Probabilistic programming: A new paradigm in machine learning

Thomas Hamelryck
thamelry@binf.ku.dk
BIO/DIKU, University of Copenhagen
Århus, October 2018

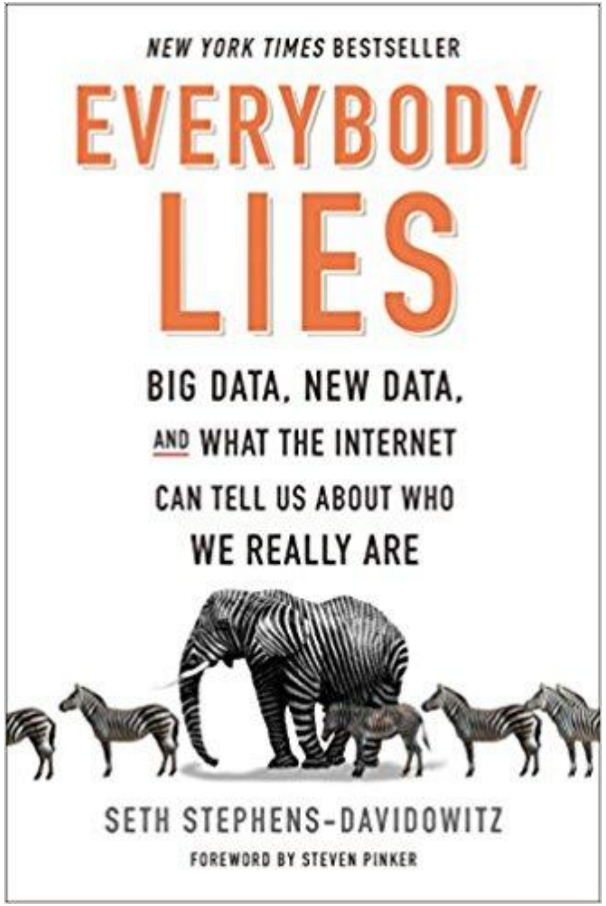
“Data, the oil of the digital era”

“A new commodity spawns a lucrative, fast-growing industry [...]. A century ago, the resource in question was oil. Now similar concerns are being raised by the giants that deal in data, the oil of the digital era. These titans — Alphabet (Google’s parent company), Amazon, Apple, Facebook and Microsoft — look unstoppable. They are the five most valuable listed firms in the world.”

-The Economist, May 6th, 2017

#1 Big Data

- Government
- Science
- Medicine
- Manufacturing
- Healthcare
- Business
- Education
- Internet of Things (IoT)
- Data anthropology
- …



From data to wisdom - inference

- Inference is reasoning guided by data
- Peirce distinguishes three kinds of inference
- Deduction
  - Logic, symbolic manipulation
  - No uncertainty, deterministic
- Induction
  - Estimate the parameters of a model from data, under uncertainty
- Abduction
  - Choose any of the models that fit the data, under uncertainty

Charles Sanders Peirce (1839-1914)

The abstraction explosion

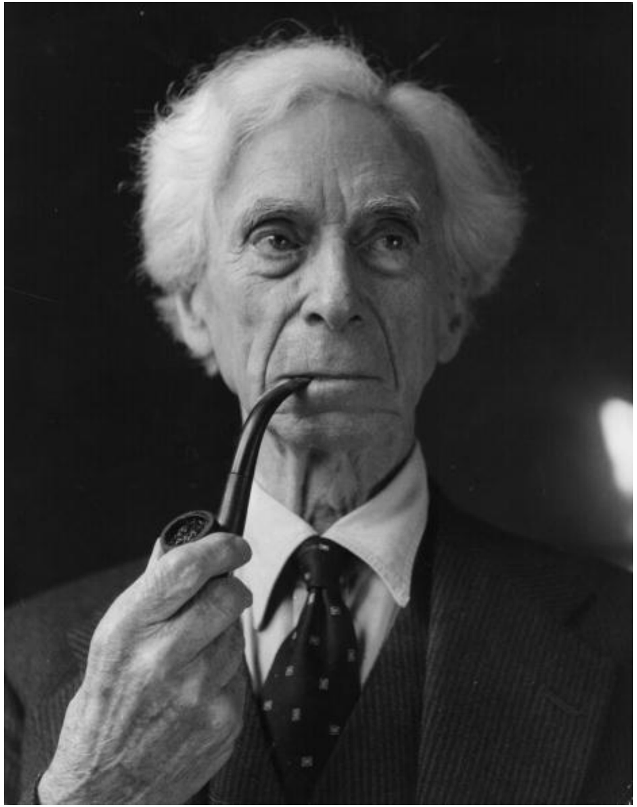
Year	*Model*
2004	Cyc knowledge management, 6 million FOPC/CycL propositions
2012	34.000 lines of Python/Cuda for Imagenet (Krizhevsky *et al.*)
2013	1.571 lines of Lua to play Atari games
2017	196 lines of Keras to implement Deep Dream
2018	<100 lines of Keras for research paper level results

Source: Monica Anderson, Syntience Inc.

Abstraction is power

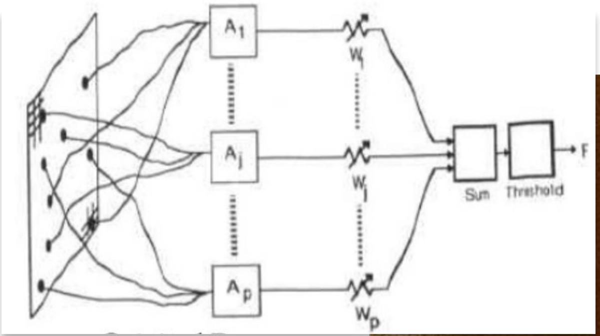
“Abstraction, difficult as it is, is the source of practical power. A financier, whose dealings with the world are more abstract than those of any other ‘practical’ person, is also more powerful than any other practical person.”

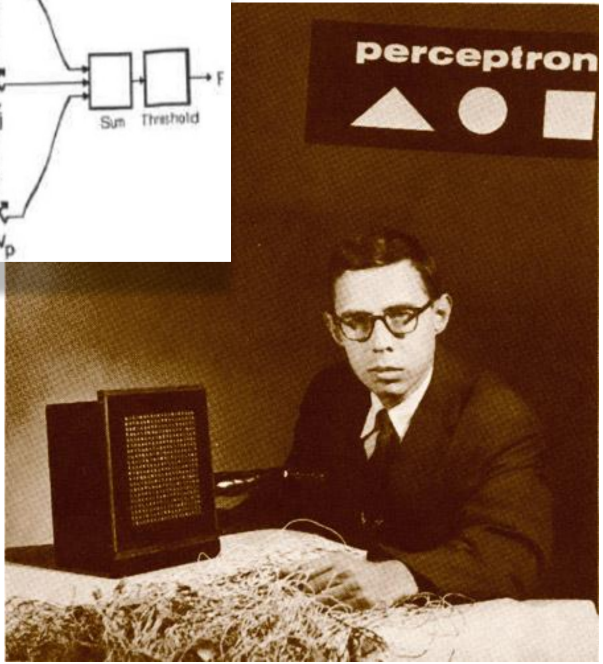
– Bertrand Russell, British philosopher, logician and social critic (1872-1970)



#2 Deep Learning

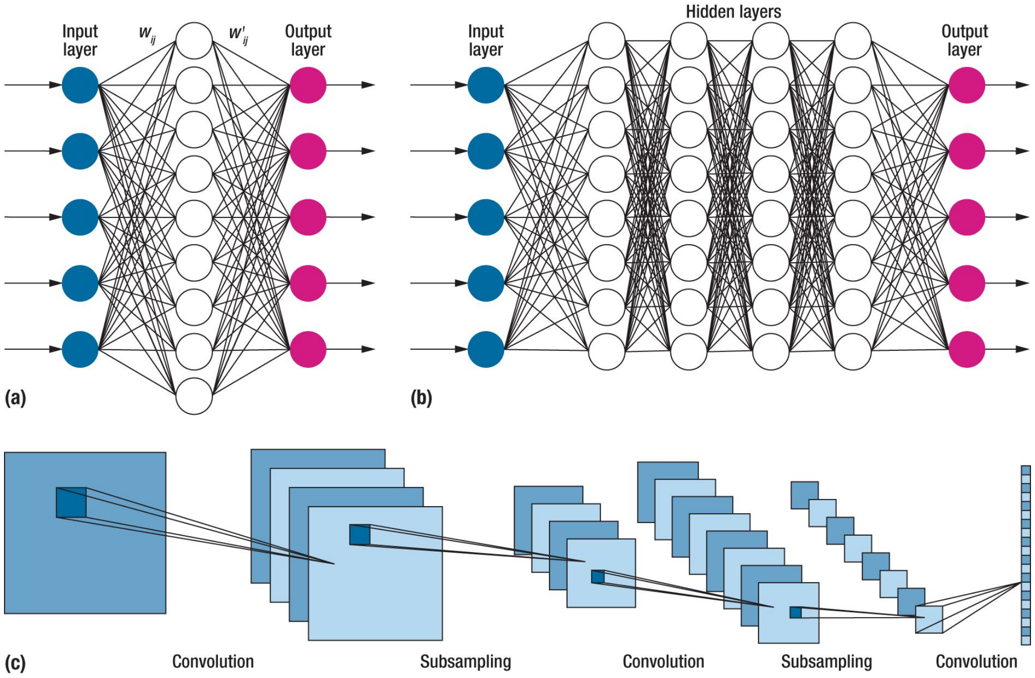
- Roots: the perceptron
  - Frank Rosenblatt, 1957
- Deep neural networks, 2012
  - Neural network revival
  - GPUs
  - Large data sets
  - Algorithms & software
- Problems
  - Black box
  - Overfitting, uncertainties





#2 Deep Learning

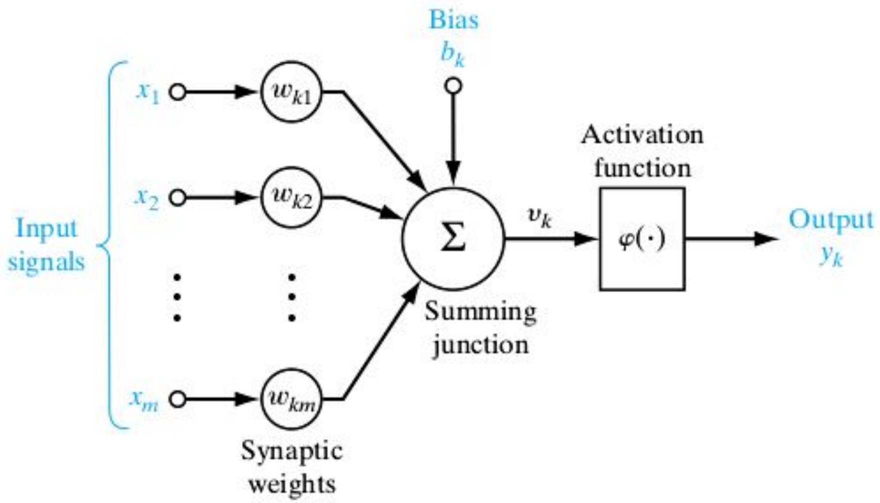
- Roots: the perceptron
  - Frank Rosenblatt, 1957
- Deep neural networks, 2012
  - Neural network revival
  - GPUs
  - Large data sets
  - Algorithms & software
- Problems
  - Black box
  - Overfitting, uncertainties



Picture: IEEE Software 2017 vol. 34

The humble digital neuron...

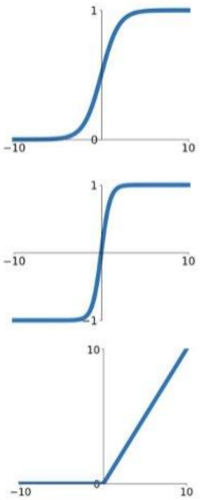
- Calculates the weighted sum of the inputs
- Applies a non-linear function to the sum



Sigmoid
$\sigma(x) = \frac{1}{1+e^{-x}}$

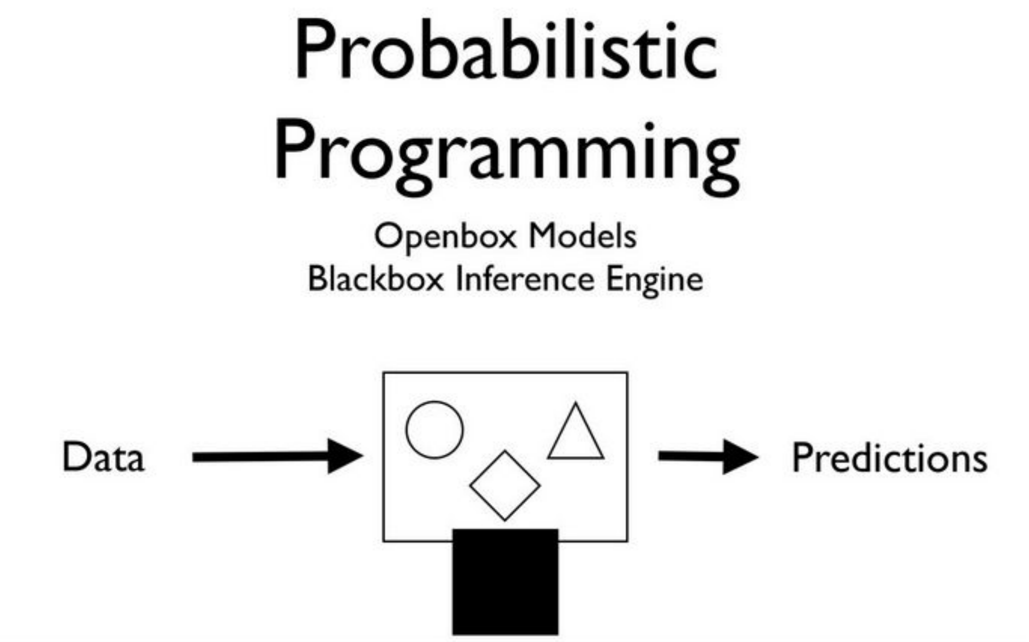
tanh
$\tanh(x)$

ReLU
$\max(0, x)$



Picture: The Men Who Stare at Codes/Shruti Jadon

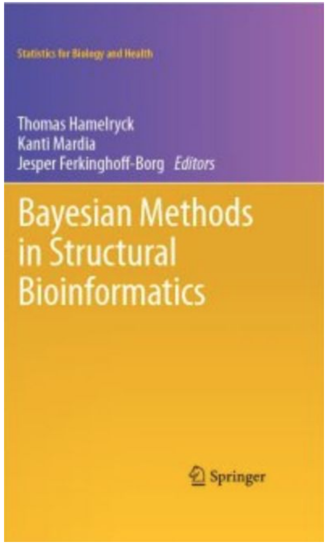
#3 Probabilistic programming

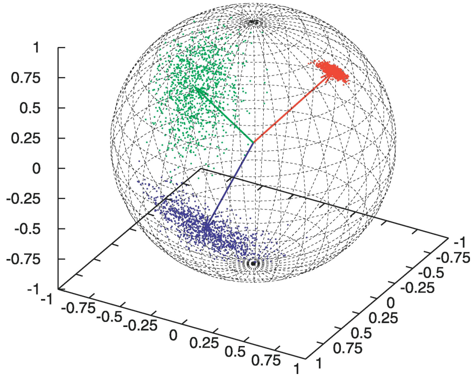


Picture: Olivier Grisel's talk on ML

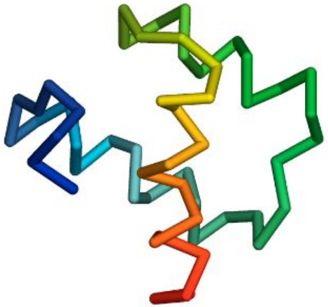
How I got involved - Mocapy

- Mocapy (2006) is a PP package for sequences and directional statistics.
- Probabilistic models of protein structure
    - Protein structure prediction
        - PLoS Comp. Biol., 2006
        - PNAS, 2008, 2014
- Inference engine
    - Gibbs sampling
    - Stochastic EM
- Such models are more than within the scope of general PP software





FB5-HMM

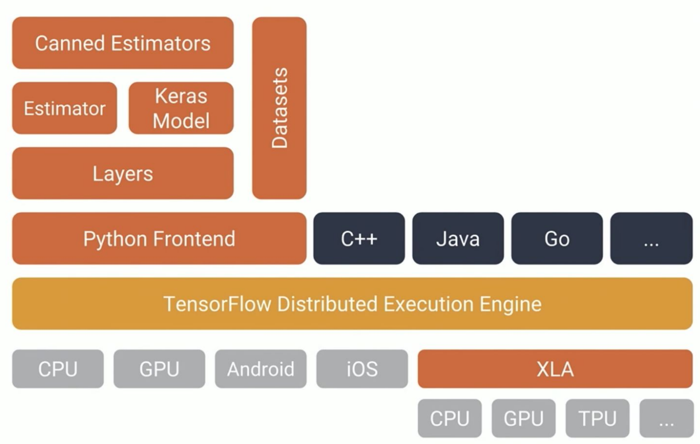


Some PP packages and their roots

- STAN (2011)
    - Hamiltonian Monte Carlo
    - Columbia University
- pyMC3
    - Academic, Quantopian
    - Theano (U. Montréal)
- Edward
    - Google/Tensorflow (Google)
- Pyro
    - Uber/PyTorch (Facebook)
- …

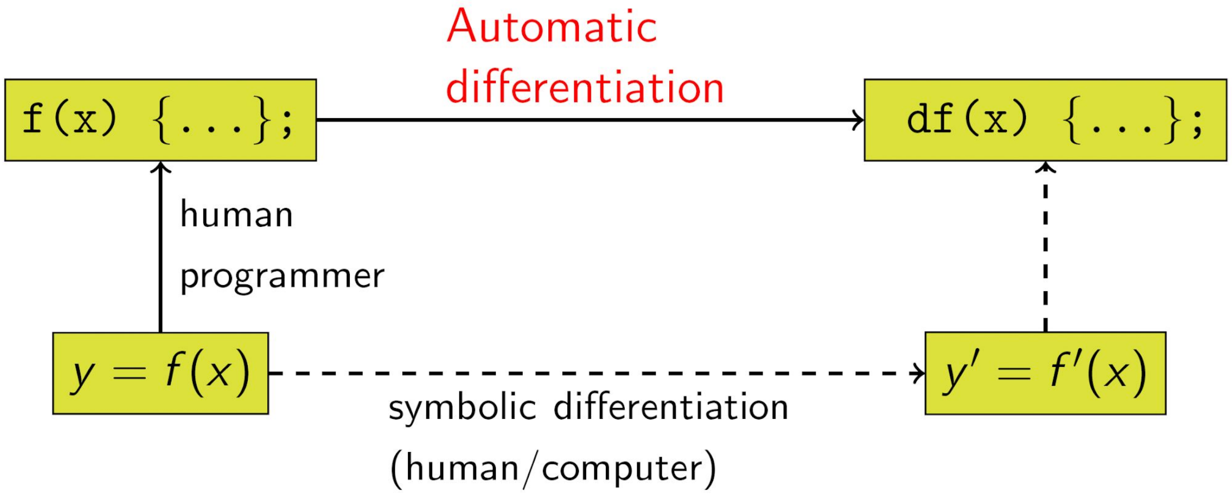
Theano, PyTorch & Tensorflow

- Theano (U. Montréal)
  - Discontinued
- Tensorflow (Google)
  - Python API based on Numpy
- PyTorch (Facebook)
- Tools for machine learning
- Similar scope, interface and goal
  - Mathematical computing
  - Automatic differentiation
  - GPU support



Deduction - Automatic differentiation

- The key development that makes probabilistic programming possible
  - Augment the algebra of real numbers and obtain a new arithmetic
- Not symbolic differentiation, nor numerical differentiation
  - Large expressions/round off errors



The Bayesian calculus

- For inference and abduction we need a calculus of uncertainty
- This is provided by Bayesian statistics
  - Thomas Bayes (1701-1761)
  - Pierre-Simon Laplace (1749-1827)
- Probability is a measure of belief
  - Alternatively, probability can be seen as a frequency of occurrence
  - Predominant until end of 20th c.
- Core idea: prior belief is updated in the light of new data.

The Bayesian calculus

- For inference and abduction we need a calculus of uncertainty
- This is provided by Bayesian statistics
  - ~~Thomas Bayes (1701-1761)~~
  - Pierre-Simon Laplace (1749-1827)
- Probability is a measure of belief
  - Alternatively, probability can be seen as a frequency of occurrence
  - Predominant until end of 20th c.
- Core idea: prior belief is updated in the light of new data.

The Bayesian calculus

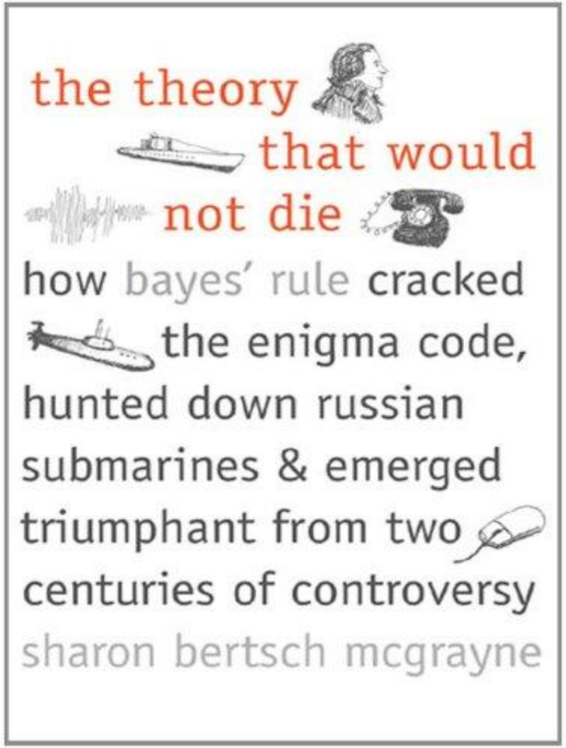
- For inference and abduction we need a calculus of uncertainty
- This is provided by Bayesian statistics
  - ~~Thomas Bayes (1701-1761)~~
  - Pierre-Simon Laplace (1749-1827)
- Probability is a measure of belief
  - Alternatively, probability can be seen as a frequency of occurrence
  - Predominant until end of 20th c.
- Core idea: prior belief is updated in the light of new data.

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}}$$

$$p(\boldsymbol{\theta} \mid \mathbf{d}) = \frac{p(\mathbf{d} \mid \boldsymbol{\theta})\pi(\boldsymbol{\theta})}{p(\mathbf{d})}$$

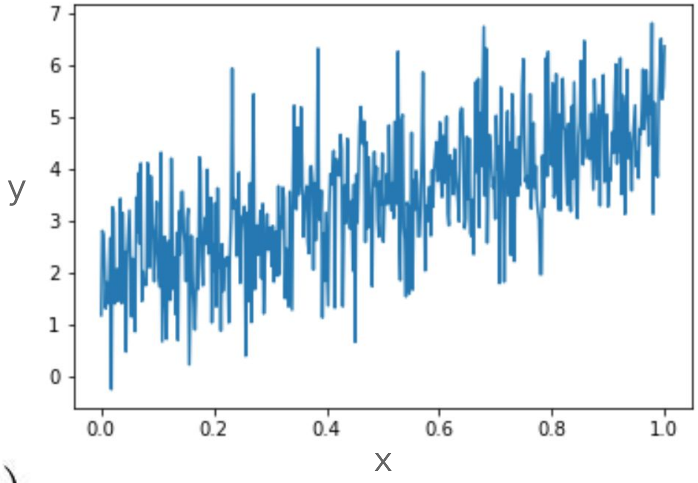
The theory that would not die

- Frequentist methods reigned supreme until the end of the 20th century
  - Ideological considerations (Fisher)
  - Analytic convenience
- Due to fast computers, the Bayesian view has now largely taken over
- The Bayesian calculus is now the paradigm of choice in machine learning
  - Yarin Gal (2015): dropout as approximate Bayesian inference in deep Gaussian processes



Bayesian linear model in pyMC3

- A simple linear model
- Data set of (*x*,*y*) values
- Parameters
  - a=2, b=3, σ=1
- Bayesian inference using sampling
- Priors/Likelihood



$$a \sim N(a \mid 0, 1)$$

$$b \sim N(b \mid 0, 1)$$

$$\sigma \sim N_{+}(0, 1)$$

$$y \sim N(y \mid \mu, \sigma)$$

$$\mu = a + bx$$

Inference by sampling

- Direct calculation of the posterior distribution is typically intractable.
- Therefore, the posterior is typically approximated by sampling. We need:
  - A starting point
  - A proposal distribution *q(x'|x)*
  - A an acceptance/rejection criterion
- Fast computers led to the resurrection of Bayesian methods in the 20th century.

$$\alpha = p(x \to x') = \min\left(1, \frac{p(x')q(x|x')}{p(x)q(x'|x)}\right)$$

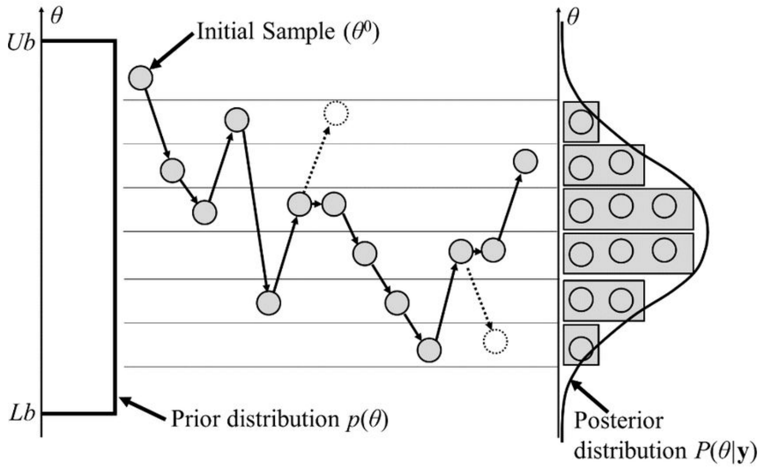


Illustration of Markov chain Monte Carlo sampling (Metropolis & Hastings 1953).

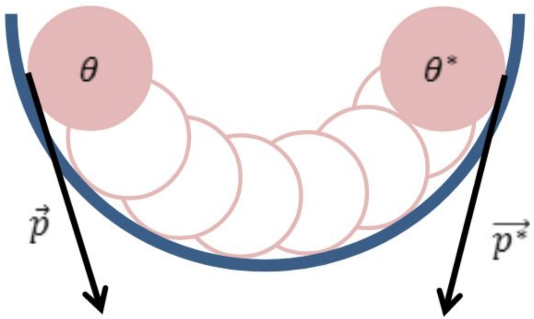
Picture: Lee, Sung & Choi, 2015

Hamiltonian Monte Carlo

- Proposal from molecular dynamics
  - Accept/reject as before
- Physics: position θ
  - Momentum p
  - Potential energy $E_{pot}(\theta)$
  - Kinetic energy $E_{kin}(p)$
- Statistics: parameters θ
  - Auxiliary momentum p
  - $E_{pot}(\theta) = -\log p(\theta|d)$
  - $E_{kin}(p) \sim N(0,1)$

$$\frac{\partial \theta}{\partial t} = \frac{\partial E_{kin}}{\partial p} = \frac{p}{m}$$

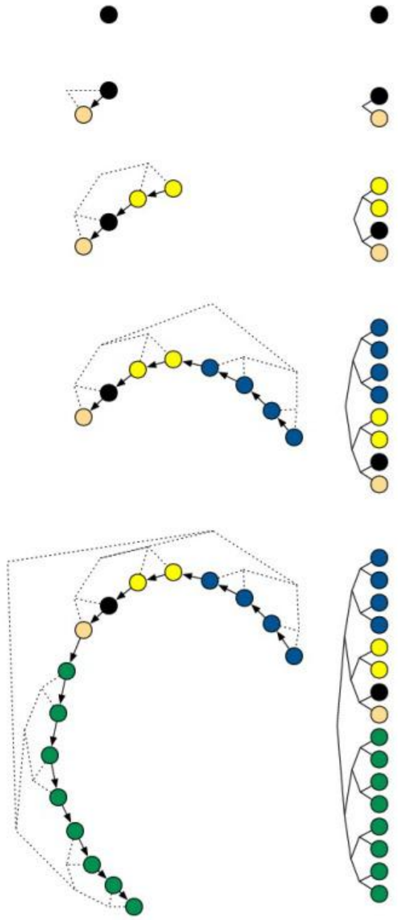
$$\frac{\partial p}{\partial t} = -\frac{\partial E_{pot}}{\partial \theta}$$



Pictures: Mathieu Lê

Sampling goes NUTS

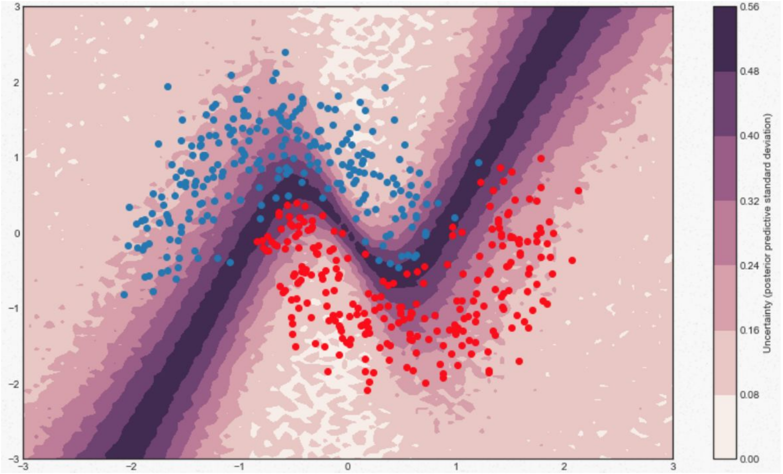
- Hamiltonian MC is difficult to automate due to two hyperparameters needed for integration with the Leapfrog algorithm
  - Number of steps L
  - Step size ε
- This was fully automated in 2011 by Hoffman & Gelman
  - No U-turn Sampling (NUTS)
  - Do 2^i leapfrog steps for step i
  - Choose random forward or backward direction in time at each step
  - Stop when particle retraces its steps (U-turn)



Bayesian deep learning

- Deep learning
  - + Fast enough for large datasets
  - - Point estimates, uncertainty
  - - Overfitting
- Bayesian deep learning
  - + Priors avoid overfitting
  - + Modelling of uncertainties
  - - Computational efficiency
  - - Big data

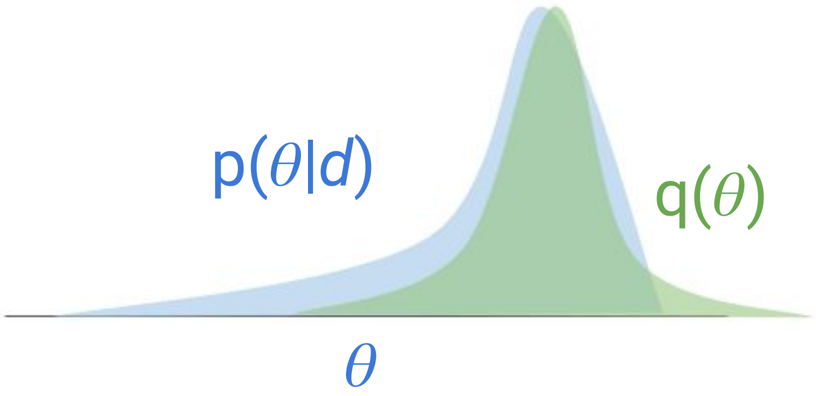
Picture: Thomas Wiecki, pyMC3



A Bayesian decision boundary of a neural network, estimated with pyMC3

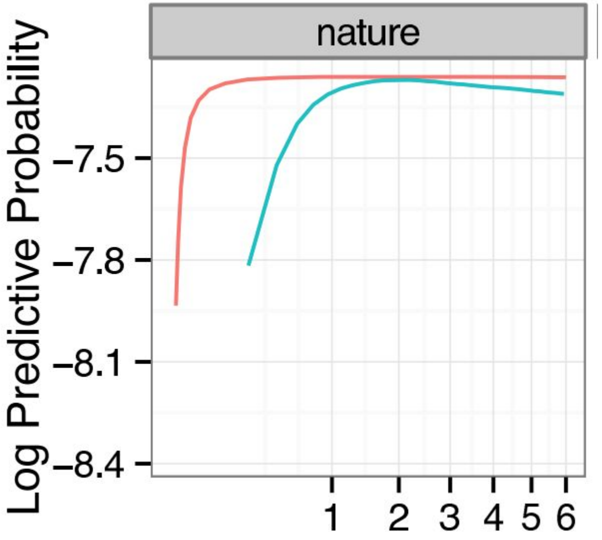
Variational Bayes to the rescue

- Sampling - even NUTS - is slow
- Sampling does not scale to massive data sets
- Variational Bayes turns inference into an optimization problem
    - Chose an approximation q(θ) of the posterior p(θ|d)
    - Find θ that minimizes the Kullback-Leibler divergence between q(θ) and p(θ|d)



ADVI and Mini-batch ADVI

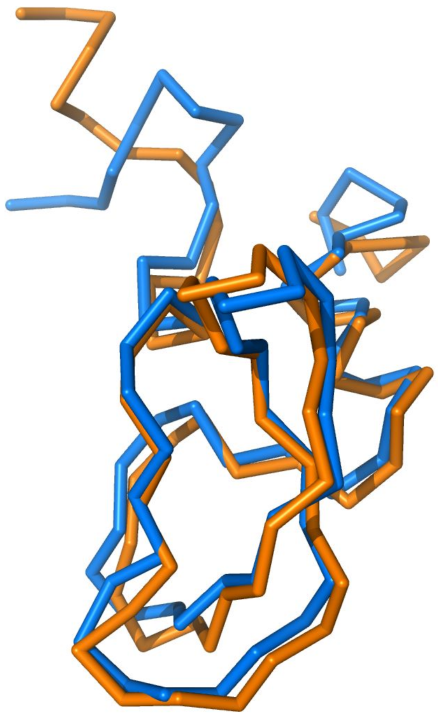
- Automatic differentiation variational inference (ADVI)
  - Automated variational Bayes
- Mini-batch ADVI
  - Train on batches of data
  - The batches are used to estimate a stochastic expectation of the gradient
  - Much faster, for large data sets
  - ...and faster convergence
- Towards Bayesian Deep Learning and Big Data



Training time (h) for classification of 300K articles from Nature (Hoffman et al, 2013). Mini-batch in red.

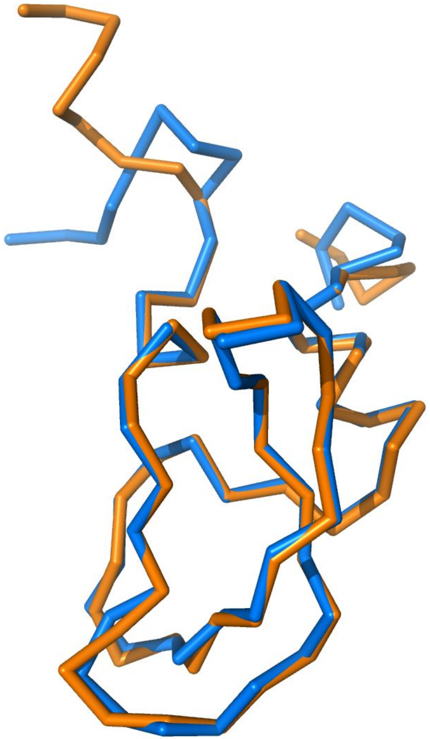
Protein structure alignment

- A classic bioinformatics application
- Normally done by minimizing the sum of the squared distances between the atoms
  - Singular value decomposition
- Alternative: a probabilistic model inspired by Douglas Theobald's THESEUS program
  - Full Bayesian posterior
  - Realistic error model based on the Matrix Normal distribution.
  - Closer to biological reality

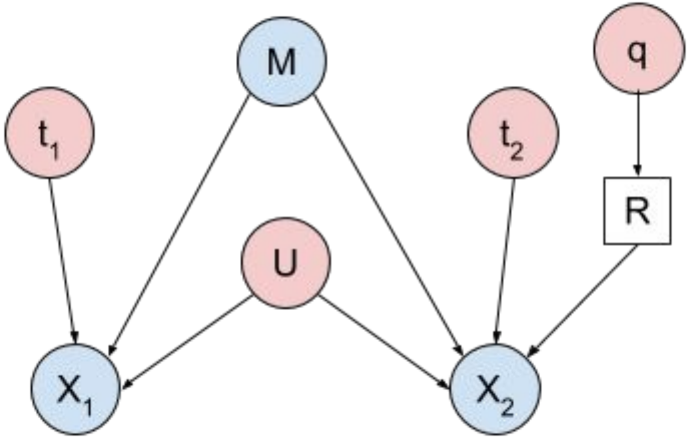


Protein structure alignment

- A classic bioinformatics application
- Normally done by minimizing the sum of the squared distances between the atoms
  - Singular value decomposition
- Alternative: a probabilistic model inspired by Douglas Theobald's THESEUS program
  - Full Bayesian posterior
  - Realistic error model based on the Matrix Normal distribution.
  - Closer to biological reality

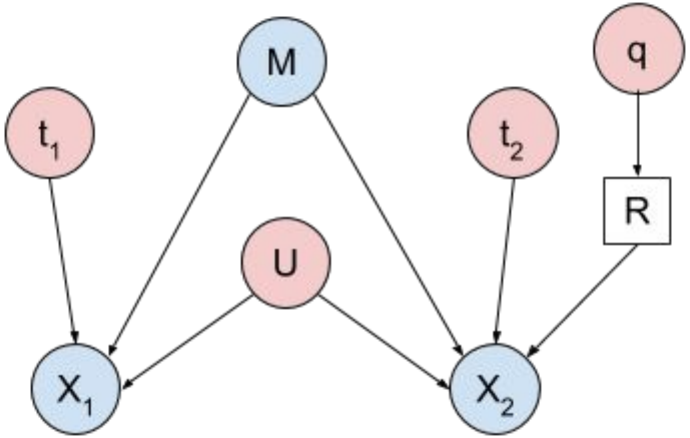


Protein structure alignment - the model



$$M \sim \text{RandomWalk}(d = 3.8, n)$$
$$M_0 \leftarrow \text{center}(M)$$
$$t_1 \sim \mathcal{N}(\mathbf{0}, I_3)$$
$$t_2 \sim \mathcal{N}(\mathbf{0}, I_3)$$
$$q \sim \text{UnitQuaternion}()$$
$$R \leftarrow \text{RotationMatrix}(q)$$
$$\sigma \sim N_{+}(0, 1)$$
$$U \leftarrow \sigma^2 I_n$$
$$V \leftarrow I_3$$
$$X_1 \sim \mathcal{MN}(M_0 + t_1, U, V)$$
$$X_2 \sim \mathcal{MN}(RM_0 + t_2, U, V)$$

Protein structure alignment - the model



No training needed - only prior knowledge!

$$
\begin{aligned}
M &\sim \text{RandomWalk}(d = 3.8, n) \\
M_0 &\leftarrow \text{center}(M) \\
t_1 &\sim \mathcal{N}(\mathbf{0}, I_3) \\
t_2 &\sim \mathcal{N}(\mathbf{0}, I_3) \\
q &\sim \text{UnitQuaternion}() \\
R &\leftarrow \text{RotationMatrix}(q) \\
\sigma &\sim N_+(0, 1) \\
U &\leftarrow \sigma^2 I_n \\
V &\leftarrow I_3 \\
X_1 &\sim \mathcal{MN}(M_0 + t_1, U, V) \\
X_2 &\sim \mathcal{MN}(RM_0 + t_2, U, V)
\end{aligned}
$$

Conclusions

- Probabilistic Programming is the next big thing after Big Data and Deep Learning
- Complex probabilistic reasoning has now become accessible and computationally affordable
  - NUTS sampling
  - Variational Bayes (ADVI)
  - Batch variational Bayes (Mini-batch ADVI)
  - This is a very active field!
- In the future, we will see the emergence of Deep Probabilistic Programming, featuring Deep Learning components combined with classic Bayesian models

Acknowledgements

InnovationsFonden
FORSKNING, TEKNOLOGI & VÆKST I DANMARK

bilagscan

- Fritz Henglein, DIKU
- Ahmad Salim, DIKU/Bilagscan
- William Bullock, Basile Rommes, BINF

DET FRIE FORSKNINGSRÅD
DANISH COUNCIL FOR INDEPENDENT RESEARCH

Workshop

Install pyMC3 (assuming Anaconda Python):

conda install pymc3

Jupyter notebook files:

git clone https://github.com/thamelry/ppl-arhus

Run anaconda-navigator, start Jupyter and open files

View static Jupyter notebook files:

https://nbviewer.jupyter.org/

Type “thamelry/ppl-arhus” in box and press Go