Machine Learning for Modelling Physical Systems

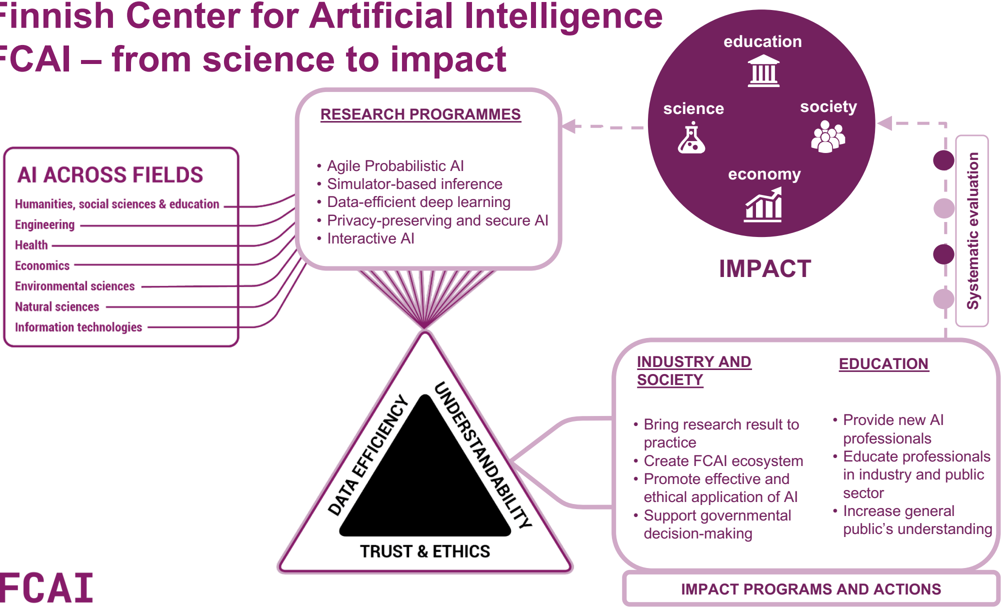
Antti Honkela

Finnish Center for Artificial Intelligence FCAI
Department of Computer Science
University of Helsinki

4TU.AMI Models and Data in Digital Twins
14 June 2019

FCAI Finnish Center for Artificial Intelligence

Finnish Center for Artificial Intelligence FCAI – from science to impact



FCAI

Outline

Machine learning and AI overview

Learning for differential equations with probabilistic models

Other interesting probabilistic models

Probabilistic programming

Outline

Machine learning and AI overview

Learning for differential equations with probabilistic models

Other interesting probabilistic models

Probabilistic programming

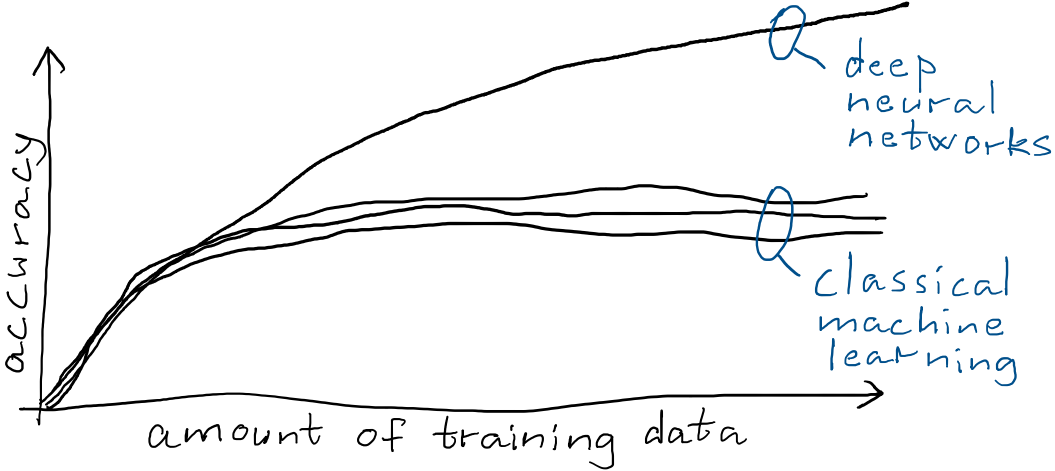
What is AI?

- Two types of AI
  - Symbolic/logical
  - Machine learning (ML): imitation-based AI
- Current revolution in machine-learning-based AI
  - Combination of big data, models that benefit from big data, more computing power (GPUs) and accessible programming environments
- We are nowhere close to human-level intelligence
  - Imitation of examples in the data, not thinking

Flavours of ML

- Supervised learning
  - E.g. classification, regression, time series prediction, emulators for expensive simulators
  - Outcome: map: $x \mapsto y$
- Reinforcement learning
  - Planning
  - Outcome: policy: $(\text{state}, \text{observations}) \mapsto \text{actions}$
- Unsupervised learning
  - E.g. dimensionality reduction, generative modelling

Big data revolution in ML



Deep neural networks and data

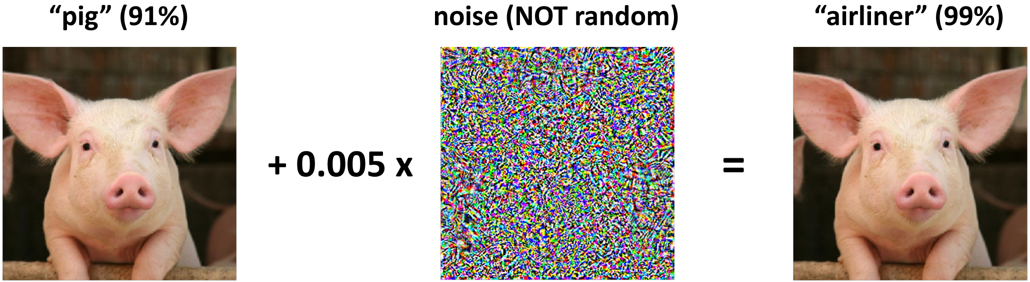
- Most typical applications in *supervised learning*
  - Require annotated (input, target output) pairs
- Current methods need a lot of data
- 100000 cases is a good start, the more the better!
  - Upper limit still has not been found!
- Research viewpoint: less data may be OK, but more work and expertise needed for good results

Limitations of deep neural networks (DNNs)

- DNNs are susceptible to *adversarial examples*
  - In classification: selected examples with imperceptible differences are seriously misclassified

Limitations of deep neural networks (DNNs)

- DNNs are susceptible to *adversarial examples*
  - In classification: selected examples with imperceptible differences are seriously misclassified



Szegedy et al. (arXiv:1312.6199) via https://adversarial-ml-tutorial.org/

Limitations of deep neural networks (DNNs)

- DNNs are susceptible to *adversarial examples*
  - In classification: selected examples with imperceptible differences are seriously misclassified
- This is a feature, not a bug
  - Robustness–accuracy trade-off
  - More prior knowledge (e.g. structured models) can help
- Major challenge for reinforcement learning and optimisation
  - Algorithms will learn to exploit any weaknesses of the model

Outline

Machine learning and AI overview

Learning for differential equations with probabilistic models

Other interesting probabilistic models

Probabilistic programming

Probabilistic modelling and differential equations

- Inference of unknown parameters θ and initial conditions x_0 in an ODE from noisy observations $Y = [y(t_1), \dots, y(t_n)]$, where

$$x'(t) = g(x(t), \theta), \quad x(0) = x_0$$
$$y(t_i) = x(t_i) + \eta_i$$

- Inference of latent driving functions $f(t)$ (latent force models)

$$x'(t) = g(x(t), f(t), \theta), \quad x(0) = x_0$$
$$y(t_i) = x(t_i) + \eta_i$$

Modelling latent driving functions: Gaussian processes

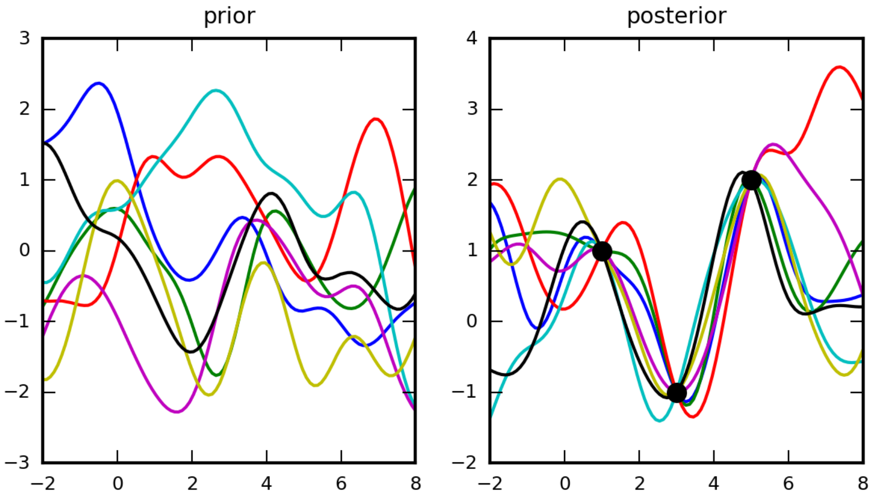
- Gaussian process priors on driving functions $f(t)$
  - Functional prior, specified by mean and covariance functions
  - No need for time discretisation
  - Can capture diverse activation profiles

$$f(t) \backsim \mathcal{GP}\left(\mu(t), k\left(t, t'\right)\right)$$

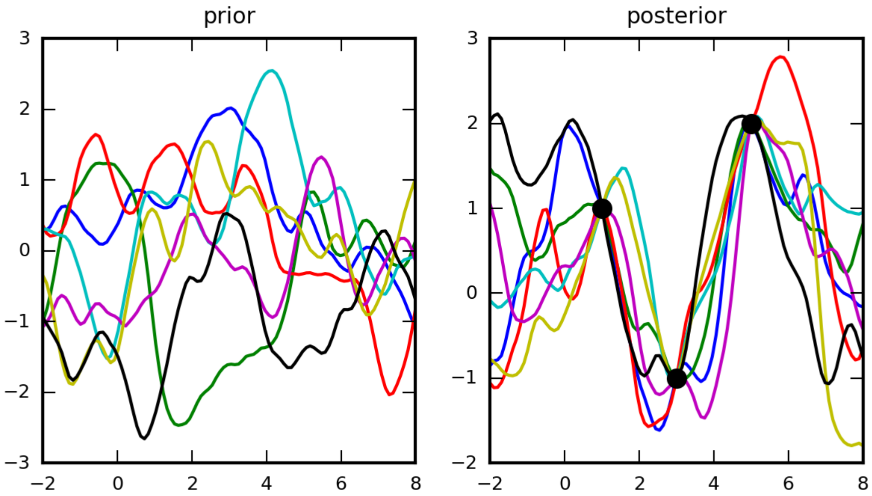
where

$$\begin{aligned}\mu(t) &= \mathbb{E}\left[f(t)\right] = \langle f(t) \rangle \\ k\left(t, t'\right) &= \mathbb{E}\left[\left(f(t) - \mu(t)\right)\left(f\left(t'\right) - \mu\left(t'\right)\right)\right]\end{aligned}$$

Gaussian process examples: squared exponential covariance



Gaussian process examples: Matern covariance



Gaussian processes and ODEs (Lawrence et al., NIPS 2006)

- Assume $x \sim \mathcal{N}(\mu, \Sigma)$
- Affine transformation $Ax + b$ follows

$$(Ax + b) \sim \mathcal{N}(A\mu + b, A\Sigma A^T)$$

Gaussian processes and ODEs (Lawrence et al., NIPS 2006)

- Assume $x \sim \mathcal{N}(\mu, \Sigma)$
- Affine transformation $Ax + b$ follows

$$(Ax + b) \sim \mathcal{N}(A\mu + b, A\Sigma A^T)$$

- Insight: an analogous property applies to Gaussian processes:
  For suitable $g()$, the solution for $x(t)$ in

$$\frac{\mathrm{d}\, x(t)}{\mathrm{d}\, t} = g(x(t), f(t), \theta)$$

  is an affine operator $x(t) = \mathcal{L}_g(f(t))$ of $f(t)$
  $\Rightarrow$ Joint Gaussian process over $f(t), x(t)$

ODE Gaussian process

- Assuming $x(t) \sim \mathcal{GP}(\mu_x(t), k_{xx}(t, t'))$, how to evaluate the mean function $\mu_x(t)$ and covariance $k_{xx}(t, t')$?

$$\mu_x(t) = \mathrm{E}_{p(f(t))}[\mathcal{L}_g(f(t))]$$
$$k_{xx}(t, t') = \mathrm{E}_{p(f(t), f(t'))}[(\mathcal{L}_g(f(t)) - \mu_x(t))(\mathcal{L}_g(f(t)) - \mu_x(t))^T]$$

- For suitable $k_{ff}(t, t')$ and linear g, these can be evaluated in closed form, leading to very efficient computation
- E.g. squared exponential covariance:

$$k_{ff}(t, t') = \alpha \exp\left(\frac{(t - t')^2}{2\ell^2}\right)$$

ODE Gaussian process applications I

- Single input motif gene regulation (Lawrence et al., NIPS 2006; Gao et al., Bioinformatics 2008):

$$\frac{\mathrm{d}\, x_i(t)}{\mathrm{d}\, t} = B_i + S_i f(t) - D_i x_i(t)$$

  - $x_i(t)$ target gene expression
  - $f(t)$ regulator activity

ODE Gaussian process applications II

- Translation+transcription model of gene regulation (Honkela et al., PNAS 2010; Gao et al., Bioinformatics 2008):

$$\frac{\mathrm{d}\,p(t)}{\mathrm{d}\,t} = f(t) - \delta p(t)$$
$$\frac{\mathrm{d}\,x_i(t)}{\mathrm{d}\,t} = B_i + S_i p(t) - D_i x_i(t)$$

  - $x_i(t)$ target gene expression
  - $p(t)$ regulator activity
  - $f(t)$ regulator mRNA expression

ODE Gaussian process applications III

- Modelling transcription+expression (Honkela et al., PNAS 2015):

$$\frac{\mathrm{d}\, x(t)}{\mathrm{d}\, t} = B + Sf(t - \Delta) - Dx(t)$$

  - $x(t)$ gene expression
  - $f(t)$ transcriptional activity

Non-linear ODEs

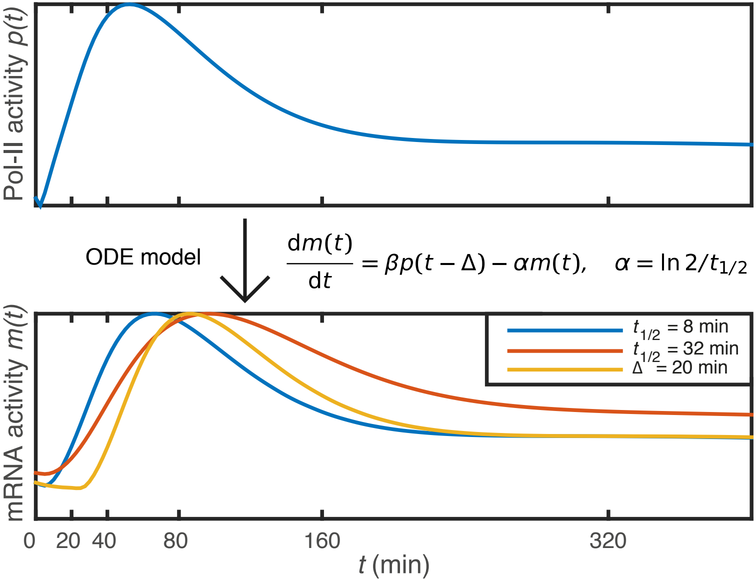
- If $\mathcal{L}_g$ is not affine, $x(t)$ will not follow Gaussian process
- Approximations still possible
- E.g. non-linear gene regulation model by Titsias et al. (BMC Systems Biology, 2012):

$$\frac{\mathrm{d}\, p_i(t)}{\mathrm{d}\, t} = f_i(t) - \delta_i p_i(t)$$
$$\frac{\mathrm{d}\, x_j(t)}{\mathrm{d}\, t} = b_j + s_j G\left(p_1(t), \ldots, p_I(t); \theta_j\right) - d_j x_j(t)$$

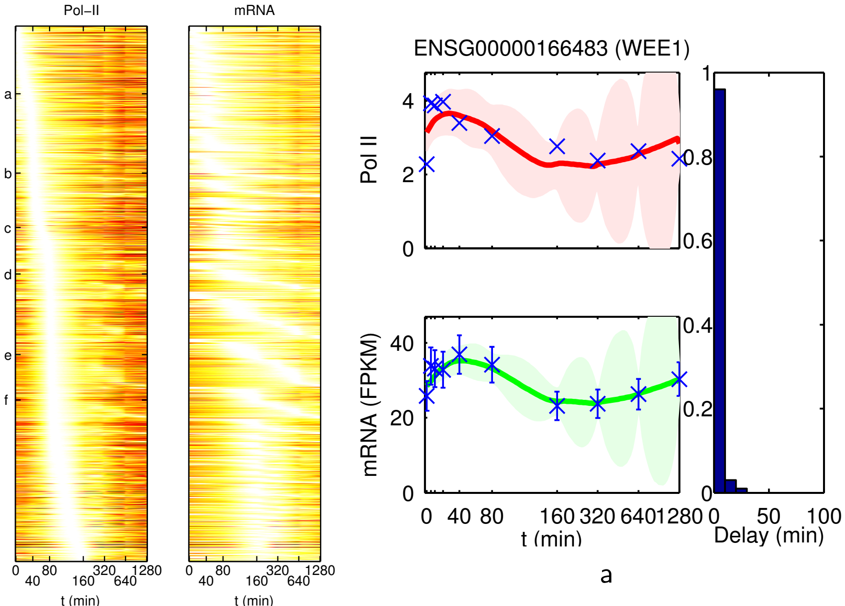
with

$$G(p_1(t), \ldots, p_I(t); \mathbf{w}_j, w_{j0}) = \frac{1}{1 + e^{-w_{j0} - \sum_{i=1}^{I} w_{ji} \log p_i(t)}}$$

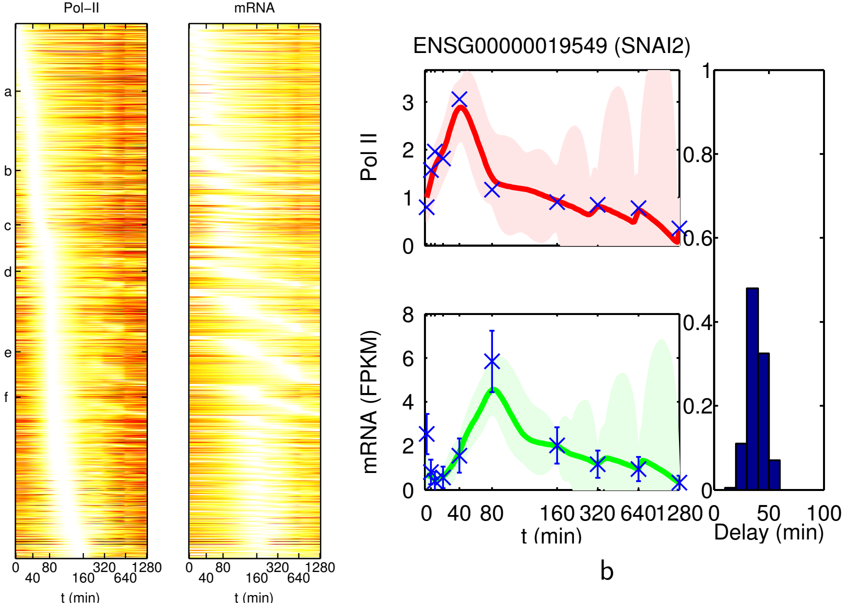
Gene transcription and expression model (Honkela et al., PNAS 2015)



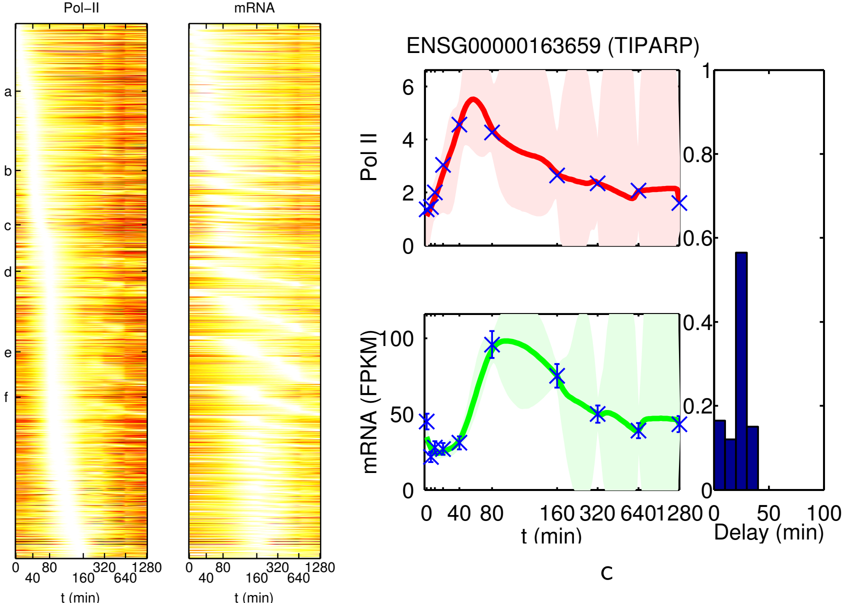
Gene transcription and expression fits (Honkela et al., PNAS 2015)



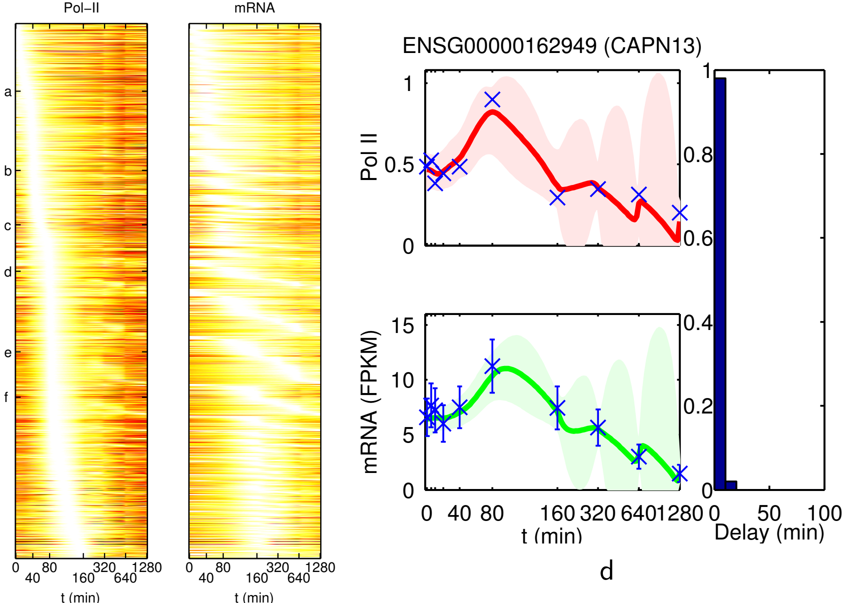
Gene transcription and expression fits (Honkela et al., PNAS 2015)



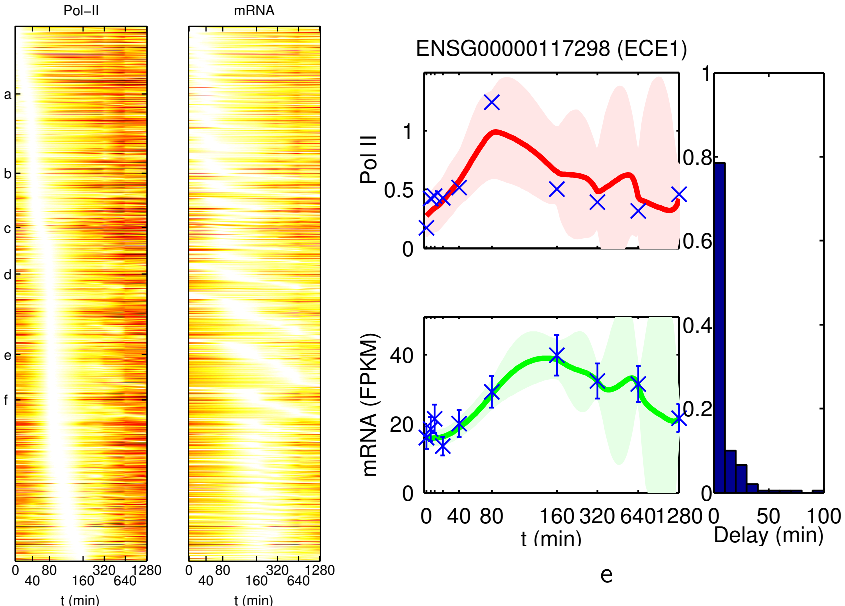
Gene transcription and expression fits (Honkela et al., PNAS 2015)



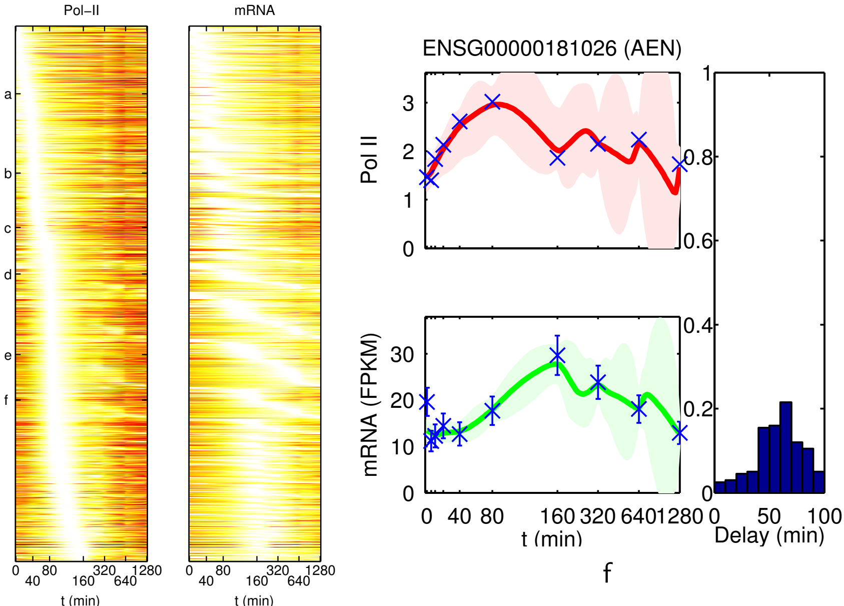
Gene transcription and expression fits (Honkela et al., PNAS 2015)



Gene transcription and expression fits (Honkela et al., PNAS 2015)



Gene transcription and expression fits (Honkela et al., PNAS 2015)



f

Outline

Machine learning and AI overview

Learning for differential equations with probabilistic models

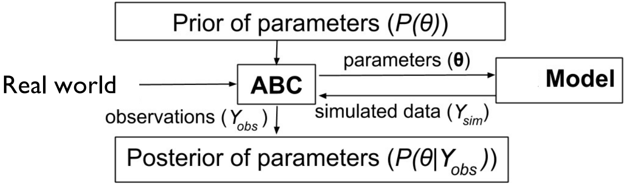
Other interesting probabilistic models

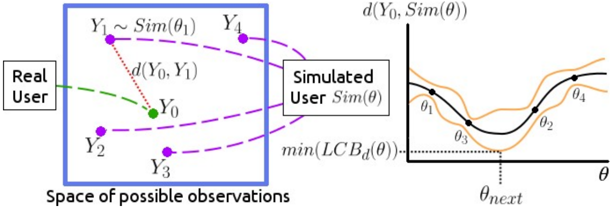
Probabilistic programming

Inferring simulators from data

How to fit a model to data when no standard tools apply

- only indirect observations
- likelihoods intractable

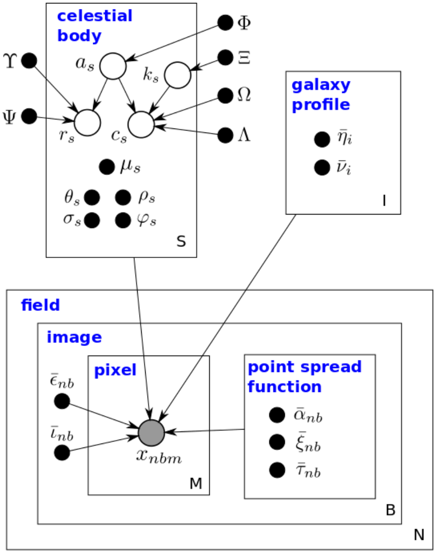




BOLFI: Gutmann & Corander 2016

Example: Probabilistic modelling in cosmology (Regier et al., ICML 2015)

Figure 1. An image from the Sloan Digital Sky Survey (SDSS, 2015) of a galaxy from the constellation Serpens, 100 million light years from Earth, along with several other galaxies and many stars from our own galaxy.



Outline

Machine learning and AI overview

Learning for differential equations with probabilistic models

Other interesting probabilistic models

Probabilistic programming

Implementation: Probabilistic programming

- Probabilistic inference is hard
  - Typically expert derivations, coding & tuning are required for good results
  - Some easy-to-use frameworks exists, but often limited in scope
  - Almost all real applications require computational approximations
  - Non-trivial to judge if these are accurate enough
- Idea of probabilistic programming: user writes a description of the model, the machine takes care of the rest
- Cf. writing machine code in assembly language vs. high level code
- Key challenge: how to perform inference efficiently
- Emerging solutions: Stan, Edward, PyMC3, Pyro, ELFI, ...

Probabilistic programming example (Carpenter et al., JASS 2016)

```
parameters {
  simplex[K] theta[M];   // topic dist for doc m
  simplex[V] psi[K];     // word dist for topic k
}
model {
  for (m in 1:M)
    theta[m] ~ dirichlet(alpha);  // prior
  for (k in 1:K)
    psi[k] ~ dirichlet(beta);     // prior
  for (n in 1:N) {
    real gamma[K];
    for (k in 1:K)
      gamma[k] <- log(theta[doc[n],k]) + log(psi[k,w[n]]);
    increment_log_prob(log_sum_exp(gamma));  // likelihood
  }
}
```

Conclusion

- Deep neural networks (most) useful for unstructured problems with massive data
- Probabilistic models allow incorporating structure such as known physics
- Likelihood-free inference can incorporate existing simulators
- Gaussian processes are a powerful tool for modelling latent functions
- Probabilistic programming big help for implementation

http://mc-stan.org

http://edwardlib.org

http://pyro.ai

http://gpflow.org

http://elfi.ai