AI Programming

Kick Off

Markus Böck and Jürgen Cito

Research Unit of Software Engineering

194.193 AI Programming: Organisation

All information on TISS/TUWEL and website:
`https://probprog-ai-tuwien.github.io/2025/`

Registration:

Deadline October 9th

Drop-date: October 20th

You have to complete A1 to officially register

Modality / Grading:

6-7 Lectures, 4 **individual** assignments (copying -> 0 marks)

2 assignment discussions (mandatory, Zoom), 1 project in groups of two, oral exam

Grading: 40% assignments, 60% project and oral exam

Elective:

066 645 Data Science

066 926 Business Informatics

066 937 Software Engineering & Internet Computing

Individual Assignments:

Jupyterlab (mostly Python, link in TUWEL)

A1 deadline 20.10.
A2 deadline 13.11.
Discussion A1 & A2 20.11. (Zoom, link in TUWEL)

A3 deadline 27.11.
A4 deadline 11.12.
Discussion A3 & A4 11.12. (Zoom, link in TUWEL)

194.193 AI Programming: Organisation

Group Project:

Work in pairs
Written report 09.01.
1) Apply probabilistic programming on real world data
2) Implement and evaluate inference algorithm

Oral exam:

26.01 - 30.01
in pairs
question catalog
questions on project

Outlook

- What is probabilistic programming?
- Probability Theory
- Probabilistic Modelling
- Bayesian Inference
- Probabilistic Programming Languages
- Applications

What is Probabilistic Programming?

What is Probabilistic Programming?

Textbook definition:

> *Probabilistic programming is a* ***programming paradigm*** *in which* ***probabilistic models*** *are specified and* ***inference*** *for these models is performed automatically.*

▷ Probabilistic models as programs

▷ Automatic posterior inference

(Explained later)

Where is the AI?

Probabilistic Programming is AI!

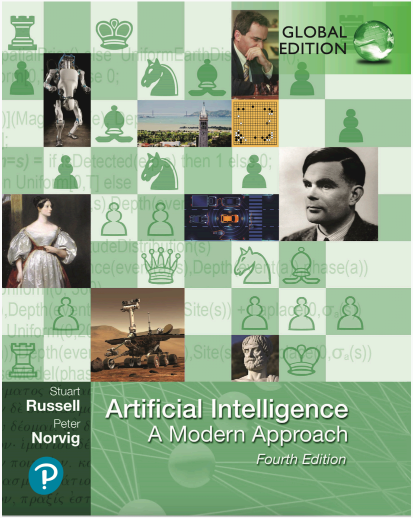
Machine Learning

- Programs define neural networks
- Data: input-output pairs
- Encodes how input maps to output
- Optimise parameters with automatic differentiation to minimise error in mapping
- Black-box approach

Probabilistic Programming

- Programs define probabilistic models
- Data: some observed data
- Encodes how unknown variables generated data
- Find distribution over unknown variables with automatic inference that ”fits” the data
- Explicit modelling + uncertainty quantification

Probabilistic Programming is AI!



IV Uncertain knowledge and reasoning

18 Probabilistic Programming . . . **641**
18.1 Relational Probability Models . . . 642
18.2 Open-Universe Probability Models . . . 648
18.3 Keeping Track of a Complex World . . . 655
18.4 Programs as Probability Models . . . 660
Summary . . . 664
Bibliographical and Historical Notes . . . 665

Probabilistic Programming is AI!

What is thinking?

How can we describe the intelligent inferences made in everyday human reasoning?

How can we engineer intelligent machines?

Computational theory of mind

mind = computer

mental representations = computer programs

run(program)

thinking = running a program

Probabilistic Programming is AI!

What kind of programs can represent thinking?

Structure

Knowledge

Probability

Uncertainty

Why Probabilistic Programming?

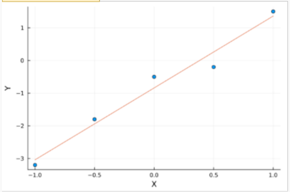
- Probabilistic models allow us to
  - incorporate prior knowledge
  - describe dependencies between variables
  - handle uncertainty
- Probabilistic programs specify probabilistic models
- Inference is concerned about updating our knowledge / belief about unknown or uncertain quantities in the program
- This is achieved by conditioning / constraining the model with observed data

Why Probabilistic Programming?

- Traditionally statisticians developed probabilistic models on paper and implemented inference algorithms
- **Probabilistic programming separates modelling from inference**
- **Expressivity:** Any probabilistic model can be implemented as a probabilistic program
- **General-purpose inference algorithms** + inference engineering
- **Enable incorporation of programming language and software engineering advances** (program analysis, debugging, visualisations,...)

First Look at Probabilistic Programming

Data



$$y = \underbrace{k}_{\text{slope}} \cdot x + \underbrace{d}_{\text{intercept}}$$

Probabilistic Model

```julia
using Turing
@model function linear_regression(x, y)
    # prior over latents
    slope ~ Normal(0, 3)
    intercept ~ Normal(0, 3)

    # likelihood
    for i in 1:length(x)
        # y ≈ slope * x + intercept
        y[i] ~ Normal(slope * x[i] + intercept, 1.)
    end
end
```

Posterior Inference

```julia
using AdvancedMH
function do_inference()
    x = [-1., -0.5, 0.0, 0.5, 1.0]
    y = [-3.2, -1.8, -0.5, -0.2, 1.5]
    model = linear_regression(x, y)
    res = sample(model,
        MH(
            :slope => RandomWalkProposal(Normal(0,0.1)),
            :intercept => RandomWalkProposal(Normal(0,0.2))
        ),
        1000
    )
    maximum_a_posteriori_ix = argmax(res[:lp])
    return (
        res[:slope][maximum_a_posteriori_ix],
        res[:intercept][maximum_a_posteriori_ix]
    )
end
```

First Look at Probabilistic Programming

```julia
using Turing
@model function linear_regression(x, y)
    # prior over latents
    slope ~ Normal(0, 3)
    intercept ~ Normal(0, 3)

    # likelihood
    for i in 1:length(x)
        # y ≈ slope * x + intercept
        y[i] ~ Normal(slope * x[i] + intercept, 1.)
    end
end
```

```julia
using AdvancedMH
function do_inference()
    x = [-1., -0.5, 0.0, 0.5, 1.0]
    y = [-3.2, -1.8, -0.5, -0.2, 1.5]
    model = linear_regression(x, y)
    res = sample(model,
        MH(
            :slope => RandomWalkProposal(Normal(0,0.1)),
            :intercept => RandomWalkProposal(Normal(0,0.2))
        ),
        1000
    )
    maximum_a_posteriori_ix = argmax(res[:lp])
    return (
        res[:slope][maximum_a_posteriori_ix],
        res[:intercept][maximum_a_posteriori_ix]
    )
end
```

Choice of Priors

Choice of Likelihood

Choice of Inference

Choice of Visualistation

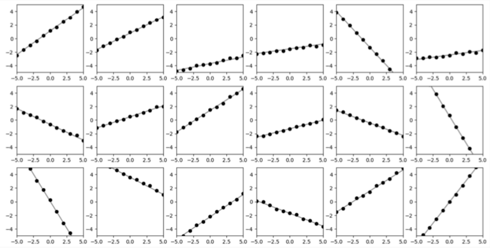
Feedback Cycle

SE for PPL Research in our research group

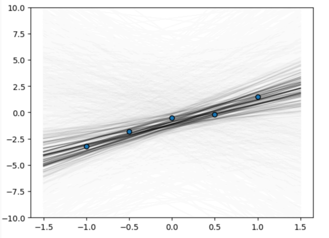
- Program Comprehension (Reasoning about Programs)
- Software Evolution (Reasoning about Change)
- Software Visualization (Reasoning about Large-scale Traces)
- Software Testing (Reasoning about Correctness)

First Look at Probabilistic Programming: Visualisation

Possible worlds according to model



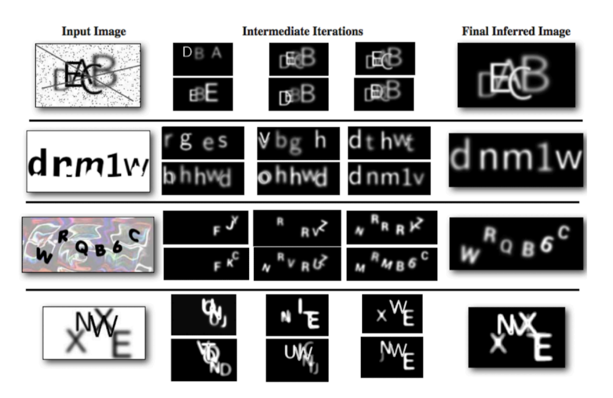
Posterior distribution



Applications

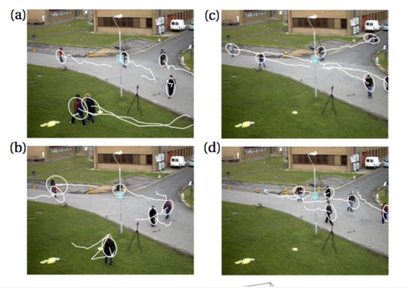
Captcha breaking

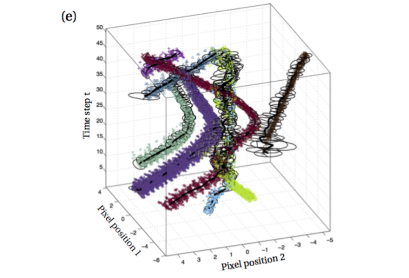
Mansinghka, V. K., Kulkarni, T. D., Perov, Y. N., & Tenenbaum, J. (2013). Approximate bayesian image interpretation using generative probabilistic graphics programs. Advances in Neural Information Processing Systems, 26.



Object Tracking

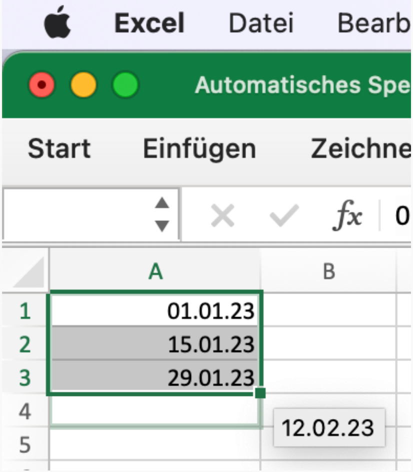
Neiswanger, W., Wood, F., & Xing, E. (2014, April). The dependent Dirichlet process mixture of objects for detection-free tracking and object modeling. In Artificial Intelligence and Statistics (pp. 660-668). PMLR.





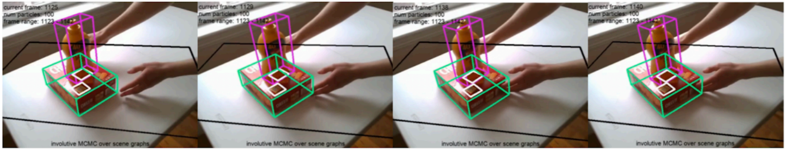
Excel Auto-Fill

Gulwani, S. (2011). Automating string processing in spreadsheets using input-output examples. ACM Sigplan Notices, 46(1), 317-330.



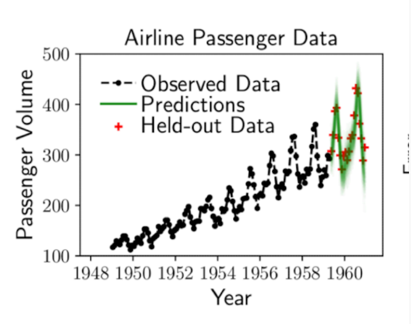
Pose Estimation

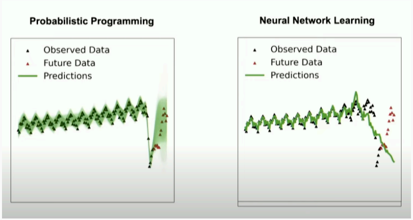
Cusumano-Towner, M. F. (2020). Gen: a high-level programming platform for probabilistic inference (Doctoral dissertation, Massachusetts Institute of Technology). Kulkarni, T. D., Kohli, P., Tenenbaum, J. B., & Mansinghka, V. (2015). Picture: A probabilistic programming language for scene perception. In Proceedings of the ieee conference on computer vision and pattern recognition (pp. 4390-4399).



(b) For each frame in (a), the inferred 6DoF object poses and object-object contact planes

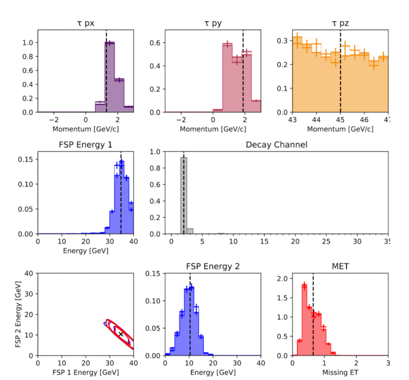
Cusumano-Towner, M. F. (2020). Gen: a high-level programming platform for probabilistic inference (Doctoral dissertation, Massachusetts Institute of Technology).





Hadron Collider

Baydin, A. G., Shao, L., Bhimji, W., Heinrich, L., Meadows, L., Liu, J., … & Wood, F. (2019, November). Etalumis: Bringing probabilistic programming to scientific simulators at scale. In Proceedings of the international conference for high performance computing, networking, storage and analysis (pp. 1-24).



Nuclear Test Detection

Arora, N. S., Russell, S., & Sudderth, E. (2013). NET-VISA: Network processing vertically integrated seismic analysis. Bulletin of the Seismological Society of America, 103(2A), 709-729.