Emulating Cosmological Likelihoods with Machine Learning

MSc in Physics of Data

MSc Candidate: Marco Giunta
Supervisor: Michele Liguori (UniPD)
Co-Supervisor: Marco Raveri (UniGE)
7 September 2023

Università degli Studi di Padova

Overview

1 Inference Acceleration via Emulators in Cosmology

2 CosmoLIME: A New Approach To Emulation

3 Example: Discovering Dark Energy

4 Conclusions And Future Work

Overview

Università degli Studi di Padova

1 Inference Acceleration via Emulators in Cosmology

2 CosmoLIME: A New Approach To Emulation

3 Example: Discovering Dark Energy

4 Conclusions And Future Work

CMB Temperature Anisotropy Field

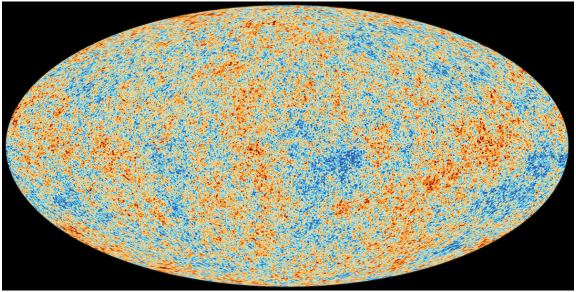

Figure: CMB temperature anisotropy field.

CMB Temperature Anisotropy Field

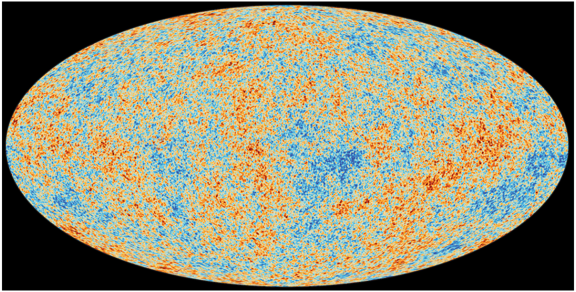

Figure: CMB temperature anisotropy field.

- Temperature differences are due to quantum fluctuations in the early universe, and are thus random in nature

CMB Temperature Anisotropy Field

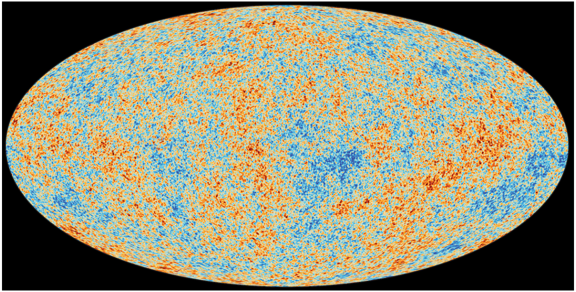

Figure: CMB temperature anisotropy field.

- Temperature differences are due to quantum fluctuations in the early universe, and are thus random in nature
- Their statistical properties are described by the *CMB power spectrum* $C_\ell(\theta)$, a compressed equivalent representation

CMB Power Spectrum

By computing the power spectrum for parameters $\tilde{\theta}$ and comparing it with the *observed* spectrum we can evaluate how likely $\tilde{\theta}$ is:

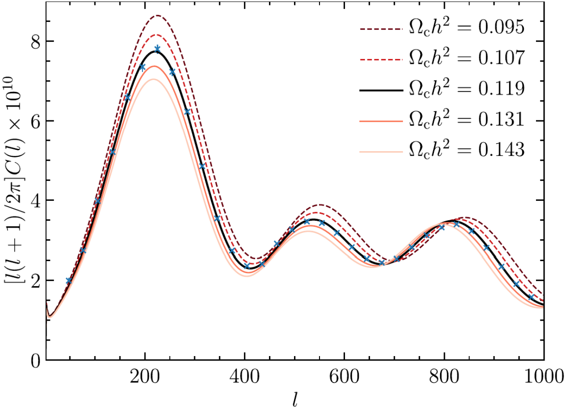

Figure: Comparison between observed and theoretical CMB power spectrum for multiple CDM density values.

CMB-based Inference

To use the CMB to obtain a Bayesian posterior over θ values we need to evaluate how likely each theoretical power spectrum is given the observed one.

To use the CMB to obtain a Bayesian posterior over θ values we need to evaluate how likely each theoretical power spectrum is given the observed one.
The CMB field is a *Gaussian Process*:

$$\ln \mathcal{L}(D|\theta) \propto (\hat{C}_\ell(D) - C_\ell(\theta))^T \Sigma^{-1} (\hat{C}_\ell(D) - C_\ell(\theta))$$

To use the CMB to obtain a Bayesian posterior over θ values we need to evaluate how likely each theoretical power spectrum is given the observed one.
The CMB field is a *Gaussian Process*:

$$\ln \mathcal{L}(D|\theta) \propto (\boxed{\hat{C}_\ell(D)} - C_\ell(\theta))^T \Sigma^{-1} (\hat{C}_\ell(D) - C_\ell(\theta))$$

- $\boxed{\hat{C}_\ell(D)}$: dataset dependent (fixed during posterior sampling)

CMB-based Inference

To use the CMB to obtain a Bayesian posterior over θ values we need to evaluate how likely each theoretical power spectrum is given the observed one.
The CMB field is a *Gaussian Process*:

$$\ln \mathcal{L}(D|\theta) \propto \left(\hat{C}_\ell(D) - C_\ell(\theta) \right)^T \Sigma^{-1} \left(\hat{C}_\ell(D) - C_\ell(\theta) \right)$$

- $\hat{C}_\ell(D)$: dataset dependent (fixed during posterior sampling)
- $C_\ell(\theta)$: parameter dependent (variable during posterior sampling)

- To infer the value of cosmological parameters we need to compare an experimentally measured quantity (e.g. $\hat{C}_\ell(D)$) with the value of that same quantity predicted theoretically assuming a certain choice of the cosmological parameters/model (e.g. $C_\ell(\theta)$); *this is a general property of cosmological inferences, i.e. it always holds.*

Cosmological Emulators

- To infer the value of cosmological parameters we need to compare an experimentally measured quantity (e.g. $\hat{C}_\ell(D)$) with the value of that same quantity predicted theoretically assuming a certain choice of the cosmological parameters/model (e.g. $C_\ell(\theta)$); *this is a general property of cosmological inferences, i.e. it always holds.*
- Theoretical predictions (e.g. $C_\ell(\theta)$) can be computed exactly using expensive specialized solvers, which are usually the computational bottleneck in inference pipelines

Cosmological Emulators

- To infer the value of cosmological parameters we need to compare an experimentally measured quantity (e.g. $\hat{C}_\ell(D)$) with the value of that same quantity predicted theoretically assuming a certain choice of the cosmological parameters/model (e.g. $C_\ell(\theta)$); *this is a general property of cosmological inferences, i.e. it always holds.*
- Theoretical predictions (e.g. $C_\ell(\theta)$) can be computed exactly using expensive specialized solvers, which are usually the computational bottleneck in inference pipelines
- For this reason it makes sense to train an *emulator*, i.e. a machine learning model that replaces the expensive exact computation; this is possible because e.g. the mapping $\theta \mapsto C_\ell$ is smooth

Cosmological Emulators

- To infer the value of cosmological parameters we need to compare an experimentally measured quantity (e.g. $\hat{C}_\ell(D)$) with the value of that same quantity predicted theoretically assuming a certain choice of the cosmological parameters/model (e.g. $C_\ell(\theta)$); *this is a general property of cosmological inferences, i.e. it always holds.*
- Theoretical predictions (e.g. $C_\ell(\theta)$) can be computed exactly using expensive specialized solvers, which are usually the computational bottleneck in inference pipelines
- For this reason it makes sense to train an *emulator*, i.e. a machine learning model that replaces the expensive exact computation; this is possible because e.g. the mapping $\theta \mapsto C_\ell$ is smooth
- Common procedure: train an emulator valid under certain assumptions, then publish it

In order to use a pre-existing emulator to accelerate parameter inference several requirements must be met:

Shortcomings of the Out-of-the-box Approach

In order to use a pre-existing emulator to accelerate parameter inference several requirements must be met:

- appropriate cosmological likelihood/model

Shortcomings of the Out-of-the-box Approach

In order to use a pre-existing emulator to accelerate parameter inference several requirements must be met:

- appropriate cosmological likelihood/model
- required accuracy

Shortcomings of the Out-of-the-box Approach

In order to use a pre-existing emulator to accelerate parameter inference several requirements must be met:

- appropriate cosmological likelihood/model
- required accuracy
- suitable ML model/preprocessing

Shortcomings of the Out-of-the-box Approach

In order to use a pre-existing emulator to accelerate parameter inference several requirements must be met:

- appropriate cosmological likelihood/model
- required accuracy
- suitable ML model/preprocessing

Shortcomings of the Out-of-the-box Approach

In order to use a pre-existing emulator to accelerate parameter inference several requirements must be met:

- appropriate cosmological likelihood/model
- required accuracy
- suitable ML model/preprocessing

If any of these conditions do not hold then a custom emulator must be trained from scratch; this means trading *computer time* for *human time*, which negates the time benefit of using emulators...

Shortcomings of the Out-of-the-box Approach

In order to use a pre-existing emulator to accelerate parameter inference several requirements must be met:

- appropriate cosmological likelihood/model
- required accuracy
- suitable ML model/preprocessing

If any of these conditions do not hold then a custom emulator must be trained from scratch; this means trading *computer time* for *human time*, which negates the time benefit of using emulators...

...unless the process of implementing an emulator can be *automated*!

Shortcomings of the Out-of-the-box Approach

In order to use a pre-existing emulator to accelerate parameter inference several requirements must be met:

- appropriate cosmological likelihood/model
- required accuracy
- suitable ML model/preprocessing

If any of these conditions do not hold then a custom emulator must be trained from scratch; this means trading *computer time* for *human time*, which negates the time benefit of using emulators...

...unless the process of implementing an emulator can be *automated*! In particular we propose a *change of paradigm*: we replace prebuilt emulators with DIY ones, which are to be trained using a standardized, fully automated procedure

Overview

1 Inference Acceleration via Emulators in Cosmology

2 CosmoLIME: A New Approach To Emulation

3 Example: Discovering Dark Energy

4 Conclusions And Future Work

We introduce CosmoLIME, the *Cosmological Likelihood Machine learning Emulator*, to automate the process of building custom emulators.

Relevant features include:

Introducing CosmoLIME

Università degli Studi di Padova

We introduce CosmoLIME, the *Cosmological Likelihood Machine learning Emulator*, to automate the process of building custom emulators.

Relevant features include:

- fully automated data generation and ML model training

Introducing CosmoLIME

We introduce CosmoLIME, the *Cosmological Likelihood Machine learning Emulator*, to automate the process of building custom emulators.

Relevant features include:

- fully automated data generation and ML model training
- support for arbitrary likelihood functions/cosmological models

Introducing CosmoLIME

We introduce CosmoLIME, the *Cosmological Likelihood Machine learning Emulator*, to automate the process of building custom emulators.

Relevant features include:

- fully automated data generation and ML model training
- support for arbitrary likelihood functions/cosmological models
- support for arbitrary ML models/preprocessing

Introducing CosmoLIME

We introduce CosmoLIME, the *Cosmological Likelihood Machine learning Emulator*, to automate the process of building custom emulators.

Relevant features include:

- fully automated data generation and ML model training
- support for arbitrary likelihood functions/cosmological models
- support for arbitrary ML models/preprocessing
- support for automated testing

Introducing CosmoLIME

Università degli Studi di Padova

We introduce CosmoLIME, the *Cosmological Likelihood Machine learning Emulator*, to automate the process of building custom emulators.

Relevant features include:

- fully automated data generation and ML model training
- support for arbitrary likelihood functions/cosmological models
- support for arbitrary ML models/preprocessing
- support for automated testing
- other useful tools (logging, model caching, etc.)

Implementing An Emulator With CosmoLIME

In general training an emulator from scratch is a three-step process:

Implementing An Emulator With CosmoLIME

In general training an emulator from scratch is a three-step process:

- obtain a suitable training dataset

Implementing An Emulator With CosmoLIME

In general training an emulator from scratch is a three-step process:

- obtain a suitable training dataset
- train the ML model of choice

Implementing An Emulator With CosmoLIME

Università degli Studi di Padova

In general training an emulator from scratch is a three-step process:

- obtain a suitable training dataset
- train the ML model of choice
- test the result via accuracy metrics/statistical tests

Implementing An Emulator With CosmoLIME

In general training an emulator from scratch is a three-step process:

- obtain a suitable training dataset
- train the ML model of choice
- test the result via accuracy metrics/statistical tests

Implementing An Emulator With CosmoLIME

In general training an emulator from scratch is a three-step process:

- obtain a suitable training dataset
- train the ML model of choice
- test the result via accuracy metrics/statistical tests

A training dataset of arbitrarily many, noise-free samples can be obtained using the available exact solvers; the remaining tasks are solved as usual in regression problems. This means that CosmoLIME must automate both the data generation and the model optimization.

CosmoLIME Software Hierarchy

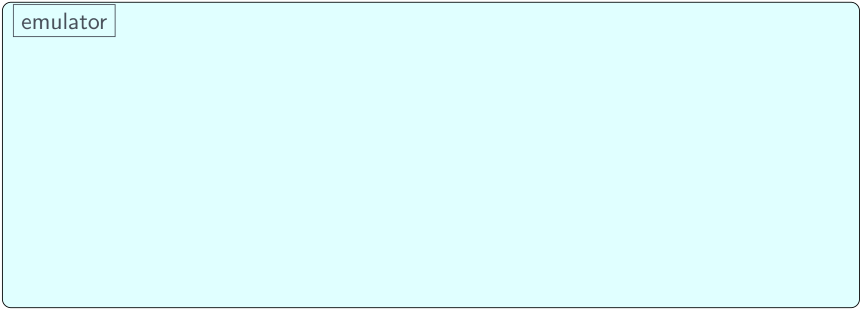

Figure: Schematic representation of CosmoLIME.

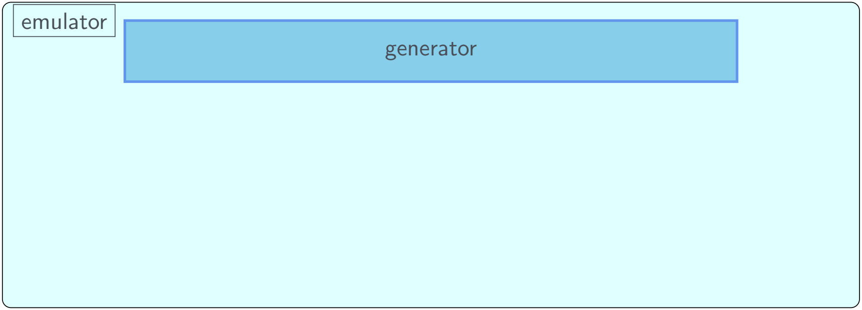

Figure: Schematic representation of CosmoLIME.

CosmoLIME Software Hierarchy

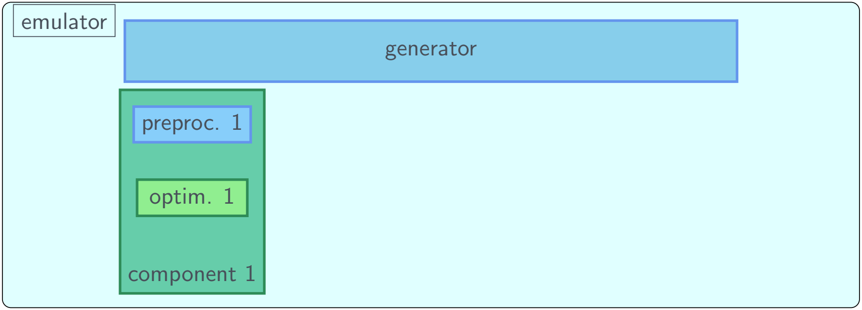

Figure: Schematic representation of CosmoLIME.

CosmoLIME Software Hierarchy

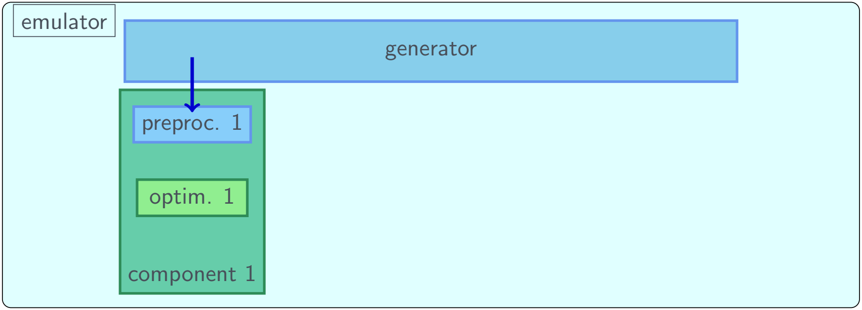

Figure: Schematic representation of CosmoLIME.

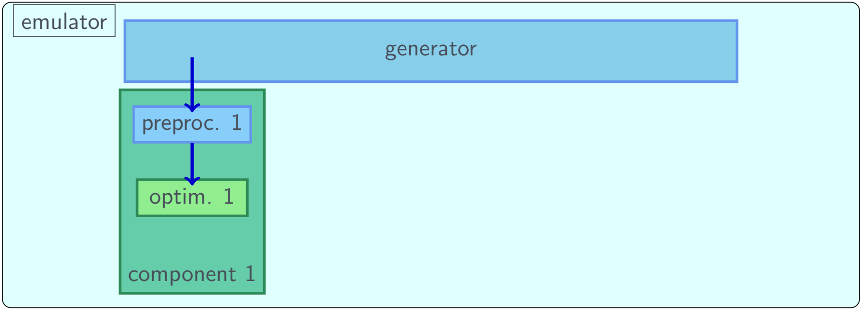

Figure: Schematic representation of CosmoLIME.

CosmoLIME Software Hierarchy

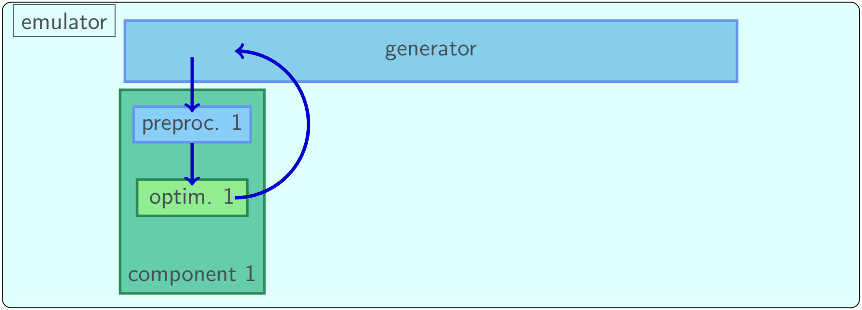

Figure: Schematic representation of CosmoLIME.

CosmoLIME Software Hierarchy

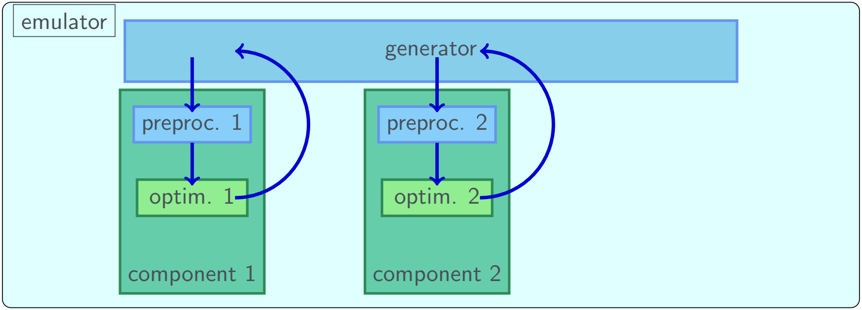

Figure: Schematic representation of CosmoLIME.

CosmoLIME Software Hierarchy

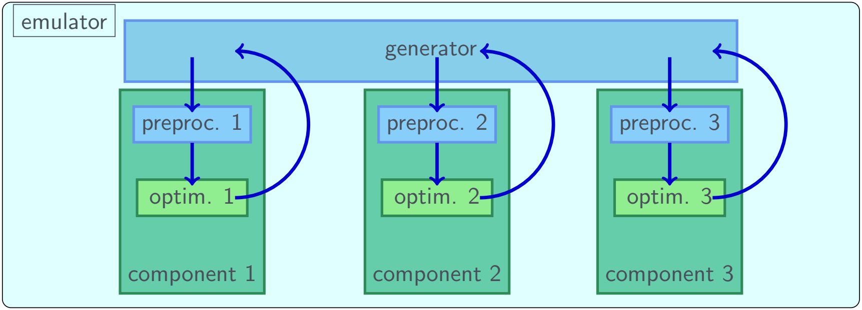

Figure: Schematic representation of CosmoLIME.

Overview

1 Inference Acceleration via Emulators in Cosmology

2 CosmoLIME: A New Approach To Emulation

3 Example: Discovering Dark Energy

4 Conclusions And Future Work

Dark Energy Simplified Model

As a simple application we can prove the existence of dark energy using CosmoLIME. We use the standard Λ-CDM cosmological model with all parameters fixed to their fiducial values, except for *relative matter density* and *relative dark energy density*:

$$\Omega_m \equiv \frac{\rho_m}{\rho_{\text{tot}}}, \quad \Omega_{\text{DE}} \equiv \frac{\rho_{\text{DE}}}{\rho_{\text{tot}}} \quad \text{with} \quad \Omega_m + \Omega_{\text{DE}} = 1$$

Dark Energy Simplified Model

As a simple application we can prove the existence of dark energy using CosmoLIME. We use the standard Λ-CDM cosmological model with all parameters fixed to their fiducial values, except for *relative matter density* and *relative dark energy density*:

$$\Omega_m \equiv \frac{\rho_m}{\rho_{\text{tot}}}, \quad \Omega_{\text{DE}} \equiv \frac{\rho_{\text{DE}}}{\rho_{\text{tot}}} \quad \text{with} \quad \Omega_m + \Omega_{\text{DE}} = 1$$

Then the problem of whether dark energy exists becomes equivalent to asking whether $\Omega_m = 1$ or $\Omega_m \neq 1$; this means we have a simple 1-parameter model, making inference trivial.

In this simplified model the Ω_m parameter is the only quantity influencing the *luminosity distances of type Ia supernovae*, which are normally distributed around:

$$D_L(z) = D_0(1+z)\int_0^z \frac{\mathrm{d}x}{\sqrt{\Omega_m(1+x)^3 + \Omega_{\mathrm{DE}}}}$$

$$\mu(z) \equiv 5\log_{10} D_L(z)$$

Type Ia Supernovae

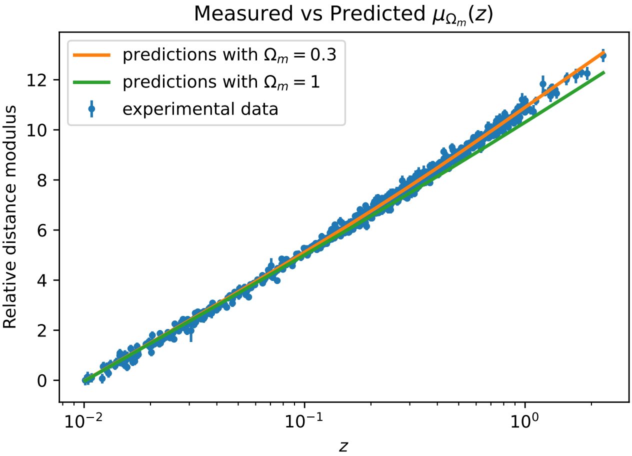

Figure: Observed/predicted $\mu(z)/\mu(0)$ assuming $\Omega_m = 1$ or $\Omega_m = 0.3$.

Inferring Ω_m

UNIVERSITÀ DEGLI STUDI DI PADOVA

By observing an experimental dataset $\{z_n, \hat{\mu}(z_n)\}$ we perform inference by exploiting the fact that *distances are distributed according to a Gaussian*:

$$\ln \mathcal{L}(\hat{\mu}|\mu, \Sigma) \propto (\mu - \hat{\mu})^T \Sigma^{-1} (\mu - \hat{\mu})$$

Inferring Ω_m

By observing an experimental dataset $\{z_n, \hat{\mu}(z_n)\}$ we perform inference by exploiting the fact that *distances are distributed according to a Gaussian*:

$$\ln \mathcal{L}(\hat{\mu}|\mu, \Sigma) \propto (\mu - \hat{\mu})^T \Sigma^{-1} (\mu - \hat{\mu})$$

Marginalizing w.r.t. nuisance parameter D_0:

$$\ln \mathcal{L}_m(\hat{\mu}|\mu, \Sigma) \propto -\frac{1}{2}(\mu - \hat{\mu})^T \Sigma^{-1} (\mu - \hat{\mu}) + \frac{1}{2} \frac{\left[(1)\Sigma^{-1}(\hat{\mu} - \mu)\right]^2}{(1)\Sigma^{-1}(1)}$$

By emulating this likelihood with CosmoLIME and using a uniform prior for simplicity we can obtain the Ω_m posterior.

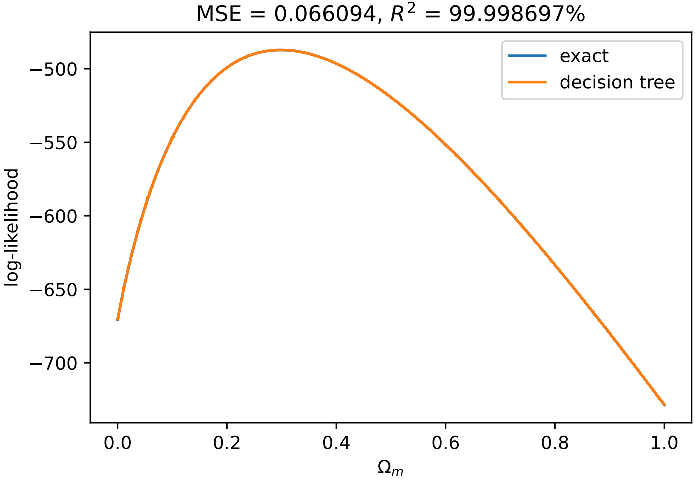

Figure: Exact vs emulated marginalized log-likelihood.

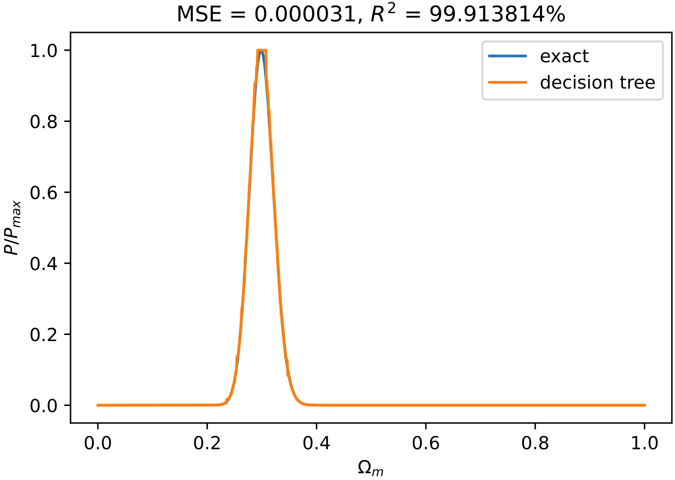

Figure: Exact vs emulated Ω_m posterior.

Inference Results

We notice that the posterior has a strong peak in $\Omega_m \approx 0.3$, thus implying that about $\sim 70\%$ of the "stuff" in our universe is dark energy. Rigorous probabilistic statements can be obtained by normalizing the posterior (e.g. via direct numerical integration in this simple example).
In particular we find

$$\Omega_m = 0.298 \pm 0.043 \text{ (MAP} \pm 68\% \text{ cred. int.)}$$

without significant differences between using the exact or emulated posterior, and in good agreement with the currently accepted value.

Inference Results

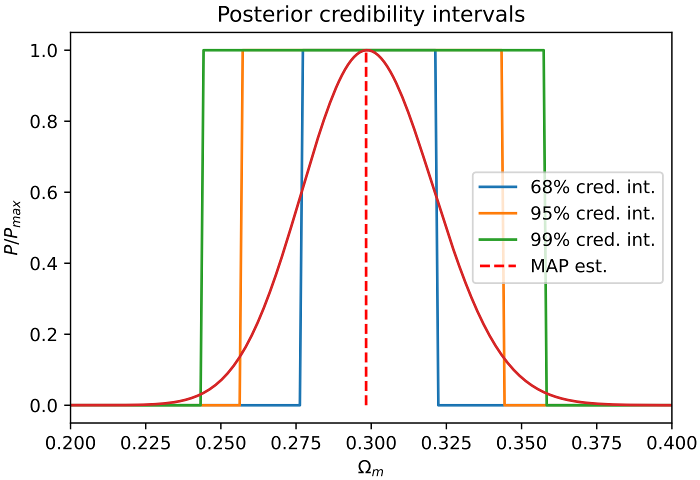

Figure: Posterior credibility intervals.

Bayesian Model Comparison

Another way to infer the existence of dark energy from this data is to perform *Bayesian model comparison* between model M_0 ($\Omega_m = 1$ exactly) and model M_1 ($\Omega_m \in [0, 1]$) using e.g. a uniform prior on models:

Bayesian Model Comparison

Another way to infer the existence of dark energy from this data is to perform *Bayesian model comparison* between model M_0 ($\Omega_m = 1$ exactly) and model M_1 ($\Omega_m \in [0,1]$) using e.g. a uniform prior on models:

$$\frac{P(M_1|D)}{P(M_0|D)} = \frac{P(D|M_1)}{P(D|M_0)} \equiv \frac{E_1}{E_0}$$

Bayesian Model Comparison

Another way to infer the existence of dark energy from this data is to perform *Bayesian model comparison* between model M_0 ($\Omega_m = 1$ exactly) and model M_1 ($\Omega_m \in [0, 1]$) using e.g. a uniform prior on models:

$$\frac{P(M_1|D)}{P(M_0|D)} = \frac{P(D|M_1)}{P(D|M_0)} \equiv \frac{E_1}{E_0}$$

E_1 is the evidence of the Ω_m posterior; $E_0 = P(\Omega_m = 1|D)$, i.e. E_0 equals the posterior in $\Omega_m = 1$.

Bayesian Model Comparison

Another way to infer the existence of dark energy from this data is to perform *Bayesian model comparison* between model M_0 ($\Omega_m = 1$ exactly) and model M_1 ($\Omega_m \in [0, 1]$) using e.g. a uniform prior on models:

$$\frac{P(M_1|D)}{P(M_0|D)} = \frac{P(D|M_1)}{P(D|M_0)} \equiv \frac{E_1}{E_0}$$

E_1 is the evidence of the Ω_m posterior; $E_0 = P(\Omega_m = 1|D)$, i.e. E_0 equals the posterior in $\Omega_m = 1$. With or without the emulator we find:

$$\ln(E_1/E_0) = \ln E_1 - \ln E_0 \approx 239$$

which means that *a universe with dark energy is strongly preferred even accounting for the Occam penalty (at least according to this data)*.

Overview

1 Inference Acceleration via Emulators in Cosmology

2 CosmoLIME: A New Approach To Emulation

3 Example: Discovering Dark Energy

4 Conclusions And Future Work

Thesis Summary

- Cosmological emulators are ML models that can be used to accelerate inference pipelines by replacing the expensive exact evaluations; unless a suitable prebuilt emulator is available this requires training the model from scratch, which can ruin the chance to actually save time to perform inferences.

Thesis Summary

- Cosmological emulators are ML models that can be used to accelerate inference pipelines by replacing the expensive exact evaluations; unless a suitable prebuilt emulator is available this requires training the model from scratch, which can ruin the chance to actually save time to perform inferences.
- To solve this we proposed a change of paradigm: from prebuilt to DIY emulators. This is achieved with CosmoLIME, the *Cosmological Likelihood Machine learning Emulator*: a model-agnostic, self-training, machine learning-based framework to emulate arbitrary likelihood functions in a fully automated way.

Thesis Summary

- Cosmological emulators are ML models that can be used to accelerate inference pipelines by replacing the expensive exact evaluations; unless a suitable prebuilt emulator is available this requires training the model from scratch, which can ruin the chance to actually save time to perform inferences.
- To solve this we proposed a change of paradigm: from prebuilt to DIY emulators. This is achieved with CosmoLIME, the *Cosmological Likelihood Machine learning Emulator*: a model-agnostic, self-training, machine learning-based framework to emulate arbitrary likelihood functions in a fully automated way.
- To design such a framework we simply need to automate a slightly modified version of the standard regression problem; we applied these results to simple but realistic examples.

Future Challenges

Even though CosmoLIME already works several features may be added to make it truly compelling to the astrophysical community:

Future Challenges

Even though CosmoLIME already works several features may be added to make it truly compelling to the astrophysical community:

- CosmoLIME has yet to be tested on serious problems; a good starting point would be to reproduce prebuilt research-ready emulators, showing how they can be reproduced with minimum effort.

Future Challenges

Università degli Studi di Padova

Even though CosmoLIME already works several features may be added to make it truly compelling to the astrophysical community:

- CosmoLIME has yet to be tested on serious problems; a good starting point would be to reproduce prebuilt research-ready emulators, showing how they can be reproduced with minimum effort.
- More statistically sound accuracy metrics are needed; simple ML metrics can lead to overconfident models, whose optimization is prematurely stopped and results in e.g. biased posteriors.

Future Challenges

Even though CosmoLIME already works several features may be added to make it truly compelling to the astrophysical community:

- CosmoLIME has yet to be tested on serious problems; a good starting point would be to reproduce prebuilt research-ready emulators, showing how they can be reproduced with minimum effort.
- More statistically sound accuracy metrics are needed; simple ML metrics can lead to overconfident models, whose optimization is prematurely stopped and results in e.g. biased posteriors.
- Further technical improvements can make CosmoLIME more enticing (better logging, complete parallelization, etc.).

Future Challenges

Università degli Studi di Padova

Even though CosmoLIME already works several features may be added to make it truly compelling to the astrophysical community:

- CosmoLIME has yet to be tested on serious problems; a good starting point would be to reproduce prebuilt research-ready emulators, showing how they can be reproduced with minimum effort.
- More statistically sound accuracy metrics are needed; simple ML metrics can lead to overconfident models, whose optimization is prematurely stopped and results in e.g. biased posteriors.
- Further technical improvements can make CosmoLIME more enticing (better logging, complete parallelization, etc.).

Achieving these points can turn CosmoLIME into a publication-ready framework.