Bayesian methods and machine learning in cosmology

Michael Hobson
Astrophysics Group, Cavendish Laboratory, Cambridge

Science on the Sphere meeting, Chicheley Hall: 14-15 July 2014

(see Shaw, Bridges, MPH – astro-ph/0701867,
Feroz, MPH – arXiv:0704.3704
Feroz, MPH, Bridges – arXiv:0809.3437,
Feroz, MPH, Cameron, Pettitt – arXiv1306.2144,
Graff, Feroz, MPH, Lasenby – arXiv:1110.2997, arXiv:1309.0790)

Basics of Bayesian inference

Rev. Thomas Bayes (1701–1761)

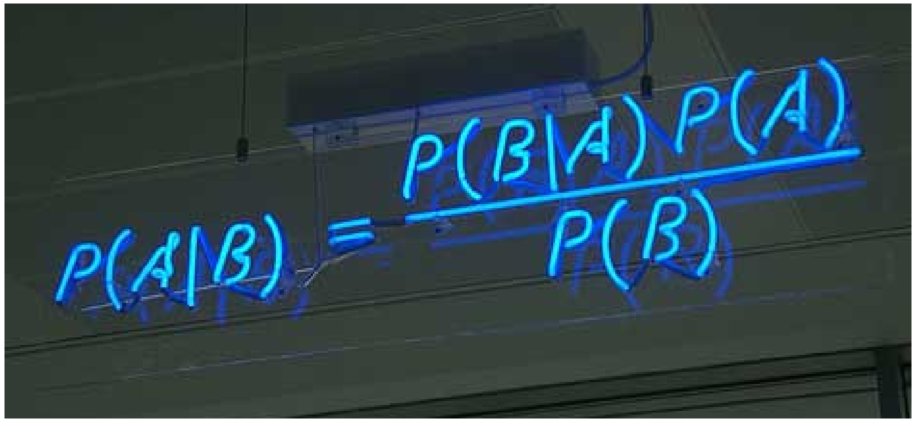
- Collect a set of N data points D_i $(i = 1, 2, \ldots, N)$, which we denote collectively as the data vector $\boldsymbol{D}$.
- Propose some model (or hypothesis) H for the data, depending on M parameters θ_j $(j = 1, \ldots, M)$, that we denote by the parameter vector $\boldsymbol{\theta}$.

- Apply Bayes' theorem

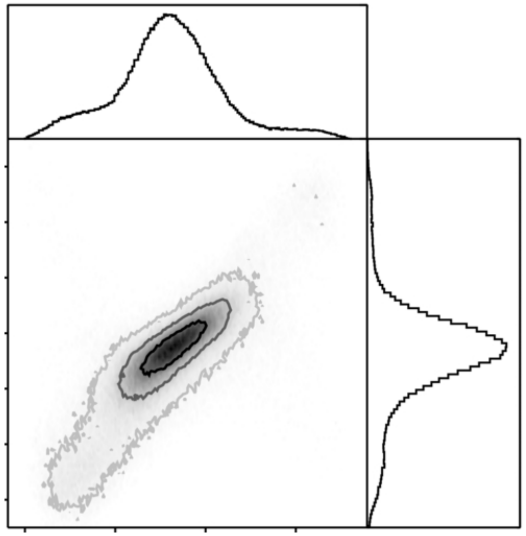
$$\Pr(\boldsymbol{\theta}|\boldsymbol{D}, H) = \frac{\Pr(\boldsymbol{D}|\boldsymbol{\theta}, H)\,\Pr(\boldsymbol{\theta}|H)}{\Pr(\boldsymbol{D}|H)} \quad \rightarrow \quad P(\boldsymbol{\theta}) = \frac{L(\boldsymbol{\theta})\pi(\boldsymbol{\theta})}{E}$$

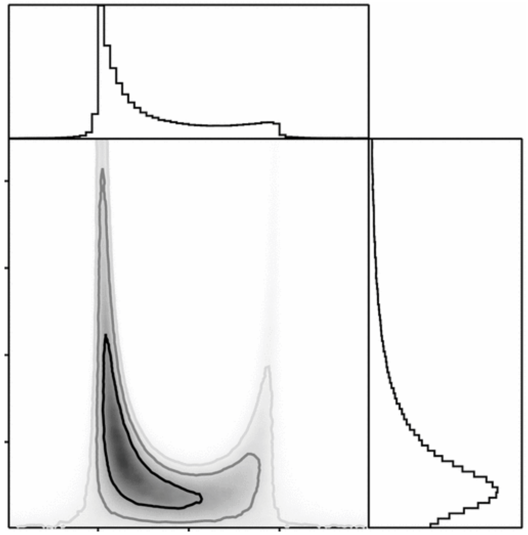
- prior $\pi(\boldsymbol{\theta}) \equiv \Pr(\boldsymbol{\theta}|H)$ represents our state of knowledge (or prejudices) of the parameter values before analysing the data
- likelihood $L(\boldsymbol{\theta}) \equiv \Pr(\boldsymbol{D}|\boldsymbol{\theta}, H)$ of the data given a particular set of parameter values, which modulates prior to give the...
- posterior $P(\boldsymbol{\theta}) \equiv \Pr(\boldsymbol{\theta}|\boldsymbol{D}, H)$ which is the result, namely the complete inference
- evidence $E \equiv \Pr(\boldsymbol{D}|H)$ provides normalisation of the posterior (and, as we see, it is much more!)

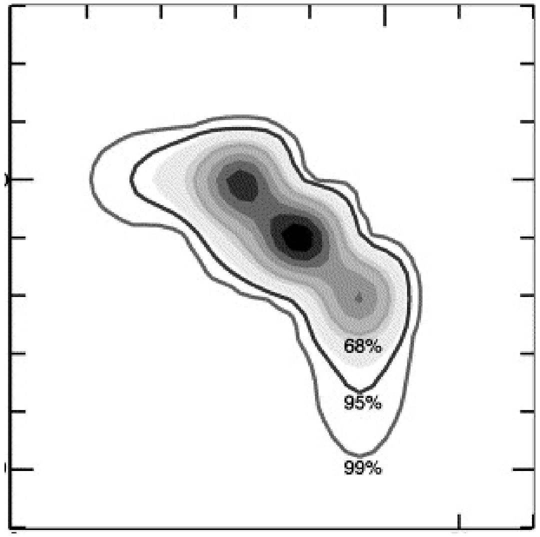
BAYESIAN PARAMETER ESTIMATION



- For parameter estimation, the posterior $P(\boldsymbol{\theta})$ is the complete inference
- Can summarise using peak(s) position(s) and covariance(s)
- Can obtain constraints on subsets of parameters by marginalisation







- Can maximise or, better, map out $P(\boldsymbol{\theta})$ (with grids or sampling)

BAYESIAN MODEL SELECTION

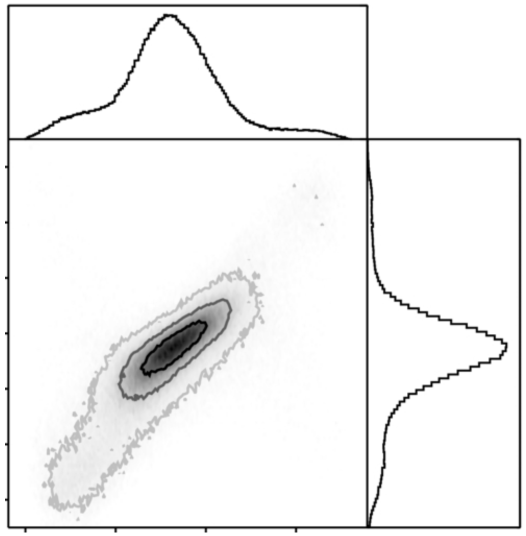
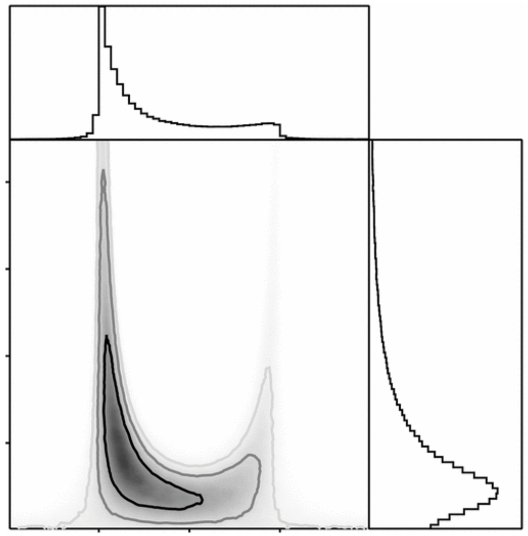
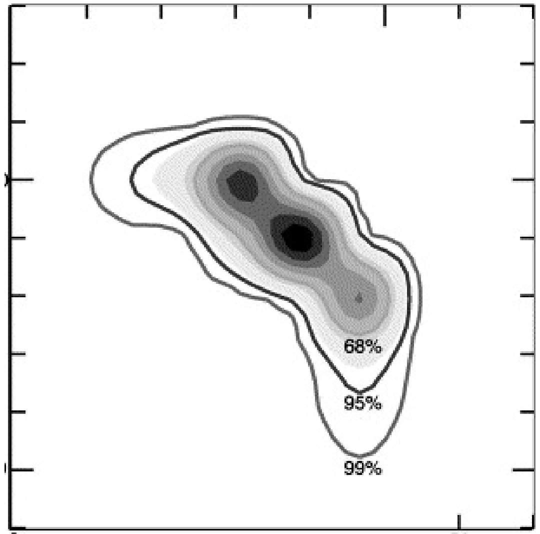
- Often wish to determine which of a set of alternative models best describes the data

- Model selection: for H_i $(i = 0, 1)$, the probability density associated with $\boldsymbol{D}$ is

$$E_i \equiv \Pr(\boldsymbol{D}|H_i) = \int L_i(\boldsymbol{\theta})\pi_i(\boldsymbol{\theta})\, d\boldsymbol{\theta}$$

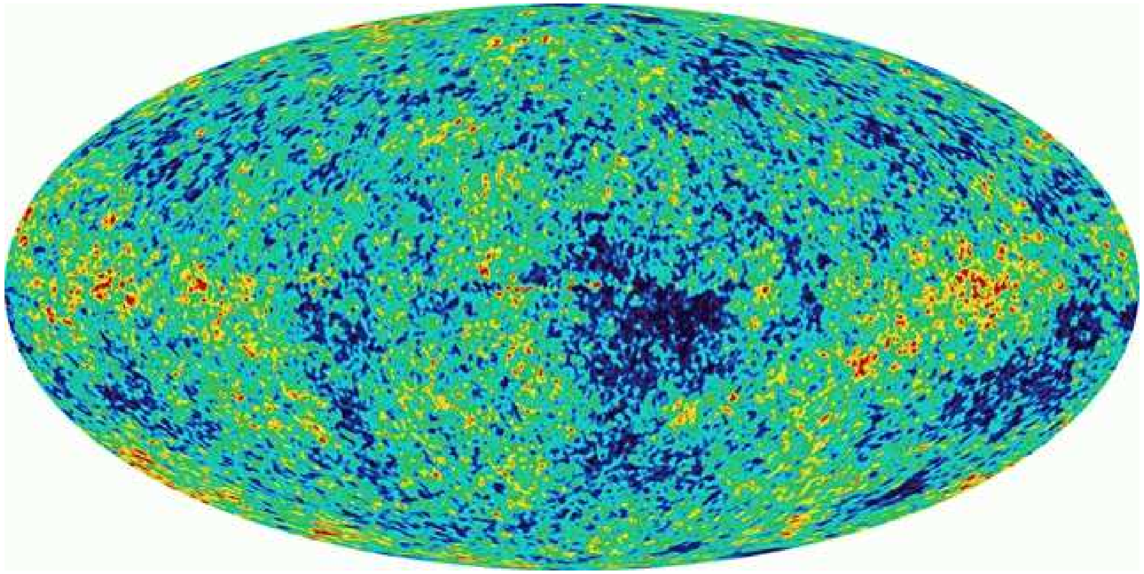
then consider ratio

$$\frac{\Pr(H_1|\boldsymbol{D})}{\Pr(H_0|\boldsymbol{D})} = \frac{E_1}{E_0}\frac{\Pr(H_1)}{\Pr(H_0)}$$

- Evidence naturally incorporates Occam's razor: a model is preferred if more of its parameter space is likely, and unfavoured if large areas in its parameter space having low likelihood values, even if the likelihood function is very highly peaked.

COSMOLOGICAL CASE-STUDY: CMB ANISOTROPIES



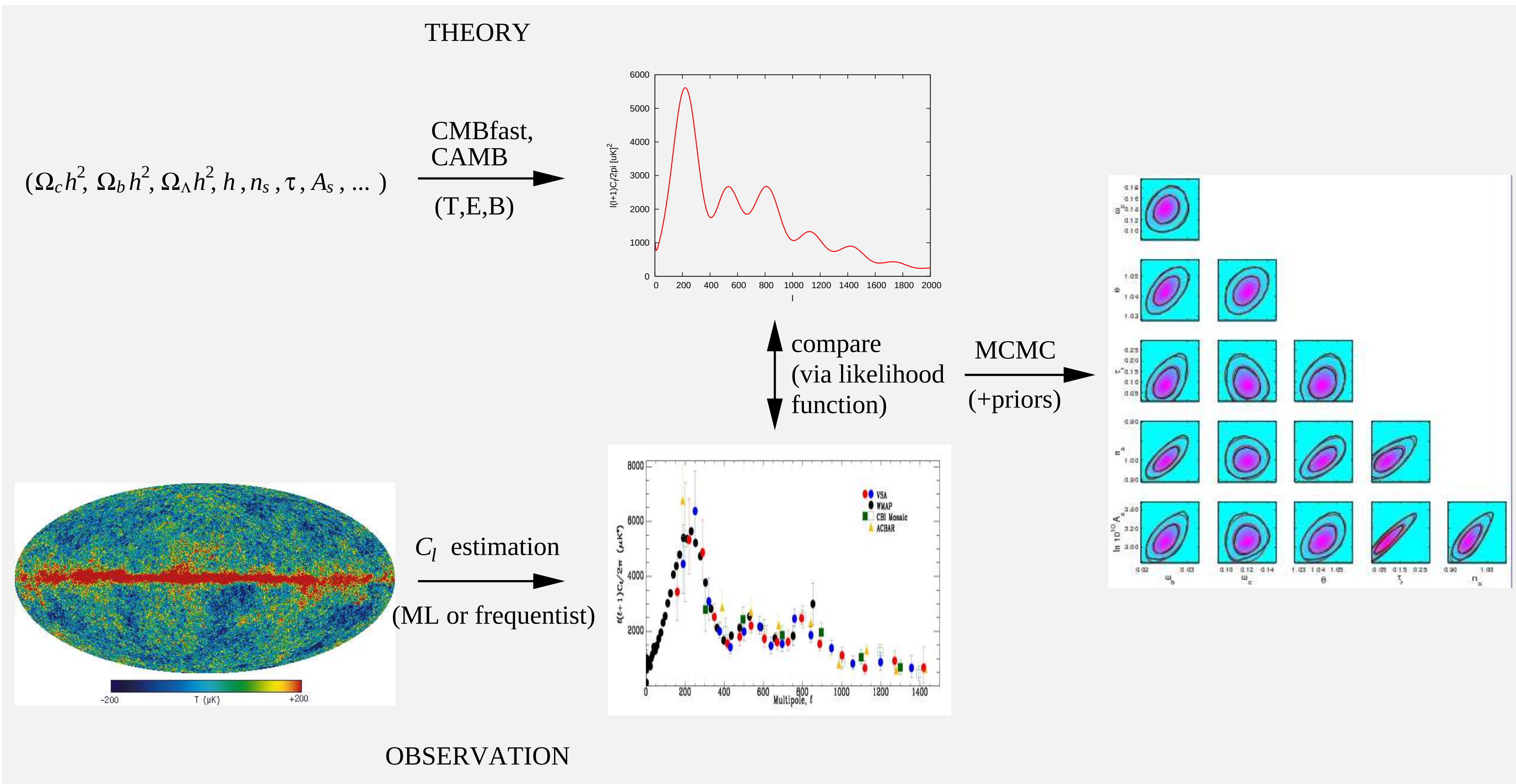
- Prior to recombination at $t \sim 300\,000$ yrs (or $z \approx 1100$) plasma and photons tightly coupled and transition to freely propagating photons occured quickly
$\Rightarrow$ CMB is snapshot of primordial density fluctuations in matter at this epoch

- These density fluctuations are of great interest for two reasons.

(i) These fluctuations later collapse under gravity to form all structure in the Universe

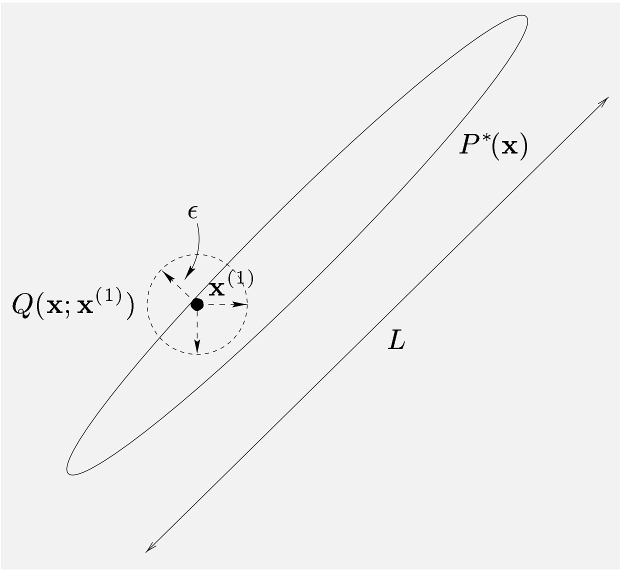
(ii) In the inflationary model, the form of these primordial density fluctuations are a powerful probe of the physics of the very early Universe

- Typical example: standard CMB data analysis pipeline



- Note parameter numbers: map ($\sim 10^7$), power spectrum ($\sim 10^3$), cosmological parameters (~ 10), cosmological models (~ 1)

METROPOLIS–HASTINGS ALGORITHM



- Metropolis–Hastings algorithm to sample $P(\boldsymbol{\theta})$:
  – start at arbitrary point $\boldsymbol{\theta}_0$
  – at each step draw trial point $\boldsymbol{\theta}' \leftarrow Q(\boldsymbol{\theta}'|\boldsymbol{\theta}_n)$ from proposal distribution
  – calculate ratio $r = P(\boldsymbol{\theta}')Q(\boldsymbol{\theta}_n|\boldsymbol{\theta}')/P(\boldsymbol{\theta}_n)Q(\boldsymbol{\theta}'|\boldsymbol{\theta}_n)$
  – if $r \geq 1$ accept $\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}'$;
  if $r < 1$ accept with probability r, else $\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n$

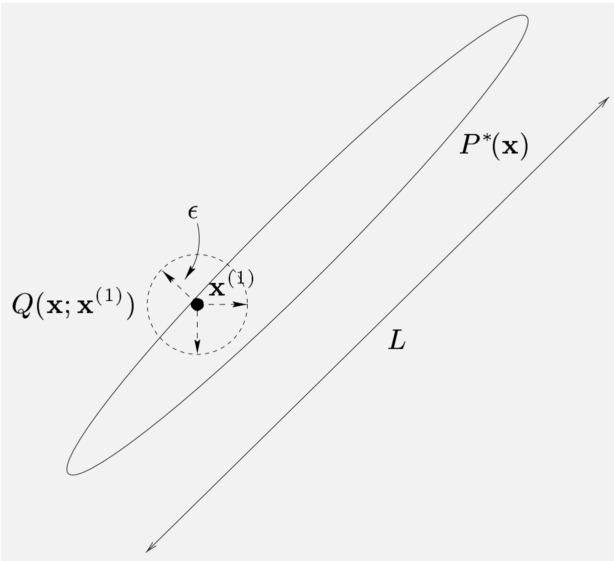
- Implementation of basic MH algorithm is trivial:

  Initialise $\boldsymbol{\theta}_0$; set $n = 0$
  Repeat [
    Sample a point $\boldsymbol{\theta}'$ from $Q(\cdot|\boldsymbol{\theta}_n)$
    Sample a uniform [0,1] random variable U
    If $U \leq \alpha(\boldsymbol{\theta}', \boldsymbol{\theta}_n)$ set $\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}'$, else $\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n$
    Increment n]

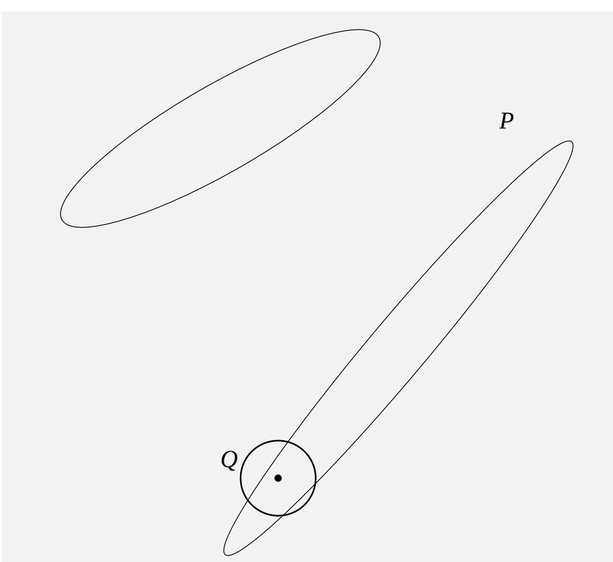
- After initial burn-in period, any (positive) proposal $Q \Rightarrow$ convergence to $P(\boldsymbol{\theta})$

- Common choice for Q is multivariate Gaussian centred on $\boldsymbol{\theta}_n$ (CosmoMC)

Metropolis–Hastings algorithm: some problems



- But... choice of Q strongly affects rate of convergence and sampling efficiency.
- Large proposal width $\epsilon \Rightarrow$ trial points rarely accepted
- Small proposal width $\epsilon \Rightarrow$ chain explores $P(\boldsymbol{\theta})$ by a random walk – very slow
- If largest scale of $P(\boldsymbol{\theta})$ is L
  $\Rightarrow$ typical diffusion time $t \sim (L/\epsilon)^2$
- If smallest scale of $P(\boldsymbol{\theta})$ is ℓ
  $\Rightarrow$ need $\epsilon \sim \ell \Rightarrow$ diffusion time $t \sim (L/\ell)^2$



- Particularly bad for multimodal distributions
- Transitions between distant modes very rare
- No choice of proposal width ϵ works
- Standard convergence tests will suggest converged, but actually only true in a subset of modes

Evaluation of the Evidence

- The evaluation of the evidence integral

$$E_i = \int L_i(\boldsymbol{\theta})\pi_i(\boldsymbol{\theta})\, d\boldsymbol{\theta}$$

presents a great numerical challenge in higher-dimensions

- Approximate/restricted methods: Gaussian approximation, Savage–Dickey ratio

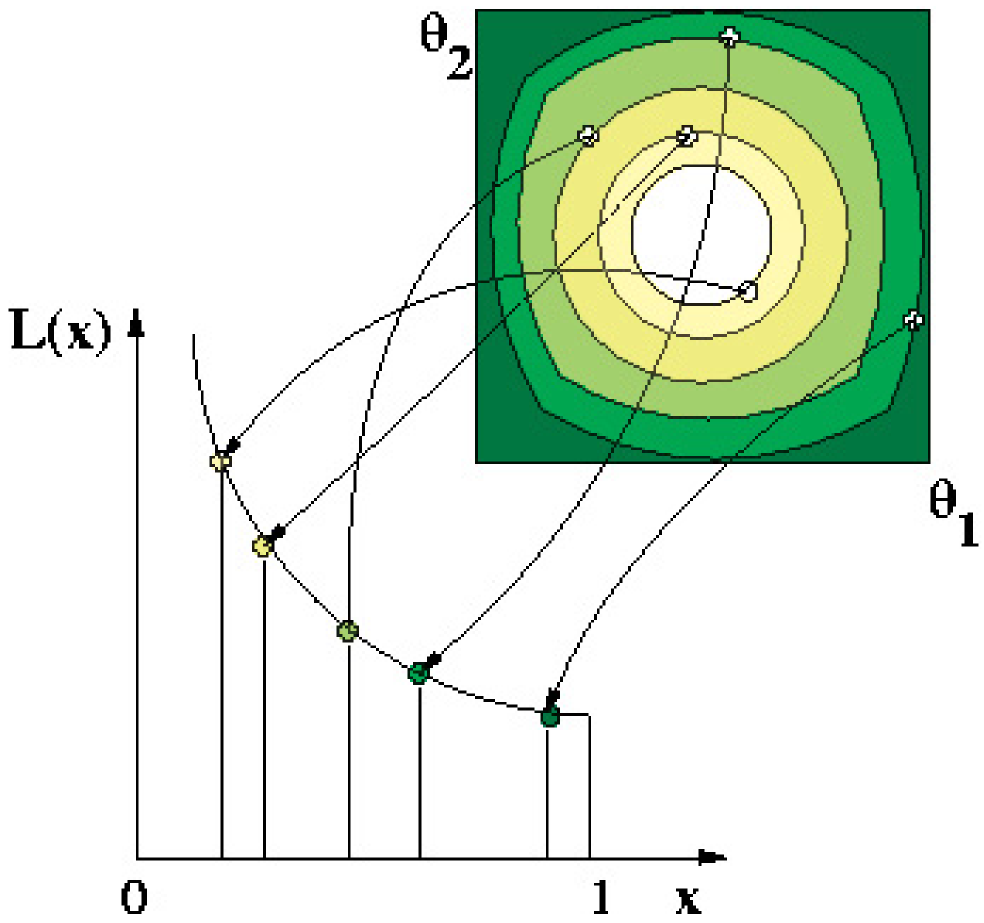
- Basic general method is thermodynamic integration: define

$$E(\lambda) = \int L^{\lambda}(\boldsymbol{\theta})\pi(\boldsymbol{\theta})\, d\boldsymbol{\theta},$$

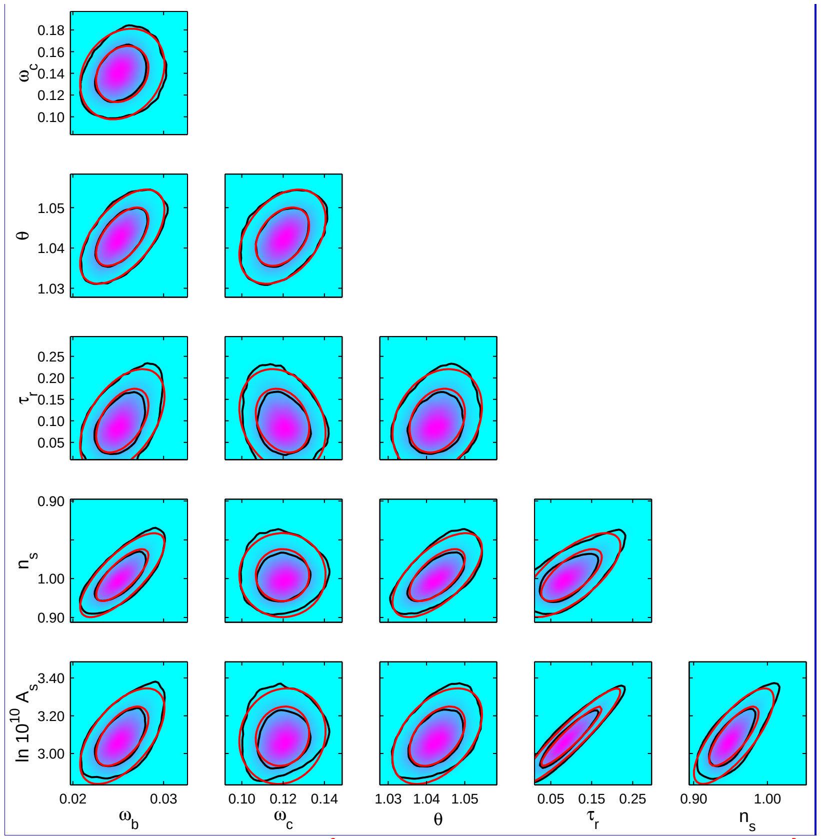
- Begin MCMC sampling from $L^{\lambda}(\boldsymbol{\theta})\pi(\boldsymbol{\theta})$, starting with $\lambda = 0$ then slowly raising the value according to some annealing schedule until $\lambda = 1$.

- BUT requires $\sim 10\times$ number of samples needed for parameter estimation

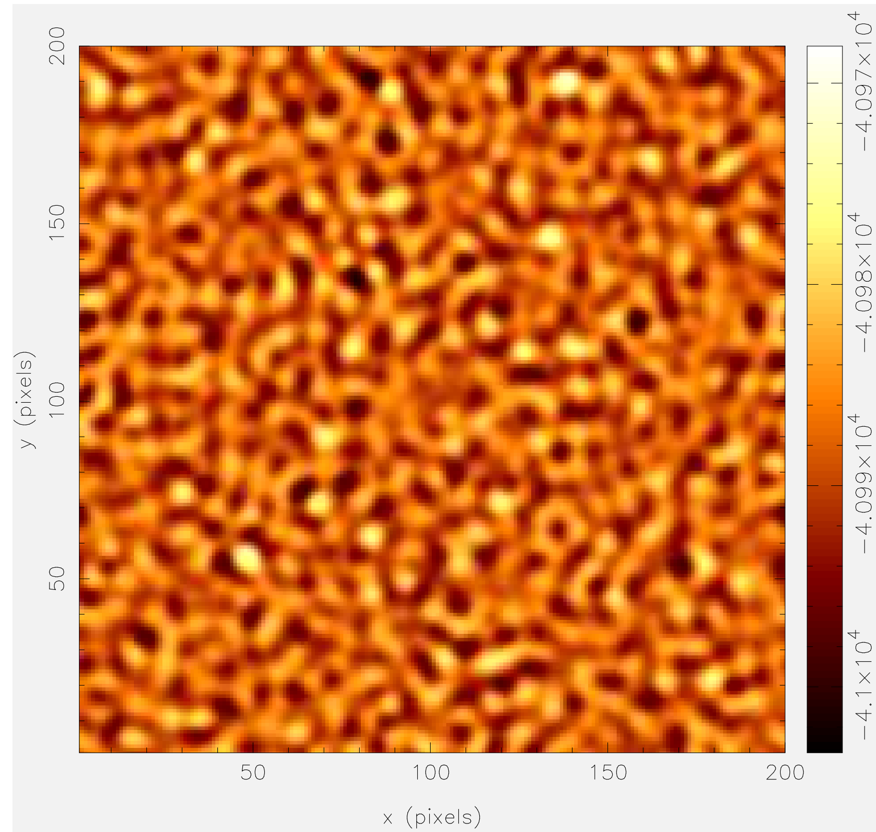
Nested sampling:
efficient parameter space exploration



- Some cosmological posteriors are nice, others are nasty



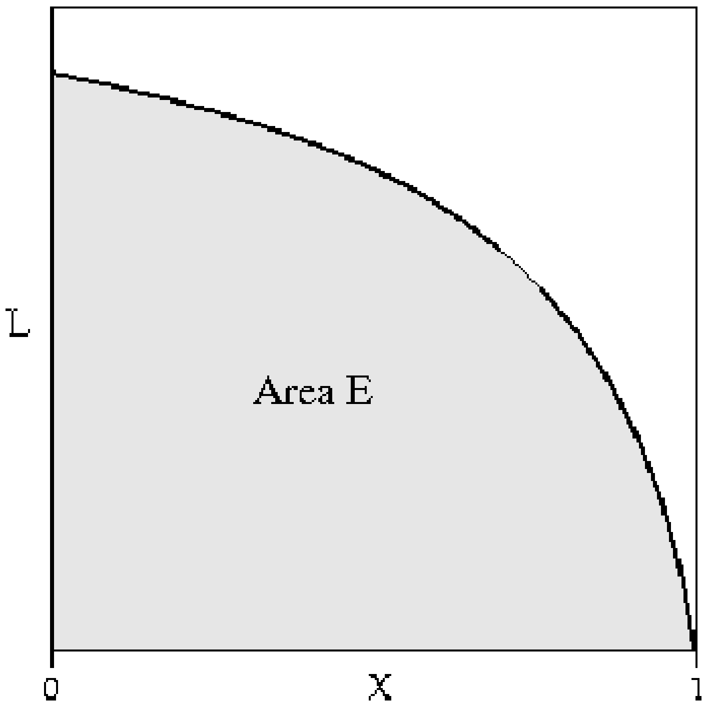
ΛCDM: $\boldsymbol{\theta} = (\omega_b, \omega_c, \theta, \tau, \ln A, n_s)$
using CMB+SDSS+HST data
(Trotta 2004)



Detecting SZ clusters in CMB:
$\boldsymbol{\theta} = (X, Y, A, R)$
(Hobson & McLachlan 2003)

- Posterior exploration (parameter estimation) and integration (model selection) traditionally performed using MCMC sampling

NESTED SAMPLING



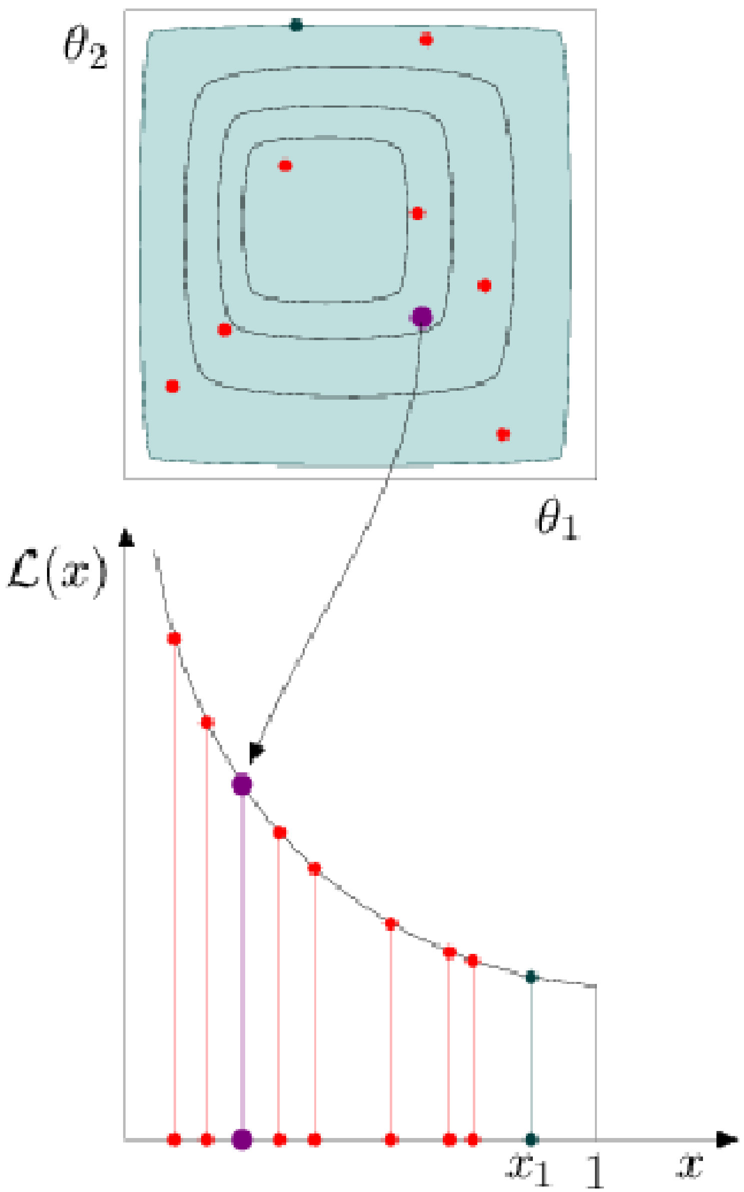
- Technique for efficient evidence evaluation (and posterior samples) (Skilling 2004)
- Define $X(\lambda) = \int_{L(\boldsymbol{\theta})>\lambda} \pi(\boldsymbol{\theta})\, d\boldsymbol{\theta}$
- Write inverse $L(X)$, i.e. $L(X(\lambda)) = \lambda$
- Evidence becomes one-dimensional integral

$$E = \int L(\boldsymbol{\theta})\pi(\boldsymbol{\theta})\, d\boldsymbol{\theta} = \int_0^1 L(X)\, dX$$

- Suppose can evaluate $L_j = L(X_j)$ where $0 < X_m < \cdots < X_2 < X_1 < 1$
  $\Rightarrow$ estimate E by any numerical method

$$E = \sum_{j=1}^{m} L_j w_j$$

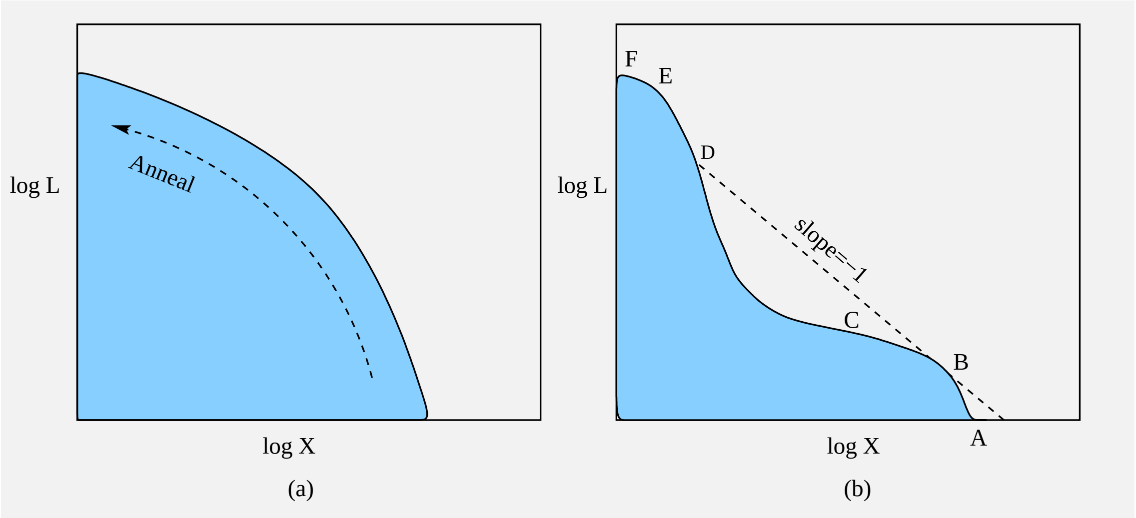
($w_j = \frac{1}{2}(X_{j-1} - X_{j+1})$ for trapezium rule)



Nested sampling approach to summation:

1. Set $i = 0$; initially $X_0 = 1$, $E = 0$
2. Sample N points $\{\boldsymbol{\theta}_j\}$ randomly from $\pi(\boldsymbol{\theta})$ and calculate their likelihoods
3. Set $i \to i + 1$
4. Find point with lowest likelihood value (L_i)
5. Remaining prior volume $X_i = t_i X_{i-1}$ where $\Pr(t_i|N) = N t_i^{N-1}$;
   or just use $\langle t_i \rangle = N/(N+1)$
6. Increment evidence $E \to E + L_i w_i$
7. Remove lowest point from active set
8. Replace with new point sampled from $\pi(\boldsymbol{\theta})$ within hard-edged region $L(\boldsymbol{\theta}) > L_i$
9. If $L_{\max} X_i < \alpha E$ (where some tolerance)
   $\Rightarrow E \to E + X_i \sum_{j=1}^{N} L(\boldsymbol{\theta}_j)/N$; stop
   else goto 3

- Advantages:
  – proceeds exponentially to high-likelihood regions ($X_i \sim e^{-i/N}$)
  – typically requires around few 100 times fewer samples than thermodynamic integration to calculate evidence to same accuracy (plus error estimate)

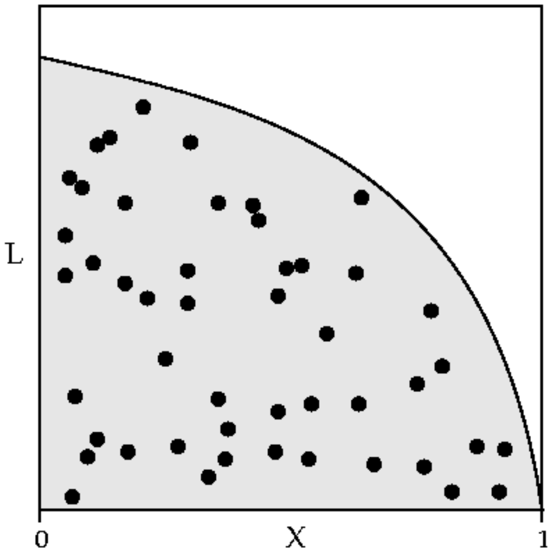


- Does not get stuck at phase changes like thermo. int.
- As $\lambda : 0 \rightarrow 1$ annealing should track along curve
- But $\frac{d \log L}{d \log X} = -\frac{1}{\lambda}$, so annealing schedule cannot navigate convex regions (phase changes)

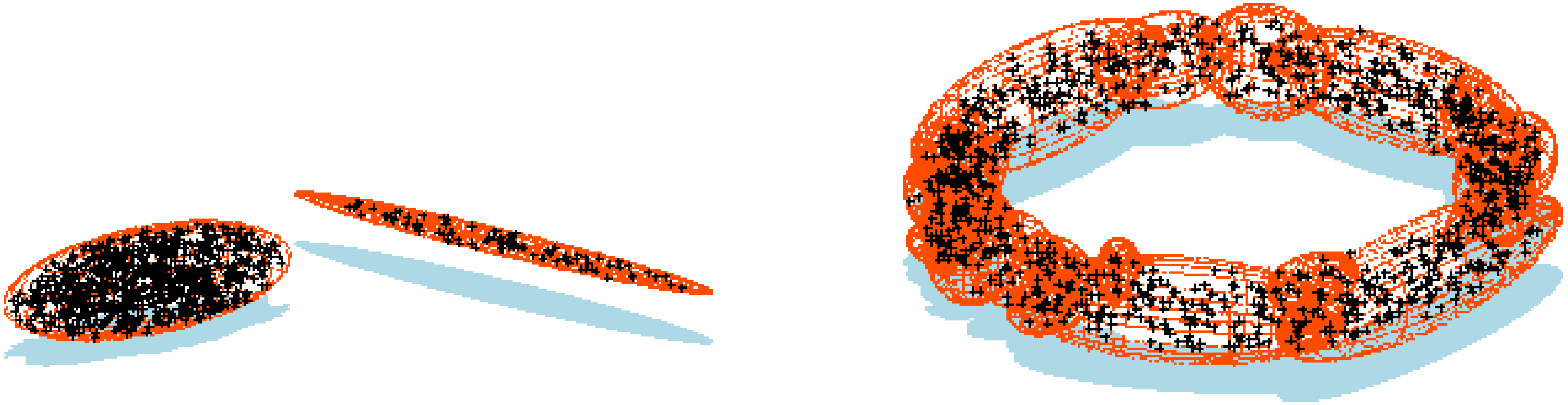
- Bonus: posterior samples easily obtained as a by-product. Simply take full sequence of sampled points $\boldsymbol{\theta}_j$ and weight jth sample by $p_j = L_j w_j / E$, e.g.

$$\mu_Q = \sum_j p_j Q(\boldsymbol{\theta}_j),$$

$$\sigma_Q^2 = \sum_j (p_j Q(\boldsymbol{\theta}_j) - \mu_Q)^2$$



MultiNest



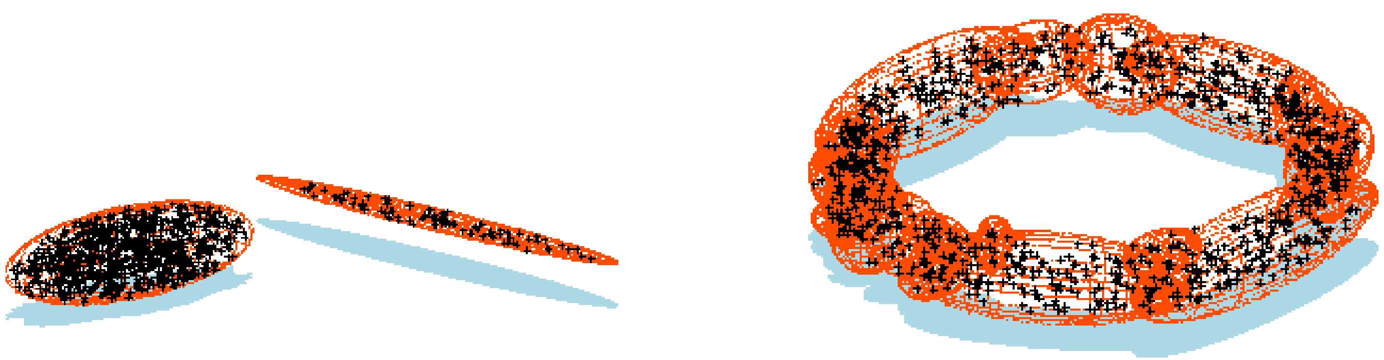
MULTIMODAL NESTED SAMPLING – MULTINEST

- Most challenging task: at each iteration i we must replace the point removed with one sampled from $\pi(\boldsymbol{\theta})$ within the complicated, hard-edged region $L(\boldsymbol{\theta}) > L_i$ for (possibly) degenerate and/or multimodal posteriors

- Could use MCMC, but typically very inefficient
  $\Rightarrow$ use analytic rejection sampling from within tailored bound to $L(\boldsymbol{\theta}) = L_i$ surface

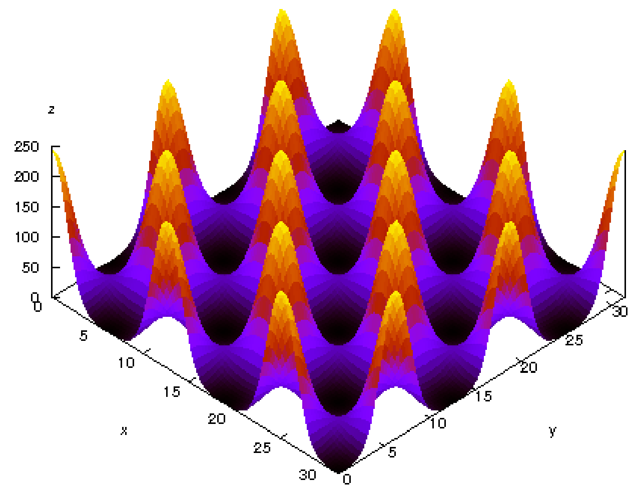
- MULTINEST – at each nested sampling iteration i:

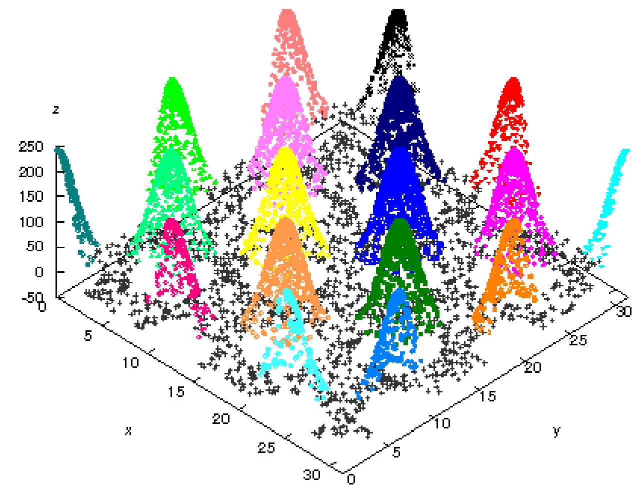
– construct optimal multi-ellipsoid bound for live points (variable ellipsoid number),
– maintains total volume exceeding expected prior volume
– determine ellipsoid overlaps using cheap exact algorithm (Alfano et al. 2003)
– pick ellipsoid randomly and sample new point with $L > L_i$, accounting for overlaps



- MULTINEST now widely used in astronomy and particle physics ($\sim$ 350 projects)

IDENTIFICATION OF POSTERIOR MODES



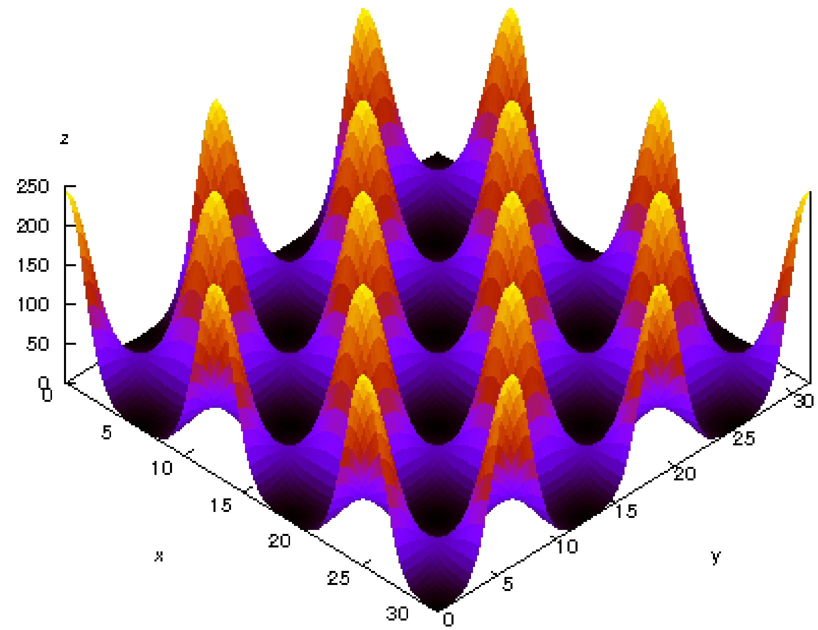


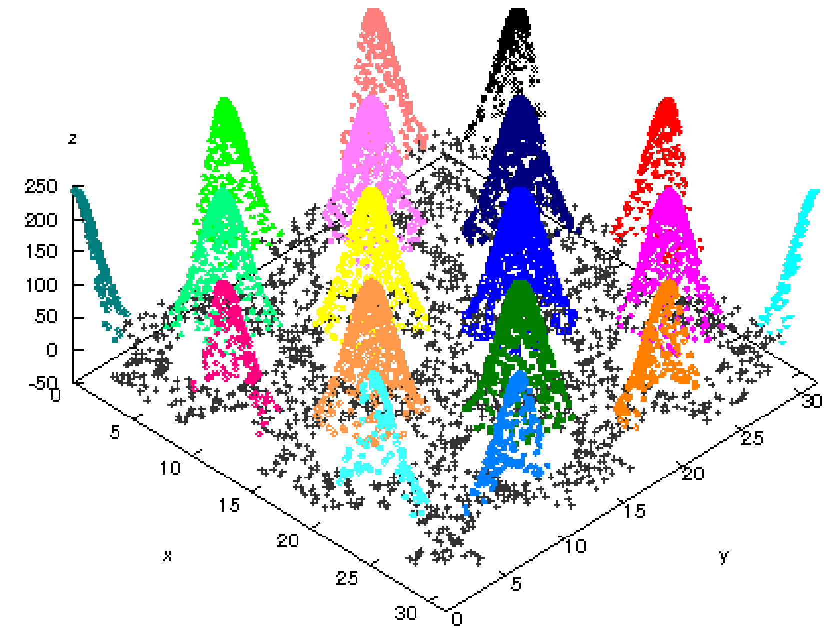
- For multimodal posteriors, useful to identify which samples 'belong' to which mode
  ⇒ automatic mode identification algorithm

- For well-defined 'isolated' modes:
  – can make reasonable estimate of posterior mass each contains ('local' evidence)
  – can construct posterior parameter constraints associated with each mode

- Partitioning and ellipsoids construction algorithm described above provides efficient and reliable method for performing mode identification
  ⇒ 'local' evidence and parameter constraints for each isolated mode
  ⇒ sum of local evidences equals 'global' evidence

TOY PROBLEM: EGG-BOX LIKELIHOOD





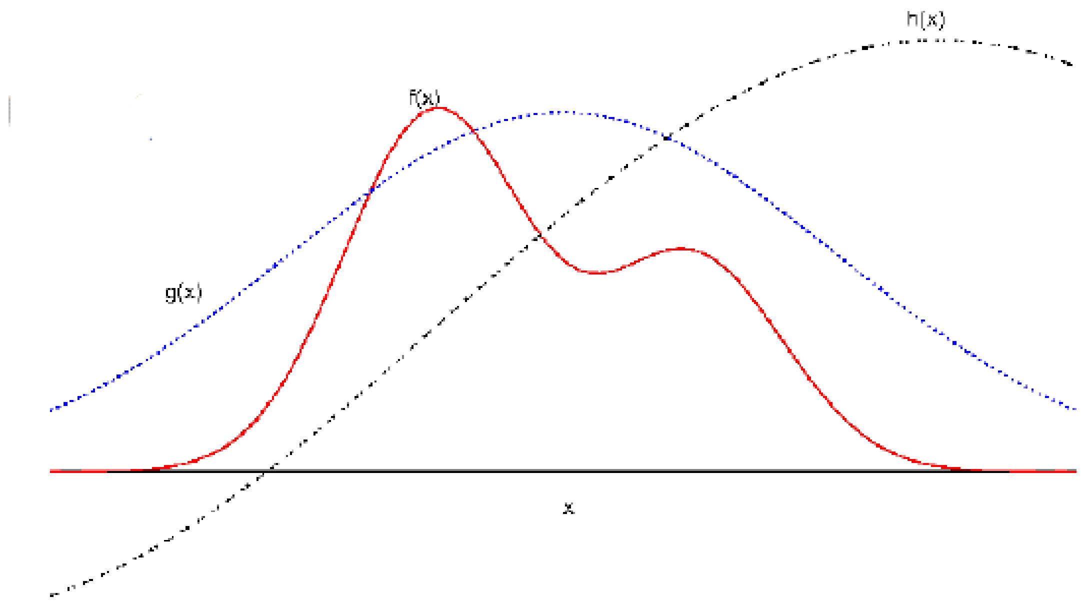
- Likelihood resembles egg-box and is given by

$$\mathcal{L}(\theta_1, \theta_2) = \exp\left[2 + \cos\left(\frac{\theta_1}{2}\right)\cos\left(\frac{\theta_2}{2}\right)\right]^5,$$

  and prior is $\mathcal{U}(0, 10\pi)$ for both θ_1 and θ_2.

- Use 2000 active points $\Rightarrow \sim 30,000$ likelihood evaluations to obtain $\log \mathcal{Z} = 235.86 \pm 0.06$ (analytical $\log \mathcal{Z} = 235.88$) [See Demo]

- Generic problem: estimate $\langle h(x) \rangle$ under $f(x)$
- If one can (easily) generate samples x_i from $f(x)$, then $\langle h \rangle = \frac{1}{N} \sum_i h(x_i)$



- If not, then one can try importance sampling:
  – generate samples from $g(x)$ and define weights $w_i = f(x_i)/g(x_i)$
  – then $\langle h \rangle = \sum_i w_i h(x_i) / \sum_i w_i$

- Apply importance sampling idea to calculation of the evidence

- Uses all the points sampled by MultiNest

- Calculate evidence as

$$Z = \frac{1}{N} \sum_{j=1}^{N} \frac{L(\boldsymbol{\theta}_j)\pi(\boldsymbol{\theta}_j)}{g(\boldsymbol{\theta}_j)}$$

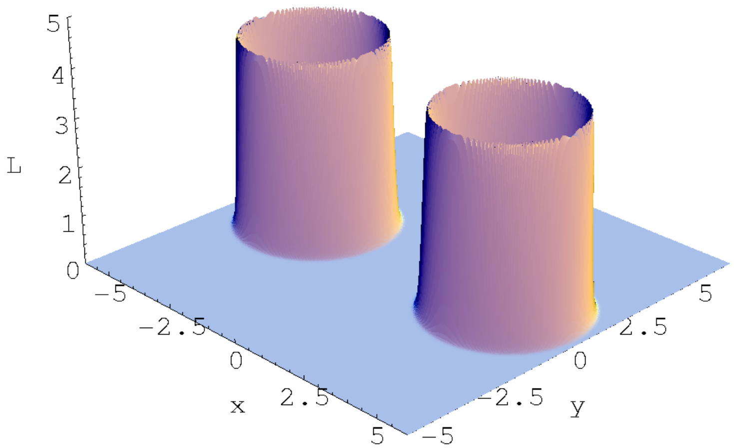
where $N = \sum_{i=1}^{n_{\text{iter}}} n_i$ is total number of points sampled and $g(\boldsymbol{\theta})$ is the importance sampling function given by

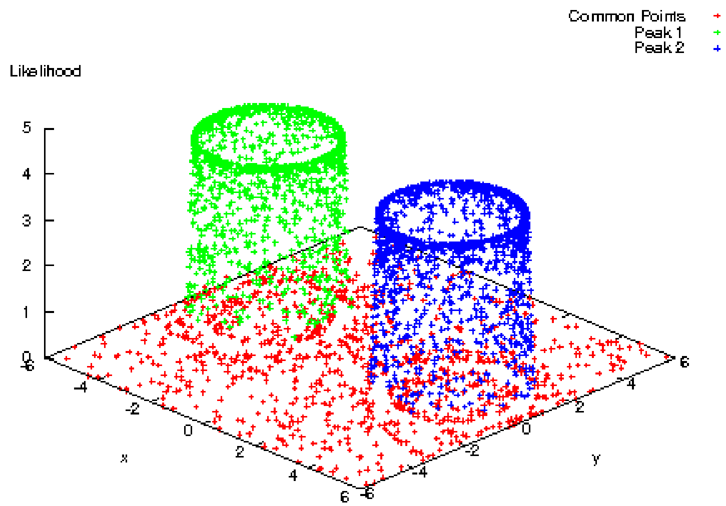
$$g(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{n_{\text{iter}}} \frac{n_i E_i(\boldsymbol{\theta})}{V_i}$$

with V_i = volume enclosed by union of ellipsoids at ith iteration and

$$E_i(\boldsymbol{\theta}) = \begin{cases} 1 & \text{if } \boldsymbol{\theta} \text{ lies in the union of ellipsoids} \\ 0 & \text{otherwise} \end{cases}$$

Application of INS to Gaussian shells



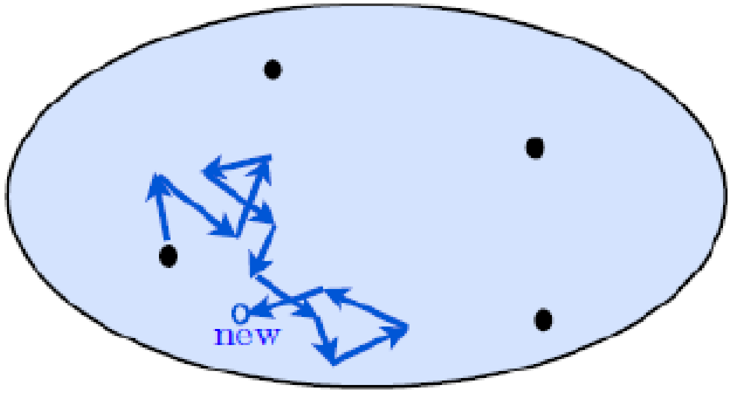


		Analytical	MultiNest without INS	MultiNest with INS
D	N_{live}	$\log Z$	$\log Z$	log Z
2	300	-1.75	-1.61 ± 0.09	-1.72 ± 0.02
10	300	-14.59	-14.55 ± 0.23	-14.60 ± 0.03
20	300	-36.09	-35.90 ± 0.35	-36.11 ± 0.03
30	500	-60.13	-59.72 ± 0.36	-60.09 ± 0.02
50	500	-112.42	-110.69 ± 0.47	-112.37 ± 0.02

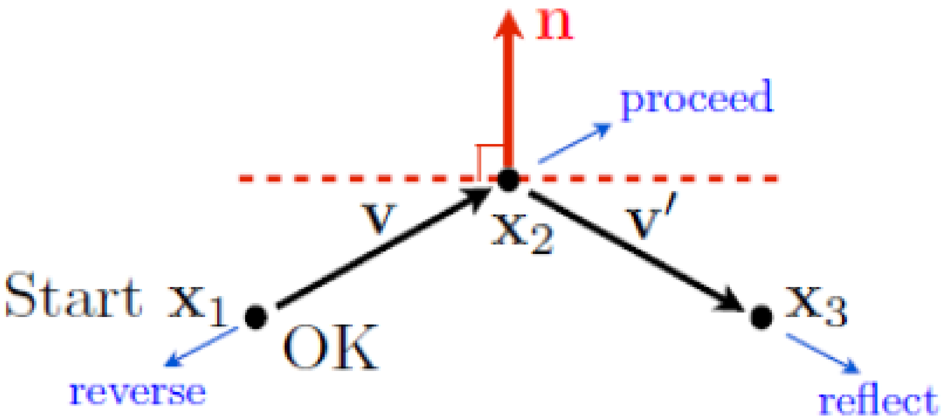
ADVANTAGES OF INS OVER VANILLA NESTED SAMPLING

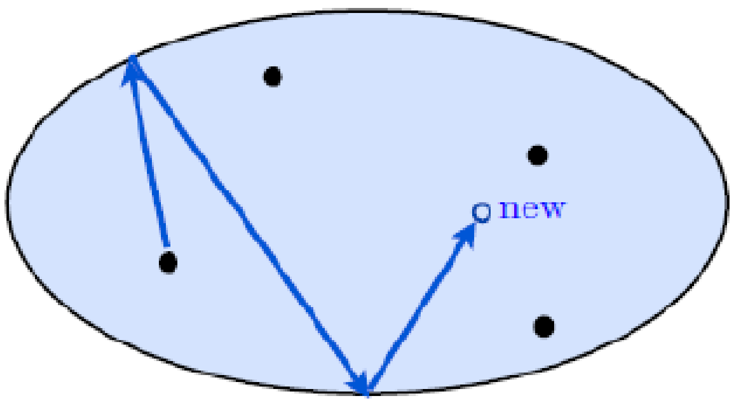
- No change in the way MULTINEST explores the parameter space
- Every sampled point contributes to the evidence calculation (no waste)
- Evidences are an order of magnitude more accurate than vanilla nested sampling
- Evidence calculation not dependent on expected prior volumes
  – mitigates mismatches between iso-likelihood contour and multi-ellipsoid bound
  – obtain accurate evidences even in MULTINEST constant efficiency mode

Nested Sampling needs to generate a new point from constrained prior



Generating new point using MCMC
Problem: Diffusion time very long



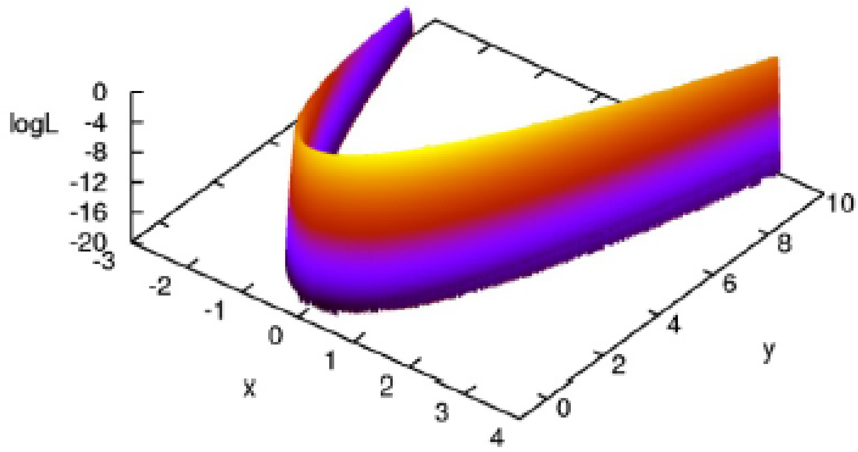


Generating new point using Hamiltonian Monte Carlo
(Reflective Slice Sampling of Radford Neal)
Problem: Don't know the edge of constrained prior

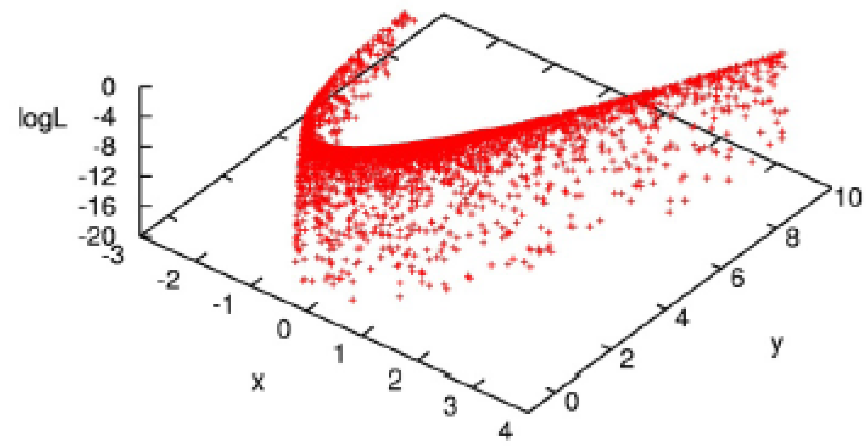
1. Start at x_1 where $L(x_1) > L'$
2. Propose $x_2 = x_1 + v$
3. If $(L(x_2) > L')$
   Proceed to x_2
   Else
   Reflect to x_3
   If $(L(x_3) > L')$
   Proceed to x_3
   Else
   Reverse

Galilean Monte-Carlo: application to Rosenbrock function

- n-D Rosenbrock function: $f(\boldsymbol{\theta}) = -\sum_{j=1}^{n}[(1-\theta_i)^2 + 100(\theta_{i+1}-\theta_i^2)^2]$



Rosenbrock Function



GMC Reconstruction of Rosenbrock Function

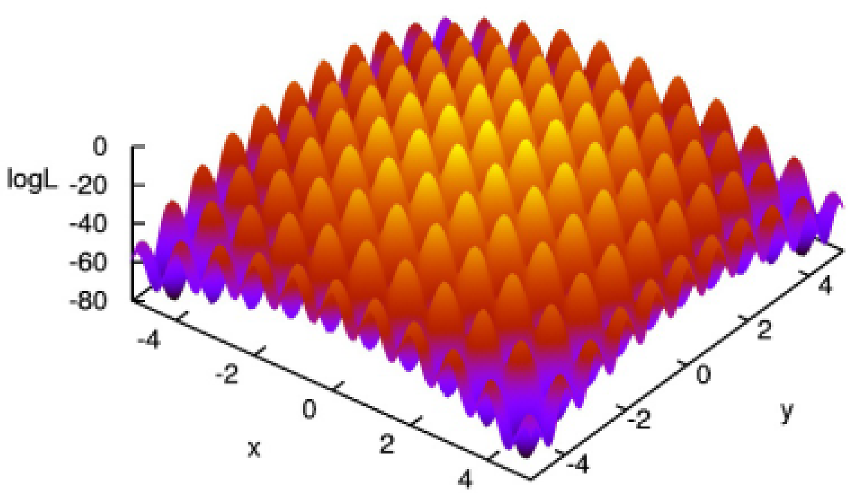
- Global maximum at $(\theta_1, \theta_2, \ldots, \theta_n) = (-1, 1, \ldots, 1)$

- Thin curving degeneracy $\Rightarrow$ finding global maximum very challenging

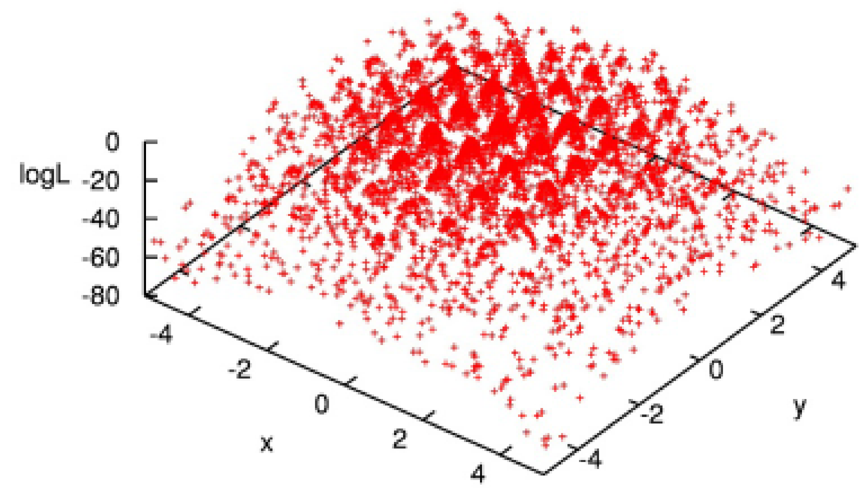
- Works well even for $n > 100$: $Z_{\text{true}} = -5.80$, $Z_{\text{GMC}} = -5.76 \pm 0.05$

GALILEAN MONTE-CARLO: APPLICATION TO RASTRIGIN FUNCTION

- n-D Rastrigin function: $f(\boldsymbol{\theta}) = -10n - \sum_{j=1}^{n}[\theta_i^2 - 10\cos(2\pi\theta_i)]$
- Highly multi-modal function with global maximum at $\theta_i = 0\ \forall i$.
- Again works well even for $n > 100$

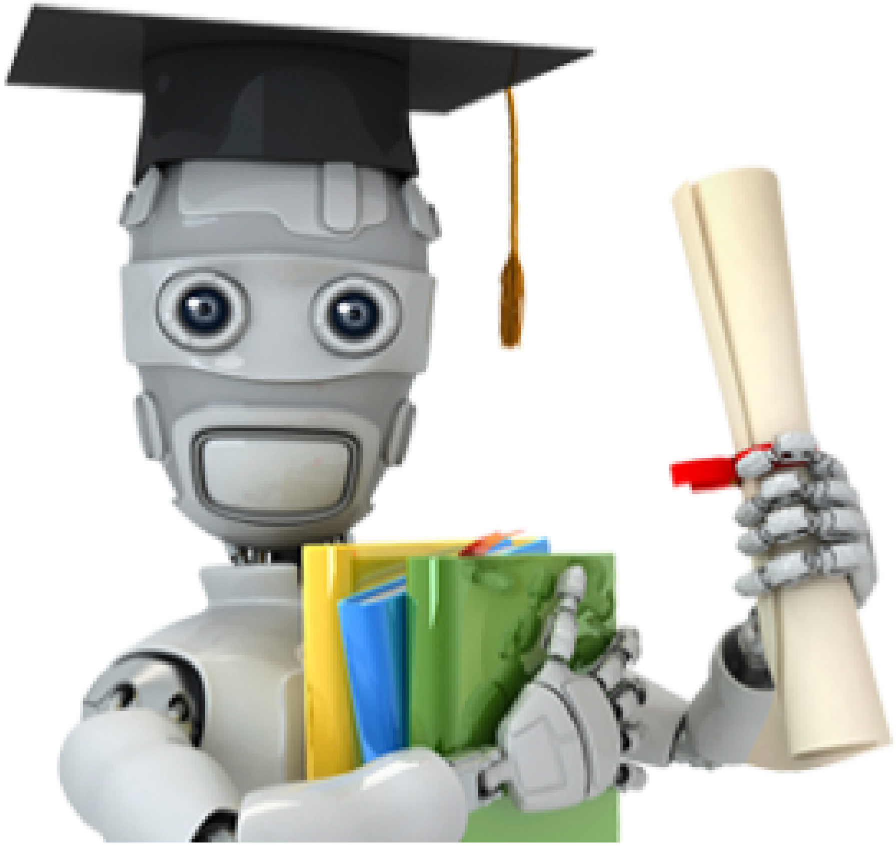


Rastrigin Function

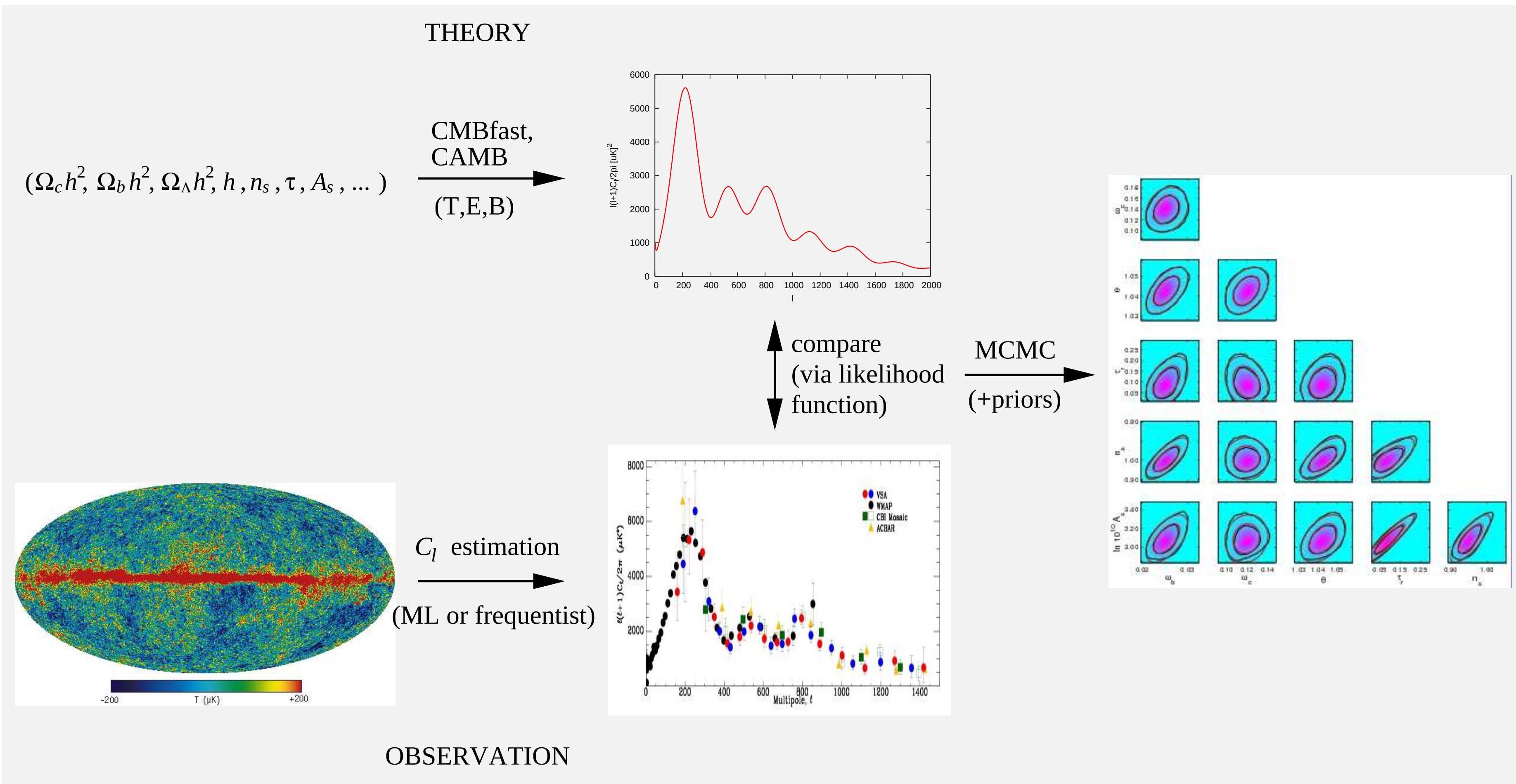


GMC Reconstruction of Rastrigin Function

Machine-learning



- Typical example: standard CMB data analysis pipeline



- Note parameter numbers: map ($\sim 10^7$), power spectrum ($\sim 10^3$), cosmological parameters (~ 10), cosmological models (~ 1)

Machine-learning in astronomy

- In modern astronomy, one is increasingly faced with the problem of analysing large, complicated and multidimensional data sets

- Such analyses typically include: data description and interpretation, inference, pattern recognition, prediction, classification, compression, and many more

- One way of performing such tasks is through machine-learning methods

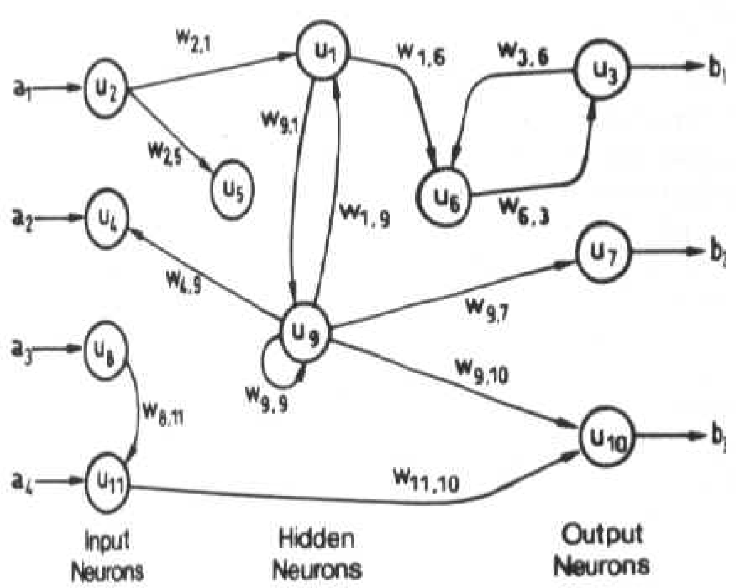
- Machine-learning software for astronomy, such as the astroML package* and SkyNet (see later), has recently started to become available

- Machine-learning can be divided into two broad categories: supervised learning and unsupervised learning.

*http://astroml.github.com/

NEURAL NETWORK APPROACH TO MACHINE-LEARNING

- NNs are an intuitive and well-established approach to machine learning, both supervised and unsupervised.



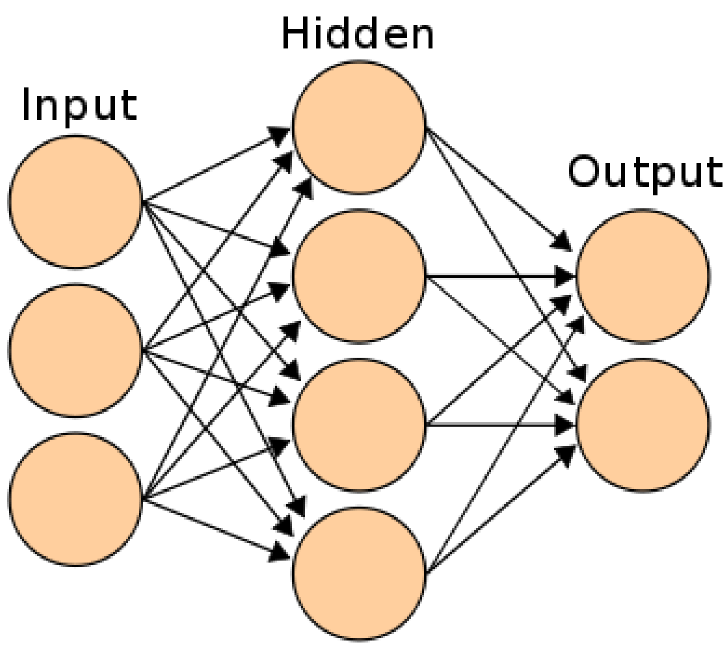
- Loosely inspired by structure and functional aspects of a brain

- Consist of a group of interconnected nodes, each of which processes information it receives and passes result to other nodes via weighted connections

- NNs constitute a non-linear statistical data modeling tool, which may be used to:
  - model complex relationships between a set of inputs and outputs
  - find patterns in data
  - capture the statistical structure between observed variables

Feed-forward networks

- In general, NN structure can be arbitrary, but many machine-learning applications can be performed using feed-forward NNs

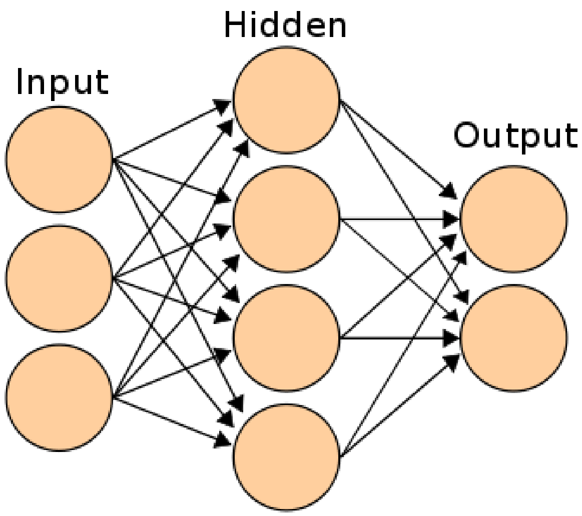


- Structure is directed: input layer of nodes passes information to output layer via zero, one, or many 'hidden' layers
- Such a network can 'learn' mapping between inputs and outputs, given a set of training data, then make predictions of the outputs for new input data

- Moreover, a universal approximation theorem assures us that any L_2-function $f : \Re^n \to \Re^m$, can be approximated to arbitrary mean square error accuracy by feed-forward NN with 1 or more hidden layers

FEED-FORWARD NETWORKS...

- Consider 3-layer NN: input layer, hidden layer and output layer



hidden layer: $h_j = g^{(1)}(f_j^{(1)}); \quad f_j^{(1)} = \sum_l w_{jl}^{(1)} x_l + b_j^{(1)},$

output layer: $y_i = g^{(2)}(f_i^{(2)}); \quad f_i^{(2)} = \sum_l w_{ij}^{(2)} h_j + b_i^{(2)},$

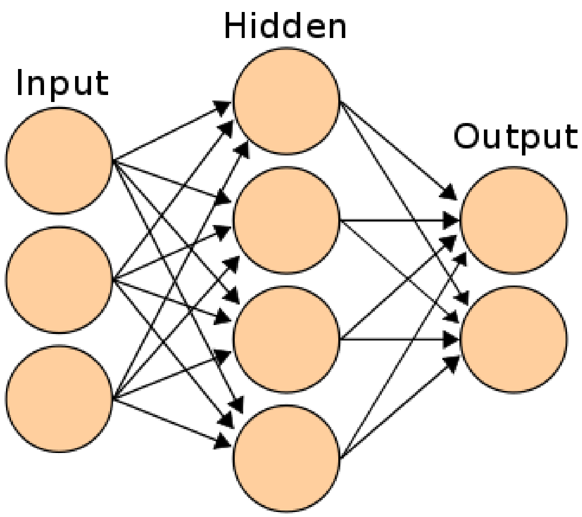
- Use non-linear activation function (e.g. $g_1(x) = \tanh x$ or $\text{sig}(x)$) on outputs of all hidden layer neurons; use $g_2(x) = x$

- Feed-forward NNs have been used in astronomy for over 20 years.
  But... widespread use has been limited by difficulty in training networks using standard techniques such as backpropagation, in particular networks having many nodes and/or numerous hidden layers (i.e. 'large' and/or 'deep' networks)

SkyNet

NETWORK TRAINING

- Training a NN = finding set of network weights and biases that maximise accuracy of predicted outputs; denote them collectively by the network parameter vector $\boldsymbol{a}$



- But, must be careful to avoid overfitting to our training data at the expense of making accurate predictions for unseen input values.

- Set of training data inputs and outputs, $\mathcal{D} = \{\boldsymbol{x}^{(k)}, \boldsymbol{t}^{(k)}\}$, is provided by the user

- Approximately 75 per cent should be used for actual NN training and remainder retained as a validation set used to determine convergence and avoid overfitting

NETWORK TRAINING: OBJECTIVE FUNCTIONS

- SKYNET considers parameters $\boldsymbol{a}$ to be random variables with a log-posterior

$$\ln \mathcal{P}(\boldsymbol{a}; \alpha, \boldsymbol{\sigma}) = \ln \mathcal{L}(\boldsymbol{a}; \boldsymbol{\sigma}) + \frac{\alpha}{2} \sum_i a_i^2,$$

  where hyperparameters $\boldsymbol{\sigma}$ describe rms of outputs and α is regulariser

- For regression problems, SKYNET assumes standard χ^2 misfit

$$\log \mathcal{L}(\boldsymbol{a}; \boldsymbol{\sigma}) = -\frac{K \log(2\pi)}{2} - \sum_{i=1}^{N} \log(\sigma_i) - \frac{1}{2} \sum_{k=1}^{K} \sum_{i=1}^{N} \left[\frac{t_i^{(k)} - y_i(\boldsymbol{x}^{(k)}; \boldsymbol{a})}{\sigma_i} \right]^2,$$

- For classification problems, SKYNET again uses continuous, interpreted as probabilities that inputs belongs to a particular output class. First softmax outputs:

$$y_i(\boldsymbol{x}^{(k)}; \boldsymbol{a}) \to \frac{\exp[y_i(\boldsymbol{x}^{(k)}; \boldsymbol{a})]}{\sum_{j=1}^{N} \exp[y_j(\boldsymbol{x}^{(k)}; \boldsymbol{a})]},$$

  then log-likelihood is cross-entropy of targets and softmaxed output values

$$\log \mathcal{L}(\boldsymbol{a}; \boldsymbol{\sigma}) = -\sum_{k=1}^{K} \sum_{i=1}^{N} t_i^{(k)} \log y_i(\boldsymbol{x}^{(k)}; \boldsymbol{a}),$$

NETWORK TRAINING: SKYNET APPROACH

- NN training typically performed using backpropagation: first-order gradient optimisation of log-likelihood $\ln \mathcal{L}(\boldsymbol{a})$ (with fixed $\boldsymbol{\sigma}$) $\Rightarrow$ convergence problems

- SKYNET takes very different approach:
  – whitening of input/output values
  – pre-training using layer-by-layer restricted Boltzmann machine contrastive divergence optimisation $\Rightarrow$ parameters $\boldsymbol{a}$ 'close' to 'good' optimum
  – optimisation using second-order truncated Newton method, but without need to calculate or store Hessian
  – automated updating of hyperparameters $\boldsymbol{\sigma}$ and α

- Combination of all the above methods
  $\Rightarrow$ avoids poor local optima
  $\Rightarrow$ practical use of second-derivative information even for large networks
  $\Rightarrow$ significantly improves rate of convergence to good optimum
  $\Rightarrow$ able to train large and/or deep networks (unlike backpropagation)

- Also, after training, SKYNET has fast algorithm to calculate accuracy of network's predicted outputs

NETWORK TRAINING: CONVERGENCE

- Following each iteration, the posterior, likelihood, correlation, and error squared values are calculated both for the training data and for the validation data

- Correlation of network outputs is defined for each output i as

$$\text{Corr}_i(\boldsymbol{a}) = \frac{\sum_{k=1}^{K}(t_i^{(k)} - \bar{t}_i)(y_i - \bar{y}_i)}{\sqrt{\sum_{k=1}^{K}(t_i^{(k)} - \bar{t}_i)^2 \sum_{k=1}^{K}(y_i^{(k)} - \bar{y}_i)^2}},$$

  where $\bar{t}_i$ and $\bar{y}_i$ are means of these output variables over all training data
  – provides relative measure of how well predicted outputs match true ones
  – correlations from each output can be averaged to give overall correlation

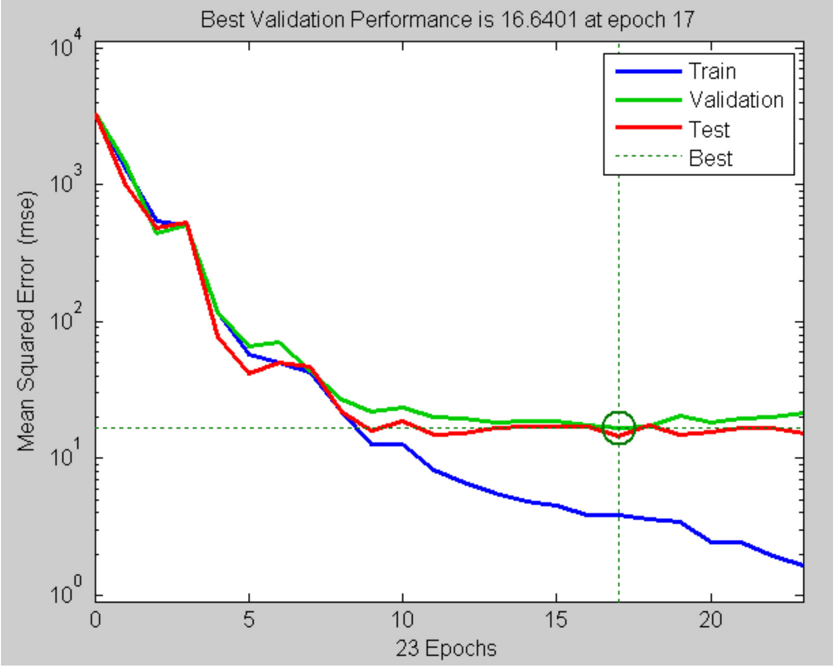
- Average error-squared of network outputs is defined by

$$\text{ErSq}(\mathbf{a}) = \frac{1}{NK}\sum_{k=1}^{K}\sum_{i=1}^{N}\left[t_i^{(k)} - y_i(\mathbf{x}^{(k)};\mathbf{a})\right]^2,$$

  and is complementary to correlation, since it is an absolute measure of accuracy

NETWORK TRAINING: CONVERGENCE...

- As optimisation proceeds, there is a steady increase in posterior, likelihood, correlation, and negative of error squared, both for the training and validation data



- But, eventually algorithm will begin to overfit $\Rightarrow$ continued increase of these quantities for training data, but decrease for validation data

- This divergence in behaviour is taken as indicating that the algorithm has converged and the optimisation in terminated

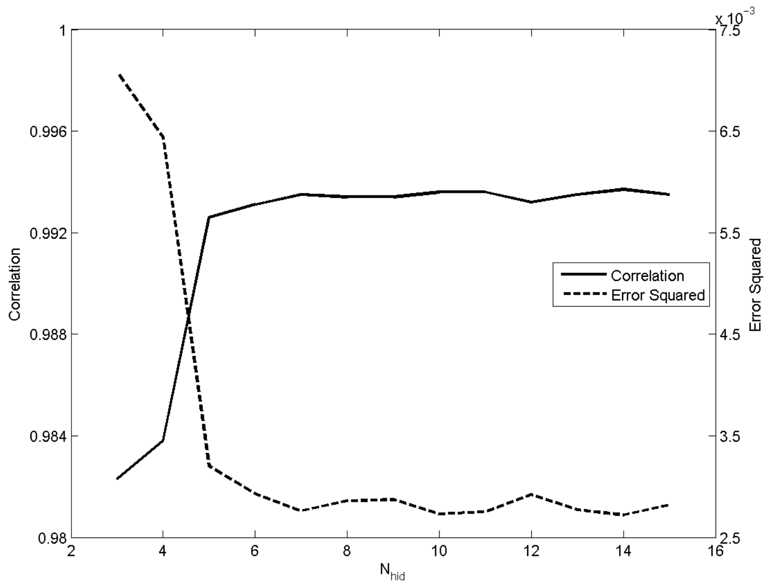
- User may choose which of the four quantities is used to determine convergence. The default the error squared (independent of hyperparameters $\boldsymbol{\sigma}$ and α)

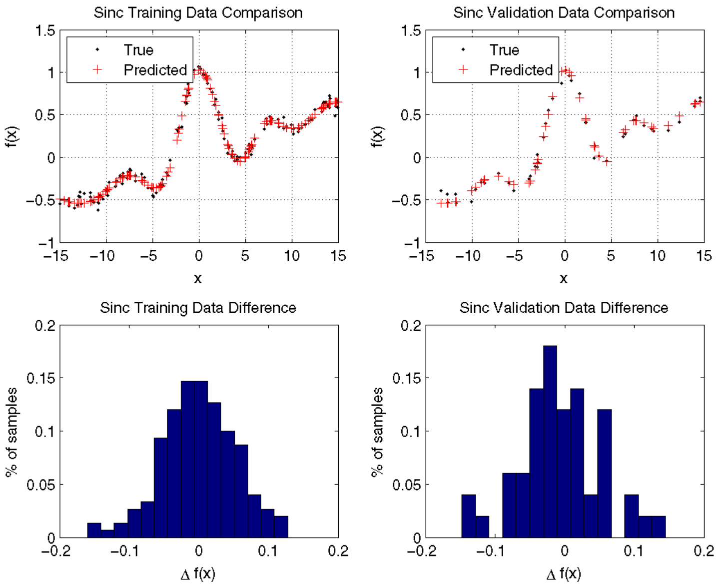
- Also, correlation and error-squared provide quantitative measuresto compare performance of different network architectures (no. of hidden nodes/layers)

Toy example: regression

- Generate 200 points randomly in $x \in [-5\pi, 5\pi]$ and evaluate ramped sinc function
$$y(x) = \frac{\sin(x)}{x} + 0.04x,$$
then add Gaussian noise $\mathcal{N}(0, 0.05^2)$ (prevents exact solution)

- Divide data (x, y) into $3:1$ for training: validation and use $1 + N + 1$ networks

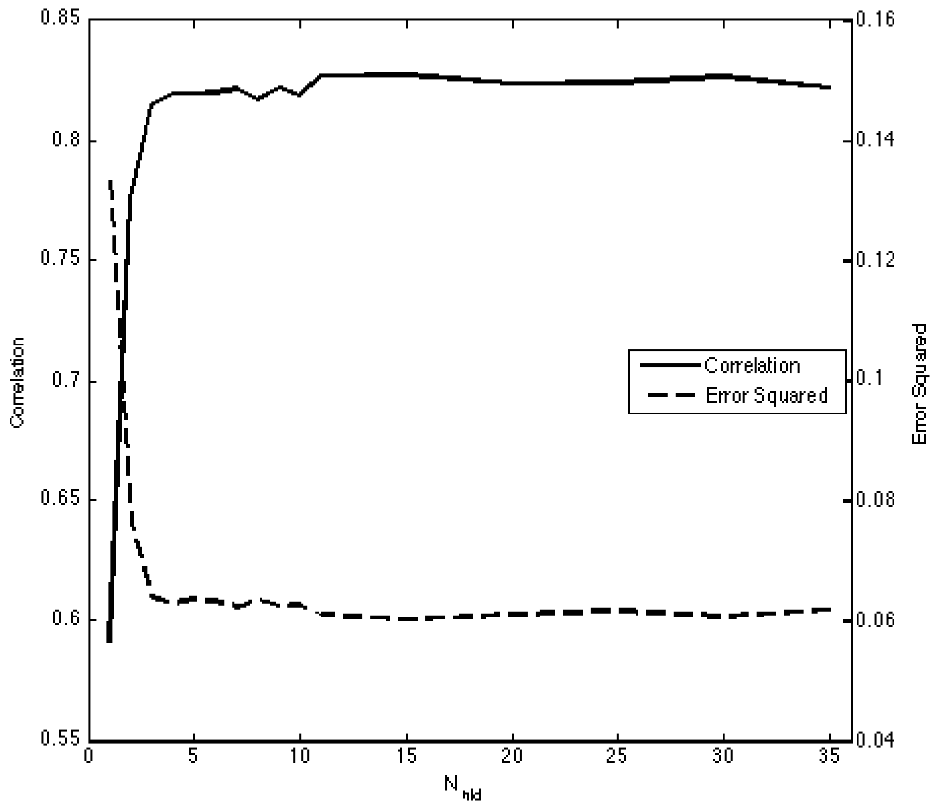




- Astronomical applications: ellipticities of lensed galaxies, accelerated inference, ...

Toy Example: Classification

- 3-way classification problem proposed by Radford Neal:
  - each of four variables x_1, x_2, x_3, and x_4 is drawn 1000 times from $\mathcal{U}[0, 1]$
  - if distance between (x_1, x_2) and $(0.4, 0.5)$ is $< 0.35 \Rightarrow$ point in class 0
  - if $0.8x_1 + 1.8x_2 < 0.6 \Rightarrow$ point in class 1
  - if neither of true $\Rightarrow$ point in class 2
  - Gaussian noise $\mathcal{N}(0, 0.1^2)$ then added to input values
  - note values of x_3 and x_4 play no part in classification

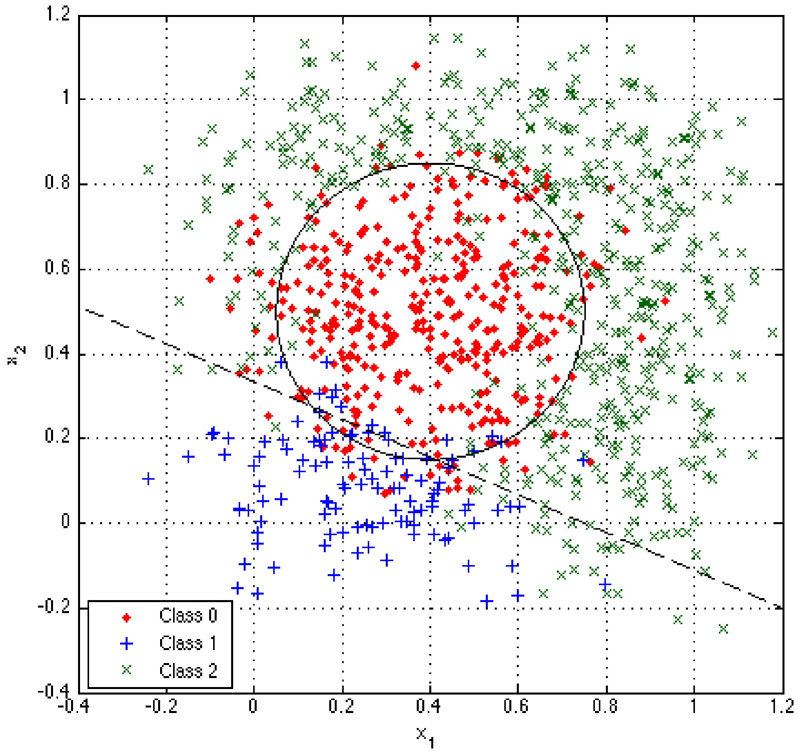


- Divide data into $3:1$ for training: validation and consider networks $4 + N + 3$. Final class assigned is output with largest probability

TOY EXAMPLE: CLASSIFICATION...

- For network with $N = 8$ hidden nodes, 87.8% of training data points and 85.4% of validation data points were correctly classified

	True class	Number	Predicted class (%) 0	1	2
Training data	0	282	84.0	4.96	11.0
	1	93	14.0	82.8	3.2
	2	386	7.0	1.3	91.7
Validation data	0	99	75.7	6.1	18.2
	1	19	21.1	78.9	0.0
	2	121	5.0	0.8	94.2

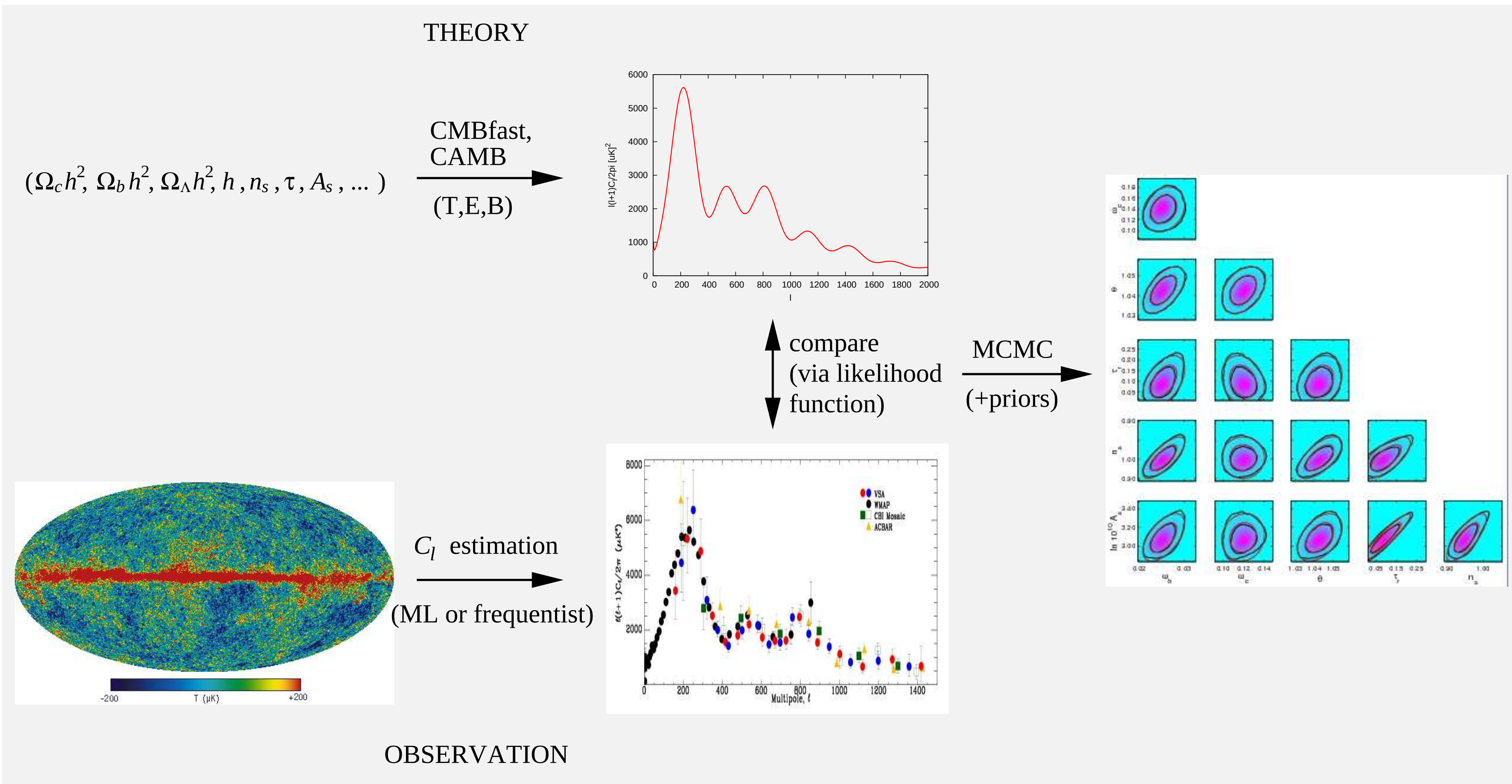


- Compares well with Neal's own original results and are similar to classifications based on applying original criteria directly to data points after noise added

- Astronomical applications: SNe typing, gamma-ray burster identification, ...

Combining nests and nets: BAMBI

- Typical example: standard CMB data analysis pipeline



- Note parameter numbers: map ($\sim 10^7$), power spectrum ($\sim 10^3$), cosmological parameters (~ 10), cosmological models (~ 1)

Blind Accelerated Multimodal Bayesian Inference (BAMBI)

- General Bayesian inference engine with wide applicability: only requires choice of priors on the parameters in model (Graff et al., arXiv:1110.2997)

- Combines neural networks (SkyNet – a new, general-purpose, standalone NN training code) and nested sampling (MultiNest) in complementary manner

- Basic idea is as follows:

  - early stage (prior-driven) nested samples ⇒ (incremental) training data set

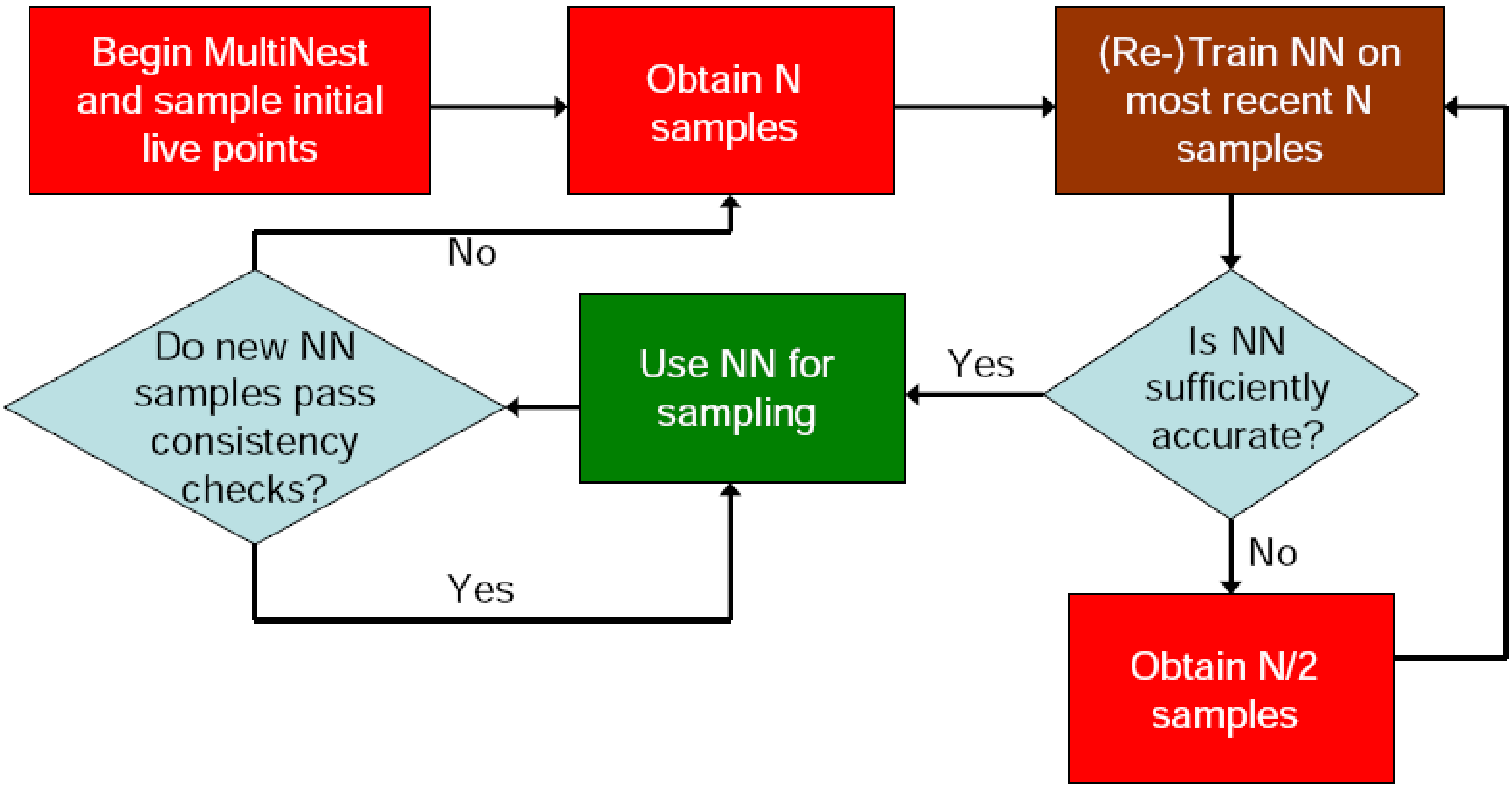
  - simultaneous training of neural network(s) ⇒ ‘learn’ likelihood function

  - clustering in nested sampler ⇒ accelerates network training

  - once trained, network(s) replace(s) likelihood code
    ⇒ completes posterior sampling and evidence evaluation extremely rapidly

  - trained likelihood network(s) available for subsequent analyses

OUTLINE OF BAMBI APPROACH



ADVANTAGES OF BAMBI

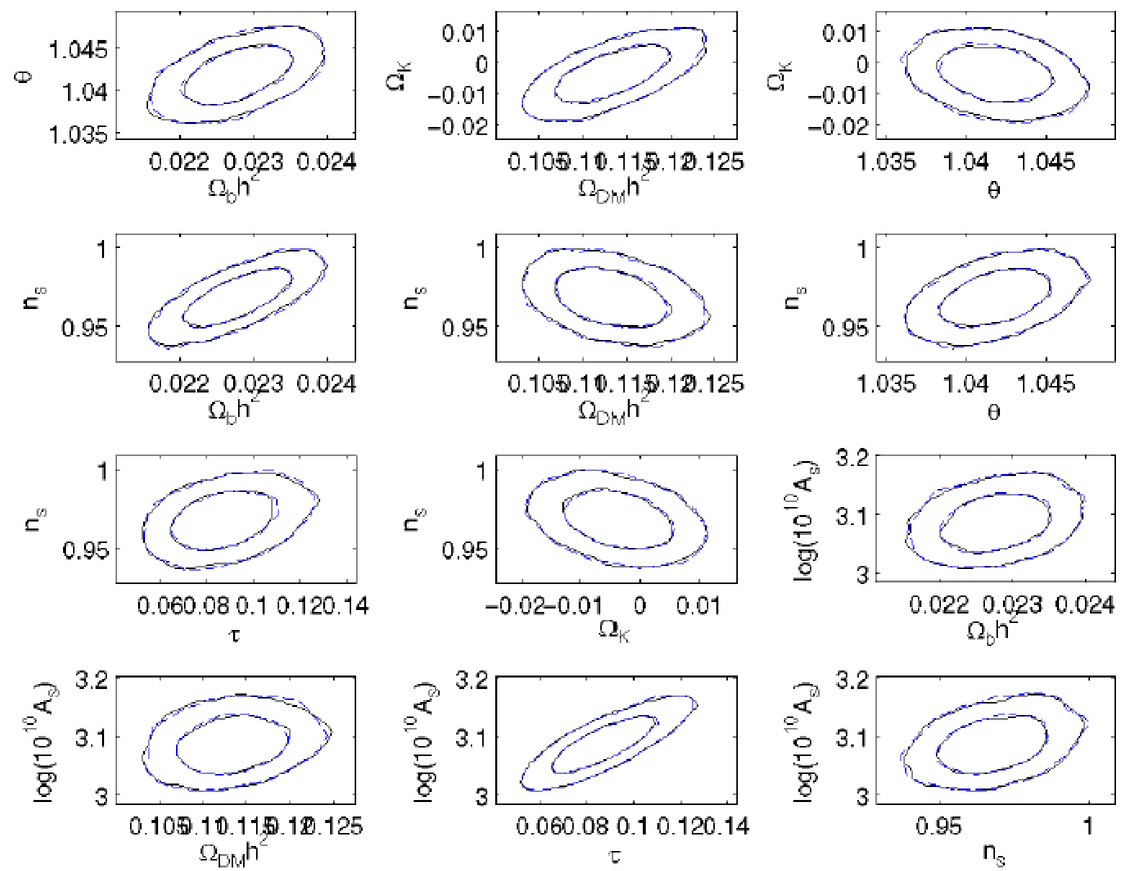
- For primary analysis, typically ~ 1.5 times faster than MULTINEST alone
  $\Rightarrow$ modest gain in speed over MULTINEST and get trained networks as a bonus

- Automated training of network(s) over the entire parameter space
  – Can also obtain gradients of likelihood from trained network(s)

- For subsequent (secondary) analyses:
  – Likelihood calls from trained networks(s) require $\sim 10^{-4}$ sec $\Rightarrow$ huge speed-ups (much faster network error calculation – new feature relative to published version)
  – May use different (smaller) priors
  – Ideal for e.g. coverage studies

APPLICATION OF BAMBI: COSMOLOGY

- Analysis of non-flat ΛCDM model with CMB and LSS data



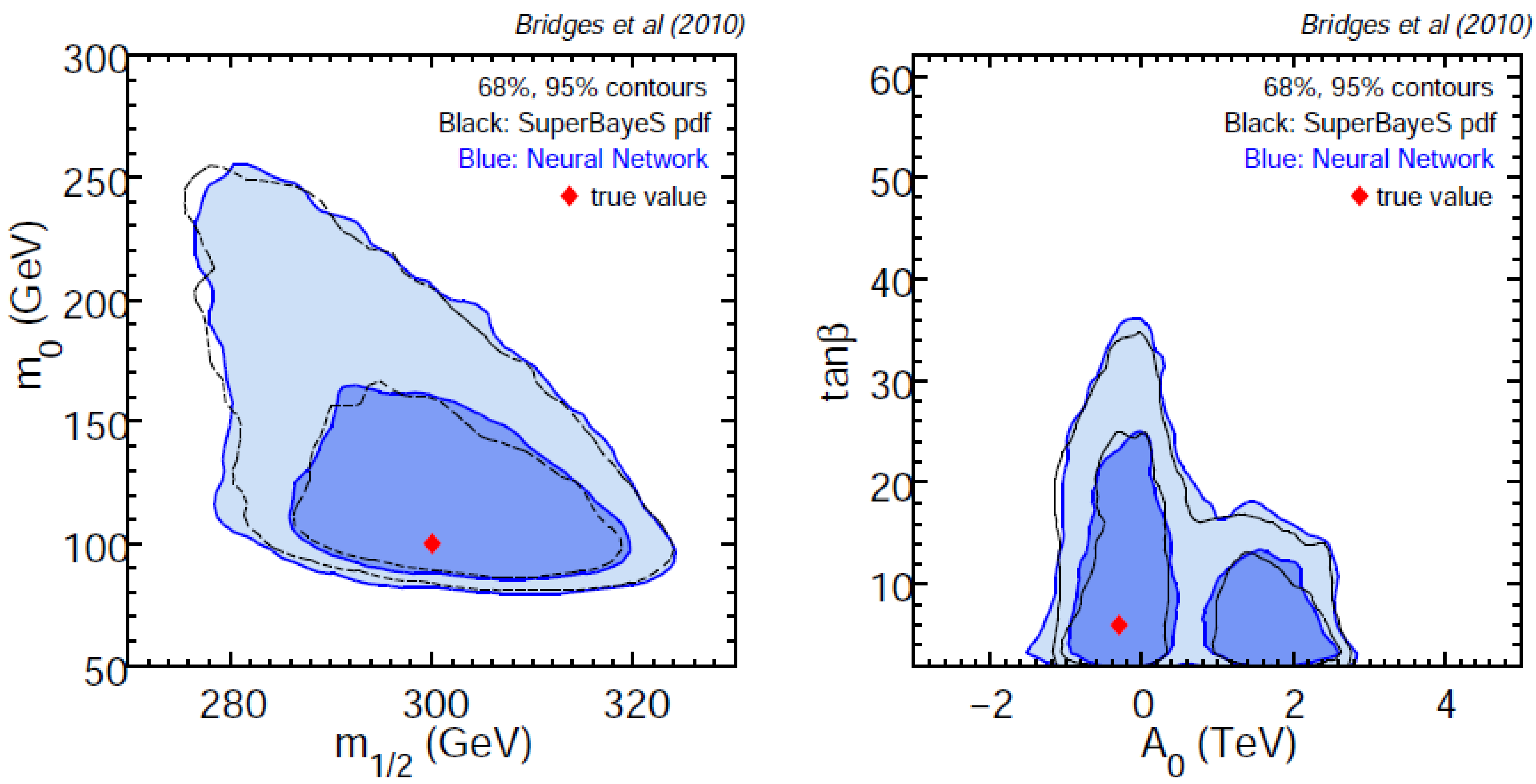
- MULTINEST results in solid (black) and BAMBI in dashed (blue)

- Log-evidence from BAMBI accurate to within 0.1 units

- BAMBI speed-up: primary analysis ~ 1.5, subsequent analyses $\sim 10^{4-5}$

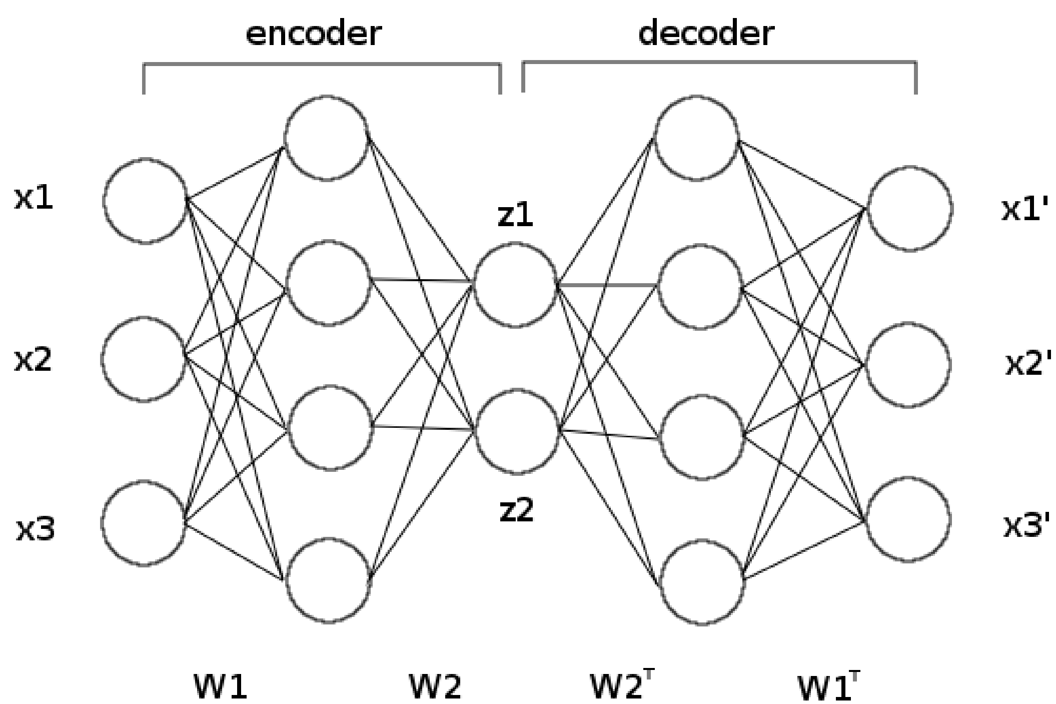
Application of BAMBI: particle physics

- Consider restricted class of SUSY models with certain universality assumptions regarding SUSY breaking parameters: cMSSM (8-D parameter space)



- Total speed-up of analysis by factor $\sim 10^6$
  $\Rightarrow$ original SoftSusy + MCMC = 720 CPU days; Bambi = 1 minute

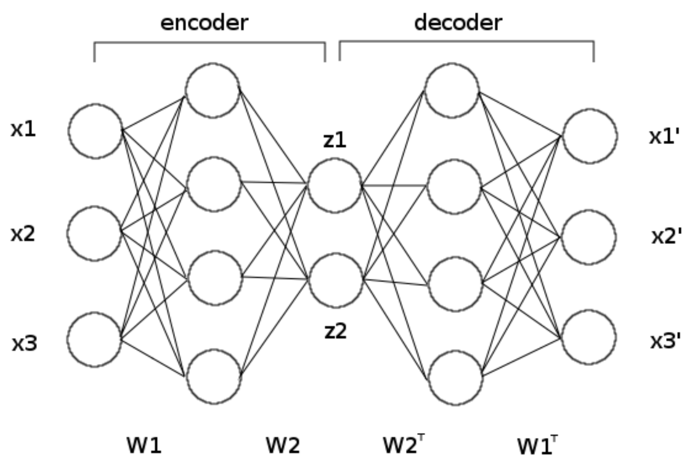
And now for something completely different. . .



. . . Autoencoders

AUTOENCODERS

- Autoencoders are a specific type of feed-forward NN, where the inputs are mapped to themselves, i.e. the network is trained to approximate the identity operation



- Typically contain several hidden layers and are symmetric about a central layer containing fewer nodes than inputs

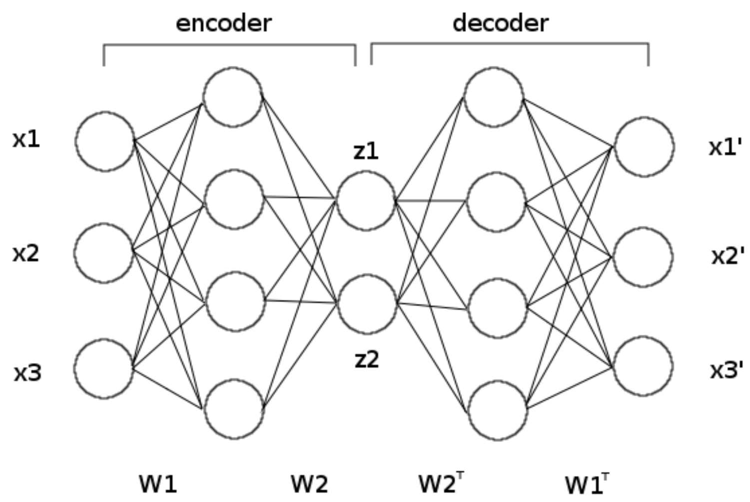
- Can be considered as two half-networks: the 'encoder' and 'decoder' map either to or from a reduced set of 'feature variables' embodied in central layer nodes

- Feature variables are, in general, non-linear functions of the original input variables

- Determine dependence for each feature variable in turn simply by decoding $(z_1, 0, 0, \ldots, 0)$, $(0, z_2, 0, \ldots, 0)$, etc. as corresponding z_i value is varied
$\Rightarrow$ maps out a curve in the original data space

- Conversely, $(z_1, z_2, \ldots, z_M)$ in central layer is feature vector of the input data

AUTOENCODERS...



- Autoencoders (AEs) thus provide a very intuitive approach to non-linear dimensionality reduction

- Constitute a natural generalisation of linear methods such as PCA and ICA, which are widely used in astronomy. Indeed, an antoencoder with a single hidden layer and linear activation functions is identical to PCA.

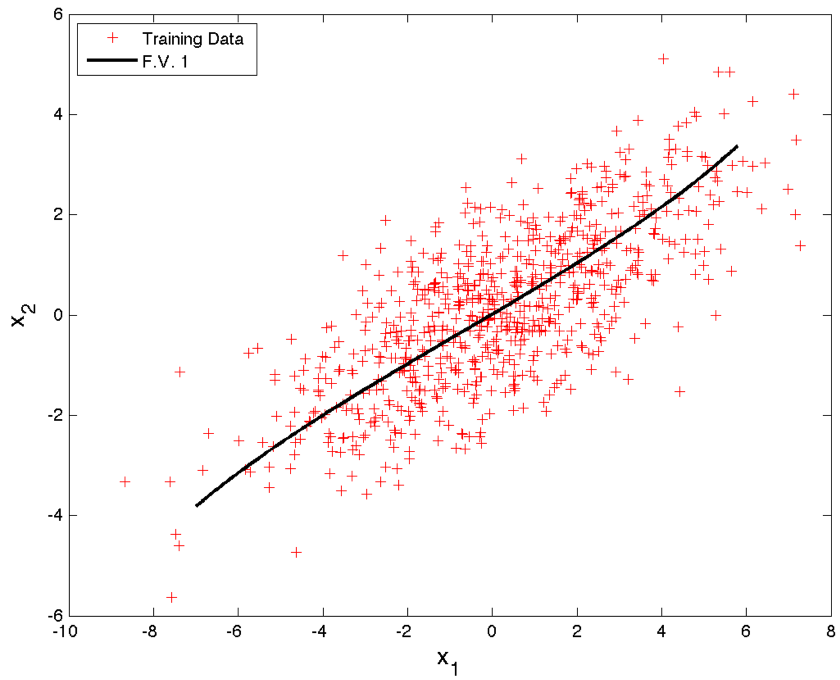
- Encoding from input data to feature variables also useful in clustering tasks

- Autoencoders are, however, notoriously difficult to train, since objective function contains a broad local maximum where all outputs = average value of the inputs, but can be overcome with pre-training methods

DIMENSIONALITY REDUCTION USING AUTOENCODERS

- Dimensionality reduction is a very common task in astronomy. Antoencoders provide a natural non-linear generalisation of PCA and ICA, which reduces to PCA in the special case of a single hidden layer and linear activation functions

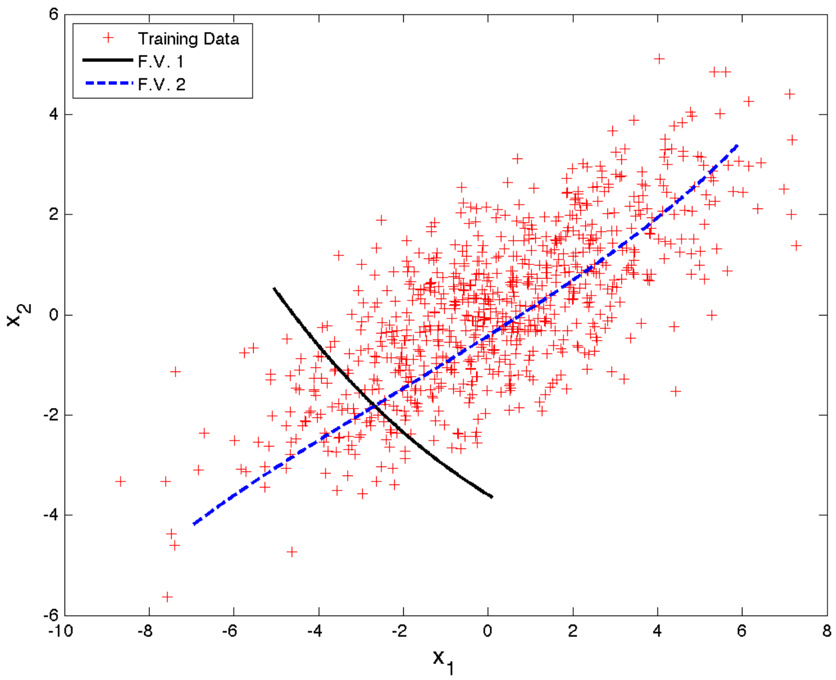
- 2-D Gaussian data (x_1, x_2) using $2 + 1 + 2$ autoencoder



- Output curve traced in data space as one varies feature value z_1
- Approximates eigenvector with larger eigenvalue of data covariance matrix
- Dimensionality reduction performed conversely by (non-linear) encoding of each input (x_1, x_2) to obtain z_1
- Error-squared and correlation for antoencoder are 0.476 and 90.5%

DIMENSIONALITY REDUCTION USING AUTOENCODERS...

- 2-D Gaussian data (x_1, x_2) using $2 + 2 + 2$ autoencoder (so no dimensionality reduction)



- Curves traced out as one varies feature values $(z_1, 0)$ and $(0, z_2)$
- Approximate both eigenvectors of data covariance matrix
- Error-squared and correlation for autoencoder are 0.022 and 99.8%

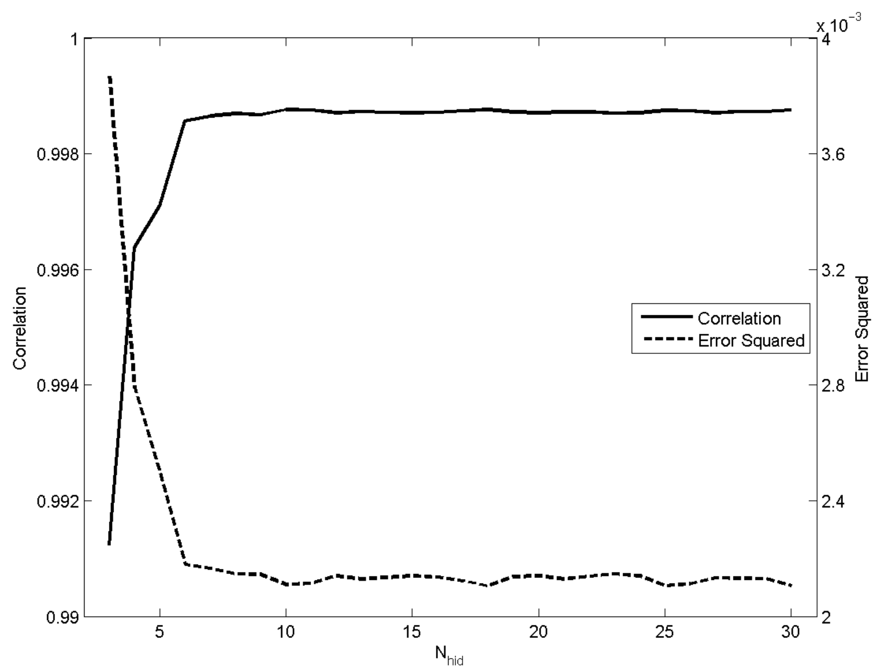
- Note that error-squared and correlation very close to perfect, as one would expect for this two-dimensional data set

- Data (x_1, x_2) distributed about a partial ring $\Rightarrow$ long curving degeneracy:

$$x_1 = 0.5 + (0.5 - n)\cos\theta, \quad (1a)$$
$$x_2 = 0.5 + (0.5 - n)\sin\theta, \quad (1b)$$

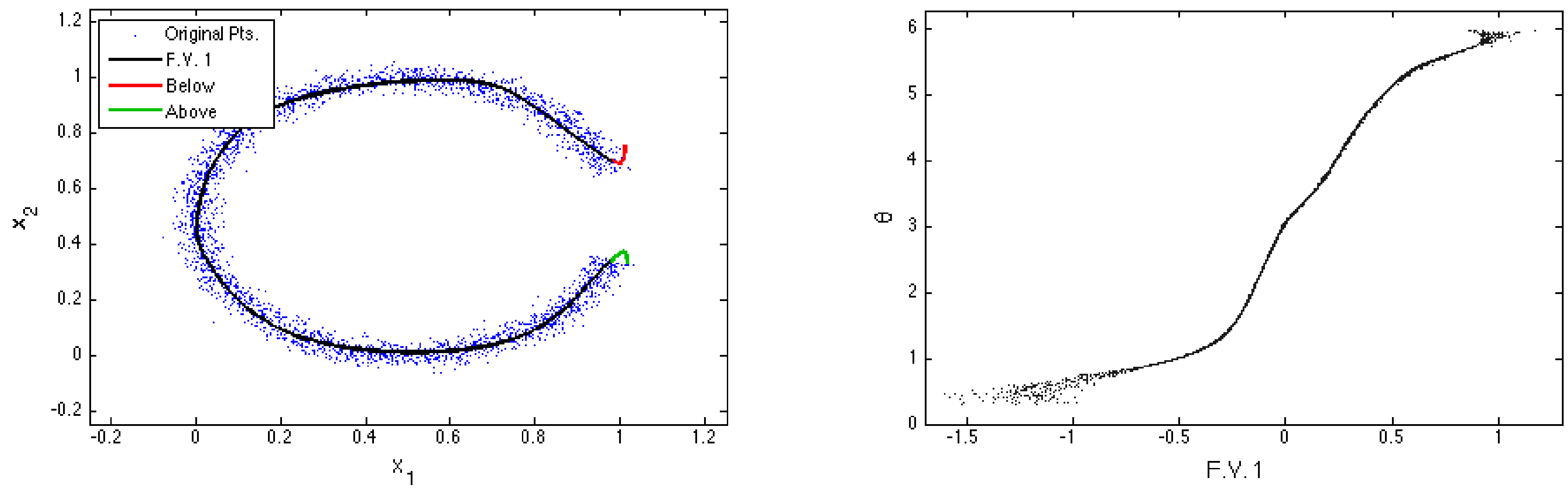
where $\theta \sim \mathcal{U}(0.1\pi, 1.9\pi)$ and $n \sim \mathcal{N}(0, 0.1^2)$.

- Train deeper autoencoders with architecture $2 + N + 1 + N + 2$:



DIMENSIONALITY REDUCTION USING AUTOENCODERS...

- For autoencoder with architecture $2 + 13 + 1 + 13 + 2$:



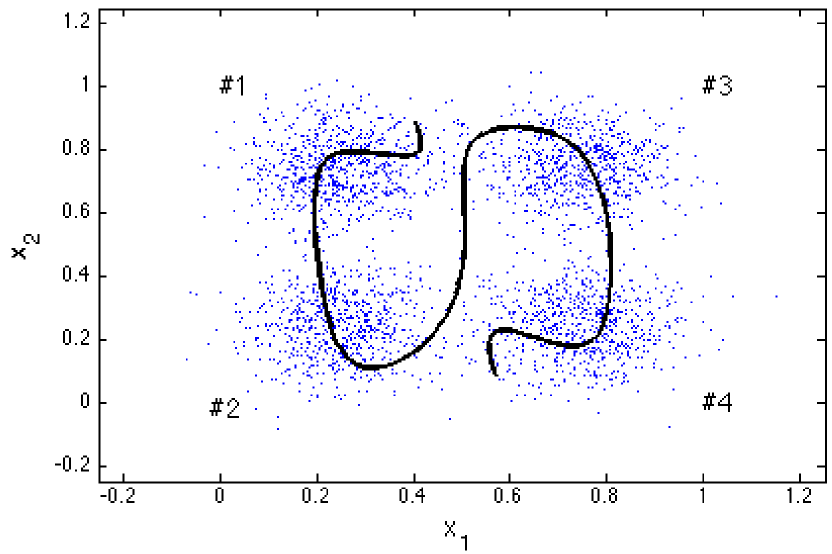
- PCA is unable to represent this data set accurately in a single component. Indeed, dominant principal component lies along straight, horizontal (symmetry) line

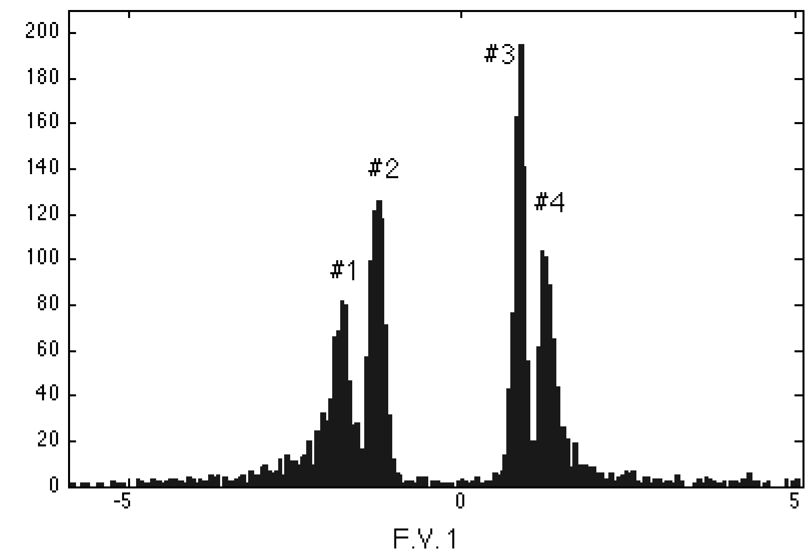
- Projections onto dominant principal component do not distinguish between data points having same x_1-coordinate, but lying on opposite sides of symmetry line

DIMENSIONALITY REDUCTION USING AUTOENCODERS...

- Data (x_1, x_2) drawn from sum of four equal Gaussians $\Rightarrow$ multimodal distribution

- For autoencoder with architecture $2 + 10 + 1 + 10 + 2$:





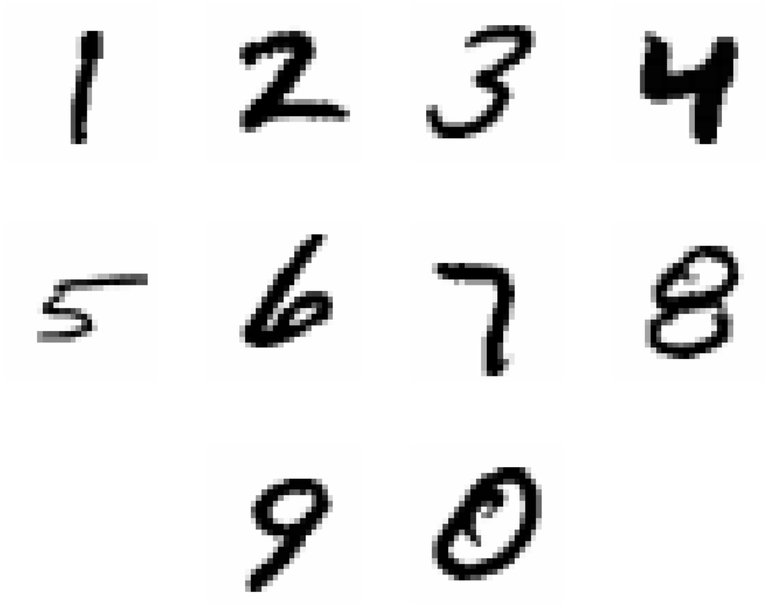
- PCA is unable to represent data in single component. Indeed, two principal directions (with $\sim$ equal eigenvalues) lie along diagonal (symmetry) lines at $\pm 45^\circ$

- Projections onto line at $+45$ degrees (say) $\Rightarrow$ conflate modes 1 and 4

MNIST HANDWRITING RECOGNITION

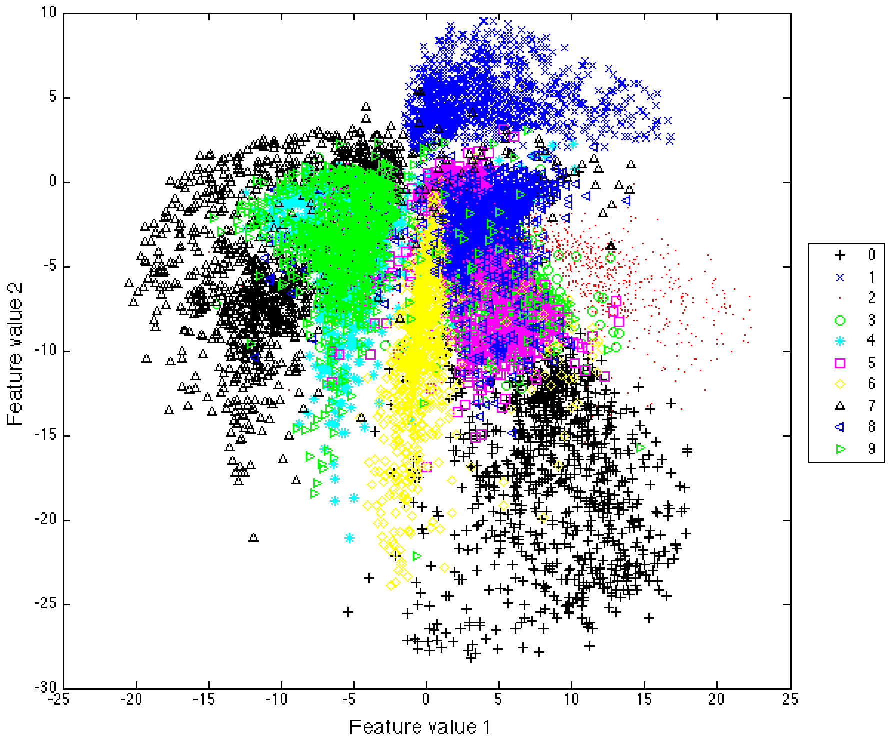
- MNIST database: $60,000$ training, $10,000$ validation images of handwritten digits

- Each digit has been size-normalised and centred in 28×28 pixel greyscale image



- Data set has become a standard for testing of machine-learning algorithms

- Perform massive compression to just two feature variables by training very large and deep autoencoder with hidden layers $1000 + 500 + 250 + 2 + 250 + 500 + 1000$ (and 784 inputs/outputs) ⇒ simple illustration of clustering

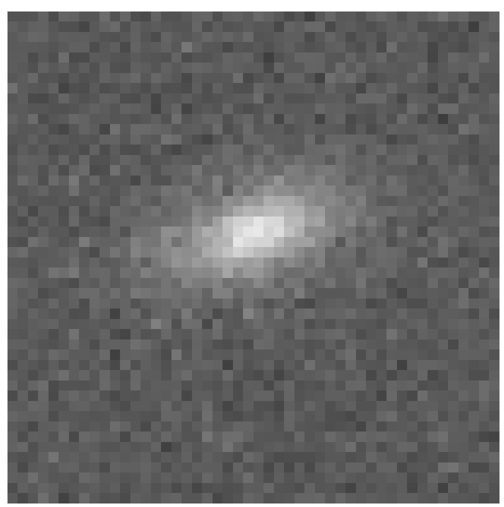


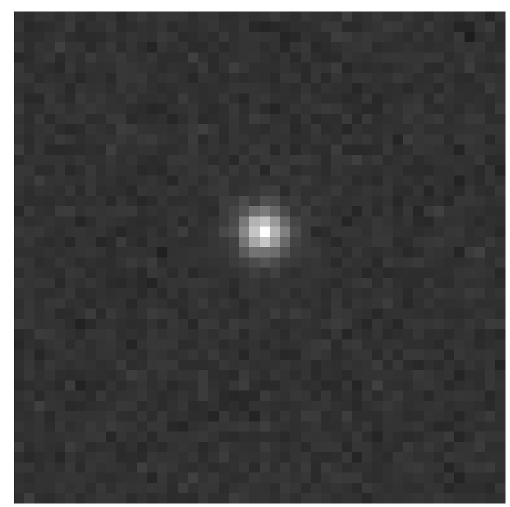
- There is significant overlap between digits with similar shapes, but some digits do occupy distinct regions of the feature vector space

Mapping dark matter challenge

- Mapping Dark Matter (MDM) Challenge was presented on `www.kaggle.com` as a simplified version of GREAT10 Challenge

- Each data item consists of two 48×48 greyscale images of a galaxy and a star
  – each pixel value is Poisson distributed with mean equal to underlying intensity
  – both images convolved with same, but unknown, point spread function

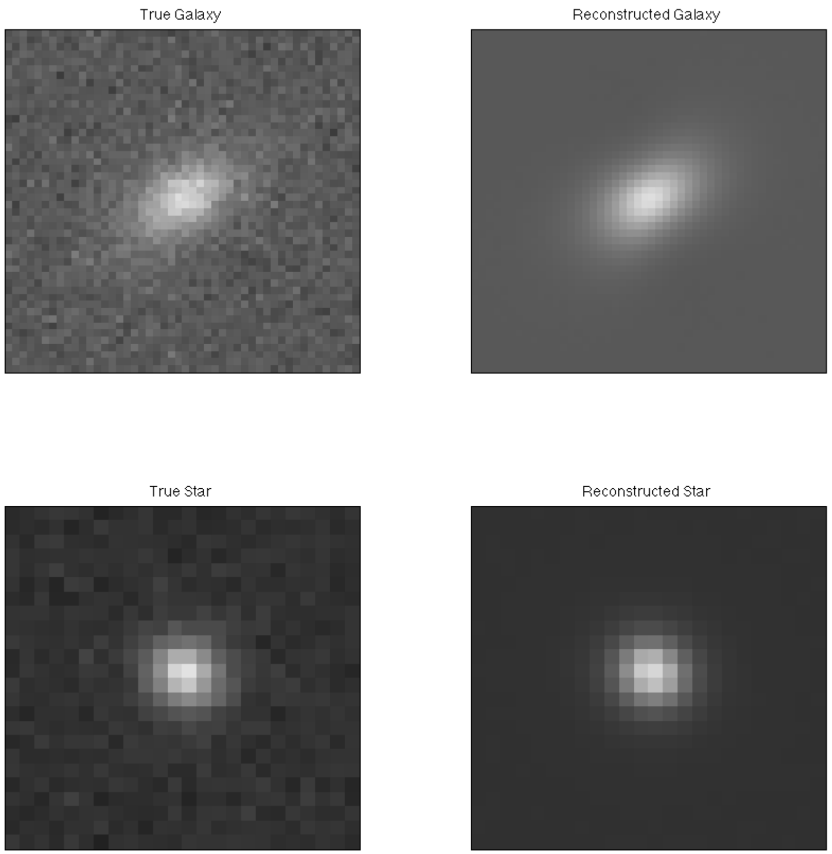




MDM CHALLENGE...

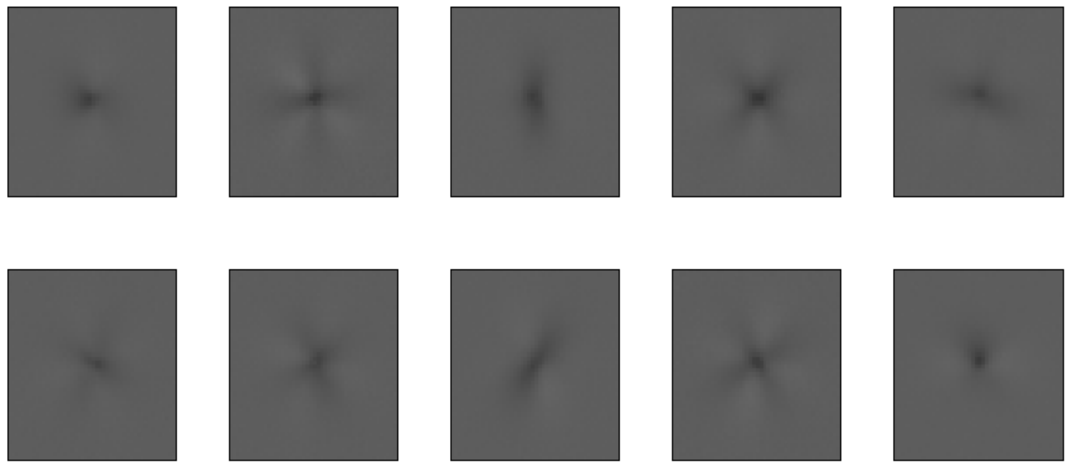
- Perform image compression and denoising using dimensionality reduction

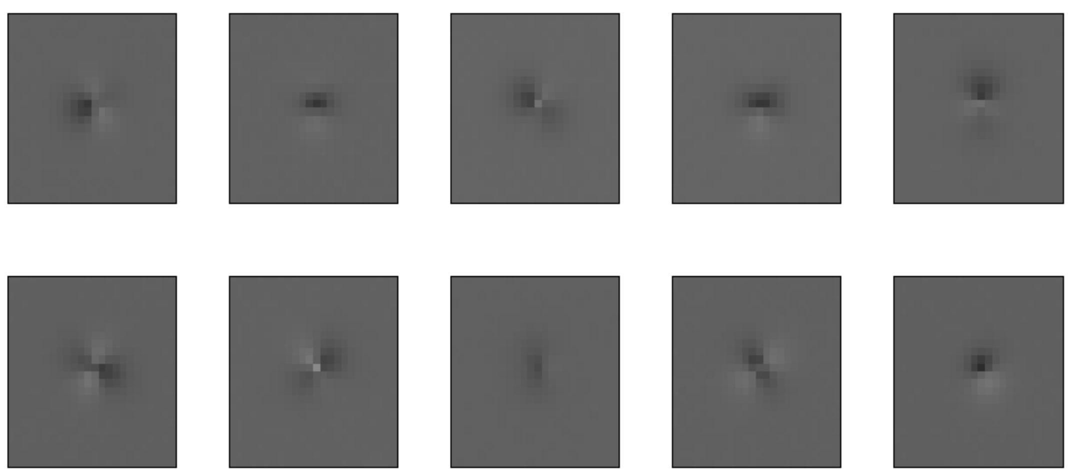
- Train autoencoder on MDM full galaxy and cropped star images ($48 \times 48 + 24 \times 24 = 2880$ pixels) with architecture $2880 + N + 2880$. Determine $N \sim 10$ automatically (as before) $\Rightarrow$ massive compression



MDM CHALLENGE. . .

- Can determine feature vectors by decoding unit inputs to each hidden layer node





CONCLUSIONS

- Neural networks are an intuitive and useful approach to machine-learning
- Can be used for regression, classification, dimensionality reduction, clustering, accelerated inference, and much more...
- SKYNET is an efficient and robust generic NN training tool
- BAMBI combines nets and nests $\Rightarrow$ generic approach to accelerated Bayesian inference, already applied in many areas, giving overall speed-up $\sim 10^6$ compared to standard MCMC and original likelihoods (in cosmology)
- MULTINEST (arXiv:0809.3437), `www.mrao.cam.ac.uk/software/multinest`
  SKYNET (arXiv:1309.0790), `www.mrao.cam.ac.uk/software/skynet`
  BAMBI (arXiv:1110.2997), `www.mrao.cam.ac.uk/software/bambi`
- Autoencoders are interesting and may be useful in astronomy and beyond...