ASTRODAT
AstroSTatistics and Research-Oriented DATa analysis

ASTRODAT Abstract Booklet

Abstracts of the Invited Seminars

Title: Version management with git for collaborative coding

Speaker: Katherine Harborne

In this session, I will begin by getting you familiar with git for managing the collaborative packages that you're working on throughout the week. I'll provide an overview of the common git operations, advice about branch protections and some suggestions for building an environment that's easy to collaborate within. This session will be interactive, as we go about setting up repositories for the week. Materials will be provided to participants following the session for reference throughout the week.

Title: Scientific software development

Speaker: Samuel Farrens

In this presentation I will highlight some key considerations and good practices for software development, with a particular focus on tools designed for scientific applications. The main objective is to introduce concepts that every developer should keep in mind at all times—principles that remain constant even as technologies evolve. To reinforce these ideas, I will use concrete examples of tools and community standards, while stressing that such tools are transitory and should never distract from the bigger picture. The content is based on a course I have given at several schools (https://github.com/sfarrens/Scientific-Software-Dev-Demo), aimed primarily at early-career scientists but offering plenty of useful tips for more experienced developers as well.

<u>Title</u>: Test-driven development

<u>Speaker</u>: James Nightingale

Test-driven development (TDD) is a software practice in which unit tests are written *before* the corresponding code. Development proceeds through a simple six-step cycle:

1. Write a unit test
2. Check the test fails
3. Write the code
4. Check the test (and all previous tests) pass
5. Refactor
6. Repeat

By focusing on code that satisfies tests, developers clearly specify required functionality and avoid unnecessary complexity. This approach emphasises interface over implementation, resulting in simpler, cleaner, and more maintainable code. In the long term, it produces a better-designed and fully tested codebase.

For large-scale collaborative projects, TDD is especially valuable: it makes the intended behaviour of each component explicit, enabling other developers to modify or extend the code with confidence. In this talk, I will give a run-through of how TDD works and illustrate its use in astronomy software development.

<u>Title</u>: Project management at scale

<u>Speaker</u>: Alastair Basden

Managing large scale projects is often a key part of an astronomer's career, whether that is large collaborations, complex software, instruments and telescopes or large scale compute facilities. This session will discuss some of the different project management skills required, and give an overview and summary of several different types of large projects.

<u>Title</u>: Debugging your code

<u>Speaker</u>: Sarah Johnston

When developing code, especially in larger projects, we need to be vigilant about checking the quality and functionality of our code. Debugging is the process of finding and resolving errors in our code, and it is one of the most important development stages for software. In this session we'll discuss some key debugging methods, useful tools for debugging, and how we might implement these for shared repositories or on remote machines as part of larger software development teams.

Title: JAX for scientific computing and inference

Speaker: Matt Graham

JAX is an open-source Python package for high-performance numerical computing. It provides a familiar NumPy style interface but with the advantages of allowing computations to be dispatched to accelerator devices such as graphics and tensor processing units, and supporting transformations to automatically differentiate, vectorize and just-in-time compile functions. While extensively used in machine learning applications, JAX's design also makes it ideal for scientific computing tasks such as simulating numerical models and fitting them to data. This interactive tutorial will introduce JAX's interface, its computation model and functional transforms, and illustrate how it can be used in an example task of inferring the posterior distribution on the parameters of a numerical model given data. The tutorial will also demonstrate how the Array API can be used to write portable code which works across JAX, NumPy and other array backends.

Title: Bayesian inference and sampling techniques

Speaker: Natalia Porqueres

The growing complexity of astronomical datasets requires robust statistical methods to extract reliable scientific insights. Most researchers in astrophysics will, at some point, perform some form of data analysis, ranging from a simple parameter estimation to complex and computationally demanding statistical models. This lecture will provide an understanding of principled data analysis. We will discuss Bayesian statistics and sampling techniques and explore how to design and apply these methods to data.

Title: Bayesian Hierarchical Models & Graphs

Speaker: James Nightingale

Bayesian statistical analyses are increasingly applied to astronomical datasets of rapidly growing size. However, memory limitations make it infeasible to simultaneously fit massive datasets, particularly when global parameters (e.g. cosmology) must be inferred alongside millions of nuisance parameters (e.g. individual galaxy morphologies). Graphical models and expectation propagation (https://arxiv.org/abs/1412.4869) address this “curse of dimensionality” by propagating and refining globally important information across many serial likelihood optimisations on individual galaxies. Hierarchical models, a natural extension of graphical models, provide additional flexibility by capturing population level distributions while accounting for object-to-object variation.

This framework enables fully Bayesian analyses of extremely large datasets, which I demonstrate using the example of measuring the Hubble constant, H_0, from time delays of strongly lensed quasars. Expectation propagation extracts significantly more information than traditional approaches, which typically infer H_0 from each lens independently and then combine results afterwards.

The statistical methods are implemented in the open-source probabilistic programming library PyAutoFit (https://github.com/rhayes777/PyAutoFit). I will conclude by encouraging others to consider how such approaches can scale their own science cases to much larger datasets.

Title: Introduction to Simulation-Based Inference (SBI)

Speaker: Kiyam Lin

In Bayesian parameter inference, accurately modelling data becomes increasingly complex as data fidelity grows, particularly when dealing with computationally intensive calculations such as high-order statistics. To address these challenges, simulation-based inference (SBI) using neural density estimators (NDEs) emerges as a powerful solution. Unlike traditional methods that require direct computation of likelihood functions, SBI enables the learning of posterior distributions, implicit likelihoods, or likelihood ratios directly from simulations, bypassing the need for computationally expensive calculations.

This seminar will provide an in-depth exploration of neural density estimation approaches within SBI, with a particular focus on two methodologies: neural posterior estimation (NPE) and neural likelihood estimation (NLE). In NPE, we learn the posterior distribution $p(\theta|\text{data})$ directly, eliminating the need for sampling methods like MCMC or nested sampling. This approach is particularly efficient in scenarios where numerous posterior distributions are required, such as in astrophysics, where each star may have a unique posterior. On the other hand, NLE involves learning the likelihood function, which can then be used with traditional sampling techniques. The use of NDEs in this context makes the forward pass computationally inexpensive and efficient. With NLE, researchers also gain access to a likelihood function that integrates seamlessly with existing workflows and prior choices through amortization. The tutorial will highlight applications of these methods in cosmology and astrophysics, where they offer significant advantages in handling complex models. Additionally, we will discuss methods to obtain Bayesian evidence estimates using only posterior samples such as with the learned harmonic mean estimator or in the case of nested models, the Savage-Dickey density ratio.

This workshop provides a comprehensive introduction to SBI and NDEs, equipping participants with the knowledge and skills needed to apply these cutting-edge methods effectively in their research across various fields.

<u>Title</u>: SBI: Testing and forward models

<u>Speaker</u>: Maximilian von Wietersheim-Kramsta

Simulation-based inference (SBI) has become an indispensable tool for performing rigorous Bayesian inference where the likelihood function is either intractable or unknown. In this talk, I will provide a broad review of SBI as a method for accurate inference and model testing, particularly in the context of its recent applications in astrophysics and cosmology. I will address some important practical considerations, including data compression schemes, relevant diagnostics and tests, and strategies for ensuring robustness against model misspecification. Additionally, I will discuss how to choose forward simulations fit-for-purpose, what a simulator should actually simulate, and how statistical simulations can be harnessed for efficient SBI. I will also discuss the exciting prospects of transfer learning and multi-fidelity approaches with SBI. Throughout, I will highlight recent applications of SBI and point to practical implementations of these methods. I will conclude by outlining challenges in the fields of astrophysics and cosmology which SBI may help address in the near future.

<u>Title</u>: How to package and release your project

<u>Speaker</u>: Bron Reichardt Chu

Now that you've written a great coding project, how do you share it in a way that makes it easy for others to install, use and contribute to it? I will work through some of the options for releasing code publicly, and the core elements of preparing a research software package for publication following the Journal of Open Source Software (JOSS) submission guidelines. We'll cover licenses, documentation and what you need to supply to your reviewers.