

Введение в SBI

30 января 2026 г.

Содержание

1	Знакомьтесь: Simulation Based Inference	3
2	SBI-software	6
2.1	ELFI	6
2.2	BayesFlow	6
2.3	LtU-ILI	6
2.4	Библиотеки sbi, pydelfi и lampe	7
3	Approximate Bayesian Computation и его особенности	7
3.1	Руководство по ABC	8
3.2	Синтетическое правдоподобие	10
4	Нейросетевые SBI	11
4.1	Статья [Cranmer2020]	11
4.1.1	Симуляторы и статистический вывод	12
4.1.2	Машинное обучение, активное обучение и интеграция	14
4.1.3	Рабочие процессы SBI	17
4.1.4	Суррогатные модели	19
4.1.5	Пре- и постобработка	20
4.2	Статья [Lueckmann2021]	22
4.2.1	Алгоритмы	22
4.2.2	Метрики	24
4.2.3	Задачи	27
4.2.4	Результаты	28
4.3	Статья [Ho2024]	28
4.3.1	Валидация метода	33
4.3.2	Конвейер LtU-ILI	34
4.3.3	Модельные эксперименты	37
4.3.4	Научные эксперименты	39
4.3.5	Дискуссия и выводы	39
4.4	Статья [Gloeckler2024]	40
4.4.1	Трансформеры, механизм внимания и диффузионные модели	41
4.4.2	Simformer	42

5	Model Misspecification	45
5.1	Статья [Schmitt2021]	45
5.2	Статья [Cannon2022]	46
5.3	Статья [Hermans2022]	46
5.4	Статья [Pierre2025]	46
5.5	Статья [Kelly2025]	47
6	Simulation-Based Calibration и смежные вопросы	47
6.1	Статья [Talts2020]	47
7	Байесовское принятие решений	48
7.1	Статья [Gorecki2023]	48
7.2	Статья [Alsing2023]	48
8	Заметки на полях...	49
8.1	Про ABC	49
A	Полезные примеры	50
A.1	Оценка массы скоплений галактик по рентгеновским изображениям	50
A.2	Оценка параметров темной материи по спектру мощности	51
A.3	Оценка параметров темной материи по полевым данным	52
A.4	Гравитационные волны в результате слияния черных дыр	53
A.5	Фотометрическое исследование межзвездной пыли	55
A.6	Галактический ветер: массовая и энергетическая нагрузка	56
A.7	Реконструкция направления прилета UHECR по радиосигналам	58
A.8	Примеры из нейронауки	59
A.8.1	Модель рецептивных полей	59
A.8.2	Модель ионных каналов	59
A.8.3	Модель Ходжкина-Хаксли	59

1 Знакомьтесь: Simulation Based Inference

Цитата из [Ho2024]:

В течение почти столетия практика построения линейных или пертурбативных физических моделей на основе первых принципов позволила добиться существенного прогресса в понимании Вселенной. Тем не менее, как подчеркивается в 2020 Decadal Survey (National Academies of Sciences, Engineering and Medicine 2023), изучение сложного нелинейного режима физических явлений с помощью выводов, основанных на данных (data-driven inference), обещает значительный выигрыш в ограничивающей силе. Большой объем данных в исследованиях нового поколения, усовершенствование моделирования с высоким разрешением и стремительное развитие методов ML привели к резкому росту интереса к тому, как можно автоматически и быстро изучать сложные физические явления.

Из заголовка рабочего совещания [PHYSTAT-SBI 2024](#):

Благодаря последним достижениям в области ML за последнее десятилетие было разработано новое поколение методов решения задач статистического вывода в «likelihood-free» случаях, когда генерация данных возможна (например, с помощью стохастических симуляторов), но оценка плотности распределения в явном виде недоступна. Эта группа методов известна как simulation-based inference (SBI) или likelihood-free inference (LFI) [или Implicit Likelihood Inference (ILI)].

SBI подразумевает наличие симулятора – стохастического генератора, который для заданных значений параметров модели θ генерит данных x . Несмотря на отсутствие аналитически выраженной функции правдоподобия $p(x|\theta)$ симулятор позволяет провести вероятностный (байесовский) вывод (inference), аппроксимируя тем или иным способом либо непосредственно постериорное распределение $p(\theta|x)$, либо функцию правдоподобия $p(x|\theta)$. Зачастую, помимо самого симулятора, в таком подходе требуется задание способа сэмплирования параметров θ – в литературе он получил название proposal distribution (или просто proposal, оно может отличаться от априорного распределения, используемого непосредственно в байесовском выводе).

Одним из наиболее распространенных методов SBI является ABC – Approximate Bayesian Computation, см. например на WIKI: [Approximate Bayesian computation](#) или в статье [Marin2012], [ABC methods](#). Различают варианты ABC, использующие в качестве proposal distribution прайоры (REJ-ABC, от rejection sampling), и методы, последовательно улучшающие proposal (SMC-ABC, от sequential MC). Подробнее об ABC см. в разделе 3 этого документа.

Наряду с ABC в практике научных исследований активно (на самом деле – куда более активно, чем ABC!) используют методы SBI, оценивающие плотности распределений (DE, Density Estimations). Выделяют следующие классы методов SBI с DE (см. рис. 1, взятый из [Lueckmann2021]): Likelihood Estimation, Posterior Estimation, Ratio Estimation. Современные варианты с оценкой плотности включают аппроксимацию на основе *нейросетей* и поэтому получили названия: (S)NLE, (S)NPE, (S)NRE (здесь и далее S означает вариант алгоритма с последовательным улучшением proposal distribution).

Алгоритмы SBI также можно разделить на группы в зависимости от того, как они представляют выходные данные: 1) некоторые возвращают выборки из (приближенного) апостериорного распределения $\theta \sim q(\theta|x_{\text{obs}})$ (REJ-ABC, SMC-ABC); 2) другие не только возвращают выборки, но и позволяют оценить ненормированные апостериорные распределения $\tilde{q}(\theta|x_{\text{obs}})$ ((S)NLE, (S)NRE); 3) в некоторых случаях апостериорную плотность $q(\theta|x_{\text{obs}})$ можно оценить и получить выборку напрямую, без MCMC ((S)NPE). Подроб-

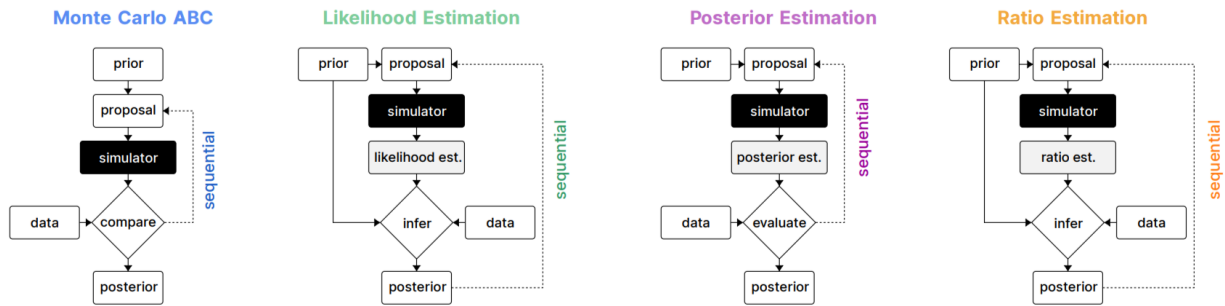


Рис. 1: Четыре различных подхода к SBI: классические ABC-методы и подходы, основанные на моделях, аппроксимирующих правдоподобие, апостериорное распределение или отношение плотностей. Алгоритмы, использующие прайоры, противопоставлены алгоритмам, которые последовательно адаптируют proposal (пунктир).

нее см. в статье [Cranmer2020] *The frontier of SBI* или в [Ho2024] *LtU-ILI: An All-in-One Framework for Implicit Inference in Astrophysics and Cosmology*. Полноценный обзор существующих SBI-методов (с формулировкой соответствующих алгоритмов) приведен в работе [Lueckmann2021] *Benchmarking Simulation-Based Inference*. Краткий анализ этих работ см. в разделе 4.

подавляющее большинство нейросетевых SBI используют для оценки так называемые Masked Autoregressive Flow (MAF), подробно описанные в цикле работ George Papamakarios. Все эти работы вошли в его PhD Thesis *Neural Density Estimation and Likelihood-free Inference*. В частности во введении к главе 2 диссертации перечисляются разнообразные алгоритмы ABC: Rejection ABC, Likelihood-based Metropolis-Hastings, Pseudo-marginal Metropolis-Hastings, Markov-chain Monte Carlo ABC, Importance-sampling ABC, Sequential Monte Carlo ABC.

Еще одну весьма перспективную альтернативу представляет собой относительно новый амортизированный метод Simformer, использующего для вывода нейросетевую архитектуру Transformer. С этим методом можно познакомиться по работе [Gloeckler2024] *All-in-one simulation-based inference*, частично разобранный нами в разделе 4.4.

Несмотря на такое стремительное распространение SBI-методов в науке, появляется и ряд работ, критикующих неосторожное их использование и выявляющих их существенные ограничения. Для ознакомления рекомендуем работу [Hermans2022] *A Trust Crisis In Simulation-Based Inference? Your Posterior Approximations Can Be Unfaithful*. В работе [Cannon2022] *Investigating the Impact of Model Misspecification in Neural Simulation-based Inference* исследуется вопрос к чему приводит SBI при неточной спецификации модели, т.е. в ситуациях, когда генерационная и интерпретационные модели не совпадают. Краткий обзор этих и других работ см. в разделе 5.

Еще одной важной темой, связанной с симуляторами и байесовским выводом, является так называемая *калибровка, основанная на моделировании* (SBC, simulation based inference). Познакомиться с этим направлением можно по статье [Talts2018] *Validating Bayesian Inference Algorithms with Simulation-Based Calibration*. Разбор этой (и некоторых других) работы приведен в разделе 6.

Существует большое число превосходных работ, посвященных применению SBI в научном исследовании. Примеры успешного применения ILI в астрономии и космологии (ссылки доступны в [Ho2024]): 1) cosmological galaxy lensing and clustering [Jeffrey2021]; [Makinen2021, 2022a]; [deSanti2023a]; [Hahn2023a], 2) gravitational waves [Dax2021a]; [Cheung2022], 3) galaxy cluster mass estimation [Ho2022]; [deAndres2022], 4) galaxy morphology [Walmsley2020];

[Ghosh2022], 5) stellar streams [Hermans2021]; [Alvey2023], 6) exoplanets [Rogers2023]; [Aubin2023]. В Приложении A мы приводим обзор некоторых наиболее интересных с педагогической точки зрения. В частности, там подробно разобраны примеры из [Ho2024] при демонстрации эффективности работы с пакетом LtU-ILL.

Амортизированные нейросетевые оценки постериоров используются также в цикле работ Keming Zhang, этот подход получил название Amortized Neural Posterior Estimation (ANPE). Особый интерес для наших образовательных целей могут представлять статьи [Zhang2021] [Real-time LFI of Roman Binary Microlensing Events with ANPE](#) и [Zhang2023] [Stellar Spectra Fitting with ANPE and nbi](#). Специально для решения этих задач был разработан пакет [nbi](#) (от Neural Bayesian Inference), см. его краткое описание в работе [Zhang2023a] [nbi: the Astronomer’s Package for Neural Posterior Estimation](#).

Полезные ссылки по SBI и смежным темам (список будет пополняться):

- Сайт Kyle Cranmer and Jason Lo [Simulation-based inference](#): ресурс, где собраны последние научные статьи о методологических разработках и приложениях в области BI (Bayesian Inference) и SBI.
- Сайт [Awesome Neural SBI](#): список статей и ресурсов по BI на основе нейронного моделирования, охватывающий как методологические разработки, так и приложения из предметной области. На сайте также есть лекция [SBI Tutorial](#), оформленная в виде jupyter-ноутбука.
- Обзоры по SBI: [Cranmer2020] [The frontier of SBI](#) и [Zammit-Mangion2024] [Neural Methods for Amortized Inference](#).
- Глава из книги M.Osvaldo (html-версия): [Approximate Bayesian Computation](#).
- Документация и примеры PyMC: [ABC Simulator, Using a “black box” likelihood function, SMC-ABC and Lotka–Volterra,...](#)
- Рабочее совещание [PHYSTAT-SBI 2024](#), посвященное SBI in Fundamental Physics (с доступными для скачивания презентациями!).
- Группа Торстена Исслина (Torsten Esslin) [Information Filed Theory group](#): их основной подход – продвинутые версии VI. В частности, команда разработала программный пакет [NIFTy](#) (и его ускоренную версию под JAX – [NIFTy.re](#)).
- [SMT](#): Surrogate Modeling Toolbox.
- Сайт [A Living Review of Machine Learning for Particle Physics](#): исчерпывающий (и обновляющийся!) список ссылок с рубрикаторм для тех, кто разрабатывает и применяет современные ML-подходы к экспериментальному, феноменологическому или теоретическому анализу в HEP.
- Про особую трактовку байесовского вывода как вероятностного программирования см. статью [Wood2015] [A New Approach to Probabilistic Programming Inference](#).

2 SBI-software

Большой [список программного обеспечения и бенчмарка по SBI](#) (со ссылками на сам код, документацию и поясняющую статью) приведен на сайте [Awesome Neural SBI](#). В частности, здесь можно найти информацию по пакетам: `sbi`; `BayesFlow`; `sbibm`; `swyft`; `NeuralEstimators.jl`; `SimulationBasedInference.jl`; `sbi-ax`; `lampe`; `sbijax`; `nbi`; `MadMiner`; `pydelfi`; `carl`.

2.1 ELFI

Engine for Likelihood-Free Inference (ELFI) includes an easy to use generative modeling syntax, where the generative model is specified as a directed acyclic graph (DAG). This provides an intuitive means to describe rather complex dependencies conveniently.

- [Статья \[Lintusaari2018\] ELFI: Engine for Likelihood-Free Inference](#).
- [Исходники](#), [Документация](#), [Tutorial](#).
- Реализованные LFI-методы: 1) ABC rejection sampler; 2) Sequential Monte Carlo ABC sampler; 3) ABC-SMC sampler with adaptive distance; 4) ABC-SMC sampler with adaptive threshold selection; 5) Bayesian Optimization for Likelihood-Free Inference (BOLFI) framework; 6) Robust Optimization Monte Carlo (ROMC) framework; 7) Bayesian Optimization for Likelihood-Free Inference by Ratio Estimation (BOLFIRE); 8) Bayesian Synthetic Likelihood (BSL).
- Дополнительные (non-LFI) методы: 1) Bayesian Optimization; 2) No-U-Turn-Sampler, a Hamiltonian Monte Carlo MCMC sampler.
- Additionally, ELFI integrates tools for visualization, model comparison, diagnostics and post-processing.

2.2 BayesFlow

BayesFlow is a Python library for simulation-based Amortized Bayesian Inference with neural networks. It provides users and researchers with: 1) A user-friendly API for rapid Bayesian workflows, 2) A rich collection of neural network architectures, 3) Multi-backend support via Keras3 (you can use PyTorch, TensorFlow, or JAX).

- [Статья \[Radev2020\] BayesFlow: Learning complex stochastic models with INN](#).
- Сайт проекта [BayesFlow](#).
- [Examples](#).

2.3 LtU-ILI

The Learning the Universe Implicit Likelihood Inference (LtU-ILI) pipeline is an all-in-one framework for performing ML parameter inference in astrophysics and cosmology. Given labeled training data $(x_i, \theta_i)_{i=1}^N$ or a stochastic simulator $x(\theta)$, LtU-ILI is designed to automatically train state-of-the-art neural networks to learn the data-parameter relationship and produce robust, well-calibrated posterior inference.

- Статья [Ho2024] LtU-ILI: An All-in-One Framework for Implicit Inference in Astrophysics and Cosmology (см. также обзор этой статьи в разделе 4.3).
- Исходники, Документация, Tutorials, Examples.
- Реализованные методы: 1) Posterior-, Likelihood-, and Ratio-Estimation methods for ILI, including Sequential learning analogs. 2) Various neural density estimators (Mixture Density Networks, Conditional Normalizing Flows, ResNet-like ratio classifiers). 3) Fully-customizable, exotic embedding networks (including CNNs and Graph Neural Networks).
- Метрики: Multiple marginal and multivariate posterior coverage metrics.
- Бэкенд и интерфейс: A unified interface for multiple ILI backends ([sbi](#), [pydelfi](#), [lampe](#)). Jupyter and command-line interfaces. A parallelizable configuration framework for efficient hyperparameter tuning and production runs.

2.4 Библиотеки `sbi`, `pydelfi` и `lampe`

`sbi` is a Python package for simulation-based inference, designed to meet the needs of both researchers and practitioners. Whether you need fine-grained control or an easy-to-use interface, `sbi` has you covered. Основная статья: [Tejero-Cantero2020] SBI - A toolkit for simulation-based inference.

`pydelfi` is Density Estimation Likelihood-Free Inference with neural density estimators and adaptive acquisition of simulations. The implemented methods are described in detail in Alsing, Charnock, Feeney and Wandelt 2019.

`lampe` is a SBI package that focuses on amortized estimation of posterior distributions, without relying on explicit likelihood functions; hence the name Likelihood-free AMortized Posterior Estimation (LAMPE). The package provides PyTorch implementations of modern amortized SBI algorithms like neural ratio estimation (NRE), neural posterior estimation (NPE) and more. Similar to PyTorch, the philosophy of LAMPE is to avoid obfuscation and expose all components, from network architecture to optimizer, to the user such that they are free to modify or replace anything they like. **In an effort to unite communities, the development of LAMPE has stopped in favor of the `sbi` project.**

- `sbi`: Статья, Исходники, Документация, Tutorials.
- `pydelfi`: Исходники, Документация, статья J.Alsing and B.Wandelt, Nuisance hardened data compression for fast likelihood-free inference.
- `lampe` (проект закрыт): Исходники, Документация.

3 Approximate Bayesian Computation и его особенности

Основная проблема ABC-методов заключается в том, что вероятность принятия (acceptance rate) экспоненциально уменьшается с ростом размерности параметров. Это требует значительно большего количества симуляций. Проблему можно смягчить, увеличив значение ϵ , но ценой этого является то, что метод будет нацелен на более широкое апостериорное распределение, чем истинное. Это ограничение делает ABC highly intractable в условиях, где проведение симуляций является вычислительно дорогостоящим.

3.1 Руководство по АВС

Подробно АВС рассматривается в книге [Handbook of Approximate Bayesian Computation](#) под редакцией Scott A. Sisson, Yanan Fan, Mark Beaumont (1-е издание 2018 г.). Несмотря на то, что ее нет в свободном доступе, большинство глав выложено в архиве (на данный момент не удалось найти только главы 2 и 11).

- [Глава 1. Overview of Approximate Bayesian Computation](#). Основная цель этой вводной главы – познакомить с LFI с интуитивной точки зрения. Различные алгоритмы АВС демонстрируются на простых примерах.
- [Глава 3. Regression approaches for Approximate Bayesian Computation](#). В главе представлены регрессионные подходы и регрессионная корректировка (regression adjustment) для АВС. Регрессионная корректировка подправляет значения параметров после сэмплирования с отбраковкой, чтобы учесть несоответствия между симуляциями и наблюдениями. Из-за «проклятия размерности» доверительные интервалы, полученные с помощью регрессионных подходов, могут быть уширены в сравнении с истинными. Представлена теорема, сравнивающая теоретические свойства апостериорных распределений, полученных с регрессионной корректировкой и без неё.
- [Глава 4. ABC Samplers](#). Подробно описаны основные идеи и алгоритмы, используемые для сэмплирования из апостериорного распределения в рамках АВС, включая методы, основанные на отбраковке/выборке по значимости, методе Монте-Карло с марковской цепью (MCMC) и последовательном методе Монте-Карло (Sequential MC).
- [Глава 5. Summary Statistics in Approximate Bayesian Computation](#). Для получения полезных результатов байесовского вывода необходимо использовать сводную статистику (summary statistic). Это связано с тем, что АВС страдает от «проклятия размерности», когда использование многомерных входных данных приводит к большим ошибкам аппроксимации. Поэтому крайне важно найти низкоразмерные сводные данные, которые будут информативны для задачи восстановления параметров или выбора модели. Рассматриваются методы, предложенные для выбора таких сводных данных с учётом последних разработок. Также обсуждаются связанные с этим теоретические результаты о «проклятии размерности» и достаточности.
- [Глава 6. Likelihood-free Model Choice](#). Помимо выявления потенциальных недостатков полученных на основе АВС апостериорных распределений основное внимание уделено использованию случайных лесов (random forest) для агрегирования сводной статистики и оценки апостериорной вероятности наиболее вероятной модели с помощью вторичных случайных лесов.
- [Глава 7. ABC and Indirect Inference](#). Косвенный вывод (Indirect Inference) – классический подход в отсутствие правдоподобия (LF), который появился раньше основных разработок АВС и предполагает моделирование интересующей нас параметрической модели для получения точечных оценок параметров. Неудивительно, что некоторые байесовские LF-подходы опираются на И. В этой главе вводится понятие И и подробно рассматривается связь между АВС и И. Особое внимание уделяется использованию вспомогательной модели с tractable функцией правдоподобия – подходу, который часто применяется в литературе по И для упрощения байесовских LF-выводов.

- [Глава 8. High-dimensional ABC](#). Подробно изложены, с примерами и иллюстрациями, основные идеи и концепции, лежащие в основе масштабирования методов ABC до более высоких размерностей.
- [Глава 9. Theoretical and methodological aspects of MCMC computations with noisy likelihoods](#). Цель главы – показать связь MCMC-метода ABC с псевдомаргинальными алгоритмами, рассмотреть существующие теоретические результаты и обсудить, как их можно применить на практике и как они могут способствовать плодотворному развитию методологии.
- [Глава 10. Asymptotics of ABC](#). Представляем неформальный обзор последних работ по асимптотическому поведению ABC. В частности, обсуждается, как ведёт себя апостериорное распределение ABC или точечные ABC-оценки в пределе больших данных. Результаты показывают, что ABC хорошо справляется с точечной оценкой, но стандартные реализации завышают неопределённость параметров. Если мы используем регрессионную коррекцию, то ABC может точно количественно оценить эту неопределённость.
- [Глава 12. Approximating the Likelihood in Approximate Bayesian Computation](#). Концептуальная и методологическая основа ABC предназначена, в первую очередь, для решения задач, в которых оценка правдоподобия либо затруднена, либо невозможна. ABC использует непараметрическую оценку правдоподобия сводной статистики на основе моделирования и предполагает, что генерация данных из модели не требует больших вычислительных затрат. В главе рассматриваются два альтернативных подхода к оценке трудно поддающегося анализу правдоподобия с целью сокращения необходимого количества симуляций модели. Первый, байесовская версия синтетического правдоподобия (SL), использует многомерное нормальное приближение к правдоподобию статистики. Второй метод аппроксимации правдоподобия основан на эмпирическом правдоподобию (EL), непараметрическом методе, максимизирующем правдоподобие, построенное эмпирически с учётом набора ограничений на моменты распределения. В отличие от ABC и байесовского SL (BSL), байесовский EL (BCel) в некоторых случаях позволяет полностью избежать моделирования. Методы BSL и BCel проиллюстрированы на моделях разной сложности.
- [Глава 13. A Guide to General-Purpose Approximate Bayesian Computation Software](#). Представлено программное обеспечение общего назначения для выполнения приближённых байесовских вычислений (ABC), реализованное в R-пакетах `abc` и `EasyABC`, а также в программе `ABCtoolbox` на C++. На простых моделях продемонстрировано, как выполнять вывод параметров, выбор модели, проверку и оптимальный выбор сводной статистики. Показано, как сочетать ABC с MCMC, и описано реалистичное приложение из области популяционной генетики.
- [Глава 14. Divide and conquer in ABC: Expectation-Propagation algorithms for likelihood-free inference](#). Как известно, алгоритмы ABC требуют больших вычислительных ресурсов, поскольку для них необходимо моделировать множество полных искусственных наборов данных. Предложен подход к ABC по принципу «разделяй и властвуй», при котором правдоподобие разделяется на n факторов и каким-то образом объединяется n «локальных» аппроксимаций ABC для каждого фактора. У этого подхода есть два преимущества: (а) он обычно выполняется намного быстрее, чем стандартный ABC, и (б) он позволяет использовать локальную сводную статистику

(которая зависит только от точек данных, соответствующих одному фактору), а не глобальную статистику, зависящую от всего набора данных. Это значительно снижает систематическую ошибку, возникающую при использовании сводной статистики. Представлено две вариации метода Expectation-Propagation (EP, эффективного способа объединения n локальных аппроксимаций в глобальную): одна основана на параллельном алгоритме, который можно реализовать на параллельной архитектуре, а другая связывает стандартный EP с параллельным. Подход проиллюстрирован на байесовском выводе пространственных экстремумов.

- [Глава 21. ABC in Nuclear Imaging](#). Рассмотрено применение ABC в контексте медицинской визуализации данных. Рассмотрена оценка параметров компартментных моделей при анализе ПЭТ-изображений и предложен простой алгоритм ABC для такой оценки. Продемонстрирована полезность предложенных методов оценки на примере модели реакции нейромедиаторов и проведено сравнение с другими существующими методами.

3.2 Синтетическое правдоподобие

В отличие от ABC, который неявно оценивает вероятность сводных статистик, байесовская версия синтетического правдоподобия (BSL, Bayesian synthetic likelihood) напрямую предполагает, что совместная плотность распределения сводной статистики, зависящая от неизвестных параметров модели, является гауссовой с неизвестным значением среднего и дисперсии. Используя независимое моделирование, полученное в результате предполагаемого процесса генерации данных (DGP, data generation process), затем оцениваются среднее и дисперсия сводной статистики, необходимые для построения (смоделированной) гауссовой функции правдоподобия, которая непосредственно вводится в стандартные алгоритмы MCMC.

BSL чаще всего применяется в ситуациях, когда сложность модели, которая, как предполагается, сгенерировала наблюдаемые данные, делает невозможным точный байесовский вывод. То есть, по самой природе задач, к которым обычно применяется BSL, модель настолько сложна, что мы не можем легко получить доступ к DGP и вместо этого вынуждены прибегать к приближенному вероятностному выводу. Однако, хотя сложные, высокоструктурированные модели позволяют объяснить важнейшие особенности наблюдаемых данных, маловероятно, что какой-либо разработчик моделей способен построить абсолютно точную модель, которая отражала бы все особенности наблюдаемых данных.

Модели, к которым обычно применяется BSL, являются неточно заданными представлениями фактического, или истинного, DGP. Недавние результаты [Frazier2017] демонстрируют, что если модель неправильно задана, вывод на основе ABC может быть сомнительным. Учитывая, что принципы, лежащие в основе ABC и BSL, качественно схожи, необходим дальнейший анализ, чтобы убедиться, что BSL не страдает теми же недостатками, что и ABC, в тех случаях мисспецификации модели.

В [Fraizer2020] на нескольких модельных и эмпирических примерах продемонстрировано, что, когда предполагаемая модель задана неверно, BSL может выдавать недостоверные результаты. Для решения этой проблемы, авторы предложили две новые версии BSL, которые могут выдавать надежные результаты независимо от того, правильно ли задана модель. Эти робастные версии BSL основаны на корректировке смоделированных сводных статистик для обеспечения совместимости между предполагаемой моделью и выбранной статистикой.

4 Нейросетевые SBI

4.1 Статья [Cranmer2020]

Название: The frontier of SBI

Авторы: Kyle Cranmer, Johann Brehmer, and Gilles Louppe

Аннотация: *Многие области науки для описания интересующих их явлений создали сложные симуляторы. Хотя эти симуляторы предоставляют высокоточные модели, они плохо подходят для статистического вывода и порождают сложные обратные задачи. Мы подробно рассматриваем бурно развивающуюся область статистического вывода на основе моделирования (simulation-based inference, SBI), акцентируя внимание на тех факторах, которые придадут развитию этого направления дополнительный импульс. Наконец, мы описываем, как расширяется фронт исследований, чтобы широкая аудитория могла оценить глубокое влияние, которое эти разработки могут оказать на науку.*

Выразительность языков программирования способствует разработке сложных, высокоточных численных моделей («симуляторов»), а мощность современных вычислений позволяет генерировать на их основе синтетические данные. К сожалению, эти симуляторы плохо приспособлены для статистического вывода (inference). Источником проблемы является то, что плотность вероятности заданного наблюдения (*функция правдоподобия*) – необходимый компонент как частотных, так и байесовских методов вывода – обычно является *невыводимой* (intractable). Такие модели часто называют неявными (implicit), противопоставляя их явным (prescribed) моделям, где правдоподобие наблюдения можно вычислить явным образом.

Невыводимость правдоподобия является препятствием для научного прогресса, поскольку статистический вывод – ключевой компонент научного метода. В областях, где возникло это препятствие, учёные разработали различные ad hoc или узкоспециализированные методы его преодоления. В частности, два распространённых традиционных подхода опираются на то, что учёные используют своё понимание системы для построения информативных *сводных статистик* (summary statistics), а затем сравнивают наблюдаемые данные с модельными. В первом подходе для аппроксимации распределения сводных статистик сгенерированных симулятором данным используются методы *оценки плотности* (DE, density estimation). В частности, этот подход использовался при открытии бозона Хиггса в рамках частотного (frequentist) подхода и схематически проиллюстрирован на рис. 2E. Альтернативный метод, известный как *приближённое байесовское вычисление* (ABC, Approximate Bayesian Calculation), сравнивает наблюдаемые и смоделированные данные на основе некоторой меры расстояния в пространстве сводных статистик. ABC широко используется в популяционной биологии, вычислительной нейронауке и космологии и изображён на рис. 2A.

В последнее время набор инструментов для вывода на основе моделирования (SBI, Simulation-Based Inference) значительно расширился. В целом, три фактора придают этой области новый импульс. Во-первых, произошло значительное взаимопроникновение идей между исследователями, изучающими SBI, и специалистами по *вероятностным моделям в машинном обучении* (ML). Впечатляющий рост возможностей ML открывает пути для новых подходов. Во-вторых, *активное обучение* (active learning) – идея о непрерывном использовании полученных знаний¹ для управления симулятором – признаётся ключевой

¹Здесь и далее авторы часто используют термин knowledge, который мы переводим как «знание». Его отнюдь не тривиальный смысл (частично) становится понятным из контекста, в котором данный термин

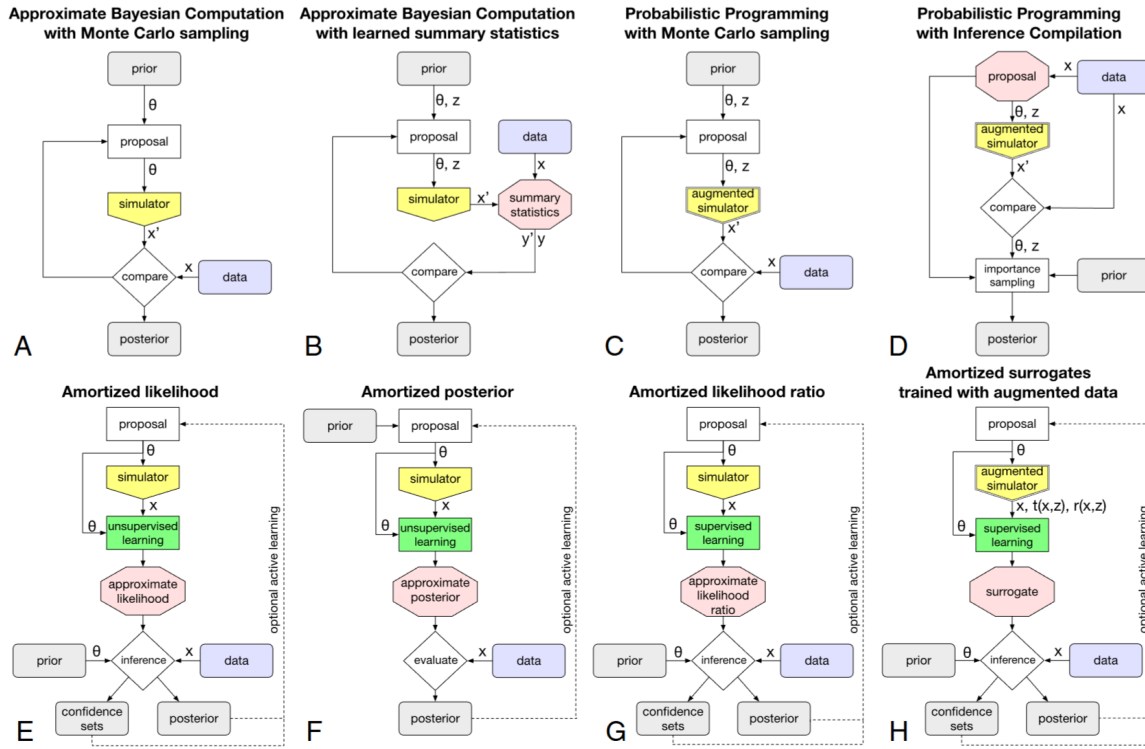


Рис. 2: Обзор различных подходов SBI.

концепцией для повышения эффективности использования данных в различных методах вывода. В-третьих, новое направление исследований перестало рассматривать симулятор как «чёрный ящик» и сосредоточилось на *интеграциях*, позволяющих механизму вывода напрямую использовать внутренние детали симулятора.

4.1.1 Симуляторы и статистический вывод

Статистический вывод выполняется в контексте статистической модели, а при SBI сама симуляция определяет статистическую модель. Симулятор – это компьютерная программа, которая принимает на вход вектор параметров θ , выбирает ряд внутренних состояний или латентных переменных $z_i \sim p_i(z_i|\theta, z_{<i})$ и, наконец, генерирует на выходе вектор данных $x \sim p(x|\theta, z)$. Программы, включающие случайное сэмплирование и интерпретируемые как статистические модели, известны как *вероятностные программы*, и симуляторы являются их примером. В рамках этой общей формулировки реальные симуляторы могут существенно различаться.

- Параметры θ описывают лежащую в основе механистическую модель и, следовательно, влияют на вероятности переходов $p_i(z_i|\theta, z_{<i})$. Обычно механистическая модель интерпретируема специалистом в предметной области, θ имеет относительно мало компонент и фиксированную размерность. Примерами являются коэффициенты в гамильтониане физической системы, вирулентность и инкубационный период патогена или фундаментальные константы природы.
- Латентные (скрытые) переменные z , возникающие в процессе генерации данных, могут прямо или косвенно соответствовать физически осмысленному состоянию систе-

употребляется в дальнейшем.

мы, но обычно это состояние ненаблюдаемо на практике. Структура латентного пространства существенно различается у разных симуляторов. Латентные переменные могут быть непрерывными или дискретными, а размерность скрытого пространства может быть фиксированной или изменяемой в зависимости от потока управления (control flow) симулятора. Симуляция может свободно комбинировать детерминированные и стохастические шаги. Детерминированные компоненты симулятора могут быть дифференцируемыми, а могут включать элементы с разрывным потоком управления. На практике некоторые симуляторы предоставляют удобный доступ к латентным переменным, в то время как другие фактически являются «чёрными ящиками». Любой симулятор может комбинировать эти различные аспекты практически любым способом.

- Наконец, выходные данные x соответствуют наблюдениям. Они могут варьироваться от нескольких неструктурированных чисел до высокоразмерных и сильно структурированных данных, таких как изображения или геопространственная информация.

Например, процессы в физике элементарных частиц часто зависят лишь от небольшого числа параметров, таких как массы частиц или константы связи. Латентный процесс объединяет высокоэнергетическое взаимодействие, строго описываемое квантовой теорией поля, с прохождением образовавшихся частиц через невероятно сложный детектор, наиболее точно моделируемый с помощью стохастических симуляций с миллиардами латентных переменных. Эпидемиологические симуляторы могут основываться на сетевой структуре с геопространственными свойствами, а латентный процесс состоит из многократно повторяющихся структурно идентичных стохастических временных шагов. Напротив, космологические симуляции эволюции Вселенной могут включать высоко структурированное стохастическое начальное состояние, за которым следует гладкая, детерминированная временная эволюция.

Научные задачи *статистического вывода* (inference) различаются в зависимости от того, что именно оценивается: имея наблюдаемые данные x , является ли целью получение входных параметров θ , латентных переменных z , или того и другого? Иногда интерес представляет только подмножество параметров, в то время как остальные являются *мешающими параметрами* (nuisance parameters).

Статистический вывод может выполняться либо в рамках частотного (frequentist), либо байесовского (Bayesian) подхода, может ограничиваться точечными оценками $\hat{\theta}(x)$ или расширяться до включения вероятностного представления неопределённости. В частотном случае доверительные множества (confidence set) часто формируются путём обращения критериев проверки статгипотез (основанных на статистике отношения правдоподобий). При байесовском выводе цель обычно заключается в вычислении *апостериорного распределения* («постериора») $p(\theta|x) = p(x|\theta) p(\theta) / \int d\theta' p(x|\theta') p(\theta')$ для наблюдаемых данных x и заданного *априорного распределения* («прайора») $p(\theta)$. В обоих случаях ключевым компонентом является *функция правдоподобия* $p(x|\theta)$.

Фундаментальная проблема для SBI заключается в том, что функция правдоподобия $p(x|\theta)$, неявно определяемая симулятором, обычно является невычислимой (intractable), поскольку соответствует интегралу по всем возможным траекториям в латентном пространстве (т.е. всем возможным трекам выполнения симулятора): $p(x|\theta) = \int dz p(x, z|\theta)$, где $p(x, z|\theta)$ – совместная плотность вероятности данных x и латентных переменных z . Для реальных симуляторов с большими латентными пространствами явно вычислить этот интеграл невозможно. Поскольку правдоподобие является центральным компонентом как частотного, так и байесовского вывода, это представляет собой серьёзную проблему для

вывода во многих областях.

К классическим вариантам SBI можно отнести ABC (рис. 2А) и подход, основанный на создании модели для правдоподобия путём оценки распределения смоделированных данных с помощью гистограмм или оценки плотности на основе ядра (kernel density estimation, KDE, после чего частотный и байесовский вывод выполняются так, как если бы правдоподобие было явным, рис. 2Е). Одно из преимуществ этого второго подхода (иногда называемого приближёнными частотными вычислениями, AFC) перед ABC заключается в том, что он *амортизирован* (amortized): после первоначальных вычислительных затрат на этапе симуляции и оценки плотности, новые точки данных могут быть оценены эффективно (на рисунке синий блок data входит только на этапе вывода и не влияет на шаг симуляции). Это свойство особенно желательно для случая с независимыми одинаково распределёнными (i.i.d.) наблюдениями.

Оба этих традиционных подхода страдают от «проклятия размерности»: в худшем случае требуемое количество симуляций возрастает экспоненциально с увеличением размерности данных x . Поэтому оба подхода полагаются на низкоразмерные сводные статистики $y(x)$, и качество вывода напрямую связано с тем, насколько хорошо эти статистики сохраняют информацию о параметрах θ . Традиционно разработка эффективных сводных статистик была задачей эксперта в предметной области, и сводные статистики определялись до начала процесса вывода.

Кроме того, оба подхода имеют недостатки в трёх критических аспектах вывода:

- *Эффективность использования данных* (Sample efficiency): Количество смоделированных выборок, необходимое для получения хорошей оценки правдоподобия или апостериорного распределения, может быть запредельно большим.
- *Качество вывода* (Quality of inference): Сокращение данных до низкоразмерных сводных статистик неизбежно приводит к потере части информации в данных о параметрах, что снижает статистическую мощность (statistical power). Большие значения меры близости (допуска) ϵ в ABC или параметра ширины окна (bandwidth) в KDE приводят к плохой аппроксимации истинного правдоподобия – и то, и другое снижает общее качество вывода.
- *Амортизация* (Amortization): Выполнение вывода с помощью ABC для нового набора наблюдаемых данных требует повторения большинства этапов цепочки вывода, особенно если предложенное (proposal) распределение зависит от наблюдаемых данных. Метод плохо масштабируется при применении к большому количеству наблюдений. С другой стороны, вывод, основанный на KDE, является амортизированным: вычислительно затратные этапы не нужно повторять для новых наблюдений.

4.1.2 Машинное обучение, активное обучение и интеграция

В последние годы появились новые возможности, позволяющие улучшить все три этих аспекта. Мы условно группируем их в три основных направления прогресса:

1. Революция в ML позволяет работать с данными более высокой размерности, что может улучшить качество вывода. Методы вывода, основанные на нейросетевых суррогатных моделях, напрямую выигрывают от впечатляющих темпов прогресса в области глубокого обучения.

2. Методы активного обучения (active learning) могут систематически повышать эффективность использования данных, позволяя работать с более вычислительно затратными симуляторами.
3. Глубокая интеграция автоматического дифференцирования и вероятностного программирования в код симуляции, а также обогащение обучающих данных информацией, которую можно извлечь из симулятора, меняют подход к использованию симулятора при выводе: он больше не является «чёрным ящиком», а становится открытым для рабочего процесса вывода.

За последнее десятилетие методы ML, в частности глубокие нейронные сети (NN), превратились в универсальные, мощные и популярные инструменты для решения разнообразных задач. Изначально NN продемонстрировали прорыв в задачах обучения с учителем, таких как классификация и регрессия. NN можно легко комбинировать для решения задач более высокого уровня, что делает их подходящими для проблем с иерархической или композиционной структурой. Были разработаны архитектуры, адаптированные к различным структурам данных, включая: 1) полносвязные сети для неструктурированных данных, 2) свёрточные нейронные сети (CNN), использующие пространственную структуру, например, в данных изображений, 3) рекуррентные нейронные сети (RNN) для последовательностей переменной длины, 4) графовые нейронные сети (GNN) для данных с графовой структурой.

Примечание. Выбор архитектуры, хорошо подходящей для конкретной структуры данных, является примером *индуктивного смещения* (inductive bias), которое в более общем смысле относится к предположениям, заложенным в алгоритм обучения, независимо от данных. Индуктивное смещение – один из ключевых факторов успеха большинства успешных применений глубокого обучения, хотя его роль сложно охарактеризовать точно.

Одна из областей, в которой NN активно развиваются, – оценка плотности распределения в пространствах высокой размерности: имея набор точек $\{x\} \sim p(x)$, цель состоит в том, чтобы оценить вероятностную плотность $p(x)$. Поскольку явных меток нет, это обычно считается задачей обучения без учителя. Нейронные оценки плотности (NDE) были обобщены для моделирования зависимости от дополнительных входных данных, т.е. для моделирования условной плотности, такой как правдоподобие $p(x|\theta)$ или апостериорное распределение $p(\theta|x)$.

Один из классов этих методов NDE – *нормализующие потоки* (normalizing flows), в которых переменные, описываемые простым базовым распределением $p(u)$ (например, многомерным гауссовым), преобразуются с помощью параметризованного обратимого преобразования $x = g_\phi(u)$, имеющего вычислимый якобиан. Тогда целевая плотность $p_g(x)$ задаётся формулой замены переменных как произведение базовой плотности и определителя якобиана преобразования. Несколько таких шагов можно объединять в стек, при этом плотность вероятности «протекает» через последовательные преобразования переменных. Параметры ϕ преобразований обучаются путём максимизации правдоподобия наблюдаемых данных в модели $p_g(x_{\text{obs}})$, что даёт модельную плотность, аппроксимирующую истинную, неизвестную плотность $p(x)$. Помимо возможности вычислять плотность, из модели можно генерировать данные, выбирая скрытые переменные u из базового распределения и применяя преобразования потока.

Генеративные состязательные сети (GANs, generative adversarial network) представляют собой альтернативный тип генеративных моделей, основанных на нейронных сетях. В отличие от нормализующих потоков, преобразование, реализуемое генератором, не обязательно быть обратимым. Хотя это позволяет добиться большей выразительности, плотность

вероятности, определяемая генератором, является невычислимой. Поскольку максимизация правдоподобия не может быть целевой функцией обучения, генератор противопоставляется сопернику (adversary), чья роль – различать сгенерированные данные и целевое распределение. Позже мы обсудим, как эта же идея может быть использована для SBI с помощью метода, известного как likelihood-ratio trick.

Простая, но очень действенная идея заключается в том, чтобы запускать симулятор в тех точках параметров θ , от которых ожидается наибольший прирост знаний. Это можно делать итеративно, так что после каждой симуляции знания, полученные из всех предыдущих запусков, используются для определения следующей точки параметров. Существует несколько технических реализаций этой идеи активного обучения. Она обычно применяется в байесовском контексте, где постериор может постоянно обновляться и использоваться для управления *предложенным* (proposal) распределением параметров симулятора.

Похожие идеи обсуждаются в контексте принятия решений, планирования эксперимента и обучения с подкреплением, и мы ожидаем дальнейшего улучшения алгоритмов вывода за счёт взаимопроникновения идей между этими областями. Например, вопрос, который иногда обсуждается в контексте обучения с подкреплением или байесовской оптимизации, но ещё не применялся в условиях отсутствия правдоподобия, – как использовать симуляторы с *множественной точностью* (multifidelity simulators), предлагающие несколько уровней точности или приближений.

И машинное обучение, и активное обучение могут существенно улучшить качество вывода и эффективность использования данных по сравнению с классическими методами. Однако в целом они не меняют кардинально базовый подход к SBI: они по-прежнему рассматривают симулятор как генеративный «чёрный ящик», который принимает параметры на входе и выдаёт данные на выходе, с чётким разделением между симулятором и механизмом вывода. Третье направление исследований меняет эту парадигму, *открывая* этот «чёрный ящик» и более тесно интегрируя вывод и симуляцию.

Одним из примеров этого сдвига является парадигма *вероятностного программирования* (probabilistic programming). Вероятностные программы описываются как обычные функциональные или императивные программы с двумя дополнительными конструкциями:

- 1) Возможностью случайным образом выбирать значения из распределений.
- 2) Возможностью обуславливать значения переменных в программе через наблюдения.

Фокусируясь на первой конструкции, которая не требует открытия «чёрного ящика», вводится понятие симулятора как вероятностной программы. Однако обуславливание наблюдениями требует более глубокой интеграции, поскольку включает управление случайностью в генеративном процессе. Этот подход абстрагирует возможности, необходимые для реализации *фильтров частиц* (particle filters) и *последовательного Монте-Карло* (SMC). Ранее для этого требовалось писать программу на специализированном языке; однако в последних работах показано, что эти возможности можно добавить к существующим симуляторам с минимальными изменениями в их коде. В конечном итоге, вероятностное программирование стремится предоставить инструменты для вывода в невероятно сложном пространстве всех треков выполнения симулятора, обусловленных наблюдением.

Дополнительная информация, характеризующая латентный процесс генерации данных, может быть извлечена из симулятора и использована для расширения (augmentation) данных, применяемых для обучения суррогатных моделей. Разработчикам алгоритмов вывода и тем, кто знаком с деталями симулятора, следует рассмотреть – помимо одной лишь возможности сэмплирования $x \sim p(x|\theta)$ – являются ли следующие величины хорошо определёнными и вычислимыми:

- (1) $p(x|z, \theta)$
- (2) $t(x, z|\theta) \equiv \nabla_{\theta} \ln p(x, z|\theta)$
- (3) $\nabla_z \ln p(x, z|\theta)$
- (4) $r(x, z|\theta, \theta') \equiv p(x, z|\theta)/p(x, z|\theta')$
- (5) $\nabla_{\theta}(x, z)$
- (6) $\nabla_z x$

Эти величины затем могут быть использованы для *аугментации* обычного выхода x из симулятора и могут применяться в целевых функциях обучения с учителем, что может значительно повысить эффективность использования данных для обучения суррогатных моделей.

Многие из упомянутых выше величин включают производные, которые теперь можно эффективно вычислять с помощью *автоматического дифференцирования* (autodiff) – семейства методов, схожих с, но более общих, чем алгоритм обратного распространения ошибки (backpropagation), повсеместно используемый в глубоком обучении. Как и вероятностное программирование, автоматическое дифференцирование включает нестандартную интерпретацию кода симуляции и разрабатывалось небольшой, но устоявшейся областью компьютерной науки. В последние годы несколько исследователей высказывали мнение, что глубокое обучение было бы лучше описывать как *дифференциальное программирование* (differential programming). С этой точки зрения, внедрение автоматического дифференцирования в существующие коды симуляторов является более прямым способом использовать достижения в области глубокого обучения, чем попытки инкорпорировать предметные знания в совершенно чужеродную среду, такую как глубокая нейронная сеть.

Извлечение необходимой информации из симулятора снова требует глубокой интеграции в код. В то время как технологии для внедрения парадигмы вероятностного программирования в существующие кодовые базы только появляются, разработка инструментов для поддержки автоматического дифференцирования в наиболее часто используемых научных языках программирования уже значительно продвинулась. Мы отмечаем, что две из перечисленных выше величин (2 и 3) включают как автоматическое дифференцирование, так и вероятностное программирование. Интеграция вывода и симуляции, а также идея аугментации обучающих данных дополнительными величинами имеют потенциал изменить то, как мы думаем о SBI. В частности, эта перспектива может повлиять на способ разработки симуляционных кодов, обеспечивающих эти новые возможности.

4.1.3 Рабочие процессы SBI

Этот широкий спектр возможностей можно комбинировать в различных *рабочих процессах* (workflows) вывода. Неотъемлемой частью всех методов вывода является запуск симулятора, отмеченный жёлтым на рис. 2. Параметры, при которых запускается симулятор, выбираются из некоторого предложенного (proposal) распределения. Оно может зависеть или не зависеть от априорного распределения в байесовской постановке и может выбираться либо статически, либо итеративно с помощью метода активного обучения. Затем потенциально высокоразмерный выход симулятора может использоваться непосредственно как вход для метода вывода или сокращаться до низкоразмерных сводных статистик.

Методы вывода можно в целом разделить на две категории: 1) те, которые, подобно ABC, используют сам симулятор во время вывода, и 2) методы, которые строят суррогатную модель и используют её для вывода. В первом случае выход симулятора непосредственно сравнивается с данными (рис. 2A-D), во втором выход симулятора используется как обучающие данные для этапа оценки или машинного обучения (отмечено зелёным на

рис. 2E-H). Полученные суррогатные модели (розовые шестиугольники) затем используются для вывода.

Алгоритмы решают проблему невычислимости истинного правдоподобия разными способами: одни методы строят вычислимую суррогатную модель для функции правдоподобия, другие – для функции отношения правдоподобий (likelihood-ratio function). Оба этих подхода делают частотный (frequentist) вывод непосредственным. В иных методах функция правдоподобия никогда не появляется явно, например, когда она заменяется вероятностью отбраковки (rejection probability).

Конечной целью байесовского вывода является апостериорное распределение. Методы различаются тем, предоставляют ли они: а) доступ к выборкам точек параметров из постериора (например, через МСМС или АВС); б) вычислимую функцию, которая аппроксимирует апостериорную функцию. Аналогично, некоторые методы требуют указания того, какие величины будут оцениваться, на раннем этапе рабочего процесса, в то время как другие позволяют отложить это решение.

Обсудим теперь, как эти блоки и вычислительные возможности могут быть объединены в методы вывода.

Причиной низкой эффективности использования данных в исходном алгоритме rejection АВС является то, что симулятор запускается в точках параметров, выбранных из прайора, который может иметь значительную вероятность в областях, сильно противоречащих наблюдаемым данным. Были предложены различные алгоритмы, которые вместо этого запускают симулятор в точках параметров, от которых ожидается наибольший прирост знаний о постериоре. По сравнению с базовым АВС, эти техники повышают эффективность использования данных, хотя они всё ещё требуют выбора сводных статистик, меры расстояния ρ и допуска ϵ .

В случае, когда финальный этап симулятора вычислим или симулятор является дифференцируемым (соответственно, свойства 1 и 6 из приведенного выше списка), возможен асимптотически точный байесовский вывод, без необходимости полагаться на допуск расстояния или сводные статистики, что устраняет основные ограничения АВС с точки зрения качества вывода.

Парадигма вероятностного программирования представляет собой более фундаментальное изменение в том, как выполняется вывод. Во-первых, она требует, чтобы симулятор был написан на языке вероятностного программирования, хотя последние работы позволяют добавить эти возможности к существующим симуляторам с минимальными изменениями в их коде. Кроме того, вероятностное программирование требует либо вычислимого правдоподобия для финального шага $p(x|z, \theta)$ (величина 1), либо введения АВС-подобного сравнения. Когда эти критерии выполнены, существует несколько алгоритмов вывода, которые могут выбирать сэмплы из апостериорного распределения $p(\theta, z|x)$ входных параметров θ и латентных переменных z при заданных наблюдаемых данных x . Эти техники основаны либо на МСМС, рис. 2С, либо на обучении NN для предоставления proposal, рис. 2D. Ключевое отличие от АВС заключается в том, что механизм вывода контролирует все шаги выполнения программы и может смещать каждую выборку случайных латентных переменных, чтобы симуляция с большей вероятностью соответствовала наблюдаемым данным, повышая эффективность использования данных.

Сильная сторона этих алгоритмов заключается в том, что они позволяют нам вывести не только входные параметры симулятора, но и весь латентный процесс, ведущий к конкретному наблюдению. Это позволяет нам отвечать на совершенно иные вопросы о научных процессах, добавляя особый тип физической интерпретируемости, которым не обладают методы, основанные на суррогатных моделях. В то время как стандартные алго-

ритмы ABC в принципе позволяют выводить z , вероятностное программирование решает эту задачу более эффективно.

4.1.4 Суррогатные модели

Ключевым недостатком непосредственного использования симулятора во время вывода является отсутствие амортизации. Когда появляются новые наблюдаемые данные, всю цепочку вывода приходится повторять. Путем обучения вычислимой суррогатной модели или эмулятора для симулятора, вывод становится амортизированным: после начальной фазы симуляции и обучения новые данные могут оцениваться очень эффективно. Этот подход особенно хорошо масштабируется на данные, состоящие из множества независимых одинаково распределённых (i.i.d.) наблюдений. Это идея не нова, и устоявшиеся методы используют классические техники оценки плотности для создания суррогатной модели функции правдоподобия. Но новые вычислительные возможности придали новый импульс этому классу методов вывода, некоторые из которых визуализированы на рис. 2E–H.

Мощный вероятностный подход заключается в обучении нейронных условных оценок плотности, таких как нормализующие потоки, в качестве суррогата для симулятора. Условная плотность может быть определена в двух направлениях: сеть может изучать либо апостериорное распределение $p(\theta|x)$, либо правдоподобие $p(x|\theta)$. Мы показываем эти две техники на рис. 2E и F; обратите внимание, что алгоритм суррогата правдоподобия структурно идентичен классическому подходу на основе оценки плотности, но использует более мощные методы оценки плотности.

Схожим образом, NN можно обучить аппроксимации функции отношения правдоподобий $p(x|\theta_0)/p(x|\theta_1)$ или $p(x|\theta_0)/p(x)$, где в последнем случае знаменатель задаётся маргинальной моделью, проинтегрированной по предложенному распределению или априорному распределению, рис. 2G. Ключевая идея тесно связана с сетью-дискриминатором в GAN, упомянутой выше: классификатор обучается с учителем для различения двух наборов данных, хотя в данном случае оба набора поступают из симулятора и генерируются для разных точек параметров θ_0 и θ_1 . Выходная функция классификатора может быть преобразована в аппроксимацию отношения правдоподобий между θ_0 и θ_1 ! Это проявление леммы Неймана-Пирсона в контексте ML часто называют likelihood-ratio trick.

Все три подхода, основанные на суррогатных моделях, являются амортизированными: после начальной фазы симуляции и обучения суррогатные модели могут эффективно оцениваться для произвольных данных и точек параметров. Они требуют предварительного определения интересующих параметров, после чего сеть неявно маргинализует по всем остальным (латентным) переменным в симуляторе. Все три класса алгоритмов могут использовать элементы активного обучения, такие как итеративно обновляемое proposal, чтобы направлять параметры симулятора θ в релевантную область параметров, повышая эффективность использования данных. Использование NN устраняет требование к низкоразмерным сводным статистикам, предоставляя используемой модели самой изучить структуры в высокоразмерных данных, что потенциально улучшает качество вывода.

Несмотря на эти фундаментальные сходства, существуют и некоторые различия между эмуляцией правдоподобия, отношения правдоподобий и постериора. Изучение апостериорного распределения напрямую даёт основную целевую величину в байесовском выводе, но вводит зависимость от прайора на каждом этапе метода вывода. Изучение правдоподобия или отношения правдоподобий позволяет проводить частотный вывод или сравнение моделей, хотя для байесовского вывода необходим дополнительный шаг MCMC или использование величины b для генерации выборок из апостериорного распределения. Независимость от прайора оценок правдоподобия или отношения правдоподобий также даёт

дополнительную гибкость для изменения априорного распределения в процессе вывода. Преимуществом обучения генеративной модели для аппроксимации правдоподобия или апостериорного распределения над изучением функции отношения правдоподобий является дополнительная функциональность возможности сэмплирования из суррогатной модели. С другой стороны, изучение правдоподобия или апостериорного распределения – это задача обучения без учителя, тогда как оценка отношения правдоподобий через классификатор является примером обучения с учителем и зачастую более простой задачей. Поскольку для цели вывода более высокого уровня правдоподобие и отношение правдоподобий могут использоваться взаимозаменяемо, изучение суррогата для функции отношения правдоподобий часто может быть более эффективным.

Ещё одна стратегия, позволяющая использовать обучение с учителем, основана на извлечении дополнительных величин из симулятора, которые характеризуют правдоподобие латентного процесса (например, величины 2 и 4). Эта дополнительная информация может быть использована для аугментации обучающих данных для суррогатных моделей. Получаемая задача обучения с учителем часто может решаться более эффективно, что в конечном итоге повышает эффективность использования данных в задаче вывода.

Подходы, основанные на суррогатных моделях, выигрывают от наложения подходящего индуктивного смещения (inductive bias) для конкретной задачи. Широко признано, что сетевая архитектура нейросетевого суррогата должна выбираться в соответствии со структурой данных (например, изображения, последовательности или графы). Другой, потенциально более значимый способ наложения индуктивного смещения заключается в том, чтобы суррогатная модель отражала причинно-следственную структуру симулятора. Ручное определение соответствующих структур и проектирование подходящих архитектур суррогатов является очень специфичным для предметной области, хотя было показано, что это улучшает производительность в некоторых задачах. В последнее время предпринимаются попытки автоматизировать процесс создания суррогатов, которые имитируют симуляцию. Заглядывая дальше вперёд, мы хотели бы научиться создавать суррогаты, которые отражают причинно-следственную структуру «огрубленной» (coarse-grained) системы. Если это возможно, это позволило бы суррогатной модели учитывать только соответствующие степени свободы для явлений, возникающих из лежащей в основе механистической модели.

4.1.5 Пре- и постобработка

Существует ряд дополнительных этапов, которые могут обрамлять эти основные алгоритмы вывода, будь то в форме этапов предварительной обработки, предшествующих основному этапу вывода, или в качестве последующего этапа обработки, следующего за основным шагом вывода.

Одним из этапов предобработки является изучение информативных сводных статистик $y(x)$. Из-за проклятия размерности как ABC, так и методы вывода на основе классической оценки плотности требуют сжатия данных в низкоразмерные сводные статистики. Они обычно задаются заранее (т.е. выбираются вручную учёными-специалистами на основе их интуиции и знаний о задаче), но полученные сводные статистики, как правило, теряют часть информации по сравнению с исходными данными. Минимально инвазивным расширением этих алгоритмов является сначала изучение сводных статистик, обладающих определёнными свойствами оптимальности, перед запуском стандартного алгоритма вывода, такого как ABC. Схематично этот подход изображен на рис. 2В для ABC, но он в равной степени применим и к выводу на основе оценки плотности.

Вектор оптимальных сводных статистик можно определить с помощью скор-функции (score) $t(x|\theta) \equiv \nabla_{\theta} p(x|\theta)$ – градиента логарифма (маргинального) правдоподобия по интересующим параметрам: в окрестности точки θ компоненты сора являются достаточными статистиками, и их можно использовать для вывода без потери информации. Как и само правдоподобие, скор обычно невычислим, но его можно оценить на основе величины \hat{z} и аппроксимации экспоненциальным семейством распределений. Если доступна величина \hat{z} , то для обучения NN оценке сора можно вместо этого использовать аугментированные данные, извлечённые из симулятора, не требуя такой аппроксимации. Изучаемые сводные статистики также могут быть сделаны устойчивыми (robust) по отношению к nuisance параметрам.

Даже если нет необходимости сокращать данные до низкоразмерных сводных статистик, в некоторых областях измеренные «низкоуровневые» (raw/low-level) данные могут быть очень высокоразмерными. В таких случаях обычной практикой является их сжатие до более управляемого набора «высокоуровневых» (high-level) признаков умеренной размерности и использование этих сжатых данных в качестве входа для рабочего процесса вывода.

Компиляция вывода (inference compilation) – этап предобработки для алгоритмов вероятностного программирования, показанный на рис. 2D. Первоначальные запуски симулятора используются для обучения нейронной сети, которая применяется для последовательного сэмплинга по важности (sequential importance sampling) как θ , так и z .

После завершения основного рабочего процесса вывода важным вопросом является надёжность результатов: можно ли доверять результату при наличии несовершенств, таких как ограниченный объём выборки, недостаточная ёмкость сети или неэффективная оптимизация?

Одно из решений – калибровка результатов вывода. Используя возможность симулятора генерировать данные для любой точки параметров, мы можем применить подход параметрического бутстрэппинга (bootstrap) для вычисления распределения любой величины, участвующей в рабочем процессе вывода. Эти распределения можно использовать для калибровки процедуры вывода, чтобы обеспечить получение доверительных множеств и апостериорных распределений с корректным покрытием и достоверностью (coverage and credibility). Хотя это возможно в принципе, такие процедуры могут требовать большого количества симуляций. Другие диагностические инструменты, которые можно применить по завершении этапа вывода, включают: обучение классификаторов для различения данных из суррогатной модели и истинного симулятора; проверки на самосогласованность (self-consistency checks); ансамблирование (ensemble methods); сравнение распределений выхода сети с известными асимптотическими свойствами. Некоторые из этих методов могут использоваться для оценки неопределённости, хотя статистическая интерпретация таких доверительных интервалов не всегда очевидна.

Ни один из этих диагностических методов не решает проблем, возникающих при *неточной спецификации модели* (model misspecification), когда симулятор не является точным описанием изучаемой системы. Неточная спецификация модели – это проблема, в равной степени затрагивающая как вывод с явными, так и с неявными моделями. Обычно это решается путём расширения модели для большей гибкости и введения дополнительных мешающих параметров (nuisance parameters).

4.2 Статья [Lueckmann2021]

Название: Benchmarking Simulation-Based Inference

Авторы: Jan-Matthis Lueckmann, Jan Boelts, David S. Greenberg, Pedro J. Gonçalves, Jakob H. Maske

Аннотация: *Последние достижения в области вероятностного моделирования привели к появлению большого количества алгоритмов вероятностного вывода, основанных на моделировании (SBI), которые не требуют численной оценки правдоподобия. Однако для таких «likelihood-free» алгоритмов, не было разработано общедоступного эталона (benchmark) с подходящими показателями эффективности (performance metrics). Из-за этого было сложно сравнивать алгоритмы и выявлять их сильные и слабые стороны. Мы решили восполнить этот пробел и подготовили benchmark с задачами вывода и подходящими метриками, а также с первоначальным набором алгоритмов, включая новейшие подходы с использованием нейронных сетей и классических методов приближённых байесовских вычислений (ABC). Мы обнаружили, что выбор метрики имеет решающее значение, что даже самые современные алгоритмы можно значительно улучшить и что последовательная оценка повышает эффективность выборки. Подходы, основанные на нейросетях, как правило, показывают более высокую производительность, но универсального лучшего алгоритма не существует. Мы даём практические рекомендации и подчёркиваем потенциал benchmark для диагностики проблем и улучшения алгоритмов.*

The benchmark consists of a set of algorithms, performance metrics and tasks. Given a prior $p(\theta)$ over parameters θ , a simulator to sample $x \sim p(x|\theta)$ and an observation x_{obs} , an algorithm returns an approximate posterior $q(\theta|x_{\text{obs}})$, or samples from it, $\theta \sim q$. The approximate solution is tested, according to a performance metric, against a reference posterior $p(\theta|x_{\text{obs}})$.

Весь код программы доступен по ссылке: <https://github.com/sbi-benchmark/sbim>.

4.2.1 Алгоритмы

REJ-ABC and SMC-ABC (см. 1 и 2)

Approximate Bayesian Computation (ABC) is centered around the idea of MC rejection sampling. Parameters θ are sampled from a proposal distribution, simulation outcomes x are compared with observed data x_{obs} , and are accepted or rejected depending on a (user-specified) distance function and rejection criterion. While rejection ABC (REJ-ABC) uses the prior as a proposal distribution, the efficiency can be improved by using sequentially refined proposal distributions (SMC-ABC). We implemented REJ-ABC with quantile-based rejection and used the scheme of [Beaumont2009] for SMC-ABC. We extensively varied hyperparameters and compared the implementation of an ABC-toolbox against our own. We investigated linear regression adjustment and the summary statistics approach.

NLE and SNLE (см. 3 и 4)

Likelihood estimation (or ‘synthetic likelihood’) algorithms learn an approximation to the intractable likelihood. While early incarnations focused on Gaussian approximations, recent versions utilize deep neural networks to approximate a density over x , followed by MCMC to obtain posterior samples. Since we primarily focused on these latter versions, we refer to them as neural likelihood estimation (NLE) algorithms, and denote the sequential variant with proposals as SNLE. In particular, we used the scheme with masked autoregressive flows (MAFs) for density estimation.

NPE and SNPE (см. 5 и 6)

Instead of approximating the likelihood, these approaches directly target the posterior. Their origins date back to regression adjustment approaches. Modern variants use neural networks

Algorithm 1: Rejection ABC

```
while in simulation budget do
  Sample  $\theta'$  from  $p(\theta)$ 
  Simulate data  $\mathbf{x}'$  from  $p(\mathbf{x}|\theta')$ 
  if  $d(\mathbf{x}', \mathbf{x}_{\text{obs}}) \leq \epsilon$  then
    | Accept  $\theta'$ 
  else
    | Reject  $\theta'$ 
  end
end
return Accepted samples  $\{\theta'\}$  from  $\hat{p}(\theta|d(\mathbf{x}, \mathbf{x}_{\text{obs}}) \leq \epsilon)$ 
```

Algorithm 2: Population Monte Carlo ABC (ABC-PMC) as in [Beaumont2009]

Set schedule ϵ (including initial ϵ_0), population indicator $t = 0$, and population size N
Initialize weights $W_0 = 1/N$ uniformly
Sample initial population $\{\theta_0^{(i)}\}$ using rejection sampling with ϵ_0

```
while in simulation budget do
  Increase population indicator  $t = t + 1$ 
  Set particle indicator  $i = 0$ 
  while  $i < N$  do
    Sample  $\theta'$  from previous population  $\{\theta_{t-1}^{(i)}\}$  with weights  $\{W_{t-1}^{(i)}\}$ ;
    Perturb  $\theta'$ :  $\theta'' \sim K_t(\theta|\theta')$ 
    Simulate data  $x''$  from  $p(\mathbf{x}|\theta'')$ 
    if  $d(\mathbf{x}'', \mathbf{x}_{\text{obs}}) \leq \epsilon_t$  then
      | Set  $\theta_t^{(i)} = \theta''$  and  $W_t^i = \frac{p(\theta_t^{(i)})}{\sum_{j=1}^N W_{t-1}^j K_t(\theta_t^{(i)}|\theta_{t-1}^j)}$ 
      | Increase particle indicator  $i = i + 1$ 
    else
      | reject  $\theta''$ 
    end
  end
  Normalize weights so that  $\sum_i W_t^{(i)} = 1$ 
end
return Weighted samples  $\{\theta_t^{(i)}\}$  from  $\hat{p}(\theta|d(\mathbf{x}, \mathbf{x}_{\text{obs}}) \leq \epsilon)$ 
```

Algorithm 3: Single round NLE as in [Papamakarios2019a]

```
Set  $\mathcal{D} = \{\}$ 
for  $n = 1 : N$  do
  | Sample  $\theta_n \sim p(\theta)$ 
  | Simulate  $\mathbf{x}_n \sim p(\mathbf{x}|\theta_n)$ 
  | Add  $(\theta_n, \mathbf{x}_n)$  to  $\mathcal{D}$ 
end
Train  $q_\psi(\mathbf{x}|\theta)$  on  $\mathcal{D}$ 
return Samples from  $\hat{p}(\theta|\mathbf{x}_{\text{obs}}) \propto q_\psi(\mathbf{x}_{\text{obs}}|\theta) p(\theta)$  via MCMC;  $q_\psi(\mathbf{x}|\theta)$ 
```

Algorithm 4: Sequential NLE as in [Papamakarios2019a]

```
Set  $\hat{p}_0(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}}) = p(\boldsymbol{\theta})$  and  $\mathcal{D} = \{\}$ 
for  $r = 1 : R$  do
  for  $n = 1 : N$  do
    Sample  $\boldsymbol{\theta}_n \sim \hat{p}_{r-1}(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}})$  with MCMC
    Simulate  $\mathbf{x}_n \sim p(\mathbf{x}|\boldsymbol{\theta}_n)$ 
    Add  $(\boldsymbol{\theta}_n, \mathbf{x}_n)$  to  $\mathcal{D}$ 
  end
  (Re-)train  $q_\psi(\mathbf{x}|\boldsymbol{\theta})$  on  $\mathcal{D}$ 
  Set  $\hat{p}_r(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}}) \propto q_\psi(\mathbf{x}_{\text{obs}}|\boldsymbol{\theta})p(\boldsymbol{\theta})$ 
end
return Samples from  $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}}) \propto q_\psi(\mathbf{x}_{\text{obs}}|\boldsymbol{\theta})p(\boldsymbol{\theta})$  via MCMC;  $q_\psi(\mathbf{x}|\boldsymbol{\theta})$ 
```

for density estimation (approximating a density over θ). Here, we used the recent algorithmic approach proposed by [Greenberg2019] for sequential acquisitions. We report performance using NSFes for density estimation, which outperformed MAFs.

Algorithm 5: Single round NPE as in [Papamakarios2016]

```
for  $j = 1 : N$  do
  Sample  $\boldsymbol{\theta}_j \sim p(\boldsymbol{\theta})$ 
  Simulate  $\mathbf{x}_j \sim p(\mathbf{x}|\boldsymbol{\theta}_j)$ 
end
 $\phi \leftarrow \arg \min \sum_j^N -\log q_{F(\mathbf{x}_j, \phi)}(\boldsymbol{\theta}_j)$ 
Set  $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}}) = q_{F(\mathbf{x}_{\text{obs}}, \phi)}(\boldsymbol{\theta})$ 
return Samples from  $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}}); q_{F(\mathbf{x}, \phi)}(\boldsymbol{\theta})$ 
```

NRE and SNRE (см. 7 и 8)

Ratio Estimation approaches to SBI use classifiers to approximate density ratios. Here, we used the recent approach proposed by [Hermans2020]: A neural network-based classifier approximates probability ratios and MCMC is used to obtain samples from the posterior. SNRE denotes the sequential variant of neural ratio estimation (NRE).

In addition, we benchmarked *Random Forest ABC* (RF-ABC), a recent ABC variant, and *Synthetic Likelihood* (SL) (см. 9 и 10). However, RF-ABC only targets individual parameters (i.e. assumes posteriors to factorize), and SL requires new simulations for every MCMC step, thus requiring orders of magnitude more simulations than other algorithms.

4.2.2 Метрики

Выбор подходящей метрики работоспособности алгоритма является ключевым для любого его тестирования. Поскольку цель алгоритмов SBI – выполнение полного вероятностного вывода, «золотым стандартом» было бы измерение сходства между истинным постериором и полученным в результате вывода с помощью подходящей меры расстояния (или дивергенции) между распределениями. Для этого потребовались бы как доступ к истинному постериору, так и надёжный способ оценки сходства между (потенциально) сложно структурированными распределениями. В предыдущих исследованиях использовались различные метрики в зависимости от ограничений, связанных со знанием об истинном

Algorithm 6: Sequential NPE with atomic proposals [Greenberg2019]

Set $\tilde{p}_1(\boldsymbol{\theta}) = p(\boldsymbol{\theta})$
 $c \leftarrow 0$
for $r = 1 : R$ **do**
 for $j = 1 : N$ **do**
 $c \leftarrow c + 1$
 Sample $\boldsymbol{\theta}_c \sim \tilde{p}_r(\boldsymbol{\theta})$
 Simulate $\mathbf{x}_c \sim p(\mathbf{x}|\boldsymbol{\theta}_c)$
 end
 $V_r(\Theta) := \begin{cases} \binom{c}{M}^{-1} & \text{if } \Theta = \{\boldsymbol{\theta}_{b_1}, \boldsymbol{\theta}_{b_2}, \dots, \boldsymbol{\theta}_{b_M}\} \text{ and } 1 \leq b_1 < b_2 < \dots < b_M \leq c \\ 0 & \text{otherwise} \end{cases}$
 $\phi \leftarrow \arg \min_{\phi} \mathbb{E}_{\Theta \sim V_r(\Theta)} \left[\sum_{\boldsymbol{\theta}_j \in \Theta} -\log \tilde{q}_{\mathbf{x}_j, \phi}(\boldsymbol{\theta}_j) \right]$
 Set $\tilde{p}_{r+1}(\boldsymbol{\theta}) := q_{F(\mathbf{x}_{\text{obs}}, \phi)}(\boldsymbol{\theta})$
end
return Samples from $\hat{p}_R(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}})$; $q_{F(x, \phi)}(\boldsymbol{\theta})$

Algorithm 7: Single round NRE as in [Hermans2019]

Set optimization criterion l (e.g., BCE)
for $j = 1 : N$ **do**
 Sample $\boldsymbol{\theta}_j \sim p(\boldsymbol{\theta})$
 Sample $\boldsymbol{\theta}'_j \sim p(\boldsymbol{\theta})$
 Simulate $\mathbf{x}_j \sim p(\mathbf{x}|\boldsymbol{\theta}_j)$
end
 $\phi \leftarrow \arg \min l(d_{\phi}(\mathbf{x}_n, \boldsymbol{\theta}_n), 1) + l(d_{\phi}(\mathbf{x}_n, \boldsymbol{\theta}'_n), 0)$
Parameterize $d_{\phi}(\mathbf{x}, \boldsymbol{\theta})$
return Samples from $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}})$ via MCMC; $d_{\phi}(\mathbf{x}, \boldsymbol{\theta})$

Algorithm 8: Sequential NRE as in [Hermans2019]

Set optimization criterion l (e.g., BCE)
Set $\tilde{p}(\boldsymbol{\theta}) = p(\boldsymbol{\theta})$
for $r = 1 : R$ **do**
 for $j = 1 : N$ **do**
 Sample $\boldsymbol{\theta}_j \sim \tilde{p}(\boldsymbol{\theta})$ (via d_{ϕ} and MCMC)
 Sample $\boldsymbol{\theta}'_j \sim \tilde{p}(\boldsymbol{\theta})$ (via d_{ϕ} and MCMC)
 Simulate $\mathbf{x}_j \sim p(\mathbf{x}|\boldsymbol{\theta}_j)$
 end
 $\phi \leftarrow \arg \min l(I^n, \boldsymbol{\theta}_n), 1) + l(d_{\phi}(\mathbf{x}_n, \boldsymbol{\theta}'_n), 0)$
 Parameterize $d_{\phi}(\mathbf{x}, \boldsymbol{\theta})$
end
return Samples from $\hat{p}(\boldsymbol{\theta}|\mathbf{x}_{\text{obs}})$ via MCMC; $d_{\phi}(\mathbf{x}, \boldsymbol{\theta})$

Algorithm 9: Random Forest ABC (RF-ABC) as in [Raynal2018]

Set $\mathcal{D} = \{\}$ Set simulation budget N

Set number of trees B

Set minimum node size N_{\min}

for $n = 1 : N$ **do**

 Sample $\boldsymbol{\theta}_n \sim p(\boldsymbol{\theta})$

 Simulate $\mathbf{x}_n \sim p(\mathbf{x}|\boldsymbol{\theta}_n)$

 Add $(\boldsymbol{\theta}_n, \mathbf{x}_n)$ to \mathcal{D}

end

Run random forest regression of \mathbf{x} on $\boldsymbol{\theta}$ using \mathcal{D} , B and N_{\min}

return N samples $\{\boldsymbol{\theta}^{(i)}\}$ and associated weights $\{w^{(i)}\}$ for drawing approximate posterior samples

Algorithm 10: Synthetic Likelihood algorithm as in [Wood2010]

Set number of simulations per step M

Set number of MCMC steps T

for $t = 1 : T$ **do**

 Get new candidate $\boldsymbol{\theta}_t$ from MCMC scheme

 Set $\mathcal{D}_t = \{\}$

for $m = 1 : M$ **do**

 Simulate $\mathbf{x}_m \sim p(\mathbf{x}|\boldsymbol{\theta}_t)$

 Add $(\boldsymbol{\theta}_t, \mathbf{x}_m)$ to \mathcal{D}_t

end

 Use \mathcal{D}_t to estimate mean and covariance of a Gaussian approximation of the likelihood $\hat{L}(\mathbf{x}_{\text{obs}}|\boldsymbol{\theta}_t)$

 Perform the next MCMC step using $\hat{L}(\mathbf{x}_{\text{obs}}|\boldsymbol{\theta}_t)$

end

return N samples $\{\boldsymbol{\theta}^{(i)}\}$ from MCMC chain

распределении и алгоритме вывода. В реальных приложениях обычно известны только наблюдения x_{obs} . Однако в рамках тестирования разумно предположить, что у нас есть доступ, по крайней мере, к истинным параметрам θ_0 (ground truth). Существуют две широко используемые метрики, которые требуют только θ_0 и x_{obs} , правда, они имеют серьёзные недостатки для наших целей.

Probability θ_0 . В литературе широко используется отрицательный логарифм вероятности истинных параметров, $-\ln q(\theta_0|x_{\text{obs}})$, усредненный по различным парам $(\theta_0, x_{\text{obs}})$. Для применения этого варианта не требуется доступ к истинному постериору. Однако использование его лишь для небольшого набора пар $(\theta_0, x_{\text{obs}})$ крайне проблематично: эта метрика валидна только при усреднении по большому множеству наблюдений, полученных из априорного распределения. Для получения надёжных результатов потребовалось бы провести вывод для сотен x_{obs} , что осуществимо только в случае, если вывод выполняется быстро (амортизирован) и плотность q может вычисляться напрямую (среди алгоритмов, используемых здесь, это применимо лишь к NPE).

Posterior-Predictive Checks (PPCs). Как следует из названия, PPC скорее проверка, а не метрика, хотя в литературе по SBI и сообщалось о медианном расстоянии (MEDDIST) между прогнозируемыми выборками и x_{obs} как о подходящем варианте метрики. Ее недостатком является то, что алгоритм, получающий хорошую MAP-оценку, может успешно пройти эту проверку, даже если оценённый постериор окажется плохим. Эмпирически мы обнаружили, что MEDDIST не согласуется с другими метриками.

Maximum Mean Discrepancy (MMD). MMD – это критерий, основанный на проверке двух выборок с применением ядра. В недавних работах сообщалось об использовании MMD с трансляционно-инвариантными гауссовскими ядрами, масштабы длины (length scales) которых определялись с помощью медианной эвристики. Эмпирически мы обнаружили, что MMD может быть чувствителен к выбору гиперпараметров, особенно для постериоров с несколькими модами и разными масштабами длины.

Classifier 2-Sample Tests (C2ST). C2ST использует обученный классификатор для различения выборок из истинного и полученного апостериорных распределений. Это делает его простым в применении и интерпретации.

Kernelized Stein Discrepancy (KSD). KSD – одновыборочный критерий, который требует доступа к $\nabla_{\theta}\tilde{p}(\theta|x_{\text{obs}})$ (\tilde{p} – ненормированное апостериорное распределение), а не к выборкам из p . Подобно MMD, современные оценки KSD используют трансляционно-инвариантные ядра.

f-Divergences. Дивергенции, такие как дивергенция полной вариации (Total Variation, TV) и дивергенция Кульбака-Лейблера (KL), могут быть вычислены только в том случае, когда плотности истинного и приближённого апостериорных распределений доступны для вычисления. Поэтому мы не используем f-дивергенции для сравнительного анализа.

4.2.3 Задачи

Чтобы обеспечить возможность вычисления двухвыборочных критериев, мы сосредоточились на задачах, для которых можно получить референтные постериорные выборки. Рассмотрены 8 чисто статистических задач и 2 прикладные проблемы с разнообразной размерностью параметров и данных. Детально процедура генерации данных для каждой из задач представлена в Appendix T оригинальной статьи.

Gaussian Linear/Gaussian Linear Uniform. Мы включили две версии простых линейных 10-мерных гауссовых моделей, в которых параметр θ представляет собой среднее значение, а матрица ковариации фиксирована. В первой версии используется гауссовый (сопряжённый) прайор, во второй – равномерный. Эти задачи позволяют проверить, как алгоритмы

справляются с тривиальным масштабированием размерности, а также с усечением носителя распределения.

SLCP/SLCP Distractors. Сложная задача вывода, разработанная так, чтобы иметь простую функцию правдоподобия, но сложный постериор: прайор является равномерным для пяти параметров θ , а данные представляют собой набор из четырёх двумерных точек, выбранных из гауссовского правдоподобия, среднее значение и дисперсия которого являются нелинейными функциями θ . Это порождает сложное апостериорное распределение с четырьмя симметричными модами и вертикальными обрезаниями. Мы также включили вторую версию с 92 дополнительными, неинформативными выходами (отвлекающими факторами – distractors), чтобы проверить способность алгоритмов выявлять информативные признаки.

Bernoulli GLM/Bernoulli GLM Raw. Обобщённая линейная модель (GLM) с 10 параметрами и наблюдениями, распределёнными по Бернулли. Вывод выполнялся либо по достаточным статистикам (размерность 10), либо по сырым данным (размерность 100).

Gaussian Mixture. Эта задача вывода стала широко распространённой в литературе по ABC. Она представляет собой смесь двух двумерных гауссовых распределений, одно из которых имеет значительно более широкую ковариацию, чем другое.

Two Moons. Двумерная задача «Две луны» с постериором, демонстрирующим как глобальную (бимодальность), так и локальную (форма полумесяца) структуру, призванная показать, как алгоритмы справляются с многомодальностью.

SIR. Динамические системы представляют собой типичные случаи использования SBI. Модель SIR – это влиятельная эпидемиологическая модель, описывающая динамику числа индивидуумов в трёх возможных состояниях: восприимчивые (S, susceptible), инфицированные (I, infectious) и выздоровевшие или умершие (R, recovered). Мы оцениваем коэффициент контактов β и среднюю скорость выздоровления γ , имея наблюдения за количеством инфицированных I в 10 равноотстоящих моментов времени.

Lotka-Volterra. Важная модель в экологии, описывающая динамику взаимодействия двух видов, широко используемая в исследованиях SBI. Мы оцениваем четыре параметра θ , связанных с взаимодействием видов, по данным о численности особей в обеих популяциях в 10 равноотстоящих моментов времени.

4.2.4 Результаты

По итогам применения перечисленных выше алгоритмов SBI к задачам и сравнения полученных результатов авторы исследования формулируют следующие выводы (кратко, подробнее см. оригинал статьи):

- 1) Выбор метрики является ключевым.
- 2) Это нерешённые задачи.
- 3) Последовательное оценивание повышает эффективность использования выборки.
- 4) Алгоритмы, основанные на оценке плотности или отношения правдоподобия, как правило, превосходят классические методы.
- 5) Не существует единого универсально лучшего алгоритма.
- 6) Бенчмарк можно использовать для диагностики проблем реализации и совершенствования алгоритмов.

4.3 Статья [\[Ho2024\]](#)

Название: LtU-ILI: An All-in-One Framework for Implicit Inference in Astrophysics and Cosmology
Авторы: Matthew Ho, Deaglan J. Bartlett, Nicolas Chartier, Carolina Cuesta-Lazaro, Simon

Ding, Axel Lapel, Pablo Lemos, Christopher C. Lovell, T. Lucas Makinen, Chirag Modi, Viraj Pandya, Shivam Pandey, Lucia A. Perez, Benjamin Wandelt, Greg L. Bryan

Аннотация: В статье представлен конвейер (pipeline) Learning the Universe Implicit Likelihood Inference (LtU-ILI) – код для быстрого, удобного и современного ML в астрофизике и космологии. Конвейер включает в себя программное обеспечение для реализации различных нейросетевых архитектур, схем обучения, прайоров и методов оценки плотности, которые легко адаптируются к любому исследовательскому процессу. Он включает в себя комплексные метрики для оценки апостериорного вывода, что повышает надёжность полученных результатов. Кроме того, конвейер легко распараллеливается и может быть предназначен для эффективного исследования гиперпараметров моделирования. Чтобы продемонстрировать его возможности, представлены реальные примеры из различных областей астрофизики и космологии: оценка массы скоплений галактик на основе рентгеновской фотометрии; определение космологических параметров на основе спектров мощности материи и облаков точек гало; определение характеристик прародителей на основе сигналов гравитационных волн; определение физических параметров пыли на основе цветов и светимости галактик; определение свойств полуаналитических моделей формирования галактик. Также проведены детальный анализ и сравнение всех реализованных методов и обсуждение проблемы и подводных камней, связанных с применением ML в астрономических науках.

For nearly a century, the practice of building linear or perturbative physical models from first-principles allowed us to make substantial progress towards understanding the universe. Exploring the complex, nonlinear regime of physical phenomena through data-driven inference promises significant gains in constraining power. The large data volume of next-generation surveys, the improvements in high-resolution simulations, and the rapid rise of ML techniques have driven an explosion of inquiry into how one can automatically and rapidly learn complex physical phenomena.

Implicit Likelihood Inference (ILI), also known as simulation-based inference (SBI) and likelihood-free inference (LFI), is an approach for learning the statistical relationship between parameters and data. ILI is a mathematically rigorous way of framing ML under the umbrella of Bayesian statistics.

Given a parameterized model and observed data, ILI estimates posterior distributions over model parameters, while accounting for predictive uncertainties. It is best explained in contrast to traditional explicit likelihood analyses, wherein one might write down an analytic likelihood to represent the probability of observations given a model and its parameters, and then use sampling methods such as MCMC to sample from the posterior. In ILI, one seeks to automatically learn the likelihood, i.e. to model the relationship between model parameters and observations through training on simulations. As such, ILI can be applied to infer parameters for any phenomenon that can be simulated, without the need for assumptions about the analytical form of the likelihood. Furthermore, ILI can accommodate complex data cuts (provided that consistent filters are applied to both the data and the forward model) something that can be hard to model in likelihood-based approaches.

LtU-ILI (Learning the Universe Implicit Likelihood Inference) – a pipeline for training ML models for regressive parameter estimation. Given a labeled training set of observed data and parameters, LtU-ILI trains neural networks to emulate posterior probability distributions. The code also provides scientists with a diverse toolkit for testing and validation. It was built under the following design principles:

- 1) The code includes state-of-the-art neural architectures, validation tools, and samplers for enabling ILI.

- 2) The pipeline is modular, easily customizable, and parallelizable for rapid testing of design choices and hyperparameters.
- 3) The interface is accessible to non-specialists while being practical for high-level production.
- 4) The methodologies automatically implement standard best practices for machine learning.

ILI learns approximate posteriors $\hat{P}(\theta|x)$ or likelihoods $\hat{P}(x|\theta)$ from the distribution of example data-parameter pairs in a training catalog, $\mathcal{D}_{\text{train}} \equiv \{(x_i, \theta_i)\}_{i=1}^{N_{\text{train}}}$. These data can take any form; they can be sensor readings in an experiment [Dingeldein2023], images of the night sky [Ntampaka2019], or even compressed summaries of high-dimensional stochastic properties [Modi2023]. They can be gathered from a pre-run simulation [Fluri2022], be simulated on-the-fly [Alsing2019], or curated from manually-labeled observations [Lanusse2018], but they must be representative of the true underlying problem. Given these data-parameter pairs, we wish to construct a model capable of doing inference on real data, i.e., to evaluate the posterior at some observed value $x = x_{\text{obs}}$.

In modern applications, implicit inference is made tractable by using ML-models to emulate conditional probability distributions, a procedure called Neural Density Estimation (NDE).

В качестве примера рассмотрим ситуацию, когда мы пытаемся обучить сеть смеси распределений (MDN, mixture density network) для непосредственного моделирования условного распределения $P(v|u)$ с помощью обучающего набора пар «данные-параметры» (x, θ) . (Пара (u, v) может представлять (x, θ) или (θ, x) в зависимости от того, что мы оцениваем – постериор или правдоподобие, см. ниже.). Цель – создать нейросетевую архитектуру $q_w(v|u)$ с весами w , которая будет выдавать распределение вероятностей по v , соответствующее условной вероятности $P(v|u)$. MDN представляет распределение вероятностей, позволяя нейронной сети задавать статистические параметры распределения плотности смеси. Так, в случае независимых гауссовских распределений это означает, что нейронная сеть выдаёт среднее значение и дисперсию для каждого компонента смеси, т.е.

$$q_w(v|u) = \frac{1}{N_c \sqrt{2\pi}} \sum_{k=1}^{N_c} \frac{1}{\sigma_k(u; w)} \exp \left\{ -\frac{(v - \mu_k(u; w))^2}{2\sigma_k^2(u; w)} \right\}$$

где N_c – выбранное количество компонентов смеси, w – обученные веса и смещения нейросети, а $\mu_k(u; w)$ и $\sigma_k(u; w)$ – выходные данные нейросети, представляющие собой среднее значение и стандартное отклонение k -го компонента в зависимости от входных данных u и весовых коэффициентов.

Чтобы научить такую сеть правильно воспроизводить условную вероятность, нам нужно лишь максимизировать совместную вероятность наших обучающих данных $\mathcal{D}_{\text{train}} = \{x_i, \theta_i\}$. Это эквивалентно и зачастую более удобно выражается в форме минимизации отрицательного логарифма вероятности $\mathcal{L}_{\text{MDN}} = -\mathbb{E}_{\mathcal{D}_{\text{train}}} [\ln q_w(v|u)]$, где матожидание вычисляется для всех пар параметры-данные в обучающем наборе. В случае апостериорного распределения ($v = \theta, u = x$), если обучающие данные достаточно разнообразны, а MDN достаточно гибка, то минимизация \mathcal{L}_{MDN} по весовым коэффициентам сети w приведет к тому, что $q_w(\theta|x)$ будет соответствовать истинному апостериорному распределению $P(\theta|x)$.

MDN – лишь один из примеров постоянно растущего разнообразия нейросетевых методов оценки плотности в научной литературе. Другой очень популярный класс методов – *нормализующие потоки* (normalizing flows), которые выдают гибкое условное распределение, определяемое с помощью обучаемых обратимых преобразований базового гауссового распределения. Нормализующие потоки часто лучше подходят для работы с негауссовыми, но низкоразмерными и унимодальными целевыми распределениями. При любом типе оценки плотности с помощью нейросети процедура обучения всегда предполагает минимизацию отрицательного логарифма вероятности обучающих данных.

The most straightforward method to do BI is to train neural networks to directly emulate the posterior distribution, also known as Neural Posterior Estimation (NPE). The loss function for NPE is simply the negative log-posterior of our learned neural density estimator $\hat{P}(\theta|x)$, and can be trained using a loss function

$$\mathcal{L}_{\text{NPE}} = -\mathbb{E}_{\mathcal{D}_{\text{train}}}[\ln \hat{P}(\theta_i|x_i)] = -\mathbb{E}_{\mathcal{D}_{\text{train}}} \left[\ln \left(\frac{p(\theta)}{\tilde{p}(\theta)} q_w(\theta|x) \right) \right]$$

where our neural posterior $\hat{P}(\theta_i|x_i)$ is decomposed into a neural network output $q_w(\theta|x)$ and a weighting factor taken as the ratio of the assumed prior $p(\theta)$ to the proposal prior $\tilde{p}(\theta)$. The proposal prior is defined as the distribution of θ present in the training dataset $\mathcal{D}_{\text{train}}$, while the assumed prior is an experimental design choice representing the assumed knowledge of the global distribution of θ . In many cases, the assumed and proposal priors are identical, however this distinction is particularly relevant in the case of sequential learning. Note, that the normalization factor $p(\theta)/\tilde{p}(\theta)$ cancels out when optimizing \mathcal{L}_{NPE} over network weights w , but is still required for constructing the posterior estimator $\hat{P}(\theta|x)$.

This method has the advantage that it allows for quick evaluation and sampling of the full posterior at inference time. This accelerated emulation of the posterior is often called *amortized* inference. However, the disadvantage is that it requires knowledge of the analytic form of the proposal prior $\tilde{p}(\theta)$ from which training data was sampled, which may not be accessible.

An alternative method which circumvents the issue of fixed priors is to only fit for the likelihood, $P(x|\theta)$, via Neural Likelihood Estimation (NLE). The loss function for NLE notably does not include the assumed or proposal prior, and simply maximizes the global log-likelihood,

$$\mathcal{L}_{\text{NLE}} = -\mathbb{E}_{\mathcal{D}_{\text{train}}}[\ln q_w(x|\theta)]$$

Multiplying a learned likelihood with an assumed prior results in a proxy which is proportional to the posterior, i.e., $\hat{P}(\theta|x) \propto q_w(x|\theta) p(\theta)$. Generating samples from the posterior is then simply a matter of using this proxy in running MCMC.

The process of obtaining a posterior is identical to the explicit likelihood-based approach, but instead of an analytic likelihood function, NLE has essentially a “black box” function instead. The advantage of this method is that one only needs to perform training once for a given model, at the cost of additional sampling once data becomes available.

As in NLE, Neural Ratio Estimation (NRE) targets a proxy that is proportional to the posterior. Instead of outputting a conditional density estimate for the likelihood, NRE models target the likelihood ratio,

$$r(x, \theta) = \frac{P(x|\theta)}{P(x|\theta_0)},$$

equal to the ratio of the likelihood at a given point in parameter space to the likelihood at a reference point, θ_0 . This formulation is convenient because it can be folded into an acceptance ratio in MCMC and also be cast as a classification problem. The output of a classification network, $d_w(x, \theta) \in [0, 1]$, can be considered equivalent to a likelihood ratio as

$$\hat{r}(x, \theta) = \frac{d_w(x, \theta)}{1 - d_w(x, \theta)}$$

The classification problem can be interpreted as quantifying whether observed data x is consistent with θ . We can then train this neural network model using the loss:

$$\mathcal{L}_{\text{NRE}} = \mathbb{E}_{\mathcal{D}_{\text{train}}}[d_w(x, \theta)] + \mathbb{E}_{\theta' \sim p(\theta)}[1 - d_w(x, \theta')],$$

wherein we associate a positive classification with assigning a data x to the correct θ , and a negative classification with those assigning data to a θ' sampled from the prior.

The main benefit of NRE over NPE and NLE is that one need not specify an approximating distribution such as MDNs or normalizing flows to emulate a posterior. The complexity is purely limited in terms of the depth and design of the neural architecture.

Оптимальный выбор между NPE, NLE и NRE в значительной степени зависит от свойств конкретной задачи, размерностей x и θ , а также от целевого применения. Во-первых, сложность обучения форме правдоподобия или постериора может радикально различаться для разных физических задач. Если форма апостериорного распределения существенно проще, чем форма правдоподобия, то NPE будет сходиться легче, чем NLE, и наоборот. В случаях, когда и постериор, и правдоподобие имеют сложную форму, модели NRE являются предпочтительным вариантом, поскольку они не требуют явного выбора NDE.

Ещё одно надёжное эмпирическое правило заключается в том, что нейронные сети проще обучать, когда на вход подаются данные высокой размерности, а на выходе ожидается результат низкой размерности, чем наоборот. Таким образом, если размерность данных велика (например, для данных в виде изображений или графов), то NPE может работать лучше, чем NLE. И наоборот, для задач вывода с высокой размерностью апостериорного распределения, то есть когда велика размерность параметров, NLE часто показывает лучшие результаты, чем NPE, особенно при наличии сильных вырожденностей между параметрами. Реализации NRE в LtU-III также сталкиваются с трудностями в этом режиме, поскольку имеющиеся данные свидетельствуют о том, что методам NRE требуется усечение (truncation) или маргинализация для разрешения областей с высокой апостериорной плотностью.

Thus far, we have phrased our problem in such a way that we first have a set of pre-run simulations, and only later do we obtain the posterior distribution given our observations. This can be the case when the simulations are particularly expensive. In these cases, we must be careful about any difference between the proposal prior for the parameters from which the simulations were run and the chosen prior for the parameters used in the inference. It may be the case that this proposal prior is significantly wider than the posterior distribution inferred once the real data have been observed, meaning that many of the simulations used in training for NPE/NLE/NRE have been “wasted” as the posterior has almost no support in the region of parameter space occupied by a (potentially significant) fraction of the simulations.

One can instead adopt a multi-round inference approach to improve simulation efficiency, referred to as *Sequential* NPE/NLE/NRE or SNPE/SNLE/SNRE. In this method, we begin by running a fraction of the total desired simulations with parameters sampled from the broad prior $p(\theta) = \tilde{p}(\theta)$, conducting the first round of training for the NPE/NLE/NRE model. This weak inference is then applied to x_{obs} to derive a first estimate for the posterior. For instance, in the case of NPE, this is equivalent to minimizing \mathcal{L}_{NPE} with $p(\theta) = \tilde{p}(\theta)$ during the first round, which gives a model $q_w(\theta|x)$, and then retraining the model with new simulations drawn from $\tilde{p}(\theta) = q_w(\theta|x = x_{\text{obs}})$ for the second round, and so on. The algorithm then identifies regions of high posterior density around the observation x_{obs} and conducts additional simulations within these areas to produce a refined posterior estimate. This can be repeated until a convergence criterion is met or the total simulation budget is exhausted. This iterative refinement of the posterior around the region of interest can lead to improved posterior estimates for a given experiment.

4.3.1 Валидация метода

The goal of model *validation* is to assess whether the learned posteriors $\hat{P}(\theta|x_{\text{obs}})$ will be accurate and reliable when applied to new data. Explicitly, we want to evaluate: (1) whether the learned posteriors are maximally constraining of θ given the observed data x_{obs} , and (2) whether the predictive uncertainties quoted by our learned model are accurately calibrated to our training data. These two criteria are naturally adversarial, and is often referred to as the *bias-variance tradeoff*. For example, a model can satisfy condition (1) by reporting tight error bars, but it will then fail condition (2) when its error bars are smaller than the true distribution of training data. Similarly, a model that produces a posterior equal to the prior will easily satisfy condition (2) but will ultimately be uninformative for solving condition (1). Despite the ‘black box’ nature of neural networks, these criteria control the quality of the learned implicit inference posteriors, mitigating the chance that the learned model is not representative of the data used to train it.

We evaluate the constraints and coverage of our learned posteriors on a labeled test set of data-parameter pairs $\mathcal{D}_{\text{test}} \equiv \{(x_i, \theta_i)\}_{i=1}^{N_{\text{test}}}$. This test set is meant to represent a fully independent sampling of the real data distribution, unseen by the model prior to validation. This test set may originate from the same source as our labeled training set $\mathcal{D}_{\text{train}}$ but must be held independent to avoid overfitting and underestimation of the uncertainty. If the distribution of data-parameter pairs in the test set matches that of the predictions of our learned posterior, then our model is calibrated and can be reliably extended to new, unlabeled data.

Мы проводим тесты, сравнивая истинные параметры из тестового набора с постериорными сэмплами, полученными с помощью наших моделей. Сравнение на основе сэмплирования обеспечивает единую основу для всех стратегий обучения NPE/NLE/NRE, поскольку не требует нормализации постериорного распределения. Технически сэмплирование из нейросетей зависит от реализованной стратегии. Здесь будем исходить из того, что для заданных входных данных x мы можем получить независимые и одинаково распределённые (i.i.d.) сэмплы из обученных постериоров, $\hat{\theta} \sim \hat{P}(\theta|x)$. Эти выборки можно использовать для сравнения уровня ограничений на θ и соответствия истинным значениям.

Чтобы проверить точность, отражающую способность обученной модели определять значения параметров, мы изучаем распределение и форму постериорных выборок $\hat{\theta}$ вокруг истинного значения θ . Эти распределения обычно визуализируют с помощью многомерных corner-диаграмм или маргинальных диаграмм «истина vs. прогноз». Контурные на этих диаграммах помогают качественно оценить способность модели определять значения параметров, выявить вырожденность постериоров и обнаружить систематические ошибки.

Количественным показателем точности является кумулятивное правдоподобие тестового набора данных, $\hat{P}(\mathcal{D}_{\text{test}}) = \prod_{i=1}^{N_{\text{test}}} \hat{P}(\theta_i|x_i)$. Чем выше тестовое правдоподобие, тем большую долю вероятности постериорная модель сконцентрирует вокруг истинного значения, а значит, будет более точной. Для NPE, которые напрямую оценивают $P(\theta|x)$, правдоподобие теста вычислить просто. Однако для NLE и NRE оценка правдоподобия потребует дополнительного этапа обучения генеративной модели $G(\theta)$ на основе полученных постериорных сэмплов $\hat{\theta}$. Этот процесс требует значительных вычислительных ресурсов, особенно для больших наборов данных, кроме того, он нечувствителен к избыточной уверенности (overconfidence).

Наконец, имея выборки из эталонного (reference) апостериорного распределения, мы можем оценить расхождение между прогнозом модели и его оптимальным значением. Эталонный постериор часто можно получить с помощью длинных цепочек MCMC для

подходов с явным правдоподобием, а затем использовать для сравнения с методами ИЛ. В [Lueckmann2021] проведен обширный количественный анализ различных выборочных расстояний для оценки методов ИЛ. Их вывод: тест «Классификатор двух выборок» (C2ST) является наиболее строгим показателем точности. C2ST определяется как точность обученного классификатора (часто это нейронная сеть), который различает истинные и постериорные выборки параметров. Высокий показатель C2ST говорит о том, что истинное и постериорное значения сильно отличаются, в то время как показатель C2ST, равный примерно 0.5, говорит о том, что они практически неразличимы.

Posterior samples are also useful for quantifying the calibration of our model uncertainty in parameter space. We can construct a direct comparison of the predicted percentiles by our inference engine to the true error observed in our validation dataset. We first define the Probability Integral Transform (PIT) as the cumulative density function (CDF) of our model posterior given an input x_{obs} ,

$$\text{PIT}(\theta; x_{\text{obs}}) = \int_{-\infty}^{\theta} d\theta \hat{P}(\theta|x_{\text{obs}}).$$

Given i.i.d. samples from the posterior, we can construct an estimator for the PIT value as

$$\hat{\text{PIT}}(\theta; x_{\text{obs}}) = \mathbb{E}_{\hat{\theta} \sim \hat{P}(\theta|x_{\text{obs}})}[\Theta(\hat{\theta} - \theta)]$$

where Θ is the Heaviside step function. Stated plainly, the PIT value counts the number of times that our posterior samples $\hat{\theta}$ fall below the true parameter value θ . If our model posteriors are globally consistent with the truth, i.e. we match the true posterior everywhere in data-parameter space, then the distribution of PIT values evaluated on data-parameter pairs from our test set must be uniformly distributed: $\text{PIT}(\theta_i; x_i) \sim U(0, 1)$, for all $(x_i, \theta_i) \in \mathcal{D}_{\text{test}}$. As a result of this property, the distribution of PIT values is a common goodness-of-fit metric for conditional density models. This is also known as Simulation Based Calibration.

To better interpret errors in the PIT distribution, we can use a percentile-percentile (P-P) plot, explicitly comparing the CDF of PIT values to the CDF of a uniform random variable. Intuitively, the P-P plot measures: ‘What percentile level is my posterior model assigning to the true value, and with what frequency does this occur in the test set?’ If the predictive posterior is well-calibrated, these should be the same, e.g., we should predict the true value below the 50-th percentile 50% of the time. If it is not, the shape of the P-P plot acts as a sensitive probe of global bias or over-/under-dispersion of predictive uncertainty. This is an extremely practical tool for understanding and correcting biases and over-/under-dispersion in complex ИЛ posteriors.

Также достаточно использовать аппроксимацию для проверки многомерного *постериорного покрытия* (posterior coverage). Проверка точности с помощью случайных точек (TARP, Tests of Accuracy with Random Points) использует выборки в многомерном пространстве параметров для оценки ожидаемых вероятностей покрытия. Анализируя плотность апостериорных выборок в областях пространства параметров, близких к истинным значениям, TARP формирует оценки постериорного покрытия, которые гарантированно сходятся к истинному постериорному покрытию при достаточном количестве сэмплов.

4.3.2 Конвейер LtU-ИЛ

LtU-ИЛ представляет собой конвейер (pipeline) для обучения ИЛ-моделей, которые позволяют получать (to infer) скалярные параметры на основе данных наблюдений. Типичный конвейер состоит из трёх основных этапов: *сбора данных* (Data), *вероятностного вывода*

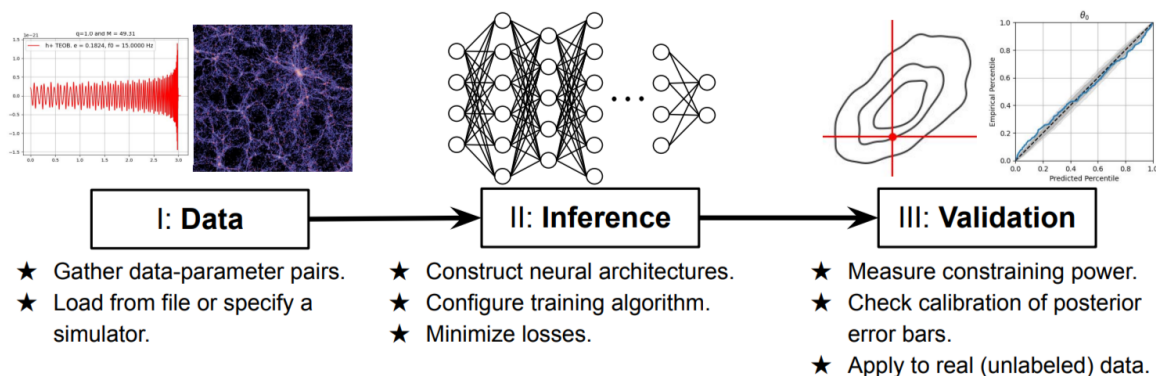


Рис. 3: Схематическое представление конвейера LtU-ILI с описанием процессов на каждом этапе. Каждый из трёх этапов (Data, Inference, Validation) настраивается независимо.

(Inference) и *валидации* (Validation), см. рис. 3. На этапе сбора данных происходит загрузка наборов данных из файлов или настройка симуляторов, чтобы предоставить пары «данные–параметры» для III. Затем на этапе вероятностного вывода эти пары используются для обучения нейронных сетей, которые связывают их с апостериорными представлениями или представлениями правдоподобия. Обученные (fitted) модели сохраняются в файл. Наконец, на этапе валидации загружается тестовый набор данных, а также обученные модели с предыдущего этапа, чтобы оценить показатели соответствия (goodness-of-fit metrics, например, уровень ограничений, постериорное покрытие и т.п.) и выполнить вероятностный вывод для неразмеченных данных.

Каждый этап является независимым и адаптивным, то есть конфигурацию одного этапа можно изменить, не затрагивая другие, и конвейер продолжит работать. Например, если у вас есть фиксированный набор данных, но вы хотите опробовать множество различных нейросетевых архитектур, вам нужно будет изменить только конфигурацию вероятностного вывода. Аналогично, если вы хотите увидеть, насколько эффективнее использование короткого, консервативного набора данных по сравнению с более длинным и информативным набором, вы можете просто изменить конфигурацию Data, чтобы загрузить разные наборы данных, а процессы Inference и Validation пройдут точно так же.

Цель этапа Data – собрать пары «данные–параметры» и представить их в удобном формате для этапа Inference. Для стандартного вывода с фиксированными обучающими и тестовыми наборами данных этап Data просто загружает пары «данные–параметры» из оперативной памяти или хранилища на диске и предоставляет вспомогательные функции `get_all_data()` и `get_all_parameters()`, чтобы передавать данные и параметры в правильном формате. Эти функции вызываются на этапе Inference для последующей предобработки в рамках бэкендов `sbi`, `pydelfi` и `lampe`. В LtU-ILI есть несколько удобных классов для подготовки данных из форматов, подобных `numpy`, `hdf5` или `torch`. Пользователи также могут создавать собственные объекты для загрузки данных.

Для многораундового вывода этап Data также предоставляет функцию `simulate()`, в которой загрузчик выполняет симуляцию «на лету» для заданных параметров, чтобы сгенерировать данные для эффективного последовательного обучения. В примере класса, предоставленного в LtU-ILI, пользователи могут инициализировать этот этап с помощью произвольной модельной функции, которая принимает параметры θ и возвращает данные x ; этап последовательного Inference будет обрабатывать все вызовы этой функции. Кроме того, пользователи могут загружать предварительно сгенерированный начальный каталог из файла и записывать любые вновь созданные симуляции на диск в процессе

обучения.

Каждый этап Inference (вывод) требует указания: типа оценки (апостериорного распределения, правдоподобия или отношения правдоподобия); одной или нескольких нейросетевых архитектур; априорного распределения; гиперпараметров обучения (например, скорость обучения, размер батча и т.д.). После предоставления этих данных Inference знает, как извлекать пары «данные-параметры» из этапа Data, выполнять предобработку данных, конструировать каждый алгоритм обучения, минимизировать функции потерь и формировать обученную модель постериора $\hat{P}(\theta|x)$. Каждый алгоритм автоматически реализует лучшие практики обучения, включая нормализацию данных, адаптивный стохастический градиентный спуск и раннюю остановку на основе валидации.

Inference построен на трёх различных вычислительных бэкендах: 1) pydelfi – пакет на Tensorflow для выполнения NLE с использованием методов активного обучения и сжатия данных. 2) sbi – пакет на PyTorch для выполнения NPE, NLE и NRE, а также их последовательных аналогов. 3) lampe – также пакет на PyTorch, ориентированный на методы NPE, с обширной поддержкой разнообразных embedding-сетей и моделей нормализующих потоков.

Бэкенды Tensorflow и PyTorch имеют разнообразные предустановленные реализации NDE и классификаторов отношений (ratio classifiers). NDE представляют собой либо MDN, либо нормализующие потоки (normalizing flows). Бэкенд pydelfi обладает собственными нейросетевыми архитектурами, тогда как модели sbi и lampe имеют свои зависимости от стороннего ПО – соответственно nflows и zuko. Оба бэкенда также поддерживают различные стандартные и настраиваемые априорные распределения. Основное требование к пользовательскому прайору – возможность вычислять $\ln p(\theta)$ (для моделей NDE) или пропорциональный прокси (для моделей NLE/NRE).

Ключевым компонентом LtU-ILI является ансамблирование моделей, при котором несколько нейронных сетей независимо обучаются на одном и том же наборе данных и их предсказания объединяются для повышения устойчивости (robustness) к переобучению и учёта неопределённости модели. Отдельные NDEs склонны к излишней уверенности (overconfidence), т.е. у них есть тенденция недооценивать прогностическую неопределённость. Для обеспечения надёжного вывода в LtU-ILI мы рекомендуем использовать глубокое ансамблирование (Deep ensembling) – хорошо зарекомендовавшую себя схему коррекции излишней уверенности путём усреднения прогнозов NDE от нескольких моделей, что позволяет расширить доверительные интервалы и корректно учесть неопределённость модели. (Это альтернатива вычислительно затратным байесовским нейронным сетям, в которых полное апостериорное распределение весов изучается во время обучения и сэмплируется во время вывода.)

Адаптивность нейронных сетей к различным типам данных (например, в астрономии и космологии) может быть расширена за счёт настраиваемых embedding-сетей (сетей внедрения/кодирования) в удобных реализациях различных NDE и классификаторов в LtU-ILI. Эти embedding-сети могут использовать классические нейронные слои в стиле Keras как в Tensorflow, так и в PyTorch, что значительно упрощает реализацию. Интеграция с embedding-сетями в lampe позволяет работать с экзотическими типами входных данных, такими как графы и последовательности. В коде мы предоставляем несколько примеров embedding-сетей, включая подключение свёрточных нейронных сетей (CNN) и графовых нейронных сетей (GNN).

Этап Validation предоставляет модульные функции для применения описанных ранее метрик к полученным апостериорным распределениям $\hat{P}(\theta|x)$. Процесс Validation состоит из следующих шагов: 1) Загрузить обученные нейросетевые постериоры с этапа Inference.

2) Сэмплировать $\hat{\theta} \sim \hat{P}(\theta|x)$ в тестовых точках данных $\mathcal{D}_{\text{test}}$ или в точке ненаблюдаемых данных x_{obs} . 3) Рассчитать метрику валидации и вернуть или сохранить её в файл.

Наиболее важной настройкой Validation является выбор метода сэмплирования. LtU-III предоставляет три класса сэмплеров: MCMC, вариационный вывод (VI) и прямое сэмплирование. Все модели (NPE/NLE/NRE) в обоих бэкендах нативно интегрированы с пакетом сэмплирования емсее, который эффективен за счёт параллелизации и аффинных преобразований, но не использует градиенты вероятности. Модели sbi также имеют доступ к сэмплерам из PyTorch-пакета rugo, включая slice sampler, НМС, NUTS. Мы также предоставляем функциональность для моделей sbi, позволяющую аппроксимировать и сэмплировать из NDE с помощью нейронного вариационного вывода (neural VI), что особенно полезно при высоких размерностях параметров. Прямое сэмплирование, которое быстрее, чем MCMC и VI, можно использовать только для моделей NPE в sbi.

4.3.3 Модельные эксперименты

Сначала строится игрушечная модель, чтобы продемонстрировать способность LtU-III изучать нелинейные постериоры. Мы рассматриваем 10-мерный вектор данных x , построенный поэлементно с помощью следующего стохастического симулятора:

$$x_i = 3 \sin(k_i + \phi_0) + \phi_1 \cdot k_i^2 + \epsilon_i,$$

где $k_i = (2i/3) - 3$, $i \in [0, 9]$. Здесь вектор данных $\mathbf{x} \equiv \{x_i\}_{i=0}^9 \in \mathbb{R}^{10}$ состоит из синусоидального сигнала с начальной фазой ϕ_0 , квадратичного сигнала с амплитудой ϕ_1 и независимой шумовой компоненты $\epsilon_i \sim \mathcal{N}(0, 1)$. Мы далее раскладываем ϕ на целевые параметры $\theta \in \mathbb{R}^3$: $\phi_0 = \theta_0 + \theta_1$, $\phi_1 = \theta_1 - 3\theta_2^2$, и ожидаем, что наш механизм вывода перенесёт ограничения на латентные переменные ϕ на целевые переменные θ . Априорное распределение для каждого интересующего нас параметра предполагается стандартным нормальным, $p(\theta_i) = \mathcal{N}(0, 1)$.

Это довольно сложная задача для вероятностного вывода, поскольку функция правдоподобия сильно нелинейна, а апостериорные распределения демонстрируют сильные вырожденности. Однако изучение параметров по таким сложным, скоррелированным векторам данных – типичная проблема в астрономии и космологии.

С помощью данного симулятора мы обучаем sbi-SNPE, используя ансамбль из двух нормализующих потоков (MAF), и получаем апостериорное распределение на параметры θ , рис. 4. Здесь оно показано в сравнении с эквивалентными ограничениями, полученными методами Rejection ABC и длинных цепочек НМС.

Несмотря на такой же бюджет симуляций, как у ABC, ограничения постериора SNPE сошлись к референтному и являются значительно более жёсткими, чем ограничения, полученные ABC. Как в постериоре SNPE, так и в НМС, мы явно наблюдаем ожидаемую линейную вырожденность между θ_0 и θ_1 , а также квадратичную вырожденность между θ_1 и θ_2 , что является прямым следствием упомянутых выше определений ϕ . Несмотря на эту нелинейность, мы видим, что многомерные 68% и 95% доверительные интервалы полностью согласуются между SNPE и НМС. В тоже время методы ABC испытывают трудности в этом режиме. Поскольку размерность данных равна десяти, ABC требует на порядки больше выборок, чтобы полностью охватить хвосты постериора.

Эта задача демонстрирует полезность наличия амортизированной модели апостериорного распределения с низкой стоимостью, такой как SNPE, особенно в случаях, когда симуляторы являются не столь простыми.

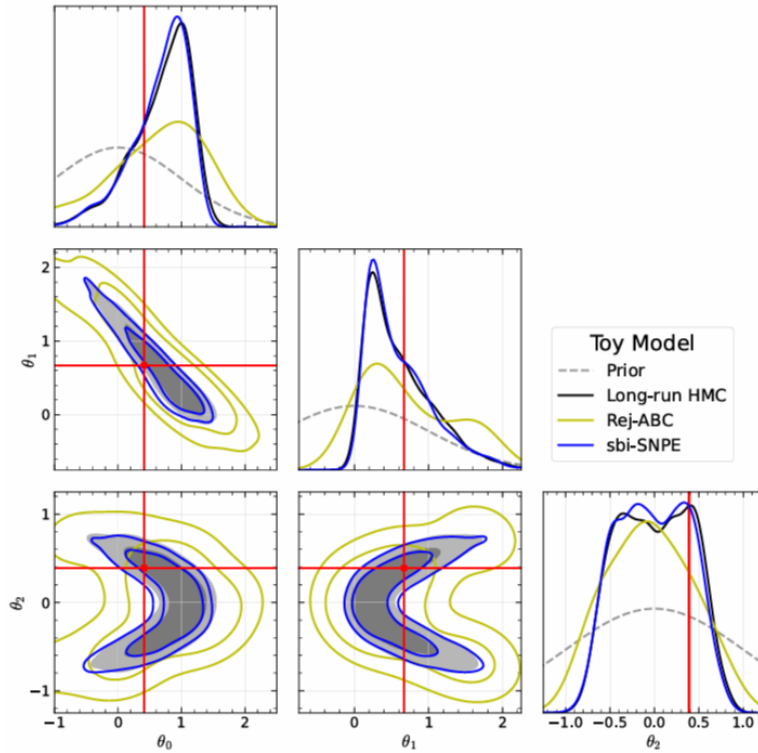


Рис. 4: Пример вывода апостериорного распределения с помощью SNPE в сравнении с классическими методами вывода Rejection ABC и НМС (с явным правдоподобием). Истинное значение каждого параметра показано красным, а контуры соответствуют центральным 68% и 95% доверительным интервалам.

Сравнительный анализ (бенчмаркинг) всех моделей в LtU-ILI проводим на стандартной задаче SLCP (Simple Likelihood Complex Posterior), широко используемой в экспериментах по SBI. SLCP имеет унимодальную функцию правдоподобия, но весьма непростое многомодальное апостериорное распределение, что делает её сложным, но достижимым эталоном для неявного вывода. Мы следуем реализации SLCP из [Lueckmann2021], в которой обучаем различные методы NPE/NLE/NRE и сравниваем полученные апостериорные распределения с референтными постериорами, определёнными с помощью длинных цепочек НМС. Мы реализуем это для всех доступных методов NPE/NLE/NRE и их последовательных аналогов в каждом из бэкендов `pydelfi`, `sbi` и `lampe`, а также включаем в сравнение Rejection ABC как базовый метод.

Для каждого испытания измерлась ошибка вывода относительно референтного постериора с помощью C2ST. На рис. 5 показана ошибка C2ST в зависимости от бюджета симуляций, использованного для создания обучающего набора. Как и следовало ожидать, точность выведенных апостериорных распределений возрастает с увеличением количества предоставленных симуляций. Все модели значительно превосходят базовый метод Rejection ABC благодаря эффективному использованию симуляционных данных. Методы NLE демонстрируют наилучшие результаты, методы NPE следуют близко за ними, а методы NRE показывают наихудшую работоспособность, что несомненно связано с тем, что SLCP по определению имеет простое, легко обучаемое правдоподобие.

Последовательные методы (SNPE/SNLE/SNRE) показывают незначительное улучшение при малых бюджетах симуляций (10^3), но демонстрируют огромный выигрыш при больших бюджетах (10^5), особенно для NLE. Это связано с тем, что для малых обучаю-

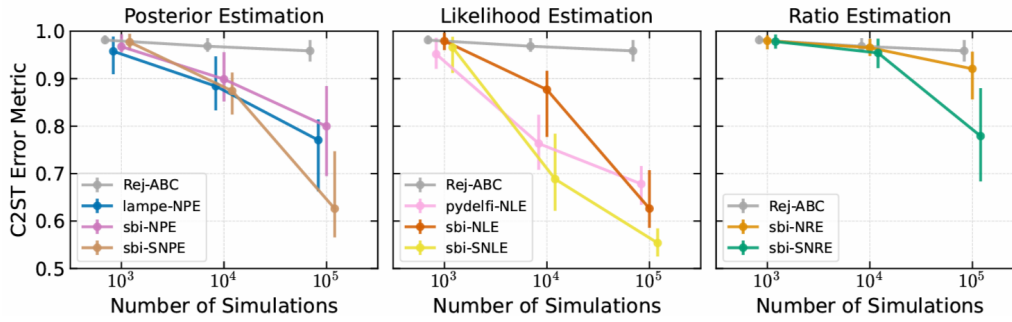


Рис. 5: Ошибка C2ST в зависимости от бюджета симуляций для различных методов LtU-ILI для задачи SLCP (Rejection ABC показан на всех графиках в качестве базовой линии). Более низкое значение C2ST указывает на более точное апостериорное распределение, причём оптимальным значением является 0.5. Значения C2ST показаны в виде медианы и центрального 95% доверительного интервала, рассчитанных по 10 независимым запускам.

щих наборов данных постериоры слабо ограничены и почти не отклоняются от прайоров. Однако при жёстких ограничениях в прогонах с большим бюджетом, последовательные методы способны более эффективно фокусировать ресурсы симуляций на малых областях пространства параметров.

4.3.4 Научные эксперименты

В статье приведены примеры применения LtU-ILI для типичных задач астрофизики и космологии. При этом демонстрируется многофункциональность программного пакета:

1. *Оценка массы скоплений галактик по рентгеновским данным*: использование CNN в качестве embedding-сетей (см. раздел A.1).
2. *Спектр мощности темной материи в проекте Quijote*: применение валидации покрытия апостериорного распределения (posterior coverage validation) (см. раздел A.2).
3. *Полевой вывод в Quijote*: включение слоёв графовых нейронных сетей (Graph Neural Network, GNN) (см. раздел A.3).
4. *Частотные сигналы гравитационно-волнового отклика с детекторов Хэнфорда (H1) и Ливингстона (L1)*: использование многораундового вывода (см. раздел A.4).
5. *Вероятностный вывод параметров межзвездной пыли*: количественная оценка информации из различных источников данных (см. раздел A.5).
6. *Массовая и энергетическая нагрузка галактических ветров*: интеграция вложений, оптимальных по информации (information optimal embeddings) (см. раздел A.6).

Код и данные всех этих экспериментов находятся в открытом доступе в репозитории <https://github.com/maho3/ltu-ili>.

4.3.5 Дискуссия и выводы

Методы ILI гарантированно восстанавливают истинное апостериорное распределение при следующих допущениях:

- 1) Имеется достаточное количество обучающих данных, чтобы охватить пространство данных и параметров.
- 2) Обучающие данные или симулятор адекватно отражают реальность.
- 3) Выбранная нейросетевая архитектура достаточно гибка, чтобы уловить зависимость между данными и параметрами.

Несмотря на лучшие практики, реализованные в LtU-ILI, смещения (biases) могут возникать при нарушении этих допущений. Прежде чем применять эти подходы к реальным наблюдательным данным, крайне важно понимать эти режимы сбоя, а также то, как строить конвейеры вывода, устойчивые к ним. Ниже мы рассматриваем важные аспекты и даём качественные рекомендации по использованию ИИ в наблюдательных исследованиях.

Во-первых, неверная спецификация модели (model misspecification) возникает, когда функция генерации данных для создания обучающих данных плохо отражает физические процессы в реальном мире [Cannon2022]. Это может привести к тому, что неявная модель интерпретирует неверную связь «данные-параметры», что приводит к смещённым предсказаниям. В астрофизике и космологии борьба с неверной спецификацией модели часто требует запуска более реалистичных, высокоразрешающих симуляторов, которые могут быть вычислительно гораздо более затратными. Это, в свою очередь, сокращает количество доступных симуляций, нарушая допущение (1). Альтернативные подходы включают: 1) Увеличение модели шума в данной симуляции (например, [Modi2023]). 2) Обучение NDE с консервативной функцией потерь, чтобы расширить неопределённости и противодействовать неверной спецификации (например, [Kelly2023]; [Huang2023]). 3) Использование латентных функций, присутствующих в промежуточных слоях байесовской иерархической модели, в качестве диагностики (например, [Leclercq2022]).

4.4 [Статья \[Gloeckler2024\]](#)

Название: All-in-one simulation-based inference

Авторы: Manuel Gloeckler, Michael Deistler, Christian Weilbach, Frank Wood, Jakob H. Macke

Аннотация: Амортизированный байесовский вывод обучает нейронные сети решать стохастические задачи вывода с использованием моделирующих симуляций, что позволяет быстро выполнять байесовский вывод для любых вновь наблюдаемых данных. Однако современным методам амортизированного SBI необходимы значительные вычислительные ресурсы. Кроме того, они обладают ограниченной гибкостью, так как требуют предварительного задания фиксированных параметрических априорных распределений, симуляторов и задач вывода. В данной работе представлен новый метод амортизированного вывода – Simformet, – который преодолевает эти ограничения. Обучая вероятностную диффузионную модель с использованием архитектур трансформеров, Simformet превосходит современные подходы к амортизированному выводу на эталонных задачах и является значительно более гибким: он может применяться к моделям с параметрами в виде функций; он может обрабатывать сценарии вывода с неполными или неструктурированными данными; он может сэмплировать произвольные условные распределения совместного распределения параметров и данных, включая как апостериорные распределения, так и правдоподобия. Мы демонстрируем работоспособность и гибкость Simformet на симуляторах из экологии, эпидемиологии и нейронауки и показываем, что он открывает новые возможности и области применения для амортизированного байесовского SBI.

В последнее время было разработано множество методов амортизированного SBI. Хо-

тя эти методы имеют различные сильные и слабые стороны, большинство из них также имеют общие ограничения. Во-первых, они часто полагаются на структурированные табличные данные (обычно векторы θ , \mathbf{x}). Однако реальные наборы данных часто более сложны: нерегулярно сэмплированные временные ряды естественным образом возникают в таких областях, как экология, климатология и науки о здоровье; пропущенные значения часто встречаются в реальных наблюдениях и нелегко обрабатываются существующими подходами. Во-вторых, входные данные симулятора могут соответствовать функции времени или пространства, т.е. бесконечномерным параметрам. Существующие амортизированные методы обычно требуют дискретизации, тем самым ограничивая их применимость конкретной, часто плотной сеткой и не позволяя оценивать апостериорное распределение параметров за её пределами. В-третьих, они требуют фиксации конкретной задачи аппроксимации: нейронная сеть может быть нацелена либо на правдоподобие (NLE), либо на апостериорное распределение (NPE). На практике пользователям может потребоваться интерактивно исследовать оба условных распределения, изучать апостериорные распределения, обусловленные подмножествами данных и параметров, или даже исследовать различные конфигурации априорных распределений. В-четвёртых, хотя методы SBI на основе нейронных сетей более эффективны, чем классические методы ABC, они всё ещё требуют большого количества симуляций. Отчасти это связано с тем, что они ориентированы на симуляторы как «чёрные ящики», т.е. не требуют доступа к внутренним механизмам модели. Однако на практике у нас есть хотя бы частичное знание (или предположения) о структуре симулятора (например, его условные независимости), но стандартные методы SBI не используют эти знания. Эти ограничения препятствовали применению SBI в интерактивных приложениях, где свойства задачи необходимо менять «на лету».

В этой работе мы разрабатываем новый метод амортизированного байесовского вывода — Simformer, который преодолевает эти ограничения, используя комбинацию *трансформеров* и *вероятностных диффузионных моделей*. Наш метод может работать с неструктурированными данными и пропусками в данных, а также обрабатывать как параметрические, так и непараметрические симуляторы (т.е. с бесконечномерными параметрами в виде функций). Кроме того, метод возвращает единую сеть, к которой можно обращаться для сэмплирования всех условных распределений совместного распределения, а также выполнять вывод, если наблюдения представляют собой интервалы, а не конкретные значения. На эталонных (benchmark) задачах этот метод имеет более высокую точность, чем предыдущие методы SBI (при заданном бюджете симуляций). Более того, используя *маски внимания* (attention masks), можно применять предметные знания для адаптации Simformer к структуре зависимостей симулятора для дальнейшего повышения эффективности симуляций.

4.4.1 Трансформеры, механизм внимания и диффузионные модели

Рассмотрим симулятор с параметрами θ , который стохастически генерирует выборки \mathbf{x} из своего неявного правдоподобия $p(\mathbf{x}|\theta)$. После получения наблюдаемых данных \mathbf{x}_{obs} мы стремимся вывести апостериорное распределение $p(\theta|\mathbf{x}_{\text{obs}})$ параметров при заданных данных, но также сохранить гибкость для получения любого другого условного распределения полного совместного распределения $p(\theta, \mathbf{x})$. Для этого введем совместный вектор $\hat{\mathbf{x}} = (\theta, \mathbf{x})$, который будет служить входом для трансформера вместе с маской, указывающей, какие значения являются наблюдаемыми. Затем трансформер будет использовать механизмы внимания (attention) для вычисления соответствующей последовательности выходных оценок (scores) того же размера. Оценки, соответствующие ненаблюдаемым

переменным, затем станут основой для диффузионной модели, представляющей распределение по этим переменным.

Трансформеры преодолевают ограничения полносвязных сетей в эффективной работе с последовательными входами. Они включают механизм внимания, который для заданной последовательности входов заменяет отдельные скрытые состояния взвешенной комбинацией всех скрытых состояний. При наличии трёх обучаемых линейных проекций каждого скрытого состояния (Q, K, V) внимание вычисляется как:

$$\text{attention}(Q, K, V) = \text{softmax}\{QK^\top/\sqrt{d}\}V$$

Оценка на основе диффузионных моделей (score-based diffusion models) описывает эволюцию данных через стохастические дифференциальные уравнения (SDEs). Типичные SDE для таких моделей можно выразить как:

$$d\hat{\mathbf{x}}_t = f(\hat{\mathbf{x}}_t, t) dt + g(t) d\mathbf{w},$$

где \mathbf{w} – стандартный винеровский процесс, а f и g представляют коэффициенты сноса (drift) и диффузии (diffusion) соответственно. Решение этого SDE определяет диффузионный процесс, который преобразует исходное распределение данных $p_0(\hat{\mathbf{x}}_0) = p(\hat{\mathbf{x}})$ в более простое распределение шума $p_T(\hat{\mathbf{x}}_T) \approx \mathcal{N}(\hat{\mathbf{x}}_T; \mu_T, \sigma_T)$.

Затем выборки из генеративной модели создаются путём симуляции обратного диффузионного процесса:

$$d\hat{\mathbf{x}}_t = [f(\hat{\mathbf{x}}_t, t) - g(t)^2 s(\hat{\mathbf{x}}_t, t)] dt + g(t) d\tilde{\mathbf{w}},$$

где $\tilde{\mathbf{w}}$ – винеровский процесс, идущий назад во времени. Это опирается на знание функции оценки (score function) $s(\hat{\mathbf{x}}_t, t) = \nabla_{\hat{\mathbf{x}}_t} \ln p_t(\hat{\mathbf{x}}_t)$ на каждом шаге. Точная маргинальная оценка обычно невычислима, но её можно оценить с помощью временного согласования оценок с удалением шума (time-dependent denoising score-matching). Учитывая, что условная оценка известна, $p_t(\hat{\mathbf{x}}_t | \mathbf{x}_0) = \mathcal{N}(\hat{\mathbf{x}}_t; \mu_t(\hat{\mathbf{x}}_0), \sigma_t(\hat{\mathbf{x}}_0))$, модель оценки $s_\phi(\hat{\mathbf{x}}_t, t)$ обучается для минимизации потерь:

$$\mathcal{L}(\phi) = \mathbb{E}_{t, \hat{\mathbf{x}}_0, \hat{\mathbf{x}}_t} [\lambda(t) \|s_\phi(\hat{\mathbf{x}}_t, t) - \nabla_{\hat{\mathbf{x}}_t} \ln p_t(\hat{\mathbf{x}}_t | \hat{\mathbf{x}}_0)\|_2^2],$$

где λ обозначает положительную весовую функцию. Таким образом, эта целевая функция требует только выборок из исходного распределения $\hat{\mathbf{x}}_0 \sim p(\hat{\mathbf{x}})$.

4.4.2 Simformer

Simformer – вероятностная диффузионная модель, которая для оценки сора (score) использует трансформер. В отличие от большинства предыдущих подходов SBI, которые используют условные оценки плотности для моделирования либо правдоподобия, либо апостериорного распределения, Simformer обучается на совместном распределении параметров и данных $p(\boldsymbol{\theta}, \mathbf{x}) \equiv p(\hat{\mathbf{x}})$. Simformer кодирует параметры и данные таким образом, что произвольные условные распределения совместной плотности (включая апостериорное распределение и правдоподобие) по-прежнему могут быть эффективно сэмплированы. Simformer может кодировать известные зависимости в маске внимания (attention mask) трансформера и тем самым обеспечивать эффективное обучение произвольных условных распределений. Наконец, Simformer использует управляемую диффузию (guided diffusion) для генерации выборок при заданных произвольных ограничениях.

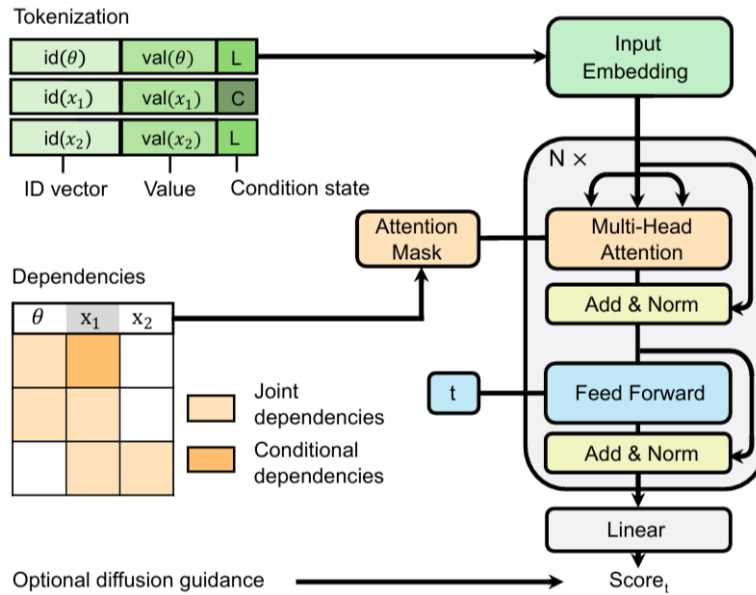


Рис. 6: Архитектура Simformer. Все переменные (параметры и данные) преобразуются в представление в виде токенов, которое включает идентификатор переменной (id), значение переменной (val), состояние условия – латентное (L) или обусловленное (C). Эта последовательность токенов обрабатывается моделью трансформера; взаимодействие переменных может быть явно контролируемо с помощью маски внимания (attention mask). Архитектура трансформера возвращает оценку (score), которая используется для генерации выборок из диффузионной модели, основанной на оценках, и может быть модифицирована.

Трансформеры обрабатывают последовательности векторов одинакового размера, называемых токенами (tokens). Проектирование эффективных токенов является сложной задачей и специфично для конкретных данных. Токенизатор представляет каждую переменную как: идентификатор, однозначно определяющий переменную; представление значения переменной; состояние условия (condition state) (рис. 6). Состояние условия – это бинарная переменная, которая указывает, является ли переменная обусловленной (conditioned on) или нет. Оно перебирается для каждой пары $(\theta, \mathbf{x}) \in \mathbb{R}^d$ на каждой итерации обучения. Мы обозначаем состояние условия всех переменных как $M_C \in \{0, 1\}$. Установка $M_C = (0, \dots, 0)$ соответствует безусловной диффузионной модели. Использование $M_C^{(i)} = 1$ для данных и $M_C^{(i)} = 0$ для параметров соответствует обучению условной диффузионной модели для апостериорного распределения. В наших экспериментах мы равномерно случайным образом выбираем либо маски для совместного распределения, апостериорного распределения, правдоподобия, либо две случайно выбранные маски. Чтобы сфокусироваться на конкретных условных распределениях, можно просто изменить распределение масок условий.

Simformer использует обучаемые векторные представления (embeddings) для идентификаторов и состояний условия. В случаях, когда параметры или данные являются функциями пространства или времени, идентификатор узла будет включать общий вектор представления и случайное фурье-представление элементов в множестве индексов. Наконец, специализированные embedding-сети, обычно используемые в алгоритмах SBI и обучаемые сквозным образом (end-to-end), могут быть эффективно интегрированы здесь путём свёртки сложных данных в единый токен. Это снижает вычислительную сложность, но лишает прямого контроля над зависимостями и состояниями условия для отдельных

элементов данных.

Для некоторых симуляторов у учёных-специалистов могут быть знания (или предположения) об условной структуре зависимостей между параметрами и данными. Например, может быть известно, что все параметры независимы, или каждый параметр может влиять только на одно значение данных. Simformer может использовать эти зависимости, представляя их в маске внимания M_E трансформера. Эти ограничения могут быть реализованы как ненаправленные зависимости (через симметричную маску внимания), или направленные зависимости (через несимметричную маску внимания), которые позволяют навязывать причинно-следственные связи между параметрами и наблюдениями. Однако последний вариант требует обновления маски, если зависимости изменяются, например, из-за условного ограничения (conditioning).

Ключевое преимущество по сравнению с прямым маскированием весов заключается в том, что маску внимания можно легко динамически адаптировать во время обучения или вывода, что позволяет навязывать структуры зависимостей, которые зависят от входных значений и состояния условия. Мы отмечаем, что одной лишь маски внимания M_E , как правило, недостаточно для обеспечения конкретных условных независимостей и свойств маргинализации в многослойных трансформерных моделях. Мы описываем свойства, которые можно надёжно гарантировать, а также исследуем, как M_E можно эффективно использовать для изучения определённых желаемых свойств.

Определив токенизатор, который обрабатывает каждую пару (θ, \mathbf{x}) , и маску внимания для указания зависимостей внутри симулятора, Simformer можно обучить с использованием согласования оценок с удалением шума (denoising score-matching). Мы выбираем уровень шума t для диффузионной модели равномерно случайным образом и генерируем (частично) зашумлённую выборку:

$$\hat{\mathbf{x}}_t^{M_C} = (1 - M_C) \cdot \hat{\mathbf{x}}_t + M_C \cdot \hat{\mathbf{x}}_0,$$

т.е. переменные, на которых мы хотим выполнять обусловливание, остаются чистыми. Затем потери можно определить как:

$$l(\phi, M_C, t, \hat{\mathbf{x}}_0, \hat{\mathbf{x}}_t) = (1 - M_C) \cdot [s_\phi^{M_E}(\hat{\mathbf{x}}^{M_C}, t) - \nabla_{\hat{\mathbf{x}}_t} \ln p_t(\hat{\mathbf{x}}_t | \mathbf{x}_0)],$$

где $s_\phi^{M_E}$ обозначает модель оценки (score model), оснащённую определённой маской внимания M_E . При усреднении по уровням шума t и данным получаем:

$$\mathcal{L}(\phi) = \mathbb{E}_{M_C, t, \hat{\mathbf{x}}_0, \hat{\mathbf{x}}_t} [||l(\phi, M_C, t, \hat{\mathbf{x}}_0, \hat{\mathbf{x}}_t)||_2^2].$$

Мы отмечаем, что для упрощения обозначений M_E здесь остаётся фиксированной, но она может зависеть от состояния условия или входных данных.

После обучения Simformer можно напрямую сэмплировать произвольные условные распределения (рис. 3). Мы берём выборки из распределения шума и запускаем обратный диффузионный процесс для всех ненаблюдаемых переменных, сохраняя наблюдаемые переменные постоянными на их условных значениях. Наличие доступа ко всем условным распределениям также позволяет нам комбинировать оценки (scores) и тем самым выполнять вывод для симуляторов с независимыми одинаково распределёнными (i.i.d.) точками данных. Аналогичным образом мы можем использовать другие преобразования оценок для последующей адаптации к другим конфигурациям априорного распределения или правдоподобия.

Управляемая диффузия (guided diffusion) позволяет сэмплировать из генеративной модели с дополнительным контекстом y и использовалась в таких задачах, как дорисовка

изображений, повышение разрешения и удаление размытия. Она модифицирует процесс обратной диффузии, чтобы согласовать его с заданным контекстом y . Управляемая диффузия изменяет оценённый скор (score) следующим образом:

$$s(\hat{\mathbf{x}}_t, t|y) \approx s_\phi(\hat{\mathbf{x}}_t, t) + \nabla_{\hat{\mathbf{x}}_t} \ln p_t(y|\hat{\mathbf{x}}_t).$$

Были разработаны различные стратегии управления диффузионным процессом, в основном различающиеся тем, как они оценивают $\nabla_{\hat{\mathbf{x}}_t} \ln p_t(y|\hat{\mathbf{x}}_t)$.

Мы используем управление диффузией (diffusion guidance), чтобы позволить Simformer выполнять обусловливание не только на фиксированные наблюдения, но и на интервалы наблюдений (или, аналогично, на интервалы априорного распределения). Диффузионные модели можно направлять с помощью произвольных функций. В соответствии с этим мы используем следующую общую формулировку для управления диффузионным процессом:

$$s_\phi(\hat{\mathbf{x}}_t, t|c) \approx s_\phi(\hat{\mathbf{x}}_t, t) + \nabla_{\hat{\mathbf{x}}_t} \ln \sigma(-s(t)c(\hat{\mathbf{x}}_t))$$

Здесь σ обозначает сигмоидную функцию, $s(t)$ – соответствующая масштабирующая функция, удовлетворяющая условию $s(t) \rightarrow \infty$ при $t \rightarrow 0$, в зависимости от выбора SDE, и c обозначает функцию ограничения $c(\hat{\mathbf{x}}) \leq 0$. Например, чтобы обеспечить верхнюю границу интервала u , мы используем $c(\hat{\mathbf{x}}) = \hat{\mathbf{x}} - u$.

5 Model Misspecification

5.1 Статья [Schmitt2021]

Название: Detecting Model Misspecification in Amortized Bayesian Inference with Neural Networks

Авторы: Marvin Schmitt, Paul-Christian Bürkner, Ullrich Köthe, Stefan T. Radev

Аннотация: Нейросетевые оценки плотности доказали свою исключительную эффективность в выполнении байесовского вывода на основе моделирования в различных областях исследований. В частности, платформа BayesFlow использует двухэтапный подход для обеспечения возможности оценки амортизированных параметров в условиях, когда функция правдоподобия неявно определяется программой моделирования. Но насколько верен такой вывод, если моделирование плохо отражает реальность? В этой статье мы концептуализируем типы ошибок в описании модели, возникающих при выводе на основе имитационного моделирования, и систематически исследуем работоспособность BayesFlow при этих неточностях. Мы предлагаем расширенную задачу оптимизации, которая накладывает вероятностную структуру на пространство скрытых данных и использует так называемую mean discrepancy (MMD) для обнаружения потенциально катастрофических ошибок при выводе, подрывающих достоверность полученных результатов. Мы проверяем наш критерий обнаружения на ряде искусственных и реалистичных ошибочных определений, начиная от игрушечных сопряженных моделей и заканчивая сложными моделями принятия решений и динамики вспышек заболеваний, применяемыми к реальным данным. Далее мы показываем, что ошибки апостериорного вывода увеличиваются в зависимости от расстояния между истинным распределением, генерирующим данные, и типичным набором симуляций в скрытом итоговом пространстве. Таким образом, мы демонстрируем двойную полезность MMD как метода обнаружения ошибок в спецификации модели и как средства проверки достоверности амортизированного байесовского вывода.

5.2 Статья [Cannon2022]

Название: Investigating the Impact of Model Misspecification in Neural Simulation-based Inference

Авторы: Patrick Cannon, Daniel Ward, and Sebastian M. Schmon

Аннотация: *Благодаря успехам в оценке плотности с помощью нейросетей в последние годы был достигнут значительный прогресс в разработке набора методов вероятностного вывода на основе моделирования (SBI), которые позволяют выполнять гибкий, приближенный байесовский вывод для стохастических моделей. Хотя было доказано, что нейросетевые SBI могут обеспечивать точные апостериорные оценки, в исследованиях, позволивших получить эти результаты, рассматривались только точно специфицированные (well-specified) задачи, то есть такие, в которых модель и процесс генерации данных полностью совпадают. Поведению таких алгоритмов в случае некорректной спецификации модели (misspecification) уделялось мало внимания. В этой работе мы впервые всесторонне изучаем поведение нейросетевых алгоритмов SBI при различных формах неточной спецификации модели. Мы обнаружили, что неточная спецификация может оказывать крайне негативное влияние на результат вывода. Были изучены некоторые стратегии смягчения последствий, но ни один из протестированных подходов не позволяет избежать сбоев во всех случаях. Вывод: для решения проблемы некорректной спецификации модели необходимы новые подходы, если мы хотим, чтобы нейронные SBI позволяли делать точные научные выводы.*

5.3 Статья [Hermans2022]

Название: A Trust Crisis In Simulation-Based Inference? Your Posterior Approximations Can Be Unfaithful

Авторы: Joeri Hermans, Arnaud Delaunoy, François Rozet, Antoine Wehenkel, Volodimir Begy, Gilles Louppe

Аннотация: *Мы приводим обширные эмпирические данные, свидетельствующие о том, что современные алгоритмы байесовского вывода, основанные на моделировании, могут давать неверные с вычислительной точки зрения апостериорные оценки. Наши результаты показывают, что все протестированные алгоритмы – (S)NPE, (S)NRE, (S)NLE и варианты ABC – могут приводить к чрезмерно уверенным (overconfident) апостериорным оценкам, что делает их ненадёжными и фальсифицирующими с научной точки зрения. Если не решить эту проблему, сфера применения SBI может значительно сузиться. По этой причине мы считаем, что необходимо направить исследовательские усилия на теоретическую и методологическую разработку консервативных алгоритмов приближённого вероятностного вывода, и предлагаем направления исследований для достижения этой цели. В связи с этим мы приводим эмпирические доказательства того, что ансамблирование постериорных суррогатов обеспечивает более надёжные приближения и решает проблему.*

5.4 Статья [Pierre2025]

Название: Mitigating Model Misspecification in Simulation-Based Inference for Galaxy Clustering

Авторы: Sebastien Pierre, Bruno Regalado-Saint Blancard, ChangHoon Hahn, and Michael Eickenberg

Аннотация: *SBI стал важным инструментом в космологии для извлечения дополнительной информации из данных наблюдений при помощи моделирования. Однако все*

космологические модели являются лишь приближением к реальной Вселенной, и методы SBI могут быть чувствительны к неправильной спецификации модели, особенно когда данные наблюдений не принадлежат обучающему распределению. Представлен метод, который повышает надёжность космологического анализа в таких условиях. Подход сначала обнаруживает и отбрасывает компоненты сводной статистики, демонстрирующие несогласованность в связанных симуляторах, а затем изучает преобразование, которое возвращает наблюдение в область задания обучающего распределения. Мы применяем этот метод в контексте недавнего анализа кластеризации галактик *SimBIG* SBI с использованием сводной статистики *wavelet scattering transform (WST)*. Предложенный метод легко применим в других SBI контекстах в космологии, и представляет собой шаг на пути к созданию более робастных алгоритмов SBI.

5.5 Статья [Kelly2025]

Название: Simulation-based Bayesian inference under model misspecification

Авторы: Ryan P. Kelly, David J. Warne, David T. Frazier, David J. Nott, Michael U. Gutmann, and Christopher Drovandi

Аннотация: Методы байесовского вывода, основанные на моделировании (*Simulation-Based Inference, SBI*), широко применяются для оценки параметров в сложных моделях, где вычисление функции правдоподобия затруднено, но моделирование данных относительно несложно. Однако эти методы обычно предполагают, что имитационная модель точно отражает реальный процесс генерации данных – предположение, которое в практических сценариях часто не выполняется. В данной статье мы сосредоточимся на проблемах, с которыми сталкиваются методы SBI при некорректной спецификации модели (*model misspecification*). Мы систематизируем последние исследования, направленные на смягчение последствий такой некорректности, выделяя три ключевые стратегии: устойчивые (робастные) сводные статистики; обобщенный байесовский вывод; моделирование ошибок и параметры корректировки. Для демонстрации как уязвимостей популярных методов SBI, так и эффективности альтернативных подходов, устойчивых к некорректной спецификации, мы представляем эмпирические результаты на наглядном примере.

6 Simulation-Based Calibration и смежные вопросы

6.1 Статья [Talts2020]

Название: Validating Bayesian Inference Algorithms with Simulation-Based Calibration

Авторы: Sean Talts, Michael Betancourt, Daniel Simpson, Aki Vehtari, Andrew Gelman

Аннотация: Валификация корректности байесовских вычислений является сложной задачей. Это особенно справедливо для сложных моделей, часто встречающихся на практике, поскольку они требуют сложных реализаций модели и алгоритмов. В статье представлена калибровка на основе моделирования (*simulation-based calibration, SBC*) – универсальная процедура для проверки выводов, полученных с помощью байесовских алгоритмов, способных генерировать выборки из апостериорного распределения. Эта процедура не только выявляет неточности вычислений и несоответствия в реализациях моделей, но также предоставляет графический вывод, который может указывать на характер возникающих проблем. SBC является важнейшей частью надёжного байесо-

вого рабочего процесса, а также полезным инструментом для разработчиков вычислительных алгоритмов и статистического программного обеспечения.

7 Байесовское принятие решений

7.1 Статья [Gorecki2023]

Название: Amortized Bayesian Decision Making for simulation-based models

Авторы: Mila Gorecki, Jakob H. Macke, Michael Deistler

Аннотация: *SBI* представляет собой мощный инструмент для определения апостериорных распределений стохастических симуляторов в самых разных областях. Однако во многих случаях апостериорное распределение не является конечной целью само по себе – вместо этого полученные значения параметров и их неопределенности используются как основа для принятия решений о дальнейших действиях. К сожалению, поскольку постериоры *SBI* являются (потенциально грубыми) приближениями истинного апостериорного распределения, принимаемые на их основе решения могут оказаться неоптимальными. В данной работе мы рассматриваем вопрос о том, как проводить байесовское принятие решений (*Bayesian decision taking*) для стохастических симуляторов и как можно обойти необходимость вычисления явного приближения постериора. Наш метод обучает нейросеть на синтетических данных и позволяет предсказывать ожидаемую стоимость (*expected cost*) для любых данных и действий, что дает возможность напрямую определять оптимальное действие с минимальными затратами. Мы применяем наш метод к нескольким тестовым задачам и показываем, что он обеспечивает сопоставимую стоимость решений с истинным апостериорным распределением. Затем мы демонстрируем его работу на реальном симуляторе в медицинской нейронауке — *Bayesian Virtual Epileptic Patient* — и подтверждаем, что метод позволяет находить действия с низкой стоимостью уже после небольшого числа симуляций.

7.2 Статья [Alsing2023]

Название: Optimal simulation-based Bayesian decisions

Авторы: Justin Alsing, Thomas D. P. Edwards, Benjamin Wandelt

Аннотация: Мы предлагаем методологию для эффективного вычисления оптимальных байесовских решений в условиях, когда функция правдоподобия аналитически неразрешима. Наш подход основан на обучении суррогатной модели, аппроксимирующей ожидаемую полезность (или её распределение) как функцию пространств действий и данных. Используя последние достижения в области *SBI* и байесовской оптимизации, мы разработали схемы активного обучения для выбора оптимальных точек в пространствах параметров и действий для проведения симуляций. Это позволяет определять оптимальное действие, требуя минимального количества вычислительных экспериментов. Предлагаемая методология демонстрирует исключительную вычислительную эффективность, так как требует значительно меньше вызовов модели, чем стандартные задачи апостериорного вывода, и превышает по эффективности МС-методы подходе. Наша разработка открывает новые возможности для байесовского принятия решений, особенно в сложных случаях: когда функция правдоподобия не имеет аналитического представления и когда вычислительная стоимость симуляций существенно ограничивает исследование.

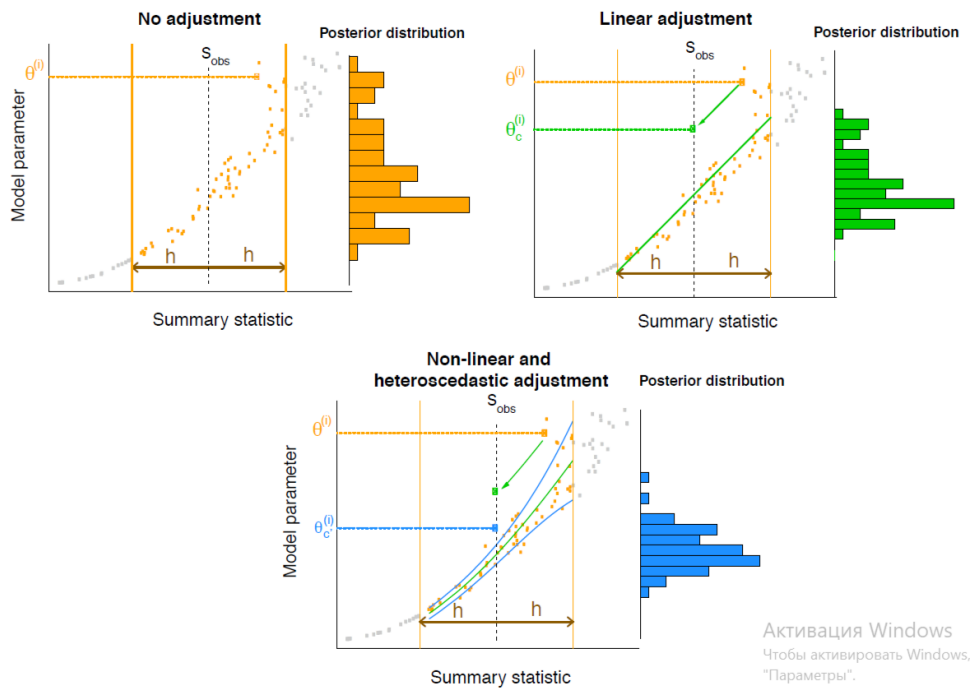


Рис. 7: Схематическое изображение регрессионной корректировки.

8 Заметки на полях...

Вопрос 1: Почему сэмплирование эффективно?

Обычно ссылаются на закон больших чисел (ЗБЧ) в формулировке:

если сэмплы $X_k \sim p(x)$, $k = 1, \dots, N$, то при $N \rightarrow \infty$ $\frac{1}{N} \sum_{k=1}^N \delta_{X_k}(x) \rightarrow p(x)$

(здесь и далее $\delta_Z(z)$ – дираковская мера).

Речь идет о так называемом *Равномерном ЗБЧ*, обобщающем ЗБЧ на среднее от некоторой функции $f(x, \theta)$, непрерывной по переменной $\theta \in \Theta$. В нашем случае $f(x, \theta) \equiv \delta_x(\theta)$ и не совсем очевидна ее непрерывность по θ . Кроме этого, в этой теореме требуется: 1) компактность Θ , 2) непрерывность $f(x, \theta)$ при каждом $\theta \in \Theta$ для почти всех x , 3) существование доминирующей функции $d(x)$ такой, что $d(x) \geq \|f(x, \theta)\|$ для всех $\theta \in \Theta$.

8.1 Про ABC

ABC – это аппроксимация постериора. И как всякая аппроксимация, она допускает неточность – аппроксимационную ошибку. Размер этой ошибки зависит от порога ϵ , метрики в пространстве данных и размера выборки. Если же от многомерных данных мы переходим к сводным статисткам (меньшей размерности), то часть информации мы теряем (если статистика не является достаточной). А значит, получаем еще ошибку. Но введение этой ошибки, позволяет уменьшить ошибку первого вида (аппроксимационную), например, понизив значение порога.

В рамках ABC отбор по близости в пространстве данных (или их статистик) приводит к искусственному уширению распределения параметров модели. Имеет смысл провести дополнительно «сужение». Такой подход получил название *регрессионной корректировки* (regression adjustment), так как осуществляется посредством регрессии (линейной или нелинейной) зависимости параметров от данных. Схематически корректировка показана на рис. 7.

А Полезные примеры

Первые шесть примеров взяты из статьи [Ho2024] (см. также раздел 4.3), седьмой – из [Macias2025].

Примеры из нейронауки взяты из статьи [Goncalves2020]

А.1 Оценка массы скоплений галактик по рентгеновским изображениям

Используя настраиваемые embedding-сети, мы обрабатываем данные изображений, чтобы оценить массу скоплений галактик по рентгеновским наблюдениям. Наша задача – оценить параметр M_{500c} скопления галактик по изображениям размером 128×128 пикселей, представляющим проекцию на небесную сферу рентгеновских фотонных отсчётов.

Примечание. В данной работе масса гало определяется как масса сферического избытка при контрасте плотности, в 500 раз превышающем критическую плотность Вселенной и обозначается как M_{500c} . Она выражается в космологически инвариантных единицах $h^{-1}M_{\odot}$. Здесь $h^{-1}\text{Mpc}$ – мегапарсек, скорректированный на постоянную Хаббла, h – безразмерный параметр Хаббла, определяемый как $H_0 = 100h \text{ km}/(\text{s}\cdot\text{Mpc})$, где H_0 – постоянная Хаббла, согласно последним данным $H_0 \approx 67 - 70 \text{ km}/(\text{s}\cdot\text{Mpc})$.

Для обучения и тестирования нашей модели мы используем набор данных из 3 285 модельных рентгеновских наблюдений скоплений, полученных из гидродинамического моделирования Magneticum, разработанного для имитации наблюдений телескопа eROSITA, см. рис. 8 слева. Эти модельные наблюдения представляют собой однополосные изображения рентгеновского фотонного излучения от горячей среды внутри скопления (ICM, intra-cluster medium), и включают реалистичное моделирование источников систематических неопределенностей, присущих реальным рентгеновским измерениям, таких как: морфология скопления, фоновое излучение, отклик телескопа и источники активных галактических ядер (AGN). Хороший метод оценки массы скопления должен уметь отделять полезный рентгеновский сигнал от шума и периферийного излучения, а также понимать физическую связь между рентгеновским излучением, содержанием газа в ICM и массой системы.

Мы выполняем NPE, применяя embedding-сеть со свёрточной архитектурой, аналогичной использованной в [Ho2023] (см. также работу [Ho2021]), т.е. вместо полносвязного слоя, отображающего данные в точечную оценку, мы передаём финальное вложение (embedding) на вход NDE. Создаём ансамбль из четырёх моделей, каждая из которых имеет одинаковую архитектуру embedding-сети, но две из них используют NDE в виде MAF, а две другие MDN. Затем мы обучаем модели с нуля, используя функции потерь для NDE (а не MSE как в [Ho2023]). Априорное распределение для массы скопления предполагается равномерным. Модельный каталог разделен на 90% обучающих данных и 10% тестовых данных для оценки работоспособности алгоритма. Вся процедура обучения и тестирования занимает около 15 минут на GPU Nvidia V100.

Сравнение истинных и предсказанных значений массы для этой модели показано на рис. 8 справа. Используя исходную архитектуру, мы достигаем разброса (scatter) предсказаний на тестовой выборке 0.0782 dex^2 . По сравнению с разбросом 0.0773 dex для однополосных рентгеновских изображений в [Ho2023], мы достигаем очень схожего уровня извлечения информации, хотя наша неопределённость несколько выше, чем в оригинальной

²Dex – логарифмическая единица, показывающая, на сколько порядков (степеней 10) различаются величины.

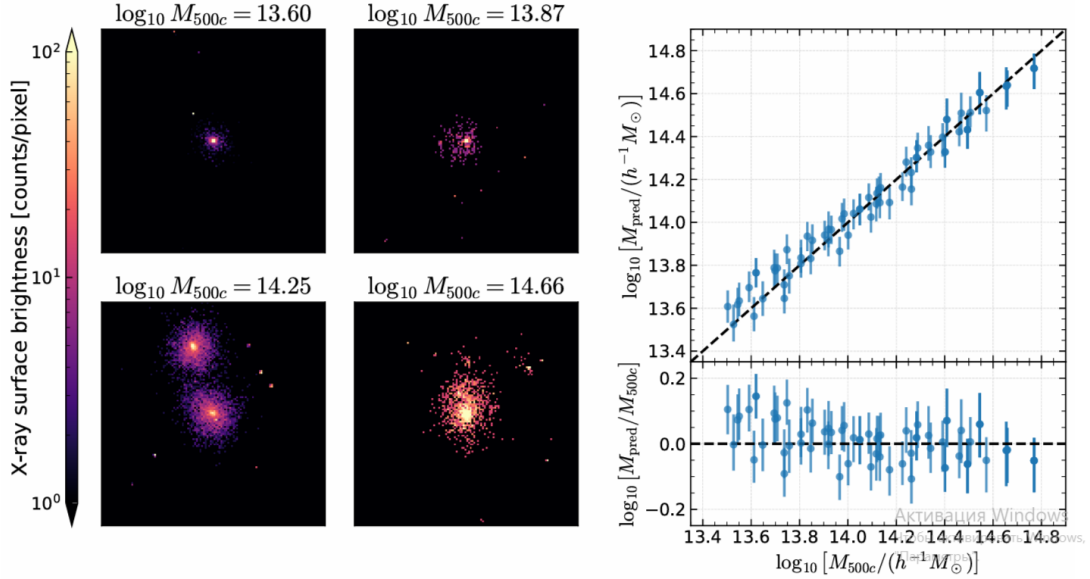


Рис. 8: Реконструкция параметра M_{500c} скопления галактик по модельным рентгеновским наблюдениям, аналогичным данным eROSITA. Слева: 4 случайных примера одноканальных рентгеновских изображений, представленных в виде яркости поверхностной проекции на небесную сферу. Справа: график сравнения истинных и предсказанных значений логарифма массы на тестовой выборке (медиана и центральный 68% доверительный интервал нейросетевых постериоров).

работе. Мы связываем это с тем, что ансамблирование моделей несколько увеличивает прогностическую неопределённость. Однако мы отмечаем, что незначительно увеличив количество свёрточных фильтров в первых двух слоях, мы наблюдали средний разброс вплоть до 0.071 dex, что свидетельствует о том, что методы NPE способны обеспечивать результаты на уровне современных стандартов для физических задач при использовании подходящей архитектуры.

A.2 Оценка параметров темной материи по спектру мощности

Распространённым бенчмарком для вероятностного вывода в космологии является оценка космологических параметров по спектру мощности материи. В этой задаче мы запускаем симуляции эволюции материи во Вселенной для различных космологических моделей, измеряем сводные статистики наблюдений, такие как спектр мощности материи, а затем пытаемся связать эти наблюдения с ограничениями на космологические параметры.

Используя спектр мощности каталога гало из симуляции Quijote, вычислим постериоры трёх космологических параметров θ : 1) полной плотности материи Ω_m , 2) амплитуды флуктуаций материи σ_8 и 3) безразмерной постоянной Хаббла h (измеряющей скорость расширения Вселенной). В качестве наблюдений x используем мультиполи спектра мощности для $l = 0, 1, 2$ в 23 линейно расположенных бинах волнового числа k в диапазоне от $0.08 h^{-1} \text{Mpc}$ до $2.8 h^{-1} \text{Mpc}$.

Мы обучаем модель sbi-NLE, применяя ансамбль из шести NDE. Каждый NDE использует облегчённую архитектуру MAF с 10 скрытыми слоями и 3 преобразованиями. Для обучения используем 1800 симуляций из Quijote, оставляя оставшиеся 200 для тестирования её производительности. Затем мы используем вариационный вывод (VI sampling), чтобы получить апостериорное распределение параметров из этого обученного ансамбля

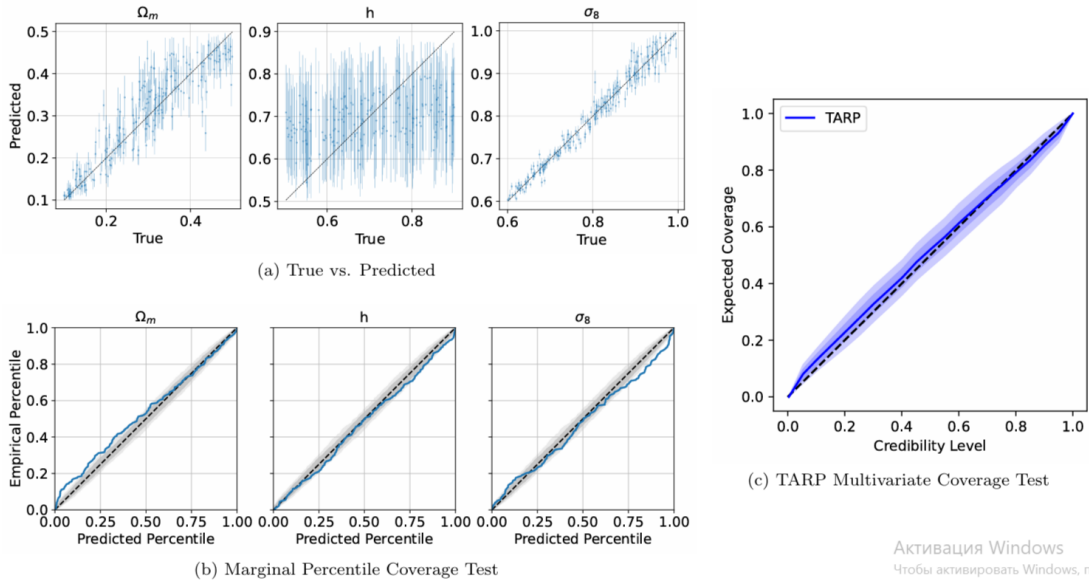


Рис. 9: Вывод трёх космологических параметров Ω_m , h и σ_8 по мультиполям спектра мощности гало в симуляциях Quijote. (a): сравнение истинных и предсказанных значений параметров в тестовых симуляциях (показаны маргинальные центральные 68% доверительные интервалы). (b): тест покрытия с использованием P-P графиков, (c): тест покрытия TARP.

правдоподобий, взвешенного по их потерям на валидации. Это значительно ускоряет процесс сэмплирования на полном тестовом наборе по сравнению с традиционным MCMC-сэмплированием.

Сравнение истинных и предсказанных значений трёх параметров показано на верхних панелях рис. 9a и обнаруживает хорошую прогностическую способность для Ω_m и σ_8 . Отсутствие предсказуемости для h ожидаемо с физической точки зрения, поскольку одни лишь мультиполи спектра мощности значительно менее чувствительны к скорости расширения без какой-либо внешней априорной информации.

Мы также проводим тесты покрытия (coverage tests) для постериоров, получая равномерно распределённые гистограммы рангов, см. нижние панели рис. 9b, а также тест покрытия с использованием TARP, рис. 9c, демонстрирующий, что апостериорное распределение корректно откалибровано.

А.3 Оценка параметров темной материи по полевым данным

Чтобы продемонстрировать способность LtU-III обрабатывать сложные типы данных, решим эту же задачу космологического вывода по набору данных в виде облака точек. Вместо сводных статистик спектра мощности будем использовать дискретные каталоги из 10 000 наиболее массивных гало тёмной материи из каждой симуляции Quijote. Ранее было показано, что информация высокого разрешения, содержащаяся в дискретных положениях гало, позволяет ограничивать космологические параметры с чрезвычайно высокой точностью.

Мы передаём этот каталог в нейросетевые архитектуры, используя графовую нейронную сеть с передачей сообщений (message-passing GNN), которая рассматривает гало как узлы графа и фильтрует полевою информацию на уровне графа в высокоинформативные нейронные представления. Затем эта графовая информация подаётся на вход NDE, построенной на основе нормализующих потоков, для выполнения NPE в lampe.

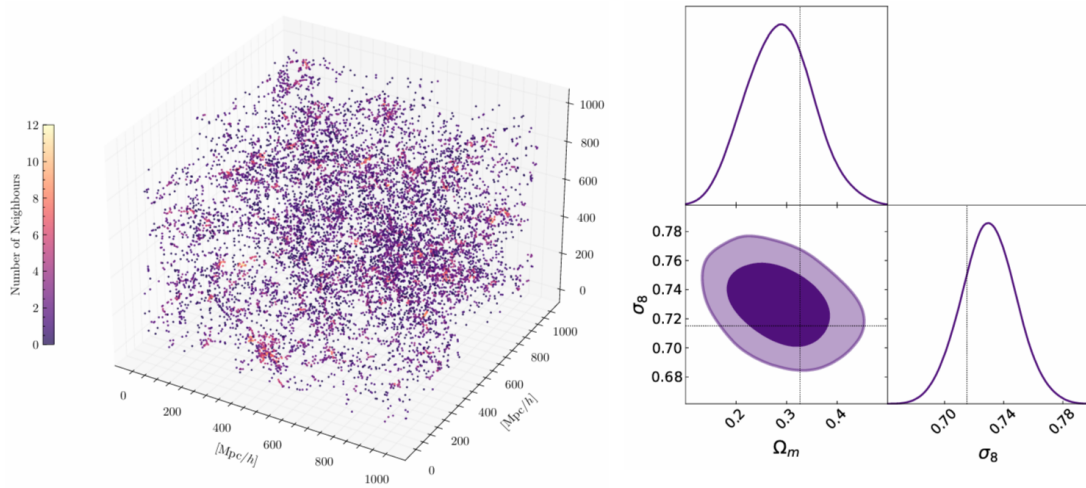


Рис. 10: Оценка космологических параметров по облакам точек гало тёмной материи. Слева: Пример облака точек из симуляций Quijote. Графовая нейронная сеть принимает на вход расстояния между гало, разделёнными менее чем на $20 h^{-1}\text{Mpc}$. Справа: Апостериорное распределение для параметров Ω_m и σ_8 на тестовой симуляции (контуры – центральные 68% и 95% доверительные интервалы), истинные значения показаны пунктиром.

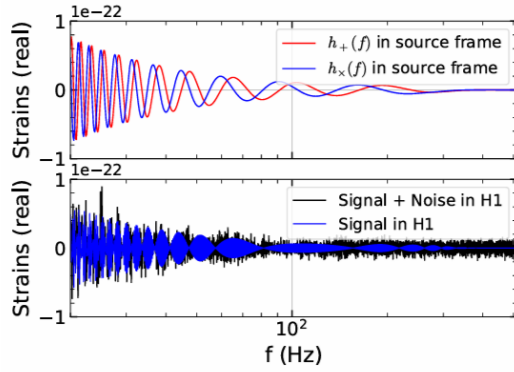
Мы обучаем эту графмодель NPE для восстановления космологических параметров Ω_m и σ_8 по положениям самых массивных гало тёмной материи в Quijote. Графовая нейронная сеть представляет собой нейронную сеть, в которой нейронные сети узлов и рёбер состоят из двух многослойных перцептронов, каждый из которых имеет 3 слоя по 128 скрытых нейронов. Сеть принимает на вход 3D-расстояния между гало и их соответствующие модули (modulus). Близлежащие гало соединяются в графе, если их расстояние (в сопутствующей системе отсчета) меньше $20 h^{-1}\text{Mpc}$. Для NDE мы обучаем ансамбль из четырёх сетей, каждая из которых использует архитектуру MAF с 5 преобразованиями и 50 скрытыми признаками.

На рис. 10 приведен пример облака точек вместе с результатами вывода, полученными с помощью графовой нейронной сети. Графовая нейронная сеть даёт информативные апостериорные распределения, согласующиеся с истинными значениями. Отмечаем, что получаемые ограничения не столь жёсткие, как те, что были получены для спектра мощности (ср. рис. 9). Вероятно, это связано с тем, что граф использует только самые массивные гало, то есть более реалистичный с физической точки зрения зонд, в то время как данные по спектру мощности используют полное, высокоразрешающее распределение тёмной материи.

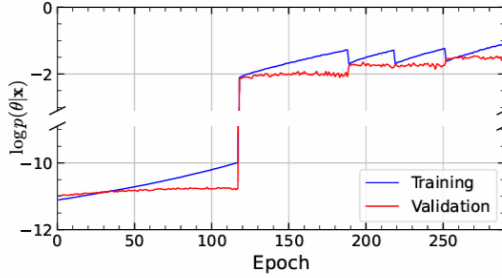
A.4 Гравитационные волны в результате слияния черных дыр

За последние годы несколько исследований, основанных на методах ILI (Implicit Likelihood Inference), рассматривали задачу восстановления параметров событий слияния черных дыр по сигналам гравитационных волн (GW).

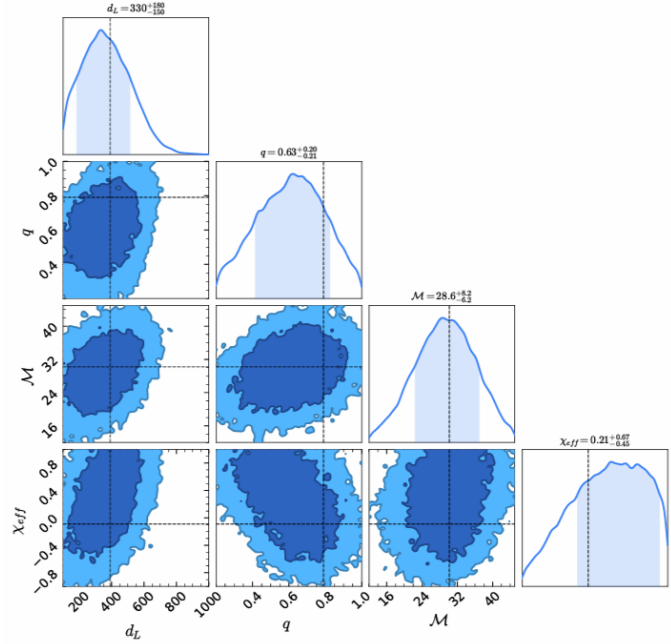
Здесь мы исследуем применение многораундового байесовского вывода для ограничения сокращённого набора параметров GW. Наш модельный вектор данных строится путём объединения смоделированных частотных сигналов отклика («деформации», strain) с каждого из детекторов Хэнфорд (H1) и Ливингстон (L1). Для моделирования используем



(a) Example of GW signal in source and detector frame.



(b) Multi-round training and validation posterior probability.



(c) Posterior corner plot for SNPE inference.

Активация Win
Чтобы активировать
"Параметры"

Рис. 11: Восстановление параметров события по наблюдениям гравитационно-волновых детекторов Хэнфорд (H1) и Ливингстон. (a) Пример смоделированного GW-сигнала: поляризации излучения h_+ и h_x в системе источника (сверху) и вещественная часть отклика («деформации») и реализация шума в системе детектора Хэнфорд (H1). (b) Эволюция логарифма вероятности при многораундовом выводе с использованием SNPE для $n_r = 5$ раундов. (c) Результирующий постериор для одного запуска многораундового SNPE (показаны контуры 68% и 95% доверительных интервалов, пунктир – эталонные значения параметров).

симулятор IMRPhenomPv2 [Khan2016] с частотным разрешением $\Delta f = 0.125$ Гц, что даёт входной вектор данных размерностью $\dim(x) = 2 \times 7872$, см. рис. 11а. Подчёркнем, что в отличие от методов сэмплирования, основанных на правдоподобии (реализованных в Bilby), которым для вычисления правдоподобия требуется только сигнальная компонента модельных данных, наш подход ILI требует включения реализаций шума в обучающий вектор данных.

Мы демонстрируем упрощённое применение LtU-ILI для SNPE на сигналах гравитационных волн, фиксируя параметры, связанные со временем и геометрией детектирования, и фокусируясь на наборе из четырёх параметров θ (m_1, m_2 – массы сливающихся объектов): 1) приведённой массе (chirp mass) $M \equiv (m_1 m_2)^{3/5} / (m_1 + m_2)^{1/5}$, 2) отношении масс (mass ratio) $q \equiv m_2 / m_1 \leq 1$, 3) эффективного выровненного спина (effective aligned spin) χ_{eff} и 4) расстояние по светимости (luminosity distance) d_L . Эти параметры выбраны из-за их сниженной корреляции и важности в оценке параметров GW.

Мы установили базовые значения параметров для нашего имитационного события \mathbf{x}_{obs} : $d_L = 390$ Мпс, $q = 0.79$, $M = 30.2 M_{\odot}$, $\chi_{\text{eff}} = -0.09$, соответствующих событию GW150914 из [Abbott2016]. Зададим прайоры равномерными в диапазонах от $[100.0, 0.2, 12.0, -1.0]$ до $[1000.0, 1.0, 45.0, 1.0]$. Для оценки плотности мы используем бэкенд sbi с MAF и одномерной свёрточной embedding-сетью, проводя пять раундов обучения с 1000 симуляций на раунд.

Рис. 11b иллюстрирует прогрессию кумулятивной апостериорной вероятности в зависимости от времени обучения. Заметное увеличение плотности вероятности как для обучающих, так и для валидационных наборов данных связано с тем, что модель фокусируется на изучении апостериорной плотности вблизи целевого наблюдения $P(\boldsymbol{\theta}|\mathbf{x} = \mathbf{x}_{\text{obs}})$. Однако ранний рост потерь на валидации в последующих раундах указывает на потенциальное переобучение, демонстрируя ограничения модели в извлечении дополнительной информации из входных симуляций. Сэмплирование из постериора при $\mathbf{x} = \mathbf{x}_{\text{obs}}$, рис. 11c, показывает, что мы способны ограничить каждый из четырёх параметров вокруг эталонной точки. Для этого конкретного примера использование LtU-ILI позволяет нам параметризовать симулятор (аппроксиматор профиля волны, количество учитываемых детекторов, геометрия детектирования, длительность сигнала,...), архитектуры для обучения и метрики для вычислений с помощью единственного набора конфигурационных файлов. Это упрощает организацию и воспроизведение унифицированных тестов байесовского вывода для этой особенно сложной задачи.

A.5 Фотометрическое исследование межзвёздной пыли

В этом приложении мы исследуем, как различные наблюдаемые величины могут быть использованы для ограничения различных вырожденностей в моделях галактической пыли, и как комбинации наблюдений могут быть использованы для максимального извлечения информации в рамках ILI. Пыль составляет около 1% межзвёздной среды, но переизлучает примерно 30% звёздного излучения. Таким образом, понимание свойств пыли и её влияния на наблюдаемые величины является ключевой целью галактической и внегалактической астрономии и фундаментальной неопределённостью в последующих космологических исследованиях. Пыль вызывает общее покраснение излучения, параметризуемое оптической глубиной и законом ослабления.

Сначала кратко опишем прямую модель для звёздного излучения и ослабления пылью. Мы сгенерировали спектры для всех галактик со звёздной массой $> 10^{10} M_{\odot}$ в симуляторе Simba, используя Synthesizer. Интегрированное собственное (до ослабления) звёздное излучение каждой галактики было получено путём сопоставления каждой звёздной частицы с моделями звёздного населения BC03 на основе её возраста и металличности. Затем мы смоделировали ослабление пылью как степенной закон, зависящий от длины волны, с наклоном α ,

$$T(\lambda, t) = \exp \{ \tau(t) (\lambda / \lambda_V)^{-\alpha} \},$$

где λ_V – длина волны V-диапазона (550 nm). Здесь оптическая глубина τ зависит от возраста звёздного населения t : предполагается, что звёздные частицы моложе 10 млн лет (Myr) всё ещё находятся в своих родительских облаках (BC, birth clouds) и, следовательно, испытывают дополнительный источник ослабления, $\tau(t) = \tau_{\text{BC}} + \tau_{\text{ISM}}$ при $t \leq 10$ Myr и $\tau(t) = \tau_{\text{ISM}}$ при $t > 10$ Myr.

Предполагая ту же базовую модель звёздного излучения, мы сгенерировали оптическую фотометрию в системе покоя в полосах g и r для всех галактик в Simba для 1000 симуляций, используя латинский гиперкуб, с равномерными априорными распределениями параметров в диапазонах: $\alpha \in [0.5, 2.0]$, $\tau_{\text{ISM}} \in [0.01, 0.5]$ и $\tau_{\text{BC}} \in [0.3, 1.5]$. Затем мы берём эти симуляции и измеряем функцию светимости в r-диапазоне и распределение цвета g-r, используя их в качестве наших сводных статистик. Мы выполняем NPE, используя ансамбль из MAF с 50 скрытыми признаками и 5 нейросетевыми преобразованиями и MDN с 50 скрытыми признаками и 5 компонентами смеси.

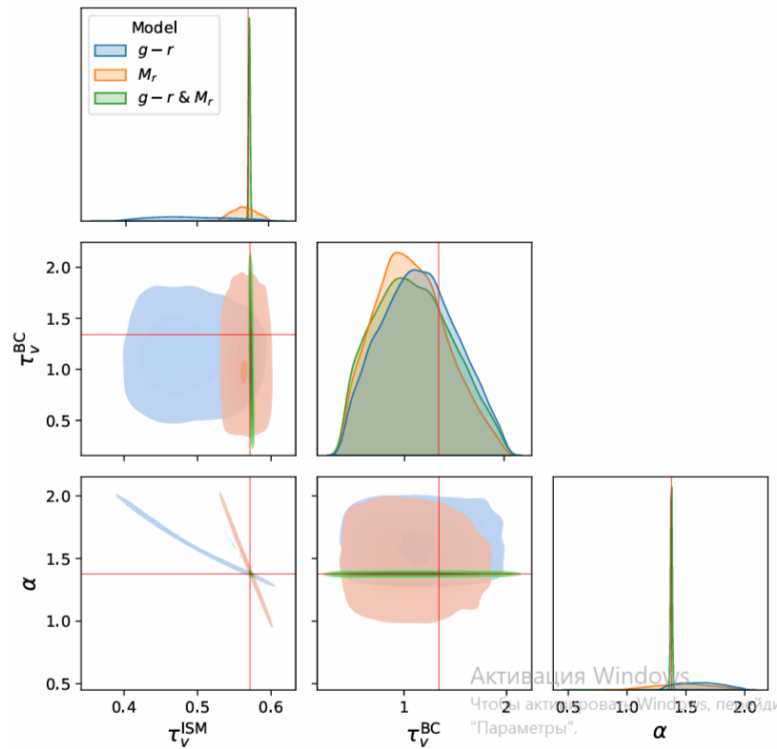


Рис. 12: Восстановление параметров пыли на моделировании Simba. Примеры апостериорных распределений для оптической глубины межзвёздной среды и родительского облака, а также наклона закона ослабления, полученные на основе: распределения цветов $g-r$ (синий), функции светимости по звёздной величине в g -диапазоне (оранжевый), комбинации обоих показателей (зелёный).

На рис. 12 показаны апостериорные распределения для одной тестовой пары «параметры-данные». Мы демонстрируем результаты обучения только на функции светимости, только на цвете и на их комбинации. Очевидно, что общая нормировка функции светимости ограничивает оптическую глубину межзвёздной среды τ_{ISM} , тогда как цвет ограничивает наклон закона ослабления α . При совместном использовании эти параметры одновременно ограничиваются значительно жёстче. Ослабление в родительском облаке оказывает более тонкое влияние на функцию светимости и распределение цвета, но их комбинация приводит к более жёстким ограничениям. LtU-III значительно упрощает настройку конвейеров вывода, использованных здесь, для быстрого тестирования и сравнения комбинаций источников данных.

А.6 Галактический ветер: массовая и энергетическая нагрузка

Здесь мы продемонстрируем, как использование пользовательских embedding-сетей в LtU-III помогает восстановить апостериорное распределения для задач с низким отношением сигнал/шум, таких как реконструкция параметров полуаналитических моделей формирования галактик. Один из таких примеров – изучение слабо ограничивающей роли обратной связи (feedback) в регуляции звездообразования и формировании взаимосвязи «звёздная масса – масса гало».

Недавно было обнаружено, что галактические ветры, вызванные сверхновыми, могут нагревать и вызывать турбулентность в окологалактической среде карликовых галактик и гало массой с порядка массы Млечного Пути. Это может подавлять охлаждение и аккре-

цию газа на галактики, тем самым способствуя регуляции звездообразования. Сила этой «превентивной обратной связи» контролируется двумя параметрами ветра: 1) коэффициентом массовой нагрузки (mass loading factor) η_M и 2) коэффициентом энергетической нагрузки (energy loading factor) η_E . Вместе они описывают удельную энергию ветров и суммируют долю массы и энергии, произведённых сверхновыми, которые покидают галактику и становятся доступными для крупномасштабного нагрева.

Здесь мы хотим исследовать связь между η_M , η_E и соотношением «звёздная масса – масса гало», предсказываемым многочисленными реализациями полуаналитической модели нового поколения sapphire. Используемая здесь модель пренебрегает турбулентностью и для простоты предполагает чисто термический нагрев околוגалактической среды (CGM). Мы предполагаем, что η_M и η_E следуют степенному закону в зависимости от вириальной скорости гало V_{vir} (показателя массы гало):

$$\eta_M = A_M (V_{\text{vir}}/125\text{km/s})^{\alpha_M}, \quad \eta_E = A_E (V_{\text{vir}}/125\text{km/s})^{\alpha_E},$$

Здесь A_M , α_M , A_E и α_E – гиперпараметры, управляющие этими степенными законами, и мы игнорировали любую зависимость от красного смещения. Мы генерируем 5 000 реализаций модели с помощью латинского гиперкуба, равномерно выбирая комбинации параметров в диапазонах: $A_M \in [0.01, 100]$, $\alpha_M \in [-2, 2]$, $A_E \in [0.01, 1]$ и $\alpha_E \in [-2, 2]$ (остальные параметры модели зафиксированы на разумных значениях в соответствии с [Pandya2023]). С этими параметрами система обыкновенных дифференциальных уравнений, описывающая модель, интегрируется с использованием историй аккреции массы для 120 гало, случайно выбранных из пяти подобъёмов симуляции TNG100. Мы считаем, что это достаточное количество гало, чтобы уловить вариации в соотношении «звёздная масса – масса гало» в зависимости от параметров модели. Мы ограничиваемся гало с $\ln M_{\text{vir}}/M_{\odot} = 10 - 12.4$ при $z = 0$, охватывая диапазон карликовых галактик и Млечного Пути, где, как считается, доминирует обратная связь от сверхновых.

В качестве embedding-сети мы используем архитектуру fishnets от [Makinen2023], чтобы извлечь информацию, которую связь «звёздная масса – масса гало» содержит о параметрах $\theta = (A_M, \alpha_M, A_E, \alpha_E)$. Выходные данные каждой реализации sapphire представляют собой набор из $n_x = 120$ независимо эволюционировавших галактик, т.е. $\mathbf{x} = \{(M_{\text{vir}}, M_*, M_{\text{vir}}/M_*)\}$, $i = 1, \dots, n_x$, где M_{vir} и M_* – масса гало и звёздная масса при $z = 0$ соответственно. Функция правдоподобия здесь является произведением правдоподобий для каждой отдельной галактики: $L(\{x_i\}|\theta) = \prod_{i=0}^{n_x} L(x_i|\theta)$, что требует агрегации по разнородным распределениям данных.

Следуя формализму оптимальных fishnets, мы встраиваем данные в нейронные score embeddings (вложения оценок) и веса Фишера, каждый из которых параметризуется полносвязными сетями размером [128, 128, 128] с активациями LeakyReLU, прежде чем передать взвешенные оценки в две NPE-сети: Neural Spline Flow и Gaussianization Flow.

На рис. 13 приведены «истинный параметр против предсказанного апостериорного распределения»: как и ожидалось, коэффициенты энергетической нагрузки галактик чувствительны к свойствам массы гало и звёздной массы, тогда как коэффициенты массовой нагрузки оказывается сложнее ограничить. Это согласуется с результатами [Carr2023], которые показали, что связь «звёздная масса – масса гало» в значительной степени нечувствительно к вариациям η_M , но очень отзывчиво к изменениям η_E . Физическая причина этого заключается в том, что увеличение η_M приводит к более высокой плотности CGM и усиленному охлаждению газа, который возвращается в галактику, не подавляя звездообразование, что в итоге приводит к схожему соотношению $M_* - M_{\text{vir}}$. Напротив, увеличение η_E может нагревать CGM, препятствовать аккреции газа, снижать звездообразование

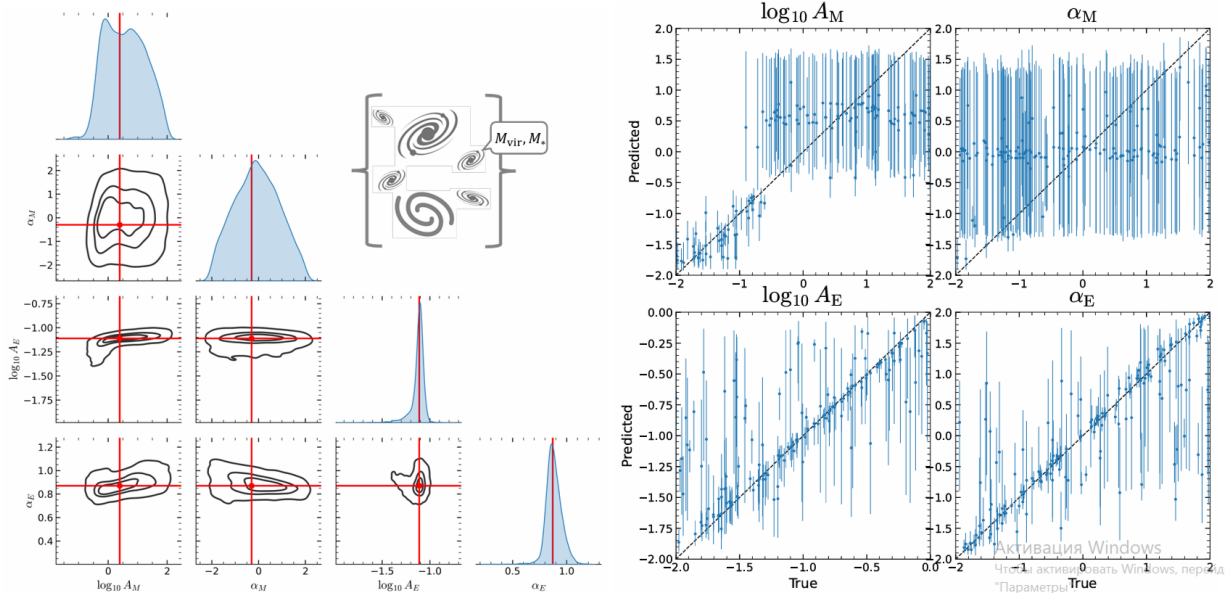


Рис. 13: Реконструкция параметров массовой и энергетической нагрузки из полуаналитической модели `sarphire` по наборам значений вириальной и звёздной массы галактик. Слева: контуры примера выведенного апостериорного распределения. Справа: сравнение истинных значений и предсказаний по всему тестовому набору.

и изменять нормировку и форму связи $M_* - M_{\text{vir}}$. Интересно, что для реализаций `sarphire` с $A_M \leq 0.25$ существует некоторая информация для ограничения параметров массовой нагрузки, которая недоступна при высоких значениях A_M . `LtU-ILI` способен уловить этот эффект и точно оценить доверительные интервалы апостериорного распределения, демонстрируя свою способность калибровать вывод как в режимах с низким, так и с высоким отношением сигнал/шум. В будущем будет интересно использовать `fishnets`, чтобы понять, какие дополнительные свойства галактик, выдаваемые `sarphire`, содержат информацию, необходимую для лучшего изучения и η_M , и η_E .

A.7 Реконструкция направления прилета UHECR по радиосигналам

Название: [\[Macias2025\] Simulation-Based Inference for Direction Reconstruction of Ultra-High-Energy Cosmic Rays with Radio Arrays](#)

Авторы: Oscar Macias, Zachary Mason, Matthew Ho, Arsène Ferrière, Aurélien Benoit-Lévy, and Matías Tüeros

Аннотация: *Обсерватории, регистрирующие космические лучи ультравысоких энергий (КЛУВЭ), требуют несмещенного восстановления направления прихода частиц для обеспечения многоканальной астрономии с использованием радио-импульсов наносекундной длительности. Традиционные методы, основанные на явных функциях правдоподобия, часто опираются на упрощённые модели, что может приводить к смещённым результатам и недооценке неопределённостей. Мы представляем конвейер байесовского вывода, основанного на моделировании (SBI), который в рамках методологии `LtU-ILI` сочетает в себе физически-информированную графовую нейронную сеть (GNN) и апостериорное распределение, параметризованное нормализующим потоком. Каждое событие инициируется аналитической аппроксимацией плоским волновым фронтом; затем*

*GNN уточняет эту оценку, обучаясь пространственно-временным корреляциям между сигналами антенн. Полученное фиксированное векторное представление (embedding) используется для условной параметризации восьмиблочного авторегрессионного потока, который возвращает полное байесовское апостериорное распределение. Модель обучена на 8 000 реалистичных моделирований ШАЛ от КЛУВЭ, сгенерированных с помощью программы ZHAireS. Постериоры проходят *SSh: температурную калибровку* для соответствия эмпирическим критериям достоверности (coverage). На тестовых событиях медианное угловое разрешение составляет менее одного градуса, а номинальные 68%-ные контуры наивысшей апостериорной плотности покрывают $71\% \pm 2\%$ истинных направлений прихода, что указывает на умеренно консервативную калибровку неопределённостей. Этот подход обеспечивает физически интерпретируемые реконструкции, хорошо калиброванные неопределённости и быстрый вероятностный вывод, что делает его идеально подходящим для предстоящих экспериментов, нацеленных на регистрацию событий с большими зенитными углами, таких как GRAND, AugerPrime Radio, IceCube-Gen2, RNO-G и BEACON.*

A.8 Примеры из нейронауки

Изначально эти примеры появились в статье [Lueckmann2017] [Flexible statistical inference for mechanistic models of neural dynamics](#).

Название: [Gonçalves2020] [Training deep neural density estimators to identify mechanistic models of neural dynamics](#)

Авторы: Pedro J Gonçalves, Jan-Matthis Lueckmann, Michael Deistler, Marcel Nonnenmacher, et al.

Аннотация: Механистическое моделирование в нейронауке ставит своей целью объяснение наблюдаемых явлений через лежащие в их основе причины. Однако определение того, какие параметры модели согласуются со сложными и стохастическими нейронными данными, представляет собой серьёзную проблему. Мы решаем эту задачу с помощью ML-инструмента, использующего глубокие нейросетевые оценщики плотности (обученные на симуляциях модели) для проведения байесовского вывода и восстановления всего пространства параметров, совместимых с исходными данными или выбранными характеристиками данных. Метод масштабируем по количеству параметров и характеристикам данных и после первоначального обучения может быстро анализировать новые данные. Мощность и гибкость подхода продемонстрирована на примере моделей рецептивных полей, ионных каналов и модели Ходжкина-Хаксли. Мы также характеризуем пространство конфигураций нейронных цепей, порождающих ритмическую активность в стоматогастрическом ганглии ракообразных, и используем эти результаты для формулирования гипотез о лежащих в основе компенсаторных механизмов. Этот подход поможет сократить разрыв между моделями нейродинамики, ориентированными на данные и ориентированными на теорию.

A.8.1 Модель рецептивных полей

A.8.2 Модель ионных каналов

A.8.3 Модель Ходжкина-Хаксли