Towards Inference Amortization for BUGS models: BUGS to Anglican compilation

Adam Goliński, Frank Wood

Department of Engineering Science
University of Oxford

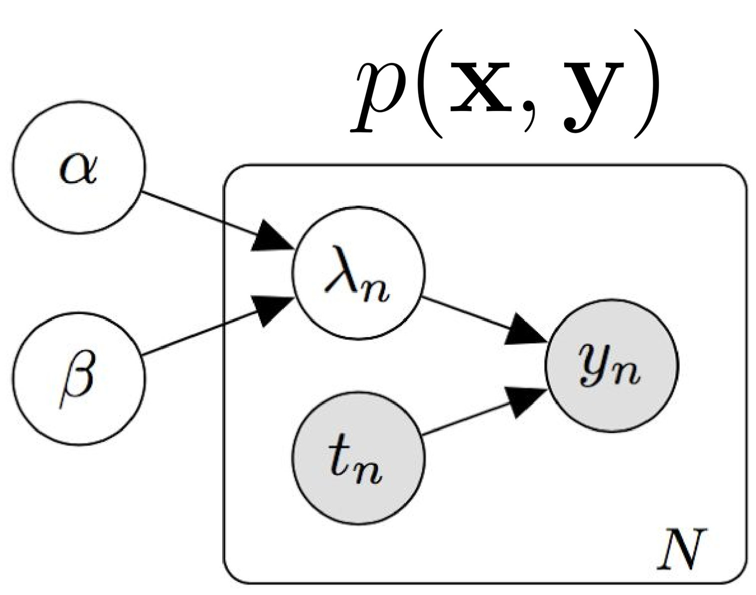
{adamg,fwood}@robots.ox.ac.uk

Outline

- **Probabilistic Programming Languages** (PPL) are a special class of programming languages which allow users to specify probabilistic models and run inference on them, i.e. find $p(\mathbf{x}|\mathbf{y})$
  x are latents and **y** are observed variables
- **BUGS** is a popular probabilistic programming language allowing to describe graphical models
- **Inference amortization** is a technique that greatly reduces the computational cost of run-time inference by training a neural network approximating the posterior distribution $q(\mathbf{x}|\mathbf{y};\, \phi) \sim p(\mathbf{x}|\mathbf{y})$ ahead of the time of the system operation
  ϕ are the learnt parameters of the neural network
- **Anglican** is a universal, research-oriented PPL which **implements** some of the cutting-edge inference techniques including **inference amortization**
- **To enable BUGS models to use inference amortization** we have created a compiler translating models from BUGS to Anglican
- **Next steps**
  - completing the translation of the entire feature set of the BUGS language
  - application and further improvement of the inference amortization approach which takes advantage of the structure of the forward graphical model [3] to automate the design of the neural network and is perfectly suited for the class of models expressible in BUGS

Pump failure model

Hierarchical model for failure rates of power plant pumps



$$\begin{aligned} \alpha &\sim \text{Exponential}(1.0) \\ \beta &\sim \text{Gamma}(0.1, 1.0) \\ \lambda_n &\sim \text{Gamma}(\alpha, \beta) \\ y_n &\sim \text{Poisson}(\lambda_n t_n) \end{aligned}$$

Figure 1. Forward graphical model [3]

$$\mathbf{x} = \{\alpha, \beta\} \cup \{\lambda_n\}_N$$
$$\mathbf{y} = \{t_n, y_n\}_N$$

λ_n rate of failure for pump *n*
y_n number of failures for pump *n*
t_n length of operation time for pump *n*

BUGS

```
model
{
  for (i in 1 : N) {
    lambda[i] ~ dgamma(alpha, beta)
    y[i] ~ dpois(lambda[i] * t[i])
  }
  alpha ~ dexp(1)
  beta ~ dgamma(0.1, 1.0)
}
```

```
data
{
  "N" <- 2
  "t" <- c(94.3, 15.7)
  "y" <- c(5, 1)
}
```

Anglican

```
(let
  [N 2
   t [94.3 15.7]
   y [5 1]
   lambda [nil nil]
   alpha (sample (exponential 1))
   beta (sample (gamma 0.1 1))
   lambda (assoc-in lambda [0] (sample (gamma alpha beta)))
   lambda (assoc-in lambda [1] (sample (gamma alpha beta)))
   _ (observe
       (poisson (* (get-in lambda [0]) (get-in t [0])))
       (get-in y [0]))
   _ (observe
       (poisson (* (get-in lambda [1]) (get-in t [1])))
       (get-in y [1]))])
```

Inference amortization

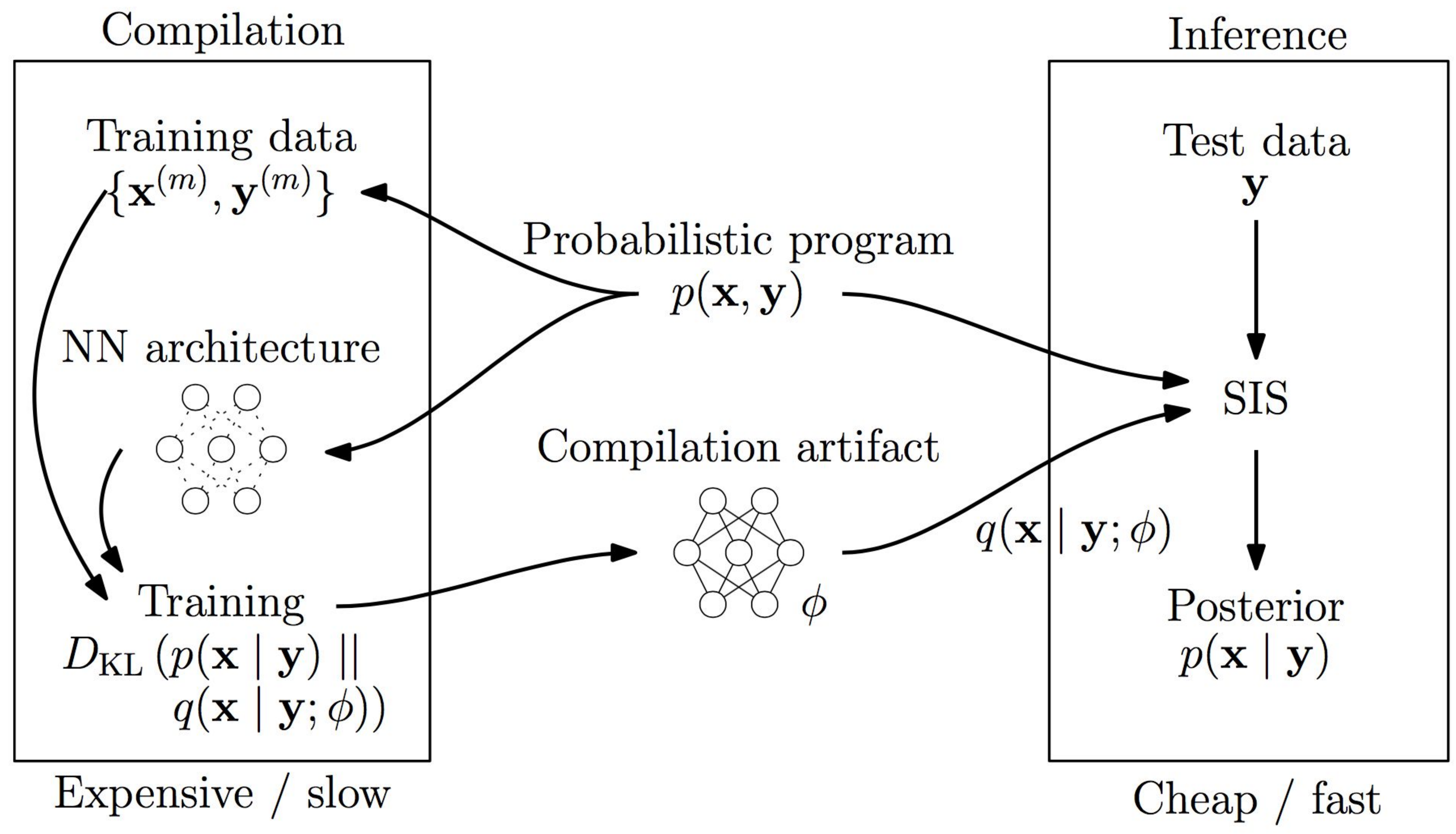


Figure 2. Inference amortization framework [2]
SIS stands for Sequential Importance Sampling

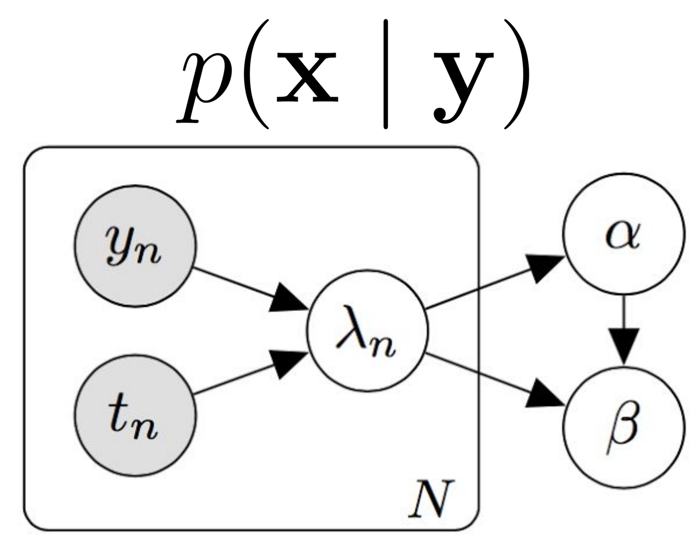


Figure 3. Inverted graphical model [3]

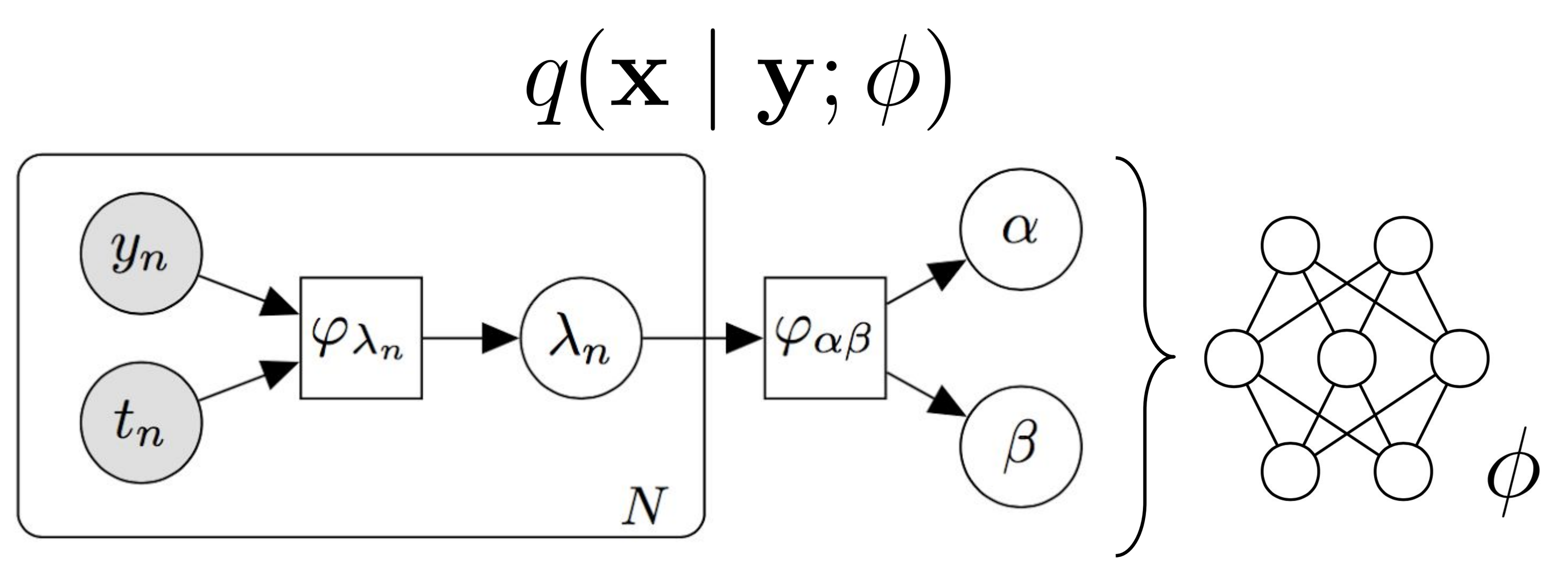


Figure 4. Inference network with MADE-like neural networks [3]

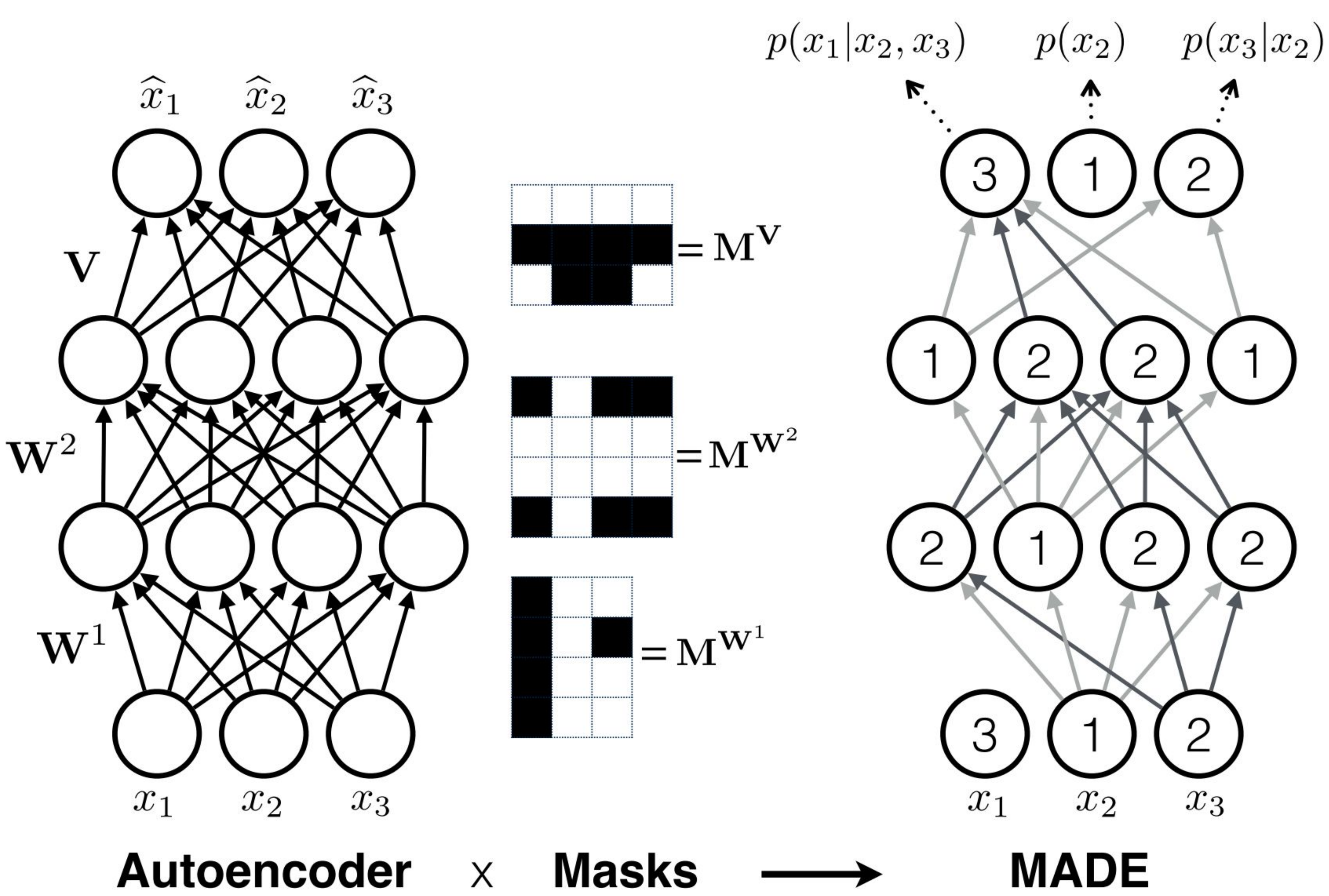


Figure 5. Masked Autoencoder for Distribution Estimation [1]

References

[1] M. Germain, K. Gregor, I. Murray, and H. Larochelle. MADE: masked autoencoder for distribution estimation. In Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015, pages 881–889, 2015.
[2] T. A. Le, A. G. Baydin, and F. Wood, "Inference compilation and universal probabilistic programming," in Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS), ser. Proceedings of Machine Learning Research, vol. 54. Fort Lauderdale, FL, USA: PMLR, 2 017, pp. 1338–1348.
[3] B. Paige and F. Wood, "Inference networks for sequential Monte Carlo in graphical models," in Proceedings of the 33rd International Conference on Machine Learning, ser. JMLR, vol. 48, 2016.