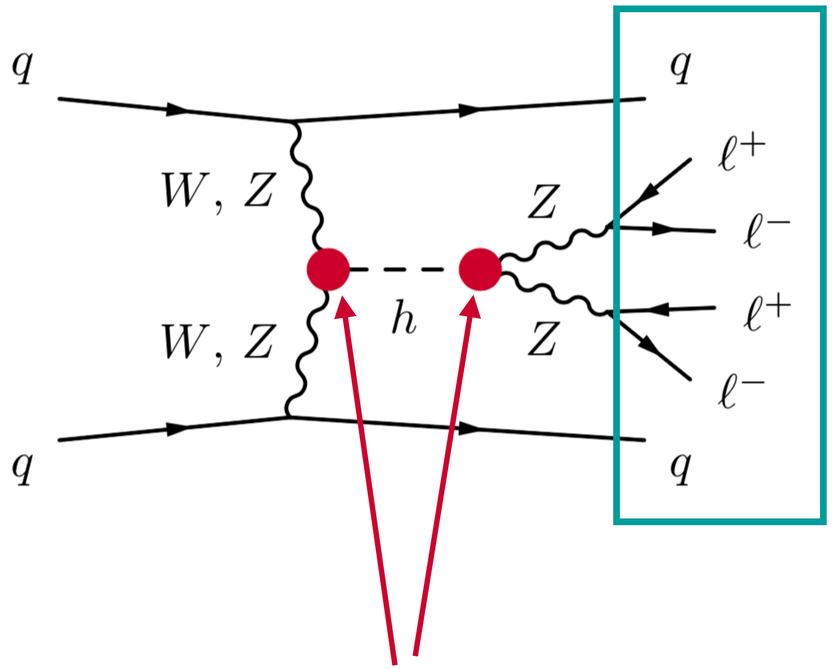
Exciting new physics might hide here!
We parameterize it with two EFT coefficients:

$$\mathcal{L} = \mathcal{L}_{\rm SM} + \boxed{\frac{f_W}{\Lambda^2}} \underbrace{\frac{ig}{2}\,(D^\mu\phi)^\dagger\,\sigma^a\,D^\nu\phi\,W^a_{\mu\nu}}_{\mathcal{O}_W} - \boxed{\frac{f_{WW}}{\Lambda^2}} \underbrace{\frac{g^2}{4}\,(\phi^\dagger\phi)\,W^a_{\mu\nu}\,W^{\mu\nu\,a}}_{\mathcal{O}_{WW}}$$

Proof of concept: Higgs production in weak boson fusion

[JB, K. Cranmer, G. Louppe, J. Pavez
1805.00013, 1805.00020]

at least 16-dimensional observable space



Exciting new physics might hide here!
We parameterize it with two EFT coefficients:

$$\mathcal{L} = \mathcal{L}_{\rm SM} + \boxed{\frac{f_W}{\Lambda^2}} \underbrace{\frac{ig}{2}\,(D^\mu\phi)^\dagger\,\sigma^a\,D^\nu\phi\,W^a_{\mu\nu}}_{\mathcal{O}_W} - \boxed{\frac{f_{WW}}{\Lambda^2}} \underbrace{\frac{g^2}{4}\,(\phi^\dagger\phi)\,W^a_{\mu\nu}\,W^{\mu\nu\,a}}_{\mathcal{O}_{WW}}$$

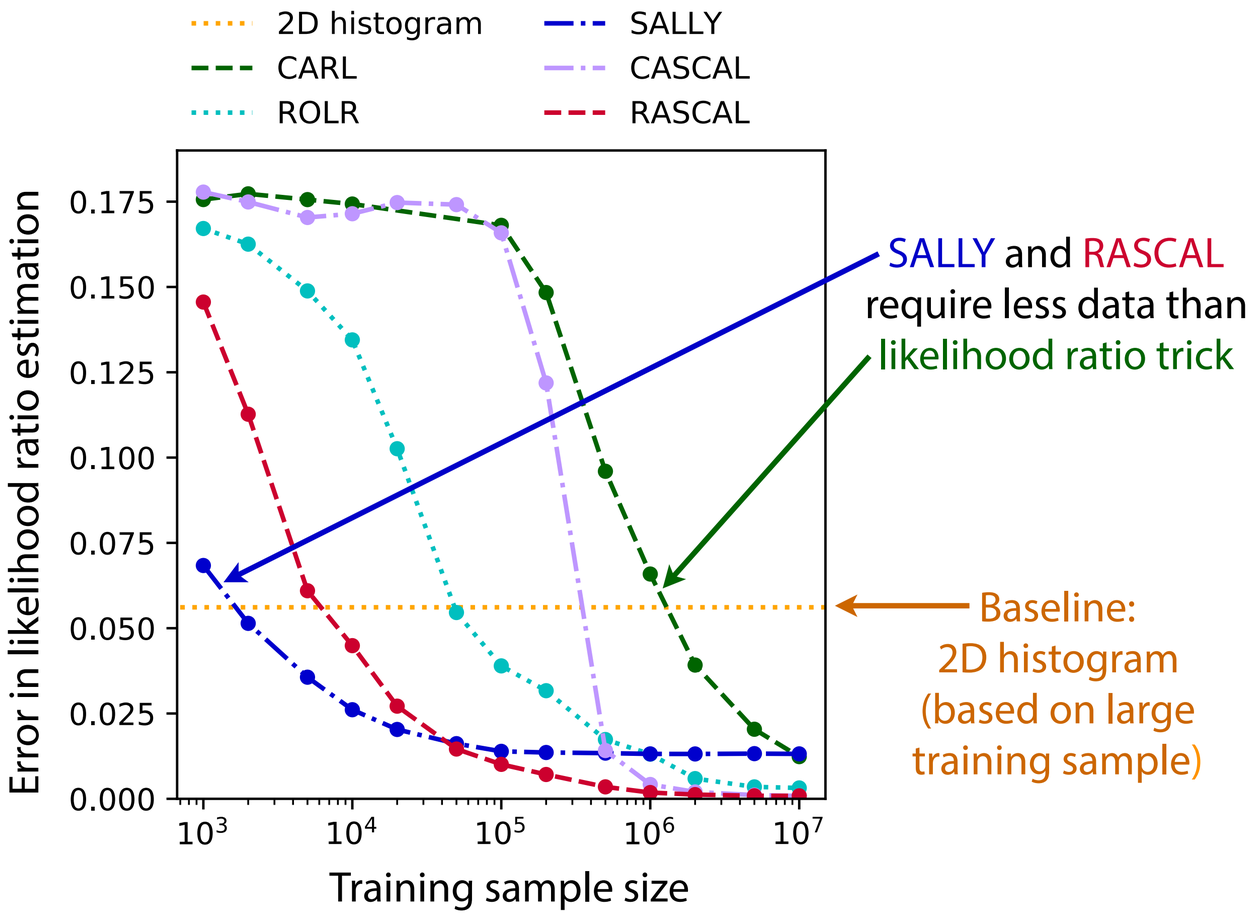
Goal: constrain the two EFT parameters

- new inference methods
- baseline: 2d histogram analysis of jet momenta & angular correlations

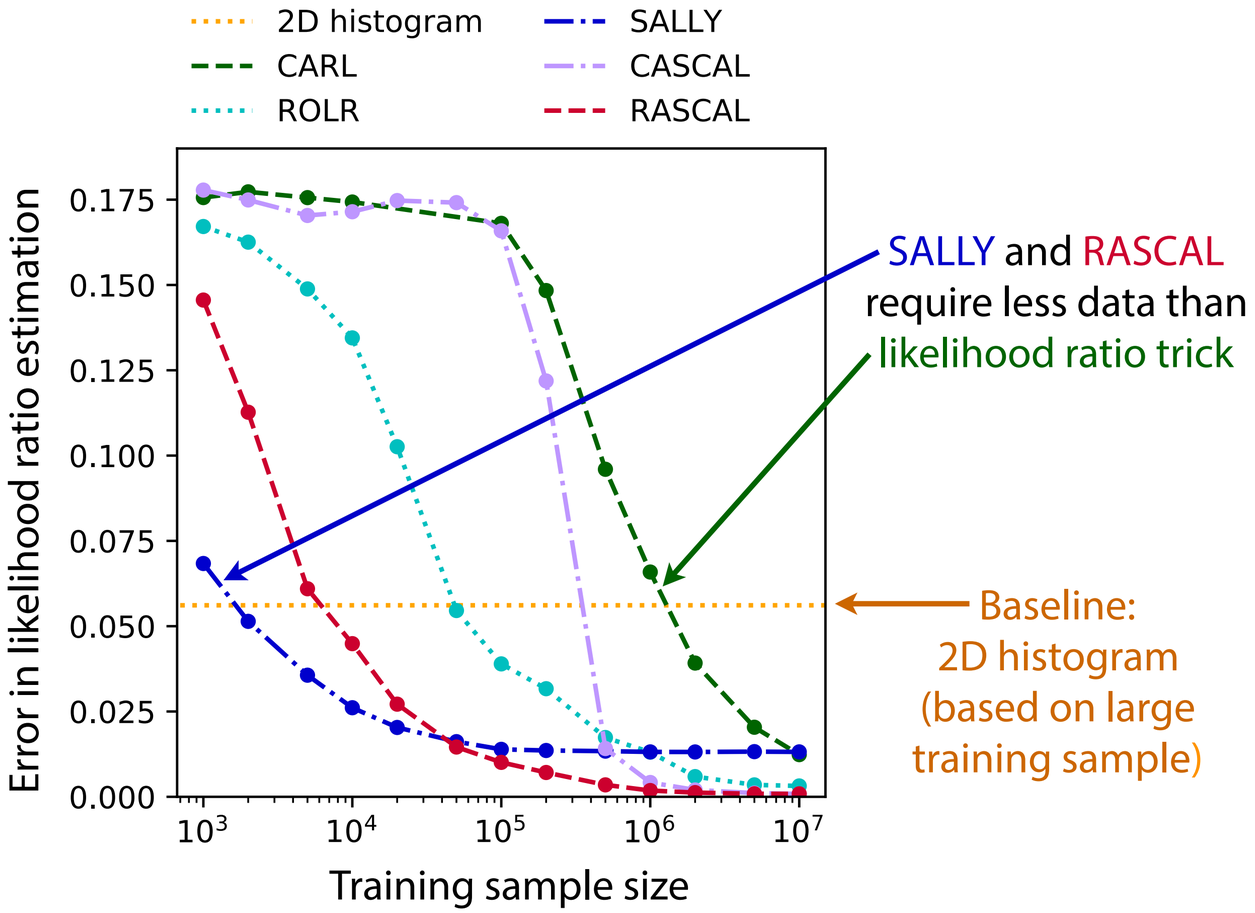
Two scenarios:

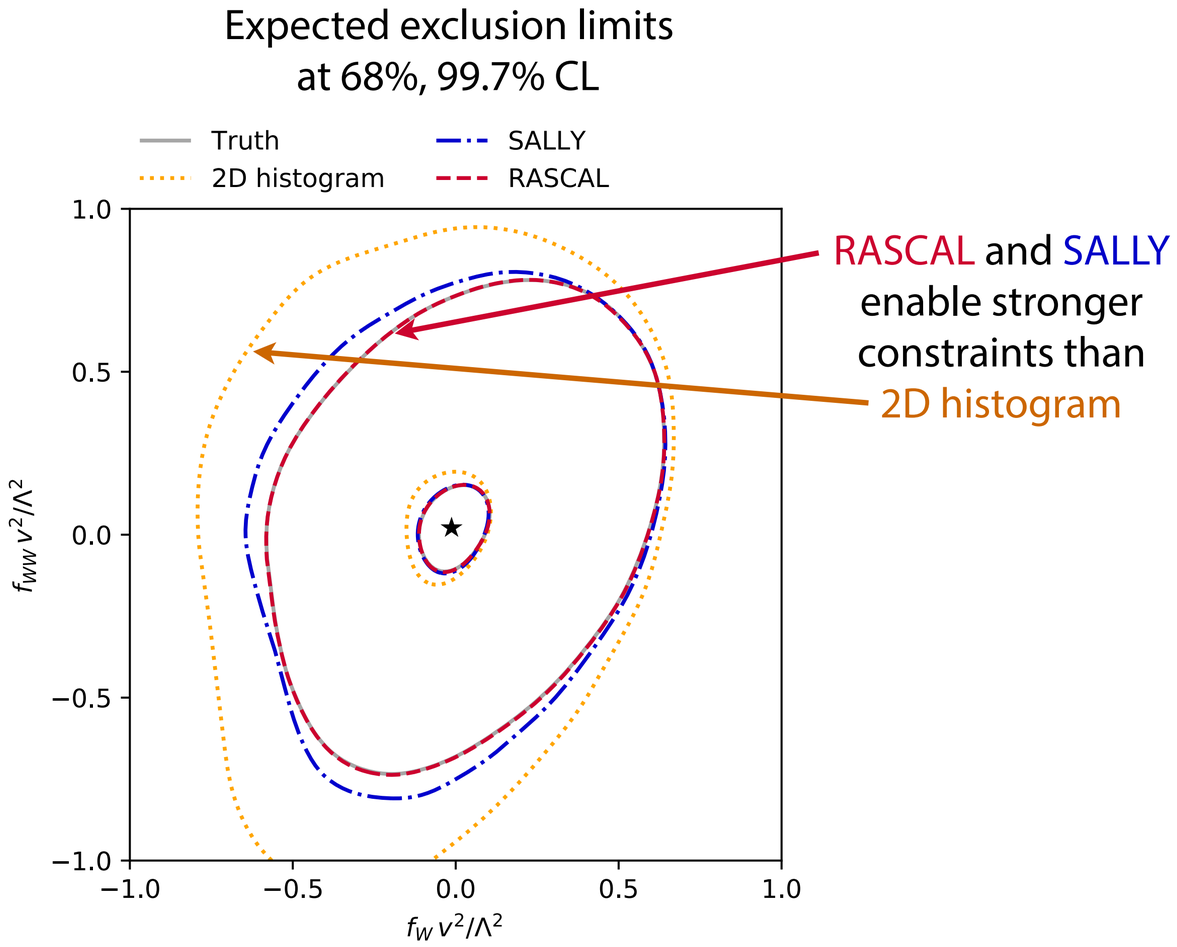
- Simplified setup in which we can compare to true likelihood
- "Realistic" simulation with approximate detector effects

Proof of concept: Stronger constraints with less training data

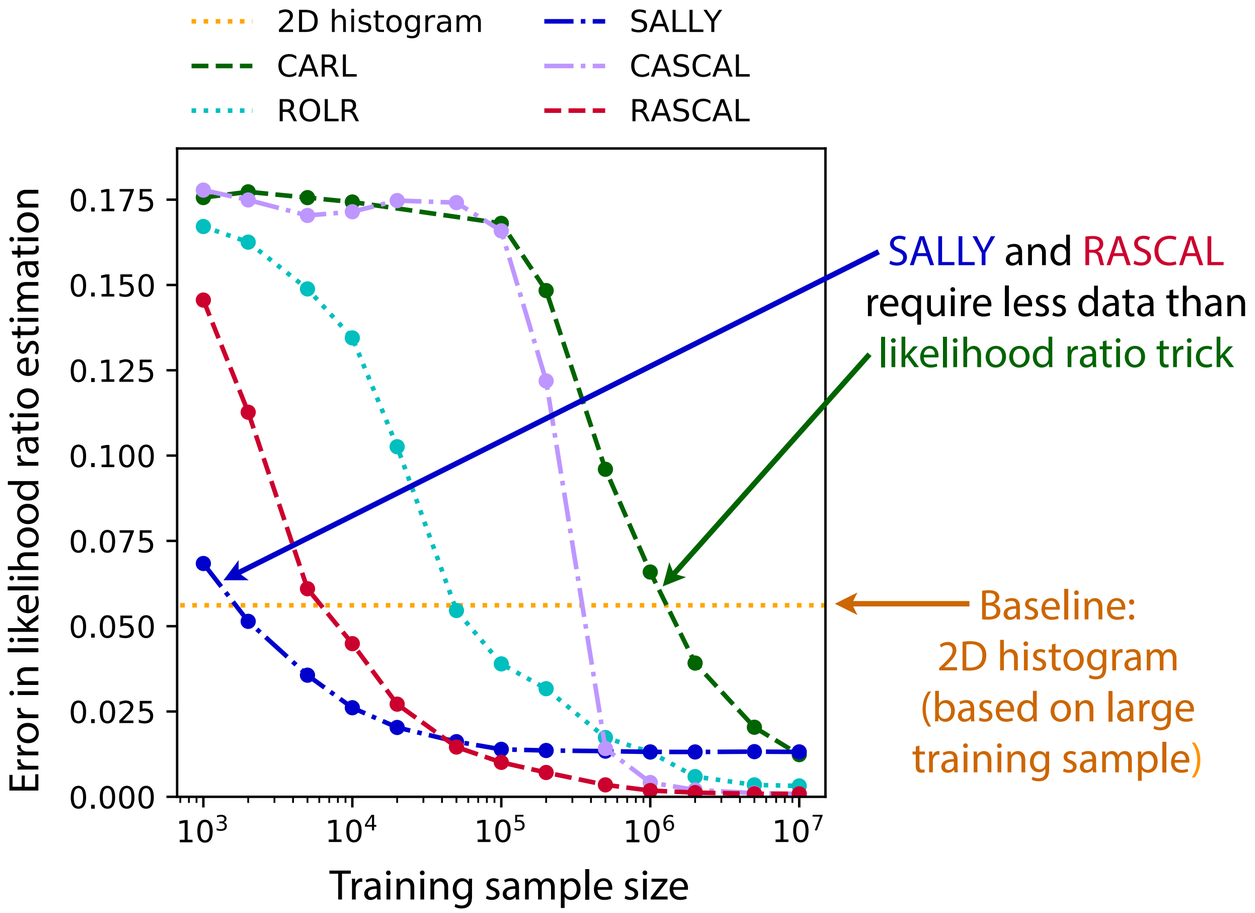


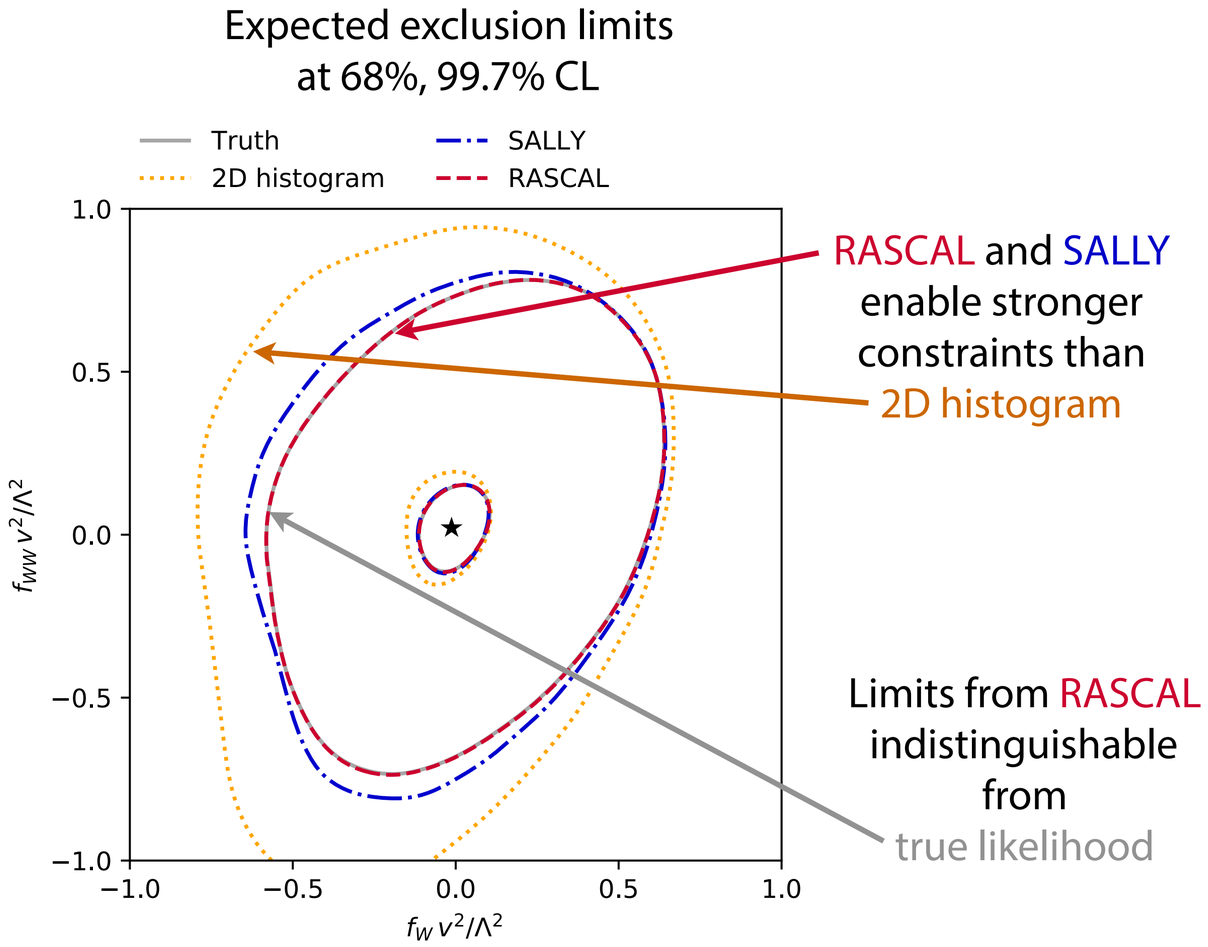
Proof of concept: Stronger constraints with less training data



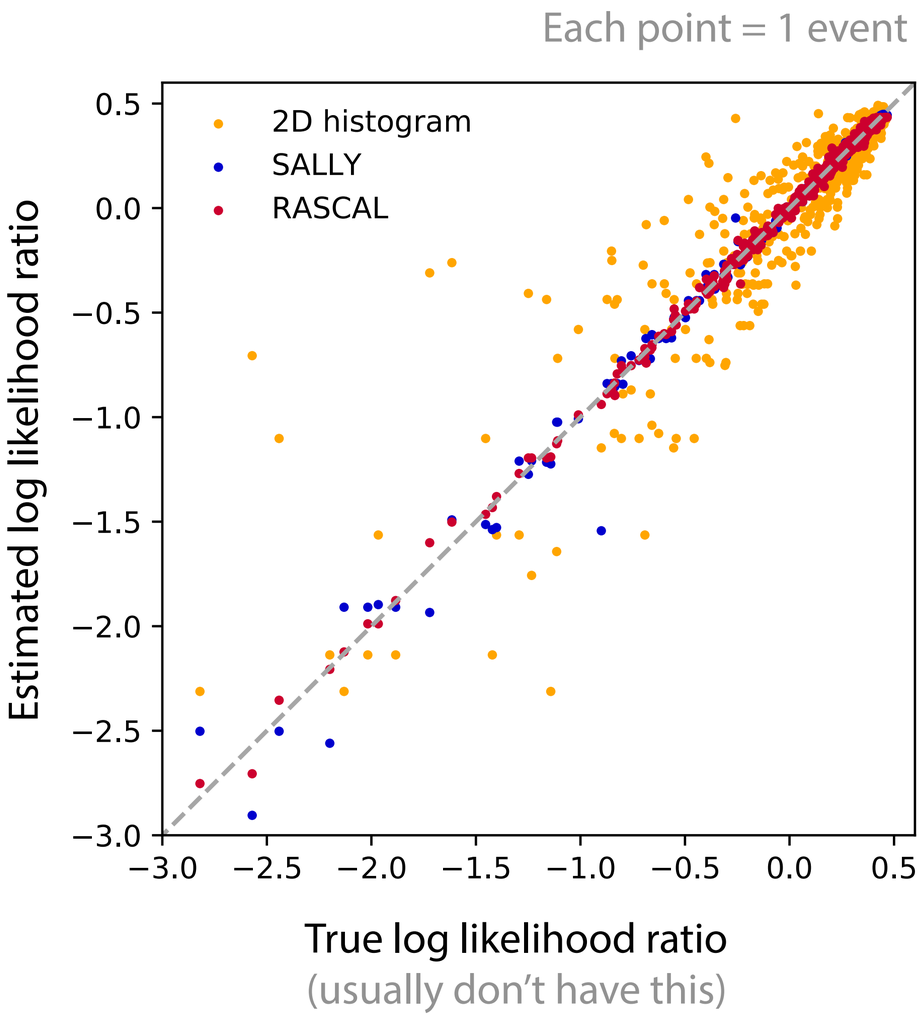


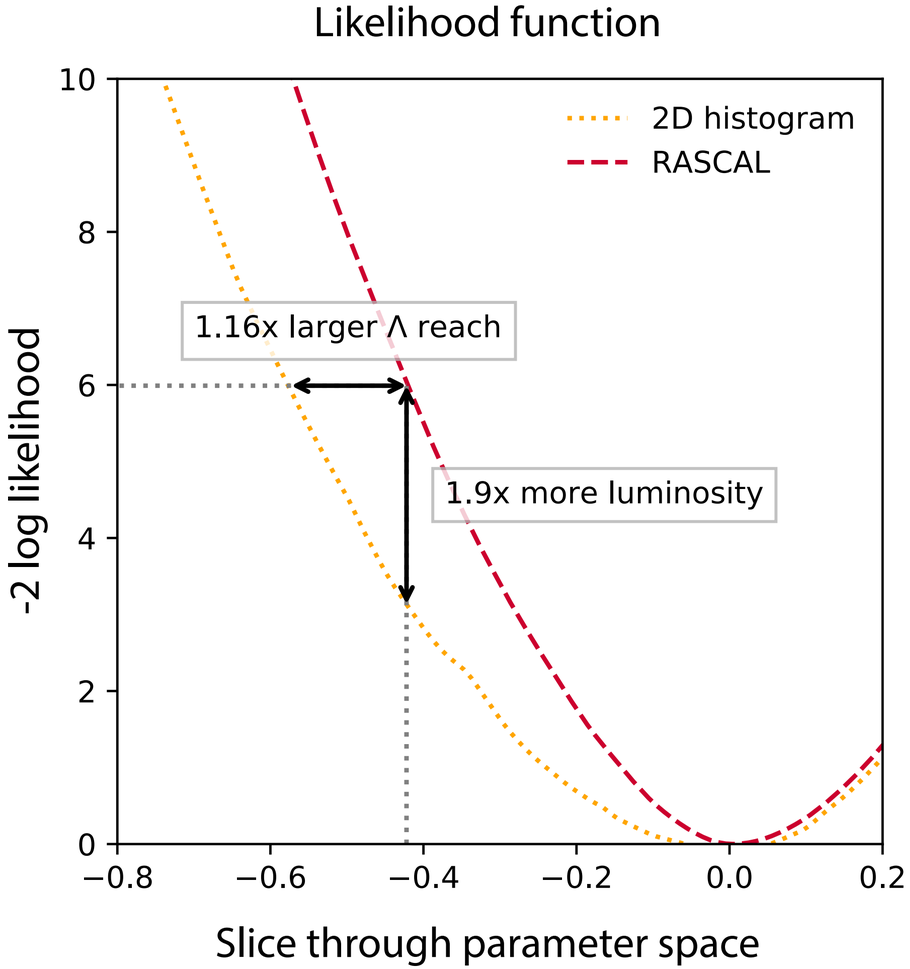
Proof of concept: Stronger constraints with less training data





Proof of concept: 16% greater reach (~90% more data)





Constraining operators in ttH effectively

[JB, F. Kling, I. Espejo, K. Cranmer 1907.10621]

- Pheno-level analysis of

  $$pp \to t\bar{t}\,h \to (b\ell^+)\,(\bar{b}\ell^-)\,(\gamma\gamma)\,E_T^{\text{miss}}$$

  with MadGraph + Pythia + Delphes

- Inference on three EFT operators:

$$\mathcal{O}_u = -\frac{1}{v^2}(H^\dagger H)(H^\dagger \bar{Q}_L)u_R\,,\ \mathcal{O}_G = \frac{g_s^2}{m_W^2}(H^\dagger H)G^a_{\mu\nu}G_a^{\mu\nu}\,,$$

$$\mathcal{O}_{uG} = -\frac{4g_s}{m_W^2}y_u(H^\dagger \bar{Q}_L)\gamma^{\mu\nu}T_a u_R G^a_{\mu\nu}$$

Constraining operators in ttH effectively

[JB, F. Kling, I. Espejo, K. Cranmer 1907.10621]

- Pheno-level analysis of

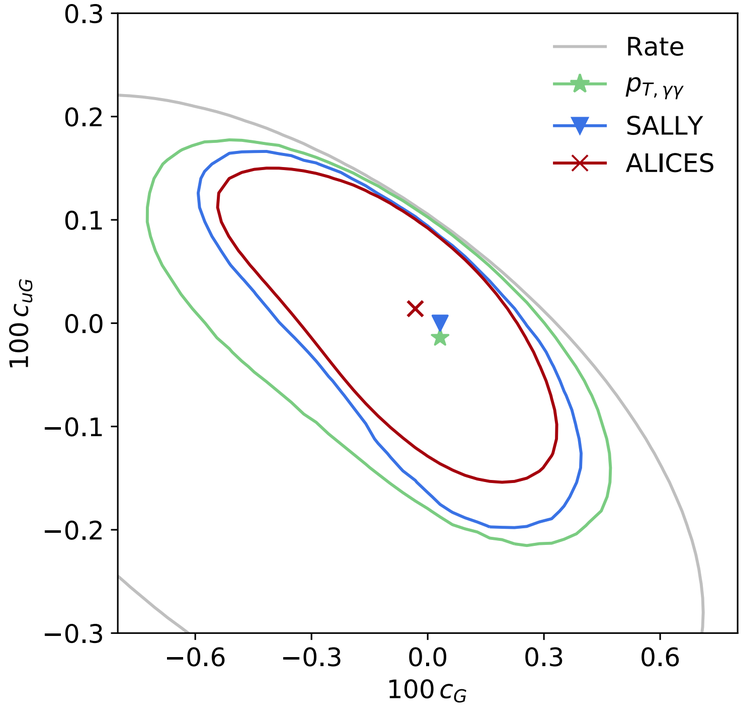
$$pp \to t\bar{t}\,h \to (b\ell^+)\,(\bar{b}\ell^-)\,(\gamma\gamma)\,E_T^{\text{miss}}$$

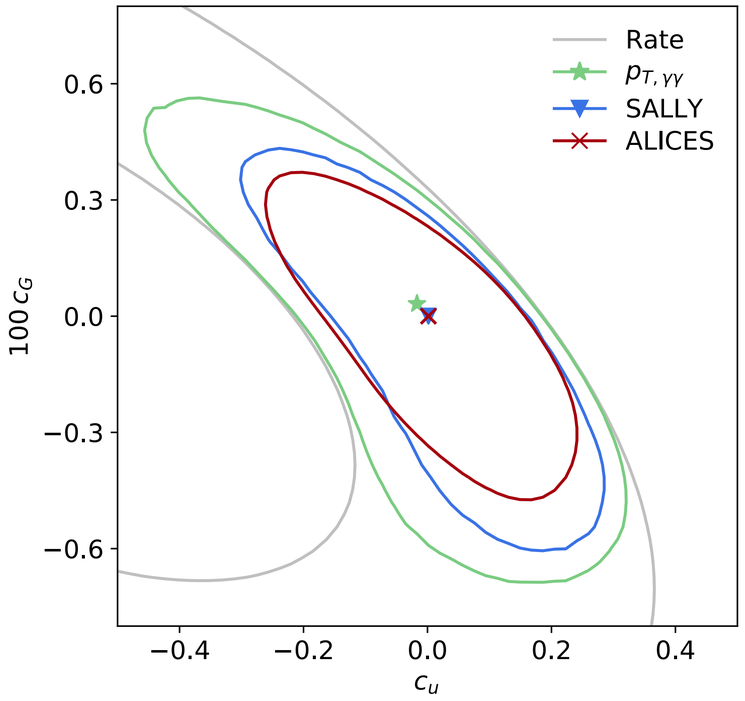
with MadGraph + Pythia + Delphes

- Inference on three EFT operators:

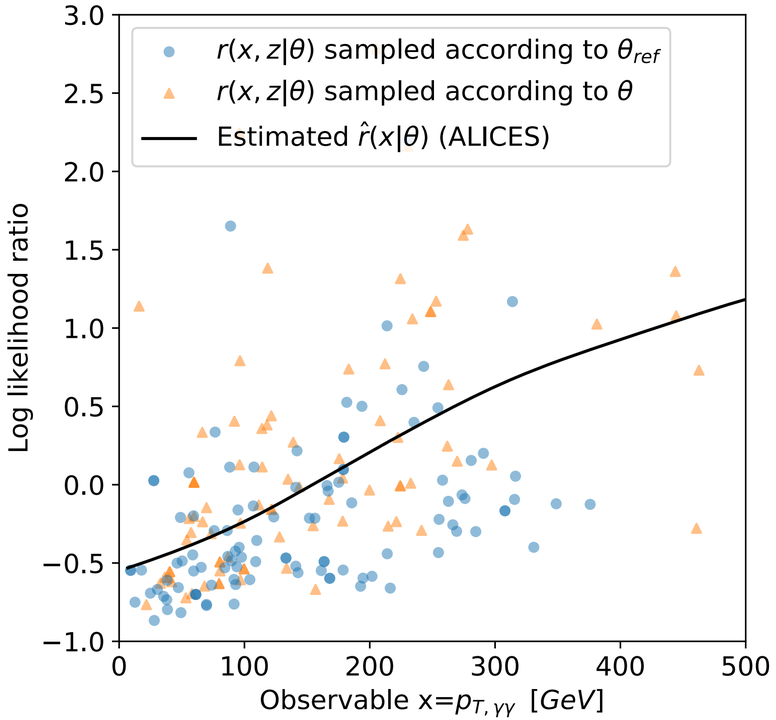
$$\mathcal{O}_u = -\frac{1}{v^2}(H^\dagger H)(H^\dagger \bar{Q}_L)u_R\,,\ \mathcal{O}_G = \frac{g_s^2}{m_W^2}(H^\dagger H)G^a_{\mu\nu}G_a^{\mu\nu}\,,$$
$$\mathcal{O}_{uG} = -\frac{4g_s}{m_W^2}y_u(H^\dagger \bar{Q}_L)\gamma^{\mu\nu}T_a u_R G^a_{\mu\nu}$$

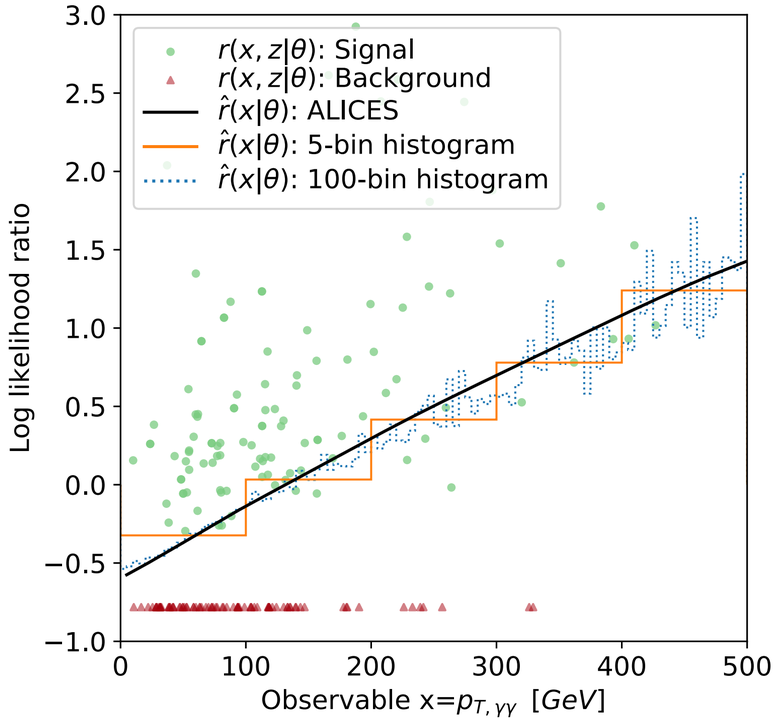
- New inference techniques improve expected HL-LHC limits compared to histogram baseline:

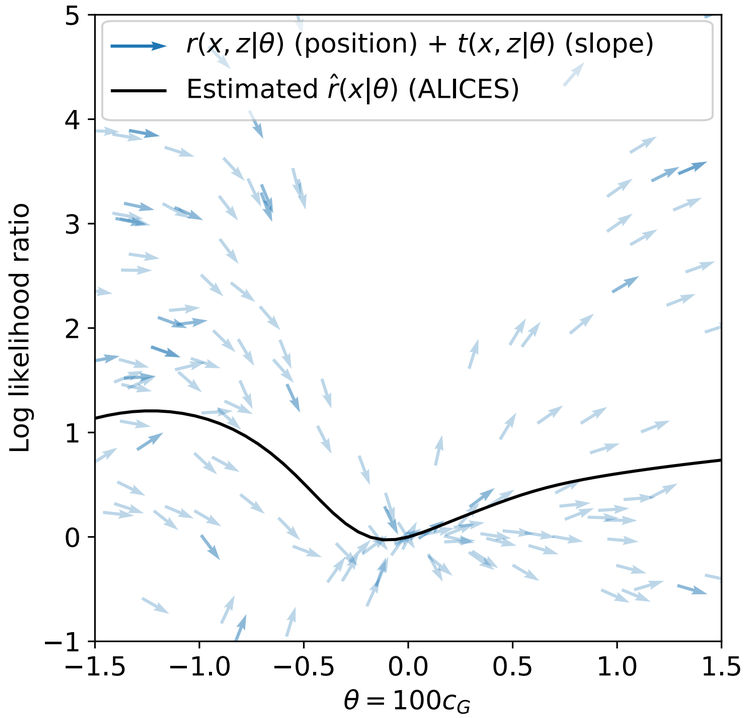




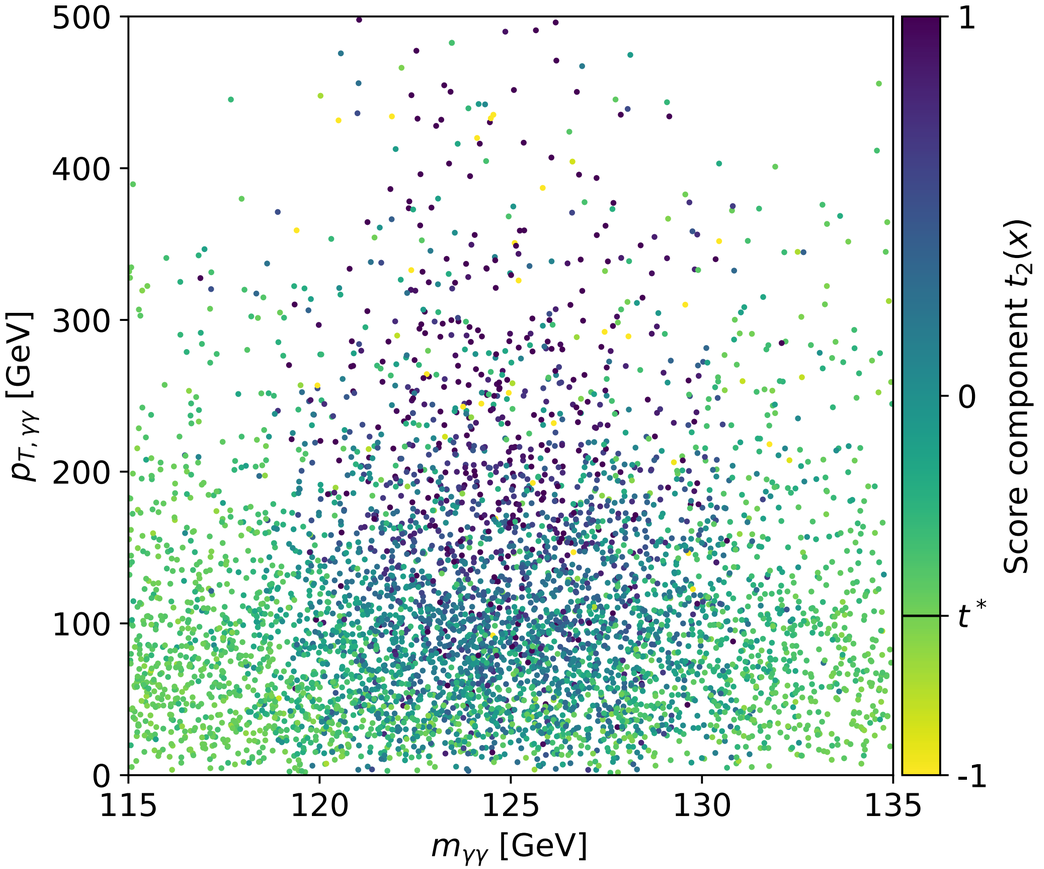
ttH: 1D illustration

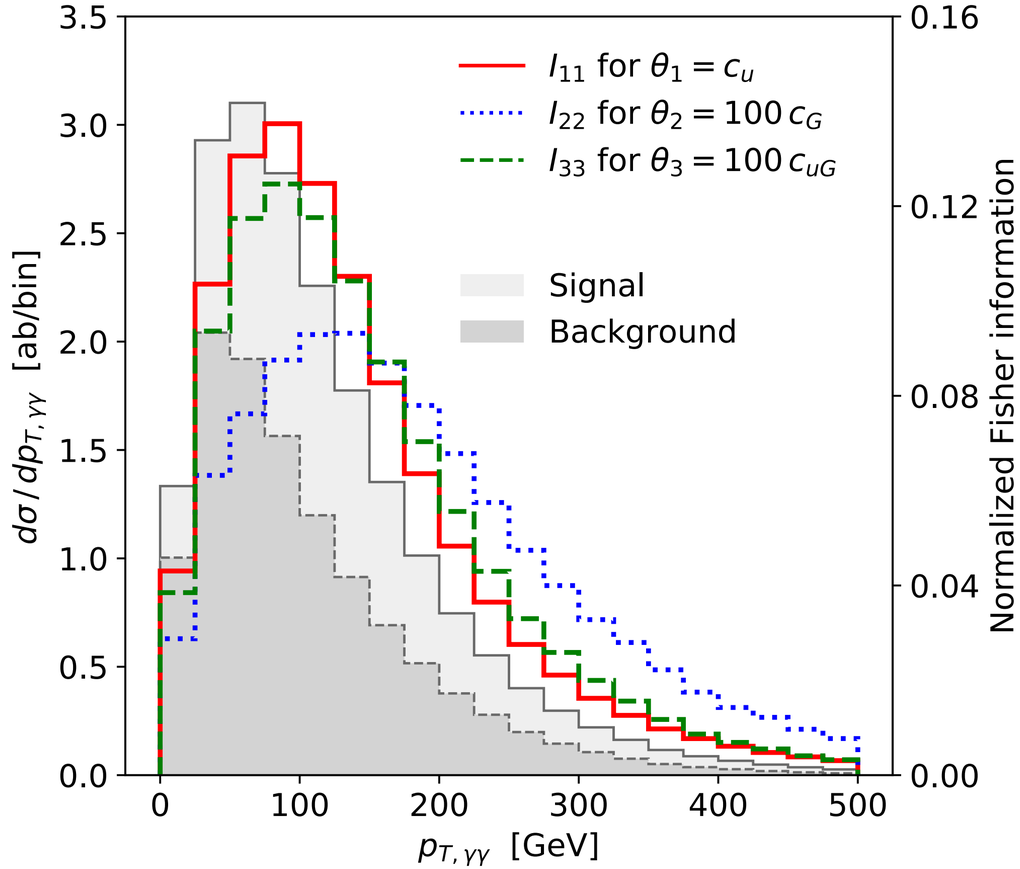




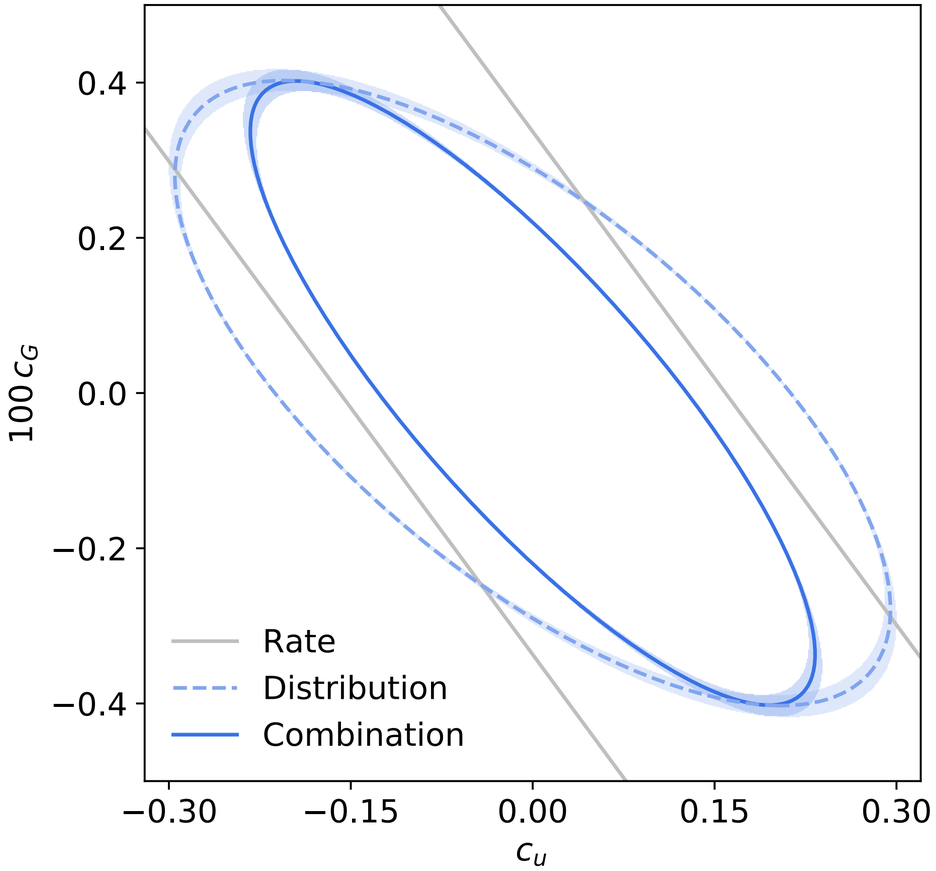


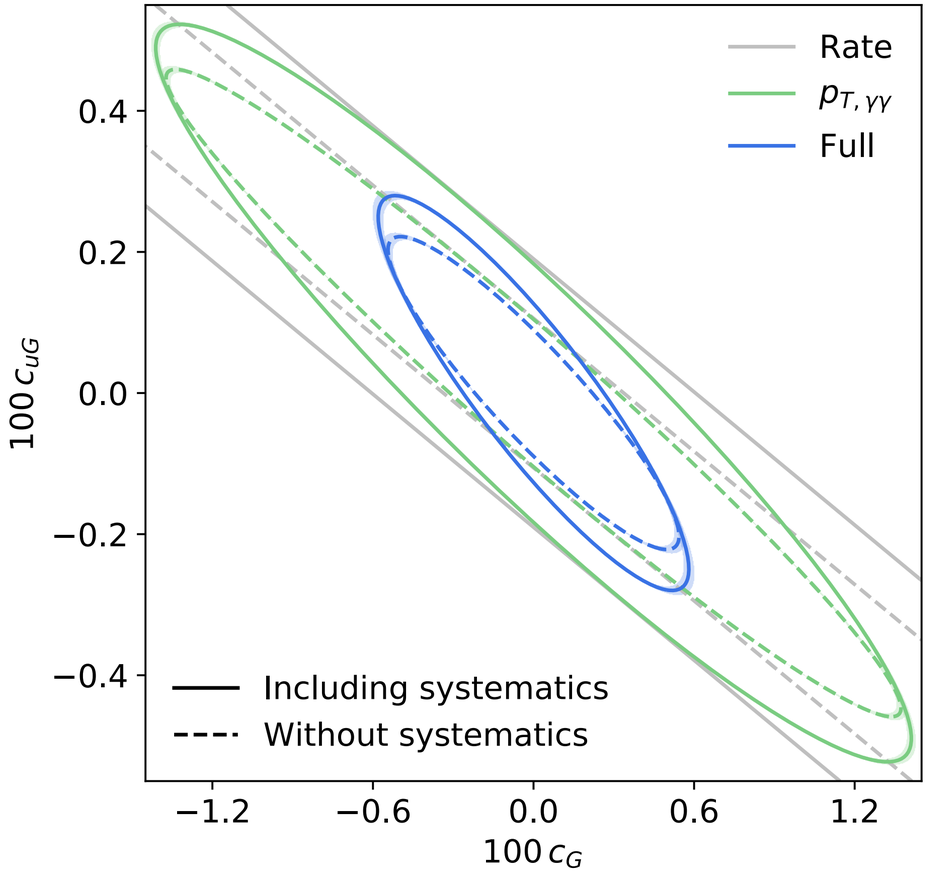
ttH: score and information vs pT and m





ttH: more information results

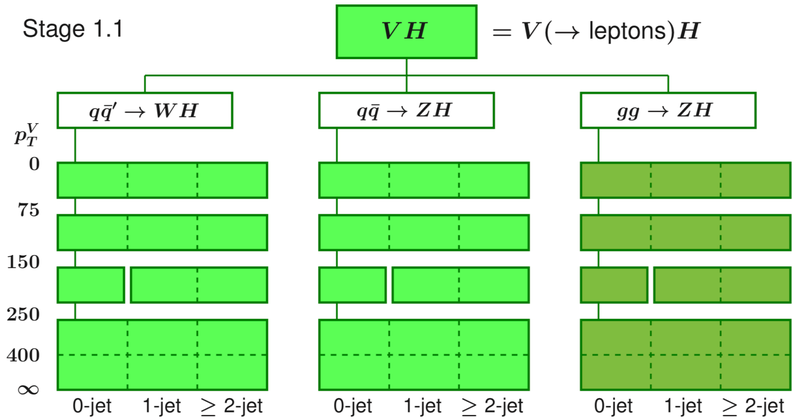




Benchmarking STXS in WH

[JB, S. Dawson, S. Homiller, F. Kling, T. Plehn 1908.06980]

- Simplified Template Cross-Sections (STXS) define observable bins that are supposed to capture as much information on NP as possible
  [N. Berger et al. 1906.02754; HXSWG YR4]



- Let's check! How much information on

$$\tilde{\mathcal{O}}_{HD} = \mathcal{O}_{H\Box} - \frac{\mathcal{O}_{HD}}{4} = (\phi^\dagger\phi)\Box(\phi^\dagger\phi) - \frac{1}{4}(\phi^\dagger D^\mu\phi)^*(\phi^\dagger D_\mu\phi)$$
$$\mathcal{O}_{HW} = \phi^\dagger\phi W^a_{\mu\nu}W^{\mu\nu a}$$
$$\mathcal{O}^{(3)}_{Hq} = (\phi^\dagger i\overleftrightarrow{D}^a_\mu\phi)(\overline{Q}_L\sigma^a\gamma^\mu Q_L)\ ,$$

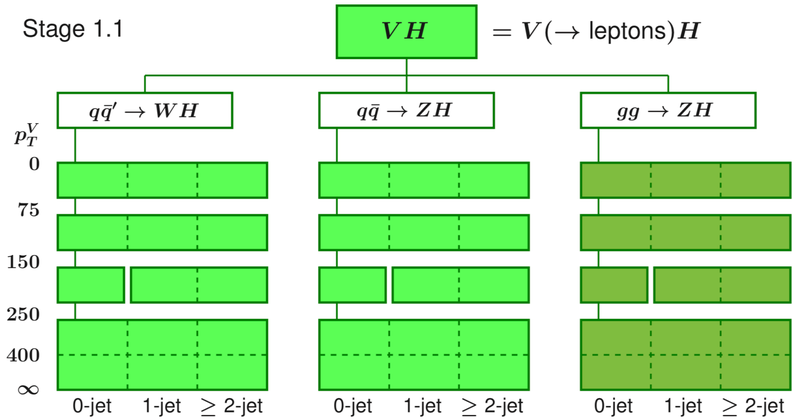
  can we extract from $pp \to WH \to \ell\nu\ b\bar{b}$?

Benchmarking STXS in WH

[JB, S. Dawson, S. Homiller, F. Kling, T. Plehn 1908.06980]

- Simplified Template Cross-Sections (STXS) define observable bins that are supposed to capture as much information on NP as possible

  [N. Berger et al. 1906.02754; HXSWG YR4]

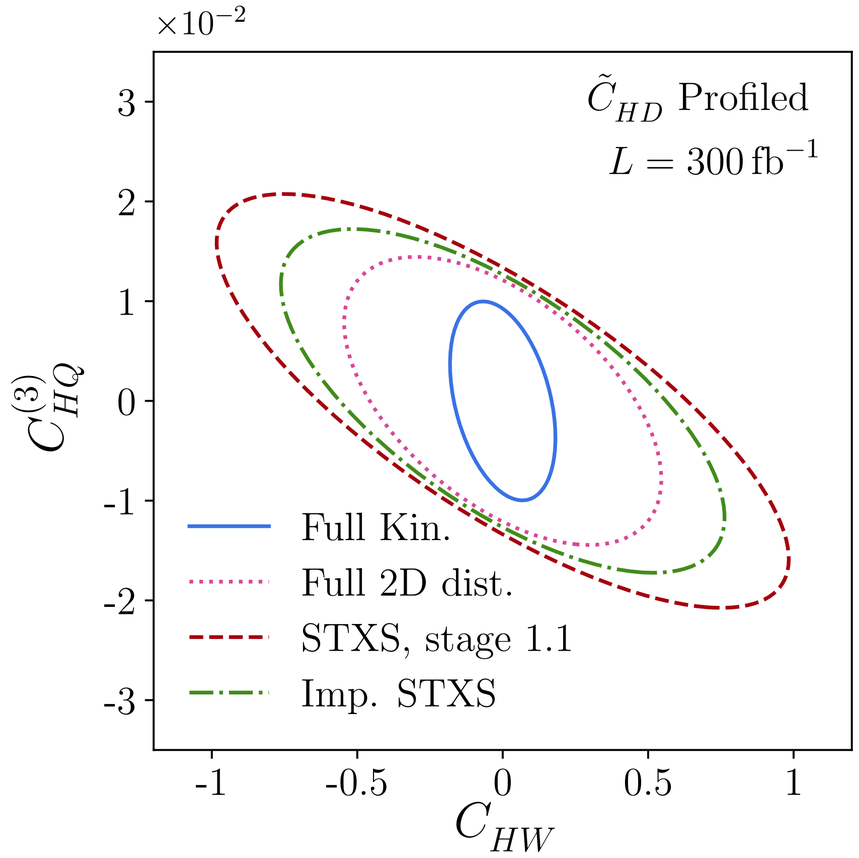


- Let's check! How much information on

$$\tilde{\mathcal{O}}_{HD} = \mathcal{O}_{H\Box} - \frac{\mathcal{O}_{HD}}{4} = (\phi^\dagger\phi)\Box(\phi^\dagger\phi) - \frac{1}{4}(\phi^\dagger D^\mu\phi)^*(\phi^\dagger D_\mu\phi)$$
$$\mathcal{O}_{HW} = \phi^\dagger\phi W^a_{\mu\nu}W^{\mu\nu a}$$
$$\mathcal{O}^{(3)}_{Hq} = (\phi^\dagger i\overleftrightarrow{D}^a_\mu\phi)(\overline{Q}_L\sigma^a\gamma^\mu Q_L)\ ,$$

  can we extract from $pp \to WH \to \ell\nu\, b\bar{b}$?

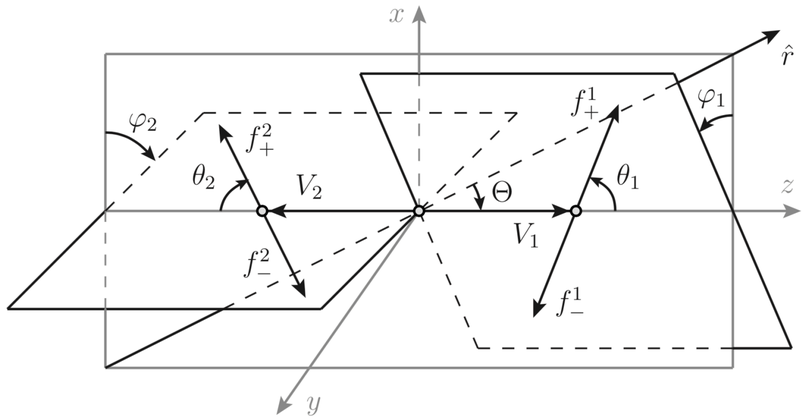
- Results: STXS are indeed sensitive to operators, adding a few more bins improve them, but a multivariate analysis is still stronger



Diboson production

- In inclusive observables, the interference between SM and new physics amplitudes vanishes

  ⇒ Reduced sensitivity to new physics

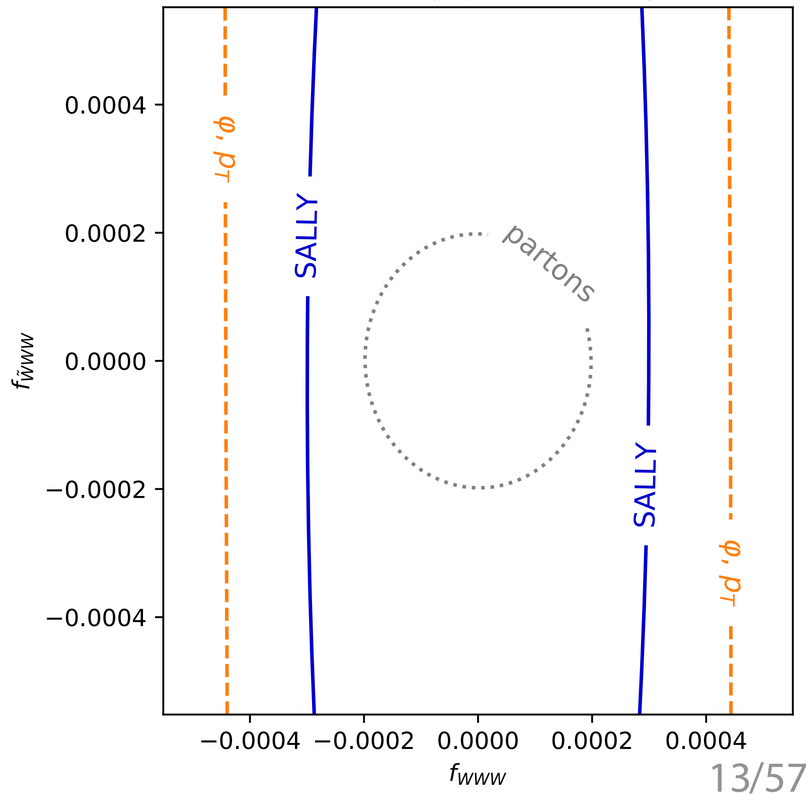


- "Diboson interference resurrection":
  an angular variable φ can be constructed to be sensitive to this interference
  [G. Panico, F. Riva, A. Wulzer 1708.07823;
  A. Azatov, D. Barducci, E. Venturini 1901.04821]

- We test the ML approach in EFT measurements in $W\gamma \to \ell\nu\,\gamma$
  [JB, K. Cranmer, M. Farina, F. Kling, D. Pappadopulo, J. Ruderman in progress]

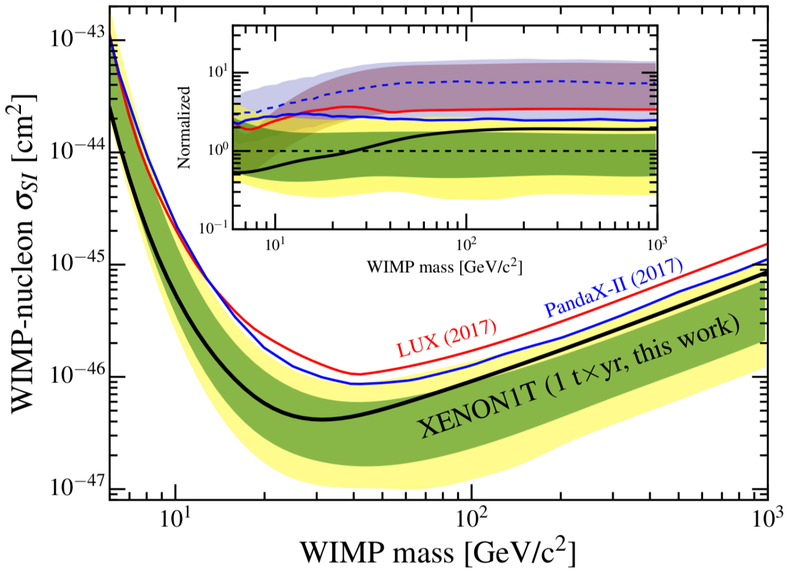
- Preliminary results: we can extract more information when we analyze events with SALLY than with histograms of φ and standard observables

Very preliminary results



These methods work beyond the LHC.

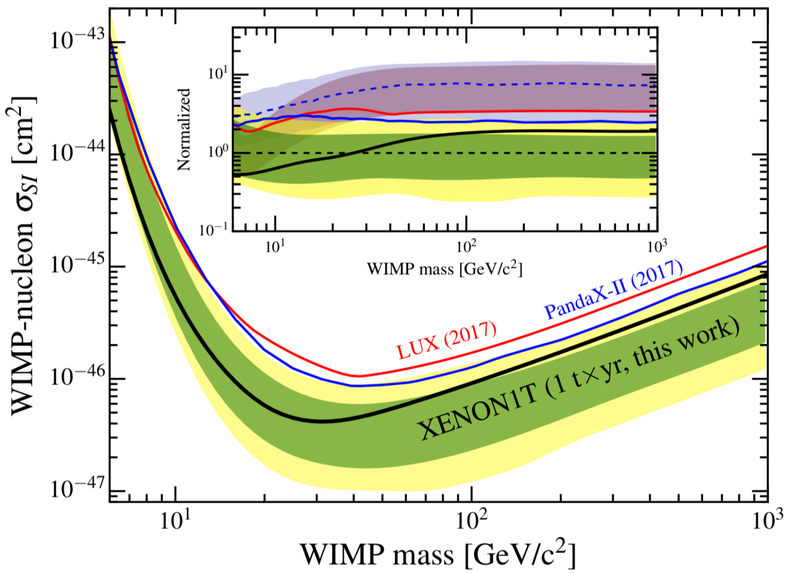
Dark matter (DM) status



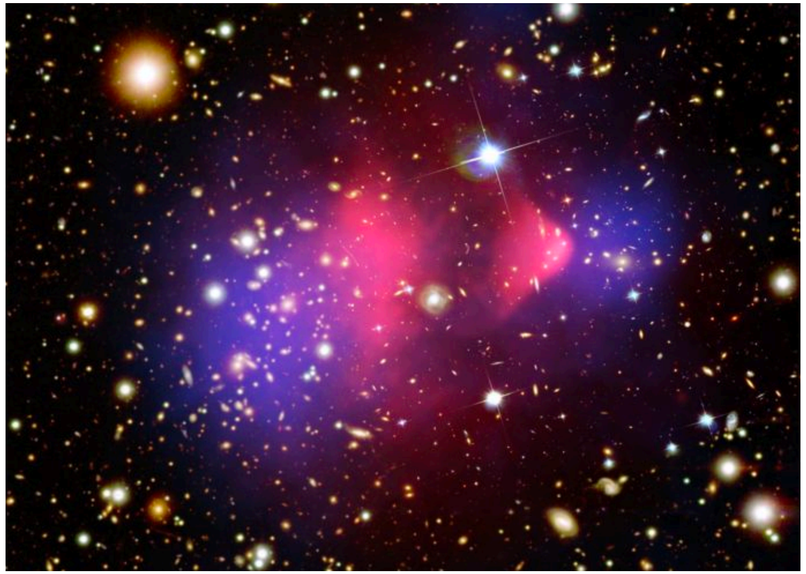
[XENON 1805.12562]

Non-gravitational interactions:
no evidence so far

Dark matter (DM) status



[XENON 1805.12562]

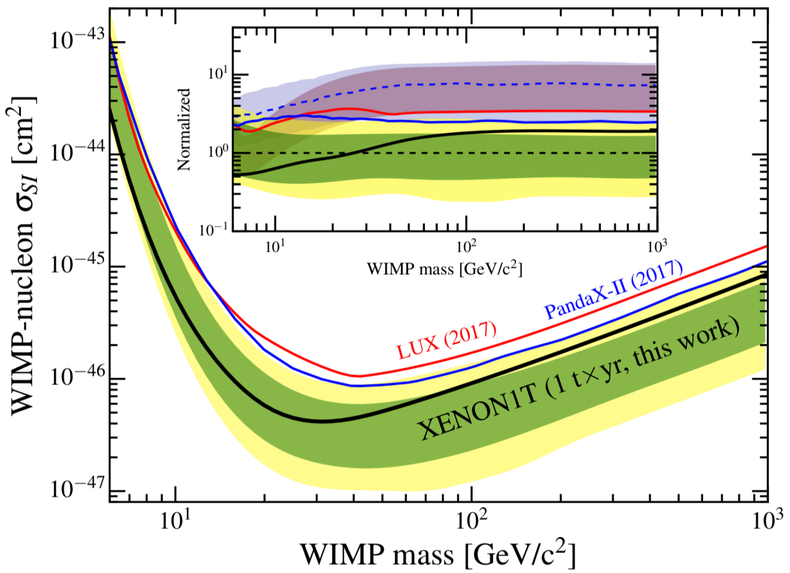


[NASA]

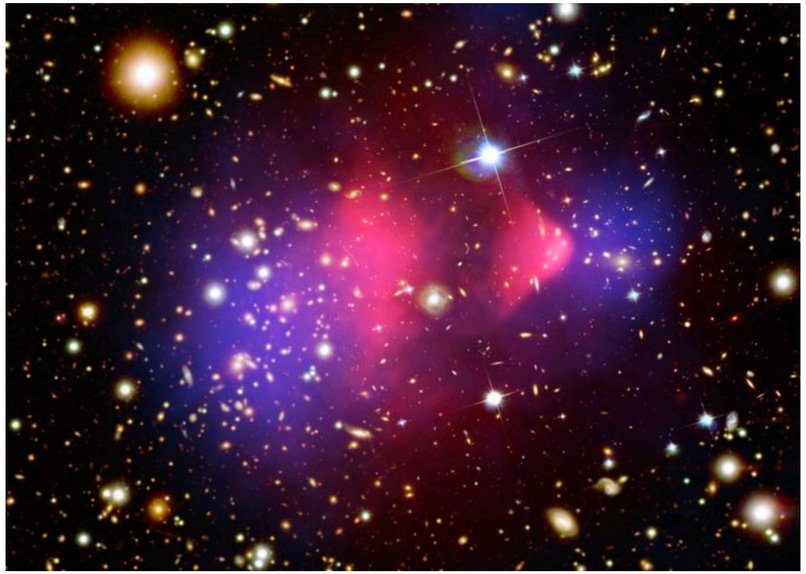
Non-gravitational interactions: no evidence so far

Gravitational interactions at large length scales: plenty of evidence, consistent with ΛCDM

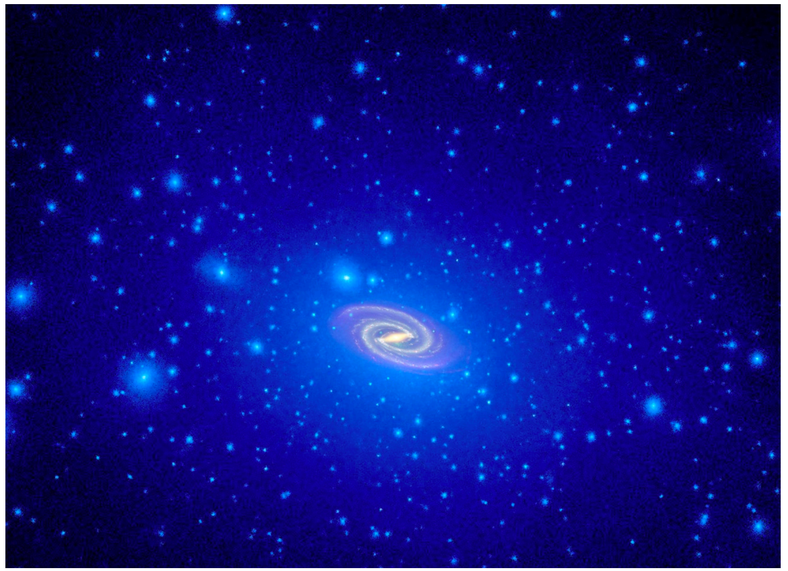
Dark matter (DM) status



[XENON 1805.12562]



[NASA]



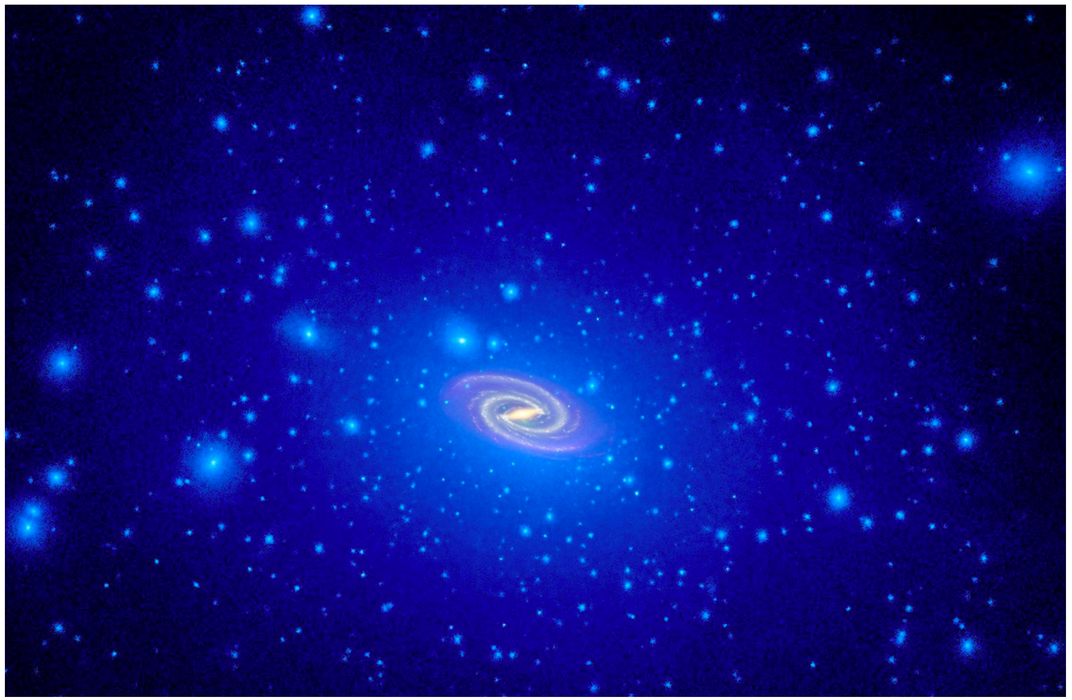
[T. Brown, J.Tumlinson]

Non-gravitational interactions: no evidence so far

Gravitational interactions at large length scales: plenty of evidence, consistent with ΛCDM

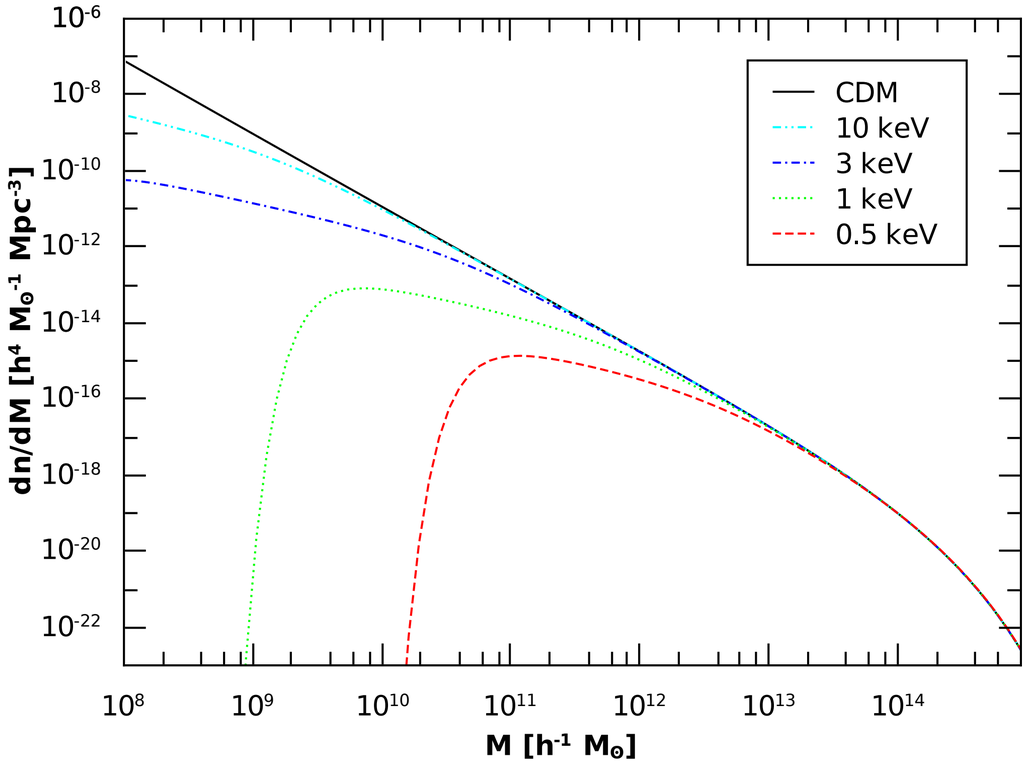
Gravitational interactions at small scales (DM substructure): beginning to be probed, many models predict deviations from ΛCDM

DM small-scale structure as a probe of DM particle properties



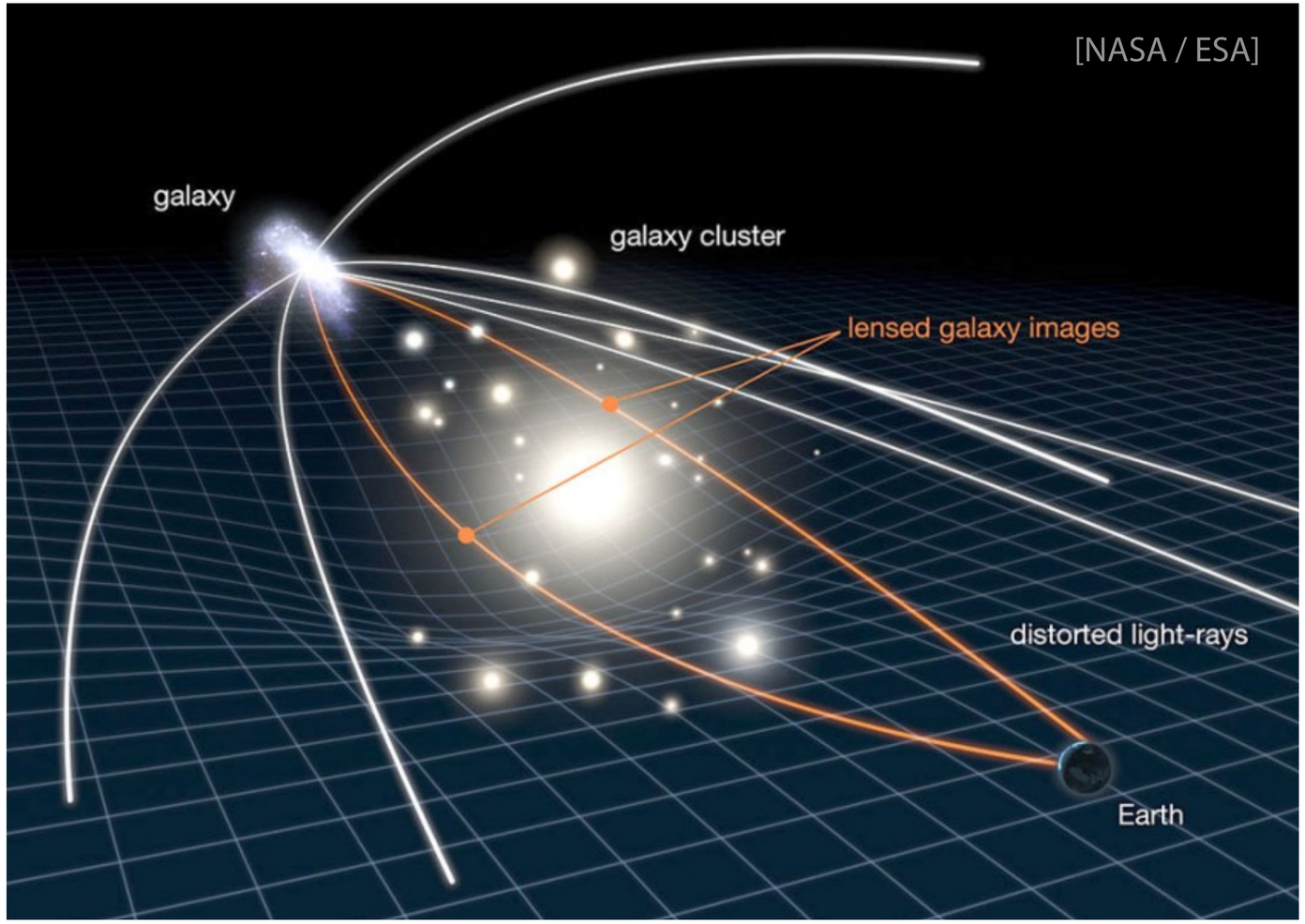
[T. Brown, J.Tumlinson]

Abundance of DM subhalos vs mass:



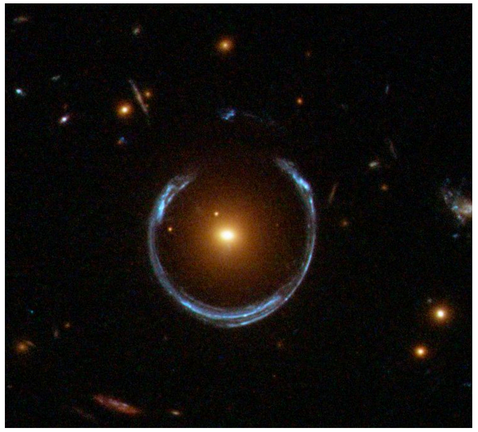
[R. *Dunstan et al* 1109.6291]

Strong gravitational lensing



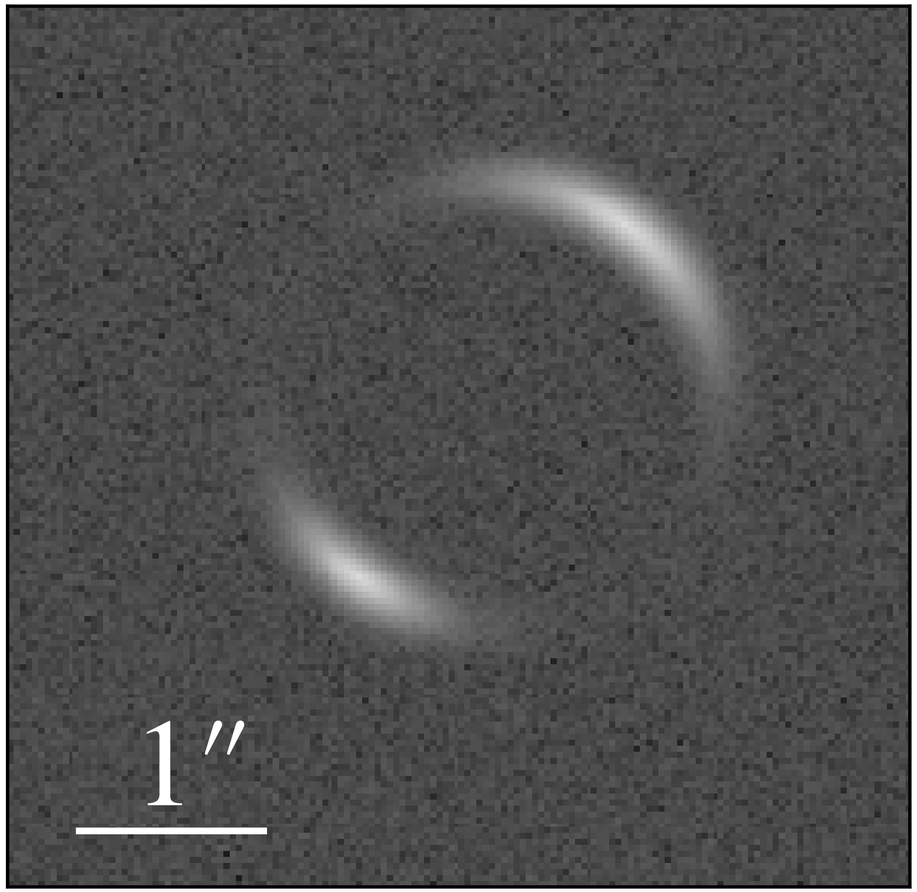
Multiple images of quasars

Extended arcs from galaxies



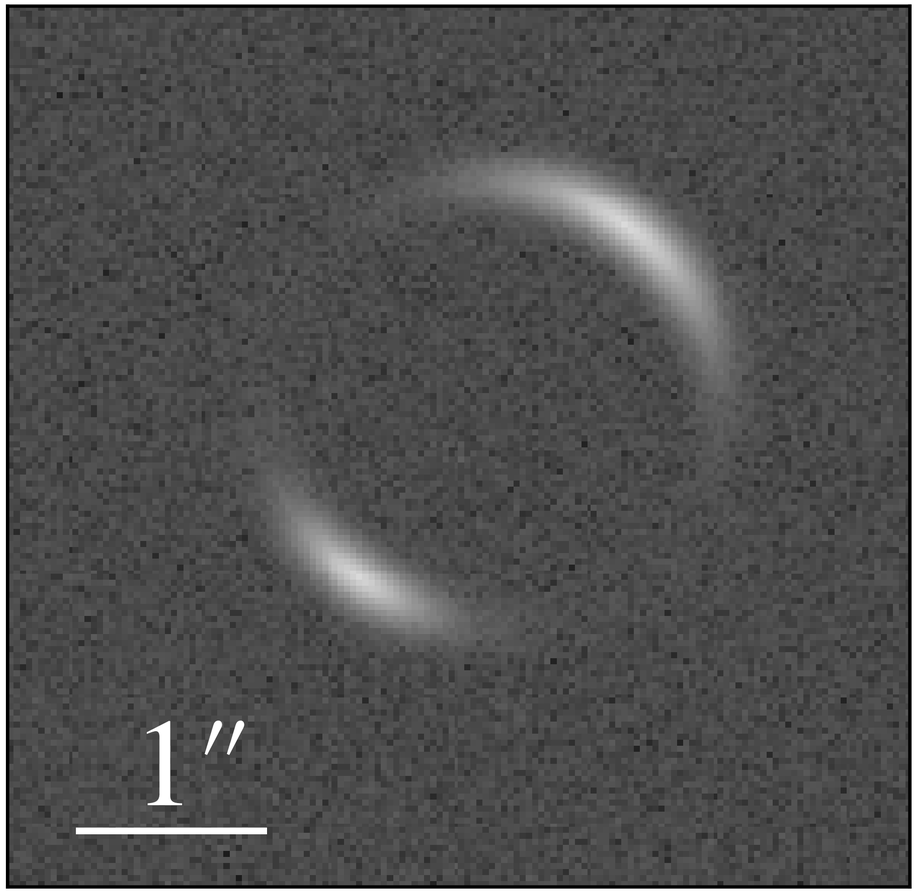
Subhalos affect strong lensing

Smooth halo only

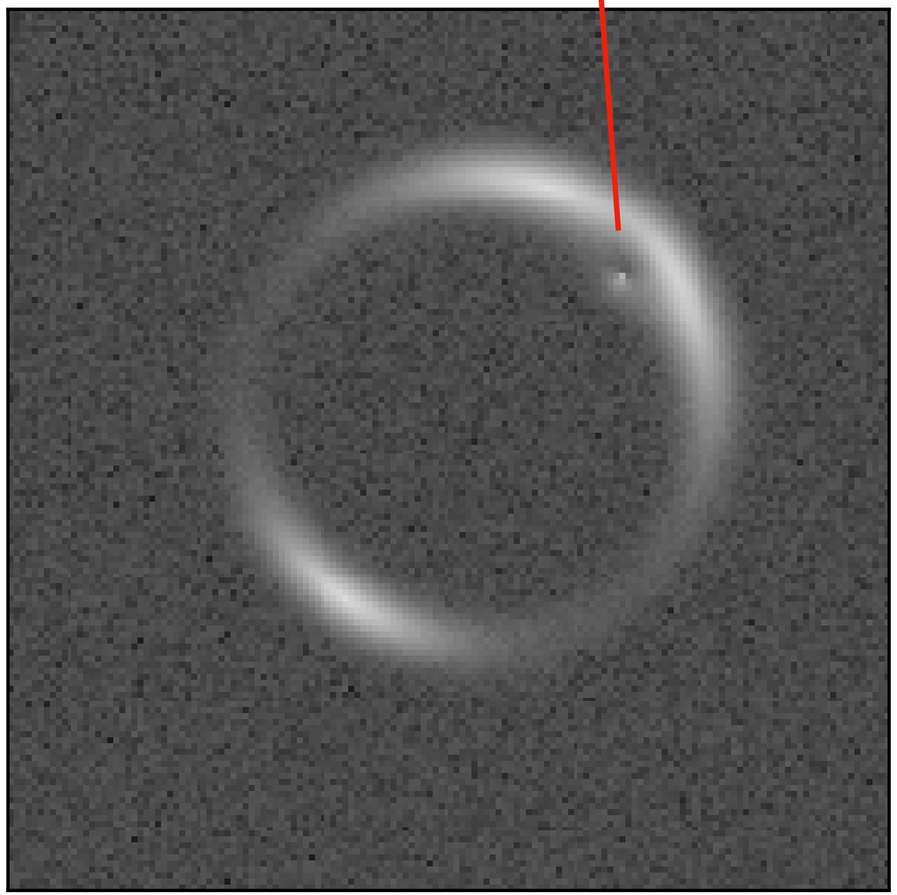


Subhalos affect strong lensing

Smooth halo only

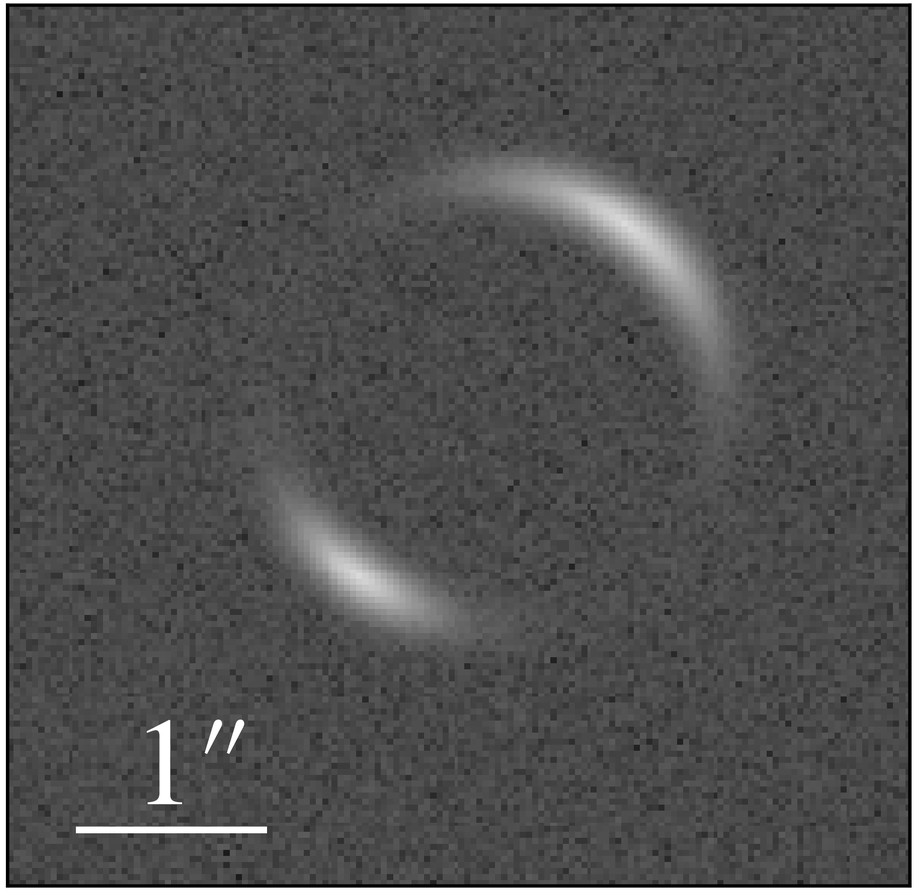


Smooth halo + subhalo

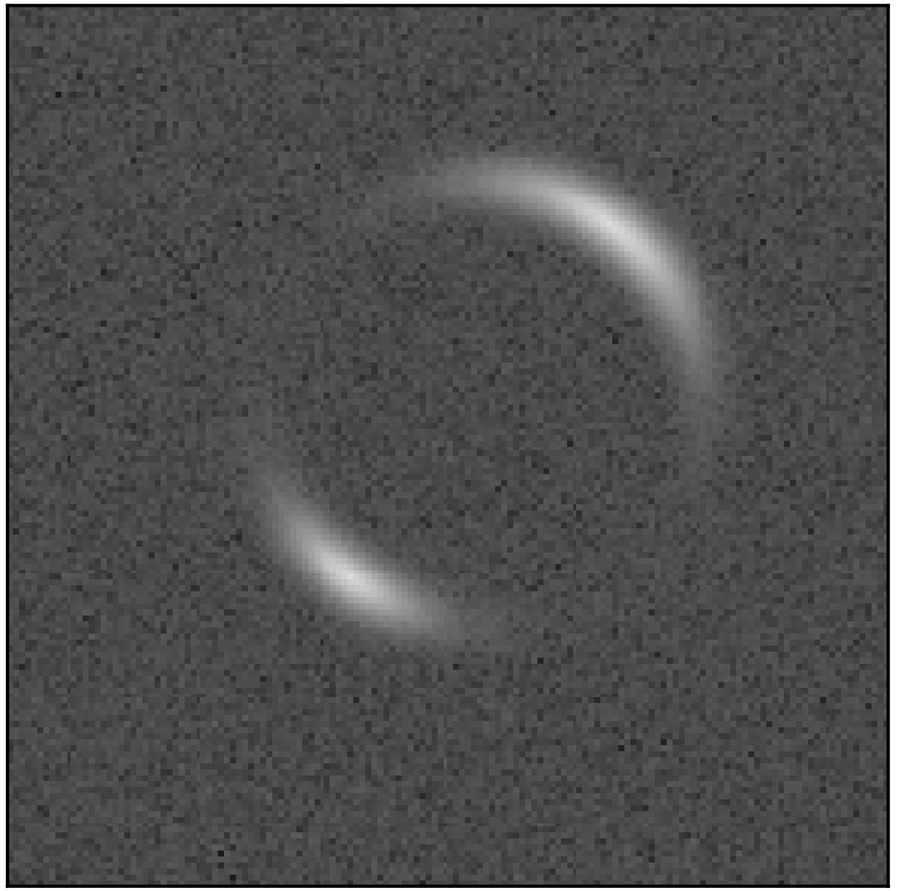


Subhalos affect strong lensing… realistically

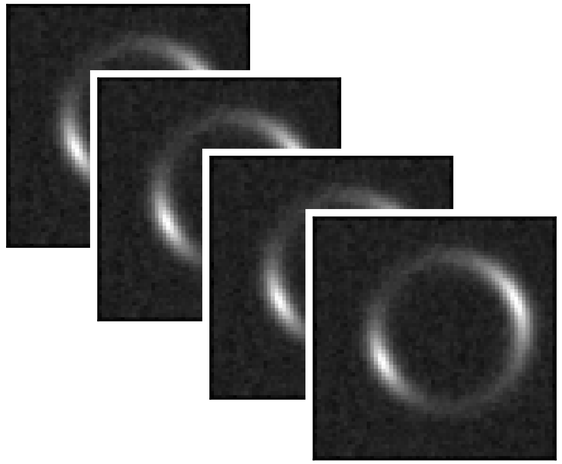
Smooth halo only

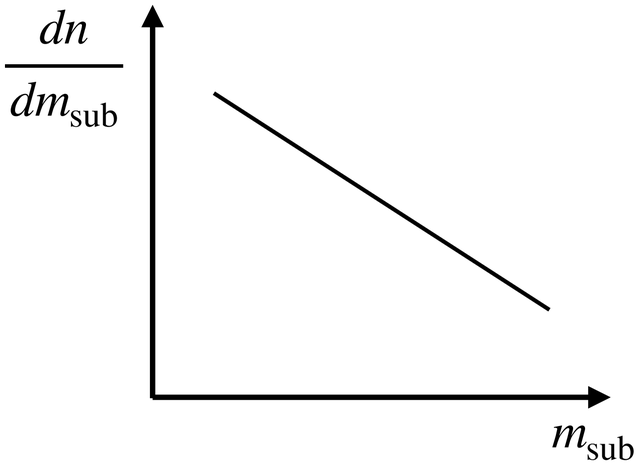


Smooth halo + subhalos



Scalable inference for small subhalos

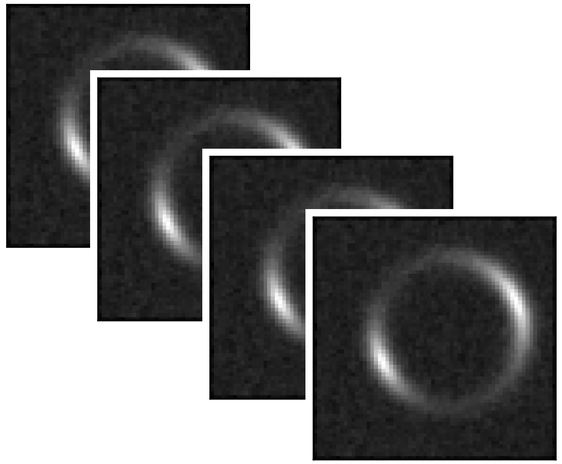


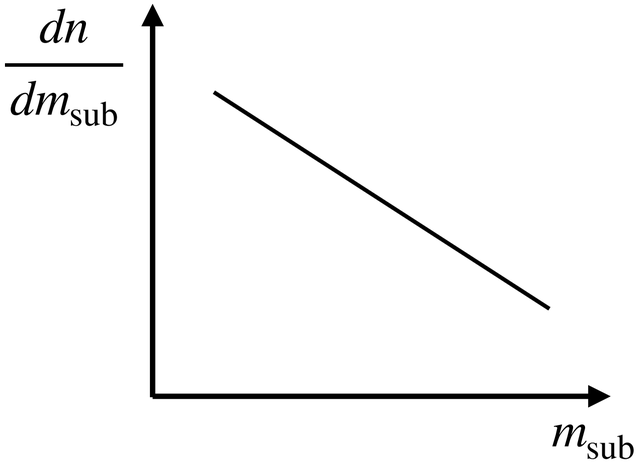


Future surveys (LSST, Euclid) are expected to deliver large samples of galaxy-galaxy strong lenses [Collett et al 1507.02657]

Goal: infer subhalo mass distribution through collective effects of many light subhalos

Scalable inference for small subhalos





Future surveys (LSST, Euclid) are expected to deliver large samples of galaxy-galaxy strong lenses [Collett et al 1507.02657]

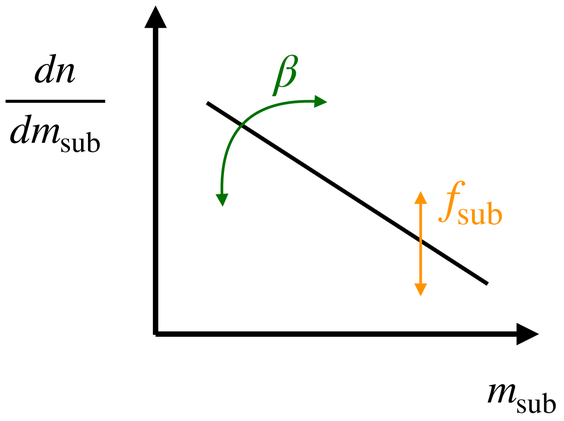
Goal: infer subhalo mass distribution through collective effects of many light subhalos

⇒ Need inference technique that

- scales to many lenses (fast evaluation)
- captures subtle effects in high-dimensional image data
- can deal with a large number of subhalos (latent variables)

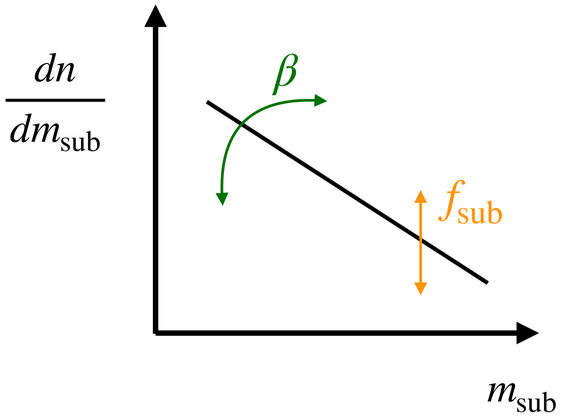
Simulation-based inference for strong lensing

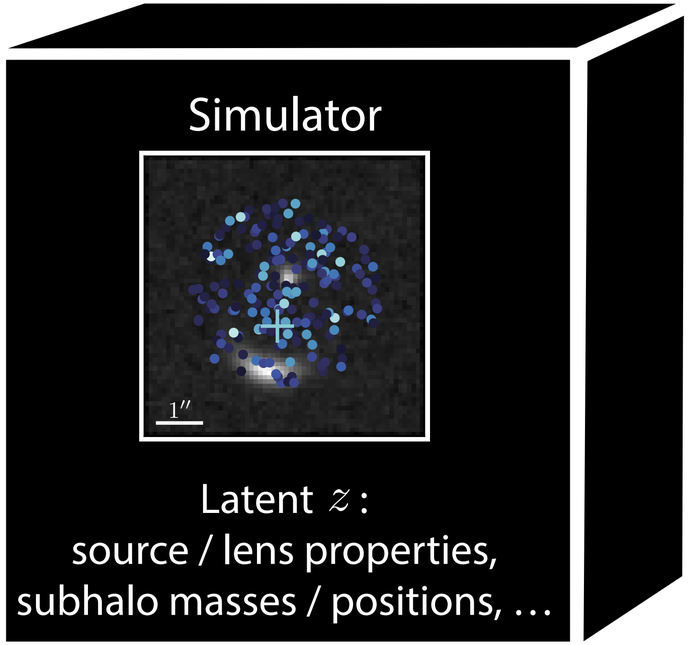
2 parameters $\theta = (\beta, f_{\rm sub})$



Simulation-based inference for strong lensing

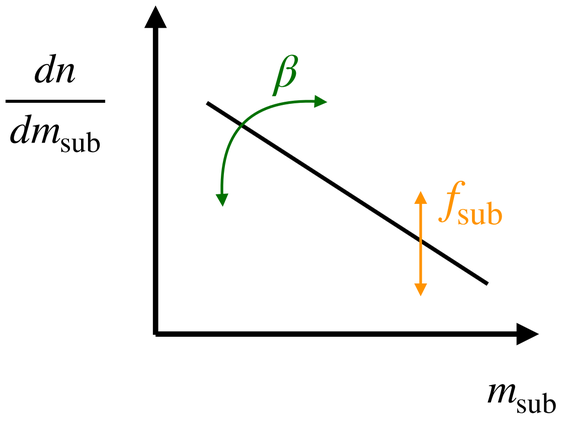
2 parameters $\theta = (\beta, f_{\rm sub})$

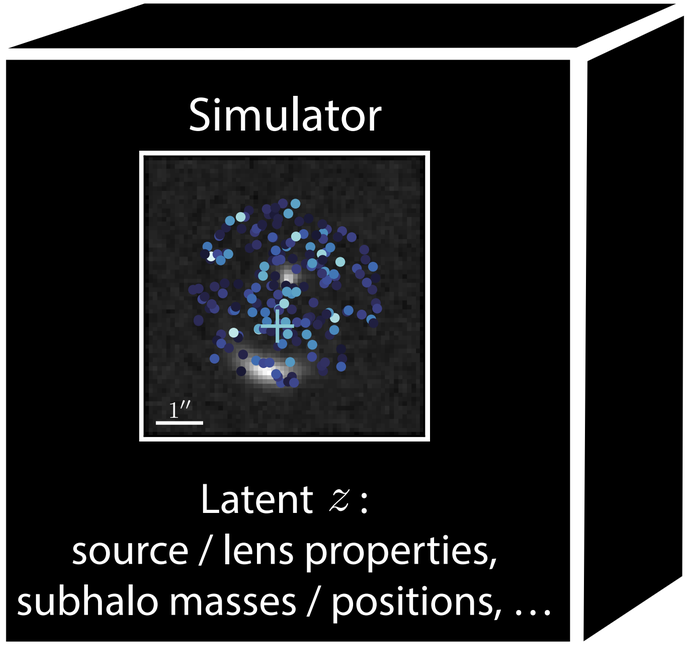




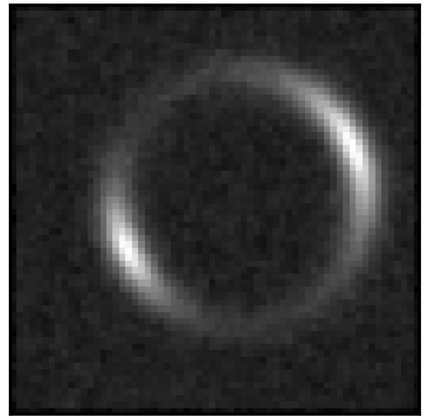
Simulation-based inference for strong lensing

2 parameters $\theta = (\beta, f_{\rm sub})$



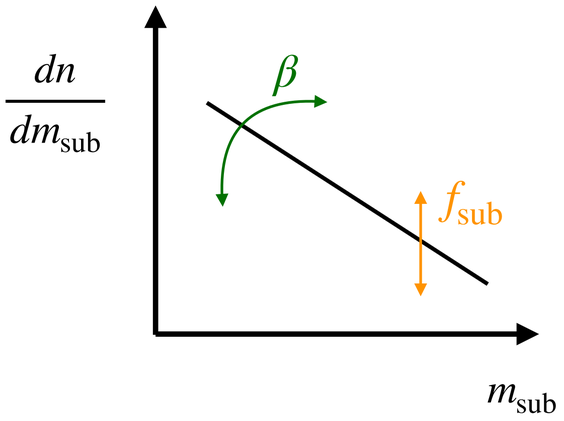


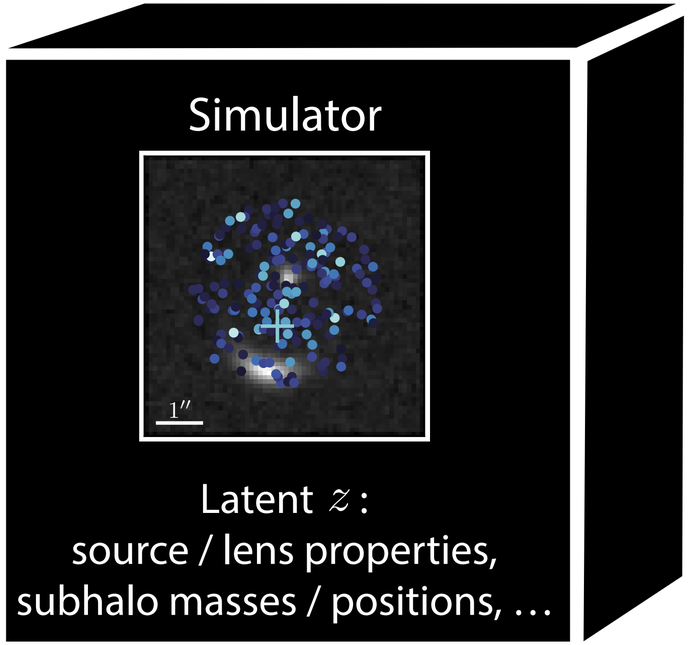
64^2 observables x



Simulation-based inference for strong lensing

2 parameters $\theta = (\beta, f_{\rm sub})$



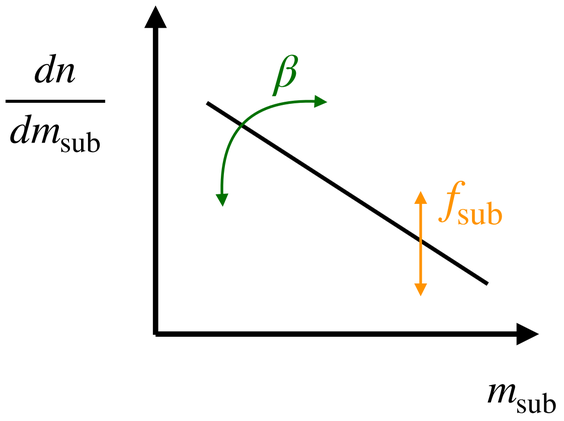


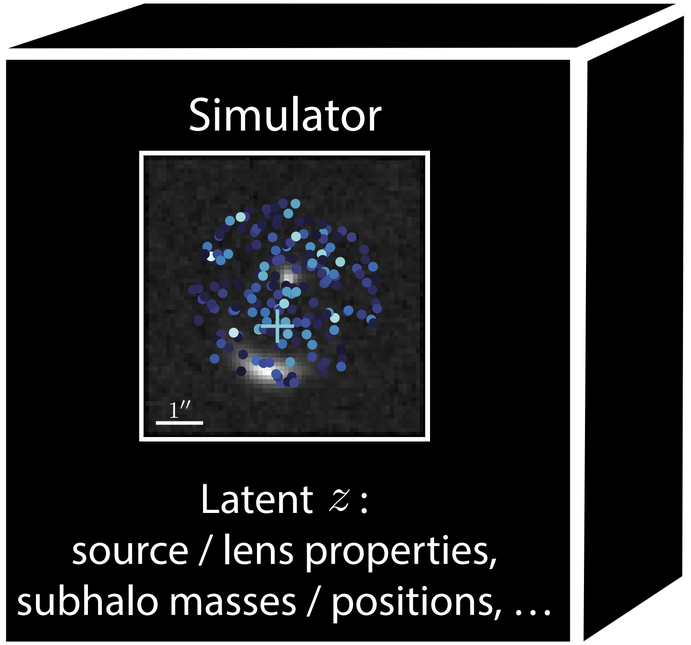
64^2 observables x

Prediction: We construct a simulator that can sample $x \sim p(x|\theta)$

Simulation-based inference for strong lensing

2 parameters $\theta = (\beta, f_{\rm sub})$





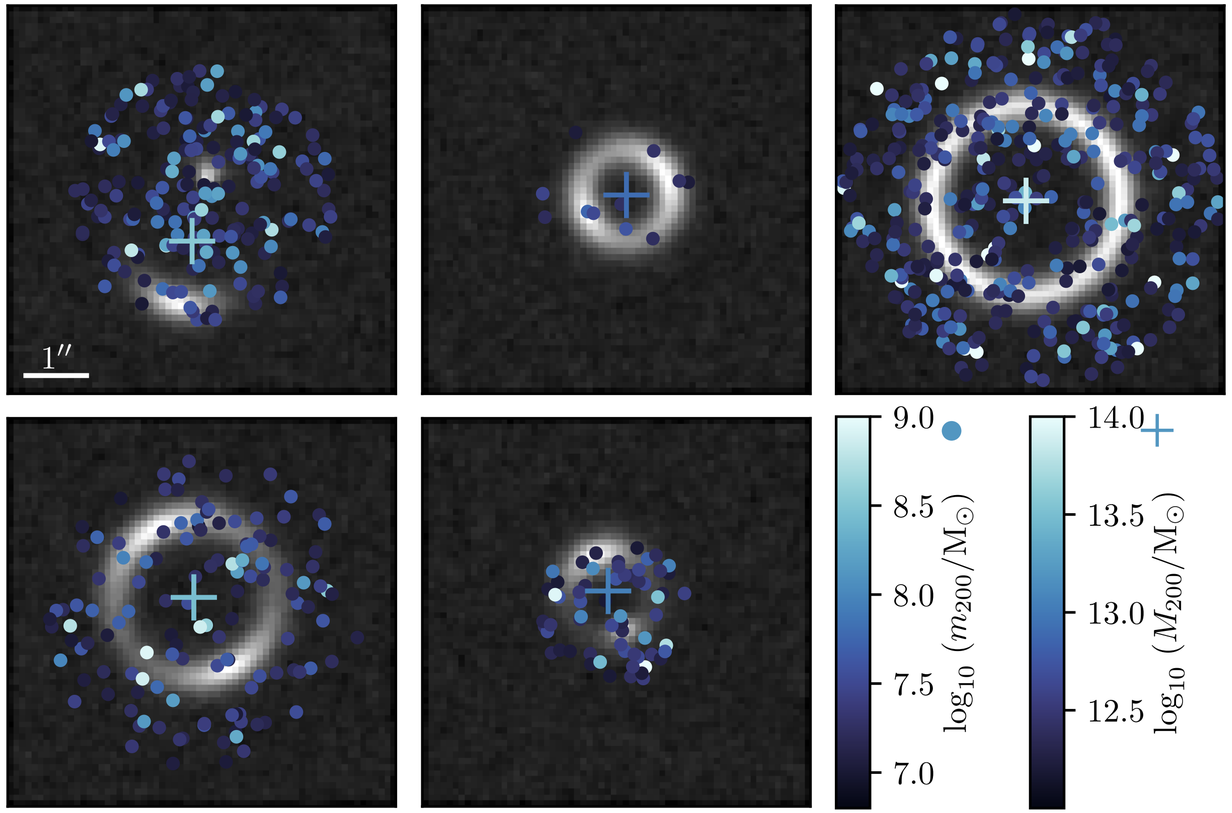
64^2 observables x

Prediction: We construct a simulator that can sample $x \sim p(x|\theta)$

Inference: We train neural likelihood ratio estimators $\hat{r}(x|\theta)$

Proof-of-principle simulator

[following T. Collett 1507.02657]

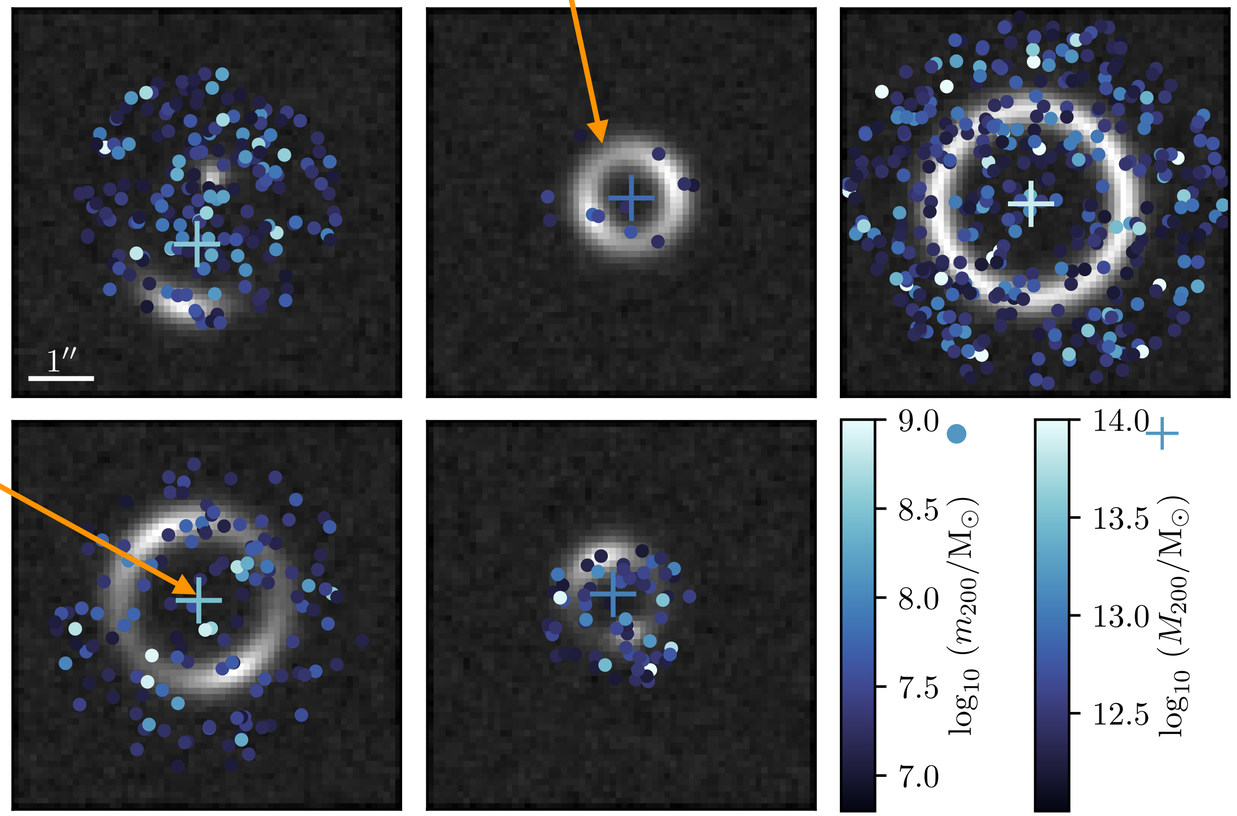


Proof-of-principle simulator

[following T. Collett 1507.02657]

Spherical lensing host galaxies
at redshift ~0.5…1

Extended galaxy sources
at redshift 1.5

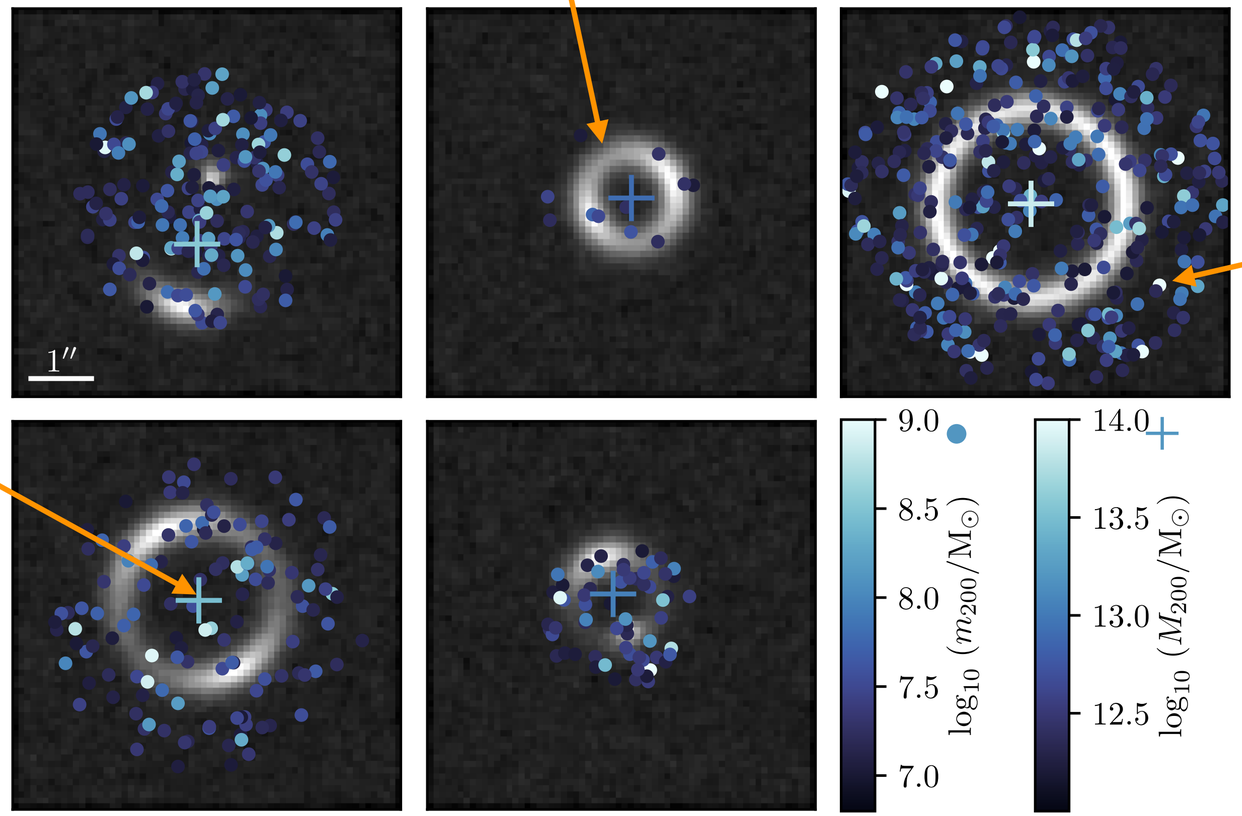


Proof-of-principle simulator

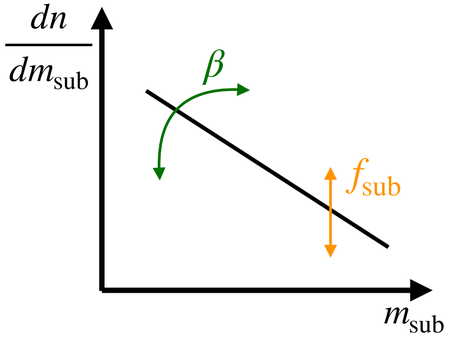
[following T. Collett 1507.02657]

Spherical lensing host galaxies
at redshift ~0.5…1

Extended galaxy sources
at redshift 1.5



Subhalos follow mass distribution with two parameters



Proof-of-principle simulator

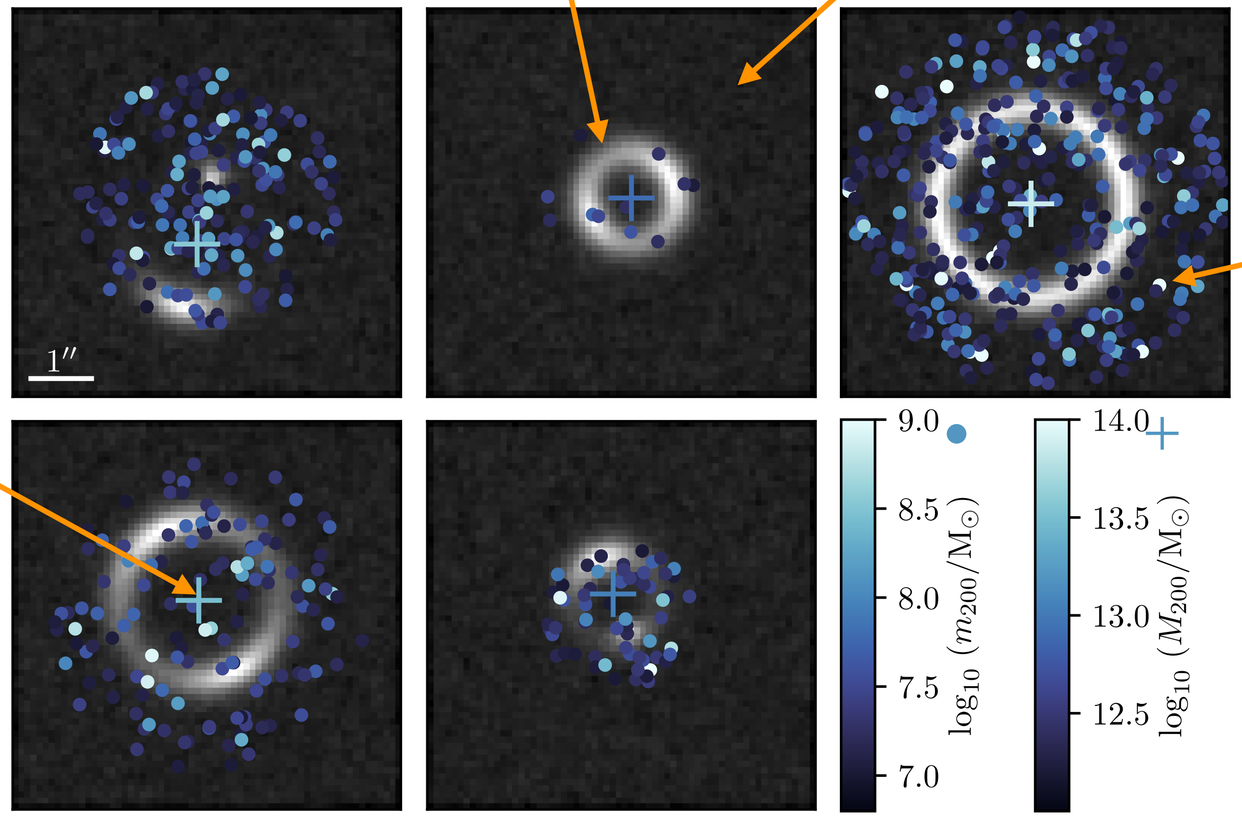
[following T. Collett 1507.02657]

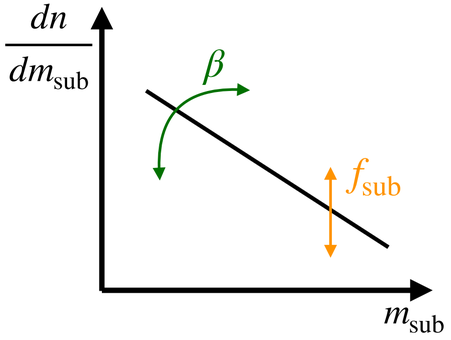
Spherical lensing host galaxies
at redshift ~0.5…1

Poisson fluctuations

Extended galaxy sources
at redshift 1.5

Subhalos follow mass distribution with two parameters





Proof-of-principle simulator

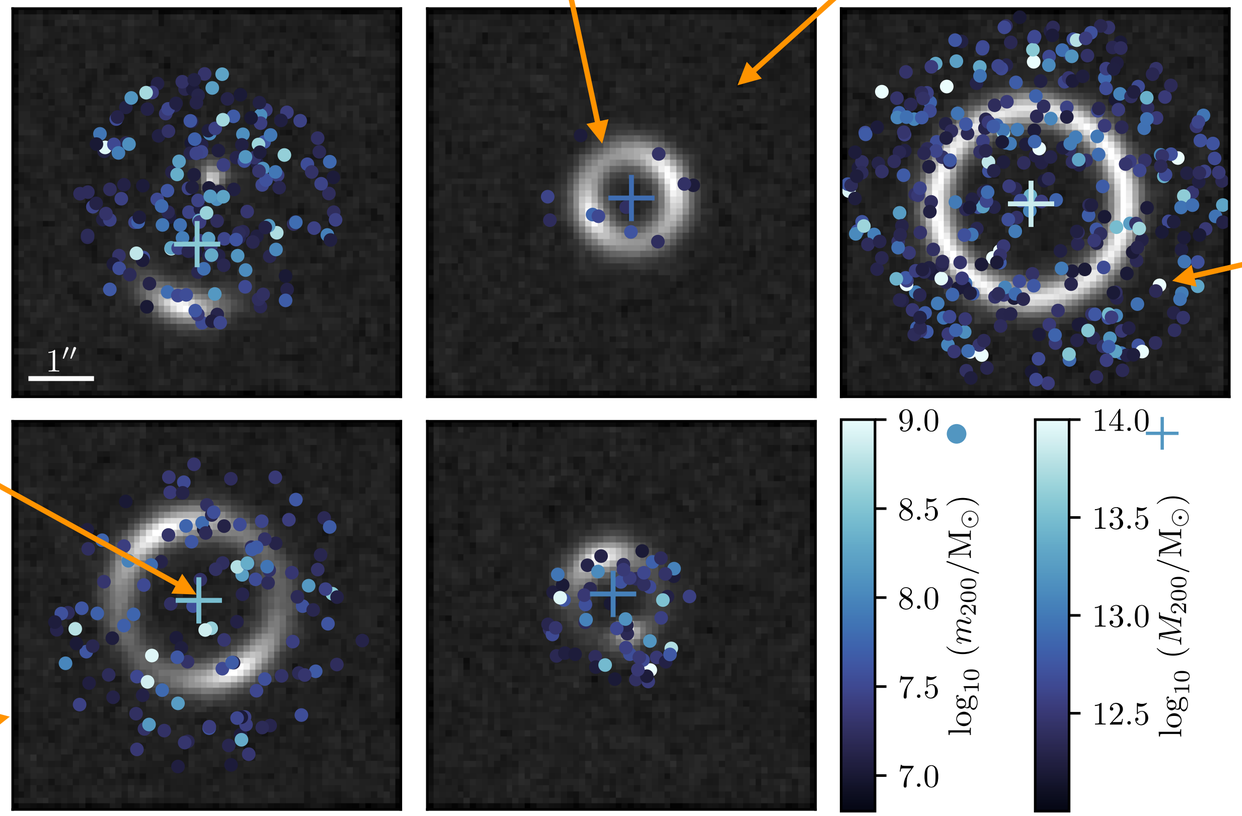
[following T. Collett 1507.02657]

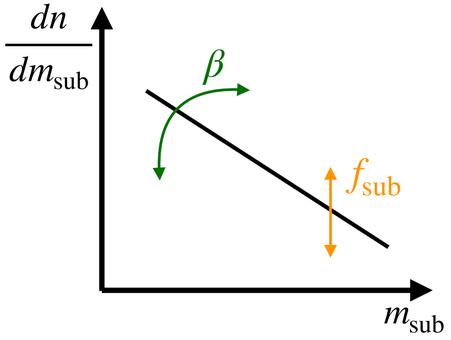
Spherical lensing host galaxies
at redshift ~0.5…1

Poisson fluctuations

Extended galaxy sources
at redshift 1.5

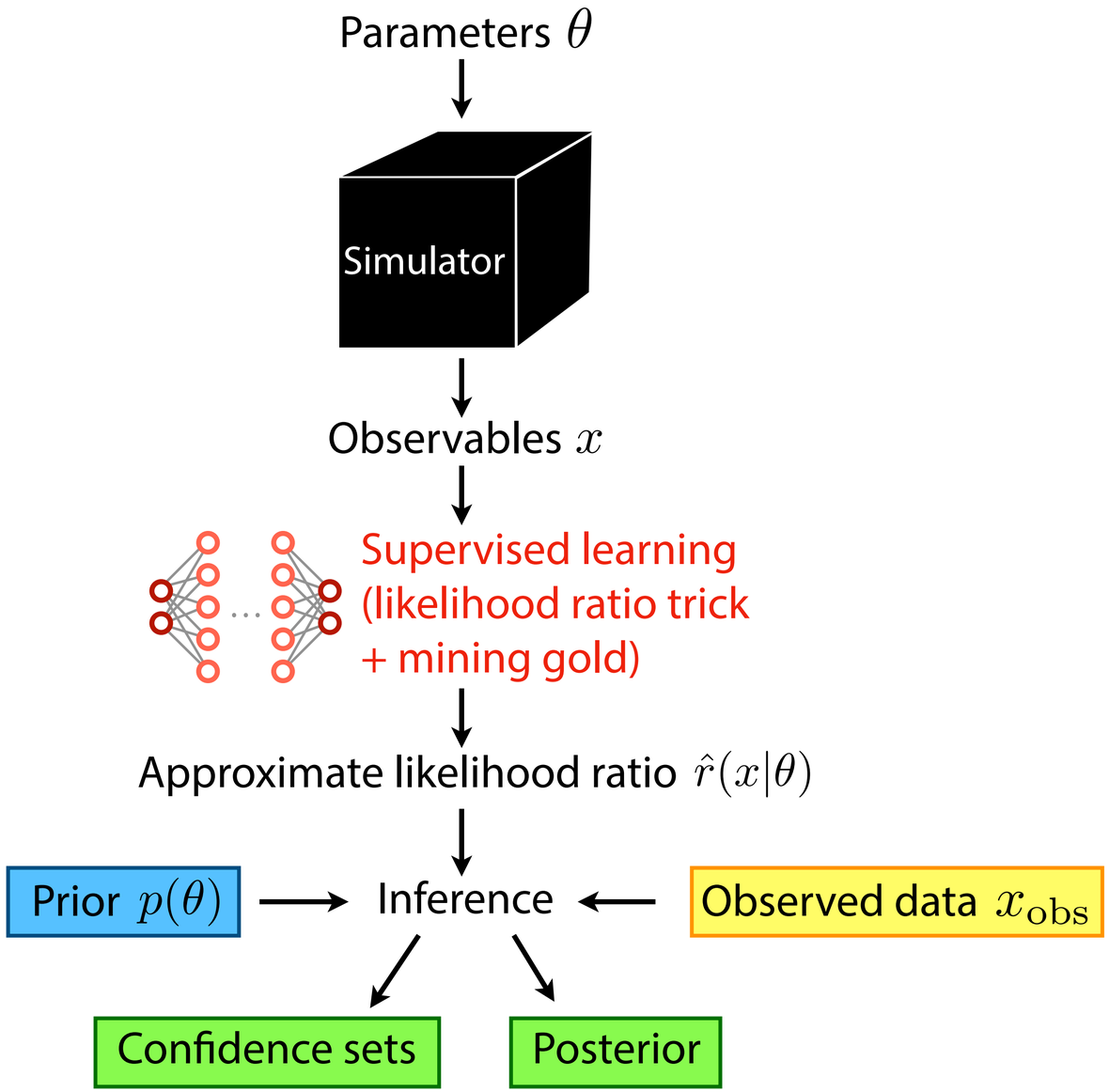
Subhalos follow mass distribution with two parameters



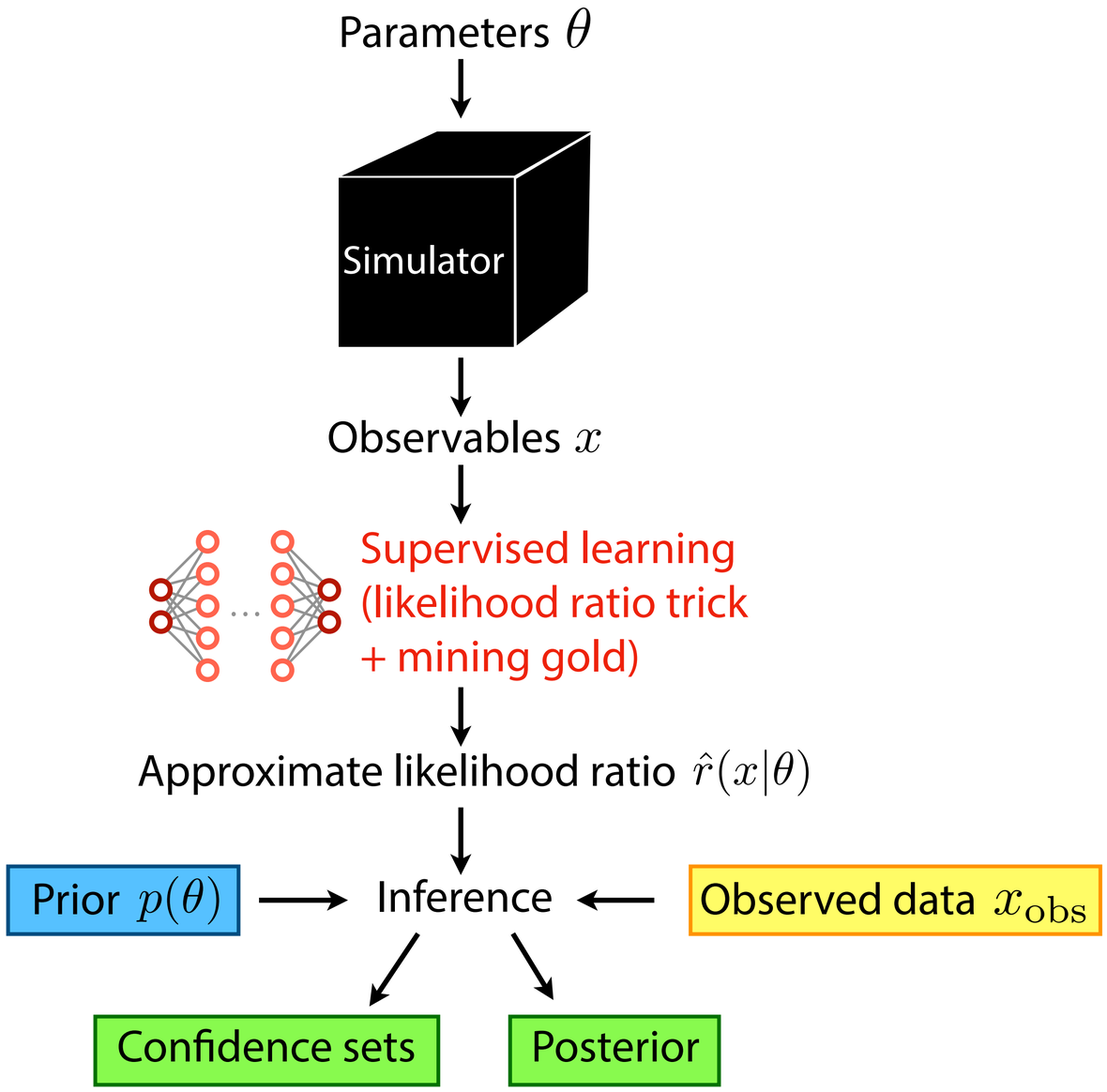


Euclid detector model
(single optical band,
64^2 pixel, Gaussian PSF)

Inference setup

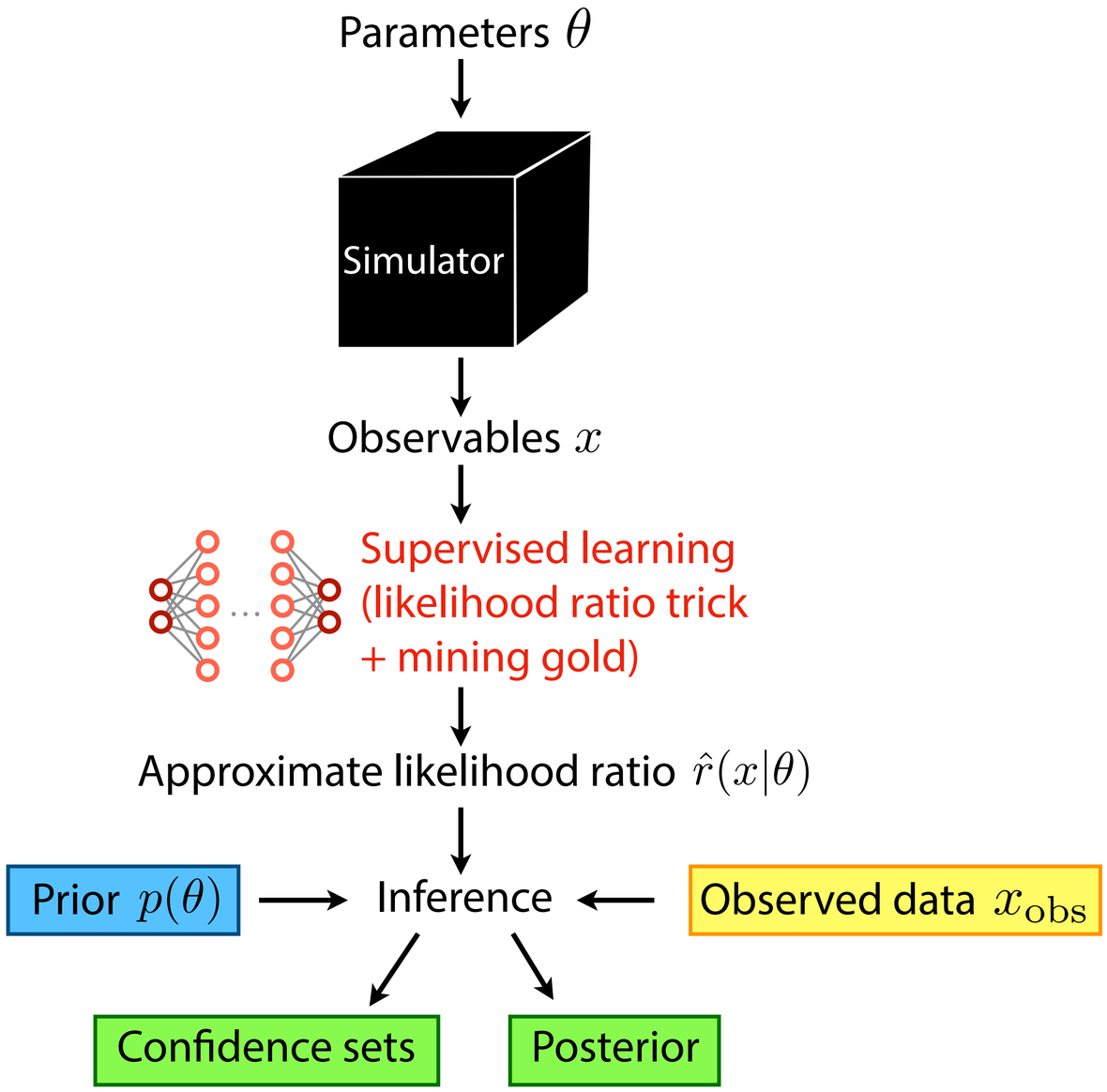


Inference setup



Training data: 10^6 lensed images with
$0 \leq f_{\rm sub} \leq 0.2, -1.5 \leq \beta \leq -0.5$

Inference setup



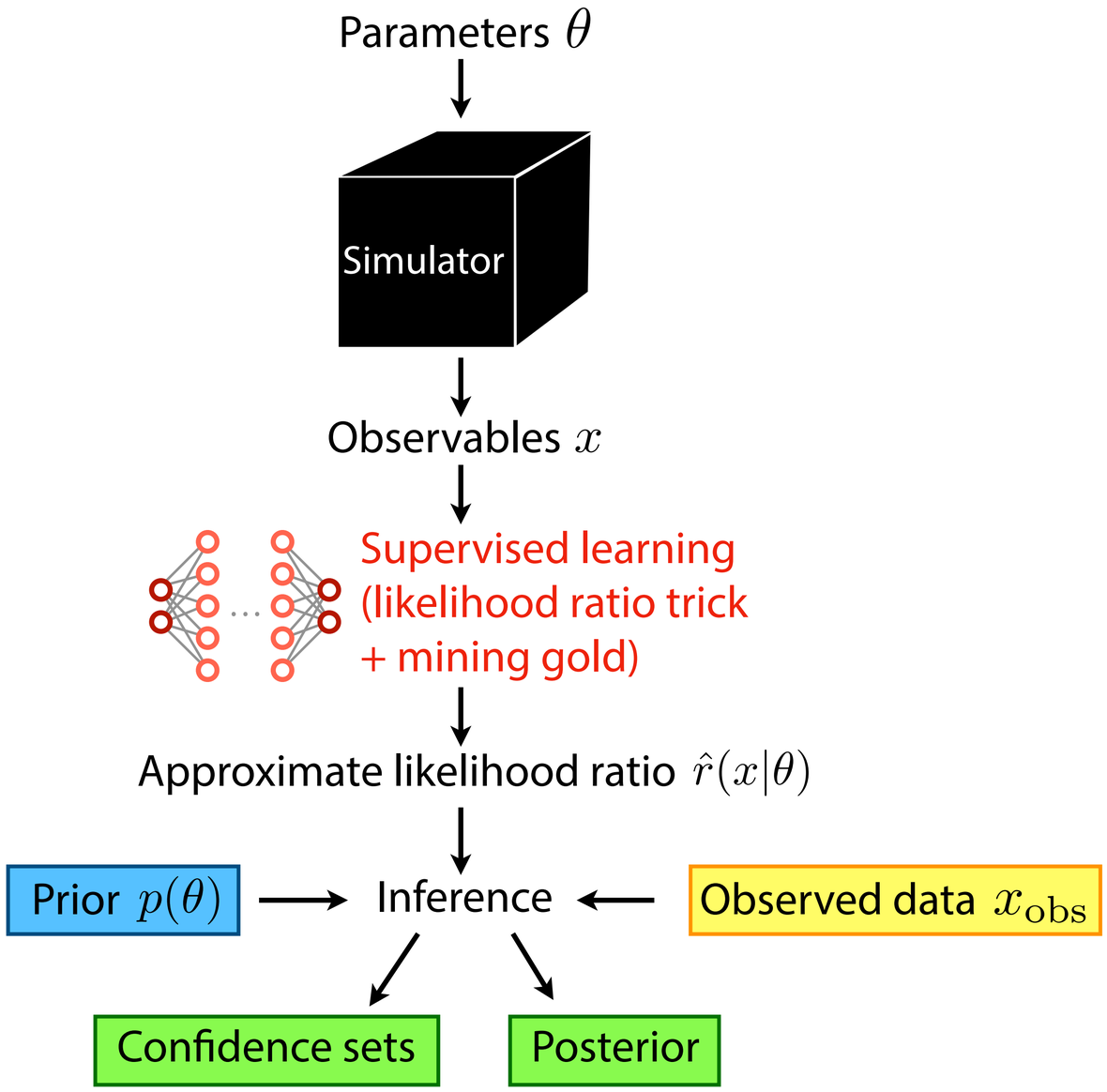
Training data: 10^6 lensed images with
$0 \leq f_{\rm sub} \leq 0.2, -1.5 \leq \beta \leq -0.5$

Convolutional neural network (modified ResNet-18) trained on ALICES loss

[M. Stoye, JB, J. Pavez, G, Louppe, K. Cranmer 1808.00973]

Probabilistic calibration of network output

Inference setup



Training data: 10^6 lensed images with
$0 \leq f_{\rm sub} \leq 0.2, -1.5 \leq \beta \leq -0.5$

Convolutional neural network (modified ResNet-18) trained on ALICES loss

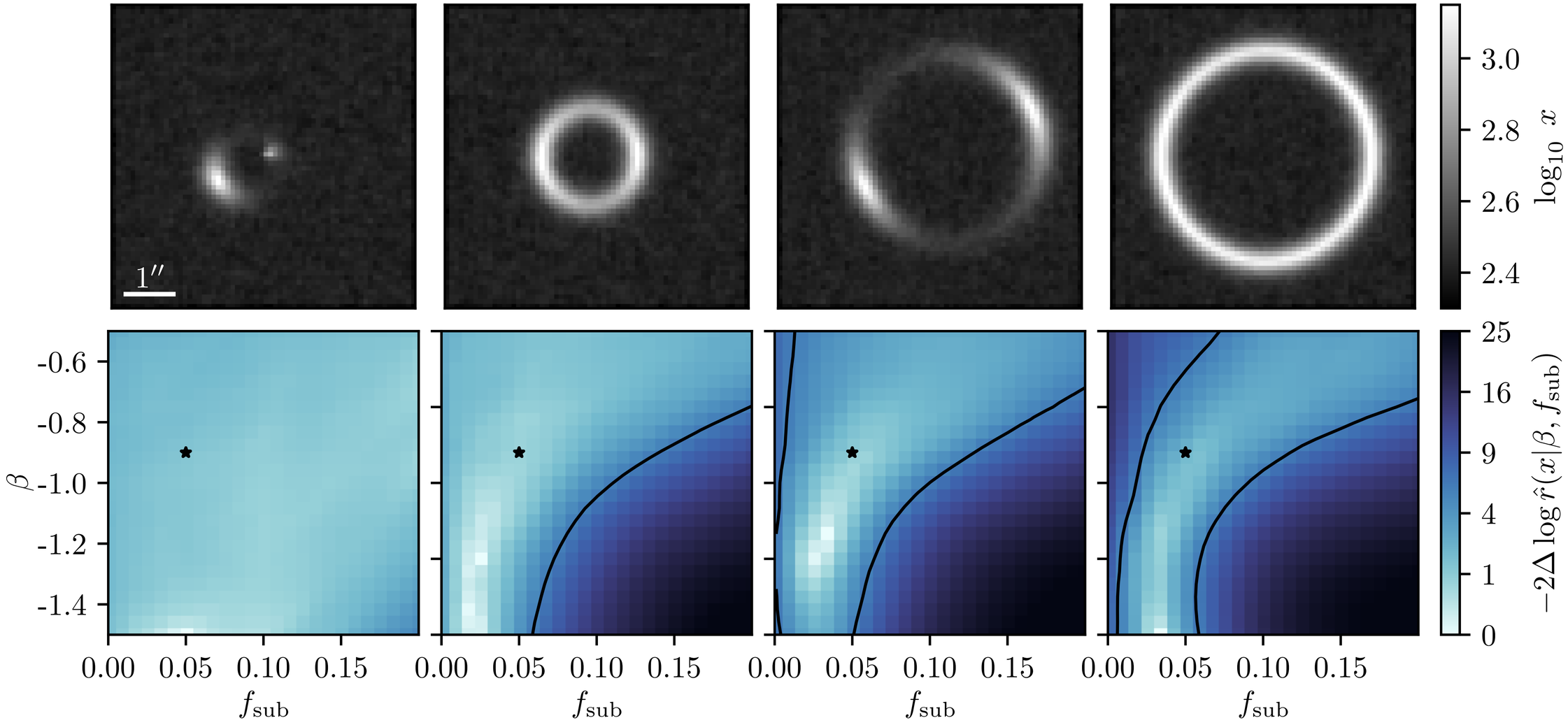
[M. Stoye, JB, J. Pavez, G, Louppe, K. Cranmer 1808.00973]

Probabilistic calibration of network output

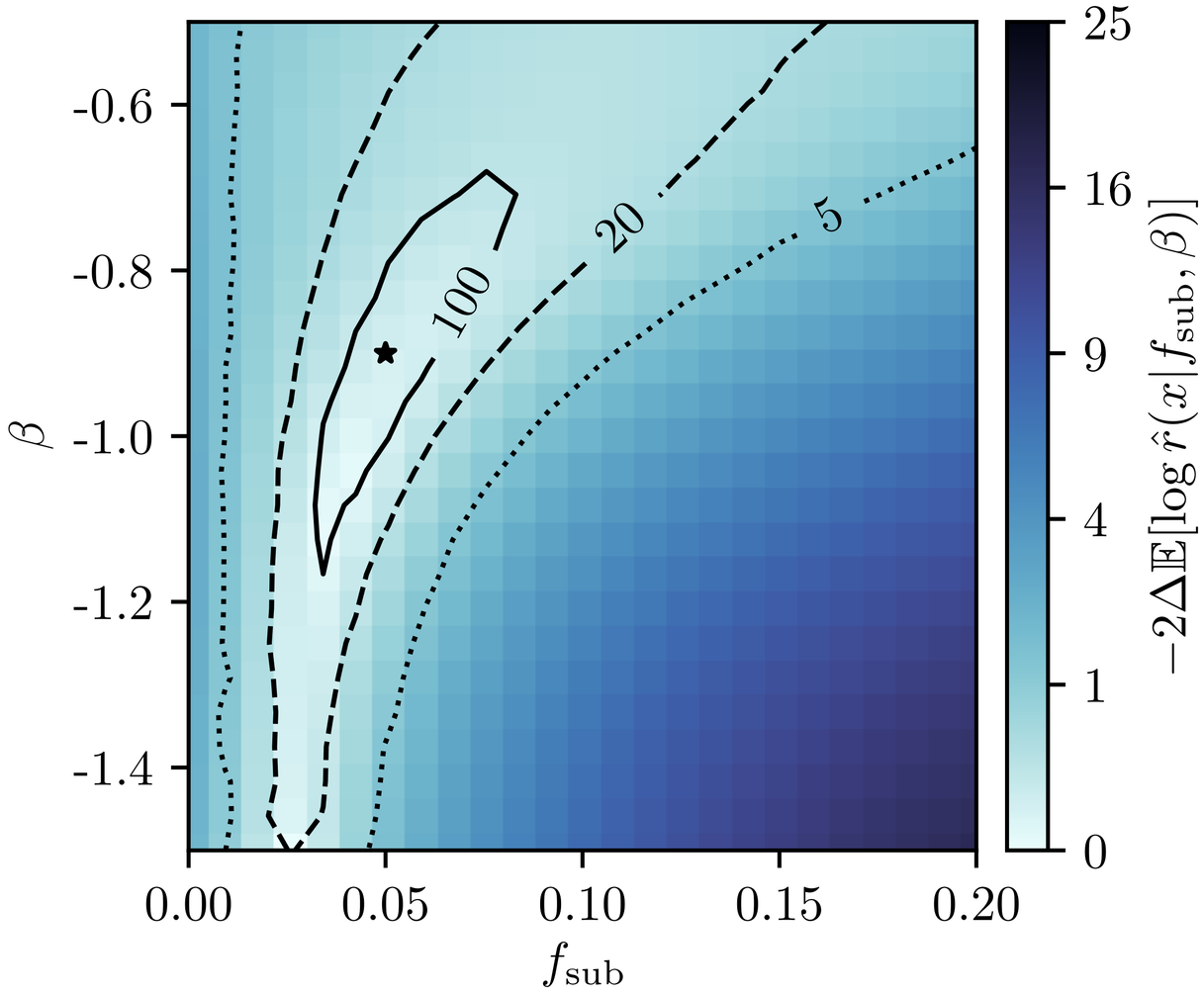
Synthetic "observed" data set simulated for
$f_{\rm sub} = 0.05, \beta = -0.9$

Bayesian & frequentist inference

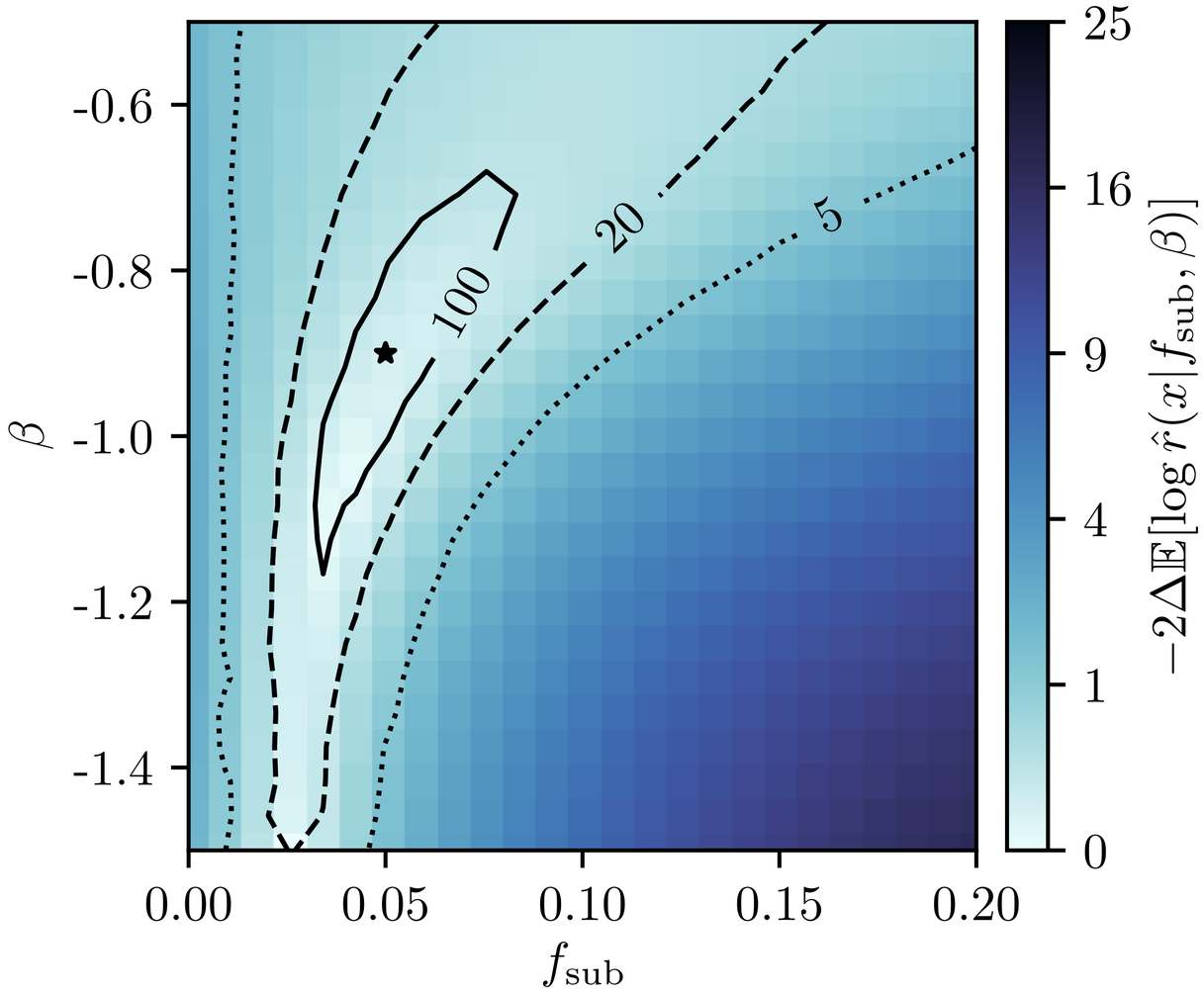
Inferring parameters from individual images

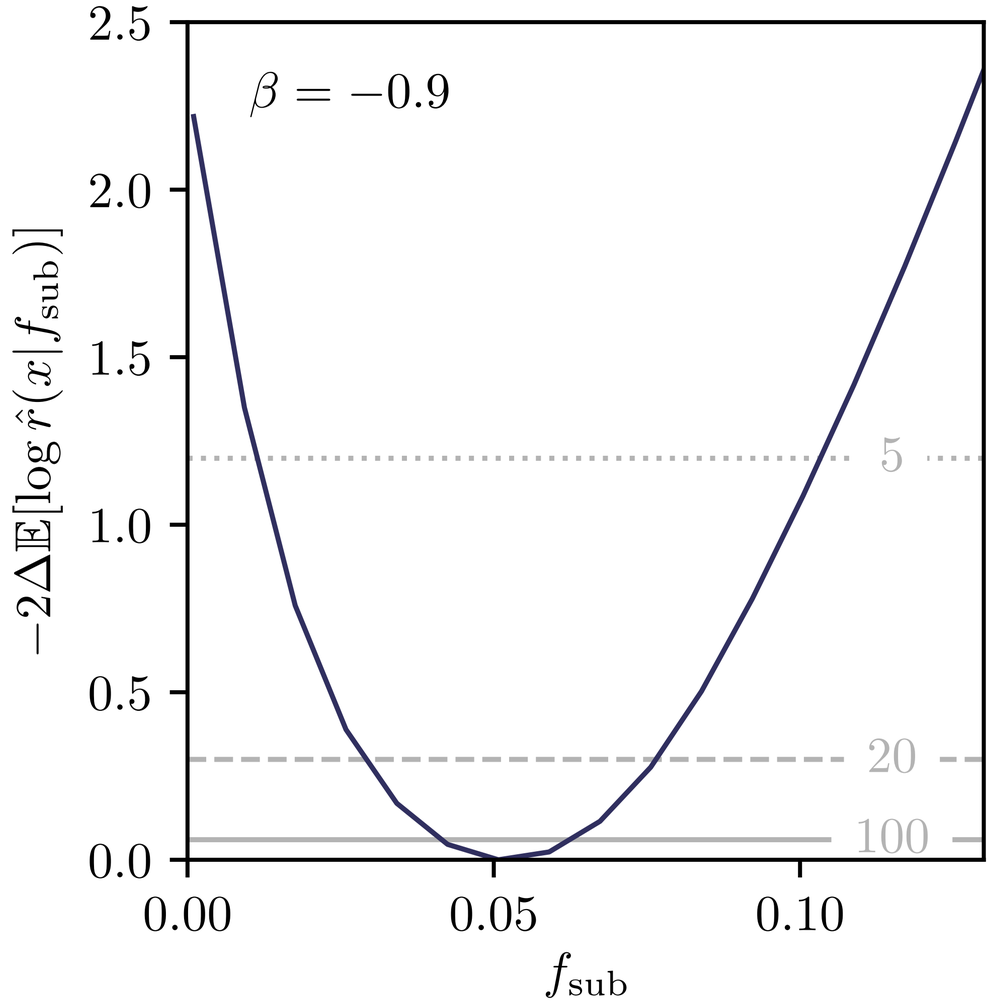


Expected likelihood ratio map

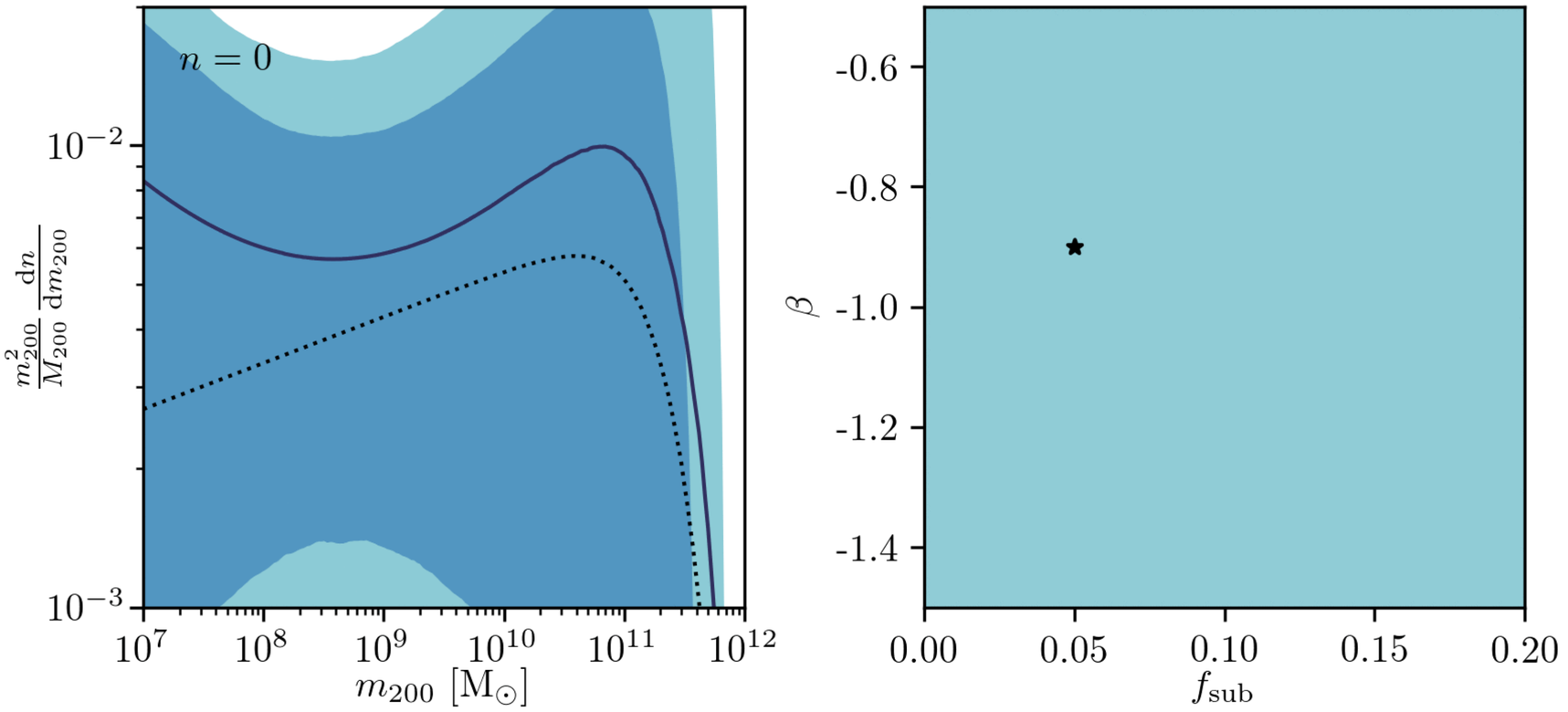


Expected likelihood ratio map

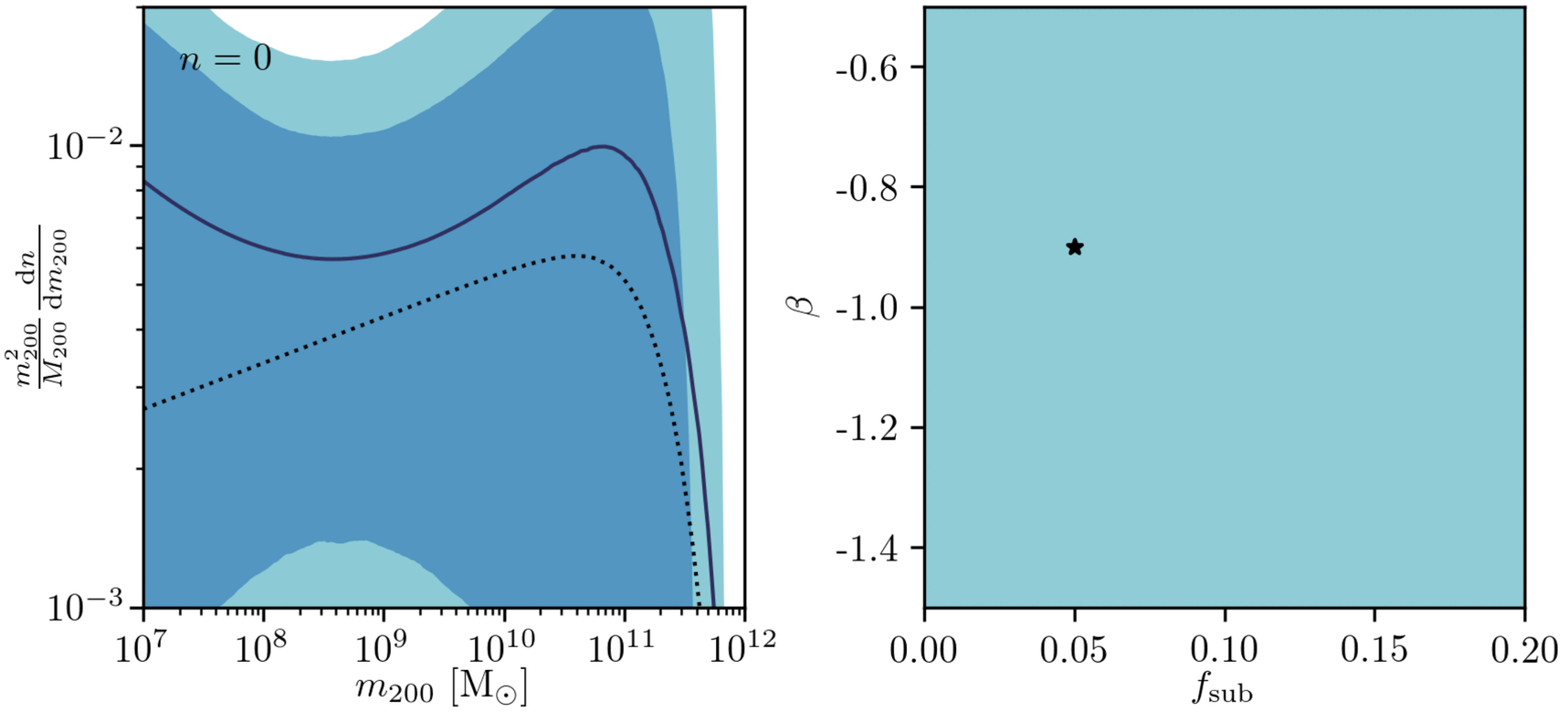


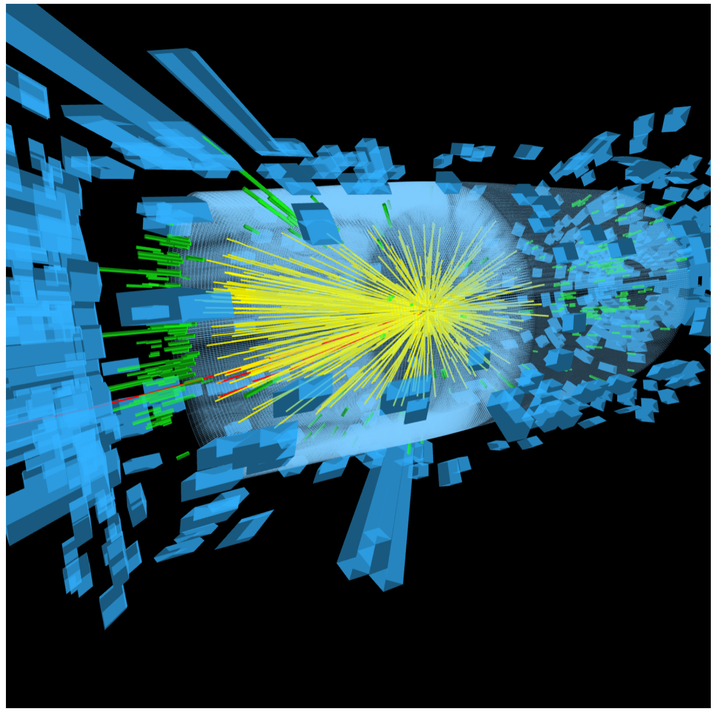


Bayesian inference

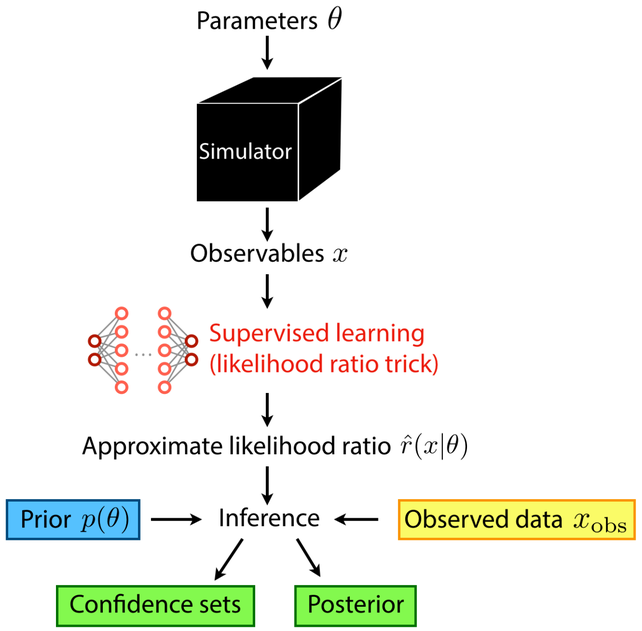


Bayesian inference

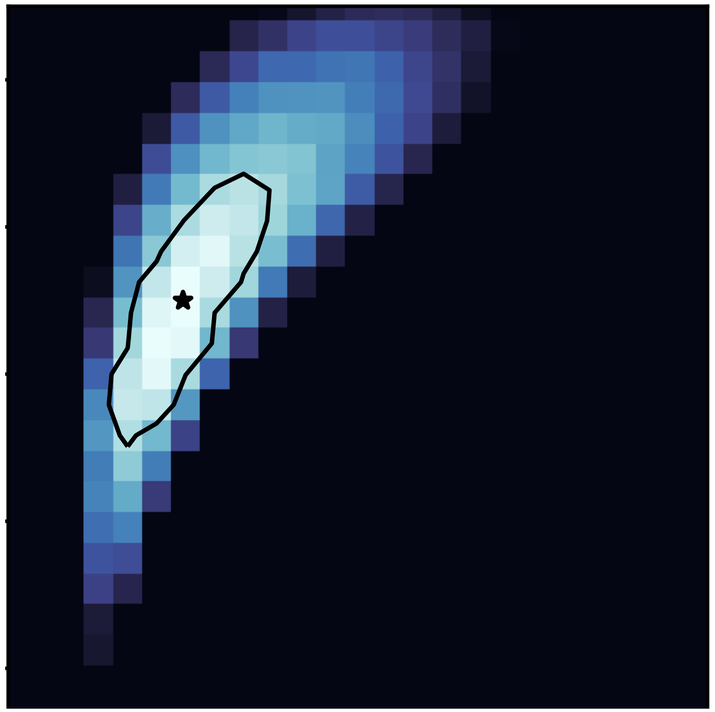




Inference is challenging when and phenomena are modeled with computer simulations and the data are high-dimensional



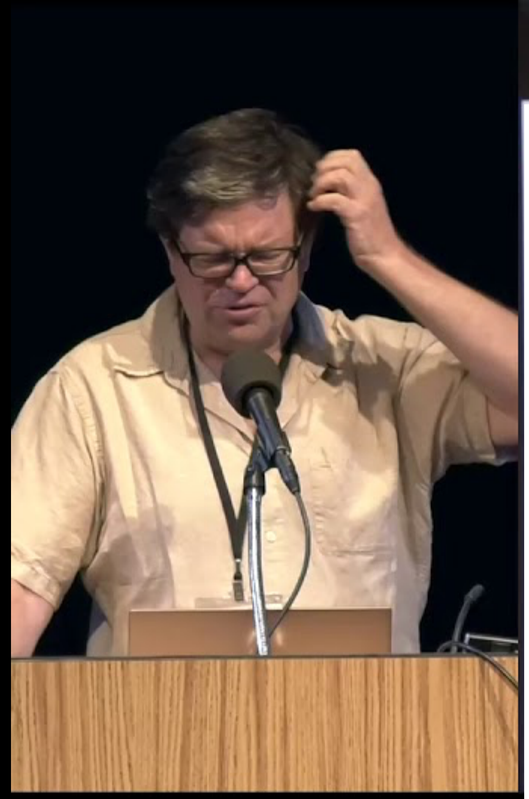
New inference algorithms are based on normalizing flows or the likelihood ratio trick;
we can improve them with physics understanding

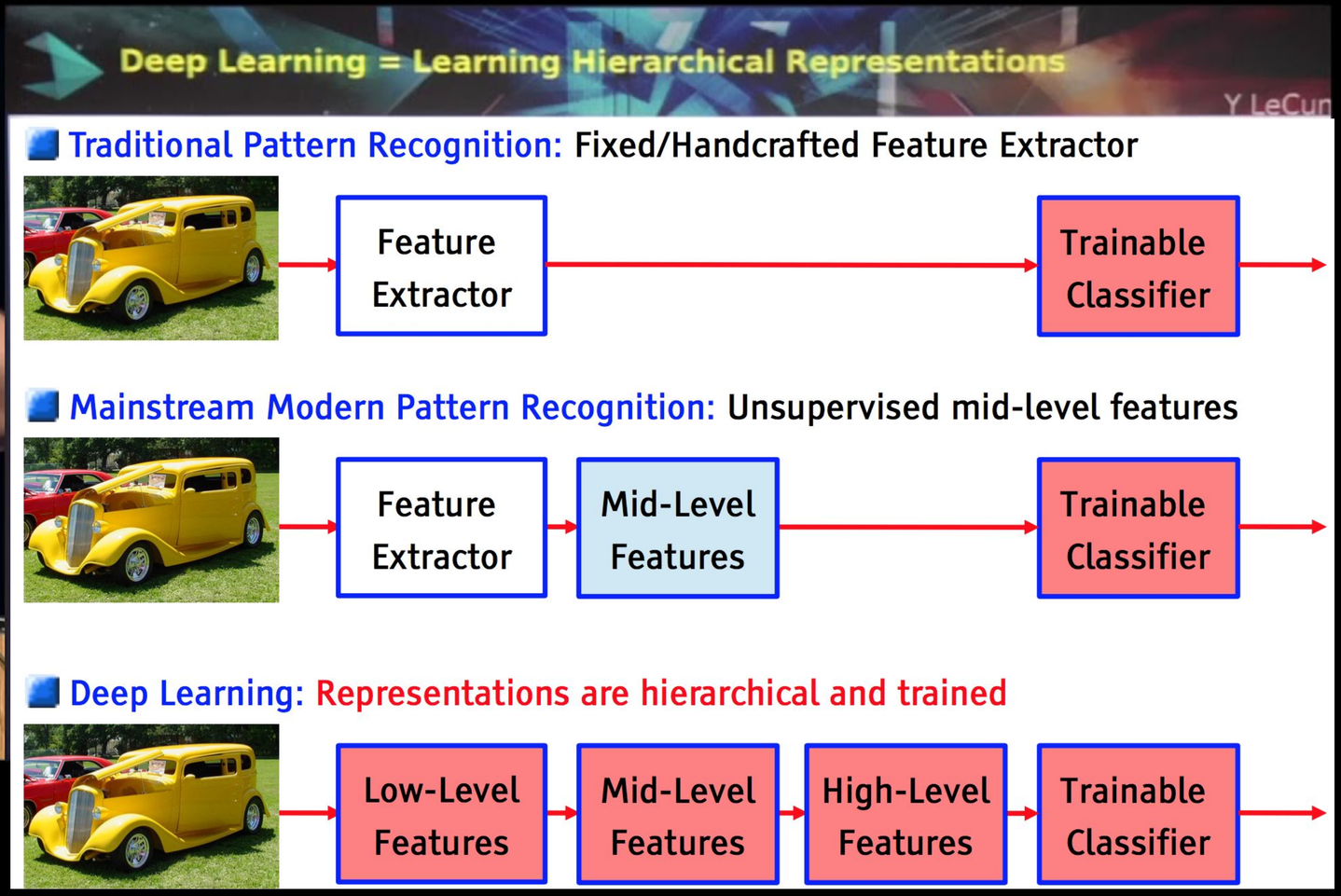


They let us extract more knowledge e.g. from particle physics experiments

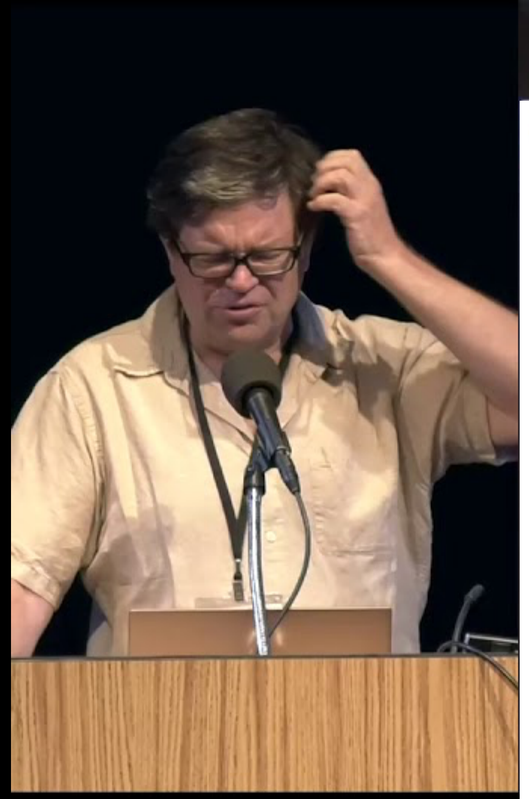
Some thoughts about how to think about machine learning as a physicist

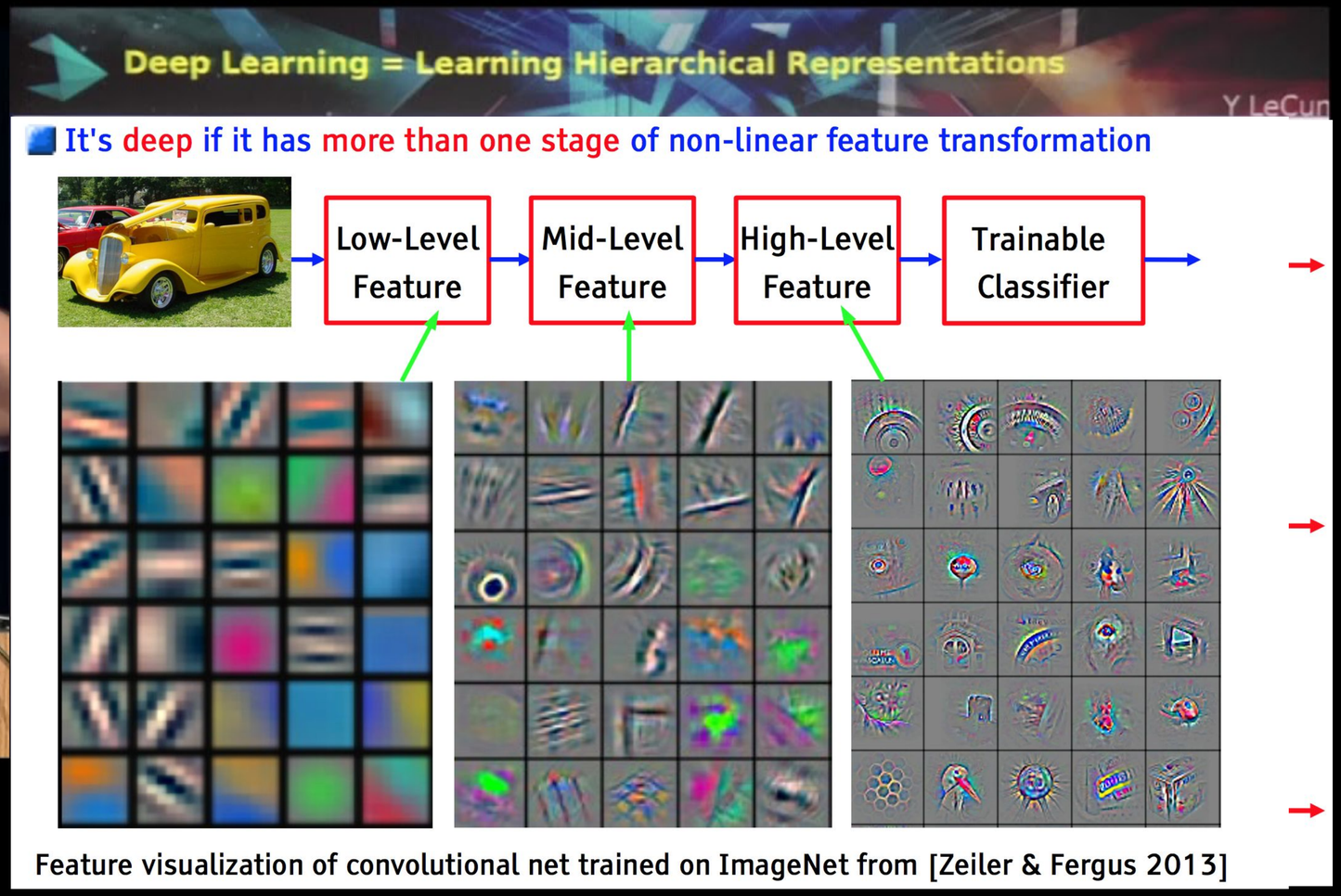
REPRESENTATIONS, EMBEDDDINGS



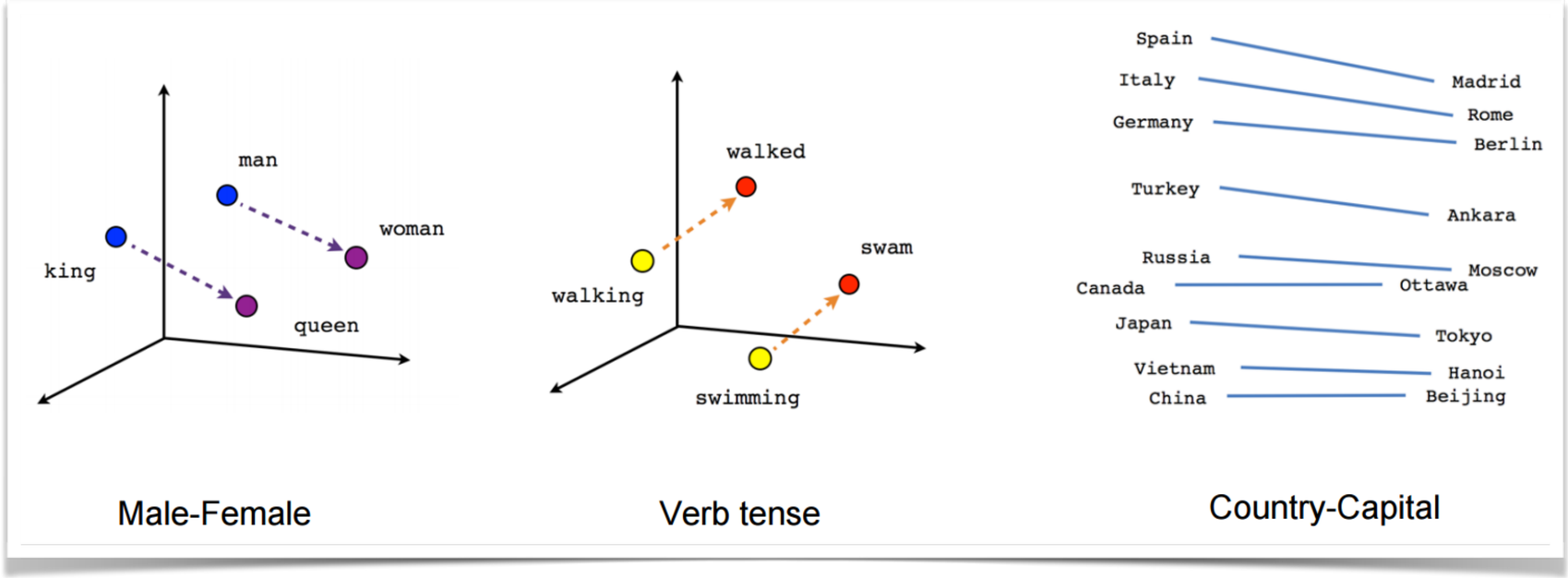


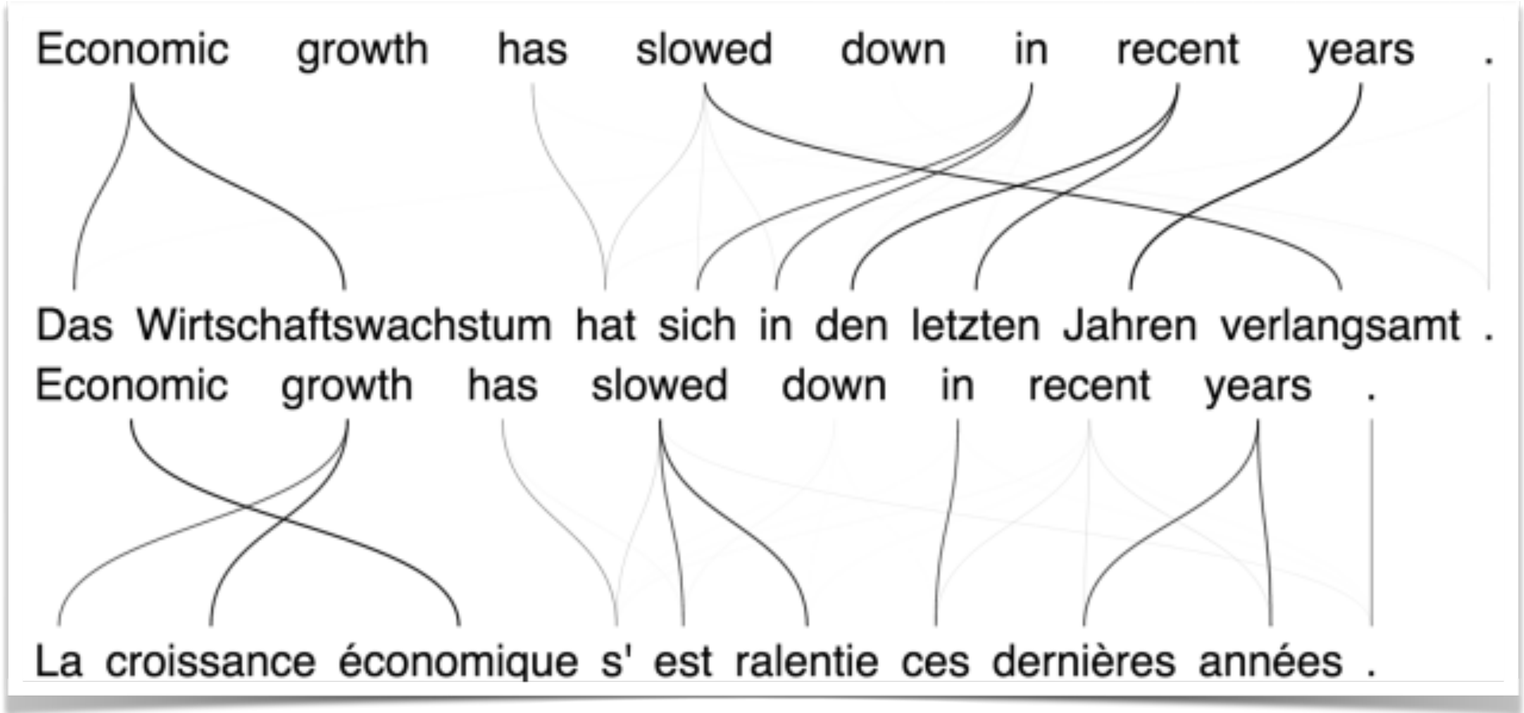
REPRESENTATIONS, EMBEDDDINGS





WORD EMBEDDINGS & TRANSLATION





"LA MIA PARABOLA"

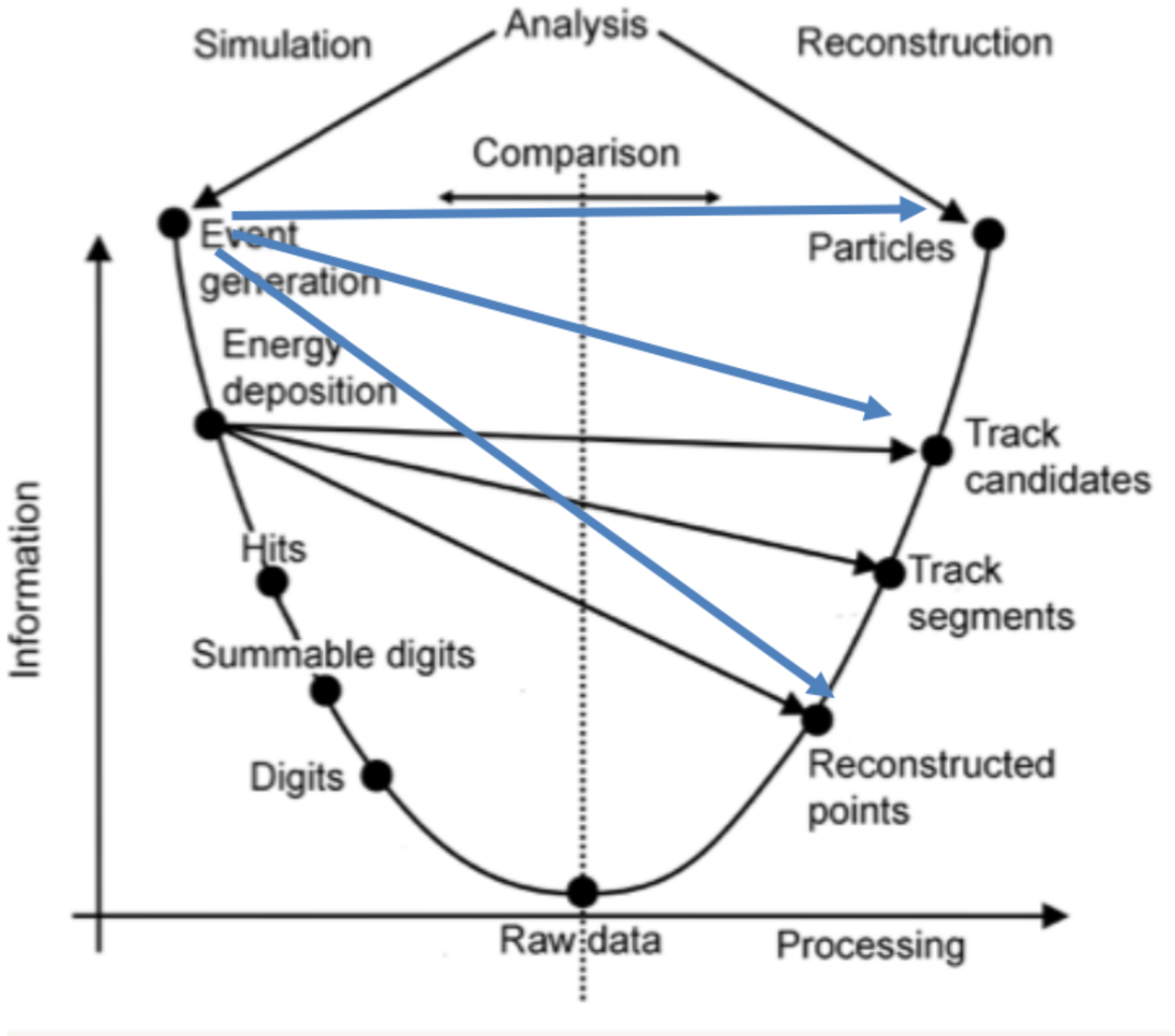
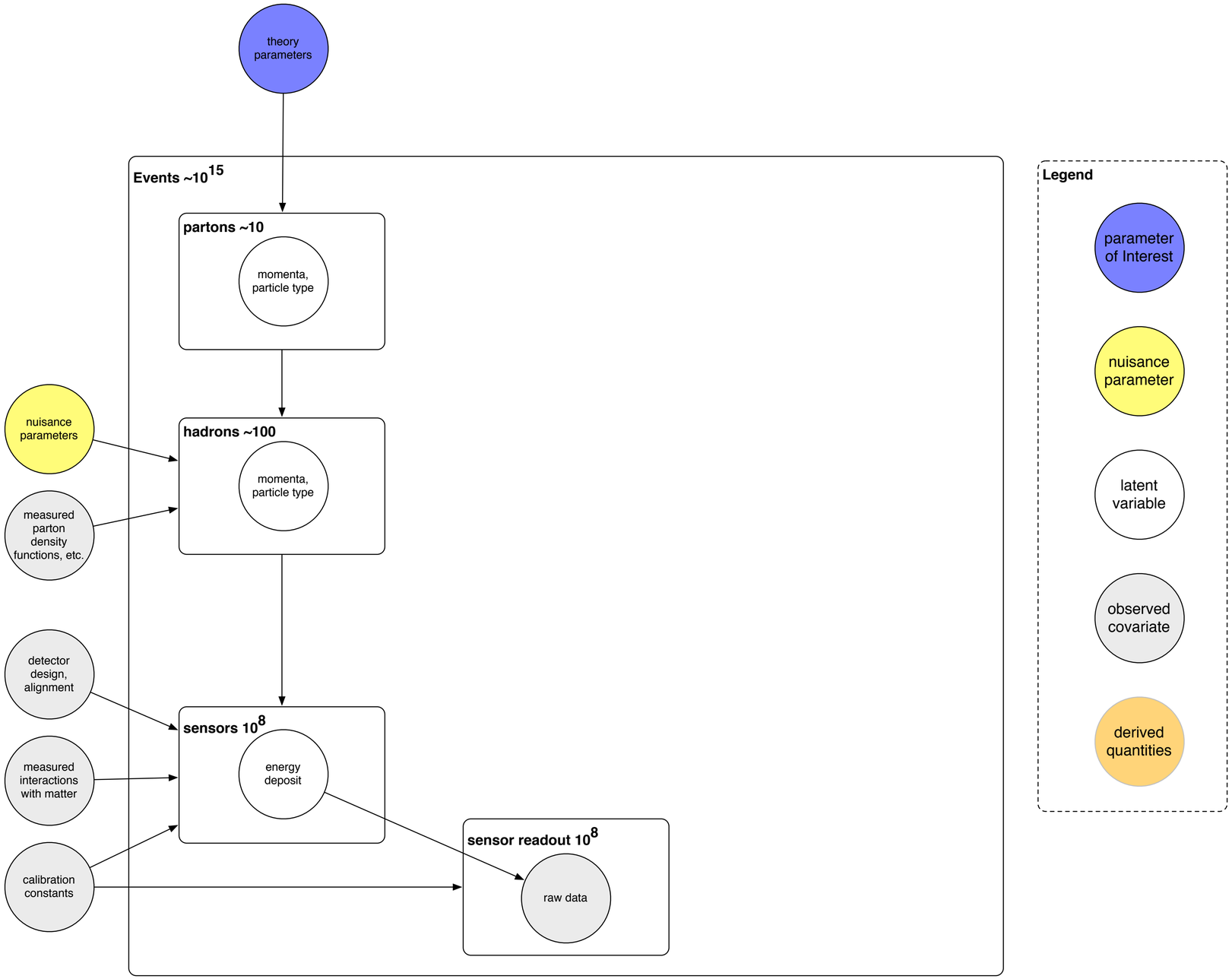
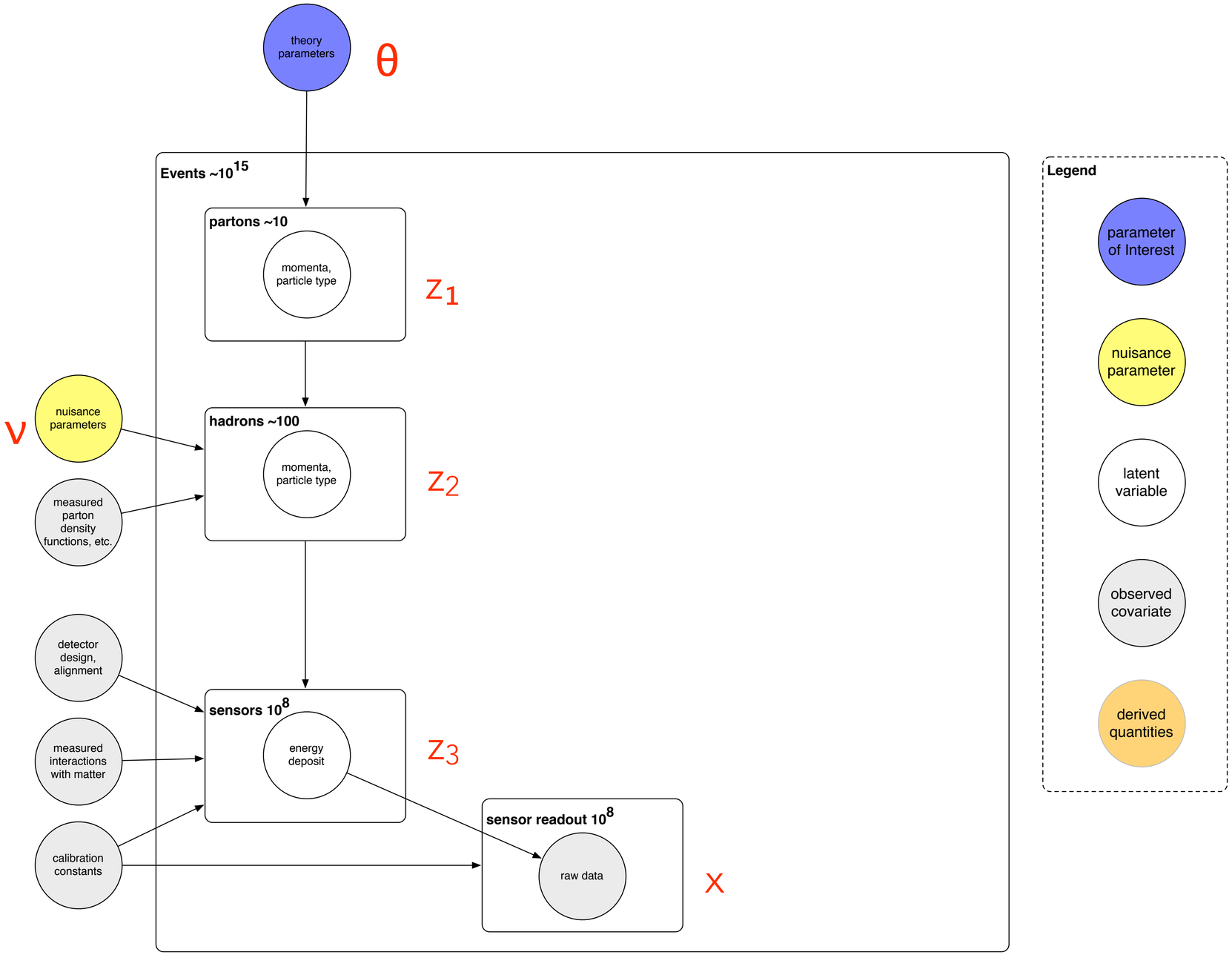


Figure by Federico Carminati, independent parallel inventions by Vincenzo Innocente & K.C.

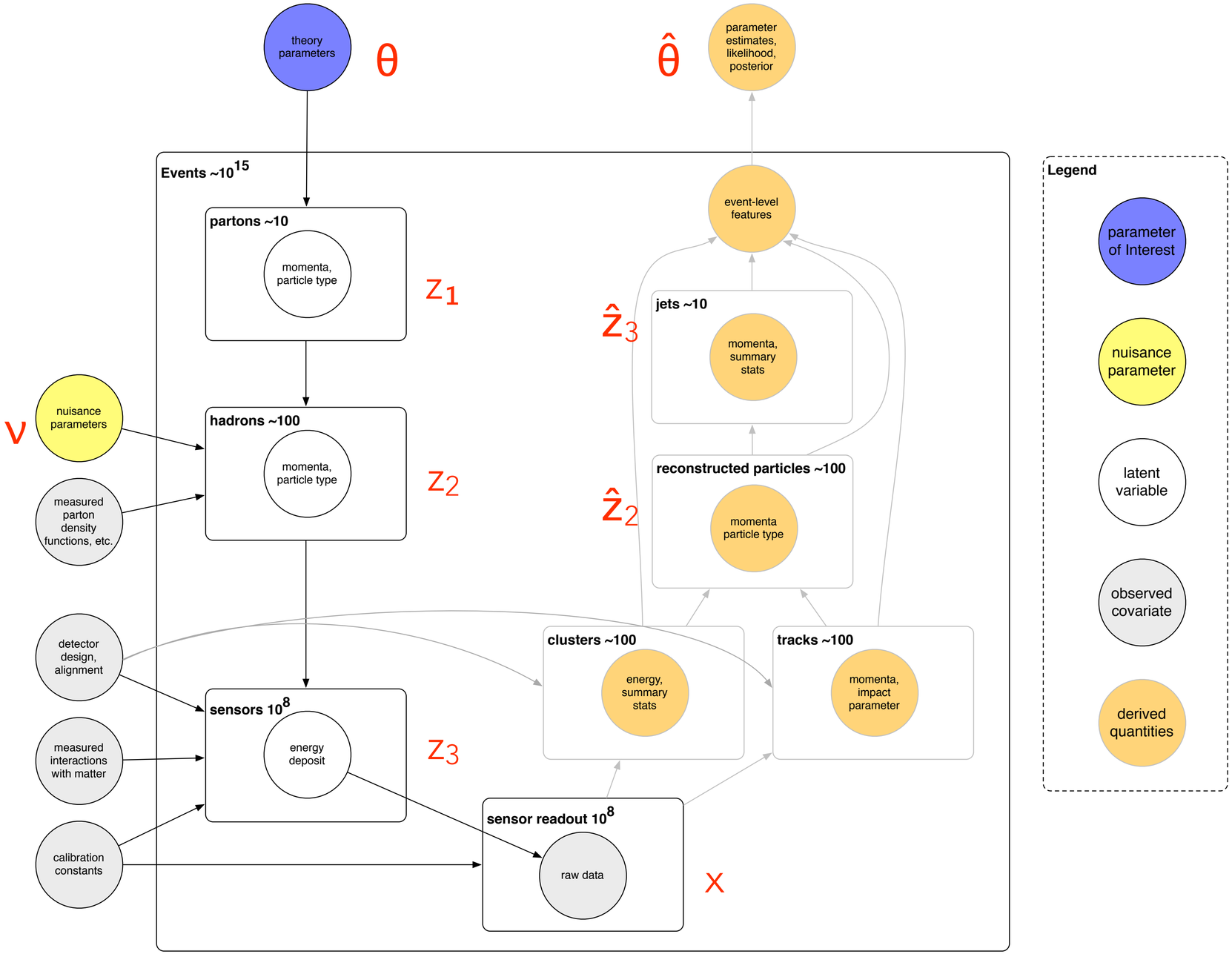
SIMULATION


theory parameters
Events ~10^15
partons ~10
momenta, particle type
nuisance parameters
hadrons ~100
momenta, particle type
measured parton density functions, etc.
detector design, alignment
sensors 10^8
energy deposit
measured interactions with matter
calibration constants
sensor readout 10^8
raw data
Legend
parameter of Interest
nuisance parameter
latent variable
observed covariate
derived quantities

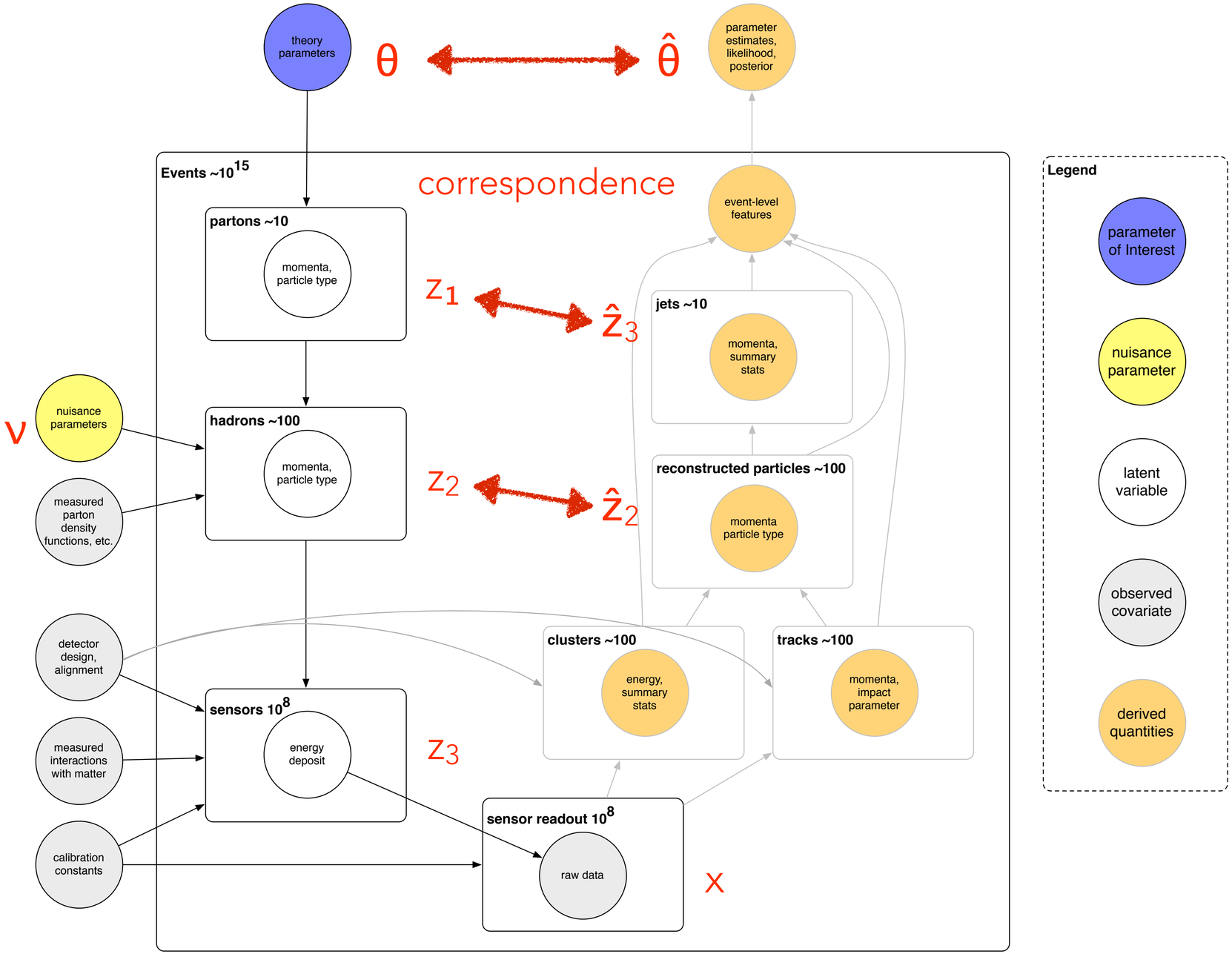
SIMULATION


theory parameters
θ
Events ~10^15
partons ~10
momenta, particle type
z1
ν
nuisance parameters
hadrons ~100
momenta, particle type
z2
measured parton density functions, etc.
detector design, alignment
sensors 10^8
energy deposit
z3
measured interactions with matter
calibration constants
sensor readout 10^8
raw data
x
Legend
parameter of Interest
nuisance parameter
latent variable
observed covariate
derived quantities

SIMULATION + RECONSTRUCTION



SIMULATION + RECONSTRUCTION



COMPOSITION & REDUCTIONISM

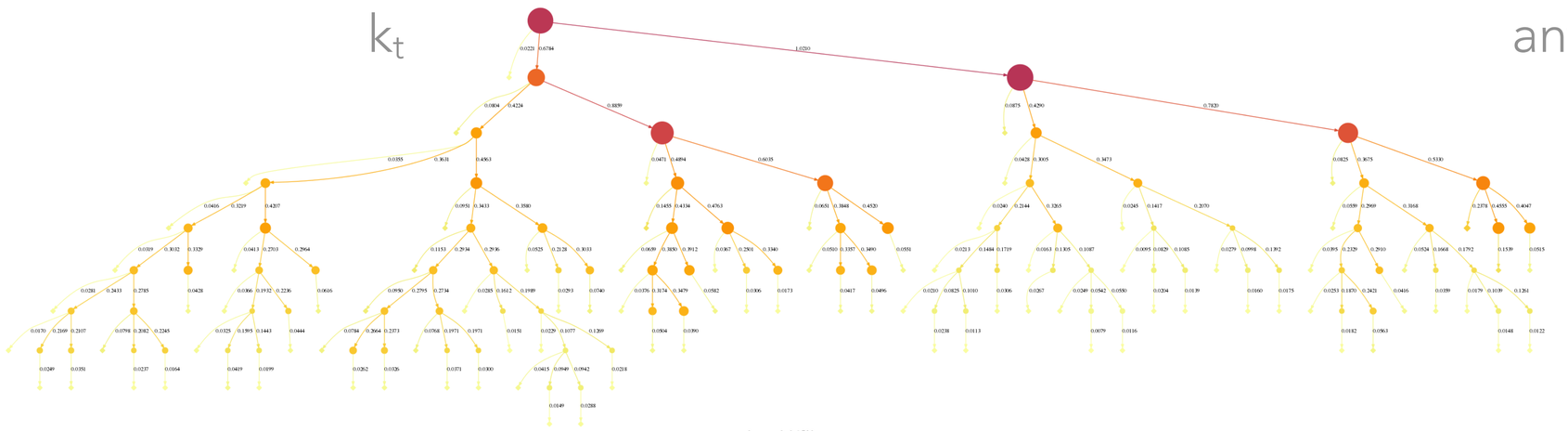
The traditional reconstruction algorithms can be seen as attempt to invert the generative process (point estimate / regression)

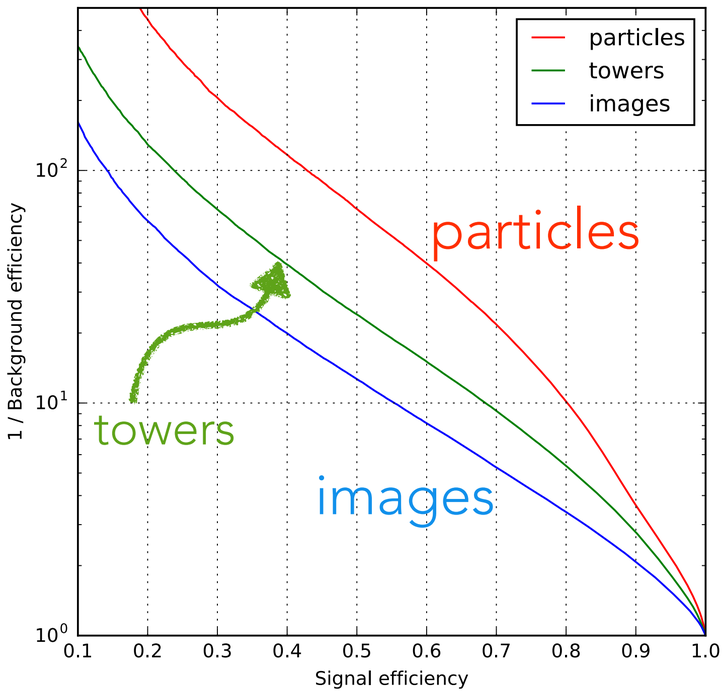
- generative model: $\theta \rightarrow z_1 \rightarrow z_2 \rightarrow z_2 \rightarrow x$
- Sequential Inversion: $x \rightarrow \hat{z}_3(x) \rightarrow \hat{z}_2(\hat{z}_3) \rightarrow \hat{z}_1(\hat{z}_2)$

Key points:

- can **characterize** & **validate** $p(\hat{z}_1 \mid z_1)$, $p(\hat{z}_2 \mid z_2)$, $p(\hat{z}_3 \mid z_3)$ with simulation
- these components are **reusable** (transfer learning)
  - e.g. an algorithm that looks for electrons in the data (segmentation & classification) and estimates their energy and momentum (regression).
- Provides a notion of "**interpretable**" that is practical and actionable
- **Composition** is at the heart of the **reductionist** paradigm of science

QCD-INSPIRED RECURSIVE NEURAL NETWORKS



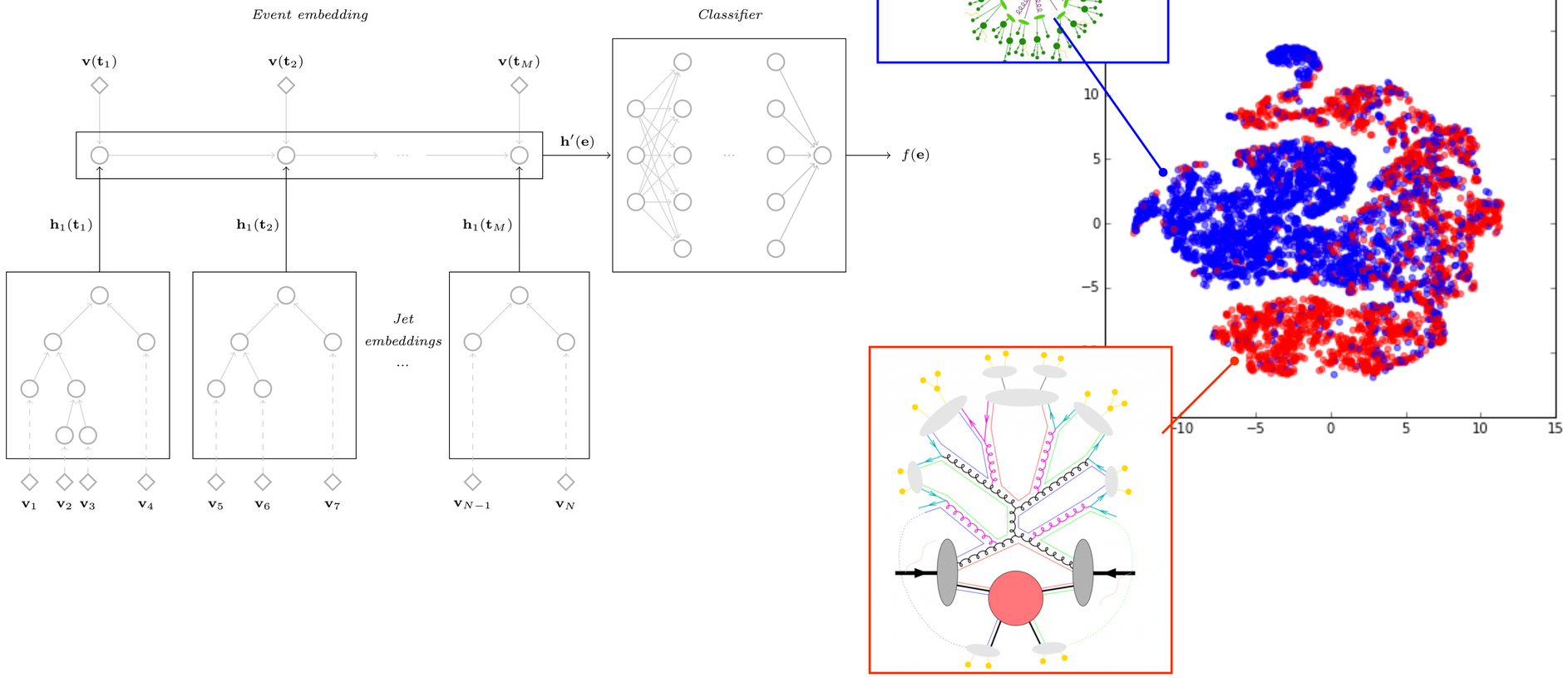


- Generative process is a tree-like, ~stationary Markov Process
- Physics algorithms exist to estimate the tree
- Tree-RNN needs much less data to train!

JOINTLY OPTIMIZE HIERARCHICAL MODEL

particle embedding → jet embedding → event embedding → classifier

It scales!



arXiv:1702.00748

Deep Learning →
Differentiable Functional Programming

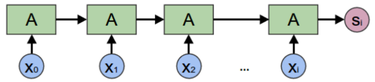
“Representations are Types”

–CHRIS OLAH

READ THIS!

http://colah.github.io/posts/2015-09-NN-Types-FP/

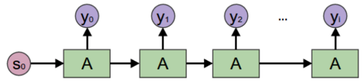
- **Encoding Recurrent Neural Networks** are just folds. They're often used to allow a neural network to take a variable length list as input, for example taking a sentence as input.



fold = Encoding RNN

Haskell: `foldl a s`

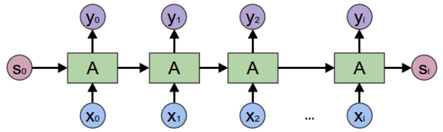
- **Generating Recurrent Neural Networks** are just unfolds. They're often used to allow a neural network to produce a list of outputs, such as words in a sentence.



unfold = Generating RNN

Haskell: `unfoldr a s`

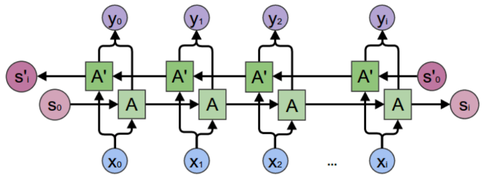
- **General Recurrent Neural Networks** are accumulating maps. They're often used when we're trying to make predictions in a sequence. For example, in voice recognition, we might wish to predict a phenome for every time step in an audio segment, based on past context.



Accumulating Map = RNN

Haskell: `mapAccumR a s`

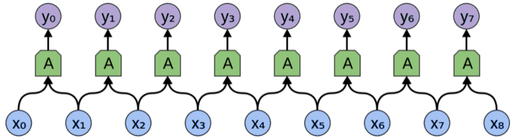
- **Bidirectional Recursive Neural Networks** are a more obscure variant, which I mention primarily for flavor. In functional programming terms, they are a left and a right accumulating map zipped together. They're used to make predictions over a sequence with both past and future context.



Zipped Left & Right Accumulating Map = Bidirectional RNN

Haskell: `zip (mapAccumR a s xs) (mapAccumL a` `s` `xs)`

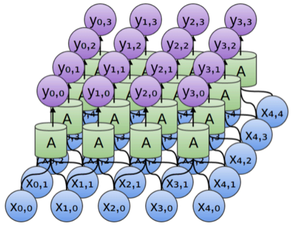
- **Convolutional Neural Networks** are a close relative of map. A normal map applies a function to every element. Convolutional neural networks also look at neighboring elements, applying a function to a small window around every element.[3]



Windowed Map = Convolutional Layer

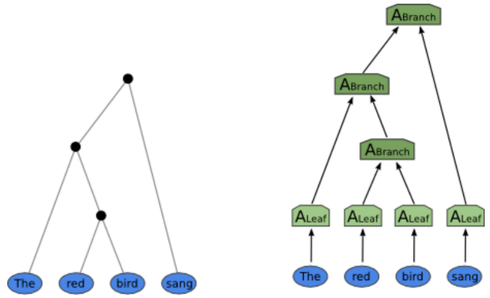
Haskell: `zipWith a xs (tail xs)`

Two dimensional convolutional neural networks are particularly notable. They have been behind recent successes in computer vision. (More on conv nets.)



Two Dimensional Convolutional Network

- **Recursive Neural Networks** ("TreeNets") are catamorphisms, a generalization of folds. They consume a data structure from the bottom up. They're mostly used for natural language processing, to allow neural networks to operate on parse trees.

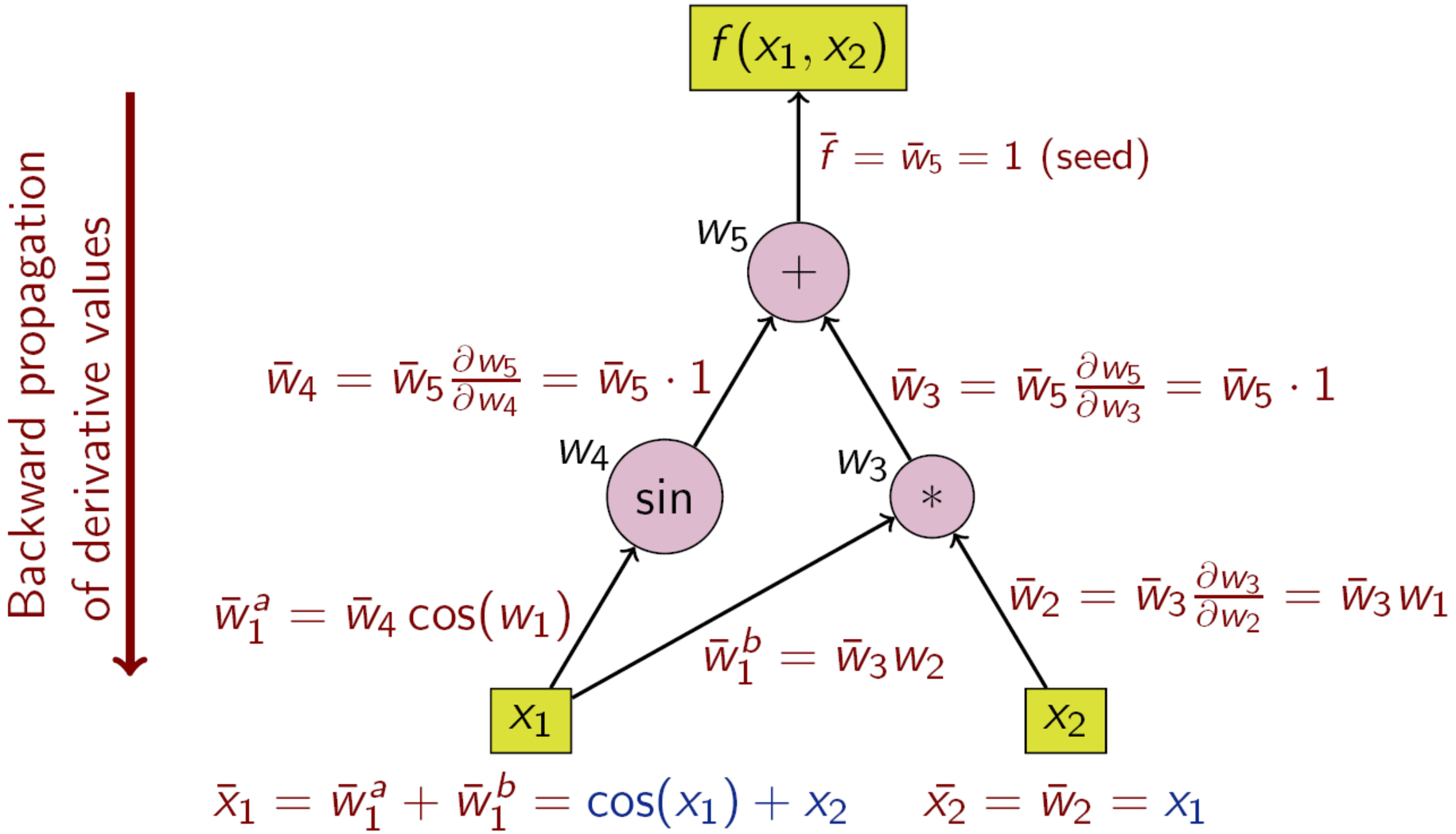


Catamorphism = TreeNet

Haskell: `cata a`

AUTOMATIC DIFFERENTIATION

2. **reverse accumulation** computes the recursive relation: $\frac{dy}{dw_i} = \frac{dy}{dw_{i+1}} \frac{dw_{i+1}}{dw_i}$ with $w_0 = x$.



DIFFERENTIABLE REDUCTIONISM

The reconstruction algorithms can be seen as attempt to invert the generative process (point estimate / regression) sequentially

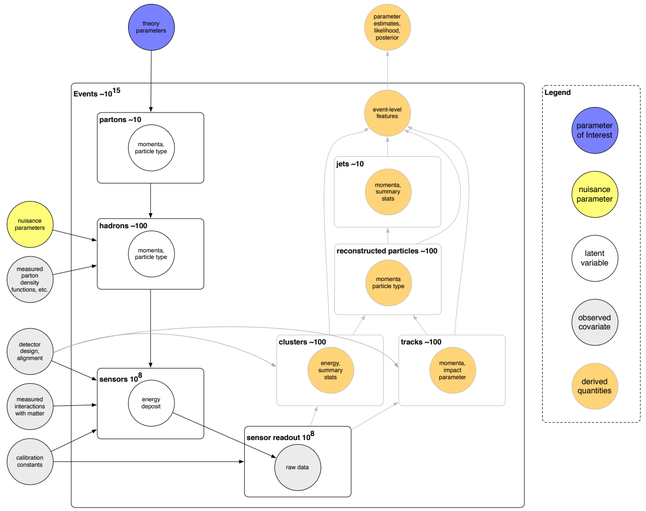
- generative model: $\theta \rightarrow z_1 \rightarrow z_2 \rightarrow z_2 \rightarrow x$
- Sequential Inversion: $x \rightarrow \hat{z}_3(x) \rightarrow \hat{z}_2(\hat{z}_3) \rightarrow \hat{z}_1(\hat{z}_2)$

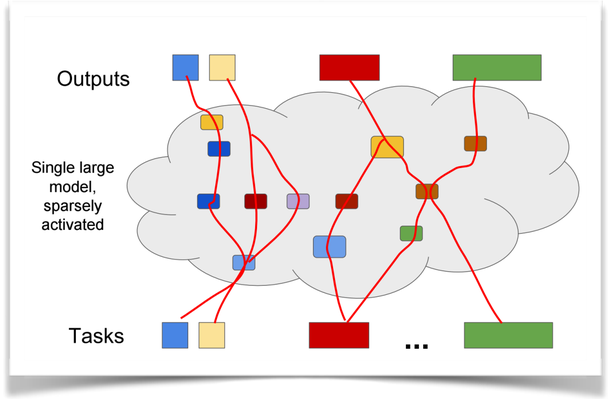
Currently both generative model and inversion algorithms involve hand-engineered, code not developed for auto-diff / back propagation (effectively not differentiable)

- big gain from just reimplementing what we have in a **Differentiable Programming framework**

We can keep the compositional structure and gradually enhance each of the stages of the with deep learning components

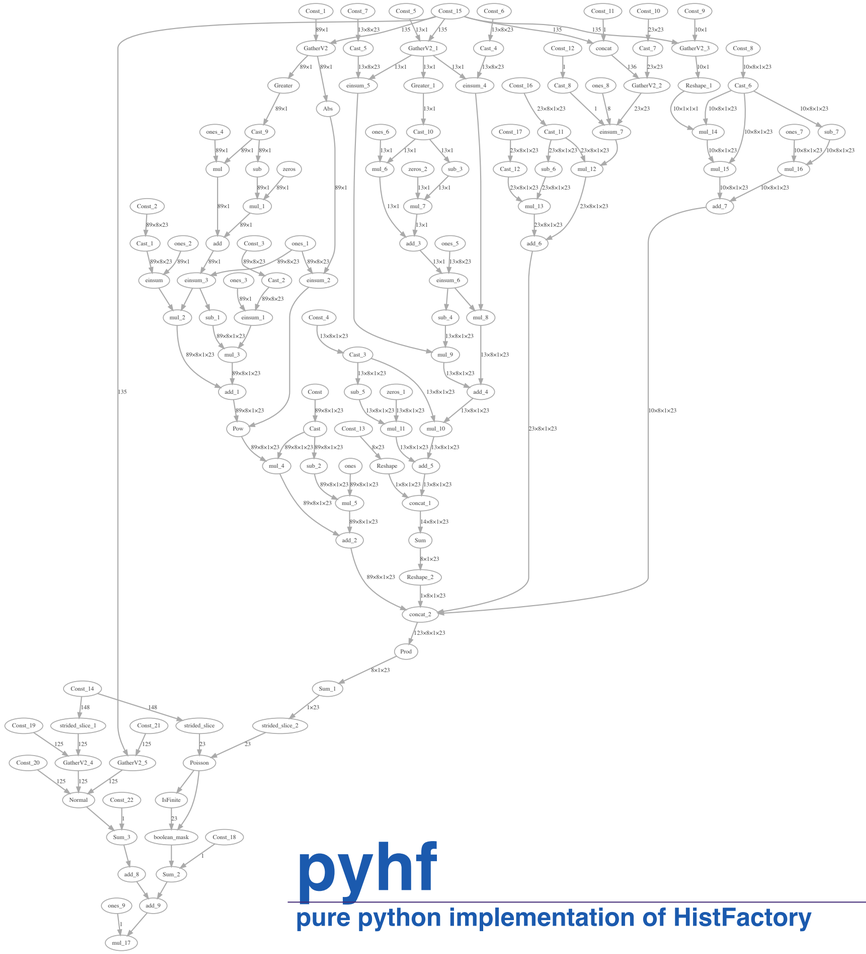
- A high-level form of inductive bias (innate structure) on the networks
- jointly optimize & borrow power from all the tasks that use a certain component
  - maintain ability to characterize, validate , and interpret individual components
- transition from deterministic point estimate to probabilistic components for improved uncertainty estimation





pyhf: auto-differentiable binned HEP likelihoods

Kyle Cranmer (NYU) Matthew Feickert (SMU)
Lukas Heinrich (CERN), Giordon Stark (UCSC)



pyhf

pure python implementation of HistFactory

implementation of HistFactory likelihood (1) as a computational graph of multi-dimensional array operations.

Use of array ("tensor") operations through a common API layer around high-performance tensor libraries: e.g.

NumPy TensorFlow PyTorch

Installation:

```
$> pip install pyhf
```

Example: simple number-counting experiment

```
import pyhf
pdf = pyhf.simplemodels.hepdata_like(
  signal_data=[12.0, 11.0], bkg_data=[50.0, 52.0], bkg_uncerts=[3.0, 7.0]
)
CLs_obs, CLs_exp = pyhf.utils.hypotest(1.0, [51, 48] + pdf.config.auxdata, pdf, return_expected=True)

print('Observed: {}, Expected: {}'.format(CLs_obs, CLs_exp))
Observed: [0.05290116], Expected: [0.06445521]
```

Auto-Differentiation:

Tensor libraries from ML community provide exact gradients for use in minimization. $\frac{\partial \mathcal{L}}{\partial \mu}, \frac{\partial \mathcal{L}}{\partial \theta_i}$

Optimizers

pyhf likeliehood are simple tensor-value python functions. Can use multiple minimization algorithms, such as `scipy.minimize` or MINUIT

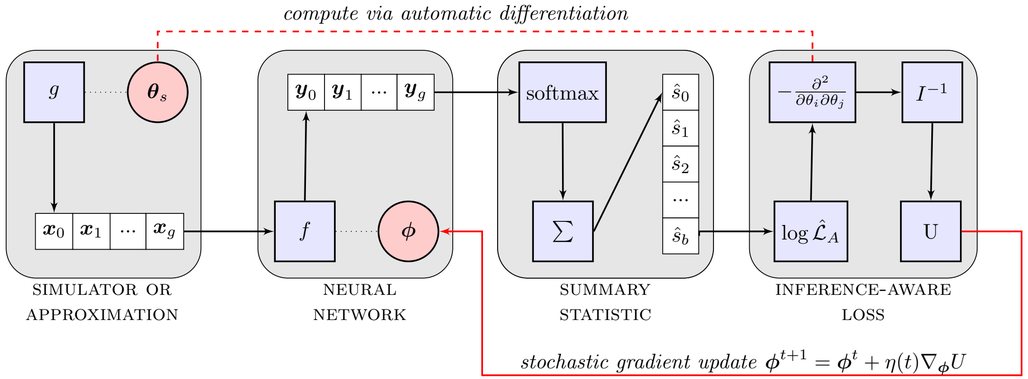
End-to-end optimization with autodial

- With tools like MadMiner the objective is to learn a likelihood ratio, which is known optimal properties for measurements etc.
- In INFERNO and Neo the inference objective is directly optimized

INFERNO: Inference-Aware Neural Optimisation

Pablo de Castro
INFN - Sezione di Padova
pablo.de.castro@cern.ch

Tommaso Dorigo
INFN - Sezione di Padova
tommaso.dorigo@cern.ch



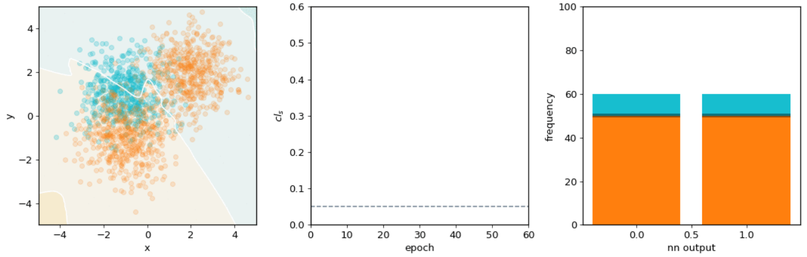
Kyle Cranmer @KyleCranmer · 19h
Take note! Here is a nice example of differentiable programming. It shows end-to-end optimization of a NN for event categorization wrt. final statistical analysis (using pyhf). Requires running gradients through results of maximum likelihood with fixed-point differentiation 👏

Nathan Simpson @ CERN
@phi_nate

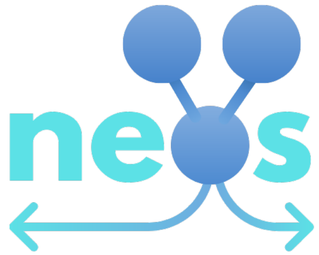
I'm *very* excited to share with you what I've been working on recently in collaboration with @lukasheinrich_ !

We've developed a module that performs end-to-end learning with respect to statistical inference in particle physics.

try it yourself at github.com/pyhf/neos! :)



10:58 AM · Mar 5, 2020 · Twitter Web App



https://github.com/pyhf/neos

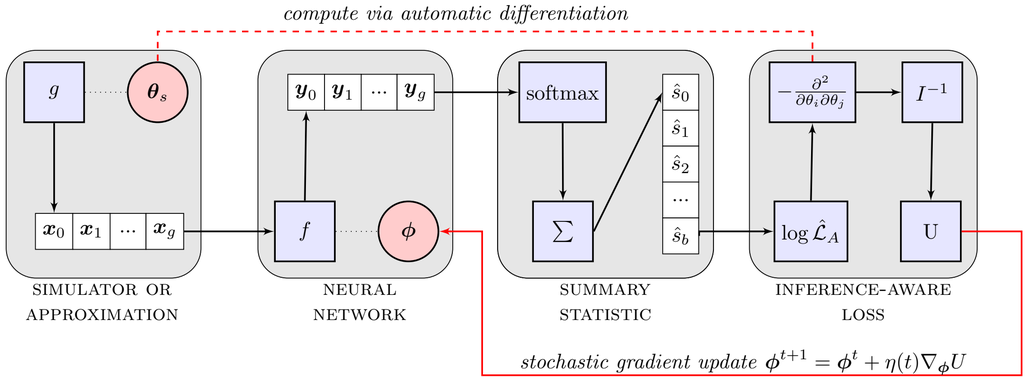
End-to-end optimization with autodial

- With tools like MadMiner the objective is to learn a likelihood ratio, which is known optimal properties for measurements etc.
- In INFERNO and Neo the inference objective is directly optimized

INFERNO: Inference-Aware Neural Optimisation

Pablo de Castro
INFN - Sezione di Padova
pablo.de.castro@cern.ch

Tommaso Dorigo
INFN - Sezione di Padova
tommaso.dorigo@cern.ch



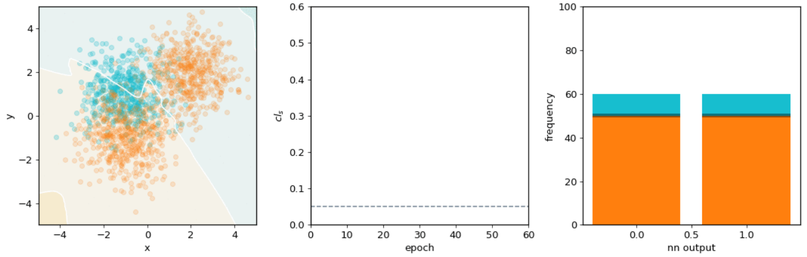
Kyle Cranmer @KyleCranmer · 19h
Take note! Here is a nice example of differentiable programming. It shows end-to-end optimization of a NN for event categorization wrt. final statistical analysis (using pyhf). Requires running gradients through results of maximum likelihood with fixed-point differentiation 👏

Nathan Simpson @ CERN
@phi_nate

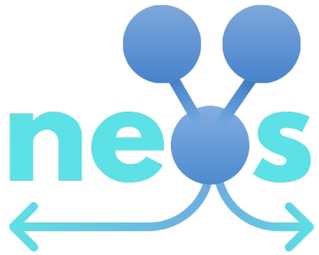
I'm *very* excited to share with you what I've been working on recently in collaboration with @lukasheinrich_ !

We've developed a module that performs end-to-end learning with respect to statistical inference in particle physics.

try it yourself at github.com/pyhf/neos! :)



10:58 AM · Mar 5, 2020 · Twitter Web App



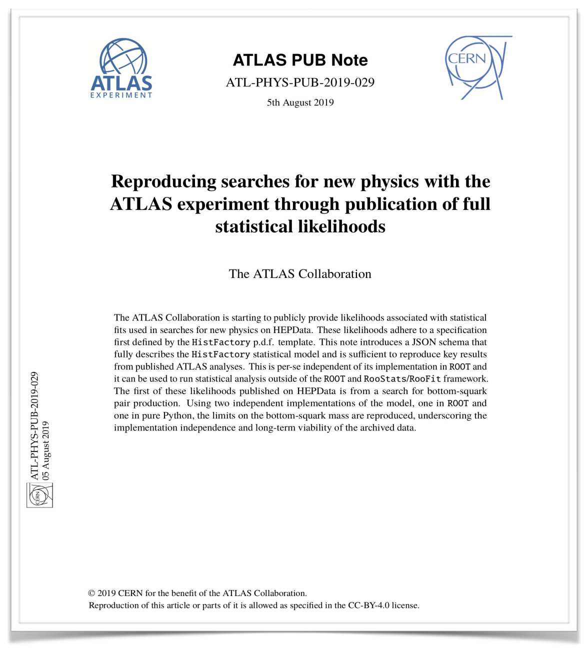
https://github.com/pyhf/neos

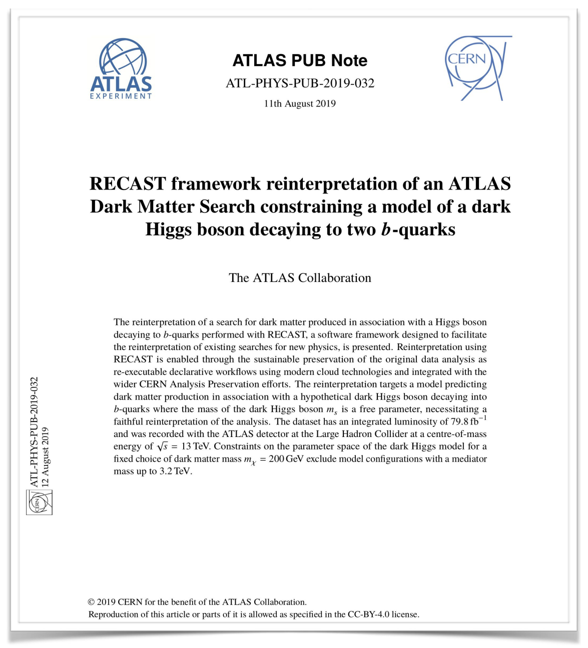
pyhf and publishing

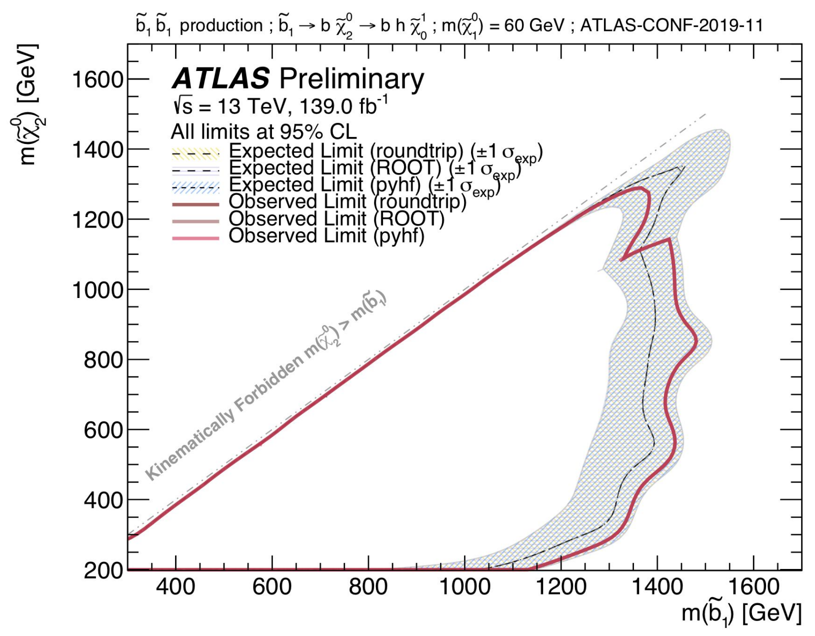
Highlight

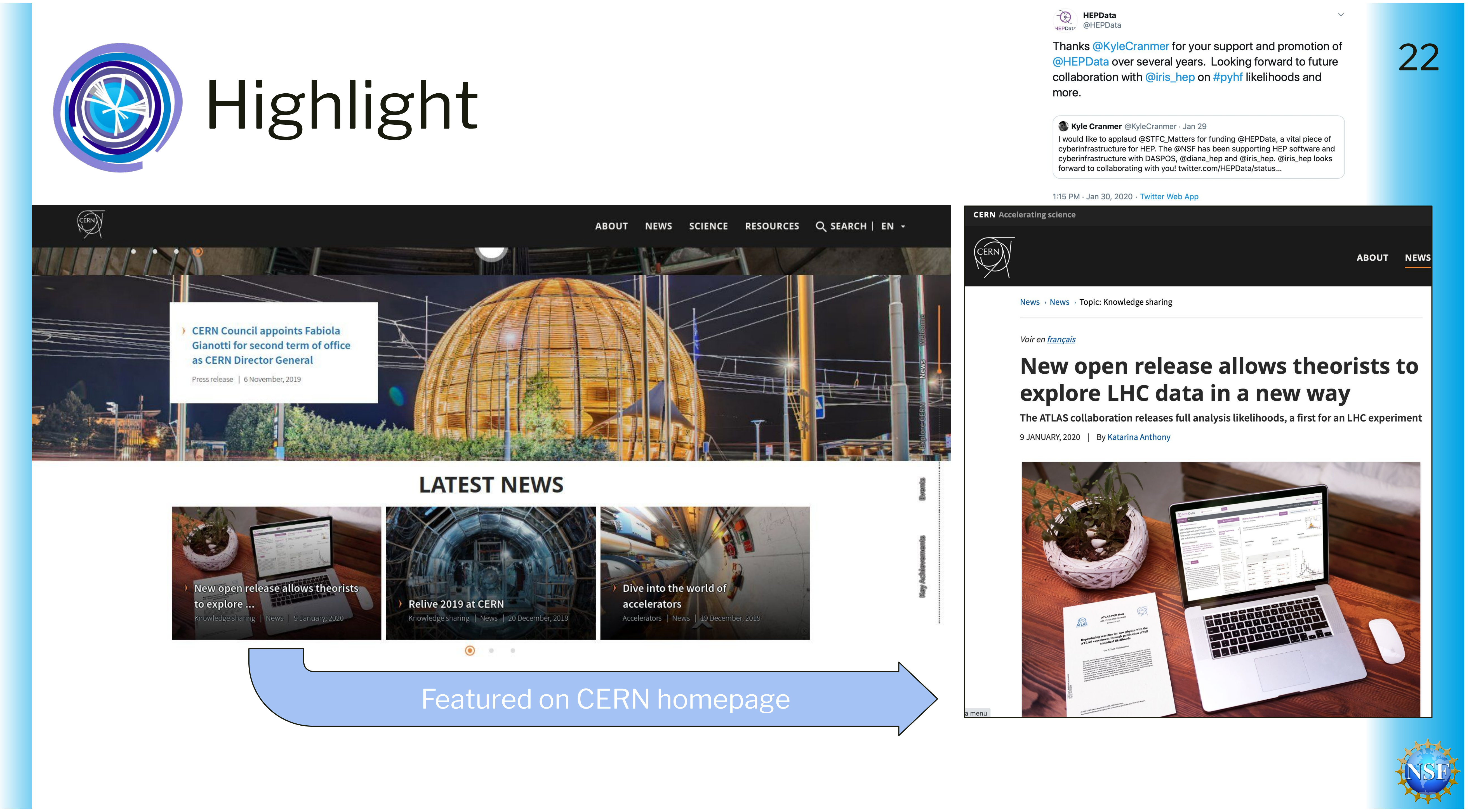
- The field is at a tipping point, DIANA/DASPOS/IRIS-HEP contributions have been transformational.
- First results using the RECAST reinterpretation framework and publishing full statistical likelihoods (using pyhf)

ROOT: 10+ hours
pyhf: < 30 minutes









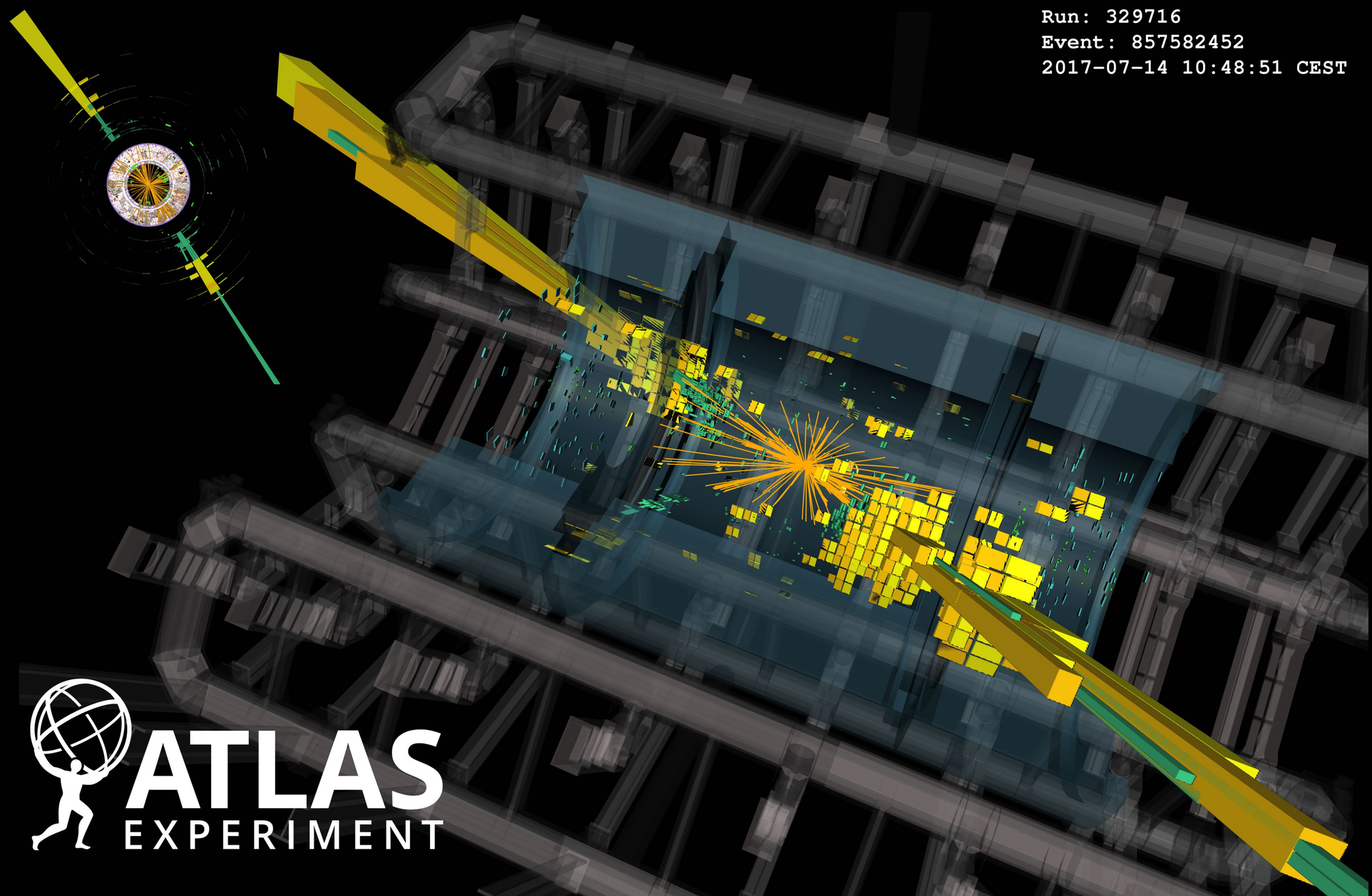
Inductive Bias

Compositionality

Symmetry

Causality

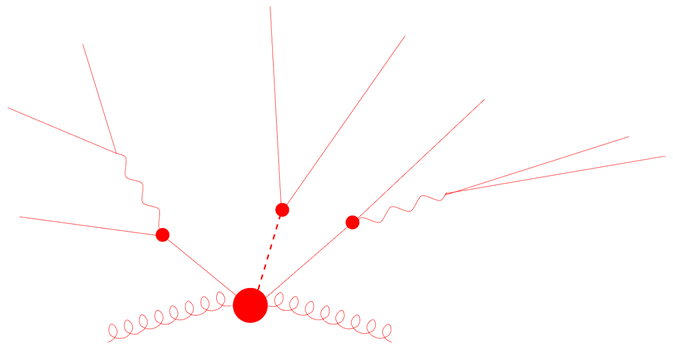
JETS



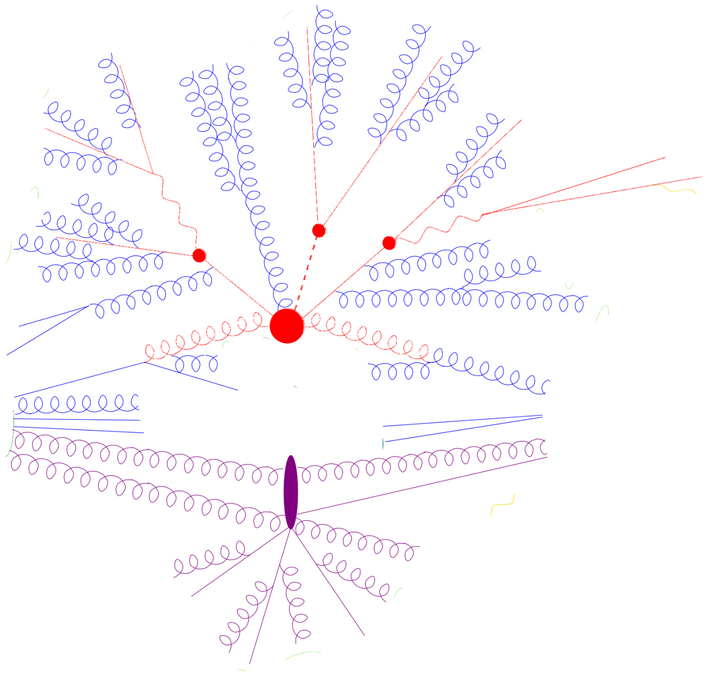
ATLAS
EXPERIMENT

CAUSAL, GENERATIVE MODEL FOR JETS

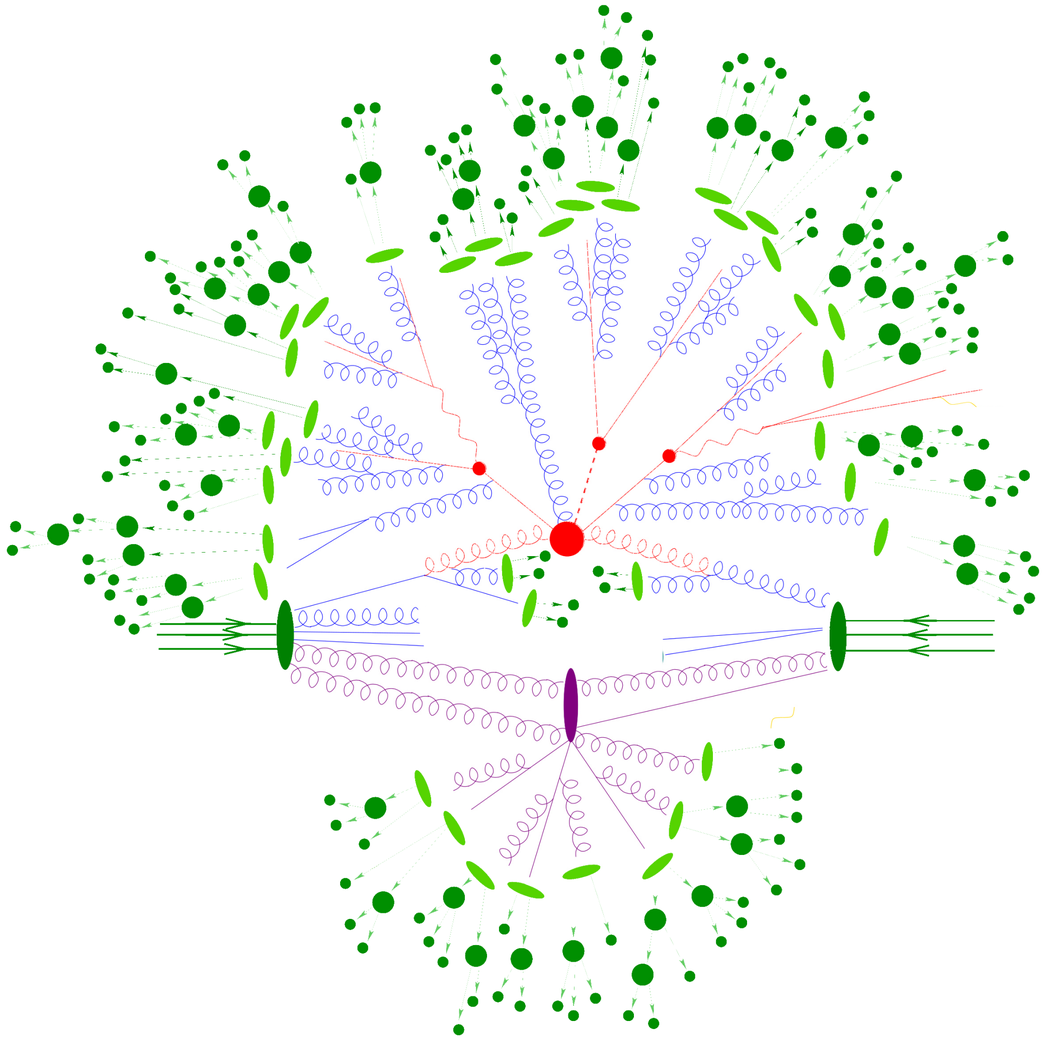
CAUSAL, GENERATIVE MODEL FOR JETS



CAUSAL, GENERATIVE MODEL FOR JETS

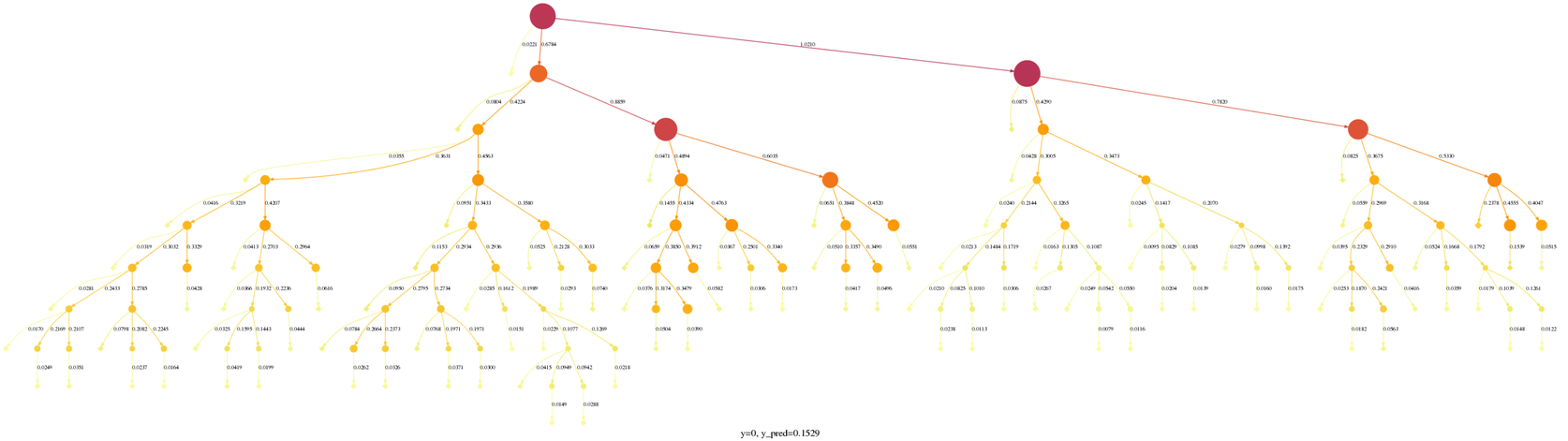


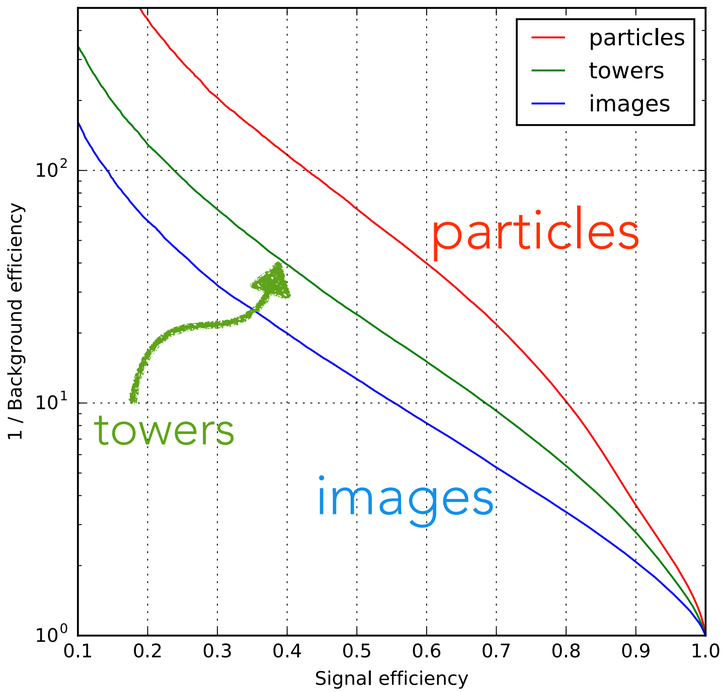
CAUSAL, GENERATIVE MODEL FOR JETS



QCD-INSPIRED RECURSIVE NEURAL NETWORKS

Insight of data generating process informs inductive bias on architecture





- Generative process is a tree-like, ~stationary Markov Process
- Physics algorithms exist to estimate the tree
- Tree-RNN needs much less data to train!

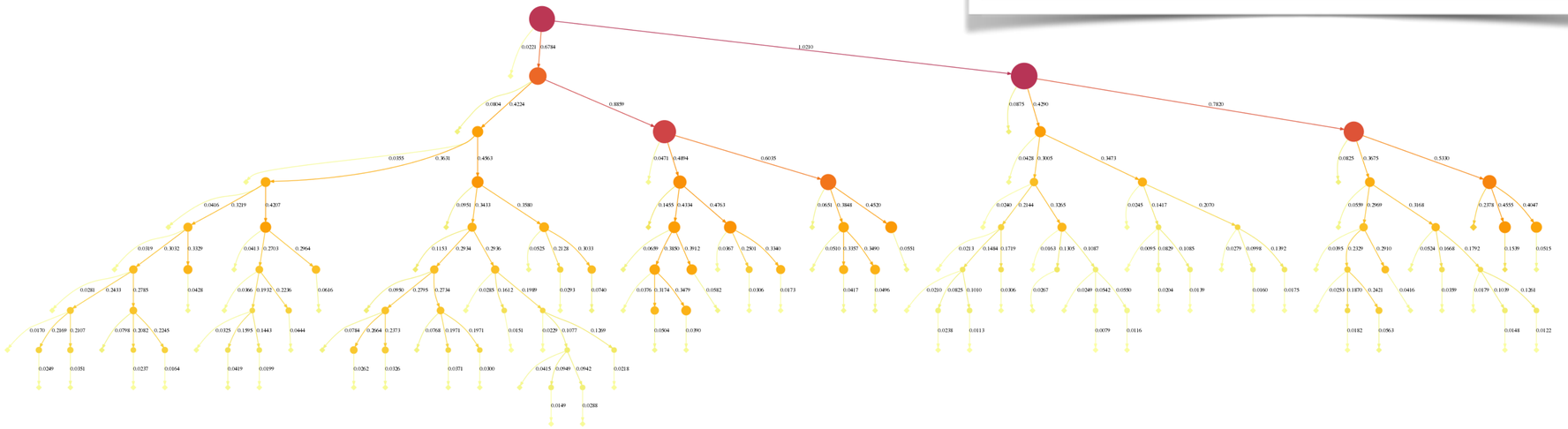
Louppe, Cho, Becot, KC [arXiv:1702.00748]

INDUCTIVE BIAS

arXiv:1902.09914

The Machine Learning Landscape of Top Taggers

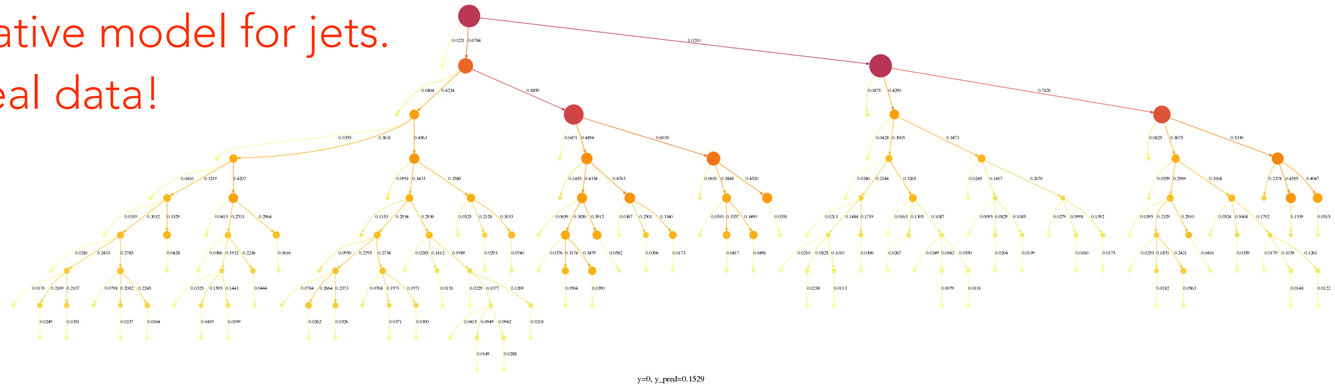
G. Kasieczka (ed)[1], T. Plehn (ed)[2], A. Butter[2], K. Cranmer[3], D. Debnath[4], M. Fairbairn[5], W. Fedorko[6], C. Gay[6], L. Gouskos[7], P. T. Komiske[8], S. Leiss[1], A. Lister[6], S. Macaluso[3,4], E. M. Metodiev[8], L. Moore[9], B. Nachman,[10,11], K. Nordström[12,13], J. Pearkes[6], H. Qu[7], Y. Rath[14], M. Rieger[14], D. Shih[4], J. M. Thompson[2], and S. Varma[5]



	AUC	Acc	$1/\epsilon_B$ ($\epsilon_S = 0.3$) single	mean	median	#Param
CNN [16]	0.981	0.930	914±14	995±15	966±18	610k
ResNeXt [30]	0.984	0.936	1122±47	1246±28	1286±31	1.46M
TopoDNN [18]	0.972	0.916	295±5	378± 5	391 ± 8	59k
Multi-body N-subjettiness 6 [24]	0.979	0.922	792±18	802±12	783±13	57k
Multi-body N-subjettiness 8 [24]	0.981	0.929	867±15	926±20	886±18	58k
TreeNiN [43]	0.982	0.933	1025±11	1209±23	1167±24	34k
P-CNN	0.980	0.930	732±24	838±13	841±14	348k
ParticleNet [47]	0.985	0.938	1298±46	1383±45	1374±41	498k
LBN [19]	0.981	0.931	836±17	852±67	971±20	705k
LoLa [22]	0.980	0.929	722±17	768±11	751±11	127k
Energy Flow Polynomials [21]	0.980	0.932	384			1k
Energy Flow Network [23]	0.979	0.927	633±31	734±13	729±11	82k
Particle Flow Network [23]	0.982	0.932	891±18	1005±21	1005±29	82k
GoaT	0.985	0.939	1368±140		1549±208	35k

CAUSAL, GENERATIVE MODELS FOR JETS

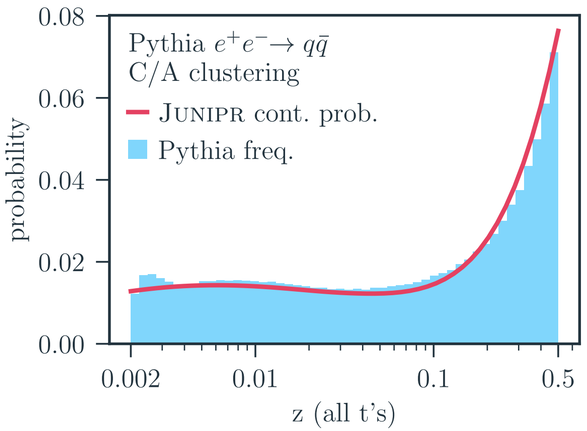
JUNIPR is a causal, generative model for jets.
Can train on real data!

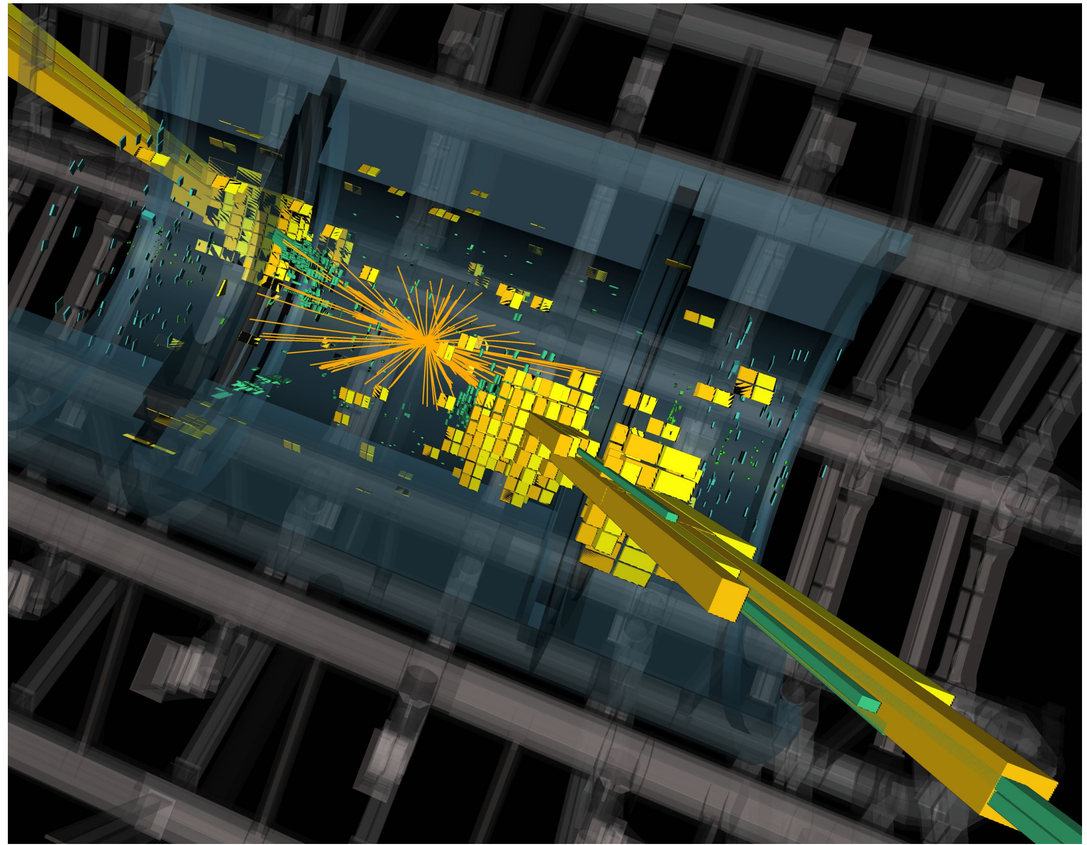


tractable likelihood

$$P_{\text{jet}}(\{p_1, \dots p_n\}) = \left[\prod_{t=1}^{n-1} P_t\left(k_1^{(t+1)}, \dots, k_{t+1}^{(t+1)} \middle| k_1^{(t)}, \dots, k_t^{(t)}\right)\right] \times P_n\left(\text{end} \middle| k_1^{(n)}, \dots, k_n^{(n)}\right).$$

… and it is interpretable





Andreassen, Feige, Frye, Schwartz arXiv:1804.09720

Rethinking Jets

Ginkgo: Toy Generative Model for Jets

Generative model to aid in ML research for jet physics.

NLP analogy: ground-truth parse trees with a known language model

K. Cranmer, S. Macaluso & D. Pappadopulo

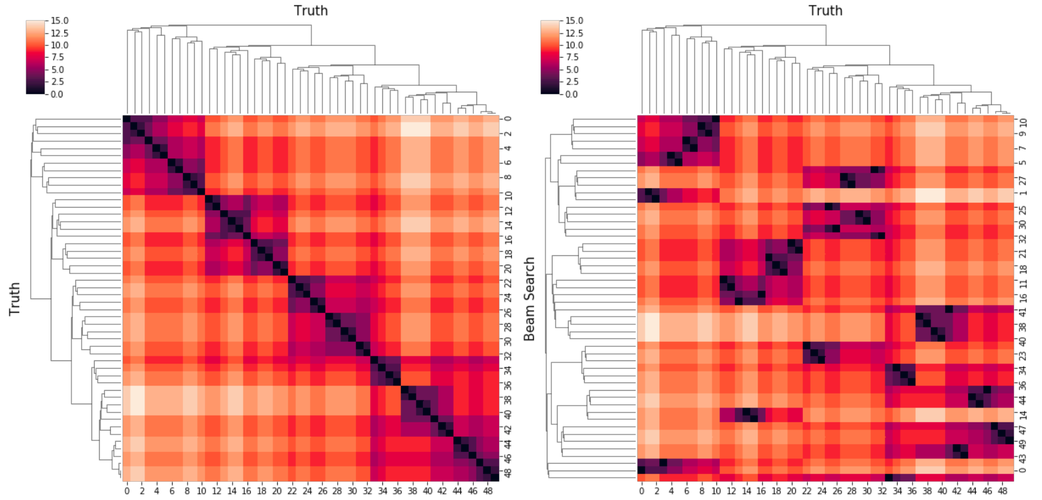
github.com/SebastianMacaluso/ToyJetsShower

Greedy Algorithm

Locally maximizing the likelihood at each step.

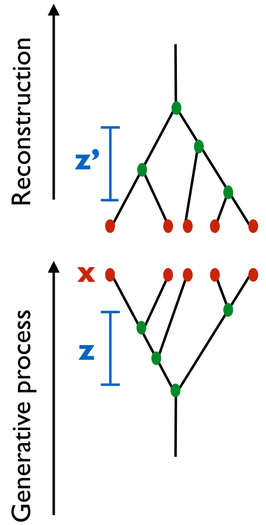
Beam Search Algorithm

Maximize the likelihood of multiple steps before choosing the latent path.



Reframe jet physics in statistical terms

- Joint likelihood $p(x, z|\theta)$
- Maximum likelihood history: $\mathrm{ArgMax}_z\ p(x, z|\theta)$
- Marginal likelihood $p(x|\theta) = \int dz\ p(x, z|\theta)$
- Maximum likelihood parameter: $\mathrm{ArgMax}_\theta\ p(x|\theta)$
- Posterior distribution on histories: $p(z|x, \theta)$
- Posterior distribution on θ : $p(\theta|x)$

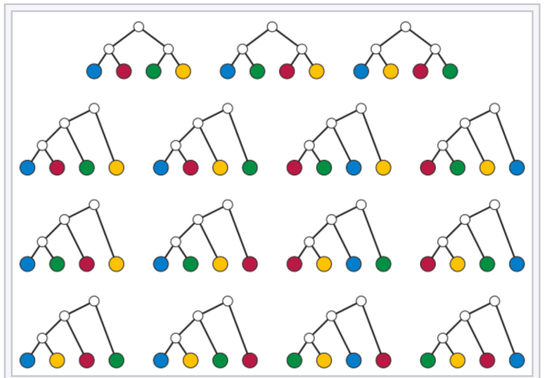


Challenge

Number of clustering histories for N leaves grows as

$$a(N) = (2N - 3)!!$$

# of leaves	Approx. # of trees
4	15
5	100
7	10 k
9	2 M
11	600 M



The fifteen different rooted binary trees (with unordered children) on a set of four labeled leaves, illustrating $15 = (2 \times 4 - 3)!!$ (see article text).

https://en.wikipedia.org/wiki/Double_factorial

Ginkgo: Toy Generative Model for Jets

Generative model to aid in ML research for jet physics.

NLP analogy: ground-truth parse trees with a known language model

K. Cranmer, S. Macaluso & D. Pappadopulo

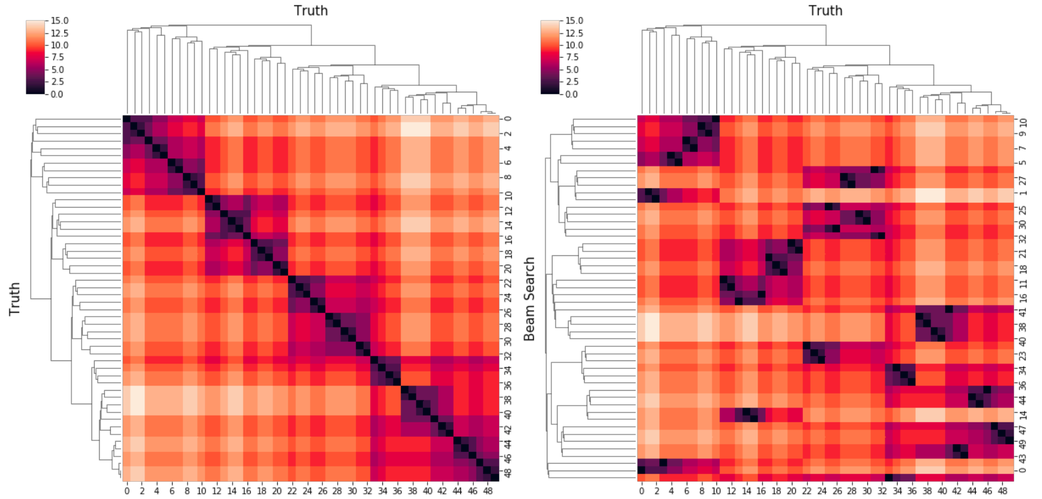
github.com/SebastianMacaluso/ToyJetsShower

Greedy Algorithm

Locally maximizing the likelihood at each step.

Beam Search Algorithm

Maximize the likelihood of multiple steps before choosing the latent path.



Standard clustering algorithm in HEP (anti-kT) is Greedy

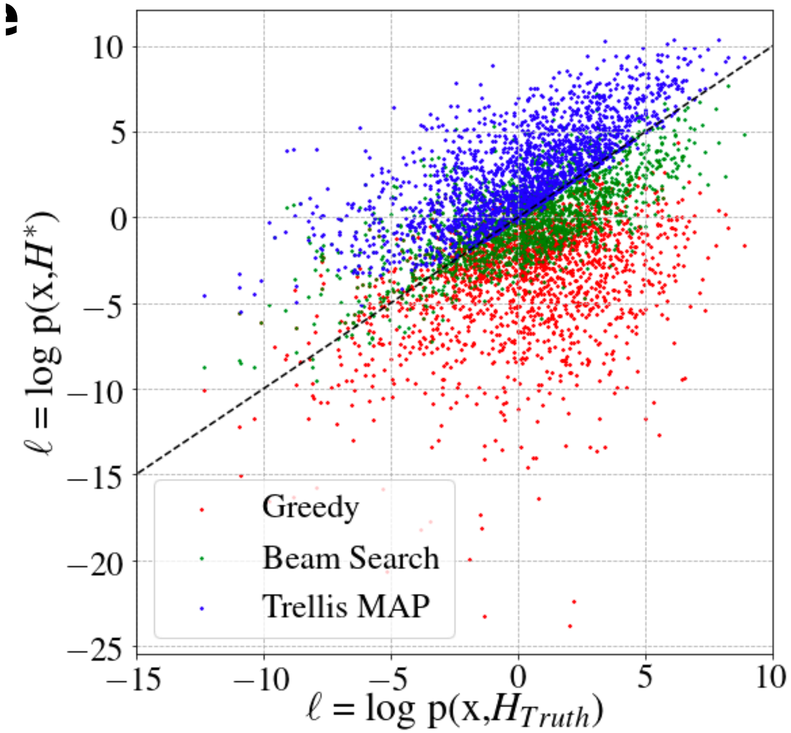
$$p_{\text{Greedy}} < p_{\text{Beam Search}} < p_{\text{Trellis}}$$

Use inspired-research

Seminar at UMass Amherst, Center for Data Science, College of Information & Computer Sciences led to collaboration for hierarchical clustering algorithms.

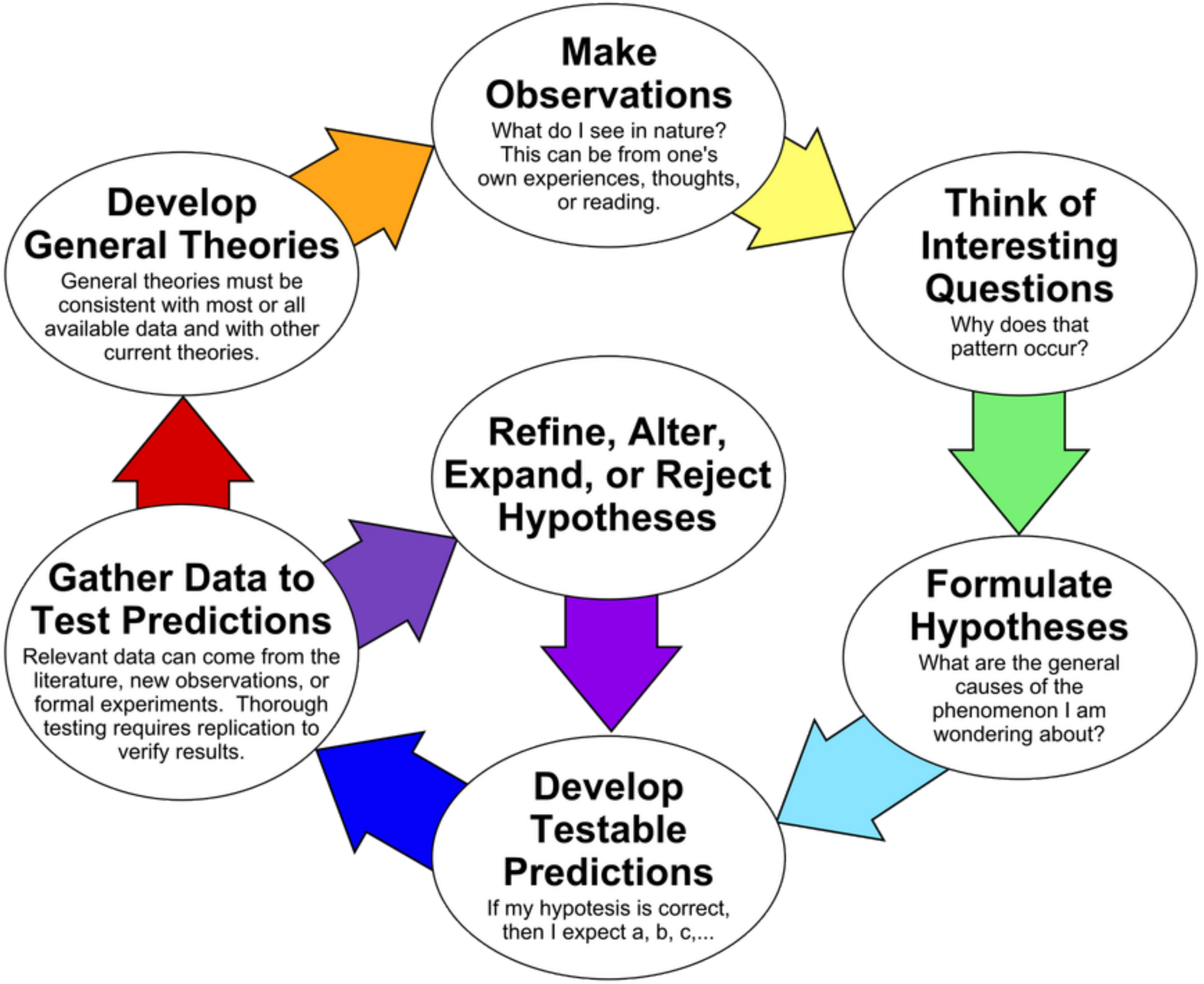
S. Macaluso, C. Greenberg, N. Monath, J. Lee, P. Flaherty, K. Cranmer, A. McGregor, A. McCallum

Applications in other domains, e.g. cancer genomics.

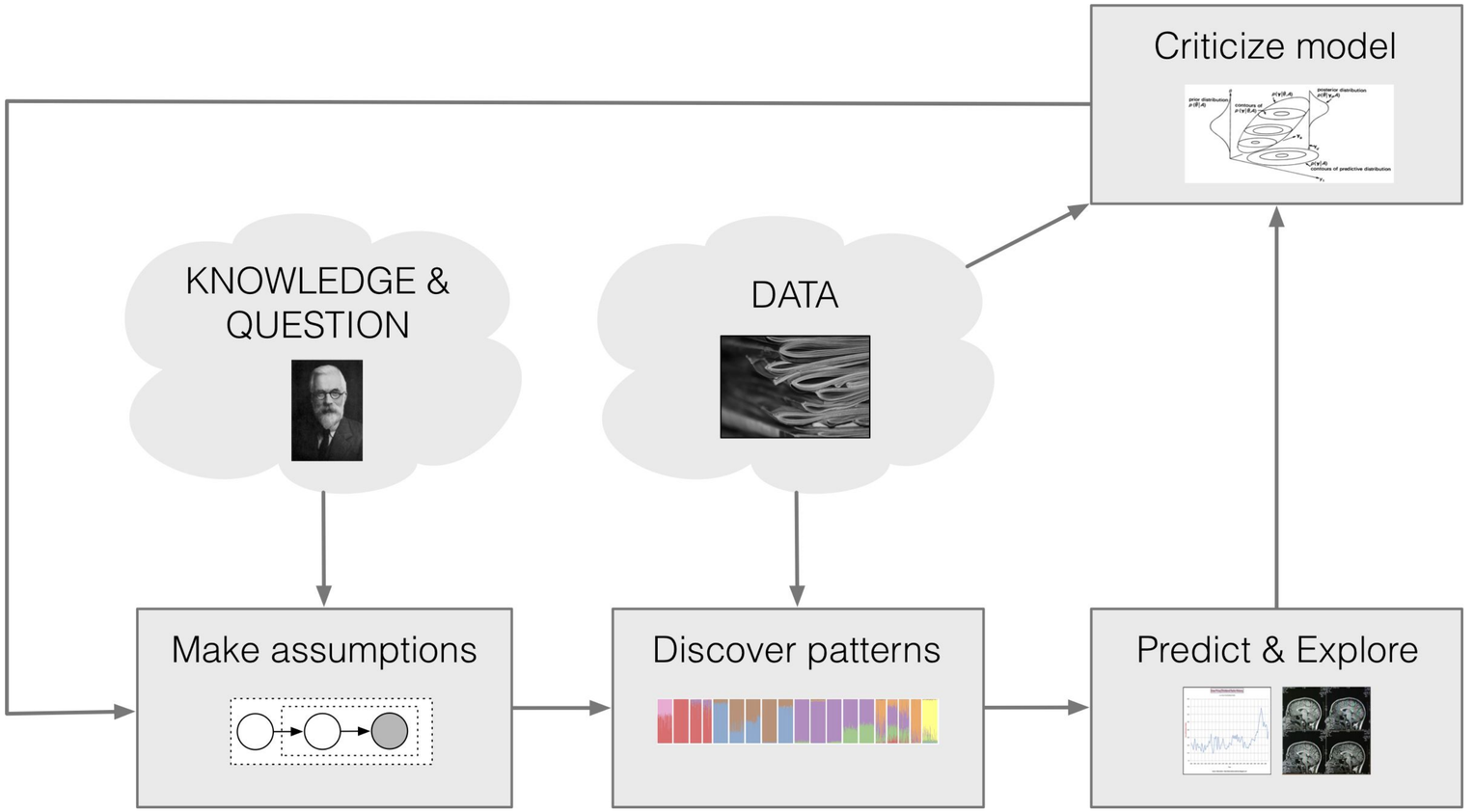


Active Learning —> Active Sciencing

The Scientific Method as an Ongoing Process



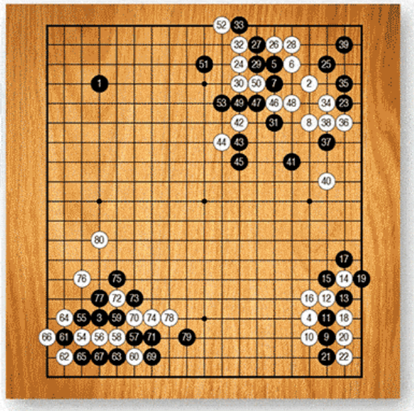
A PROBABILISTIC PIPELINE



[Box, 1980; Rubin, 1984; Gelman et al., 1996; Blei, 2014]

Slide from David Blei, Rajesh Ranganath, Shakir Mohamed, Dustin Tran

Scientist trying to decide what experiment to do next



68 at 61

Captured Stones

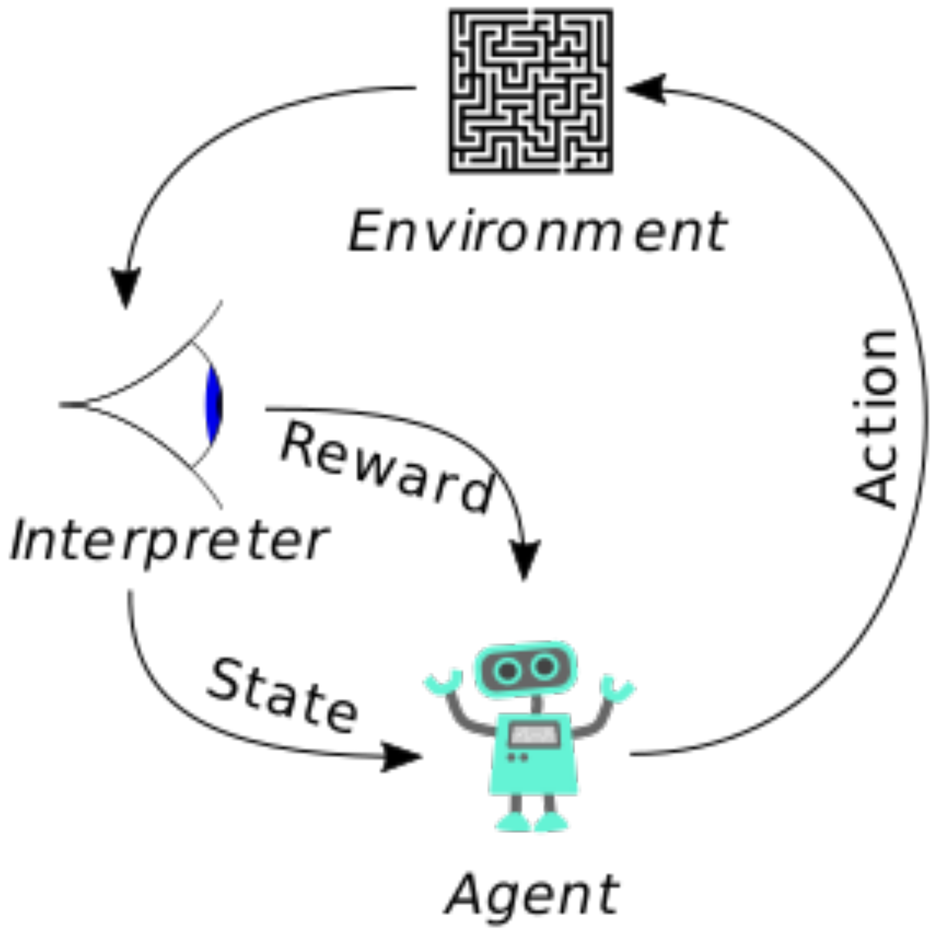
70 hours

AlphaGo Zero plays at super-human level. The game is disciplined and involves multiple challenges across the board.

40 days

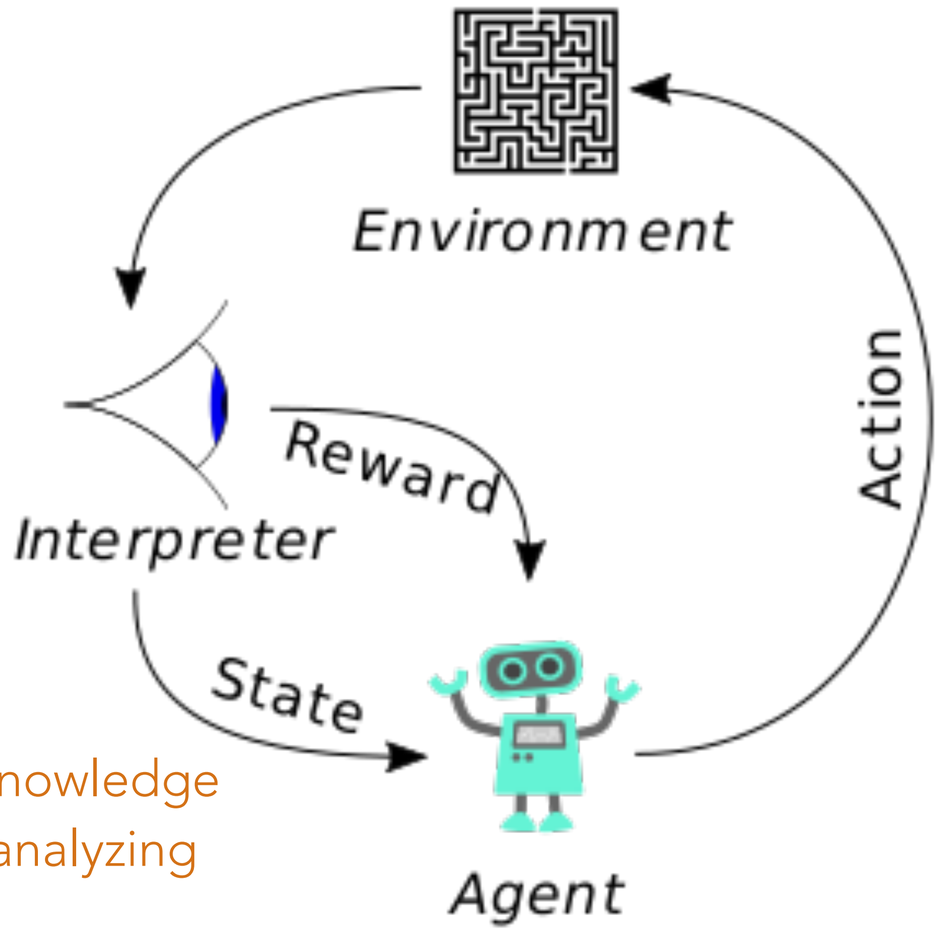
AlphaGo Zero surpasses all other versions of AlphaGo and, arguably, becomes the best Go player in the world. It does this entirely from self-play, with no human intervention and using no historical data.

Scientist trying to decide what experiment to do next



Scientist trying to decide what experiment to do next

perform experiment,
gather data

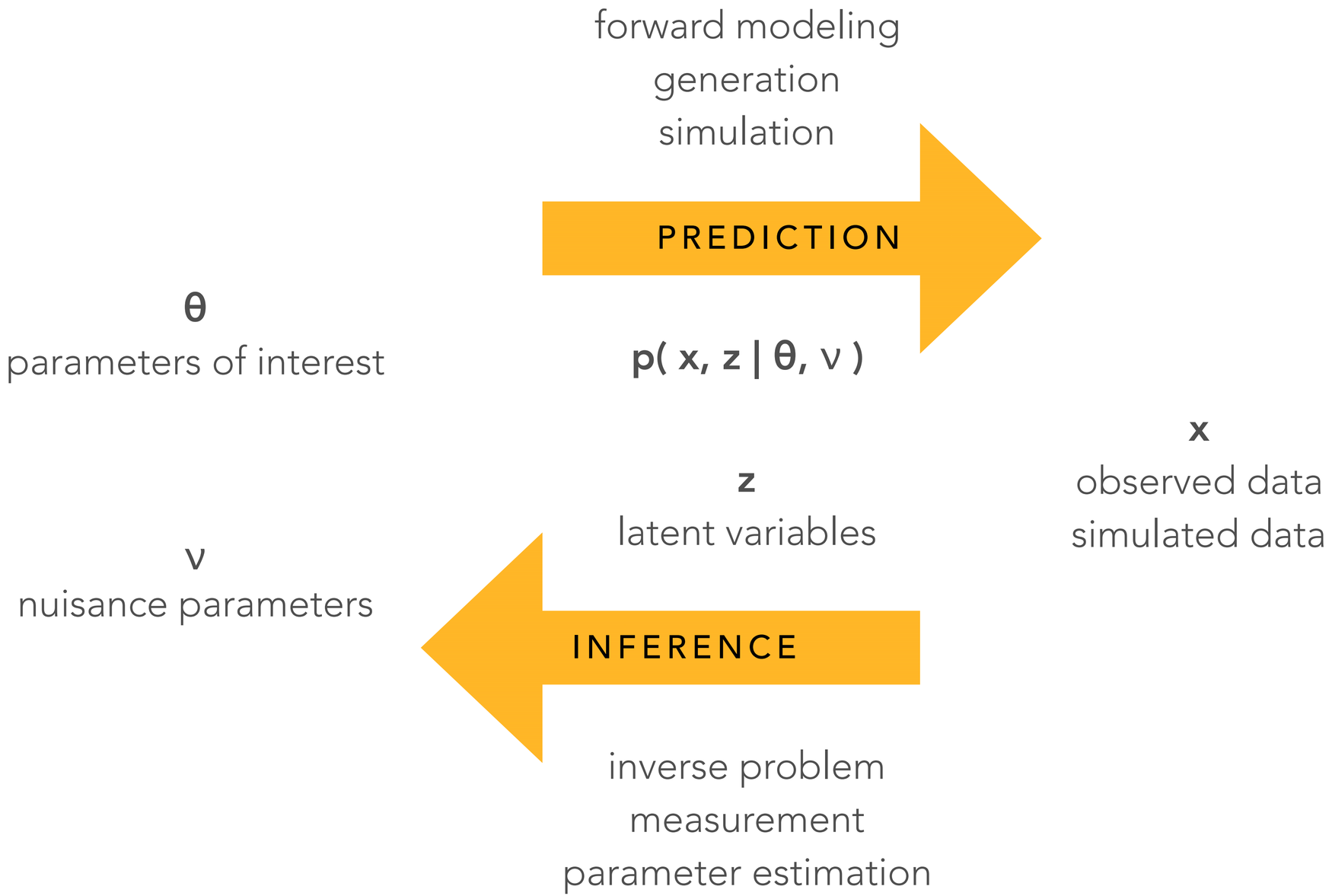


statistical analysis

decide which
experiment to
perform

updated knowledge
based on analyzing
data

NOTATION / TERMINOLOGY



STATISTICAL DECISION THEORY IN 1 SLIDE

Θ - States of nature; X - possible observations; A - action to be taken

$f(x|\theta)$ - statistical model (likelihood); $\pi(\theta)$ - prior

$\delta: X \rightarrow A$ - **decision rule** (take some action based on observation)

$L: \Theta \times A \rightarrow \mathbb{R}$ - **loss function**, real-valued function true parameter and action

$R(\theta,\delta) = E_{f(x|\theta)}[L(\theta, \delta)]$ - **risk**

$r(\pi, \delta) = E_{\pi(\theta)}[R(\theta,\delta)]$ - **Bayes risk** (expectation over θ w.r.t. prior and possible observations)

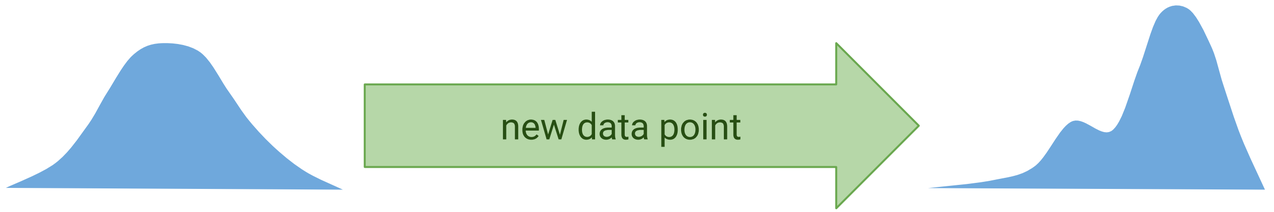
$\rho(\pi, \delta \mid x) = E_{\pi(\theta|x)}[L(\theta,\delta(x))]$ - **expected loss** (expectation over θ w.r.t. posterior $\pi(\theta|x)$)

Dustin Tran
Research Scientist at Google Brain
trandustin@google.com
Blog

Active Learning & Control

Given data points $\{x, y\}$, how to select the next data point to fit the model?

Ex. Select data points which maximize expected information gain. [Lindley et al. 1956; Mackay 1992; Houthooft et al. 2016]



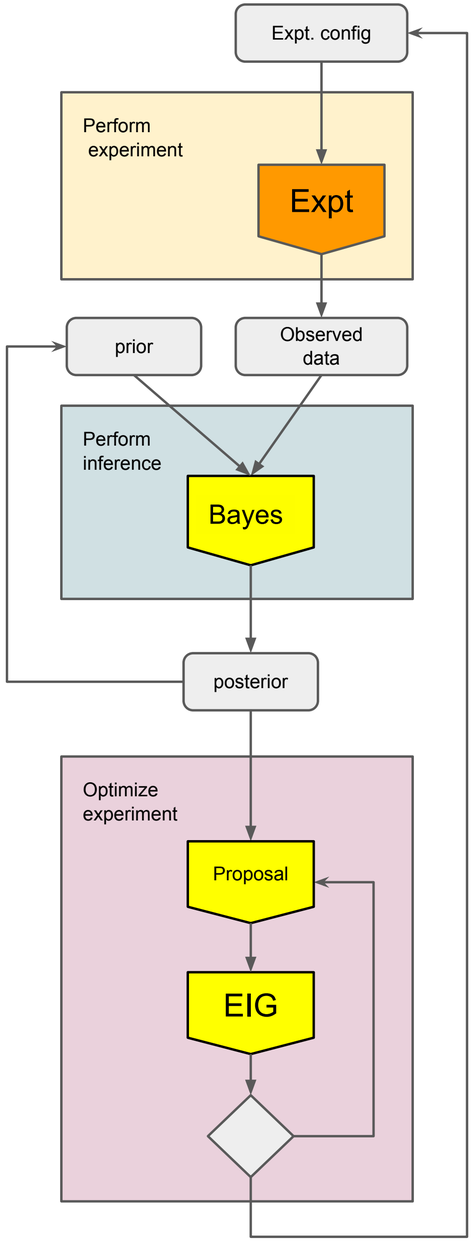
$$\arg\max_{\boldsymbol{x}} \mathrm{H}[\boldsymbol{\theta}|\mathcal{D}] - \mathbb{E}_{y\sim p(y|\boldsymbol{x}\mathcal{D})}\left[\mathrm{H}[\boldsymbol{\theta}|y,\boldsymbol{x},\mathcal{D}]\right]$$

Uncertainty determines which **x** is most informative and, therefore, the model's success.

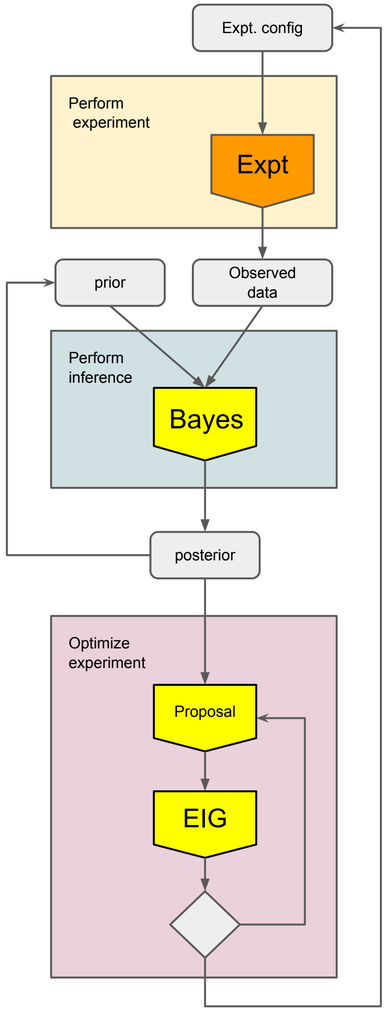
[Hafner et al., 2019]

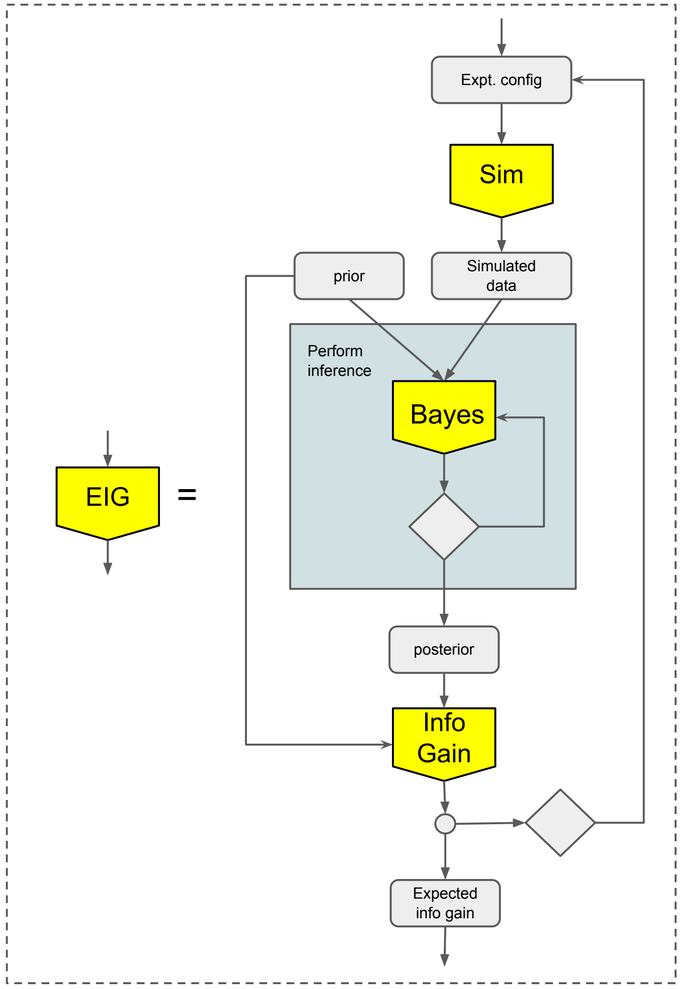
"ACTIVE SCIENCING"

https://github.com/cranmer/active_sciencing



"ACTIVE SCIENCING"





https://github.com/cranmer/active_sciencing

EXPECTED INFORMATION GAIN

Dustin Tran
Research Scientist at Google Brain
trandustin@google.com
Blog

Active Learning & Control

Given data points $\{x, y\}$, how to select the next data point to fit the model?

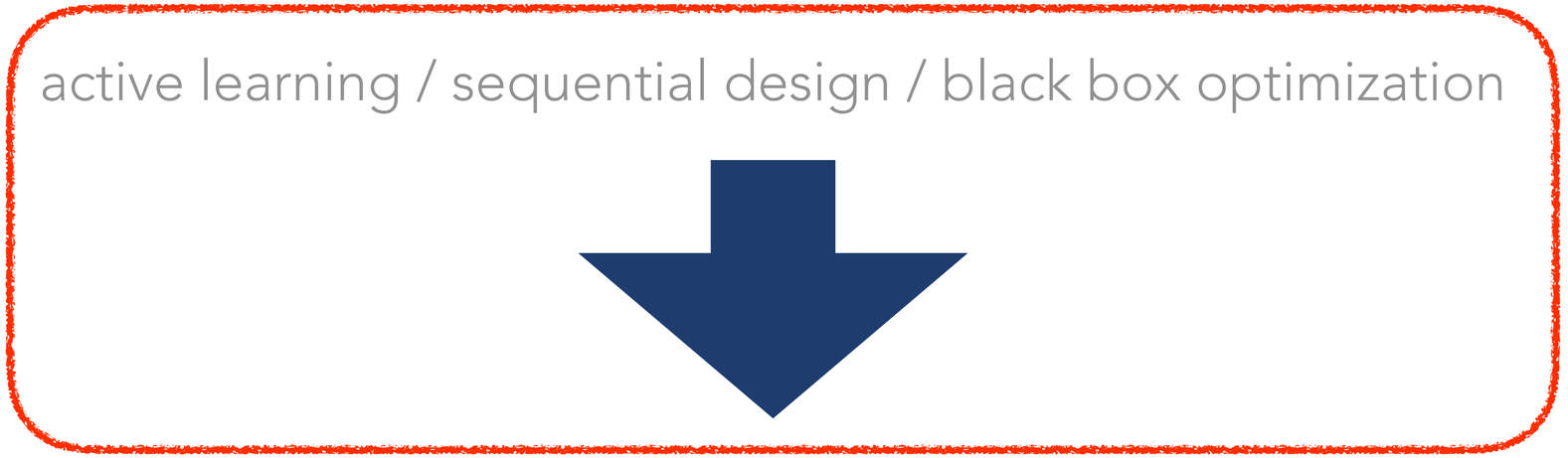
Ex. Select data points which maximize expected information gain. [Lindley et al. 1956; Mackay 1992; Houthooft et al. 2016]

$$\arg\max_{\boldsymbol{x}} \mathrm{H}[\boldsymbol{\theta}|\mathcal{D}] - \mathbb{E}_{y\sim p(y|\boldsymbol{x}\mathcal{D})}\left[\mathrm{H}[\boldsymbol{\theta}|y,\boldsymbol{x},\mathcal{D}]\right]$$

Uncertainty determines which **x** is most informative and, therefore, the model's success.

[Hafner et al., 2019]

SYNTHESIS



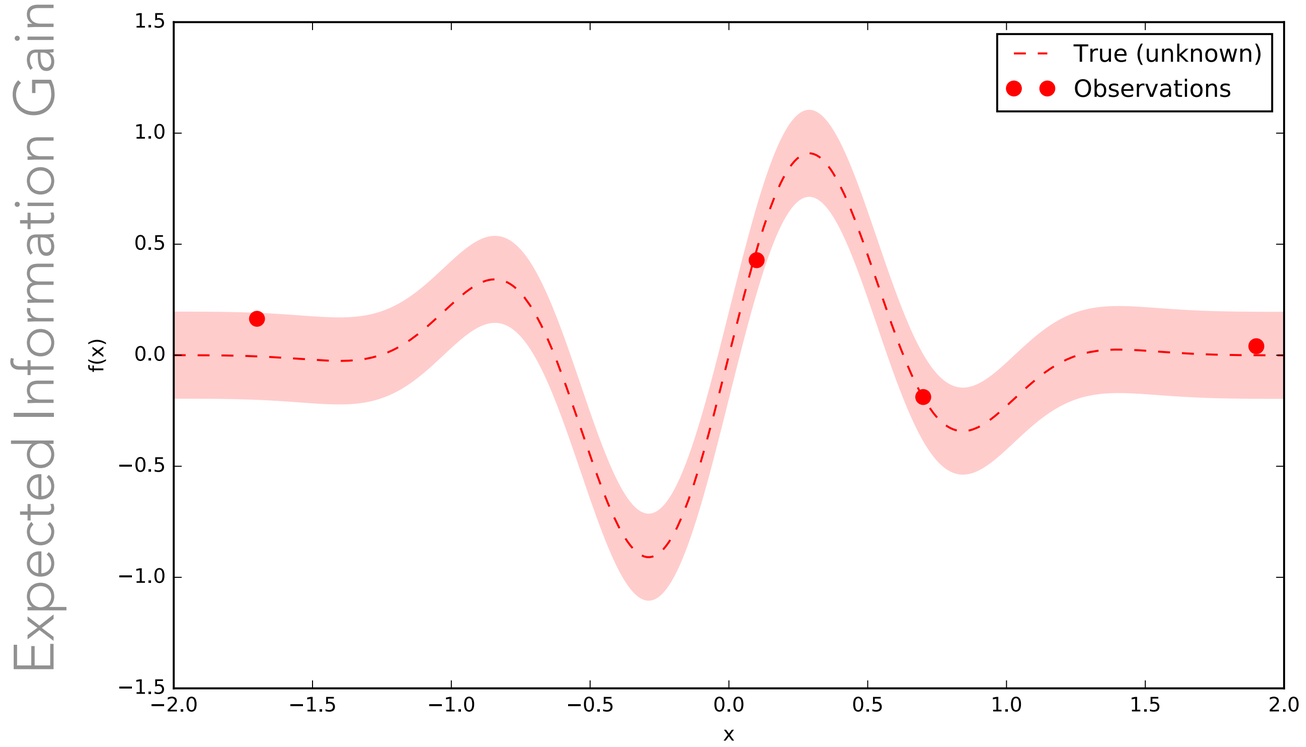
Active Sciencing

reusable workflows

simulation-based / likelihood-free inference engines

BAYESIAN OPTIMIZATION

What experimental configuration should we evaluate next?

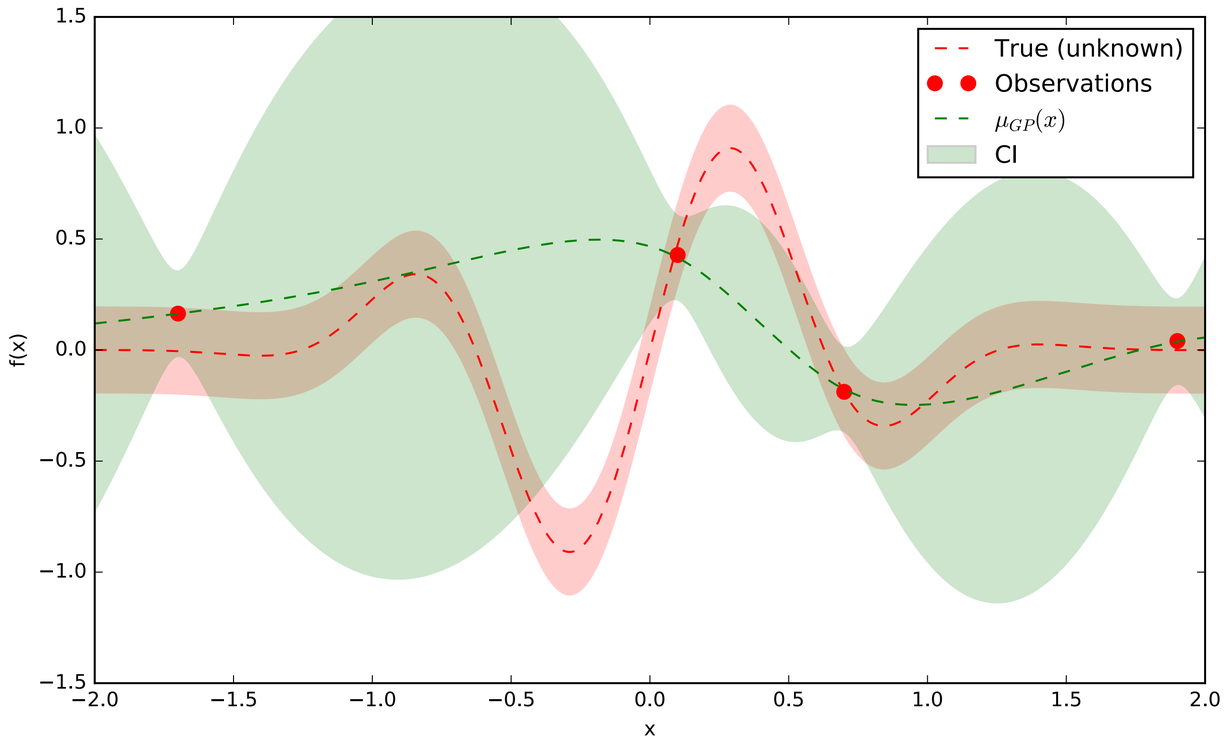


Slides from Gilles Louppe. See scikit-optimize project

BAYESIAN OPTIMIZATION

Build a probabilistic model for the EIG objective function

Expected Information Gain

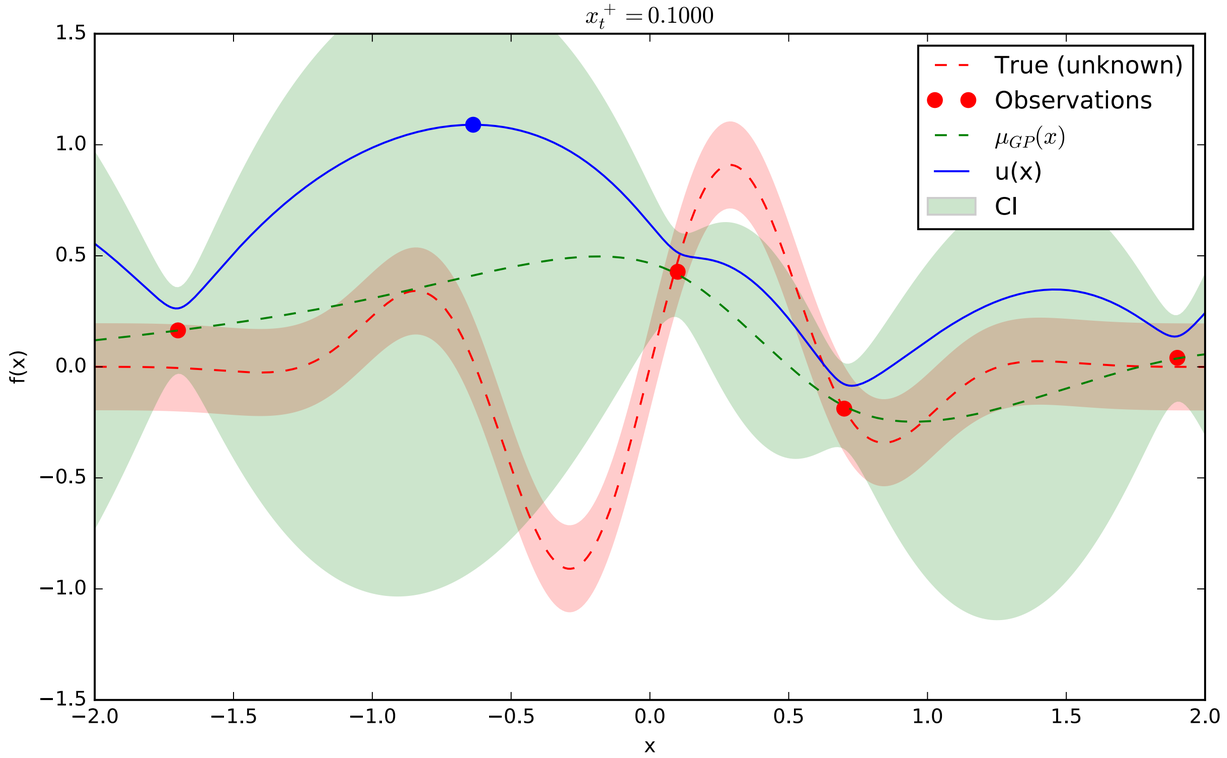


Slides from Gilles Louppe. See scikit-optimize project

BAYESIAN OPTIMIZATION

Use it to define an inexpensive acquisition function, and optimize that.
Use this as next configuration to evaluate expensive EIG

Expected Information Gain

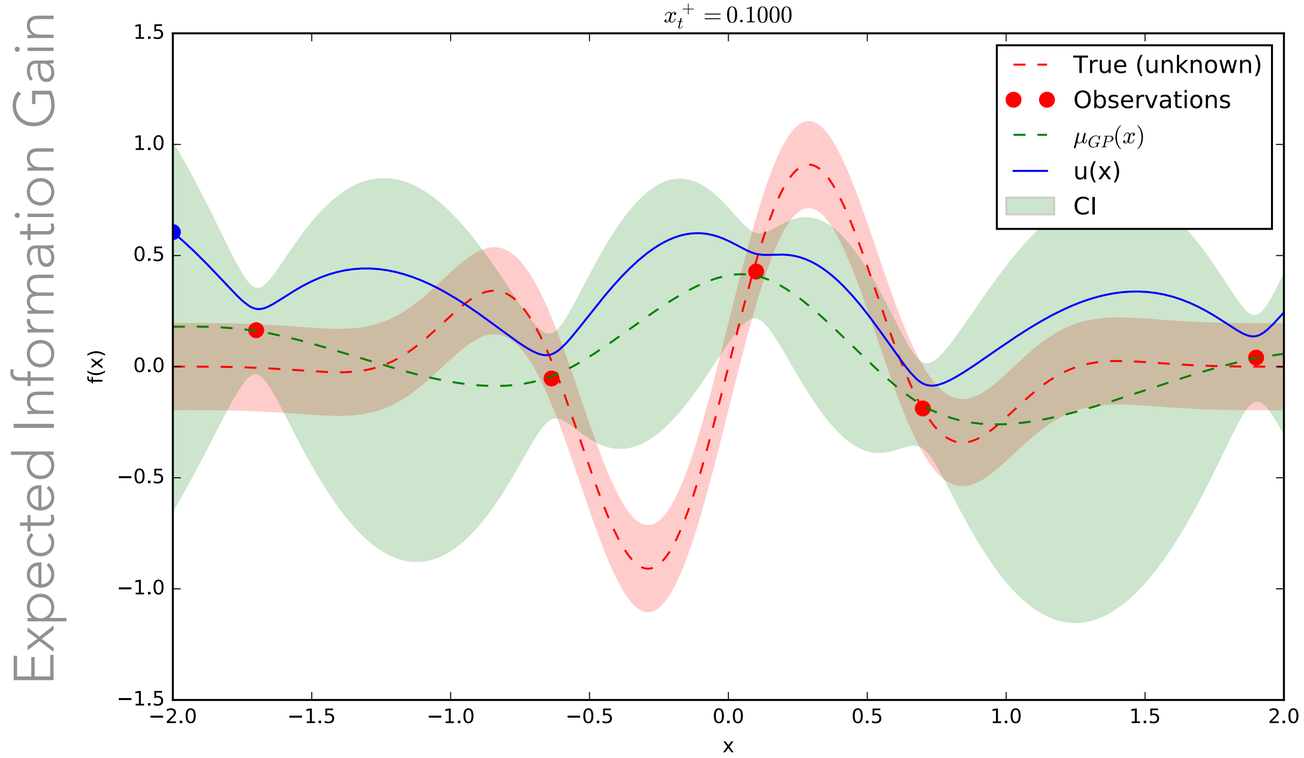


$$x_{t+1} = \arg\max_x \mathrm{UCB}(x)$$

Slides from Gilles Louppe. See scikit-optimize project

BAYESIAN OPTIMIZATION

Then evaluate EIG for that configuration. Repeat



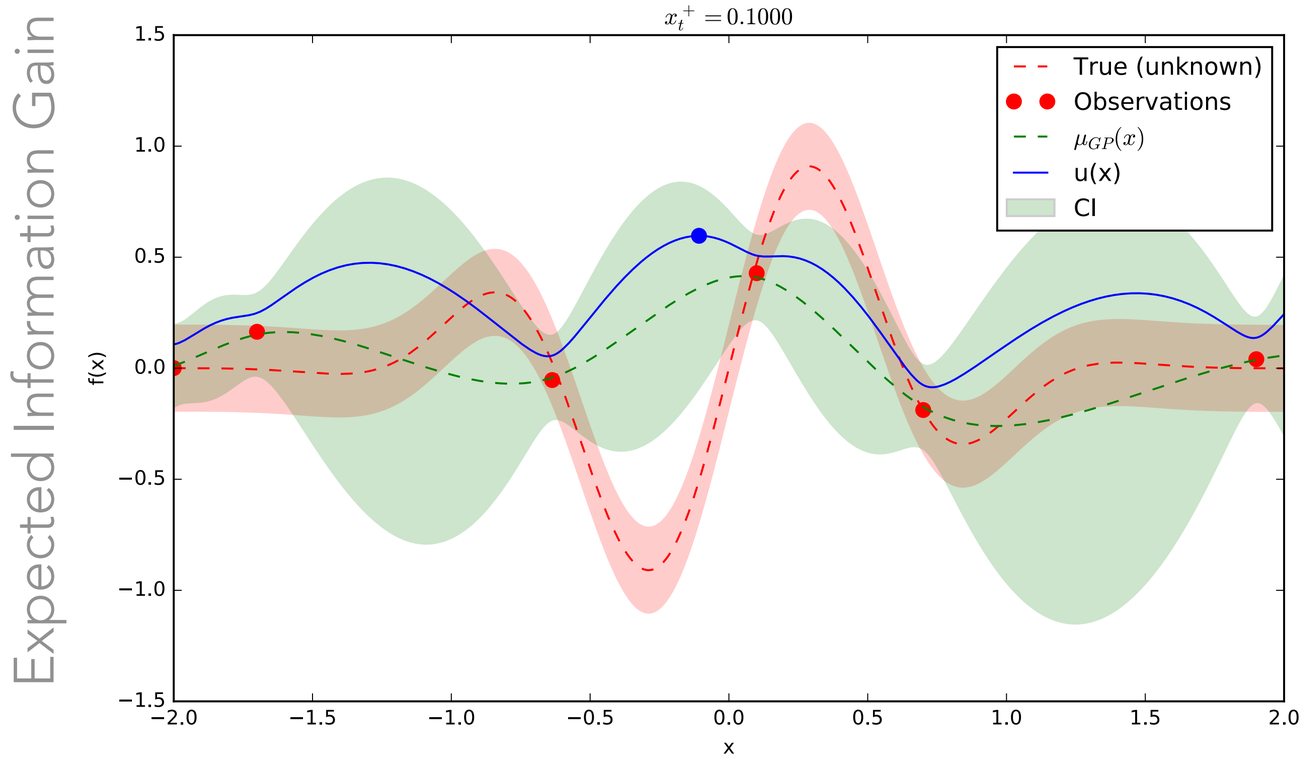
Expected Information Gain

Experimental Configuration

Slides from Gilles Louppe. See scikit-optimize project

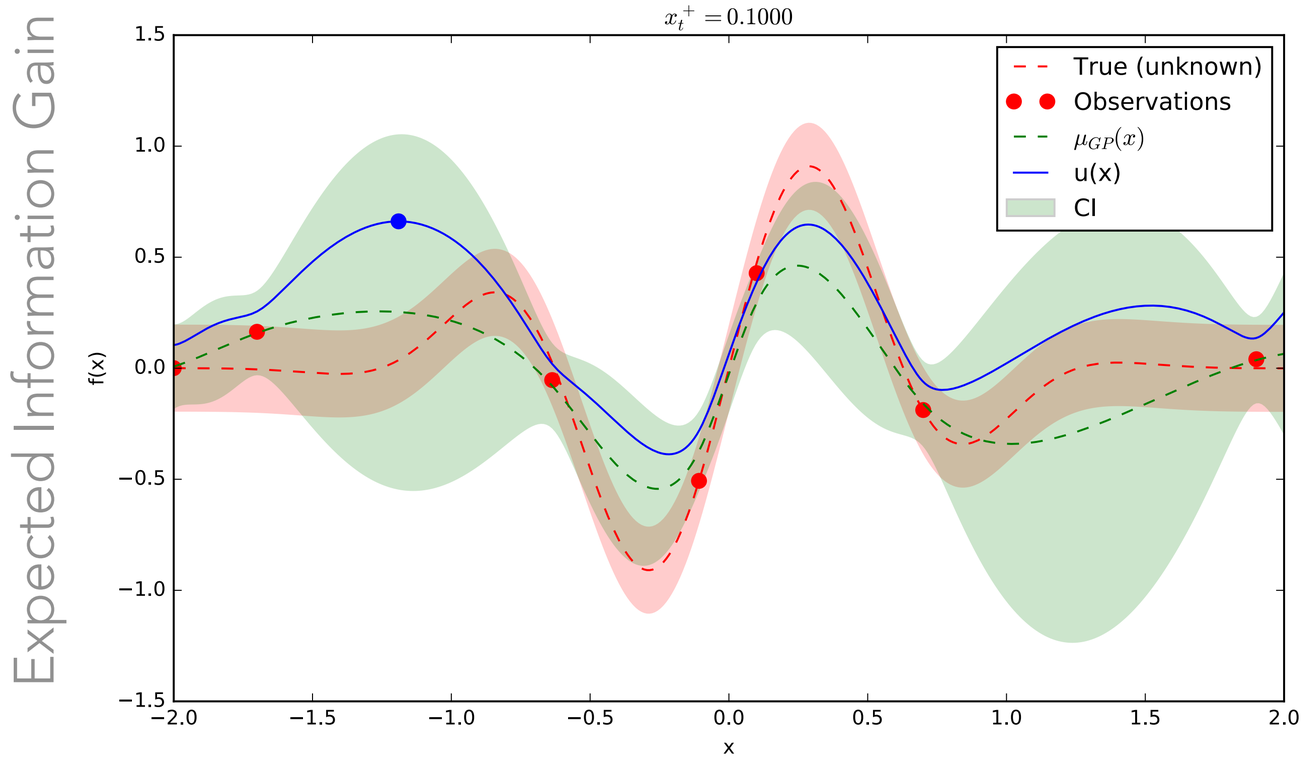
BAYESIAN OPTIMIZATION

Then evaluate EIG for that configuration. Repeat



BAYESIAN OPTIMIZATION

Then evaluate EIG for that configuration. Repeat



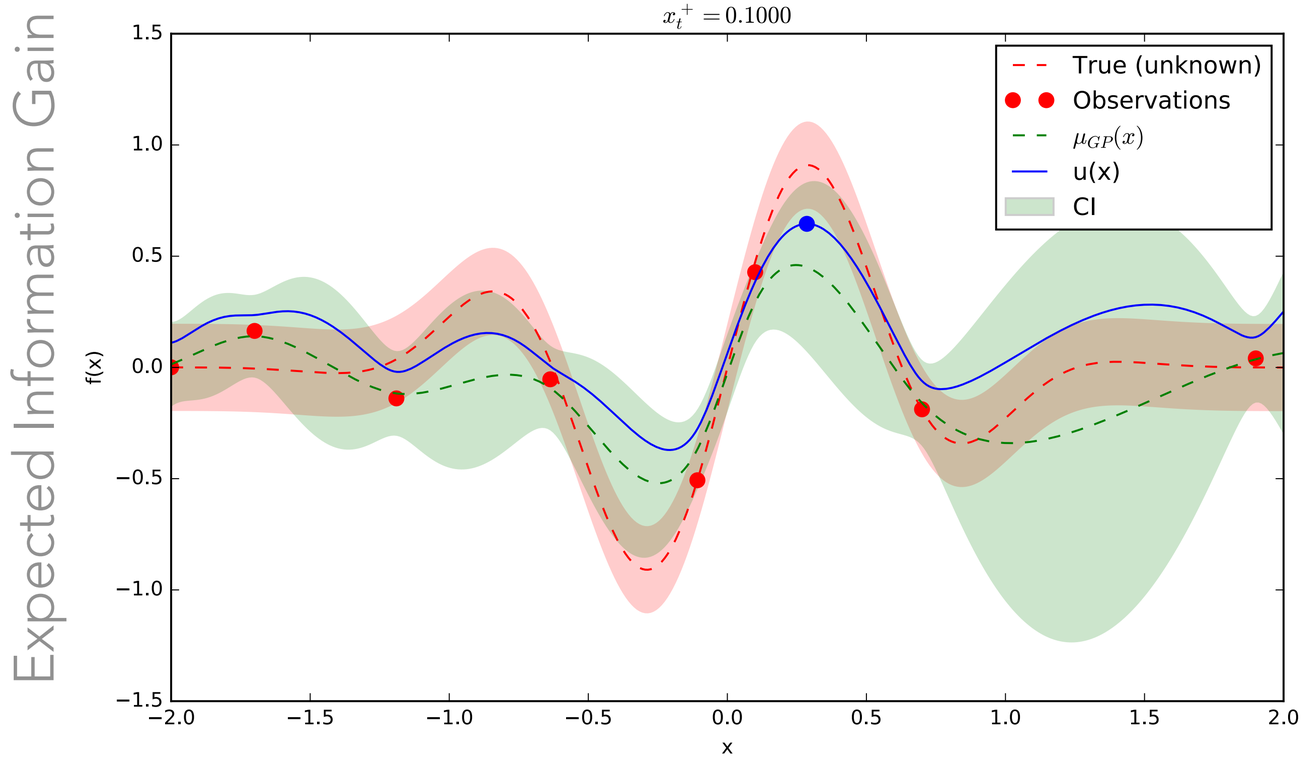
Expected Information Gain

Experimental Configuration

Slides from Gilles Louppe. See scikit-optimize project

BAYESIAN OPTIMIZATION

Then evaluate EIG for that configuration. Repeat



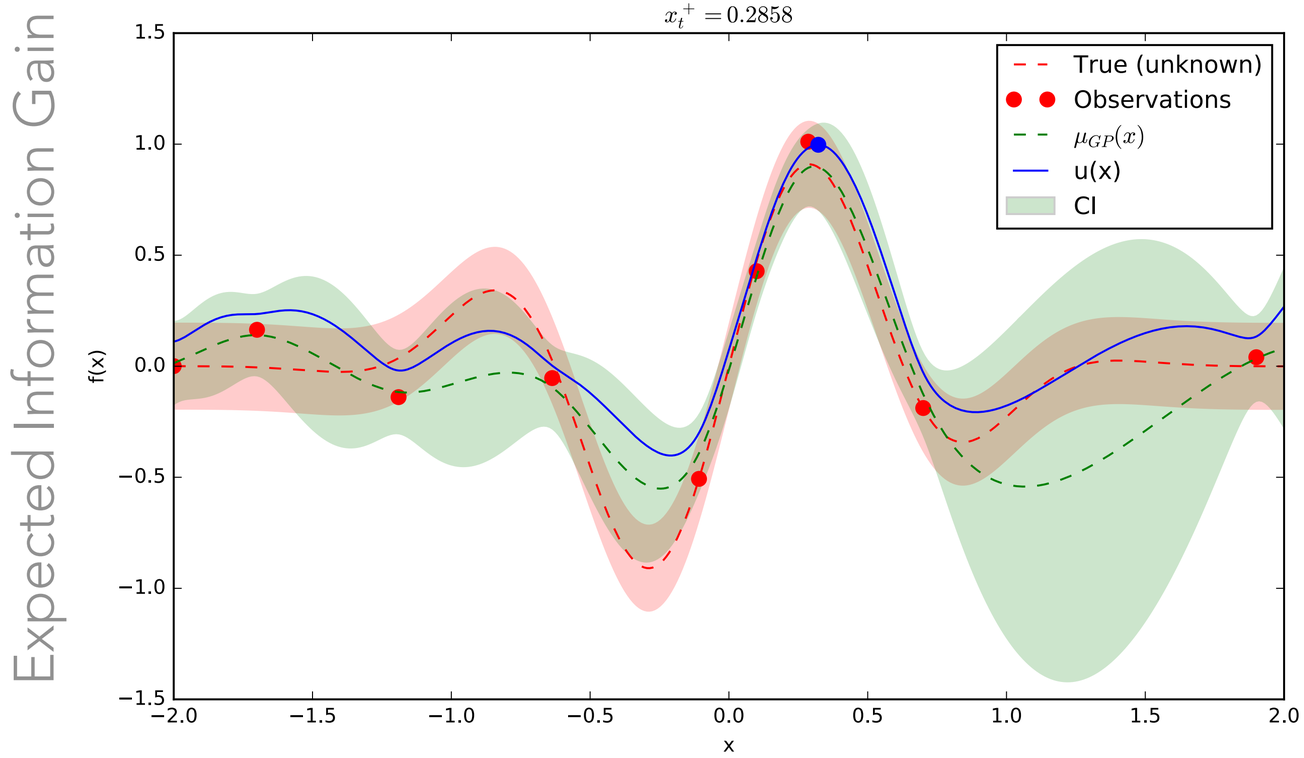
Expected Information Gain

Experimental Configuration

Slides from Gilles Louppe. See scikit-optimize project

BAYESIAN OPTIMIZATION

Then evaluate EIG for that configuration. Repeat



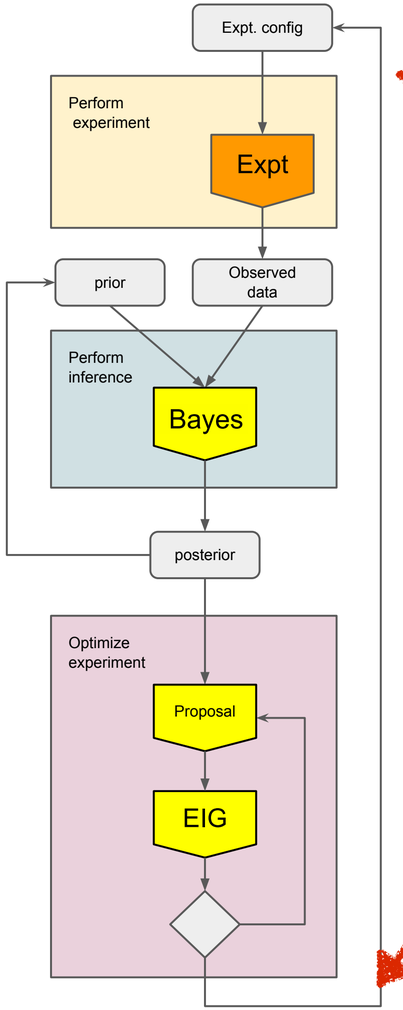
Expected Information Gain

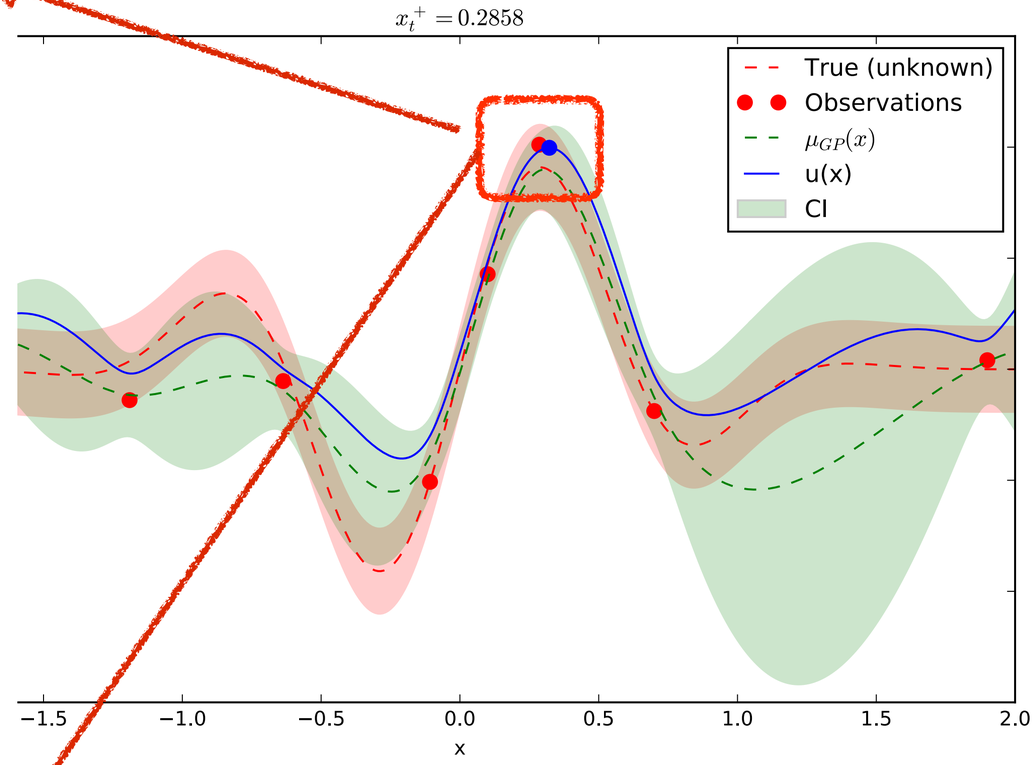
Experimental Configuration

Slides from Gilles Louppe. See scikit-optimize project

BAYESIAN OPTIMIZATION

Now that this experimental configuration is optimized (with simulation), we perform the actual experiment





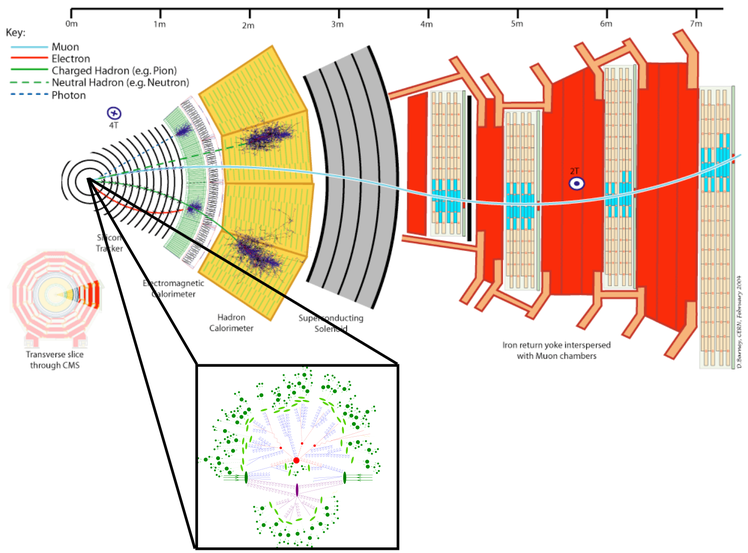
Experimental Configuration

What is the probability model used for inference?

Shouldn’t we use the same simulator we used to evaluate the expected information gain?

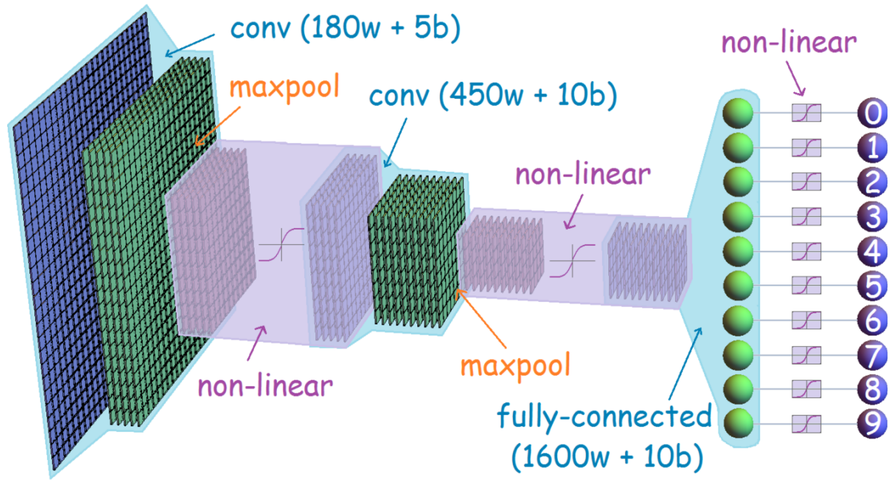
APPROACHES TO LIKELIHOOD-FREE INFERENCE

Use simulator
(much more efficiently)



- Approximate Bayesian Computation (ABC)
- Probabilistic Programming
- Adversarial Variational Optimization (AVO)

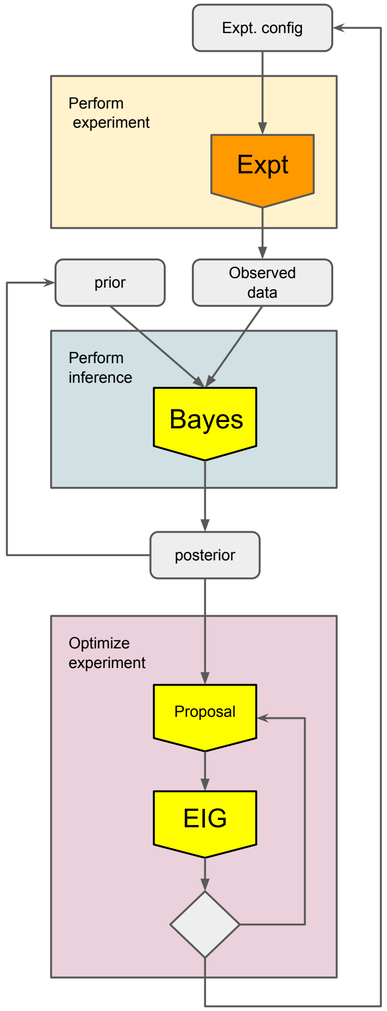
Learn simulator
(with deep learning)

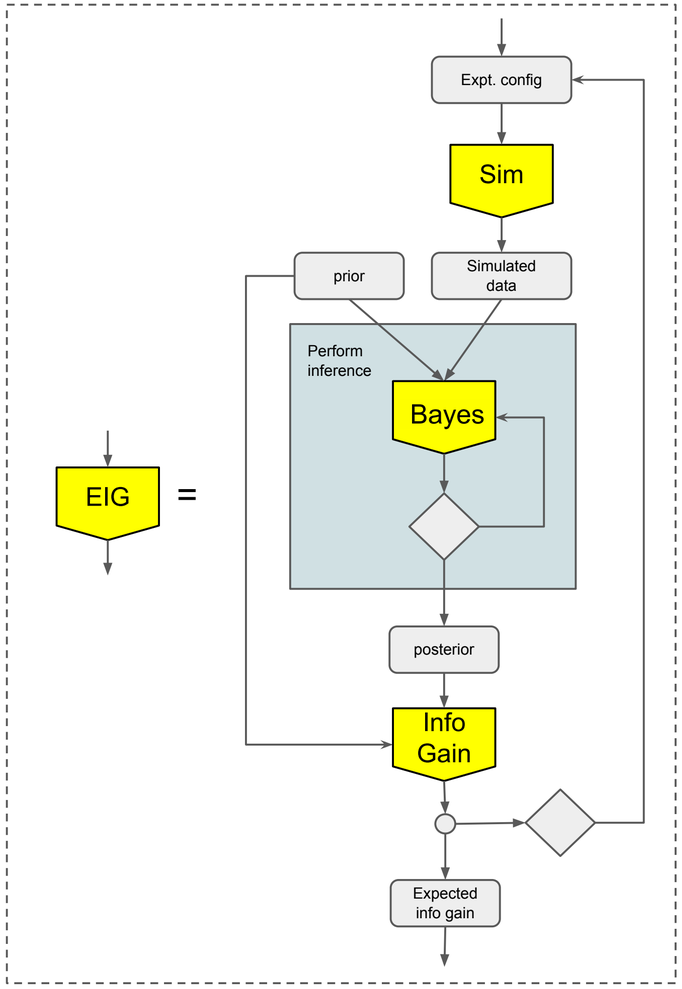


- Generative Adversarial Networks (GANs), Variational Auto-Encoders (VAE)
- Likelihood ratio from classifiers (CARL)
- Autoregressive models, Normalizing Flows

[image credit: A.P. Goucher]

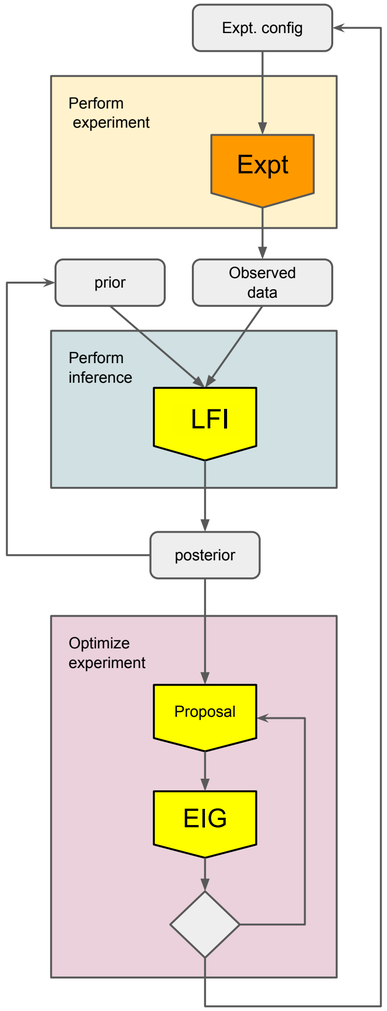
"ACTIVE SCIENCING"

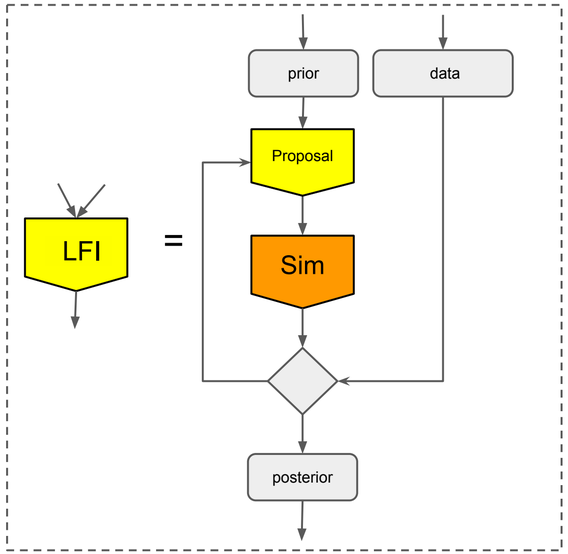


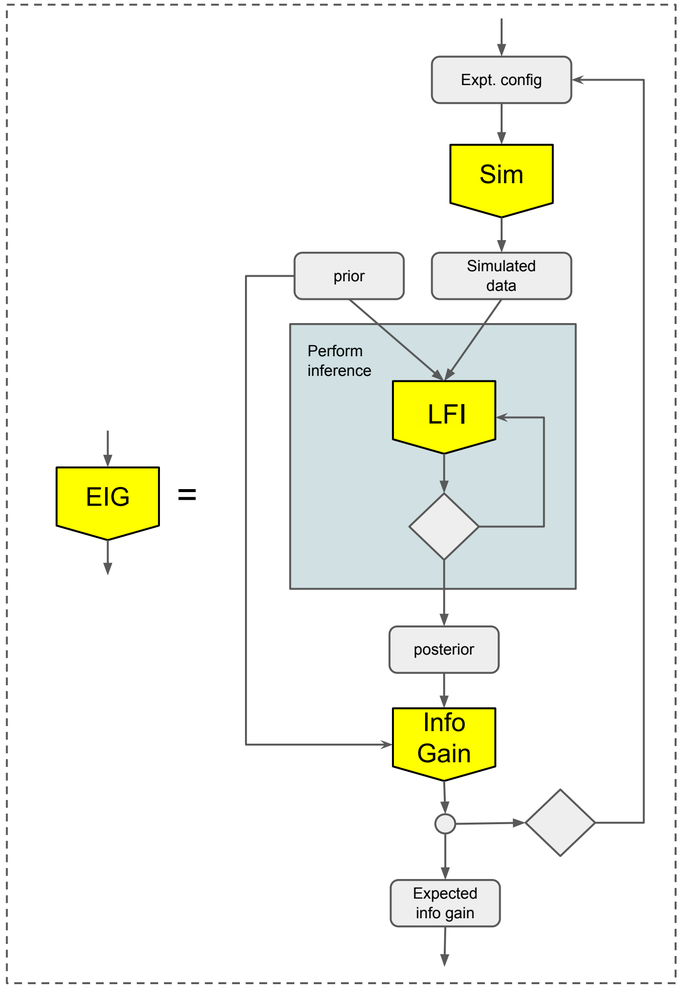


https://github.com/cranmer/active_sciencing

"ACTIVE SCIENCING"







https://github.com/cranmer/active_sciencing

SYNTHESIS

active learning / sequential design / black box optimization

Active Sciencing

reusable workflows

simulation-based /
likelihood-free
inference engines

http://reanahub.io

reana

Reproducible research data analysis platform

Flexible

Run many computational workflow engines.

COMMON WORKFLOW LANGUAGE

Scalable

Support for remote compute clouds.

kubernetes

Reusable

Containerise once, reuse elsewhere. Cloud-native.

Free

Free Software. MIT licence.
Made with ❤ at CERN.

CERN

NSF

diana hep

iris hep
Institute for Research and Innovation in Software for High Energy Physics

The SCAILFIN Project
scailfin.github.io

ENCAPSULATING THE SIMULATION

https://github.com/lukasheinrich/weinberg-test

README.md

Run HEP workflows from the web.

by Kyle Cranmer and Lukas Heinrich

An example notebook on how to generate simulated high energy physics collision events using the generator package MadGraph. Simulated datasets obtained from this notebook can then be used to train and evaluate the performance of generative models for physics.

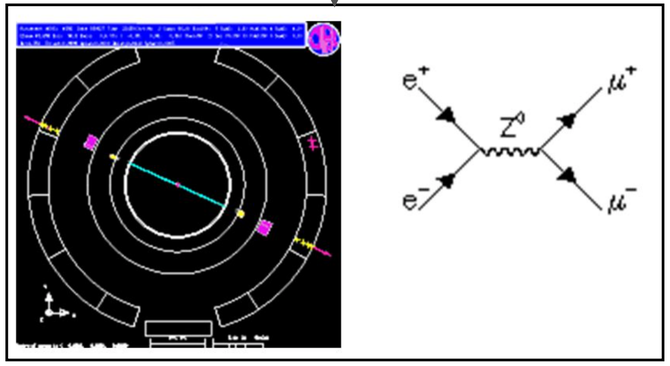
Usage:

This repository has been equipped with a Dockerfile to encapsulate its software environment. It can be used with the mybinder service to launch an ephemeral jupyter notebook server to run the notebook.

Click on the below badge and open the notebook `adage.ipynb` .

launch binder

$$\mathcal{L}_{SM} = \underbrace{\frac{1}{4}\mathbf{W}_{\mu\nu}\cdot\mathbf{W}^{\mu\nu} - \frac{1}{4}B_{\mu\nu}B^{\mu\nu} - \frac{1}{4}G^a_{\mu\nu}G^{\mu\nu}_a}_{\text{kinetic energies and self-interactions of the gauge bosons}}$$
$$+ \underbrace{\bar{L}\gamma^\mu(i\partial_\mu - \frac{1}{2}g\tau\cdot\mathbf{W}_\mu - \frac{1}{2}g'YB_\mu)L + \bar{R}\gamma^\mu(i\partial_\mu - \frac{1}{2}g'YB_\mu)R}_{\text{kinetic energies and electroweak interactions of fermions}}$$
$$+ \underbrace{\frac{1}{2}\left|(i\partial_\mu - \frac{1}{2}g\tau\cdot\mathbf{W}_\mu - \frac{1}{2}g'YB_\mu)\,\phi\right|^2 - V(\phi)}_{W^\pm,Z,\gamma,\text{and Higgs masses and couplings}}$$
$$+ \underbrace{g''(\bar{q}\gamma^\mu T_a q)\,G^a_\mu}_{\text{interactions between quarks and gluons}} + \underbrace{(G_1\bar{L}\phi R + G_2\bar{L}\phi_c R + h.c.)}_{\text{fermion masses and couplings to Higgs}}$$



other electroweak parameters. This can be shown with Eq. (2.96), giving

$$A^f_{FB}(s) \simeq A^f_{FB}(m_Z^2) + \frac{(s-m_Z^2)}{s}\frac{3\pi\alpha(s)}{\sqrt{2}\mathrm{G_F}m_Z^2}\frac{2Q_eQ_fg_{Ae}g_{Af}}{(g_{Ve}^2+g_{Ae}^2)(g_{Vf}^2+g_{Af}^2)}\ . \quad (8.30)$$

ENCAPSULATING THE SIMULATION

https://github.com/lukasheinrich/weinberg-test

README.md

Run HEP workflows from the web.

by Kyle Cranmer and Lukas Heinrich

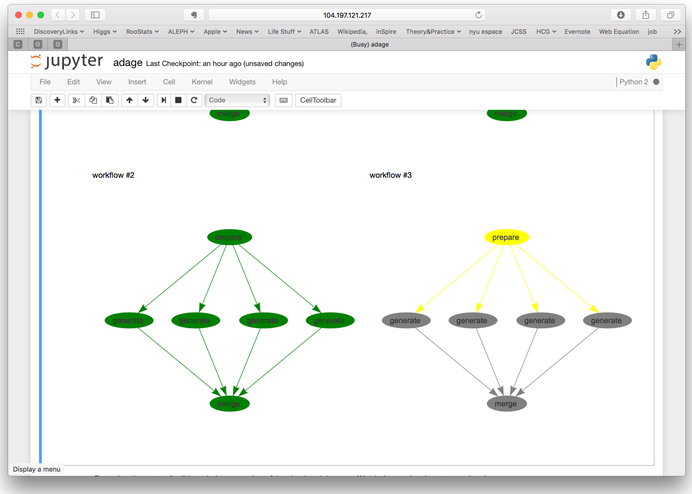
An example notebook on how to generate simulated high energy physics collision events using the generator package MadGraph. Simulated datasets obtained from this notebook can then be used to train and evaluate the performance of generative models for physics.

Usage:

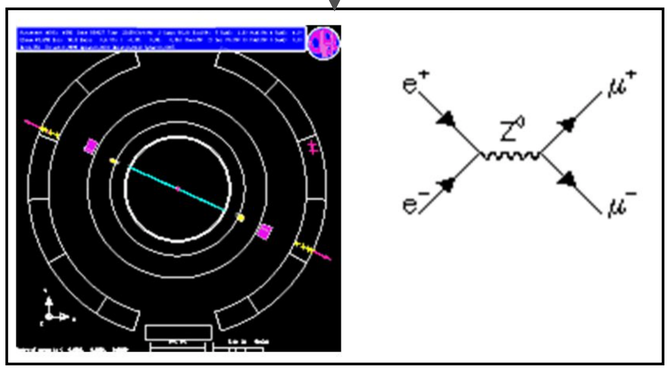
This repository has been equipped with a Dockerfile to encapsulate its software environment. It can be used with the mybinder service to launch an ephemeral jupyter notebook server to run the notebook.

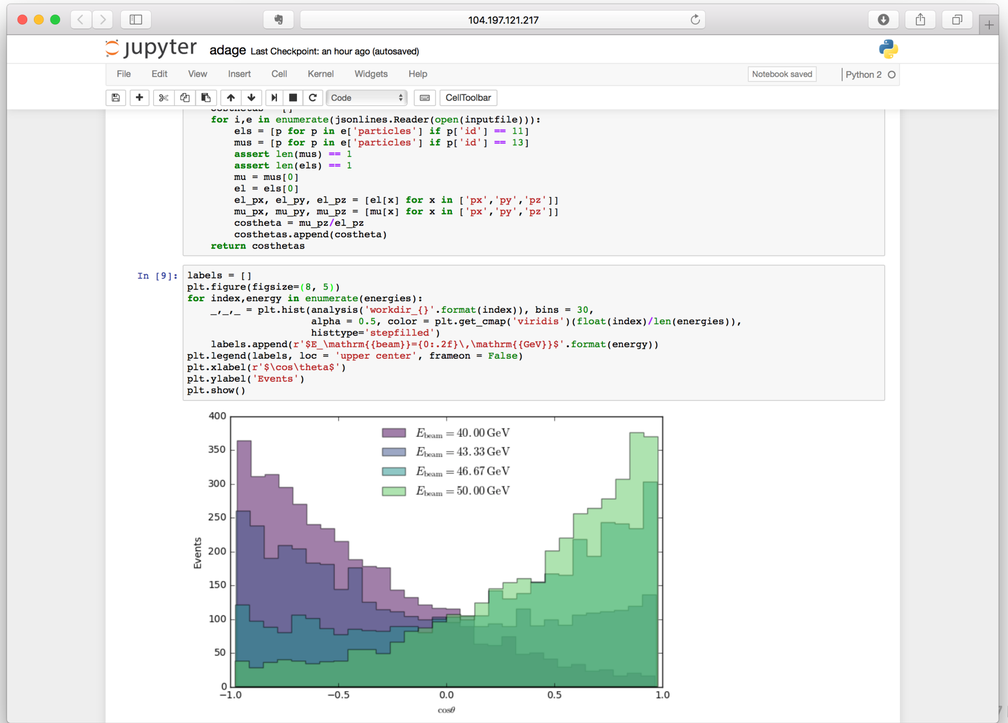
Click on the below badge and open the notebook `adage.ipynb` .

launch binder



$$\mathcal{L}_{SM} = \underbrace{\frac{1}{4}\mathbf{W}_{\mu\nu}\cdot\mathbf{W}^{\mu\nu} - \frac{1}{4}B_{\mu\nu}B^{\mu\nu} - \frac{1}{4}G^{a}_{\mu\nu}G^{\mu\nu}_{a}}_{\text{kinetic energies and self-interactions of the gauge bosons}}$$
$$+ \underbrace{\bar{L}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})L + \bar{R}\gamma^{\mu}(i\partial_{\mu} - \frac{1}{2}g'YB_{\mu})R}_{\text{kinetic energies and electroweak interactions of fermions}}$$
$$+ \underbrace{\frac{1}{2}\left|(i\partial_{\mu} - \frac{1}{2}g\tau\cdot\mathbf{W}_{\mu} - \frac{1}{2}g'YB_{\mu})\,\phi\right|^{2} - V(\phi)}_{W^{\pm},Z,\gamma,\text{and Higgs masses and couplings}}$$
$$+ \underbrace{g''(\bar{q}\gamma^{\mu}T_{a}q)\,G^{a}_{\mu}}_{\text{interactions between quarks and gluons}} + \underbrace{(G_1\bar{L}\phi R + G_2\bar{L}\phi_c R + h.c.)}_{\text{fermion masses and couplings to Higgs}}$$





SYNTHESIS

active learning / sequential design / black box optimization

Active Sciencing

reusable workflows

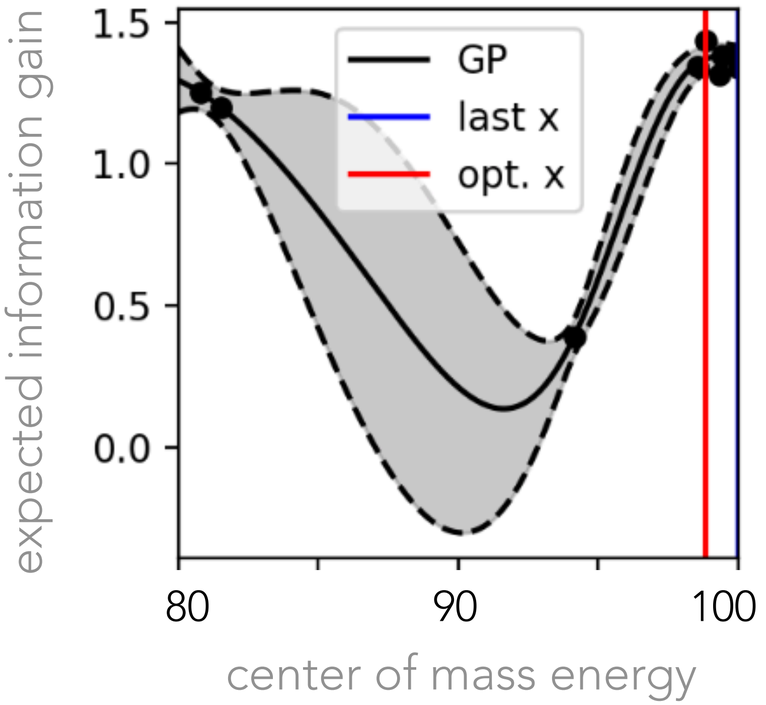
simulation-based / likelihood-free inference engines

A DEMO

https://github.com/cranmer/active_sciencing

Proof-of-principle algorithm can:

- measure parameter of theory (eg. Weinberg angle in Standard Model of particle Physics) from raw data
- optimize experiment (eg. beam energy) for most sensitive measurement



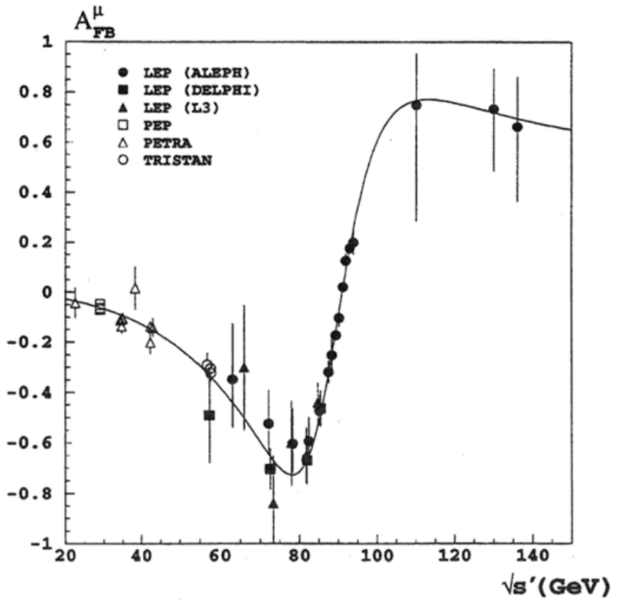
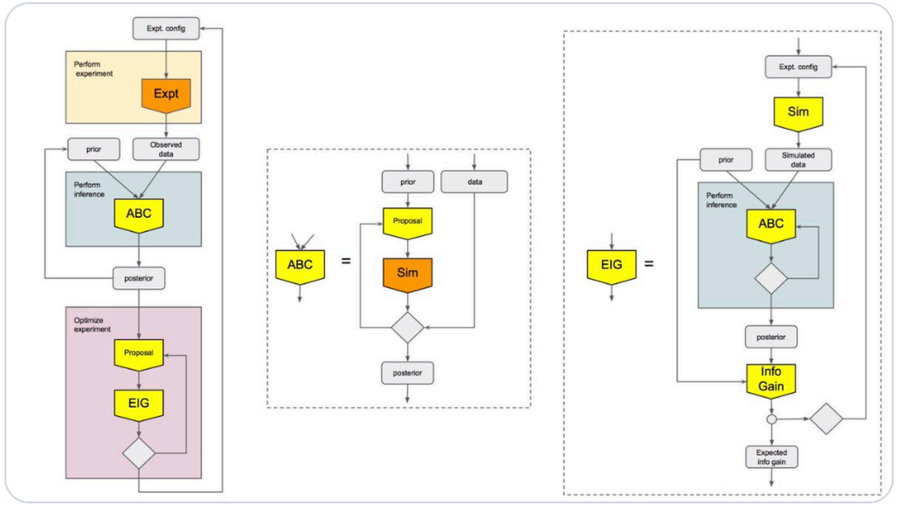


Figure 2: Measured forward-backward asymmetries of muon-pair production compared with the model independent fit results.

Kyle Cranmer @KyleCranmer · Jun 11, 2017

Demo for YComb research
active learning + workflows + implicit models = #ActiveSciencing
@lukasheinrich_ @glouppe
github.com/cranmer/active...



4 22 61

Danilo J. Rezende @DeepSpiker · Jul 19, 2017

This is great!

1 3

Kyle Cranmer @KyleCranmer · Jul 19, 2017

Thanks!!!

1 2

Danilo J. Rezende
@DeepSpiker

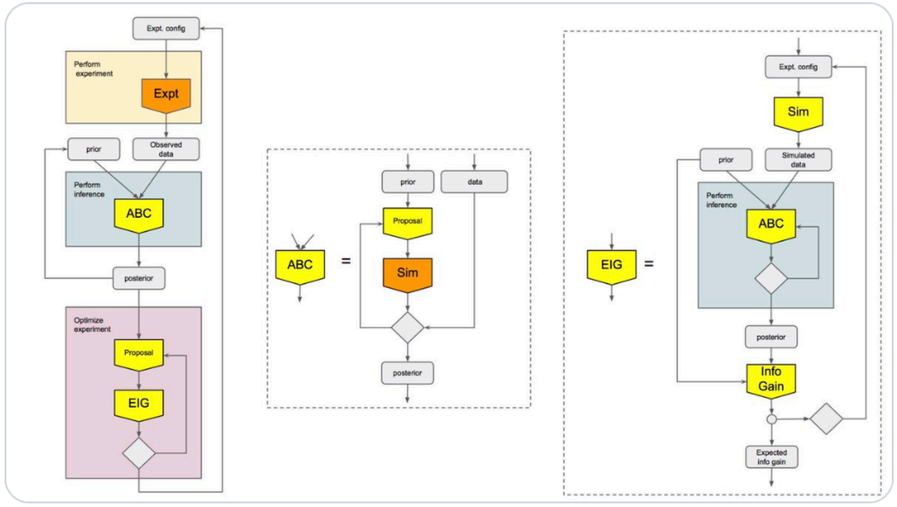
Replying to @KyleCranmer @lukasheinrich_ and @glouppe

You have the full loop of the scientific method in a python notebook :)

3:12 PM · Jul 19, 2017 · Twitter for iPhone

Kyle Cranmer @KyleCranmer · Jun 11, 2017

Demo for YComb research
active learning + workflows + implicit models = #ActiveSciencing
@lukasheinrich_ @glouppe
github.com/cranmer/active...



4 22 61

Danilo J. Rezende @DeepSpiker · Jul 19, 2017

This is great!

1 3

Kyle Cranmer @KyleCranmer · Jul 19, 2017

Thanks!!!

1 2

Danilo J. Rezende
@DeepSpiker

Replying to @KyleCranmer @lukasheinrich_ and @glouppe

You have the full loop of the scientific method in a python notebook :)

3:12 PM · Jul 19, 2017 · Twitter for iPhone

Reality check…

Keep in mind that
- the simulator model was specified
- the space of experimental configurations was well specified

Still it was hard enough!

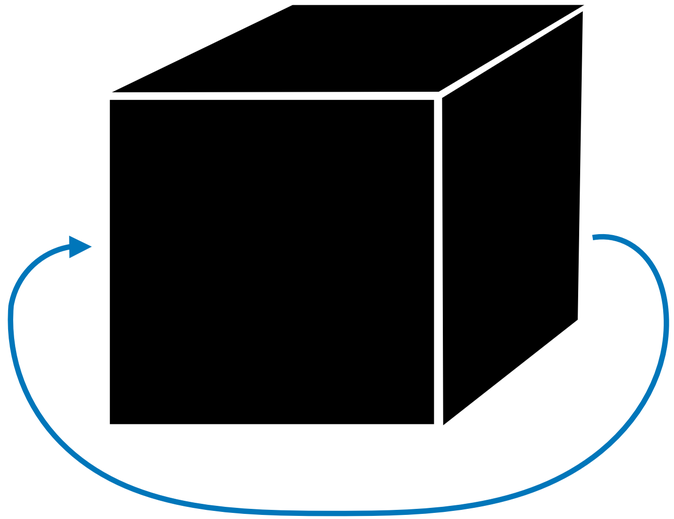
Going to open world of experimental configurations and potential models much harder.

Hypothesis generation also hard.

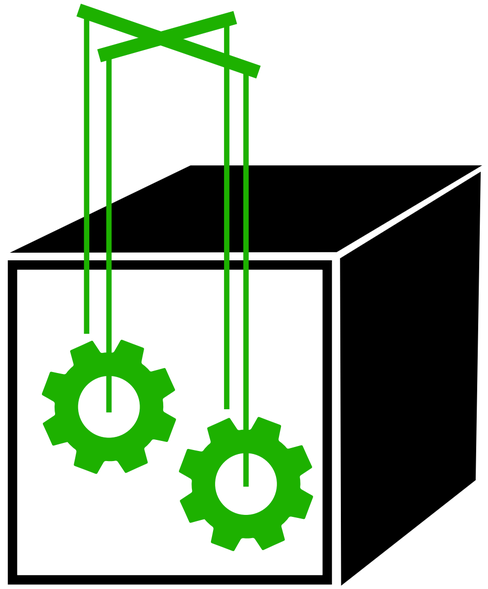
Frontiers of simulation-based inference

[K. Cranmer, JB, G. Louppe 1911.01429]

Mining gold:
Extract and leverage information from the simulator that characterizes the latent process



Active learning:
Iteratively guide simulator to important parameter regions based on past results

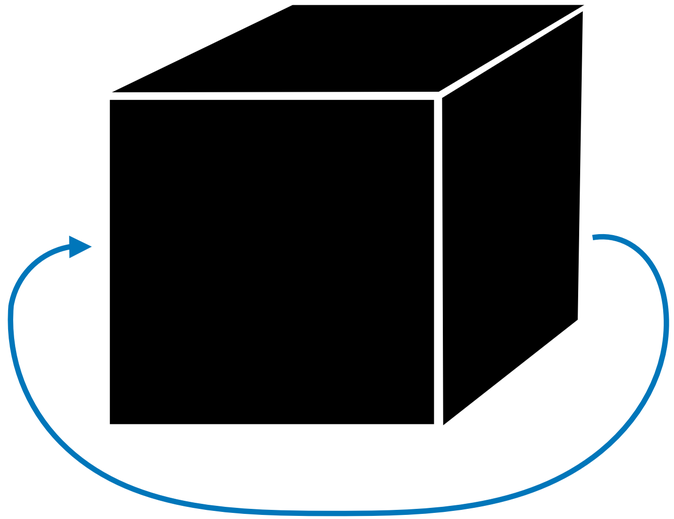


Probabilistic programming:
Explicitly write simulator as probabilistic program, with ability to condition execution trace on observations

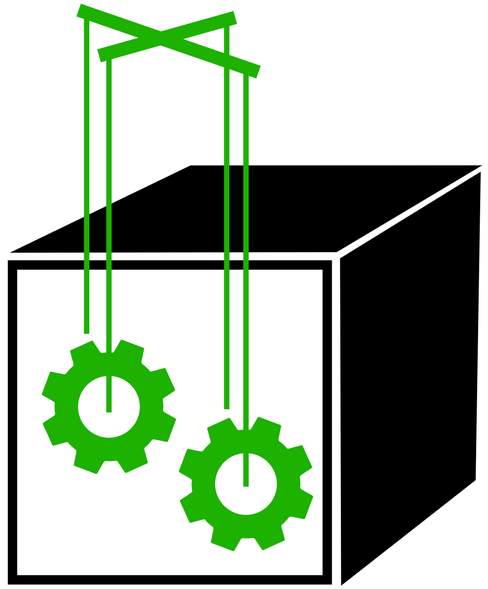
Frontiers of simulation-based inference

[K. Cranmer, JB, G. Louppe 1911.01429]

Mining gold:
Extract and leverage information from the simulator that characterizes the latent process



Active learning:
Iteratively guide simulator to important parameter regions based on past results



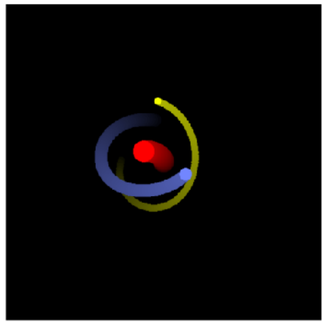
Probabilistic programming:
Explicitly write simulator as probabilistic program, with ability to condition execution trace on observations

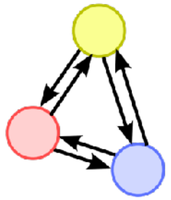
Other fun stuff I probably won’t cover

Insight of data generating process informs inductive bias on architecture

Physical systems as graphs

n-body

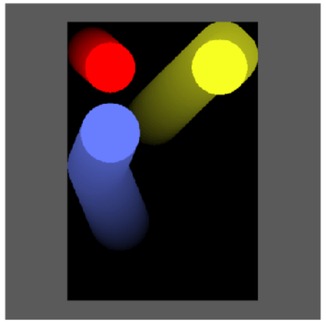


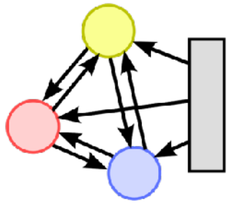


Nodes: bodies

Edges: gravitational forces

Balls

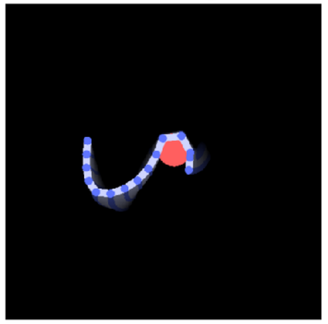


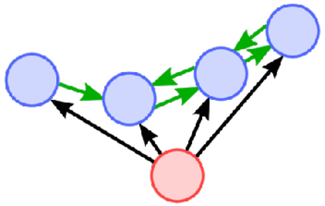


Nodes: balls

Edges: rigid collisions between balls, and walls

String





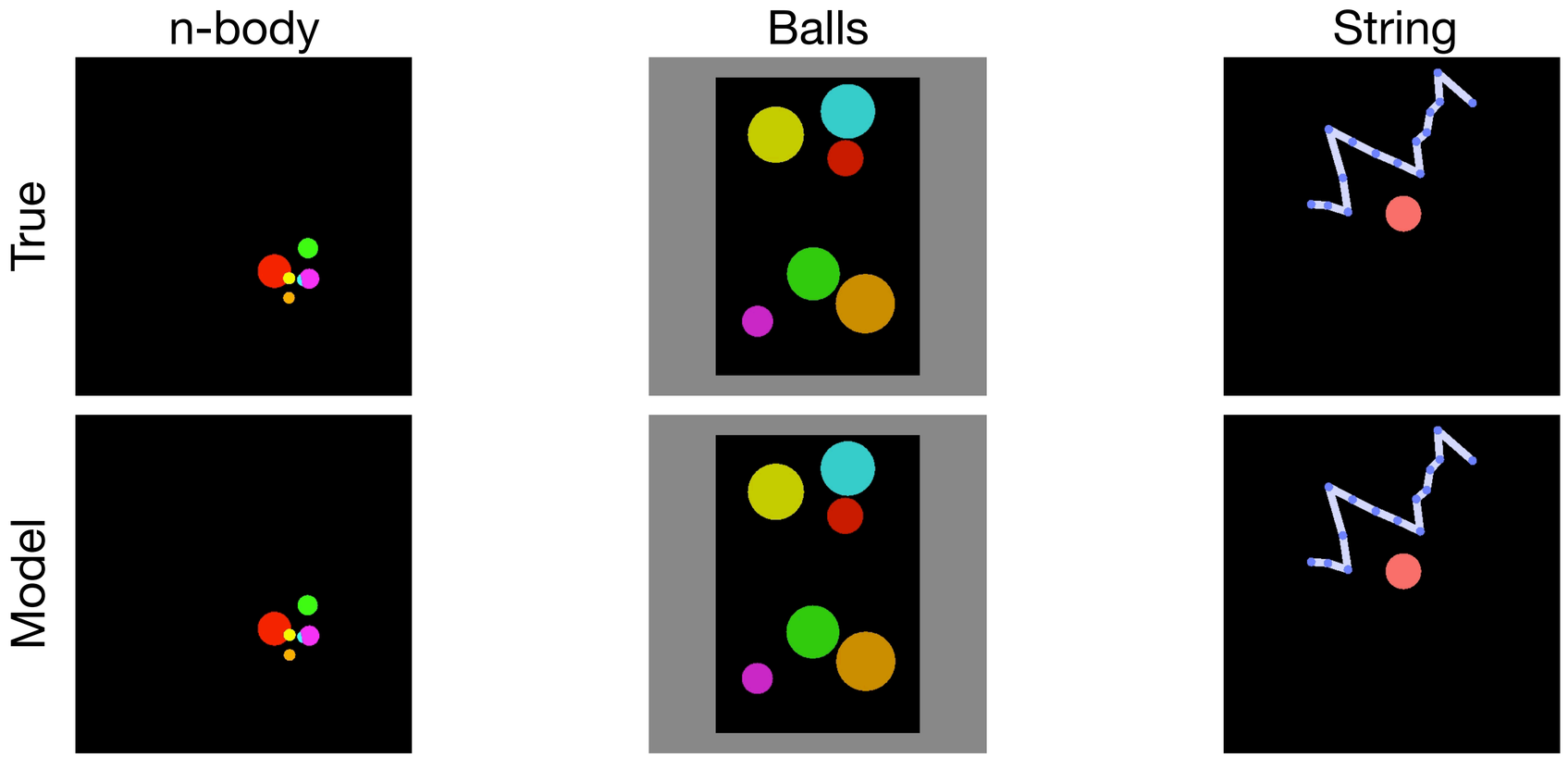
Nodes: masses

Edges: springs and rigid collisions

Battaglia et al., 2016, NeurIPS

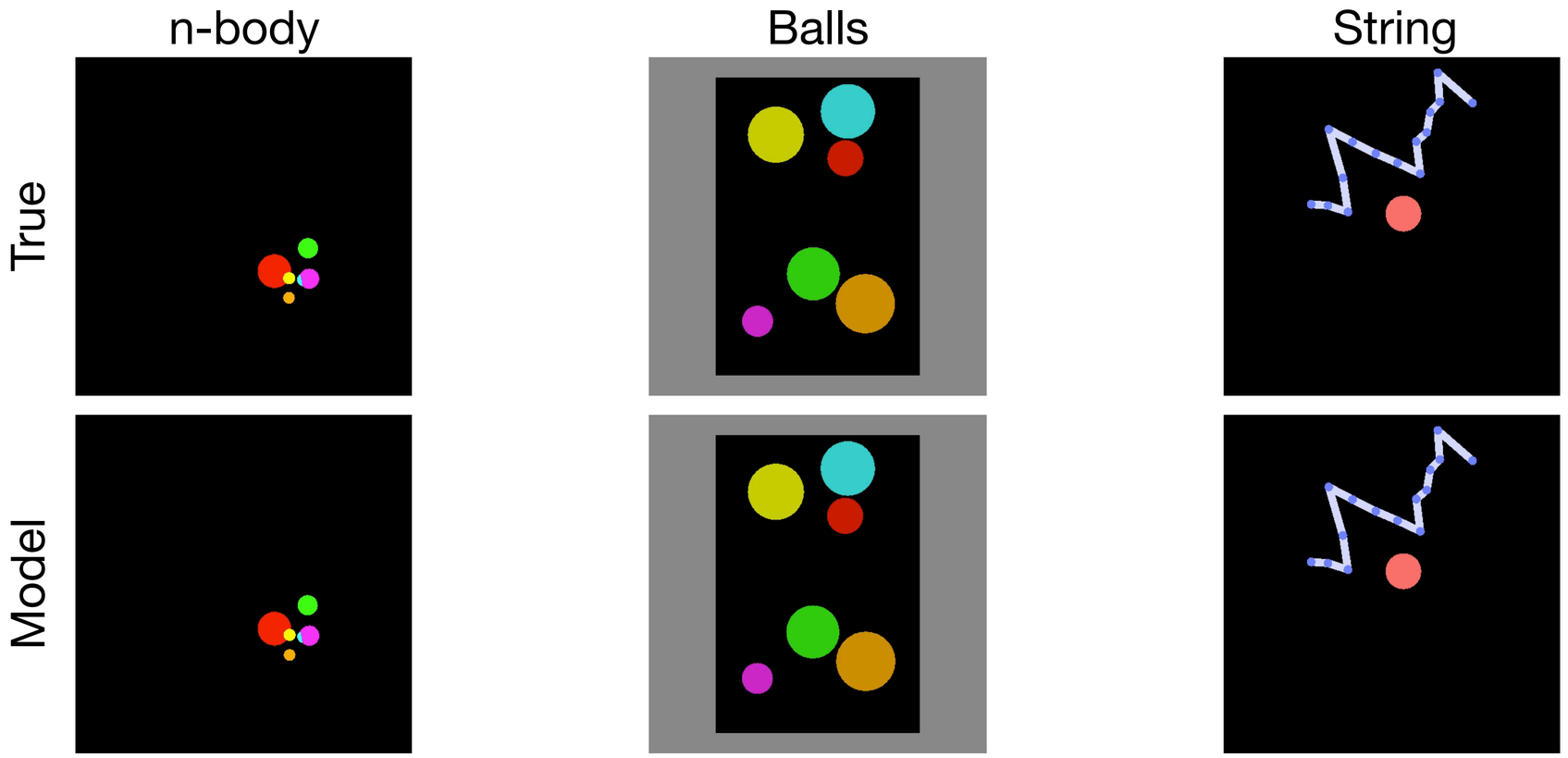
Slides borrowed with permission from Peter Battaglia based on Battaglia et al., 2016, NeurIPS

1000-step rollouts of true (top row) vs predicted (bottom row)



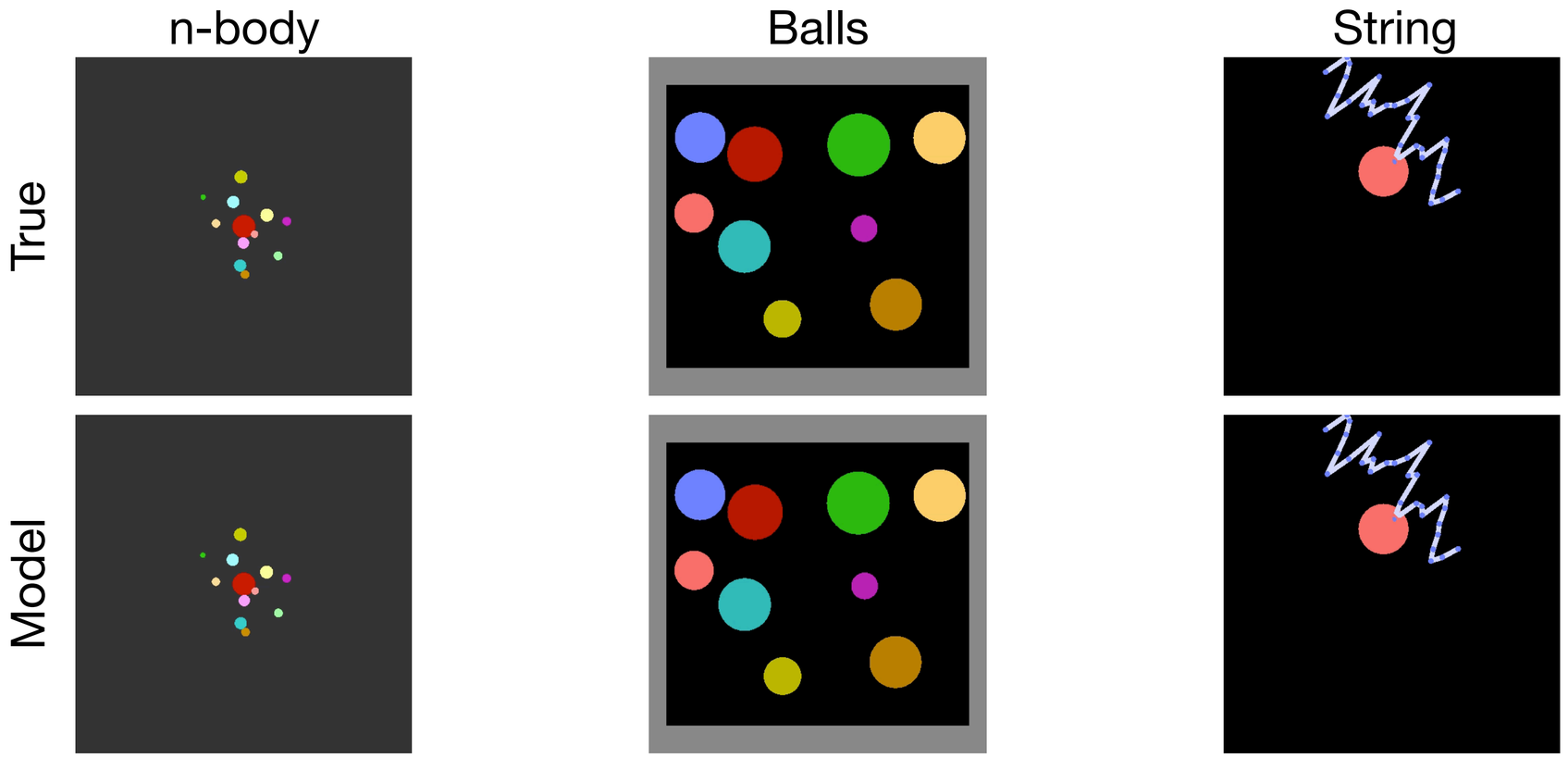
Slides borrowed with permission from Peter Battaglia based on Battaglia et al., 2016, NeurIPS

1000-step rollouts of true (top row) vs predicted (bottom row)



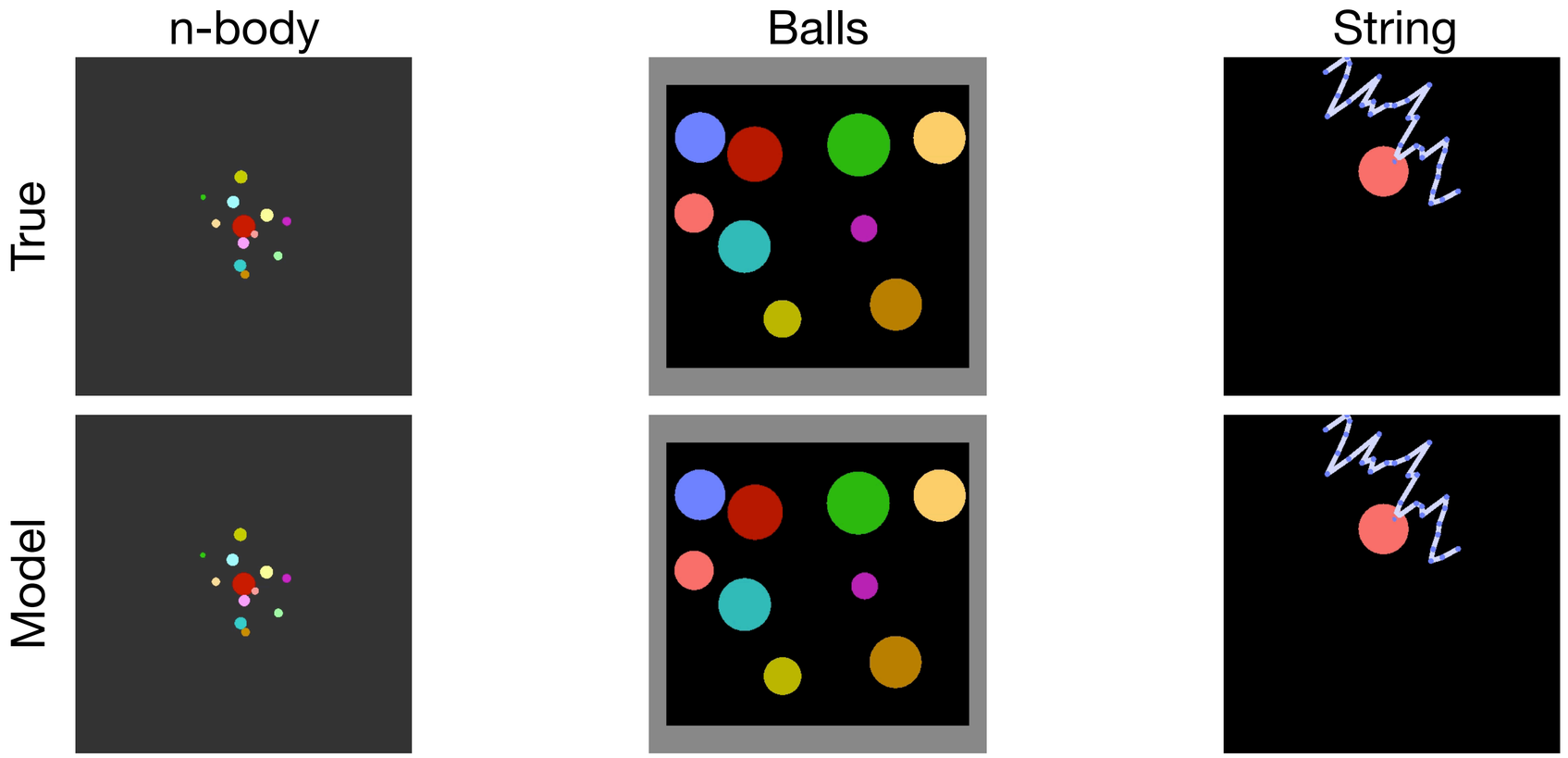
Slides borrowed with permission from Peter Battaglia based on Battaglia et al., 2016, NeurIPS

Zero shot generalisation to larger systems



Slides borrowed with permission from Peter Battaglia based on Battaglia et al., 2016, NeurIPS

Zero shot generalisation to larger systems



NEW!

We incorporated two physically informed inductive biases

- ODE integrators
- Hamiltonian mechanics

into graph networks for learning simulation, and found they could improve performance, energy, and zero-shot time-step generalization.

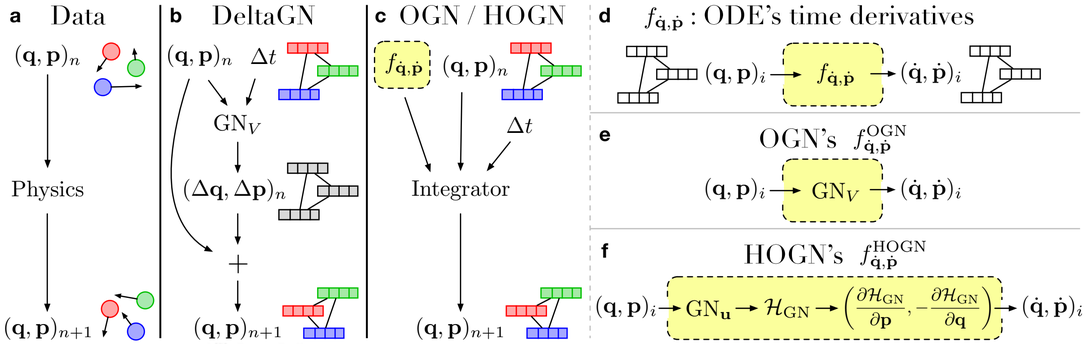
Hamiltonian Graph Networks with ODE Integrators

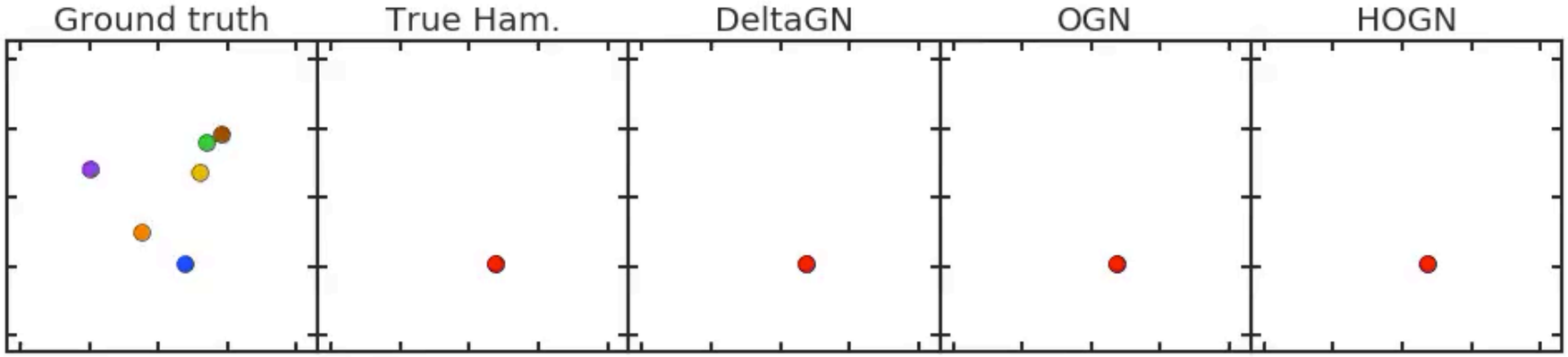
Alvaro Sanchez-Gonzalez
DeepMind
London, UK
alvarosg@google.com

Victor Bapst
DeepMind
London, UK
vbapst@google.com

Kyle Cranmer
NYU
New York, USA
kc90@nyu.edu

Peter Battaglia
DeepMind
London, UK
peterbattaglia@google.com





NEW!

We incorporated two physically informed inductive biases

- ODE integrators
- Hamiltonian mechanics

into graph networks for learning simulation, and found they could improve performance, energy, and zero-shot time-step generalization.

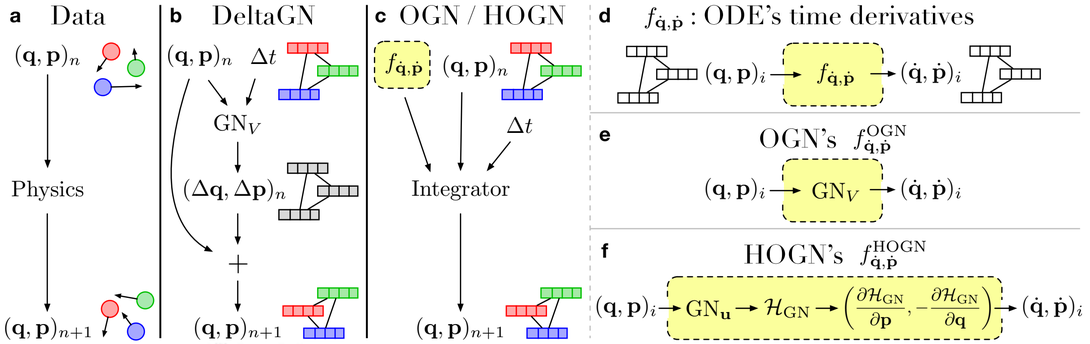
Hamiltonian Graph Networks with ODE Integrators

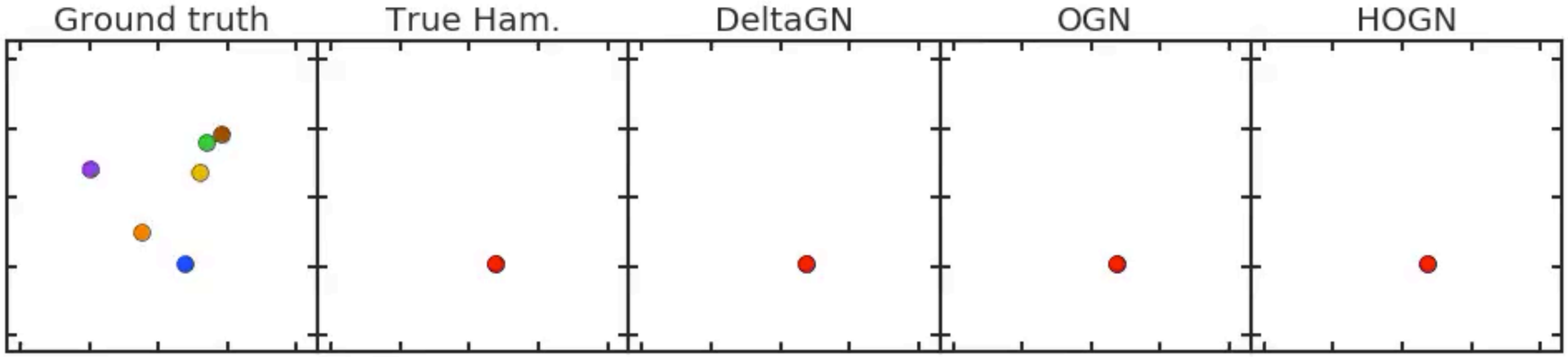
Alvaro Sanchez-Gonzalez
DeepMind
London, UK
alvarosg@google.com

Victor Bapst
DeepMind
London, UK
vbapst@google.com

Kyle Cranmer
NYU
New York, USA
kc90@nyu.edu

Peter Battaglia
DeepMind
London, UK
peterbattaglia@google.com





Generative Models:
Density Estimation & Unsupervised Learning

THE REVOLUTION WILL NOT BE SUPERVISED

Supervised

- Training data comes in input-output pairs $\{(x_i, y_i)\}$ and task is to learn f(x) such that y=f(x)+noise w.r.t. some loss function
- Function approximation, “glorified curve fitting”
- Examples: classification and regression

Unsupervised

- Training data only $\{x_i\}$, does not have labels y_i
- Examples: density estimation, clustering

HISTOGRAM $\{x_i\} \rightarrow \hat{p}(x)$

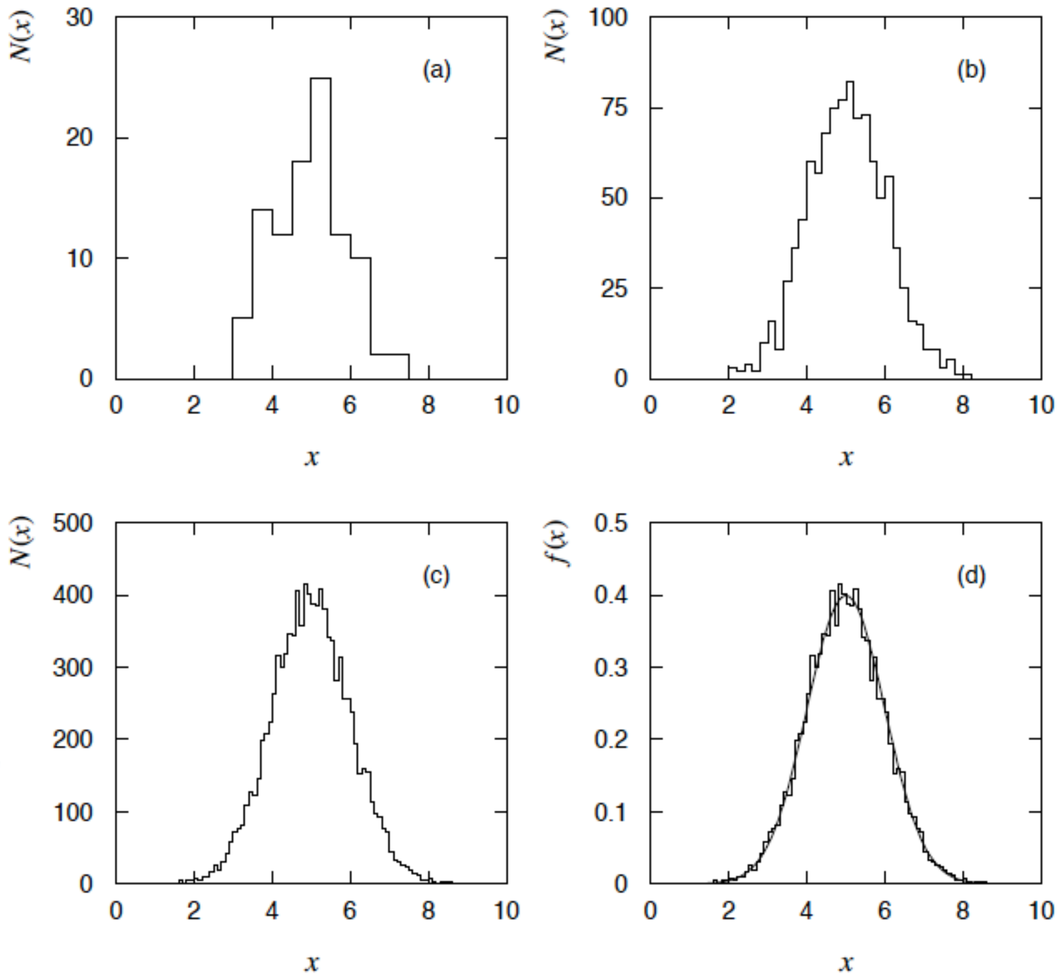
Given a set of observations $\{x_i\}$ we can approximate the pdf with a histogram.

The histogram approximates the unknown density $\hat{p}(x) \approx p(x)$:

Density estimation task,
but no "learning" here

For unsupervised learning
Need:

- a model to fit
- a loss function to optimize



What function r(x) minimizes the **cross-entropy** loss?

$$L[r] = -\int \underbrace{p(x)\log r(x)}_{F(x,r)}\, dx$$

- Subject to $\int r(x)dx = 1$

What function r(x) minimizes the **cross-entropy** loss?

$$L[r] = -\int \underbrace{p(x)\log r(x)}_{F(x,r)}\, dx \;\approx\; \frac{1}{N}\sum_{i=1}^{N} \log r(x_i)$$

- Subject to $\int r(x)dx = 1$

What function r(x) minimizes the **cross-entropy** loss?

$$L[r] = -\int \underbrace{p(x)\log r(x)}_{F(x,r)} dx \approx \frac{1}{N}\sum_{i=1}^{N} \log r(x_i)$$

- Subject to $\int r(x)dx = 1$

Euler-Lagrange Equation w/ Lagrange-multiplier

$$L[r,\lambda] = F(x,r) + \lambda r(x)$$

$$\underbrace{\frac{d}{dx}\left(\frac{\delta L}{\delta r'}\right)}_{=0} - \frac{\delta L}{\delta r} = 0 \qquad \frac{\delta L}{\delta r} = 0 = \frac{-p(x)}{r(x)} + \lambda$$

$$r(x) = p(x)/\lambda$$

imposing the constraint gives $\lambda = 1$ thus $r(x) = p(x)$

OTHER LOSS FUNCTIONS

The empirical **cross-entropy** loss is like Maximum Likelihood

$$L[r] = -\int \underbrace{p(x)\log r(x)}_{F(x,r)}\, dx \approx \frac{1}{N}\sum_{i=1}^{N} \log r(x_i)$$

Relationship to KL Divergence (aka relative entropy):

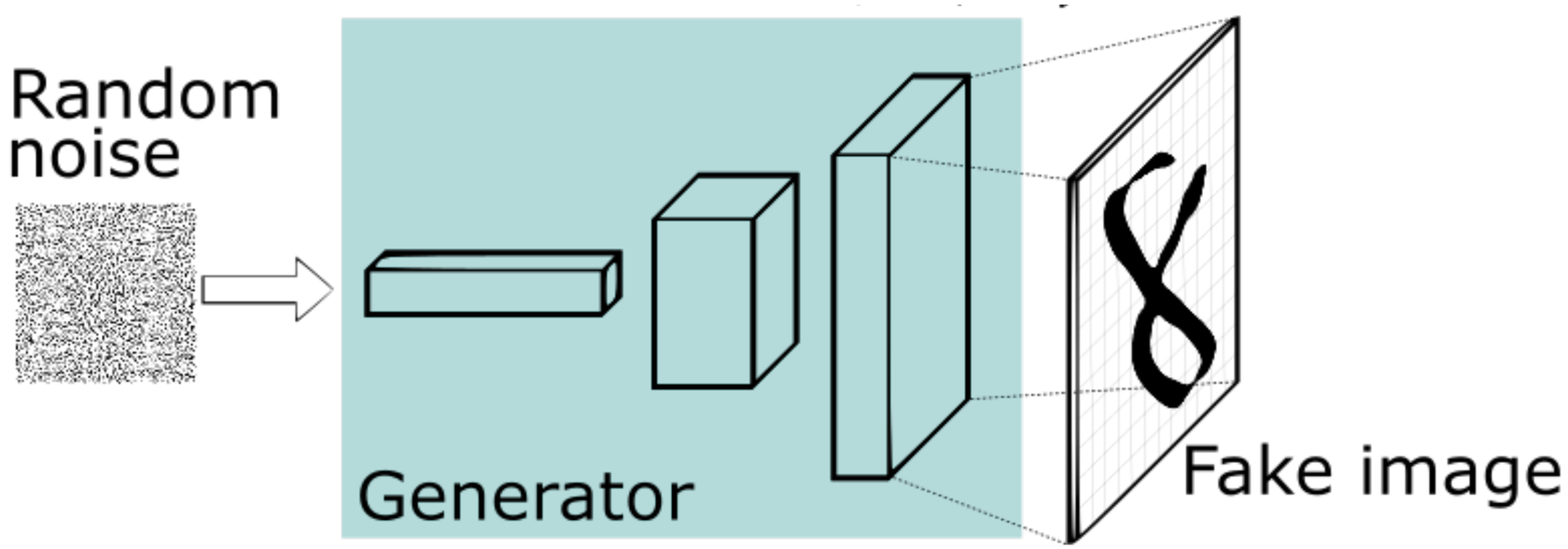
- Not symmetric, not a distance measure
- Maximum Likelihood KL[p || r] = E_p[log p - log r]
- Variational Inference KL[r || p] = E_r[log r - log p]
  - Same as free energy if p is Boltzman distribution
  - log p(x) = -H(x) or -S(x) & log Z term (constant wrt r)

How do we model distributions ?

ONE APPROACH

Use Neural Network as a map f:Z→ X

- Z is a latent space with some fixed distribution for p(Z)
- Have and feed forward through network X=f(Z)



Z f =network X = f(Z)

Change of variables

If *f(x)* is the pdf for *x* and *y(x)* is a change of variables, then the pdf *g(y)* must satisfy

$$P(x_a < x < x_b) \equiv \int_{x_a}^{x_b} f(x)dx = \int_{y(x_a)}^{y(x_b)} g(y)dy \equiv P(y(x_a) < y < y(x_b))$$

We can rewrite the integral on the right

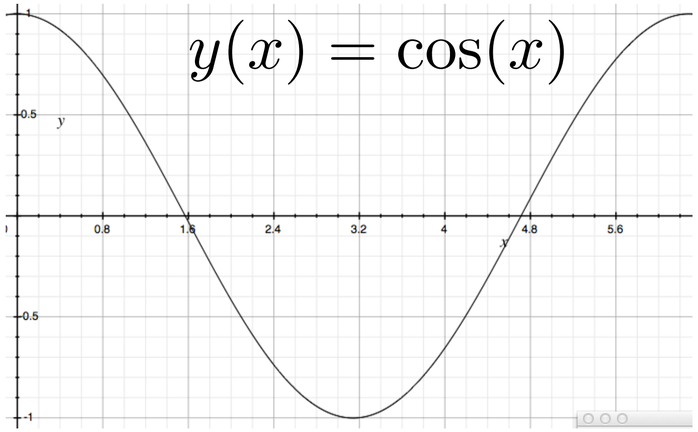
$$\int_{y(x_a)}^{y(x_b)} g(y)dy = \int_{x_a}^{x_b} g(y(x)) \left|\frac{dy}{dx}\right| dx$$

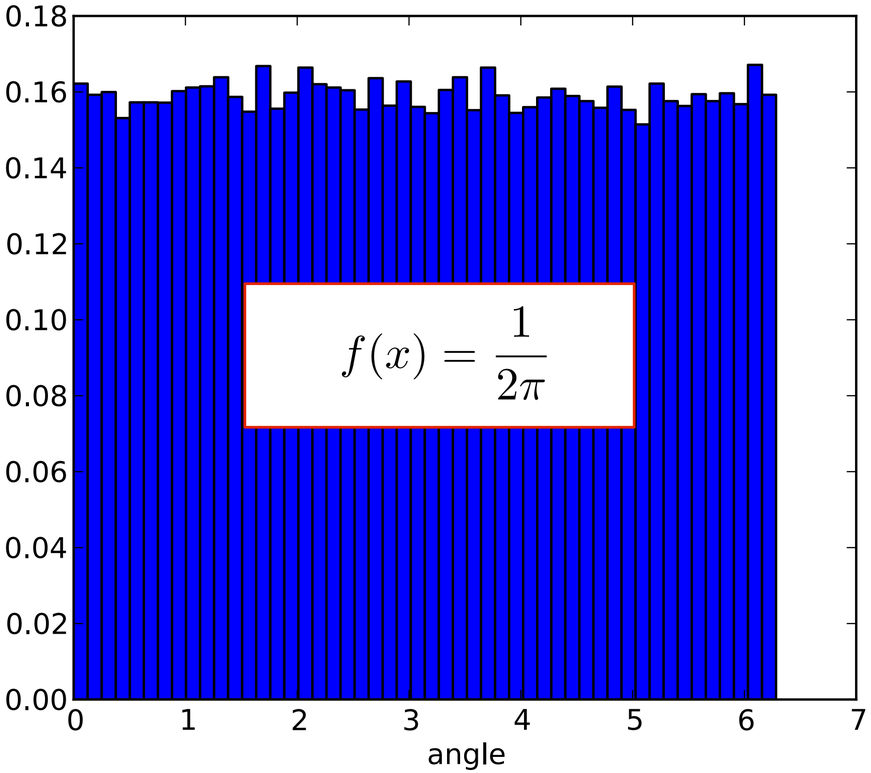
therefore, the two pdfs are related by a Jacobian factor

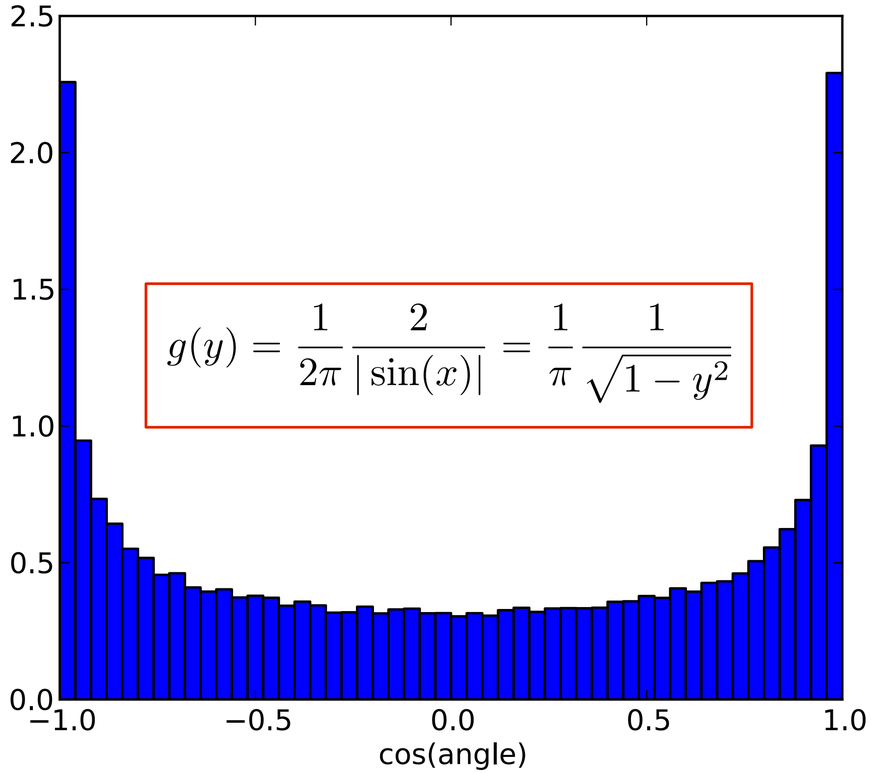
$$f(x) = g(y) \left|\frac{dy}{dx}\right|$$

An example

$$f(x) = g(y)\left|\frac{dy}{dx}\right|$$

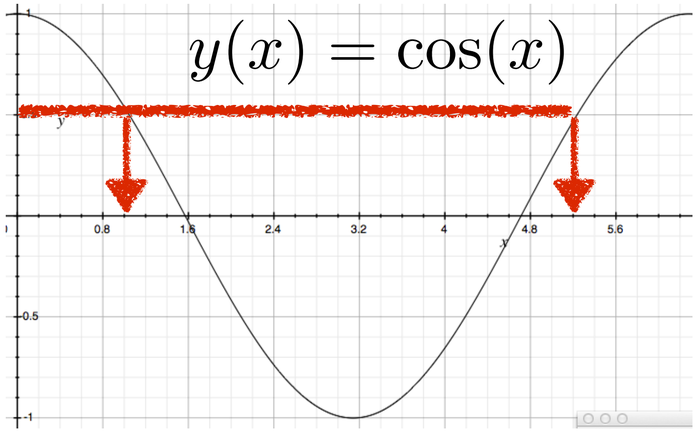


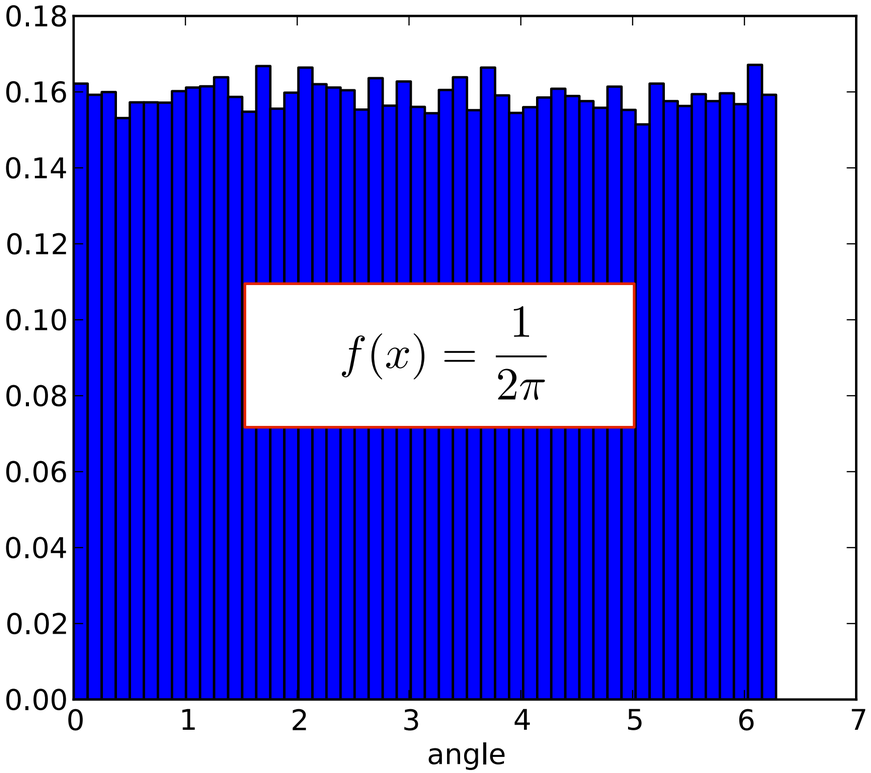


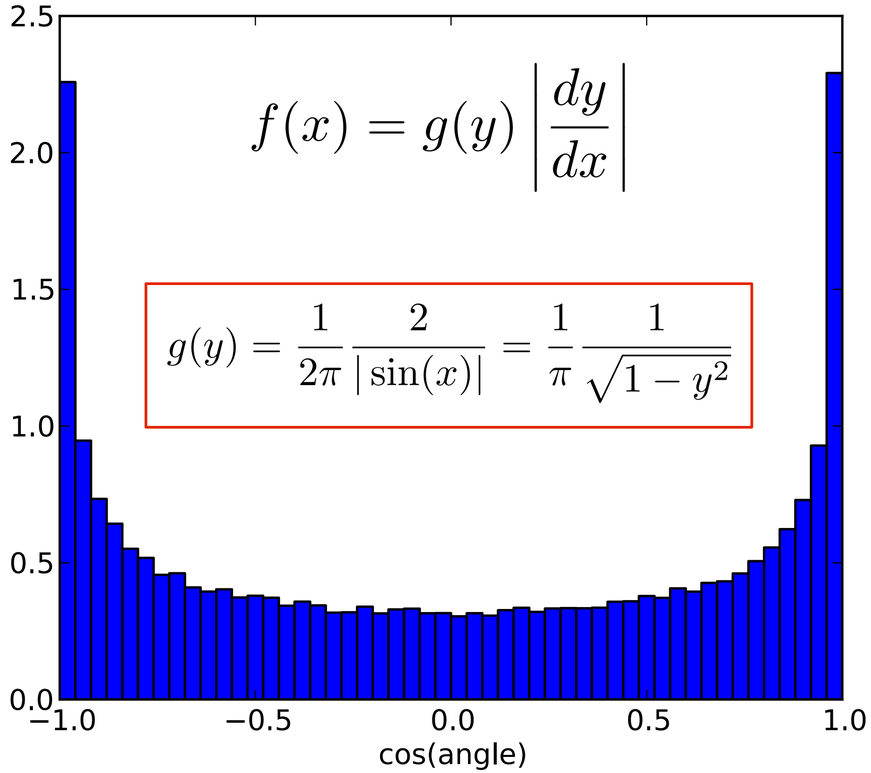


An example

I am glossing over the fact that the map is not 1-to-1. Different values of x, map into same value of y. We will need to sum/ integrate over them. Here it is easy, but in general this may become intractable… need inverse map







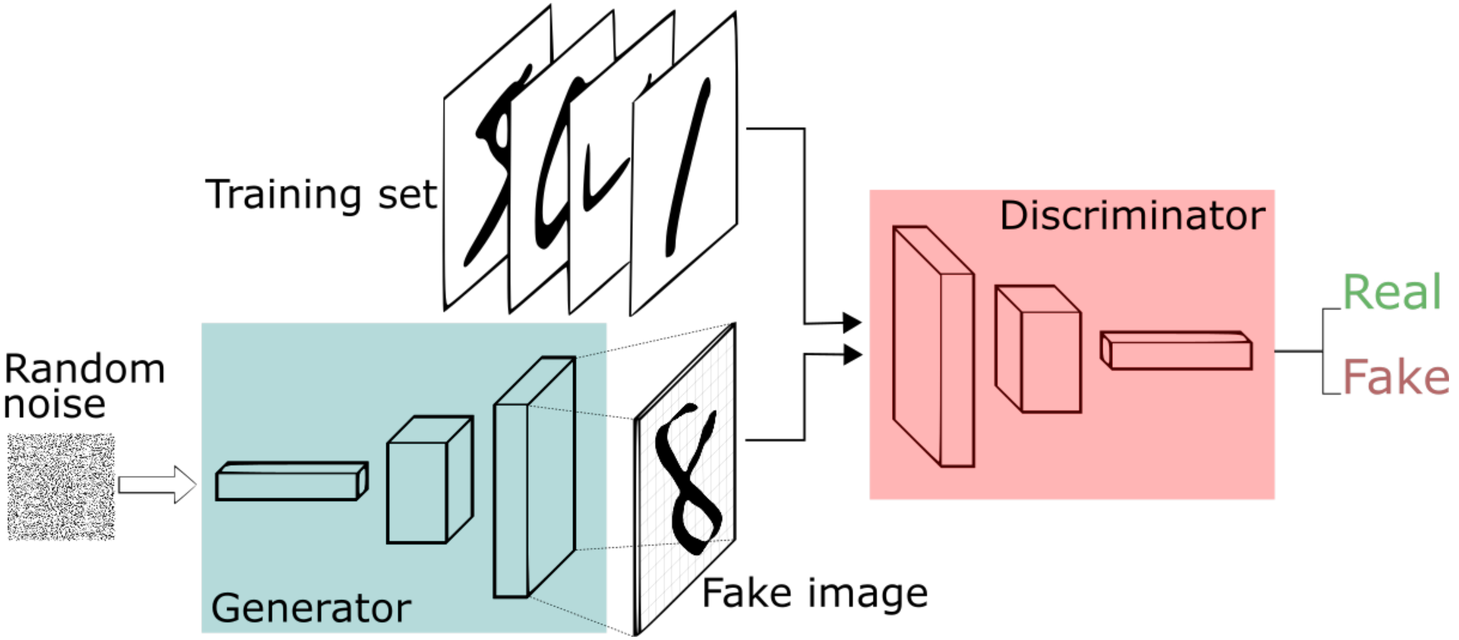
Problem:

Generic Feed Forward network is not invertible

Density p(x=f(z)) is intractable

Can’t train with maximum likelihood or cross-entropy

GENERATIVE ADVERSARIAL NETWORKS



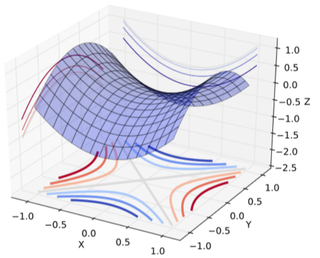
Leo is G Tom is D

$$V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

- We want to
  - For fixed G, find D which maximizes $V(D, G)$,
  - For fixed D, find G which minimizes $V(D, G)$;
- In other words, we are looking for the *saddle point*

$$(D^*, G^*) = \max_D \min_G V(D, G).$$

$$D_G^*(\boldsymbol{x}) = \frac{p_{data}(\boldsymbol{x})}{p_{data}(\boldsymbol{x}) + p_g(\boldsymbol{x})}$$



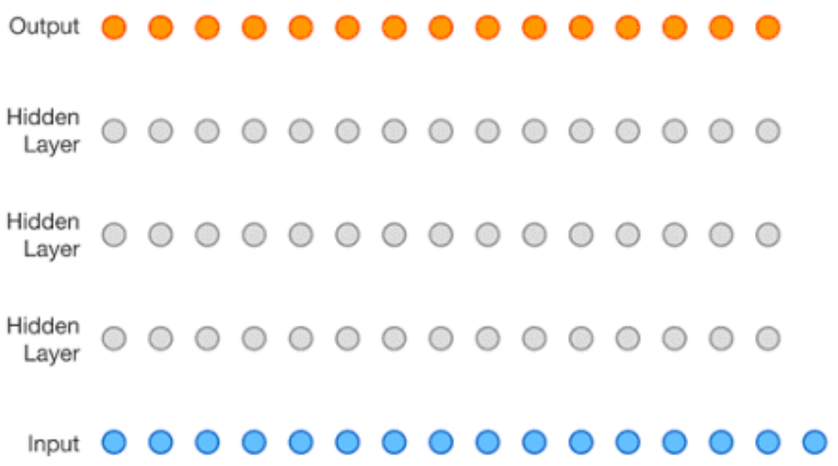
Alternatives to GANs

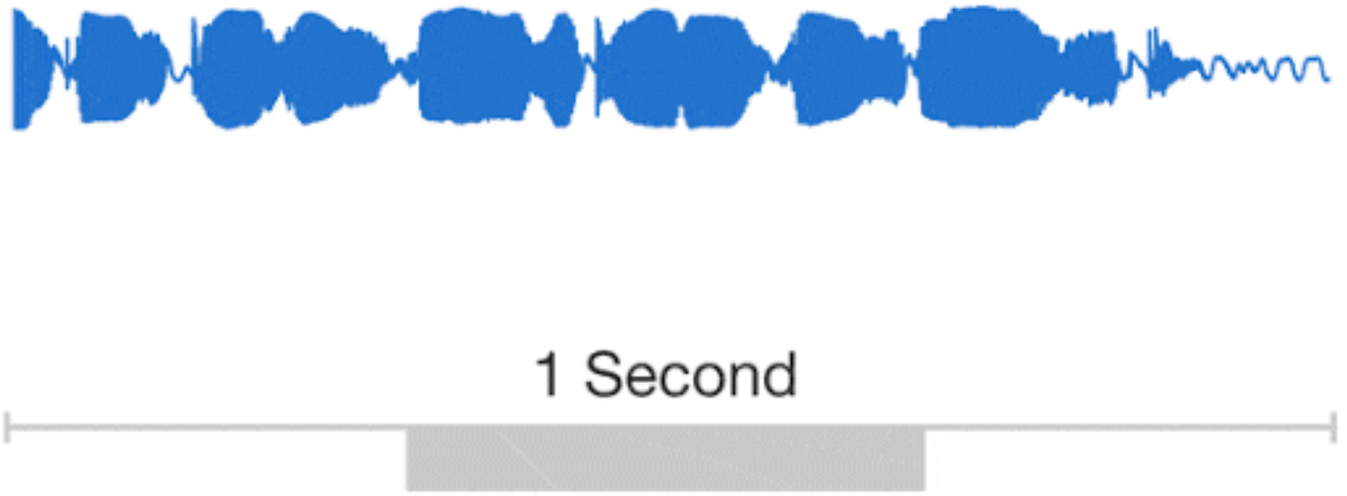
WAVENET: A GENERATIVE MODEL FOR RAW AUDIO

Autoregressive models defined by

$$p(x) = \prod_{t=1}^{T} p(x_t \mid x_{t-1}, \ldots, x_1).$$

have a tractable density. Train via maximum likelihood



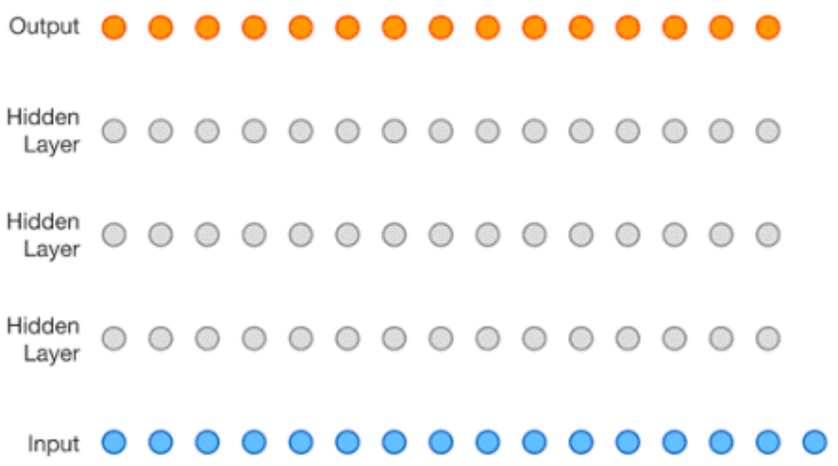


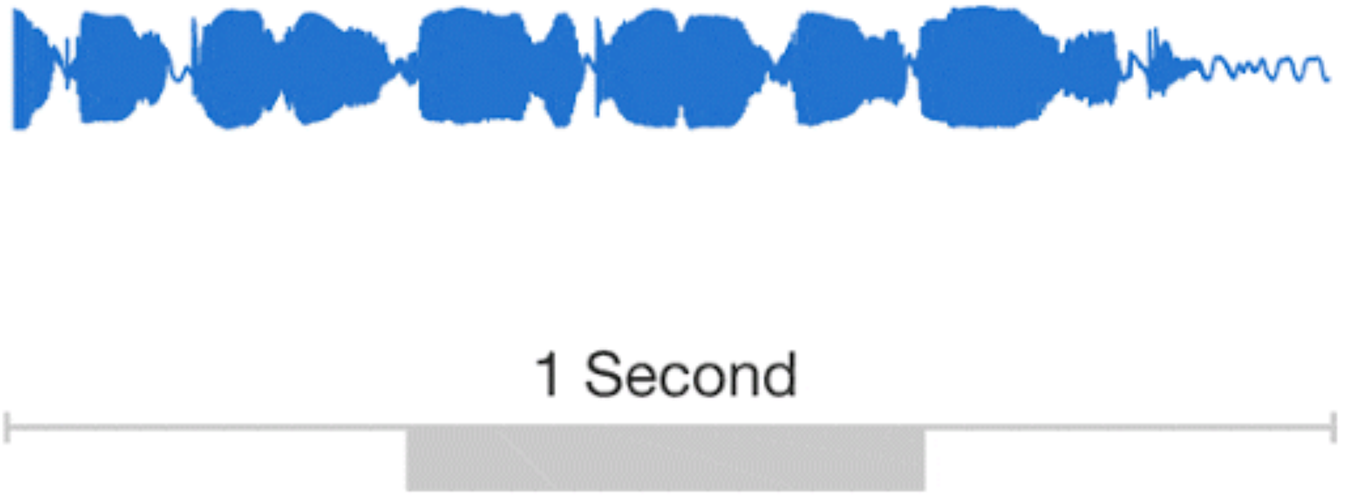
WAVENET: A GENERATIVE MODEL FOR RAW AUDIO

Autoregressive models defined by

$$p(x) = \prod_{t=1}^{T} p(x_t \mid x_{t-1}, \ldots, x_1).$$

have a tractable density. Train via maximum likelihood



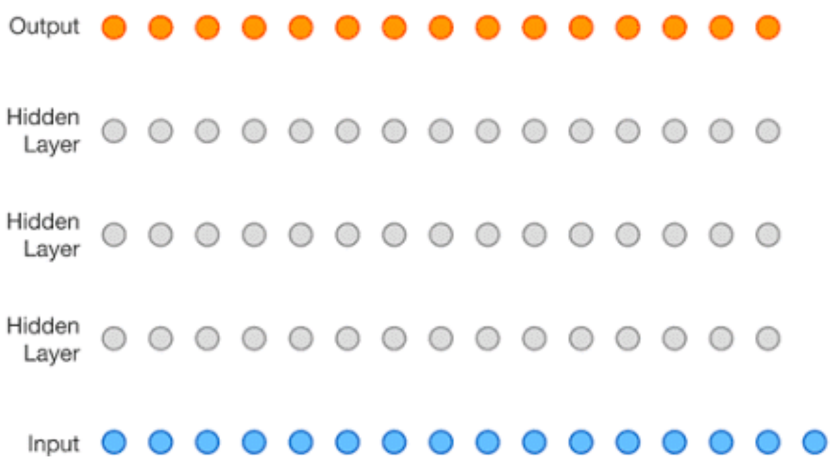


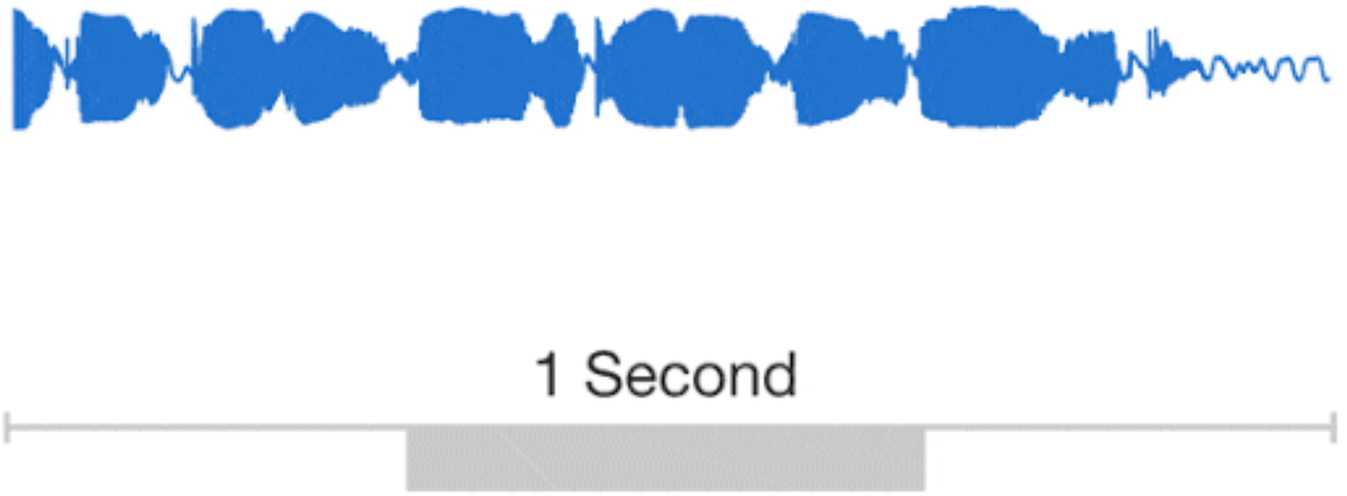
WAVENET: A GENERATIVE MODEL FOR RAW AUDIO

Autoregressive models defined by

$$p(x) = \prod_{t=1}^{T} p(x_t \mid x_{t-1}, \ldots, x_1).$$

have a tractable density. Train via maximum likelihood



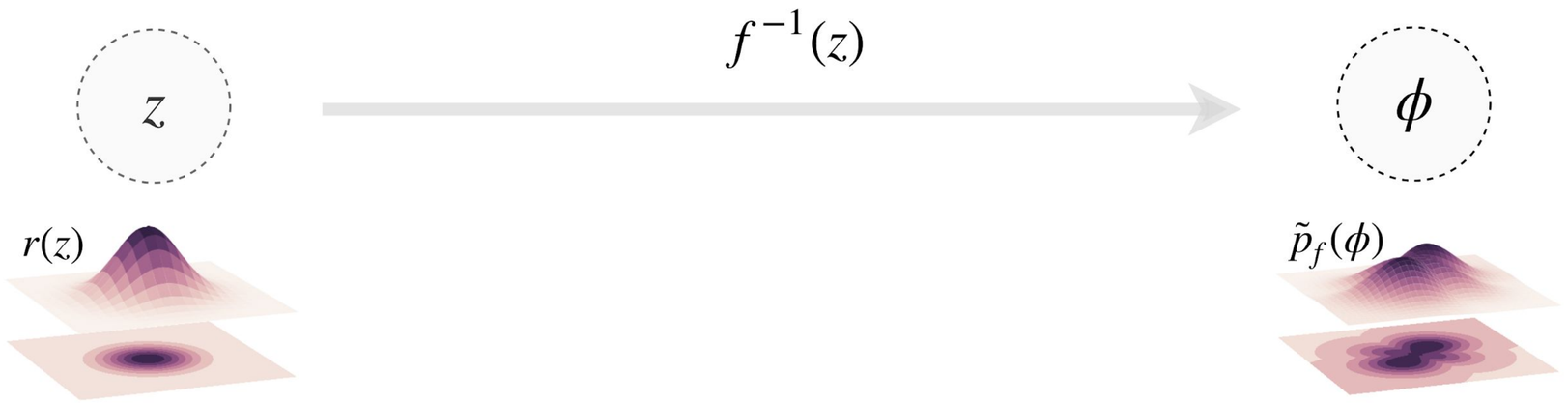


K.C. & G. Louppe: http://beta.briefideas.org/ideas/5c2f74aedbf3618ca180382e393c7617

Flow-based generative models

Using a change-of-variables, produce a distribution approximating what you want.

[Rezende & Mohamed 1505.05770]

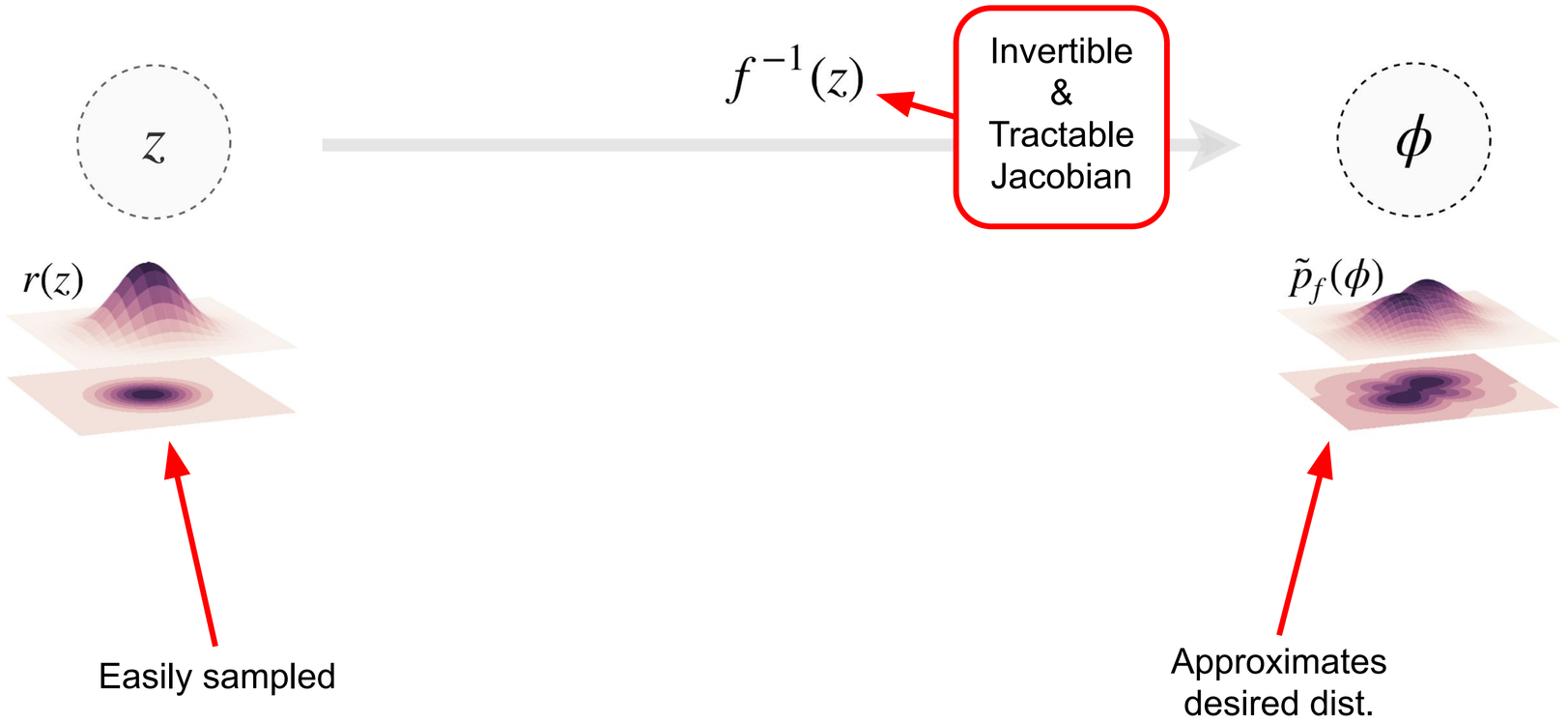


Flow-based generative models

Using a change-of-variables, produce a distribution approximating what you want.

[Rezende & Mohamed 1505.05770]

$$\tilde{p}_f(\phi) = \left| \det \frac{\partial f^{-1}(z)}{\partial z} \right|^{-1} r(z)$$



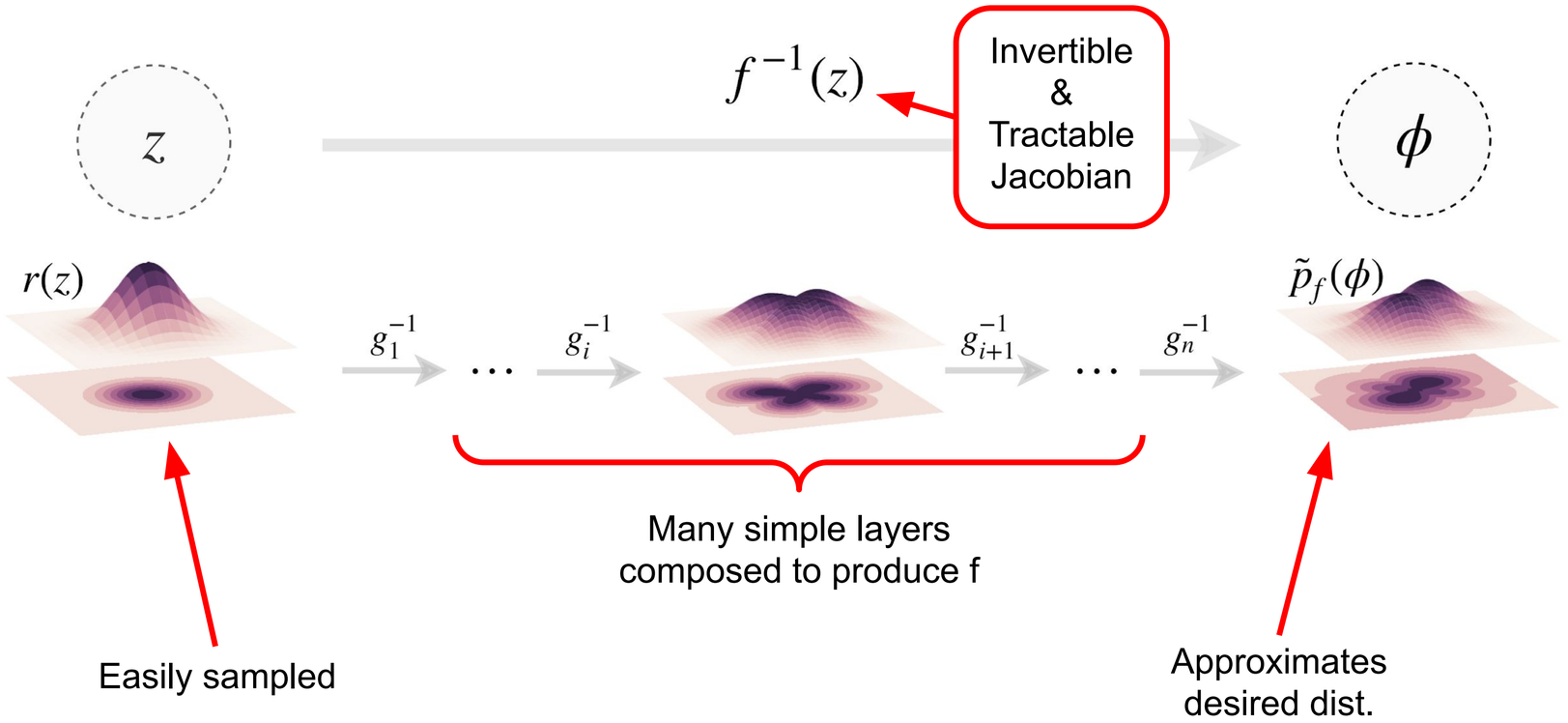
Slides from Tej Kanwar, MIT

Flow-based generative models

We chose real non-volume preserving (real NVP) flows for our work.

[Dinh et al. 1605.08803]

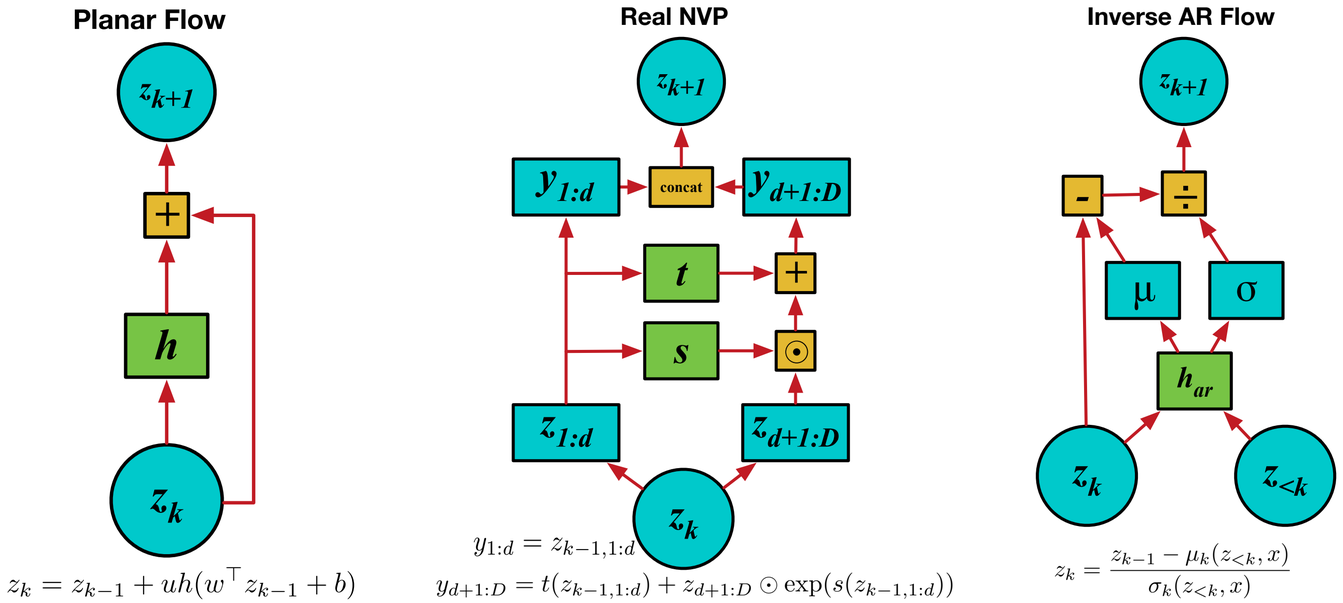
$$\tilde{p}_f(\phi) = \left| \det \frac{\partial f^{-1}(z)}{\partial z} \right|^{-1} r(z)$$



Slides from Tej Kanwar, MIT

FLOWS / BIJECTIONS

- Begin with a fully-factorised Gaussian and improve by change of variables.
- Triangular Jacobians allow for computational efficiency.



[Rezende and Mohamed, 2016; Dinh et al., 2016; Kingma et al., 2016]

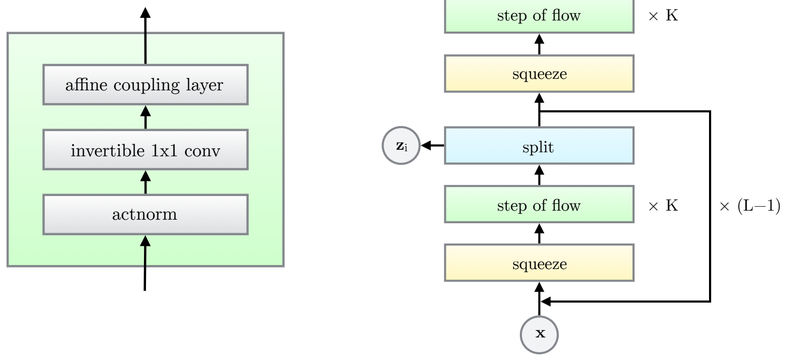
Linear time computation of the determinant and its gradient.

FLOWS WITH INVERTIBLE CONVOLUTIONS

https://arxiv.org/abs/1807.03039

Glow: Generative Flow with Invertible 1×1 Convolutions

Diederik P. Kingma*, **Prafulla Dhariwal***
OpenAI, San Francisco



(a) One step of our flow.

(b) Multi-scale architecture (Dinh et al., 2016).

https://arxiv.org/abs/1901.11137

Emerging Convolutions for Generative Normalizing Flows

Emiel Hoogeboom [1 2] **Rianne van den Berg** [1] **Max Welling** [1 3]

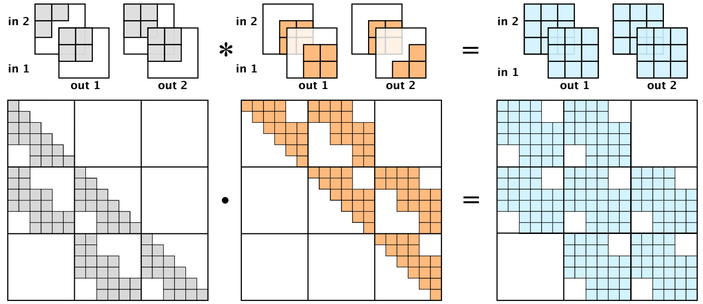


Table 3. Performance of Emerging convolutions on CIFAR10, ImageNet 32x32 and ImageNet 64x64 in bits per dimension (negative $\log_2$-likelihood), and ± reports standard deviation.

	CIFAR10	ImageNet 32x32	ImageNet 64x64
Real NVP	3.51	4.28	3.98
Glow	3.36 ±0.002	**4.09**	**3.81**
Emerging	**3.34** ±0.002	**4.09**	**3.81**

FLOWS WITH CONTINUOUS TIME

FFJORD: Free-form Continuous Dynamics for Scalable Reversible Generative Models

Will Grathwohl[*†‡], **Ricky T. Q. Chen**[*†], **Jesse Bettencourt**[†], **Ilya Sutskever**[‡], **David Duvenaud**[†]

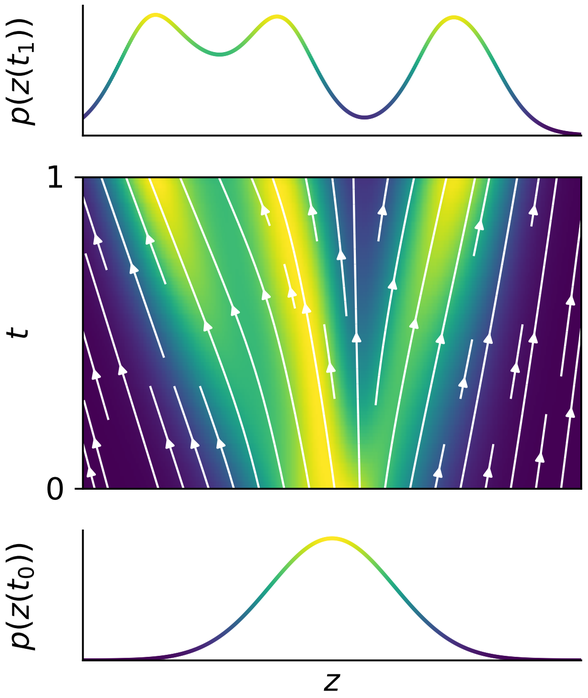
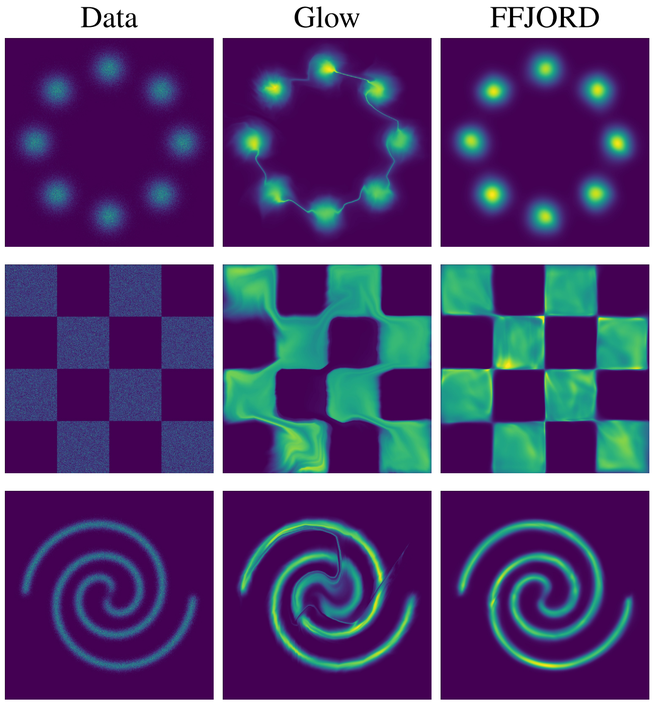


Figure 1: FFJORD transforms a simple base distribution at t_0 into the target distribution at t_1 by integrating over learned continuous dynamics.



	Method	Train on data	One-pass Sampling	Exact log-likelihood	Free-form Jacobian
	Variational Autoencoders	✓	✓	✗	✓
	Generative Adversarial Nets	✓	✓	✗	✓
	Likelihood-based Autoregressive	✓	✗	✓	✗
Change of Variables	Normalizing Flows	✗	✓	✓	✗
Change of Variables	Reverse-NF, MAF, TAN	✓	✗	✓	✗
Change of Variables	NICE, Real NVP, Glow, Planar CNF	✓	✓	✓	✗
Change of Variables	**FFJORD**	✓	✓	✓	✓

Table 1: A comparison of the abilities of generative modeling approaches.

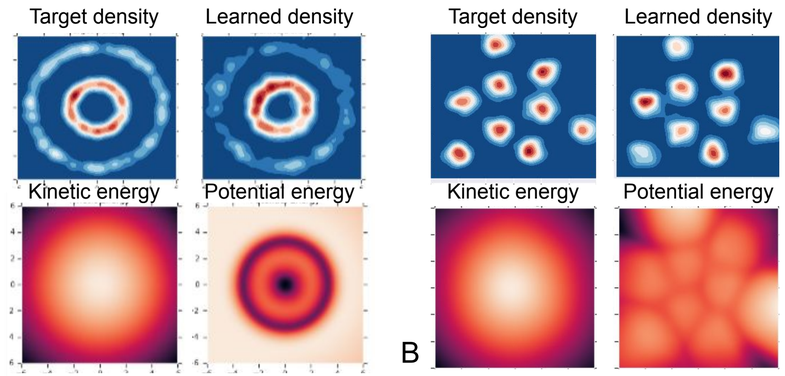
Equivariant Hamiltonian Flows

Danilo J. Rezende* **Sébastien Racanière*** **Irina Higgins*** **Peter Toth***

***{danilor, sracaniere, irinah, petertoth}@google.com**

Abstract

This paper introduces equivariant hamiltonian flows, a method for learning expressive densities that are invariant with respect to a known Lie-algebra of local symmetry transformations while providing an equivariant representation of the data. We provide proof of principle demonstrations of how such flows can be learnt, as well as how the addition of symmetry invariance constraints can improve data efficiency and generalisation. Finally, we make connections to disentangled representation learning and show how this work relates to a recently proposed definition.



Equivariant Flows: sampling configurations for multi-body systems with symmetric energies

Jonas Köhler *† **Leon Klein** *† **Frank Noé** †‡§

† Freie Universität Berlin, Department of Mathematics and Computer Science.
‡ Freie Universität Berlin, Department of Physics.
§ Rice University, Department of Chemistry.
{jonas.koehler, leon.klein, frank.noe}@fu-berlin.de
* Authors contributed equally to this work.

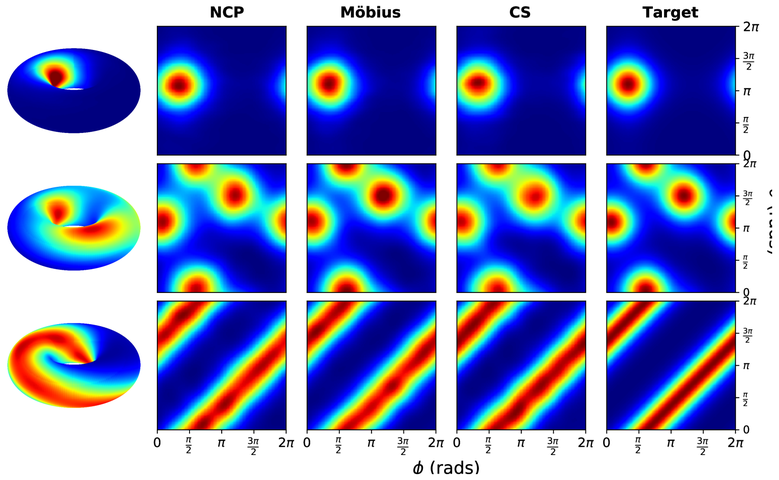
1. Permutation invariance: Swapping the labels of any two interchangeable particles $(x_1, \dots, x_K) \to (x_{\sigma(1)}, \dots, x_{\sigma(K)})$.
2. Rotation invariance: Any 2D/3D rotation of the system $Rx = (Rx_1, \dots, Rx_K)$.
3. Translation invariance: Any 2D/3D translation $x + v = (x_1 + v, \dots, x_K + v)$.

FLOWS ON MANIFOLDS

Normalizing Flows on Tori and Spheres

Danilo Jimenez Rezende [*1] **George Papamakarios** [*1] **Sébastien Racanière** [*1] **Michael S. Albergo** [2]
Gurtej Kanwar [3] **Phiala E. Shanahan** [3] **Kyle Cranmer** [2]

[arXiv:2002.02428]



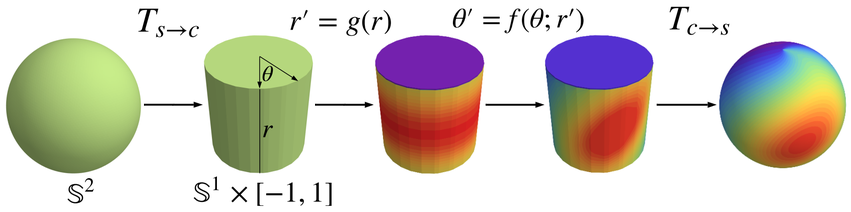


Figure 1. Illustration of the recursive flow on the sphere $\mathbb{S}^2$.

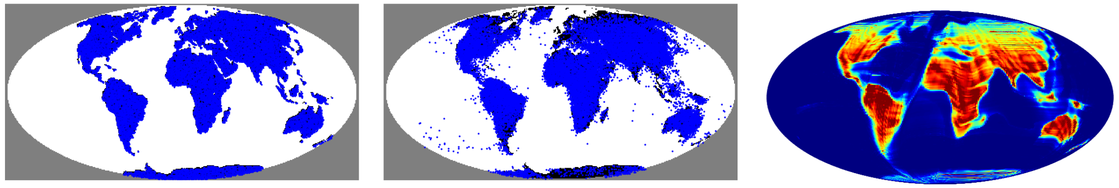
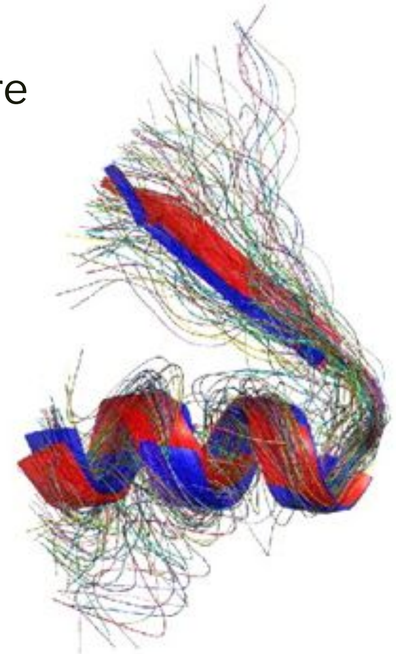


Figure 11. Learning a complex density from data samples using autoregressive spline flows. **Left**: Target density from which i.i.d. data samples were generated. **Middle**: Model samples overlaid on target density; **Right**: Heat map of the learned density.

Relevant for:

- HEP
- nuclear physics (lattice QCD)
- cosmology
- geology
- protein structure
- robotics

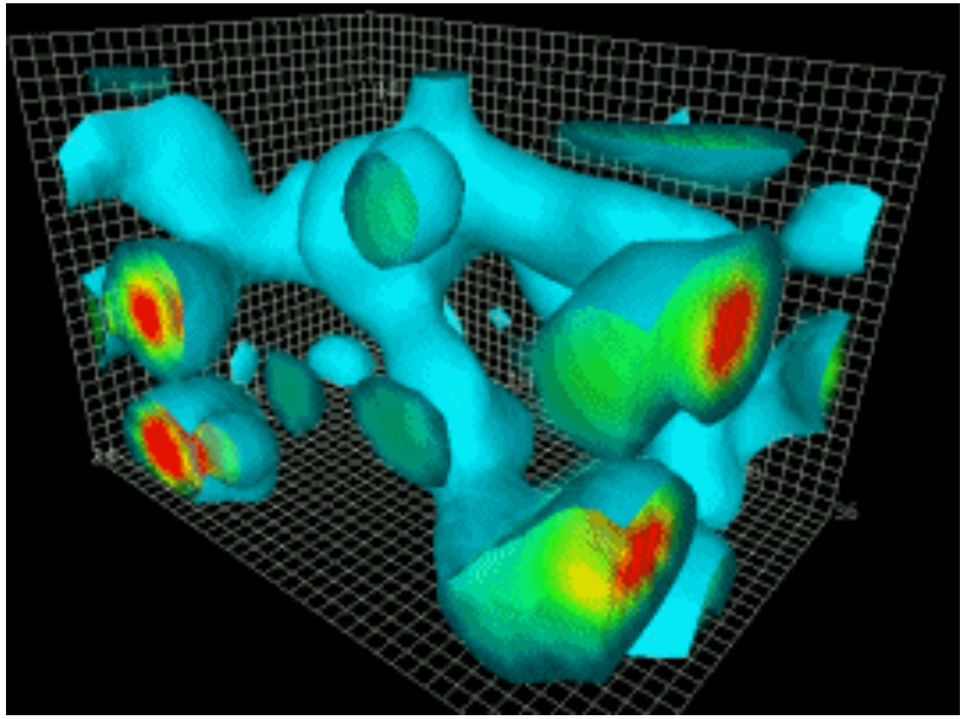


Protein figure from Boomsma

Lattice Field Theory

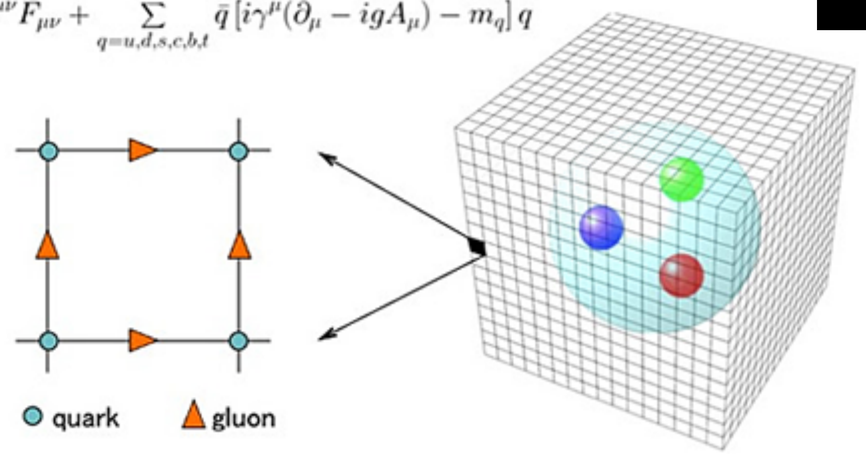
LATTICE QUANTUM CHROMO DYNAMICS

Very expensive simulations, ~1000 examples with $x \in \mathbb{R}^{10^9}$



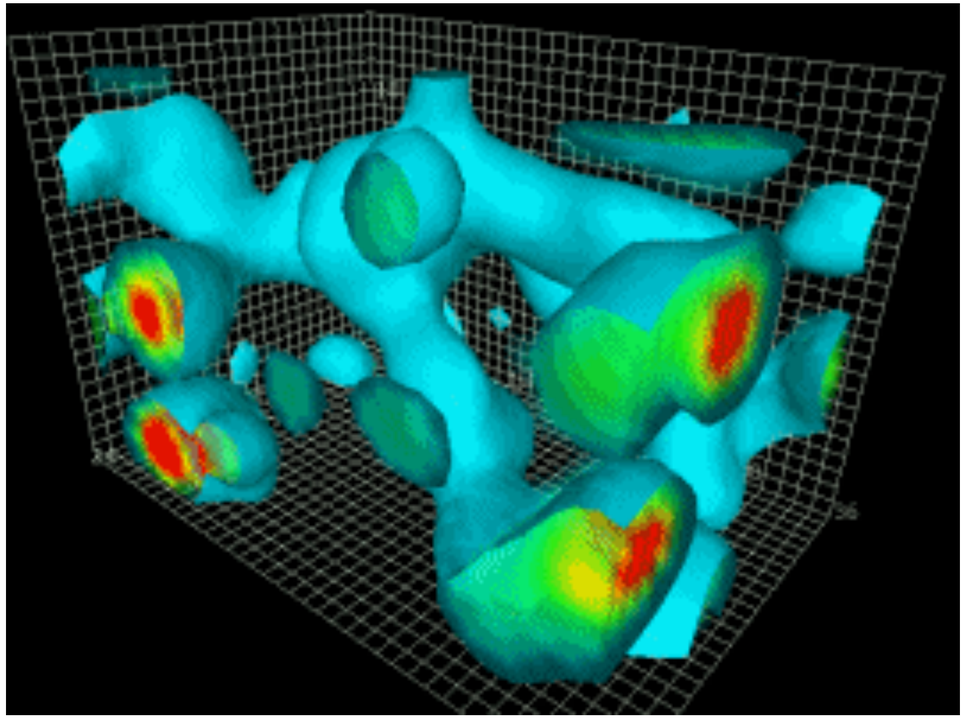
QCD Lagrangian

$$\mathcal{L} = -\frac{1}{4}F^{\mu\nu}F_{\mu\nu} + \sum_{q=u,d,s,c,b,t} \bar{q}\left[i\gamma^{\mu}(\partial_{\mu} - igA_{\mu}) - m_q\right]q$$



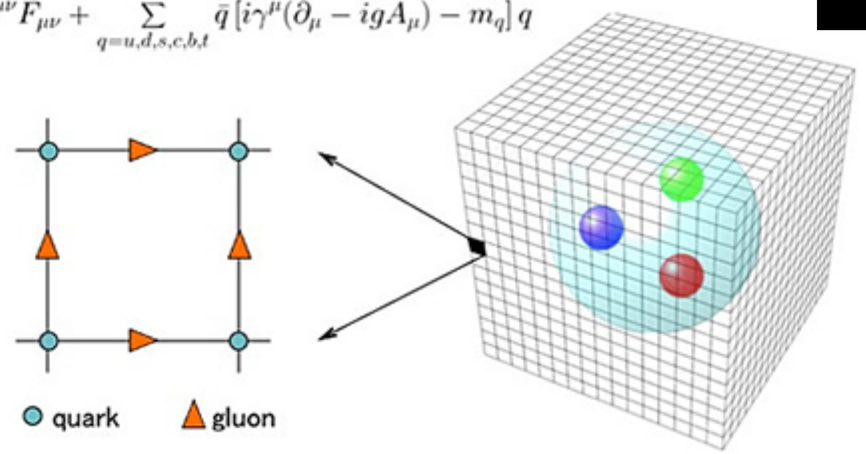
LATTICE QUANTUM CHROMO DYNAMICS

Very expensive simulations, ~1000 examples with $x \in \mathbb{R}^{10^9}$



QCD Lagrangian

$$\mathcal{L} = -\frac{1}{4}F^{\mu\nu}F_{\mu\nu} + \sum_{q=u,d,s,c,b,t} \bar{q}\left[i\gamma^{\mu}(\partial_{\mu} - igA_{\mu}) - m_q\right]q$$



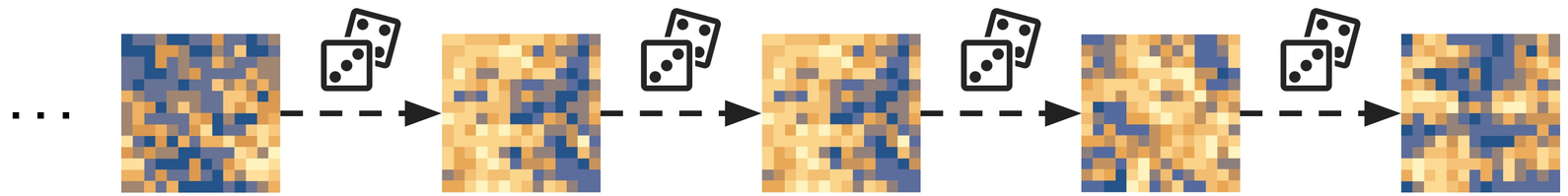
COMPUTATIONAL APPROACH TO LATTICE THEORIES

- Partition functions and path integrals are typically intractable analytically
- Numerical approximation by Monte Carlo sampling

sample integral according to $p(\phi) = e^{-S(\phi)}/Z$

estimate observables $\langle O \rangle \approx \frac{1}{N} \sum_i O_i$

- Markov Chain Monte Carlo converges to samples from p(ϕ)

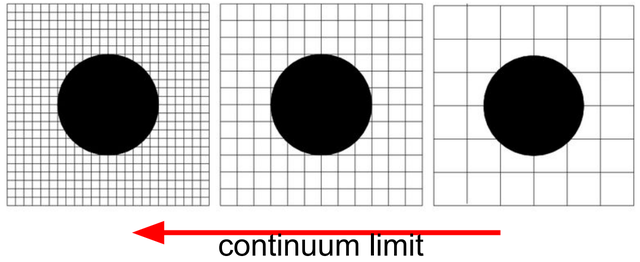


approximately distributed ~ p(ϕ)

10

Critical slowing down

- As params defining distribution approach criticality, for Markov chains using local updates, **autocorrelation time diverges**

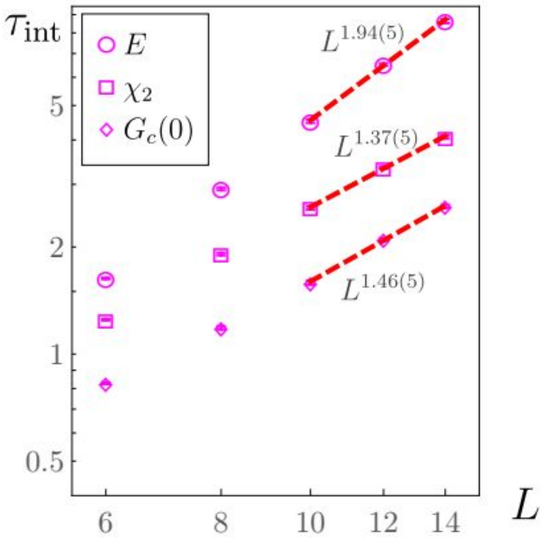


- Fitting τ^{int} to power law behavior gives dynamical critical exponents

$$\tau_{\mathcal{O}}^{\text{int}} = \alpha_{\mathcal{O}} L^{z_{\mathcal{O}}}$$

- Smaller dynamical critical exponent = cheaper, closer approach to criticality

CSD in scalar theory used in this work:



CSD also affects more realistic, complex models:

- CP^{N-1} [Flynn, et al. 1504.06292]
- O(N) [Frick, et al. PRL 63, 2613]
- QCD [ALPHA collaboration 1009.5228]
- ...

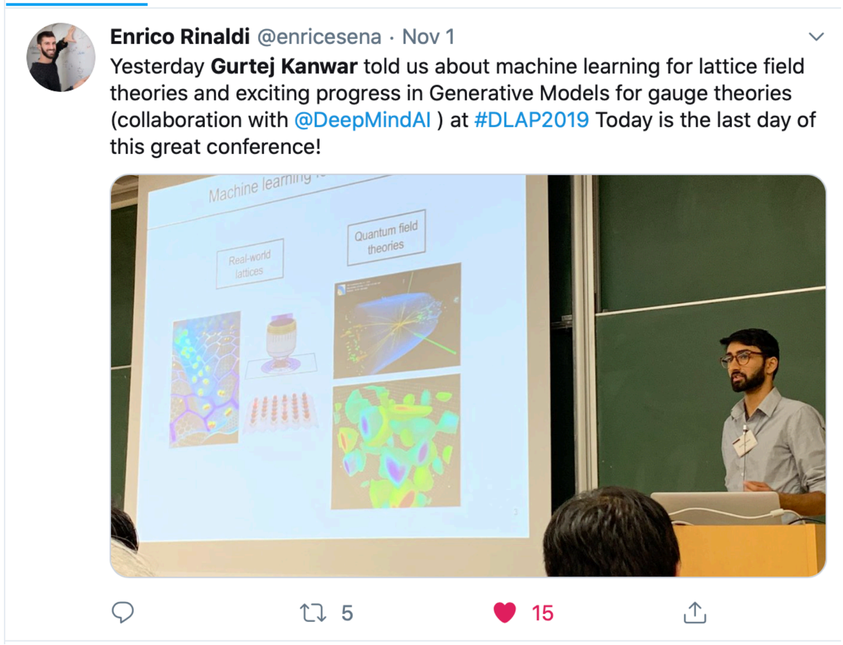
arXiv:1904.12072

Flow-based generative models for Markov chain Monte Carlo in lattice field theory

M. S. Albergo,[1,2,3] G. Kanwar,[4] and P. E. Shanahan[4,1]

[1] *Perimeter Institute for Theoretical Physics, Waterloo, Ontario N2L 2Y5, Canada*
[2] *Cavendish Laboratories, University of Cambridge, Cambridge CB3 0HE, U.K.*
[3] *University of Waterloo, Waterloo, Ontario N2L 3G1, Canada*
[4] *Center for Theoretical Physics, Massachusetts Institute of Technology, Cambridge, MA 02139, U.S.A.*

A Markov chain update scheme using a machine-learned *flow-based generative model* is proposed for Monte Carlo sampling in lattice field theories. The generative model may be optimized (trained) to produce samples from a distribution approximating the desired Boltzmann distribution determined by the lattice action of the theory being studied. Training the model systematically improves autocorrelation times in the Markov chain, even in regions of parameter space where standard Markov chain Monte Carlo algorithms exhibit critical slowing down in producing decorrelated updates. Moreover, the model may be trained without existing samples from the desired distribution. The algorithm is compared with HMC and local Metropolis sampling for ϕ^4 theory in two dimensions.



LOSS FUNCTION

Instead of KL(p || p̃) as in maximum likelihood fit to data sampled from x ~ p(x)

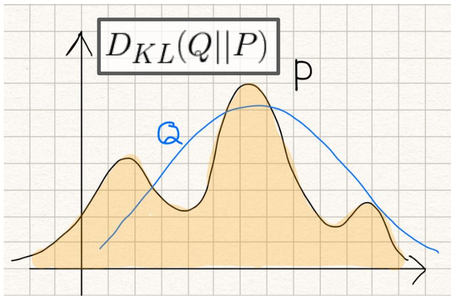
- Use KL(p̃ || p) as in variational inference
- eg. evaluate the action for configurations proposed from model p̃

- Use known target probability density: $p(\phi) = e^{-S(\phi)}/Z$

- For our application, train to **minimize shifted KL divergence**

shift removes unknown normalization Z

$$L(\tilde{p}_f) := D_{KL}(\tilde{p}_f||p) - \log Z$$
$$= \int \prod_j d\phi_j \, \tilde{p}_f(\phi) \left(\log \tilde{p}_f(\phi) + S(\phi)\right)$$



- Can apply **self-training**: sampling *model distribution* $\tilde{p}_f(\phi)$ to estimate loss

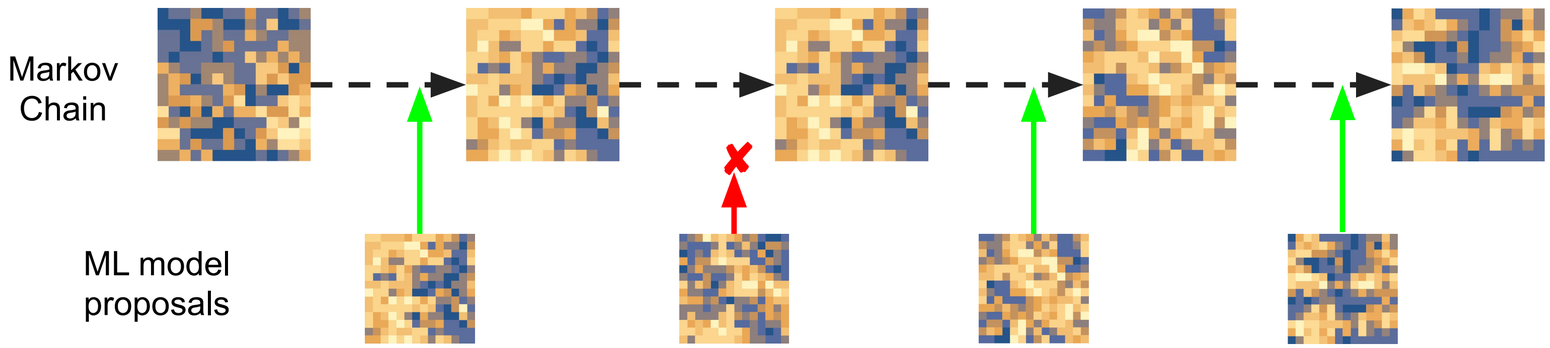
Not possible with a GAN,
Need a tractable likelihood

Correcting for model error

- Known model and target densities, many options to correct for error
- We use MCMC with proposals from ML model (interoperable with standard MC updates)
- Metropolis-Hastings step:

$$A(\phi^{(i-1)}, \phi') = \min\left(1, \frac{\tilde{p}(\phi^{(i-1)})}{p(\phi^{(i-1)})}\frac{p(\phi')}{\tilde{p}(\phi')}\right)$$

model proposal, independent of previous sample



SIMILAR WORK

Hierarchical variational models for statistical physics

Jaan Altosaar
Princeton University
altosaar@princeton.edu

Rajesh Ranganath
New York University
rajeshr@cims.nyu.edu

Kyle Cranmer
New York University
kyle.cranmer@nyu.edu

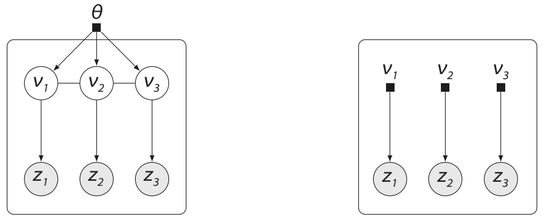


Figure 1: Hierarchical variational models (HVMs, left) capture dependencies between latent variables, compared to the mean-field variational family with independent variables (right).

RESEARCH ARTICLE SUMMARY

MACHINE LEARNING

Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning

Frank Noé*†, Simon Olsson*, Jonas Köhler*, Hao Wu

INTRODUCTION: Statistical mechanics aims to compute the average behavior of physical systems on the basis of their microscopic constituents. For example, what is the probability that a protein will be folded at a given temperature? If we could answer such questions efficiently, then we could not only comprehend the workings of molecules and materials, but we could also design drug molecules and materials with new properties in a principled way.

To this end, we need to compute statistics of the equilibrium states of many-body systems. In the protein-folding example, this means to consider each of the astronomically many ways to place all protein atoms in space, to compute the probability of each such "configuration" in the equilibrium ensemble, and then to compare the total probability of unfolded and folded configurations.

As enumeration of all configurations is infeasible, one instead must attempt to sample them from their equilibrium distribution. However, we currently have no way to generate equilibrium samples of many-body systems in "one shot." The main approach is thus to start with one configuration, e.g., the folded protein state, and make tiny changes to it over time, e.g., by using Markov-chain Monte Carlo or molecular dynamics (MD). However, these simulations get trapped in metastable (long-lived) states: For example, sampling a single folding or unfolding event with atomistic MD may take a year on a supercomputer.

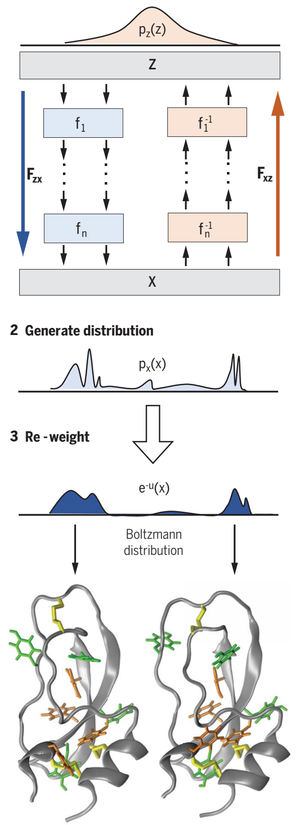
RATIONALE: Here, we combine deep machine learning and statistical mechanics to develop Boltzmann generators. Boltzmann generators are trained on the energy function of a many-body system and learn to provide unbiased, one-shot samples from its equilibrium state. This is achieved by training an invertible neural network to learn a coordinate transformation from a system's configurations to a so-called latent space representation, in which the low-energy configurations of different states are close to each other and can be easily sampled. Because of the invertibility, every latent space sample can be back-transformed to a system configuration with high Boltzmann probability (Fig. 1). We then employ statistical mechanics, which offers a rich set of tools for reweighting the distribution generated by the neural network to the Boltzmann distribution.

ON OUR WEBSITE

Read the full article at http://dx.doi.org/10.1126/science.aaw1147

RESULTS: Boltzmann generators can be trained to directly generate independent samples of low-energy structures of condensed-matter systems and protein molecules. When initialized with a few structures from different metastable states, Boltzmann generators can generate statistically independent samples from these states and efficiently compute the free-energy differences between them. This capability could be used to compute relative stabilities between different experimental structures of protein or other organic molecules, which is currently a very challenging problem. Boltzmann generators can also learn a notion of "reaction coordinates": Simple linear interpolations between points in latent space have a high probability of corresponding to physically realistic, low-energy transition pathways. Finally, by using established sampling methods such as Metropolis Monte Carlo in the latent space variables, Boltzmann generators can discover new states and gradually explore state space.

CONCLUSION: Boltzmann generators can overcome rare event–sampling problems in many-body systems by learning to generate unbiased equilibrium samples from different metastable states in one shot. They differ conceptually from established enhanced sampling methods, as no reaction coordinates are needed to drive them between metastable states. However, by applying existing sampling methods in the latent spaces learned by Boltzmann generators, a plethora of new opportunities opens up to design efficient sampling methods for many-body systems. ■



Boltzmann generators overcome sampling problems between long-lived states. The Boltzmann generator works as follows: 1. We sample from a simple (e.g., Gaussian) distribution. 2. An invertible deep neural network is trained to transform this simple distribution to a distribution $p_X(x)$ that is similar to the desired Boltzmann distribution of the system of interest. 3. To compute thermodynamics quantities, the samples are reweighted to the Boltzmann distribution using statistical mechanics methods.

The list of author affiliations is available in the full article online.
*These authors contributed equally to this work.
†Corresponding author. Email: frank.noe@fu-berlin.de

Cite this article as F. Noé *et al.*, *Science* **365**, eaaw1147 (2019). DOI: 10.1126/science.aaw1147

VALIDATION

arXiv:1904.12072

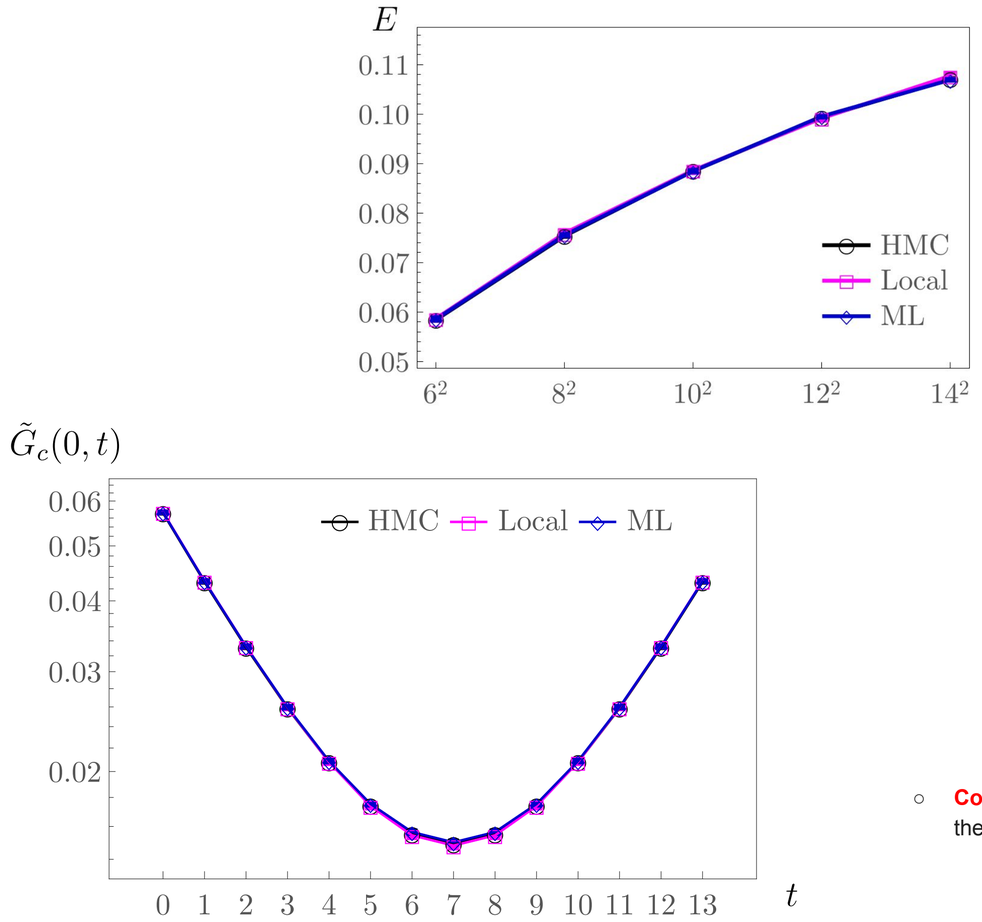
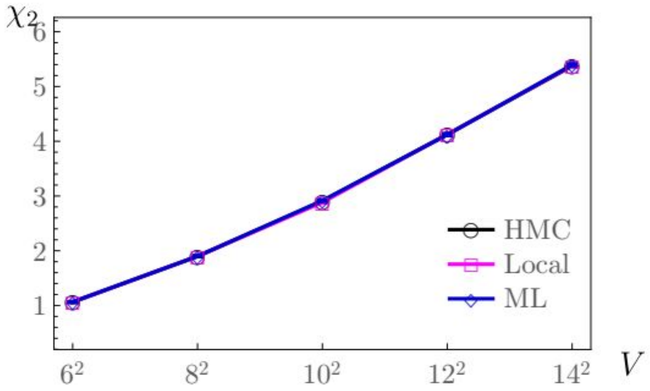


FIG. 3: Zero-momentum Green's functions evaluated for parameter set E5. Results computed using 10^6 configurations from the HMC, local Metropolis, and machine-learned (ML) ensembles are consistent within statistical errors. Error bars indicate 68% confidence intervals estimated using bootstrap resampling with bins of size 100.

Ising energy and two-point susceptibility agree.



- **Correlation functions**, relating to masses in the (discretized) quantum field theory

$$\begin{cases} G_c(x) = \dfrac{1}{V}\displaystyle\sum_y \langle \phi(y)\phi(y+x)\rangle \\ m_p = -\partial_t \log\left\langle \tilde{G}_c(0,t)\right\rangle \end{cases}$$

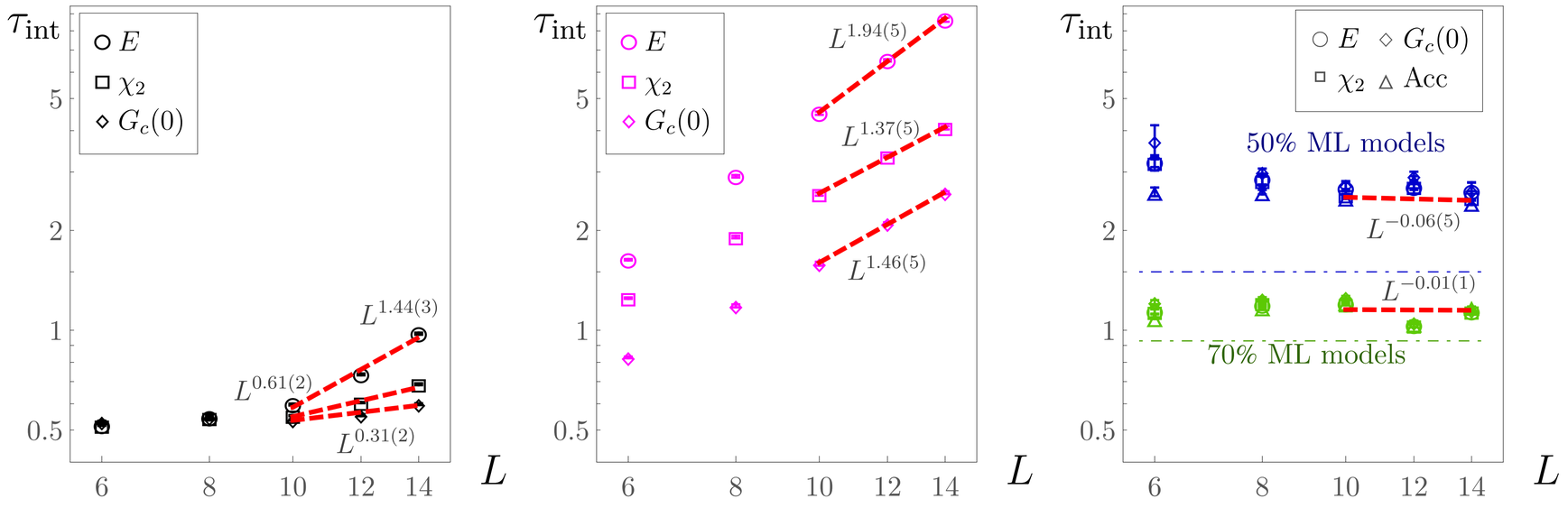
- **Response of the vacuum** to an impulse (two-point susceptibility)

$$\chi_2 = \sum_x G_c(x)$$

- **Energy measurement** relating to Ising model microstate energy in a particular limit of ϕ^4 theory

$$E = \frac{1}{d}\sum_{1\le\mu\le d} G_c(\hat{\mu})$$

Unlike MCMC, Samples from the flow are uncorrelated



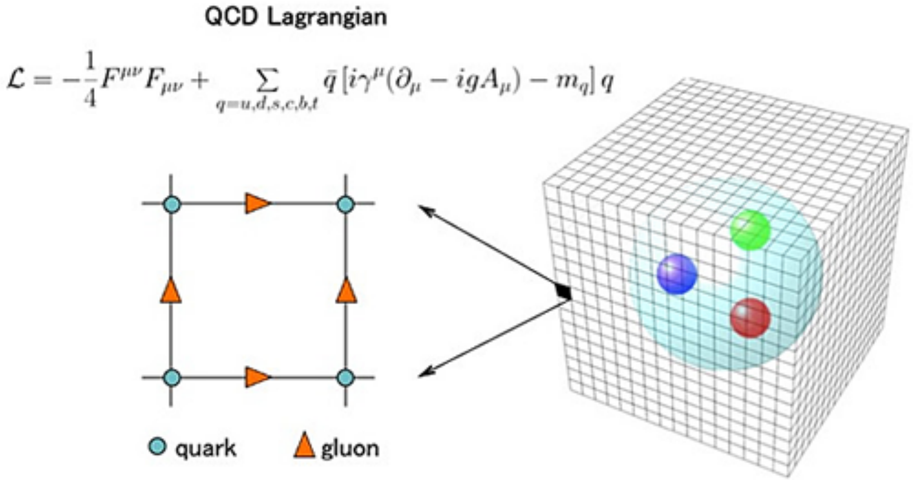
(a) HMC ensembles (b) Local Metropolis ensembles (c) Flow-based MCMC ensembles

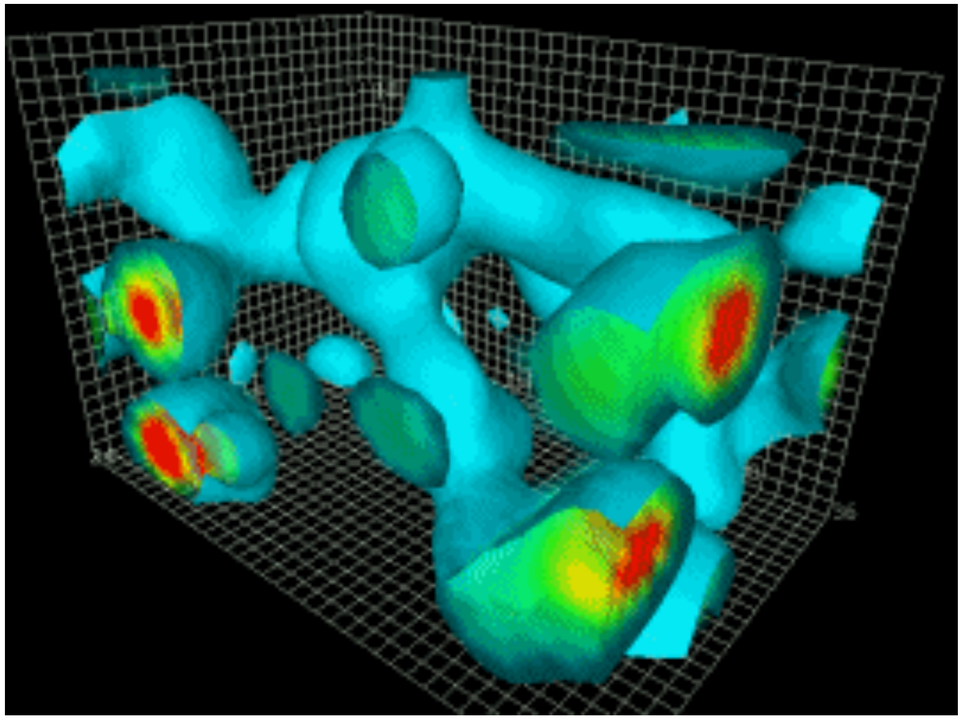
FIG. 7: Scaling of integrated autocorrelation time with respect to lattice size for HMC, local Metropolis, and flow-based MCMC. In (c) the upper sets of points in blue correspond to models trained to a mean acceptance rate of 50%, while the lower sets of points in green correspond to models trained to a mean acceptance rate of 70%. Dashed red lines display power law fits to $L = \{10, 12, 14\}$ with labels L^z specifying the scaling. The HMC and local Metropolis methods demonstrate power-law growth of τ_{int}, while τ_{int} for the flow-based MCMC is consistent with a constant in L and decreases as mean acceptance rate increases. Dot-dashed blue and green lines for the flow-based ensembles display lower bounds in terms of mean acceptance rate based on Eq. (18). Error bars indicate 68% confidence intervals estimated by bootstrap resampling and error propagation.

LQCD AS A TESTBED FOR DL

Each of the 10^7 lattice locations has data $x_i \in \mathbb{R}^{32}$ with non-trivial data with continuous local symmetry.

- space-time translation invariance → convolutional architecture
- local gauge symmetry → design group-invariant convolutional filters
- coarse graining & renormalization group → hierarchical convolutions shared weights
- few very, large training examples → rethink minibatching & SGD





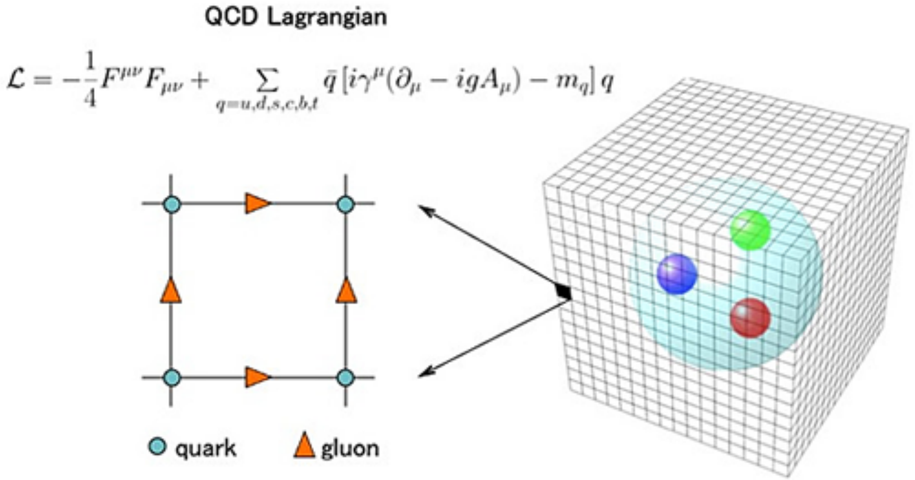
We are collaborating with: Google DeepMind

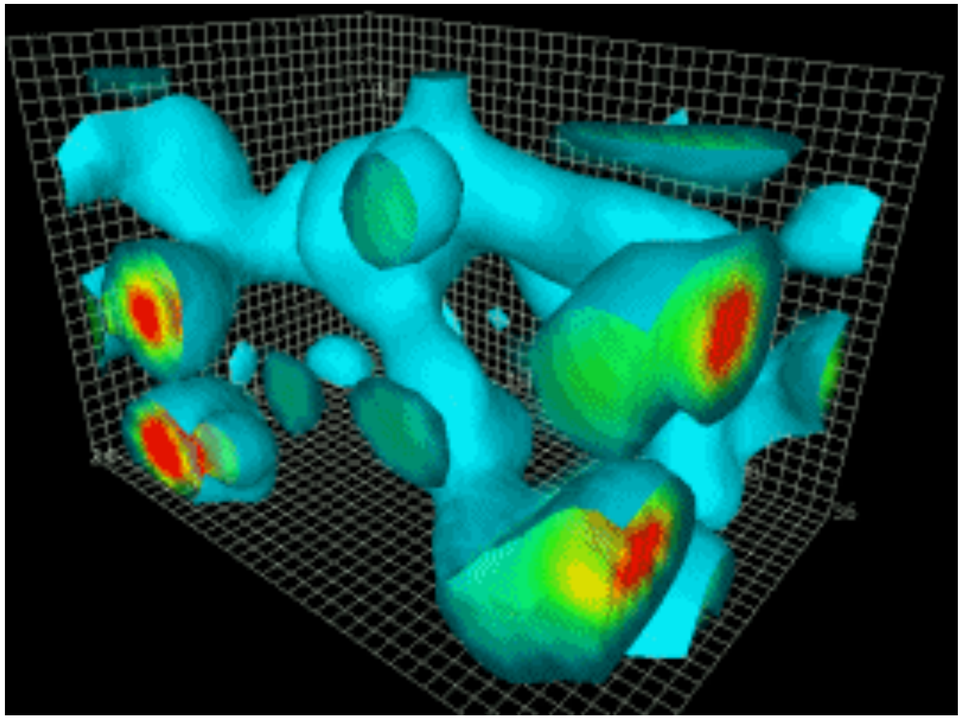
[link to explainer]

LQCD AS A TESTBED FOR DL

Each of the 10^7 lattice locations has data $x_i \in \mathbb{R}^{32}$ with non-trivial data with continuous local symmetry.

- space-time translation invariance → convolutional architecture
- local gauge symmetry → design group-invariant convolutional filters
- coarse graining & renormalization group → hierarchical convolutions shared weights
- few very, large training examples → rethink minibatching & SGD





We are collaborating with: Google DeepMind

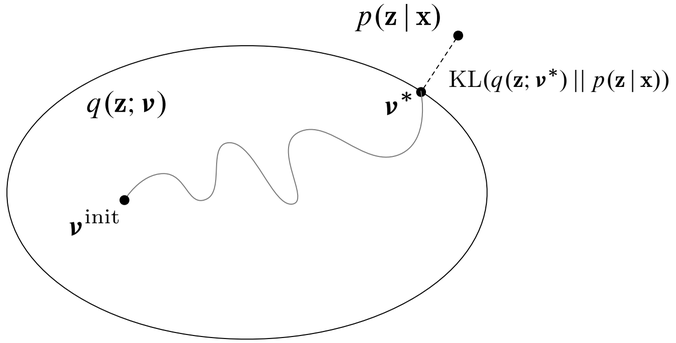
[link to explainer]

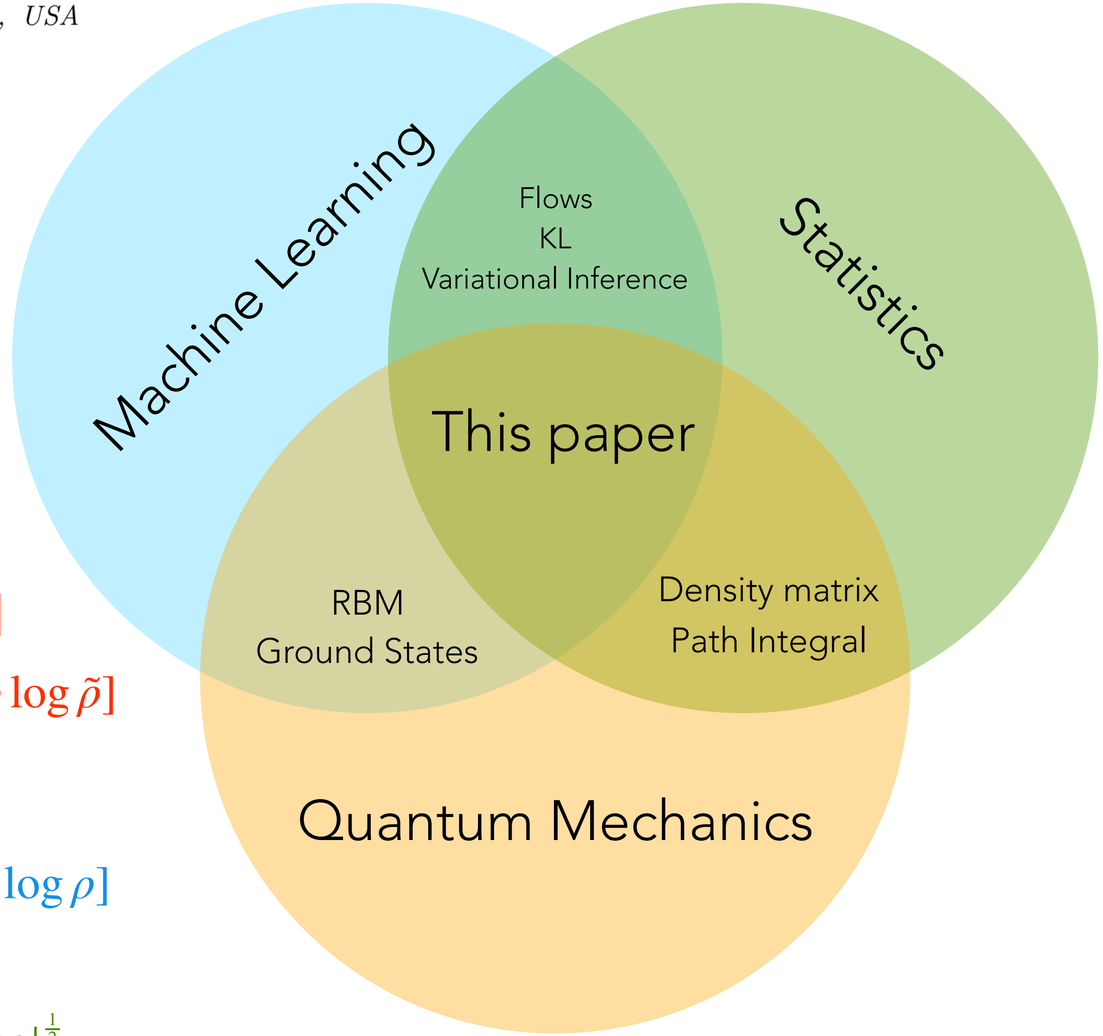
Quantum Flows

Inferring the quantum density matrix with machine learning

Kyle Cranmer* and Siavash Golkar†
Center for Cosmology and Particle Physics, Department of Physics, New York University, New York, NY 10003, USA.

Duccio Pappadopulo‡
Bloomberg LP, New York, NY 10022, USA





Quantum Maximum Likelihood

$$\mathrm{KL}[p\|\,q] \to S[\rho\|\,\tilde{\rho}] = \mathrm{Tr}[\rho\log\rho] - \mathrm{Tr}[\rho\log\tilde{\rho}]$$

Quantum Variational Inference

$$\mathrm{KL}[q\|\,p] \to S[\tilde{\rho}\|\,\rho] = \mathrm{Tr}[\tilde{\rho}\log\tilde{\rho}] - \mathrm{Tr}[\tilde{\rho}\log\rho]$$

Quantum Normalizing Flows

$$p(x) = p(f(x))\left|\det\frac{\partial f}{\partial x}\right| \quad\to\quad \psi_i(x) = \phi_i(f(x))\left|\det\frac{\partial f}{\partial x}\right|^{\frac{1}{2}}$$

KC, Golkar, Pappadopulo, doi:10.6084/m9.figshare.6197069; arXiv:1904.05903
Pfau, Petersen, Agarwal, Barrett, Stachenfeld, arXiv: 1806.02215

Inferring the quantum density matrix with machine learning

Kyle Cranmer* and Siavash Golkar†
Center for Cosmology and Particle Physics, Department of Ph
New York University, New York, NY 10003, US

Duccio Pappadopul*
Bloomberg LP, New York

Spectral Inference Networks: Unifying Spectral Methods With Deep Learning

David Pfau[1], **Stig Petersen**[1], **Ashish Agarwal**[2], **David Barrett**[1] **and Kim Stachenfeld**[1]
[1]DeepMind London, UK
[2]Google Brain Mountain View, CA, USA
{pfau, svp, agarwal, barrettdavid, stachenfeldj}@google.com

Abstract

We present Spectral Inference Networks, a framework for learning eigenfunctions of linear operators by stochastic optimization. Spectral Inference Networks generalize Slow Feature Analysis to generic symmetric operators, and are closely related to Variational Monte Carlo methods from computational physics. As such, they can be a powerful tool for unsupervised representation learning from video or pairs of data. We derive a training algorithm for Spectral Inference Networks that addresses the bias in the gradients due to finite batch size and allows for online learning of multiple eigenfunctions. We show results of training Spectral Inference Networks on problems in quantum mechanics and feature learning for videos on synthetic datasets as well as the Arcade Learning Environment. Our results demonstrate that Spectral Inference Networks accurately recover eigenfunctions of linear operators, can discover interpretable representations from video and find meaningful subgoals in reinforcement learning environments.

Statistics

Quantum Mechanics

Quantum Maxi

$\mathrm{KL}[p\|q] \to S[\rho\|\tilde{\rho}]$ -

Quantum Variatio

$\mathrm{KL}[q\|p] \to S[\tilde{\rho}\|\rho] = \mathrm{Tr}[$

Quantum Normalizing

$$p(x) = p(f(x)) \left|\det \frac{\partial f}{\partial x}\right| \quad \to \quad \psi_i(x) = q \qquad \left|\det \frac{\partial f}{\partial x}\right|^2$$

KC, Golkar, Pappadopulo, doi:10.6084/m9.figshare.6197069; arXiv:1904.05903
Pfau, Petersen, Agarwal, Barrett, Stachenfeld, arXiv: 1806.02215

Classical probability	Quantum mechanics
probability density $p(x)$	wave function $\psi(x) \equiv \langle \psi \vert x \rangle$
mixture component $p_j(x)$	pure state $\psi_j(x) \equiv \langle \psi_j \vert x \rangle$
—	superposition $\vert\psi\rangle = \sum_j a_j \vert\psi_j\rangle$
mixture model $p_{\text{mix}}(x)$	density matrix ρ
$p_{\text{mix}}(x) = \sum_j a_j p_j(x)$	$\rho(x,y) = \sum_j a_j \psi_j^*(x)\psi_j(y)$
$\mathbb{E}[O] = \int_x p(x)O(x)dx$	$\langle O \rangle = \text{Tr}\,[\rho\, O]$
Gibbs sampling	Monte Carlo approx. of path integral
KL divergence $D_{\text{KL}}(p \parallel q)$	quantum relative entropy $S(\rho \Vert \sigma)$

Consider a true thermal ensemble with unknown energy levels E_n and unknown eigenstates $|n\rangle$

$$\rho = \frac{\sum_n e^{-\beta E_n} |n\rangle\langle n|}{\sum_n e^{-\beta E_n}} = \sum_n \lambda_n |n\rangle\langle n|.$$

The variational model for the density matrix is

$$\tilde{\rho} = \sum_i \tilde{\lambda}_i |i\rangle\langle i|$$

- where states and energy levels are parametrized

We will try to estimate the true density matrix by minimizing

$$-\operatorname{Tr}[\rho \log \tilde{\rho}]$$

ESTIMATING THE VON NEUMANN ENTROPY

Identifying $\mathcal{O}$ with $\log\tilde{\rho}$ we can express in various ways

$$
\begin{aligned}
\mathrm{Tr}[\rho\ \log\tilde{\rho}] &= \int dx\,dx' \int_{q(-\beta/2)=x}^{q(\beta/2)=x'} \mathscr{D}q\ e^{-S_E[q]} \left[\sum_i \log\tilde{\lambda}_i\ \tilde{\psi}_i(x)\tilde{\psi}_i^*(x')\right] \\
&= \int dx\,dx'\,\langle x|\rho|x'\rangle \left[\sum_i \log\tilde{\lambda}_i\ \tilde{\psi}_i(x)\tilde{\psi}_i^*(x')\right] \\
&= \int dx\,dx' \sum_n \lambda_n\psi_n(x)\psi_n^*(x') \left[\sum_i \log\tilde{\lambda}_i\ \tilde{\psi}_i(x)\tilde{\psi}_i^*(x')\right] \\
&= \sum_n \lambda_n \left[\sum_i \log\tilde{\lambda}_i\ |\langle n|i\rangle|^2\right]
\end{aligned}
$$

ESTIMATING THE VON NEUMANN ENTROPY

Identifying $\mathcal{O}$ with $\log\tilde{\rho}$ we can express in various ways

$$
\begin{aligned}
\mathrm{Tr}[\rho\ \log\tilde{\rho}] &= \int dx\,dx' \int_{q(-\beta/2)=x}^{q(\beta/2)=x'} \mathscr{D}q\ e^{-S_E[q]} \left[\sum_i \log\tilde{\lambda}_i\ \tilde{\psi}_i(x)\tilde{\psi}_i^*(x')\right] \\
&= \int dx\,dx'\,\langle x|\rho|x'\rangle \left[\sum_i \log\tilde{\lambda}_i\ \tilde{\psi}_i(x)\tilde{\psi}_i^*(x')\right] \\
&= \int dx\,dx' \sum_n \lambda_n\psi_n(x)\psi_n^*(x') \left[\sum_i \log\tilde{\lambda}_i\ \tilde{\psi}_i(x)\tilde{\psi}_i^*(x')\right] \\
&= \sum_n \lambda_n \left[\sum_i \log\tilde{\lambda}_i\ |\langle n|i\rangle|^2\right] \\
&\approx \sum_{(x_i,x_i')\sim\rho(x,x')} \left[\sum_i \log\tilde{\lambda}_i\ \tilde{\psi}_i(x_i)\tilde{\psi}_i^*(x_i')\right]
\end{aligned}
$$

ESTIMATING THE VON NEUMANN ENTROPY

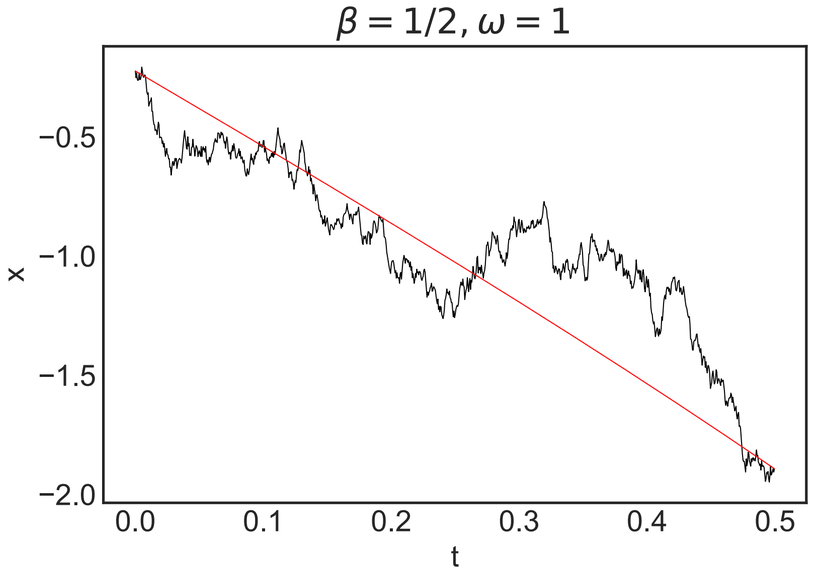
Identifying $\mathcal{O}$ with $\log\tilde{\rho}$ we can express in various ways

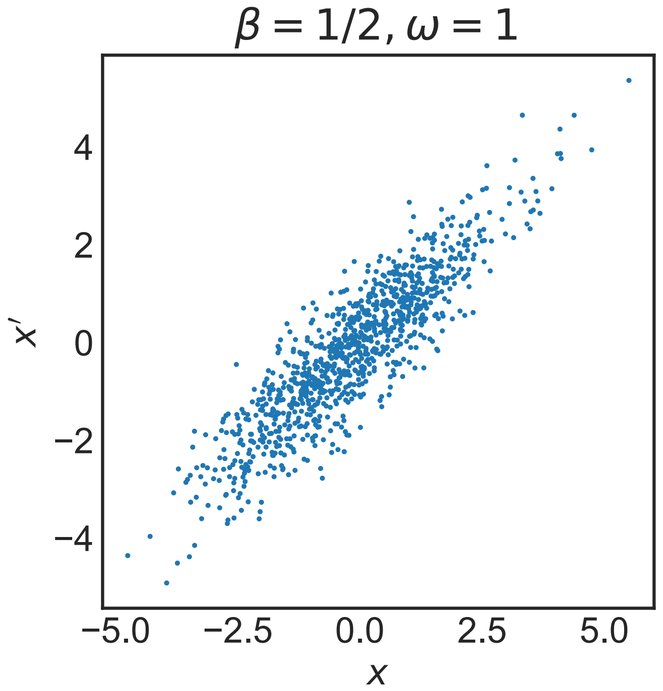
$$
\begin{aligned}
\mathrm{Tr}[\rho\ \log\tilde{\rho}] &= \int dx\,dx' \int_{q(-\beta/2)=x}^{q(\beta/2)=x'} \mathscr{D}q\ e^{-S_E[q]} \left[\sum_i \log\tilde{\lambda}_i\ \tilde{\psi}_i(x)\tilde{\psi}_i^*(x')\right] \\
&= \int dx\,dx'\,\langle x|\rho|x'\rangle \left[\sum_i \log\tilde{\lambda}_i\ \tilde{\psi}_i(x)\tilde{\psi}_i^*(x')\right] \\
&= \int dx\,dx' \sum_n \lambda_n\psi_n(x)\psi_n^*(x') \left[\sum_i \log\tilde{\lambda}_i\ \tilde{\psi}_i(x)\tilde{\psi}_i^*(x')\right] \\
&= \sum_n \lambda_n \left[\sum_i \log\tilde{\lambda}_i\ |\langle n|i\rangle|^2\right] \\
&\approx \sum_{(x_i,x_i')\sim\rho(x,x')} \left[\sum_i \log\tilde{\lambda}_i\ \tilde{\psi}_i(x_i)\tilde{\psi}_i^*(x_i')\right]
\end{aligned}
$$

* need enough samples to estimate inner products $\langle n|i\rangle$

LOSS FUNCTION

Use path integral to sample





$$(x_i, x_i') \sim \rho(x, x')$$

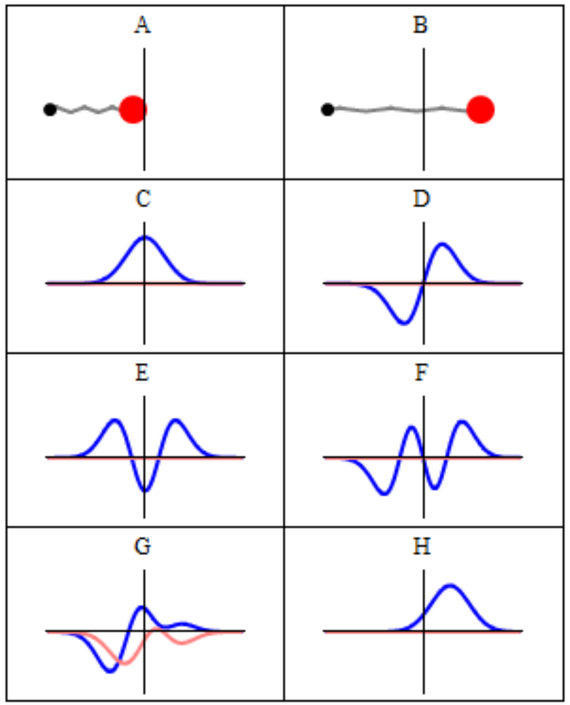
Minimize the following loss function

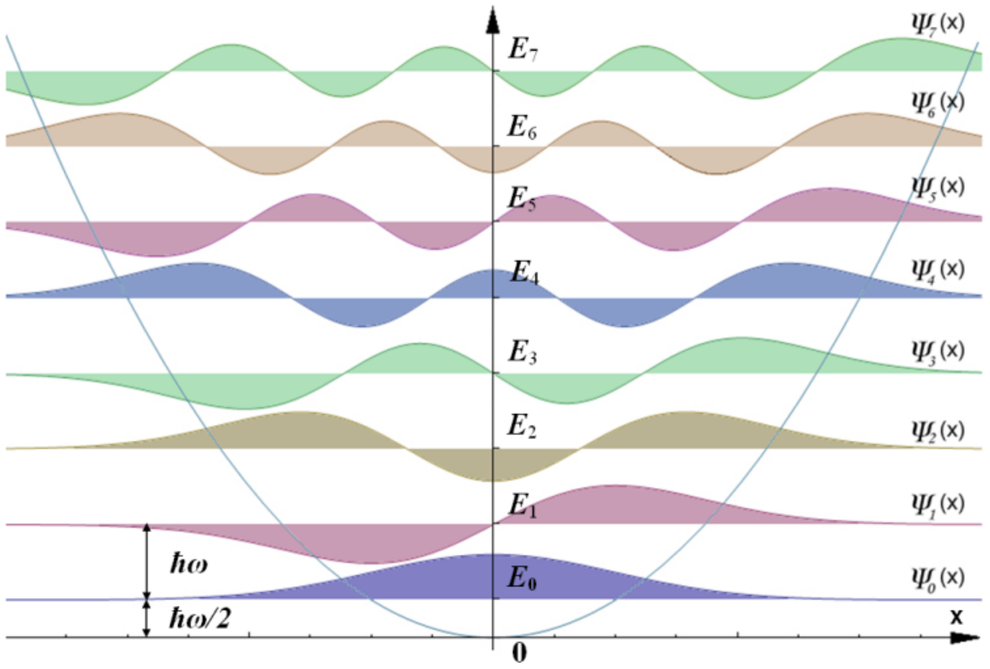
$$L[\tilde{\rho}] = \sum_{(x_i, x_i') \sim \rho(x, x')} \left[\sum_i -\log \tilde{\lambda}_i \, \tilde{\psi}_i(x_i) \tilde{\psi}_i^*(x_i') \right] + C \sum_{i<j} \sum_k a_k^{(i)} a_k^{(j)}$$

SIMPLE HARMONIC OSCILLATOR

Start simple:

- 0+1 Quantum Field Theory = 1-D Quantum Mechanics

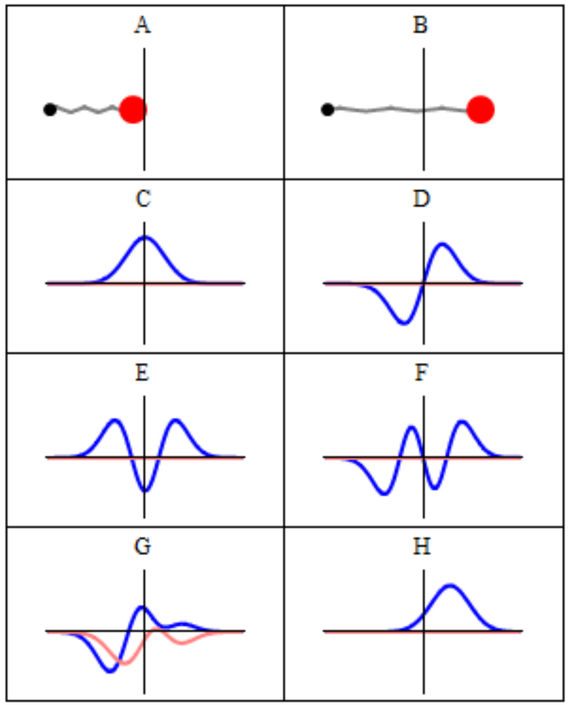


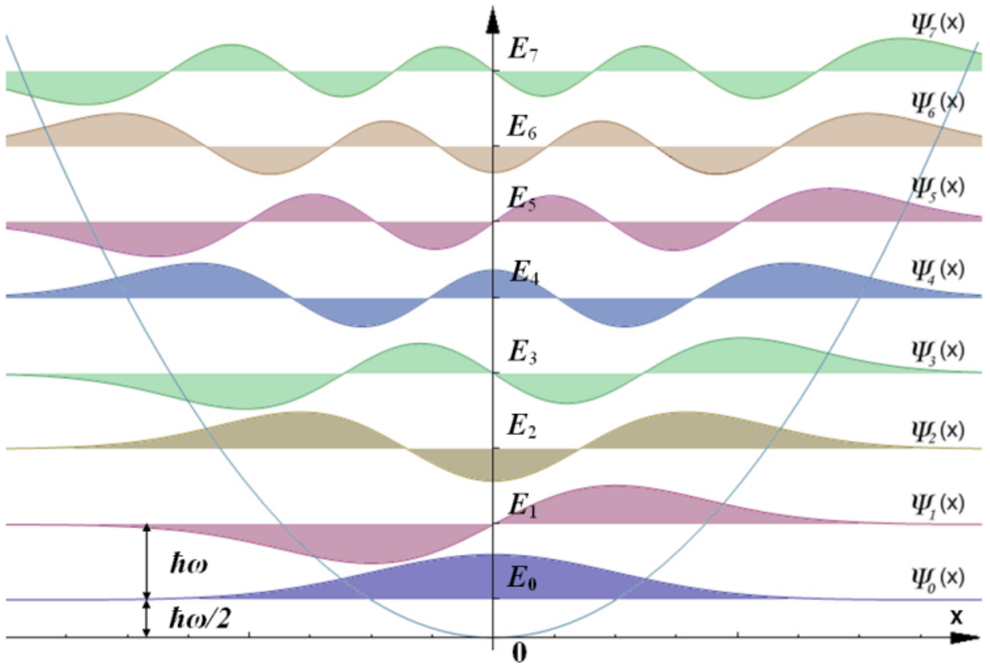


SIMPLE HARMONIC OSCILLATOR

Start simple:

- 0+1 Quantum Field Theory = 1-D Quantum Mechanics

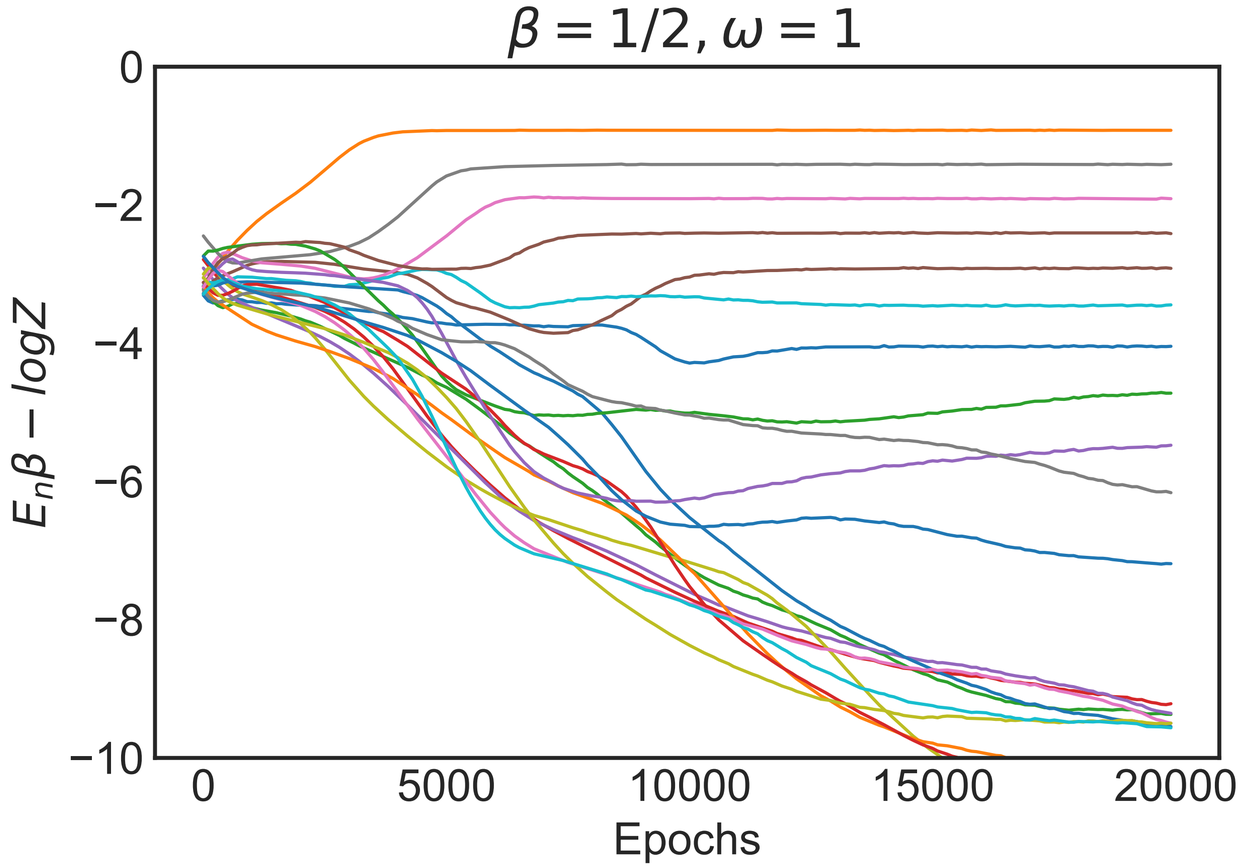




FITTING THE ENERGY EIGENVALUES

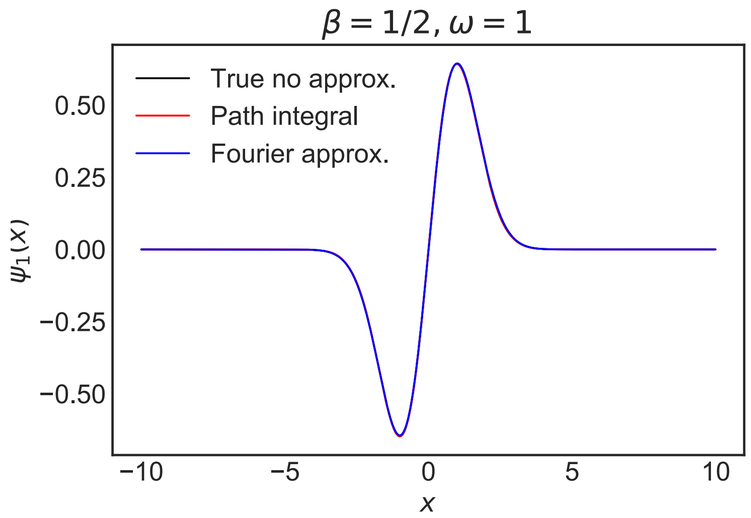
We get the first ~6 energy levels (limited by training & number of samples from path integral)

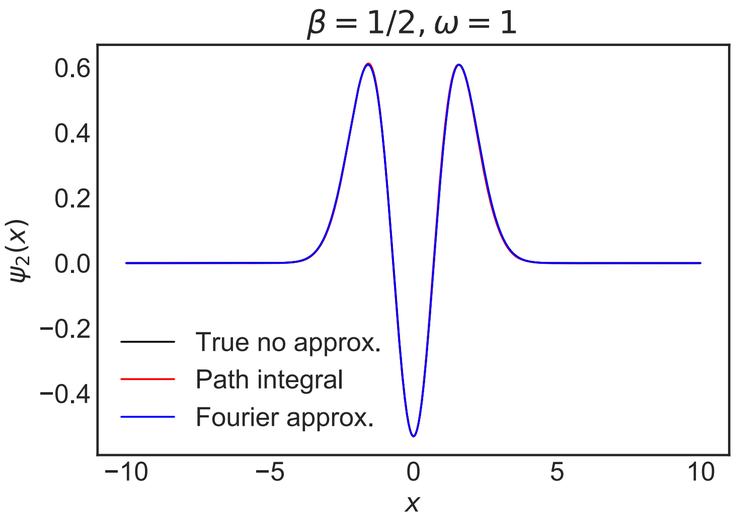
- corresponding to probabilities: 39, 24, 14, 8, 5, 3%

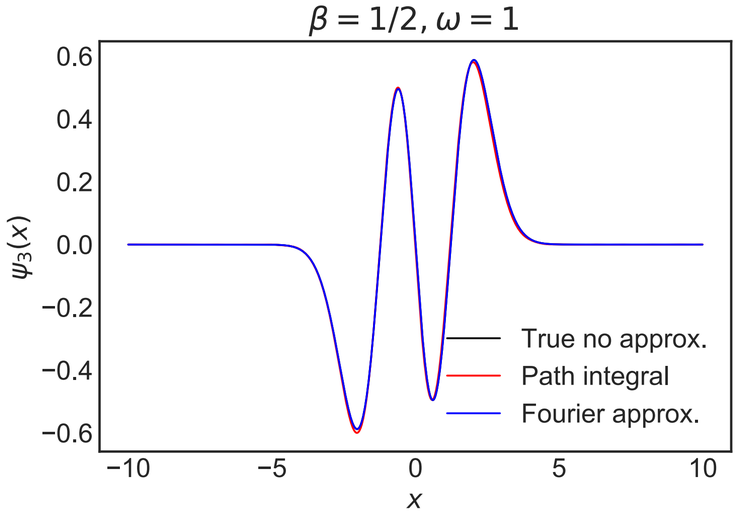


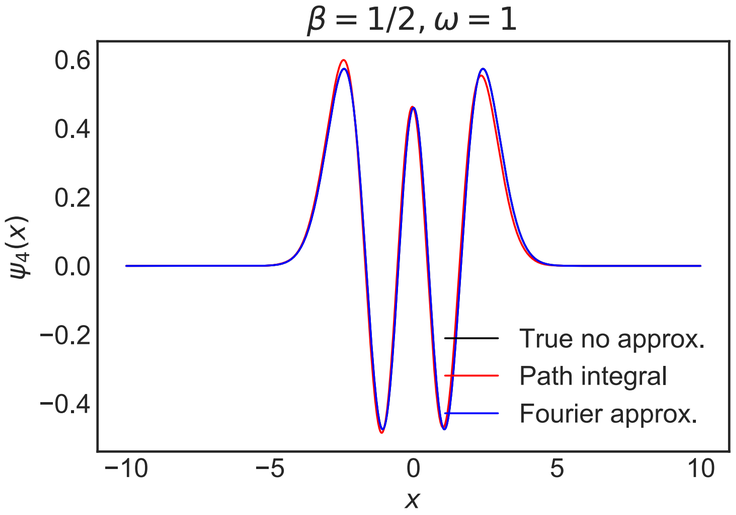
FITTING THE ENERGY EIGENSTATES

Can extract the eigenstates as well









CHALLENGE: ORTHOGONALITY

In order to express $\log \tilde{\rho}$ as

$$\mathcal{O}(x, x') = (\log \tilde{\rho})(x, x') = \sum_i \log \tilde{\lambda}_i \, \tilde{\psi}_i(x) \tilde{\psi}_i^*(x')$$

we need the states to be orthogonal

$$\int \tilde{\psi}_i(x) \tilde{\psi}_j^*(x) \, dx = \delta_{ij}$$

How do we enforce this?

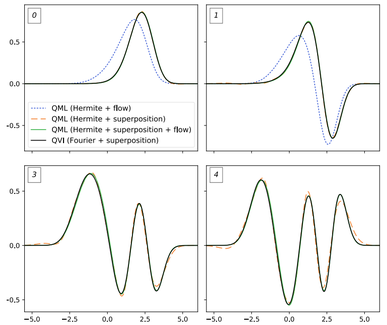
- explicitly in the loss function
- by construction in the variational family

QUANTUM FLOWS

https://arxiv.org/abs/1904.05903

Can we use the idea for flows directly on **an orthonormal basis** of complex quantum wave functions?

- instead of p(z) → p(x) can we do $\varphi_i(z) \rightarrow \psi_i(x)$?
- yes!



1) Start with:

$$\int dz\, \phi_i(z)\phi_j^*(z) = \delta_{ij}$$

2) Change variables:

$$\int dz\, \phi_i(z)\phi_j^*(z) = \int dx \left|\det \frac{\partial f}{\partial x}\right| \phi_i(f(x))\phi_j^*(f(x)) = \delta_{ij}$$

3) Profit:

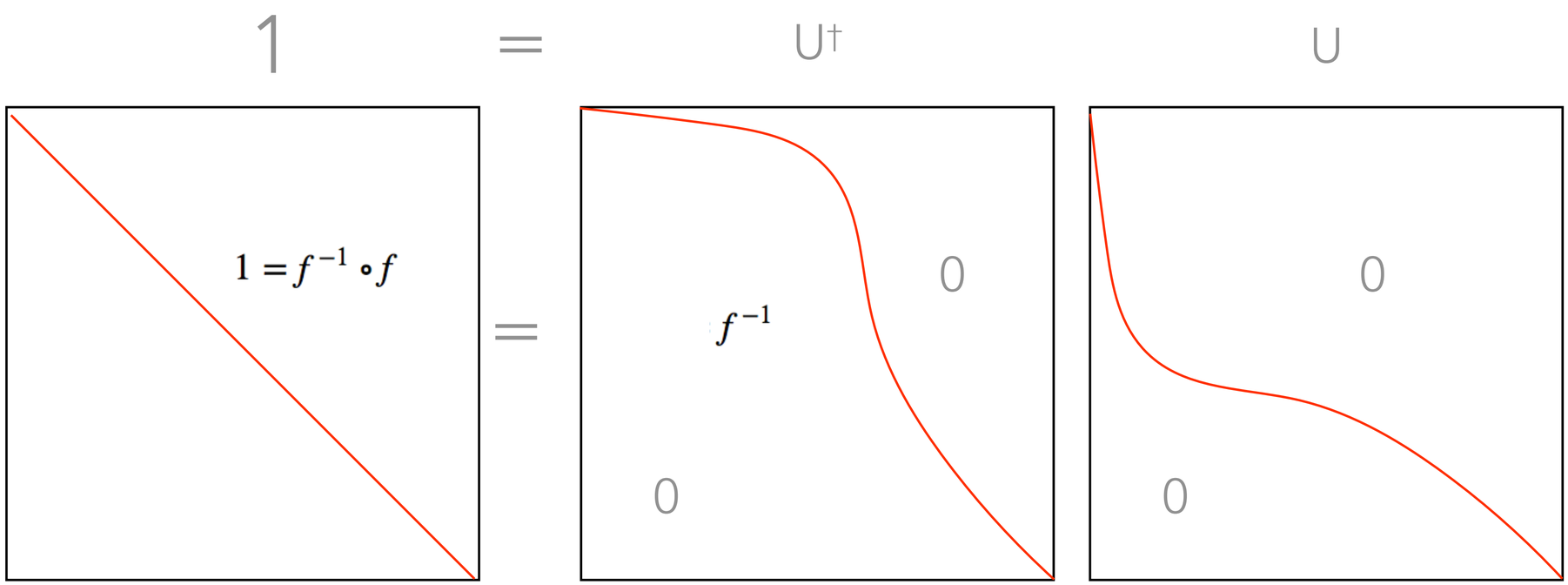
$$\psi_i(x) = \phi_i(f(x)) \left|\det \frac{\partial f}{\partial x}\right|^{\frac{1}{2}}$$

complex! & orthonormal!

FLOWS AS A UNITARY OPERATOR

Can view the quantum flow as a unitary operator

$$\psi_i(x) = U[\phi_i(z)] = \int dz\, \delta(f(x) - z) \left| \det \frac{\partial f}{\partial x} \right|^{\frac{1}{2}} e^{i\eta(x)} [\phi_i(z)]$$



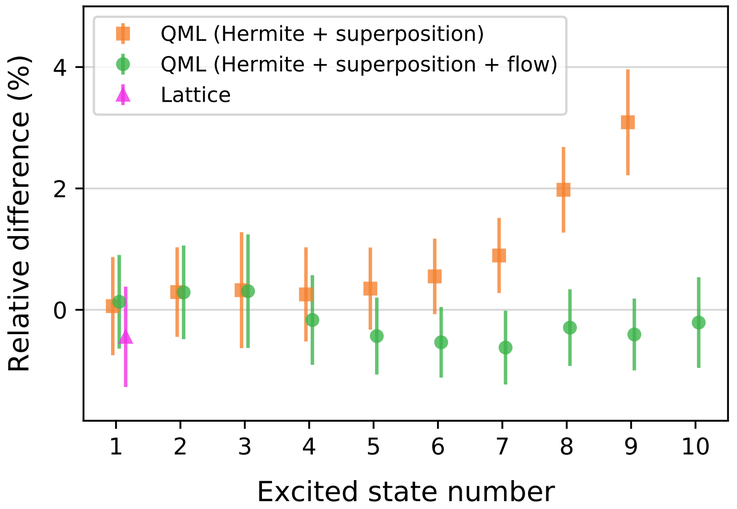
and much more: tangent space, parallel transport, …

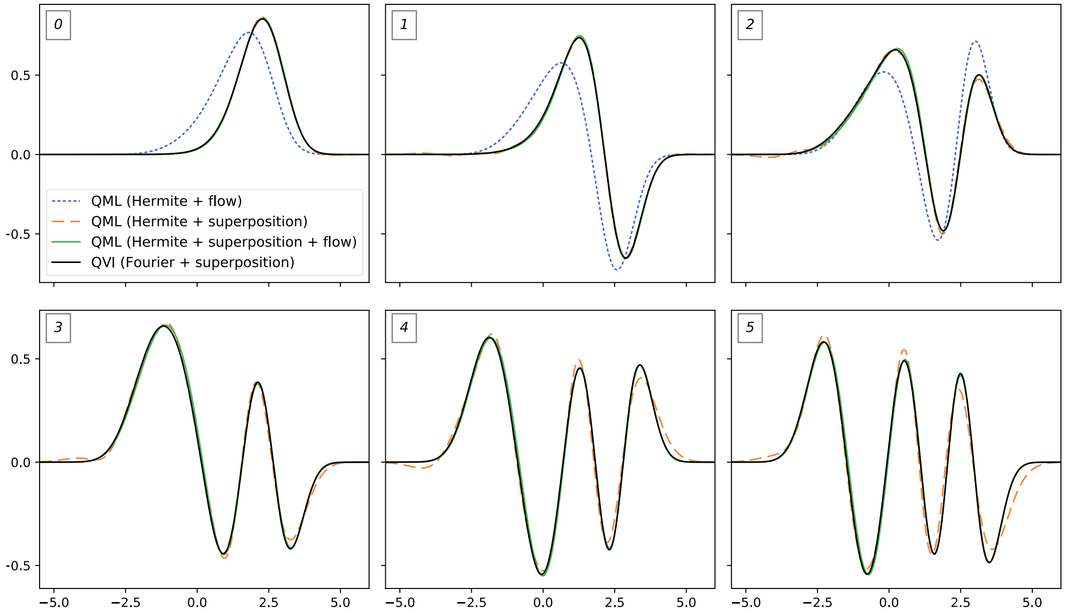
ANHARMONIC OSCILLATOR

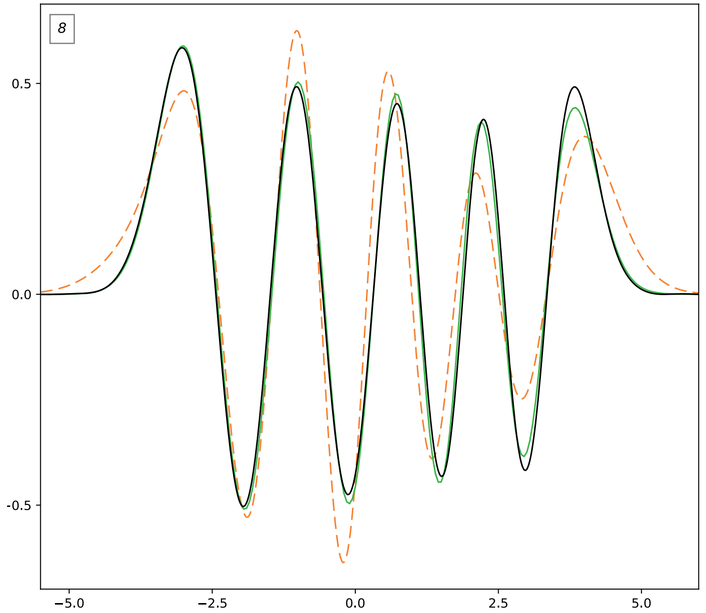
arXiv:1904.05903

Instead of modeling states in generic basis, try to perturb states of similar system with flow

QML (Hermite + flow)
QML (Hermite + superposition)
QML (Hermite + superposition + flow)
QVI (Fourier + superposition)



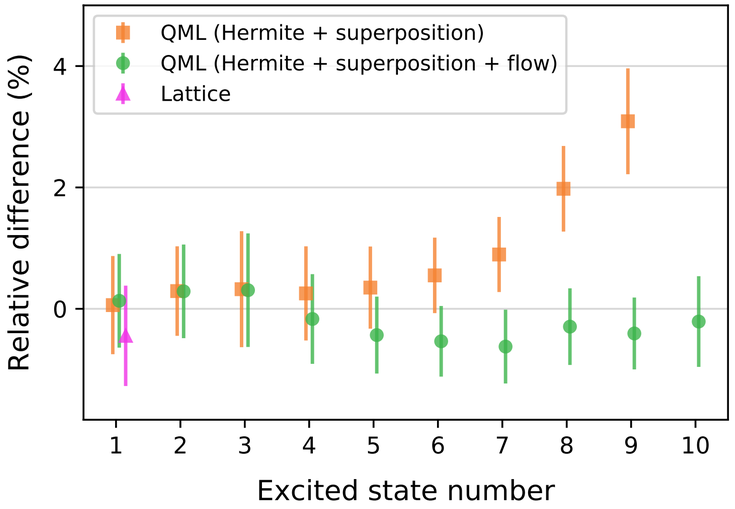


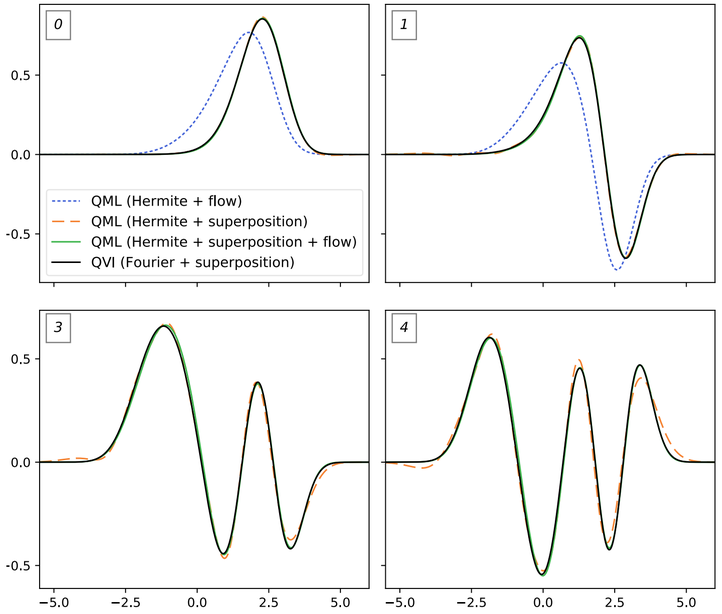


arXiv:1904.05903

Able to accurately model states with fewer Hermite components with flow

QML (Hermite + flow)
QML (Hermite + superposition)
QML (Hermite + superposition + flow)
QVI (Fourier + superposition)





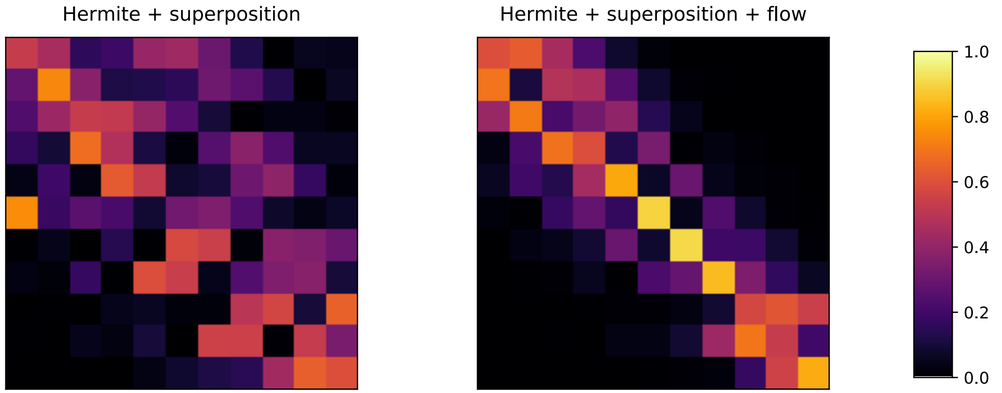


FIG. 2. The magnitude of the expansion coefficients $\tilde{a}_{j,n}$ of the putative eigenstates expanded in terms of the Hermite functions before (left) and after (right) implementing quantum flows.