Big Data and Bayesian Learning

Greek Stochastics θ

Yee Whye Teh

in collaboration with:
Charles Blundell, Balaji Lakshminarayanan, Leonard Hasenclever, Thibaut Lienart, Xiaoyu Lu, Sam Patterson, Valerio Perrone, Alex Thiery, Sebastian Vollmer, Stefan Webb, Max Welling, Minjie Xu, Kostas Zygalakis.

Big Data and Bayesian Learning?

- Large scale datasets are fast becoming the norm.

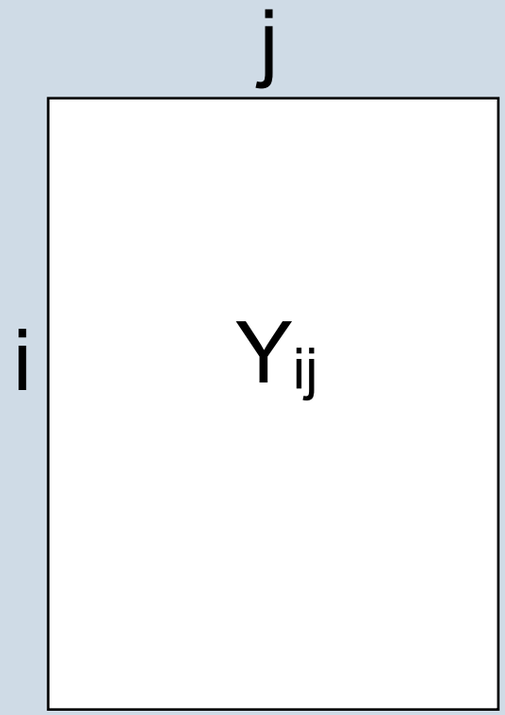
- Analysing and extracting understanding from these data is a driver of progress in many sectors of society.

- Current successes in scalable machine learning are optimization-based and non-Bayesian.

- What is the role of Bayesian learning in world of Big Data?

Product Recommendation Systems

- Data: Collection of pairs {(i,j)} and Y_{ij}: how much customer i likes product j.
- Learn about the likes and dislikes of each customer.



- Model each user and product as vectors.

$$Y_{ij}|X_{ui}, X_{pj} \sim X_{ui}^{\top} X_{pj} + \mathcal{N}(0, \epsilon)$$

Year	Name	#Ratings	#Users	#items
1999	MovieLens	.1M	943	1682
2004	EachMovie	2.8M	72916	1682
2006	Netflix	100M	480189	17770
2011	Yahoo Music	263M	1000990	624961

Topic Modelling

- Data: Collection of “documents”, each document consisting of a number of “words”.

- Learn about groups of co-occurring words, or “topics”.
- Model each document as a mixture of topics.

- Latent Dirichlet allocation [Blei et al 2003].

Topic Modelling

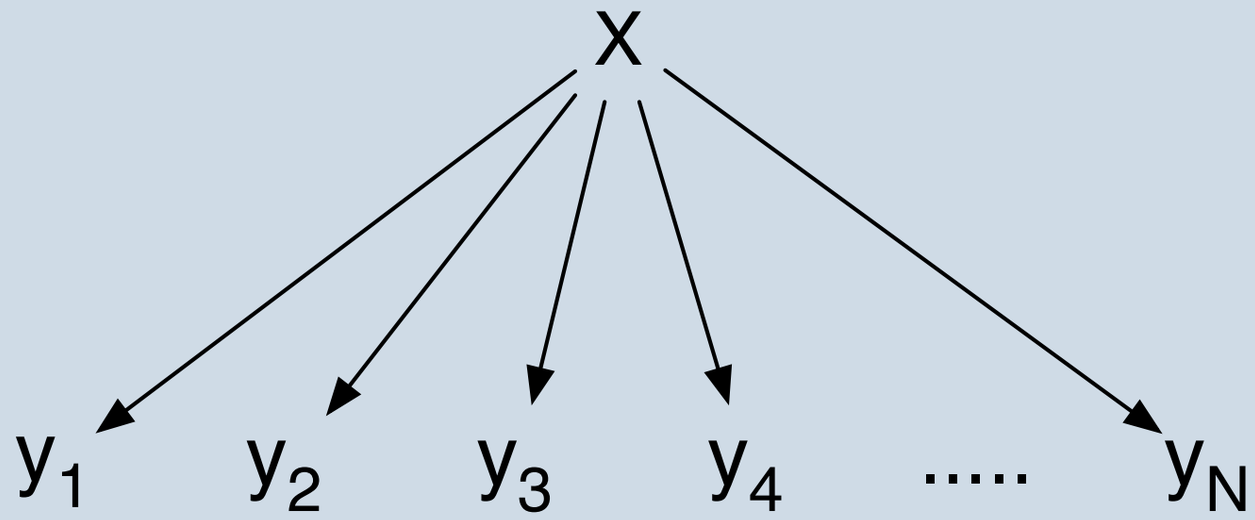
"Arts"	"Budgets"	"Children"	"Education"
NEW	MILLION	CHILDREN	SCHOOL
FILM	TAX	WOMEN	STUDENTS
SHOW	PROGRAM	PEOPLE	SCHOOLS
MUSIC	BUDGET	CHILD	EDUCATION
MOVIE	BILLION	YEARS	TEACHERS
PLAY	FEDERAL	FAMILIES	HIGH
MUSICAL	YEAR	WORK	PUBLIC
BEST	SPENDING	PARENTS	TEACHER
ACTOR	NEW	SAYS	BENNETT
FIRST	STATE	FAMILY	MANIGAT
YORK	PLAN	WELFARE	NAMPHY
OPERA	MONEY	MEN	STATE
THEATER	PROGRAMS	PERCENT	PRESIDENT
ACTRESS	GOVERNMENT	CARE	ELEMENTARY
LOVE	CONGRESS	LIFE	HAITI

The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. "Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services," Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center's share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.

Bayesian Learning: Simple Setup

- Parameter vector X.

- Data items $Y = y_1, y_2, \ldots y_N$.



- Model:

$$p(x,y) = p(x)\prod_{i=1}^{N} p(y_i|x) = p(x)\prod_{i=1}^{N} \ell_i(x)$$

- Aim:

$$p(x|y) = \frac{p(x)p(y|x)}{p(y)}$$

Important Issues Beyond This Talk and Setup

- Data:
  - Heterogeneity and complexity
  - Big collection of small data
  - High dimensional data
  - Causality

- Methodology:
  - Modelling flexibility, generality and ease of use
  - Algorithm flexibility, generality and ease of use
  - Software flexibility, generality and ease of use

Why Bayesian Machine Learning?

- An important framework to frame learning.
- Flexible and intuitive construction of complex models.
- Quantification of uncertainty.
- Mitigation of overfitting.
- Straightforward derivation of learning algorithms.

Generic (Bayesian) Learning on Big Data

- Stochastic optimisation using mini-batches.
  - Stochastic gradient optimisation.
- Distributed/parallel computations on cores/clusters/GPUs.
  - MapReduce, parameter server.

Generic (Bayesian) Learning on Big Data

- Stochastic optimisation using mini-batches.
  - Stochastic gradient optimisation.
  - Stochastic variational inference [Hoffman et al 2013, Mimno et al 2012]
  - Stochastic Gradient MCMC [Welling & Teh 2011, Patterson & Teh 2013, Teh et al 2016, Chen et al 2014, Ma et al 2015, Din et al 2015, Leimkuhler & Shang 2015…]

- Distributed/parallel computations on cores/clusters/GPUs.
  - MapReduce, parameter server.
  - Embarassingly Parallel MCMC [[Scott et al 2013, Neiswanger et al 2013, Wang & Dunson 2013, Stanislav et al 2014]
  - Sampling via Moment Sharing [Xu et al 2014]
  - Stochastic Natural-gradient EP and Posterior Server [Teh et al 2016]

Generic (Bayesian) Learning on Big Data

- Stochastic optimisation using mini-batches.
  - Stochastic gradient optimisation.
  - **Stochastic variational inference [Hoffman et al 2013, Mimno et al 2012]**
  - Stochastic Gradient MCMC [Welling & Teh 2011, Patterson & Teh 2013, Teh et al 2016, Chen et al 2014, Ma et al 2015, Din et al 2015, Leimkuhler & Shang 2015…]

- Distributed/parallel computations on cores/clusters/GPUs.
  - MapReduce, parameter server.
  - Embarassingly Parallel MCMC [[Scott et al 2013, Neiswanger et al 2013, Wang & Dunson 2013, Stanislav et al 2014]
  - Sampling via Moment Sharing [Xu et al 2014]
  - Stochastic Natural-gradient EP and Posterior Server [Teh et al 2016]

Exponential Families

$$p_\theta(x) = \exp\left(\theta^\top s(x) - A(\theta)\right)$$

- Sufficient statistics s, natural parameters θ, log partition function $A(\theta)$.

- Equivalent parameterisation as mean parameters μ:

$$\mu = \mathbb{E}_\theta[s(x)]$$

- $A(\theta)$ is convex with convex domain Θ.
- Convex conjugate is the negative entropy $A^*(\mu)$ with convex domain $\mathcal{M}$:

$$A^*(\mu) = \sup_\theta \theta^\top \mu - A(\theta) = \mathbb{E}_{\theta(\mu)}[\log p_{\theta(\mu)}(x)]$$

- Derivatives of A and A^* convert between natural and mean parameters:

$$\mu(\theta) = \nabla A(\theta) \qquad \theta(\mu) = \nabla A^*(\mu)$$

- See [Wainwright & Jordan 2008].

Arbitrary Model as Extended Exponential Family

- Prior $p_0(x)$ in exponential family, log likelihoods $l_i(x)$.

$$p(x|y) \propto \exp(\theta_0^\top s(x)) \prod_{i=1}^{N} \exp(l_i(x))$$

$$= \exp([\theta_0; 1 \dots 1]^\top [s(x); l_1(x) \dots l_N(x)])$$

$$p(x|y) = \exp(\tilde{\theta}^\top \tilde{s}(x) - \tilde{A}(\tilde{\theta}))$$

- Bayesian learning can now be posed as computing the mapping $\tilde{\theta} \mapsto \tilde{\mu}$:

$$\arg\max_{\tilde{\mu} \in \tilde{\mathcal{M}}} \tilde{\theta}^\top \tilde{\mu} - \tilde{A}^*(\tilde{\mu})$$

- $\tilde{A}(\tilde{\theta}) - A(\theta_0)$ is the log marginal probability of data.
- Example: Gaussian exponential family, s(x) = [x; x^2]

$$\tilde{\mu} = [\mathbb{E}_{\tilde{\theta}}[x]; \mathbb{E}_{\tilde{\theta}}[x^2]; \mathbb{E}_{\tilde{\theta}}[\ell_1(x)] \dots \mathbb{E}_{\tilde{\theta}}[\ell_N(x)]]$$

Variational Inference

$$\arg\max_{\tilde{\mu}\in\tilde{\mathcal{M}}} \tilde{\theta}^\top \tilde{\mu} - \tilde{A}^*(\tilde{\mu})$$

- Intractable optimization problem:
  - intractable negative entropy
  - intractable mean domain.
- Variational inference methods approximate both in different ways [Wainwright & Jordan 2008].
  - Mean-field variational inference, variational Bayes [Hinton & van Camp 1993, Beal 2003, many others]
  - Bethe approximation, loopy belief propagation [Frey & MacKay 1997, Murphy et al 1999, Yedidia et al 2001, many others]
  - Expectation propagation [Minka 2001, many others]

Mean-Field Variational Inference

- Target posterior distribution: $p(x|y) = \exp(\tilde{\theta}^\top \tilde{s}(x) - \tilde{A}(\tilde{\theta}))$
- Approximating posterior: $q(x)$

- Want q to be as close as possible to p, measured by KL divergence

$$\begin{aligned}\mathrm{KL}(q\|p) =& \mathbb{E}_q\left[\log q(x) - \log p(x|y)\right] \\ =& \mathbb{E}_q[\log q(x)] - \tilde{\theta}^\top \mathbb{E}_q[\tilde{s}(x)] + \tilde{A}(\tilde{\theta}) \geq 0\end{aligned}$$

$$\mathcal{L}(q) := \tilde{\theta}^\top \mathbb{E}_q[\tilde{s}(x)] - \mathbb{E}_q[\log q(x)] \leq \tilde{A}(\tilde{\theta})$$

- If no constraints on q, equivalent to previous formulation

$$\arg\max_{\tilde{\mu}\in\tilde{\mathcal{M}}} \tilde{\theta}^\top \tilde{\mu} - \tilde{A}^*(\tilde{\mu})$$

- Lower bound on the log marginal data probability:

$$\mathcal{L}(q) - A(\theta_0) \leq \tilde{A}(\tilde{\theta}) - A(\theta_0)$$

Mean-Field Variational Inference

- If q assumed to have some simplifying form, leads to what is typically known as variational inference or variational Bayes.
- Suppose that our model includes latent variables for each observation y_i

$$p(x, y, z) = p(x) \prod_{i=1}^{N} p(z_i) p(y_i | z_i, x)$$

- Posterior over x, z is intractable. Assume variational posterior factorises instead,

$$q(x, z) = q_x(x) q_z(z)$$

$$\begin{aligned} \mathcal{L}(q_x, q_z) &= \tilde{\theta}^\top \mathbb{E}_q[\tilde{s}(x, z)] - \mathbb{E}_q[\log q(x, z)] \\ &= \tilde{\theta}^\top \mathbb{E}_{q_x q_z}[\tilde{s}(x, z)] - \mathbb{E}_{q_x}[\log q_x(x)] - \mathbb{E}_{q_z}[\log q_z(z)] \end{aligned}$$

- To maximise $\mathcal{L}$, alternatively maximize wrt q_x, q_z,

$$q_x(x) \propto \exp(\tilde{\theta}^\top \mathbb{E}_{q_z}[\tilde{s}(x, z)])$$

$$q_z(z) \propto \exp(\tilde{\theta}^\top \mathbb{E}_{q_x}[\tilde{s}(x, z)])$$

Mean-Field Variational Inference

- With model structure: $\tilde{\theta}^\top \tilde{s}(x,z) = \theta_0^\top s(x) + \sum_{i=1}^N \underbrace{\log p(z_i, y_i|x)}_{\eta(z_i,y_i)^\top s(x)}$
- Updates become:

$$q_z(z) \propto \exp(\tilde{\theta}^\top \mathbb{E}_{q_x}[\tilde{s}(x,z)]) = \left(\sum_{i=1}^N \mathbb{E}_{q_x}[\log p(z_i, y_i|x)]\right)$$

$$q_z(z_i) \propto \exp\left(\eta(z_i,y_i)^\top \mathbb{E}_{q_x}[s(x)]\right)$$

$$q_x(x) \propto \exp(\tilde{\theta}^\top \mathbb{E}_{q_z}[\tilde{s}(x,z)]) = \exp\left(\theta_0^\top s(x) + \sum_{i=1}^N \mathbb{E}_{q_z}[\log p(z_i, y_i|x)]\right)$$

$$\propto \exp\left(\left(\theta_0 + \sum_{i=1}^N \mathbb{E}_{q_z}[\eta(z_i,y_i)]\right)^\top s(x)\right)$$

Stochastic Variational Inference

- When N>>1, updates are expensive as each iteration requires computations on all observations.

- Say the variational posterior of x is parameterised as $q_x(x) \propto \exp\left(\lambda^\top s(x)\right)$

- One can instead optimise $\mathcal{L}$ wrt λ using stochastic natural gradient ascent [Robbins & Monro 1951, Bottou 1996], [Amari 1998, Sato 2001].

[Hoffman et al 2010, Mimno et al 2012]

Stochastic Gradient Optimisation

- Given an objective function f(x) to be maximized.
- Stochastic gradient ascent:

$$x_{t+1} = x_t + \frac{\epsilon_t}{2}\nabla_x f(x_t) \approx x_t + \frac{\epsilon_t}{2}\widehat{\nabla}_x f(x_t)$$

- With unbiased and finite variance gradient estimates

$$\mathbb{E}[\widehat{\nabla}_x f(x)] = \nabla_x f(x_t) \qquad \mathbb{V}[\widehat{\nabla}_x f(x)] \leq V_0$$

- Convergent with step size condition

$$\sum_{t=1}^{\infty} \epsilon_t = \infty \qquad \sum_{t=1}^{\infty} \epsilon_t^2 < \infty$$

- [Robbins & Monro 1951]

Stochastic Gradient Ascent for Maximum a Posteriori

- Joint log probability is

$$f(x) = \log p(x) + \sum_{i=1}^{N} \log p(y_i|x)$$

- To find x^{MAP}, use stochastic gradient ascent

$$\nabla f(x) = \nabla \log p(x) + \sum_{i=1}^{N} \nabla \log p(y_i|x)$$

$$\widehat{\nabla} f(x) = \nabla \log p(x) + \frac{N}{n} \sum_{j=1}^{n} \nabla \log p(y_{\tau_j}|x)$$

$$x_{t+1} = x_t + \epsilon_t \widehat{\nabla} f(x)$$

- See [Bottou 1996] and many others.

Stochastic Variational Inference

- The variational posterior of x is parameterised as $q_x(x) \propto \exp\left(\lambda^\top s(x)\right)$ with mean parameter $\gamma = \mu(\lambda) = \nabla A(\lambda)$.
- Variational objective is

$$\begin{aligned}\mathcal{L} &= \mathbb{E}_q\left[\theta_0^\top s(x) + \textstyle\sum_{i=1}^N \eta(z_i, y_i)^\top s(x) - \log q(x, z)\right] \\ &= \gamma^\top\left(\theta_0 + \textstyle\sum_{i=1}^N \mathbb{E}_{q_{z_i}}[\eta(z_i, y_i)]\right) - A^*(\gamma) - \textstyle\sum_{i=1}^N \mathbb{E}_{q_{z_i}}[\log q_{z_i}(z_i)]\end{aligned}$$

- Gradient is

$$\nabla_\lambda \mathcal{L} = \nabla^2_\lambda A(\lambda)\left(\theta_0 + \textstyle\sum_{i=1}^N \mathbb{E}_{q_{z_i}}[\eta(z_i, y_i)]\right) - \nabla^2_\lambda A(\lambda)\lambda$$

- Stochastic natural gradient is

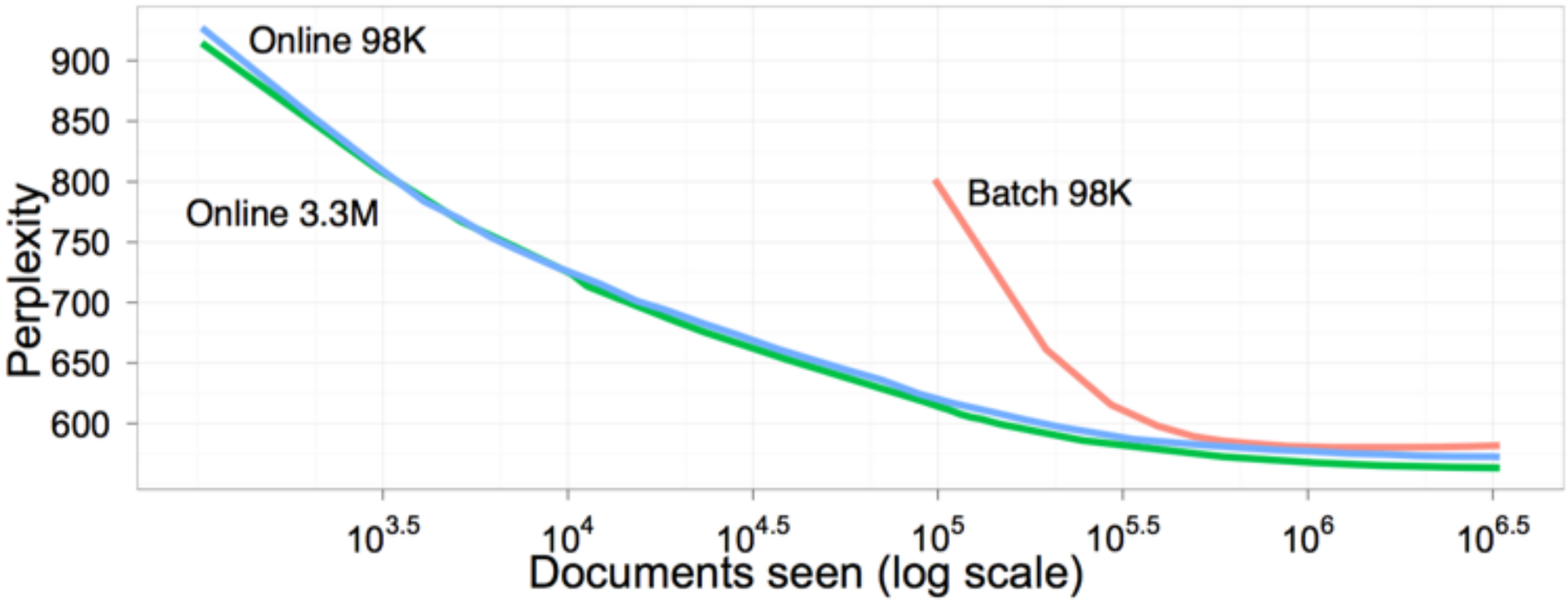
$$\nabla^2_\lambda A(\lambda)^{-1}\widehat{\nabla_\lambda}\mathcal{L} = \left(\theta_0 + \tfrac{N}{n}\textstyle\sum_{j=1}^n \mathbb{E}_{q_{z_{\tau_j}}}[\eta(z_{\tau_j}, y_{\tau_j})]\right) - \lambda$$

- Update:

$$\lambda^{\text{new}} = (1-\epsilon)\lambda + \epsilon\left(\theta_0 + \tfrac{N}{n}\textstyle\sum_{j=1}^n \mathbb{E}_{q_{z_{\tau_j}}}[\eta(z_{\tau_j}, y_{\tau_j})]\right)$$

[Hoffman et al 2013, Mimno et al 2012]

Example: Latent Dirichlet Allocation



Documents analyzed	2048	4096	8192	12288	16384	32768	49152	65536
Top eight words	systems	systems	service	service	service	business	business	business
	road	health	systems	systems	companies	service	service	industry
	made	communication	health	companies	systems	companies	companies	service
	service	service	companies	business	business	industry	industry	companies
	announced	billion	market	company	company	company	services	services
	national	language	communication	billion	industry	management	company	company
	west	care	company	health	market	systems	management	management
	language	road	billion	industry	billion	services	public	public

[Hoffman et al 2013, Mimno et al 2012]

1	2	3	4	5
Game	Life	Film	Book	Wine
Season	Know	Movie	Life	Street
Team	School	Show	Books	Hotel
Coach	Street	Life	Novel	House
Play	Man	Television	Story	Room
Points	Family	Films	Man	Night
Games	Says	Director	Author	Place
Giants	House	Man	House	Restaurant
Second	Children	Story	War	Park
Players	Night	Says	Children	Garden

6	7	8	9	10
Bush	Building	Won	Yankees	Government
Campaign	Street	Team	Game	War
Clinton	Square	Second	Mets	Military
Republican	Housing	Race	Season	Officials
House	House	Round	Run	Iraq
Party	Buildings	Cup	League	Forces
Democratic	Development	Open	Baseball	Iraqi
Political	Space	Game	Team	Army
Democrats	Percent	Play	Games	Troops
Senator	Real	Win	Hit	Soldiers

11	12	13	14	15
Children	Stock	Church	Art	Police
School	Percent	War	Museum	Yesterday
Women	Companies	Women	Show	Man
Family	Fund	Life	Gallery	Officer
Parents	Market	Black	Works	Officers
Child	Bank	Political	Artists	Case
Life	Investors	Catholic	Street	Found
Says	Funds	Government	Artist	Charged
Help	Financial	Jewish	Paintings	Street
Mother	Business	Pope	Exhibition	Shot

[Hoffman et al 2013, Mimno et al 2012]

Generic (Bayesian) Learning on Big Data

- Stochastic optimisation using mini-batches.
  - Stochastic gradient optimisation.
  - Stochastic variational inference [Hoffman et al 2013, Mimno et al 2012]
  - **Stochastic Gradient MCMC [Welling & Teh 2011, Patterson & Teh 2013, Teh et al 2016, Chen et al 2014, Ma et al 2015, Din et al 2015, Leimkuhler & Shang 2015…]**

- Distributed/parallel computations on cores/clusters/GPUs.
  - MapReduce, parameter server.
  - Embarassingly Parallel MCMC [[Scott et al 2013, Neiswanger et al 2013, Wang & Dunson 2013, Stanislav et al 2014]
  - Sampling via Moment Sharing [Xu et al 2014]
  - Stochastic Natural-gradient EP and Posterior Server [Teh et al 2016]

Variational Inference and Markov chain Monte Carlo

- Variational inference expresses posterior computation as optimisation of an approximate system.
- Access to large body of optimisation methods.
- Approximation error hard to quantify.

- Monte Carlo methods express posterior computation as random sampling.
- Markov chain Monte Carlo: posterior as the stationary distribution.
- Typically more expensive but more accurate.
- Exact asymptotically
- Approximation error also hard to quantify given finite computation.

- Variance vs bias vs computation

Random Walk Metropolis

- Current state x_t
- Proposed state $x^* \sim \mathcal{N}(x_t, \epsilon)$
- Accept proposal with probability

$$\min\left(1, \frac{p(x^*, y)}{p(x_t, y)}\right)$$

- Many, many advances since [Metropolis et al 1953, Hastings 1970].

- Big data: acceptance ratio expensive to compute.
- Random walk behaviour mixes very inefficiently.

Metropolis Adjusted Langevin Algorithm

- Use local gradient information to improve proposal distribution [Roberts & Tweedie 1996]

$$x^* \sim x_t + \frac{\epsilon}{2}\nabla_x \log p(x_t, y) + \mathcal{N}(0, \epsilon)$$

- Obtained as Euler-Maruyama discretisation of (overdamped) Langevin dynamics:

$$dx_t = \frac{1}{2}\nabla_x \log p(x_t, y)dt + dW_t$$

- Big data: both acceptance ratio and proposal distribution expensive to compute.

Stochastic Gradient Optimisation

- Proposal update very similar to stochastic gradient ascent:

$$\begin{aligned} x_{t+1} &= x_t + \frac{\epsilon_t}{2}\nabla_x \log p(x_t, y) \\ &= x_t + \frac{\epsilon_t}{2}\left(\nabla_x \log p(x_t) + \sum_{i=1}^{N} \nabla_x \log p(y_i|x_t)\right) \\ &\approx x_t + \frac{\epsilon_t}{2}\left(\nabla_x \log p(x_t) + \frac{N}{n}\sum_{j=1}^{n} \nabla_x \log p(y_{\tau_j}|x_t)\right) \end{aligned}$$

- Convergent with step size condition

$$\sum_{t=1}^{\infty} \epsilon_t = \infty \qquad \sum_{t=1}^{\infty} \epsilon_t^2 < \infty$$

Stochastic Gradient Langevin Dynamics

- Plug in stochastic gradient into Metropolis adjusted Langevin algorithm

$$x_{t+1} = x_t + \frac{\epsilon_t}{2}\left(\nabla_x \log p(x_t) + \frac{N}{n}\sum_{j=1}^{n} \nabla_x \log p(y_{\tau_j}|x_t)\right) + \mathcal{N}(0, \epsilon_t)$$

- Ignore Metropolis-Hastings acceptance step (!)
- Step-size requirements apply, $\varepsilon_t \to 0$ slowly.

- Two sources of noise:
  - Injected Brownian noise
  - Gradient noise

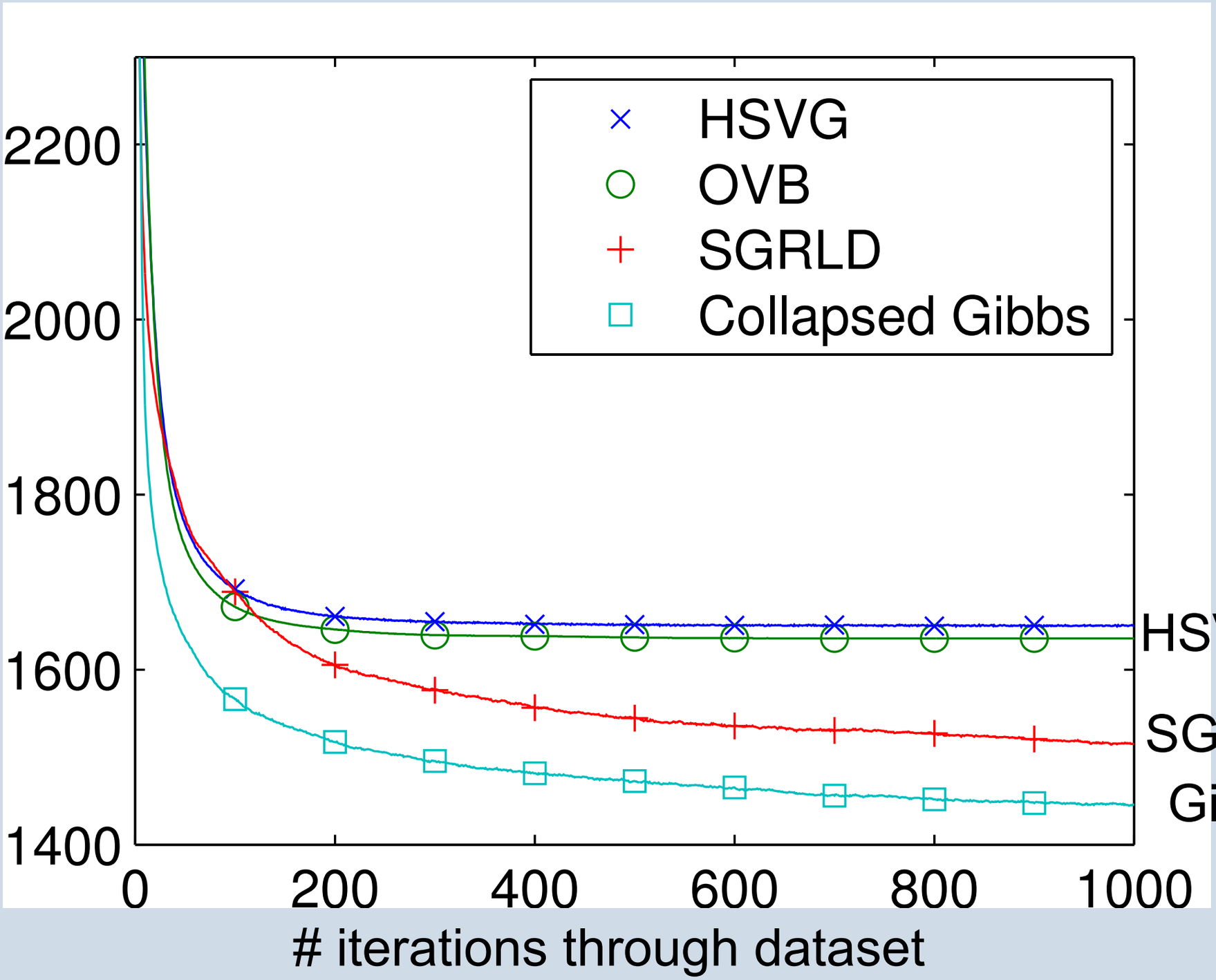
Stochastic Gradient Langevin Dynamics

$$x_{t+1} = x_t + \frac{\epsilon_t}{2}\left(\nabla_x \log p(x_t) + \frac{N}{n}\sum_{j=1}^{n} \nabla_x \log p(y_{\tau_j}|x_t)\right) + \mathcal{N}(0, \epsilon_t)$$

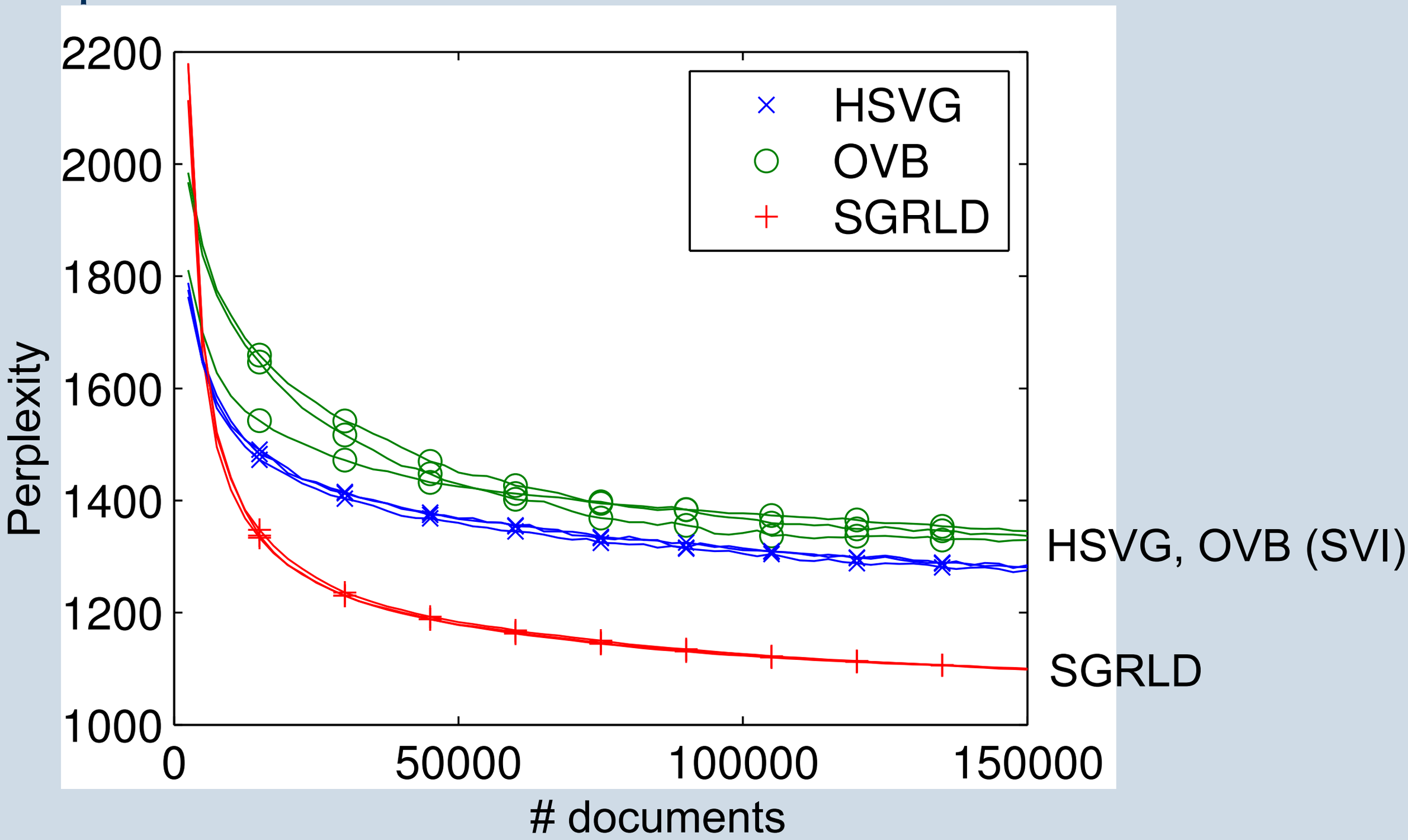
- As $\varepsilon t \to 0$:
  - Variance of gradient noise is $O(\varepsilon_t^2)$ while variance of injected noise is $\varepsilon_t >> \varepsilon_t^2$.
  - MH acceptance probability approaches 1, so we can ignore the expensive MH accept/reject step.
  - $\varepsilon_t \to 0$ slowly enough, so dynamics still able to explore whole parameter space.

- Teh et al (2016), Vollmer et al (forthcoming) more detailed analysis.
  - $O(t^{-1/3})$ convergence rate.
  - Not due to decreasing step size, rather due to lack of MH correction.

Example: Latent Dirichlet Allocation



Example: Latent Dirichlet Allocation



Convergence with Decreasing Step Sizes

- When using decreasing step sizes $\varepsilon_t \to 0$, a central limit theorem for SGLD can be derived. Let $T_t = \sum_{s \le t} \varepsilon_s$.
  - When fluctuations dominates,
$$\lim_{t\to\infty} T_t^{1/2}\{\mathbb{E}_{p_t}[\varphi] - \mathbb{E}_p[\varphi]\} = \mathcal{N}(0, \sigma^2(\varphi))$$
  - When bias dominates,
$$\lim_{t\to\infty} T_t^{1/2}\{\mathbb{E}_{p_t}[\varphi] - \mathbb{E}_p[\varphi]\} = \mathcal{N}(\mu(\varphi), \sigma^2(\varphi))$$
  - When fluctuations and bias balanced,
$$\lim_{t\to\infty} \frac{\{\mathbb{E}_{p_t}[\varphi] - \mathbb{E}_p[\varphi]\}}{T_t^{-1}\sum_{s\le t}\epsilon_s^2} = \mu(\varphi)$$
- Optimal step size sequence has form $\varepsilon_t = (t_0+t)^{-1/3}$ with $T_t \asymp t^{2/3}$.
- [Teh et al JMLR 2016]

Convergence with Constant Step Sizes

- t steps of SGLD with constant step size ε, generating $x_1,\ldots x_t$.
- Estimator:

$$\hat{\varphi}_t = \sum_{s \leq t} \varphi(x_s) \qquad\qquad \bar{\varphi} = \mathbb{E}_p[\varphi]$$

- Weak analysis Bias:

$$|\mathbb{E}[\hat{\varphi}_t] - \bar{\varphi}| = O\left(\epsilon + \frac{1}{t\epsilon}\right)$$

- Variance:

$$\mathbb{E}[(\hat{\varphi}_t] - \mathbb{E}[\hat{\varphi}_t])^2] = O\left(\epsilon^2 + \frac{1}{t\epsilon}\right)$$

- Optimal ε gives MSE of $O(t^{-2/3})$.
- [Vollmer et al (forthcoming)]

Stochastic Gradient MCMC

- SGLD obtained as discretisation of overdamped Langevin dynamics.
- Alternative SGMCMC algorithms can be constructed by
  - constructing a SDE with the posterior as the stationary distribution
  - discretising time in some way.

- Riemannian SGLD for probability simplices [Patterson & Teh 2013]
- Stochastic gradient Hamiltonian Monte Carlo [Chen et al 2014]
- Stochastic gradient Nose-Hoover thermostats [Din et al 2015, Leimkuhler & Shang 2015]

A Complete Recipe for SGMCMC

- [Ma et al 2015] showed a complete recipe for all SDEs with a desired stationary distribution p(x).
- I.e. any SDE with stationary distribution p(x) has the form

$$dx_t = ([D(x_t) + Q(x_t)]\nabla \log p(x_t) + \Gamma(x_t))dt + \sqrt{2D(x_t)}dW_t$$

$D(x)$: a symmetric positive definite diffusion matrix

$Q(x)$: a skew-symmetric curl matrix

$$\Gamma_j(x) = \sum_{k=1}^{d} \nabla_{x_k}(D_{jk}(x) + Q_{jk}(x)): \text{ a correction factor}$$

W_t : Brownian motion

- Large relevant literature in applied mathematics.

Stochastic Gradient Hamiltonian Monte Carlo

- A naive generalisation of SGLD to use Hamiltonian dynamics would be to introduce a momentum variable ϱ.

$$x_{t+1} = x_t + \epsilon_t M^{-1} \rho_t$$
$$\rho_{t+1} = \rho_t + \epsilon_t \widehat{\nabla}_x \log p(x_t, y)$$

- Does not fit into framework of [Ma et al 2015].
- Instead, need to introduce a friction term

$$x_{t+1} = x_t + \epsilon_t M^{-1} \rho_t$$
$$\rho_{t+1} = \rho_t + \epsilon_t \widehat{\nabla}_x \log p(x_t, y) - \epsilon D M^{-1} \rho_t + \mathcal{N}(0, \epsilon_t (2D - \epsilon_t \widehat{V}(x_t)))$$

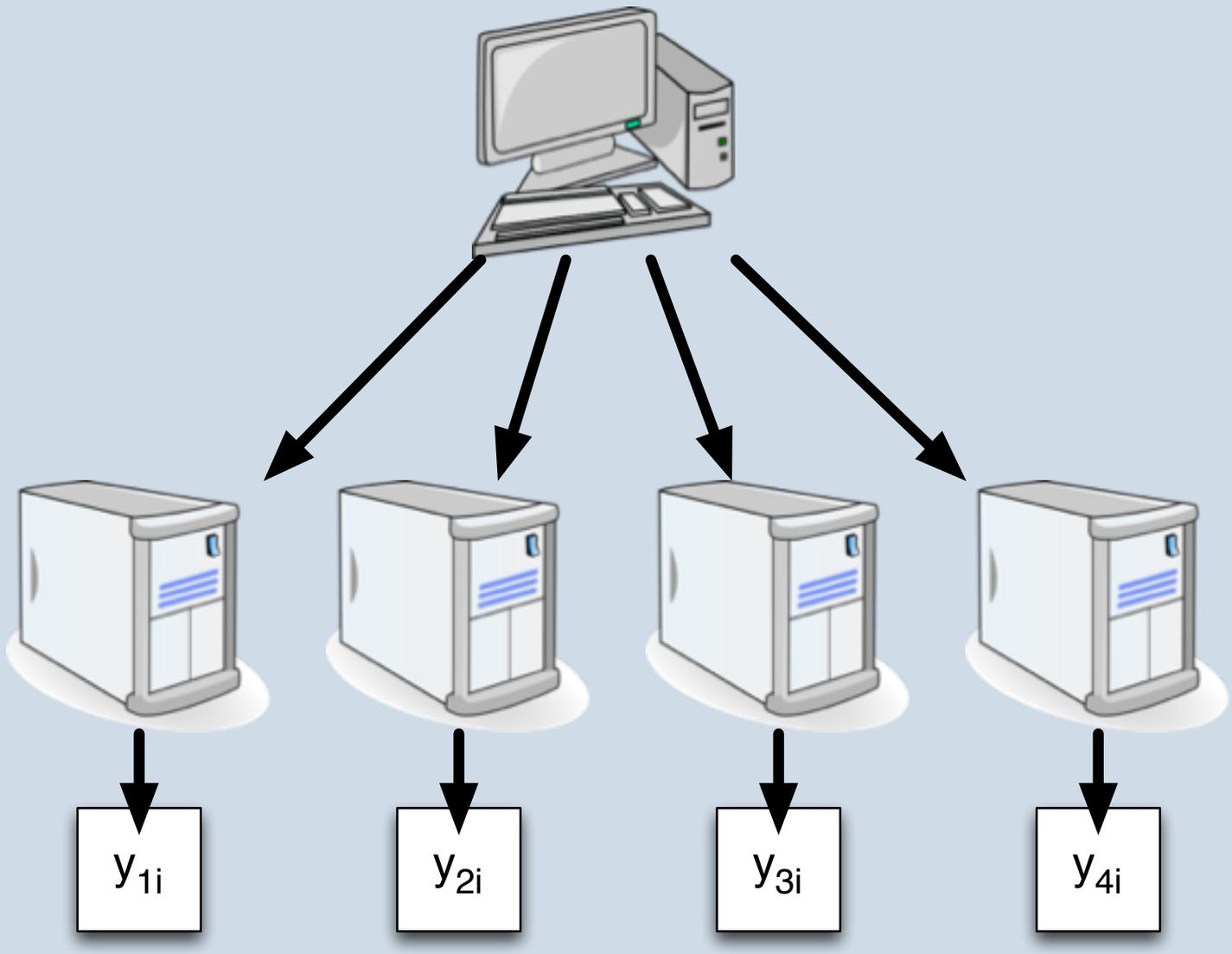
- [Chen et al 2014]

Generic (Bayesian) Learning on Big Data

- Stochastic optimisation using mini-batches.
  - Stochastic gradient optimisation.
  - Stochastic variational inference [Hoffman et al 2013, Mimno et al 2012]
  - Stochastic Gradient MCMC [Welling & Teh 2011, Patterson & Teh 2013, Teh et al 2016, Chen et al 2014, Ma et al 2015, Din et al 2015, Leimkuhler & Shang 2015…]

- Distributed/parallel computations on cores/clusters/GPUs.
  - **MapReduce, parameter server.**
  - **Embarassingly Parallel MCMC [[Scott et al 2013, Neiswanger et al 2013, Wang & Dunson 2013, Stanislav et al 2014]**
  - Sampling via Moment Sharing [Xu et al 2014]
  - Stochastic Natural-gradient EP and Posterior Server [Teh et al 2016]

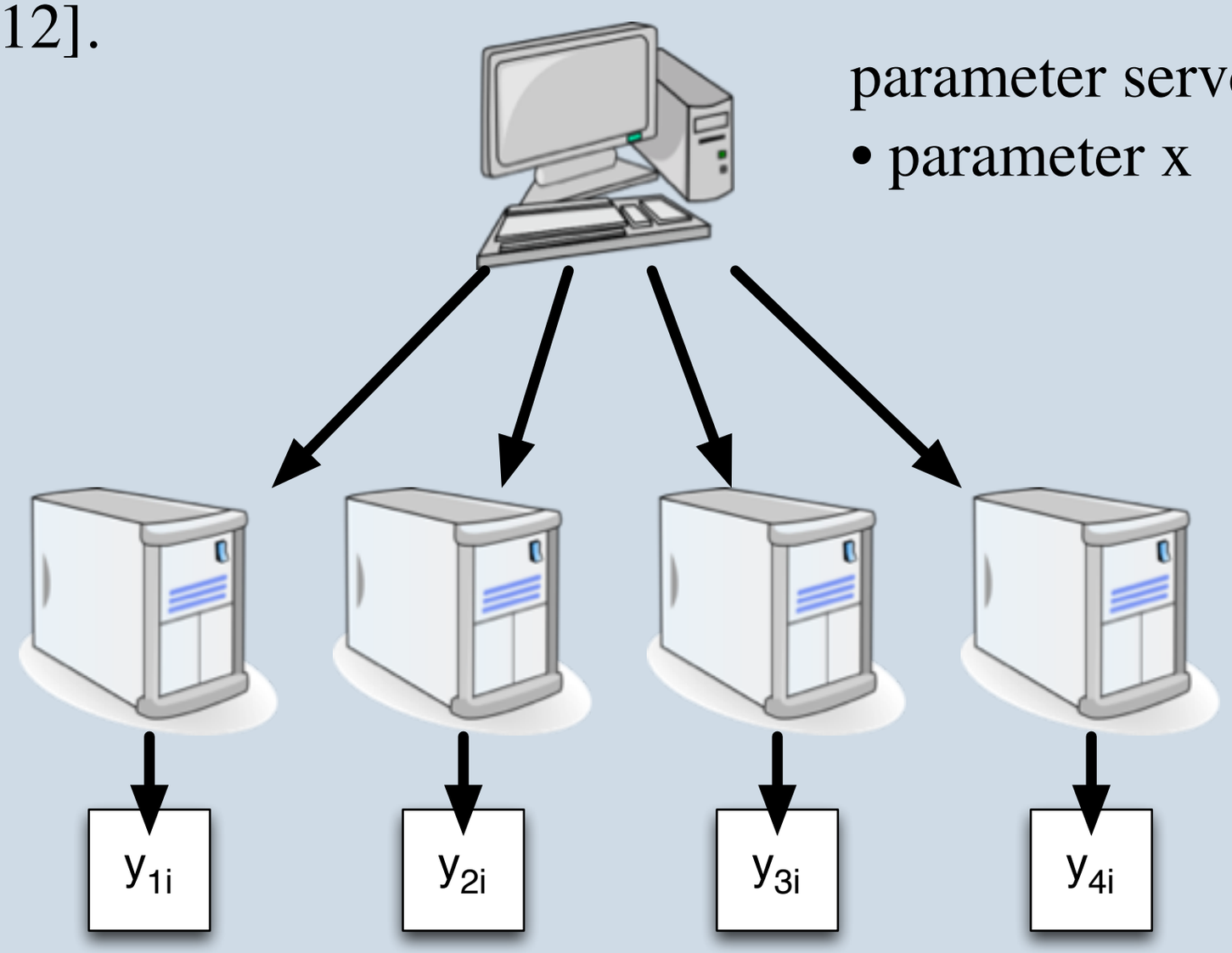
Machine Learning on Distributed Systems



- Distributed storage
- costly network communications
- Distributed computation

Parameter Server

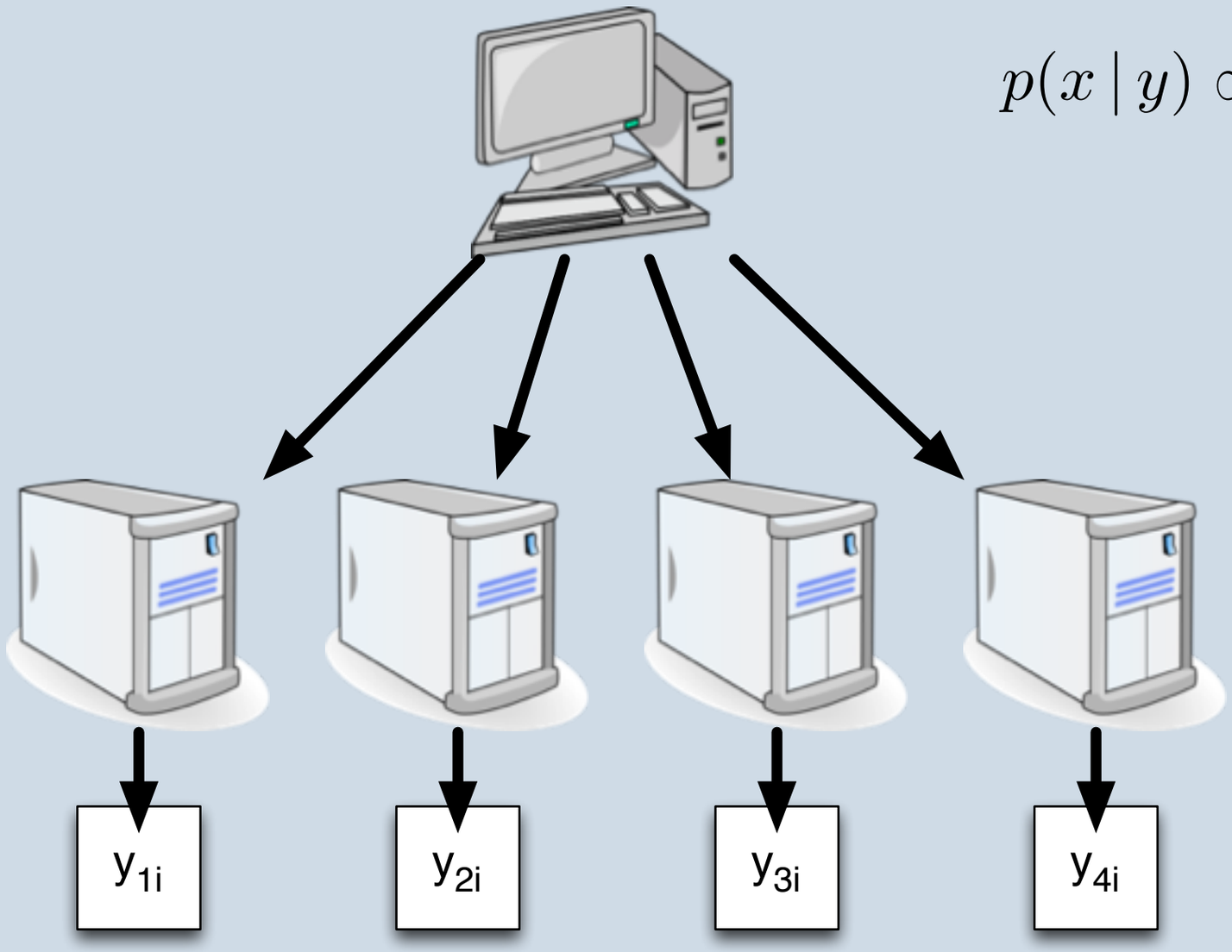
- Parameter server [Ahmed et al 2012], Downpour/DistBelief [Dean et al 2012].



parameter server:
- parameter x

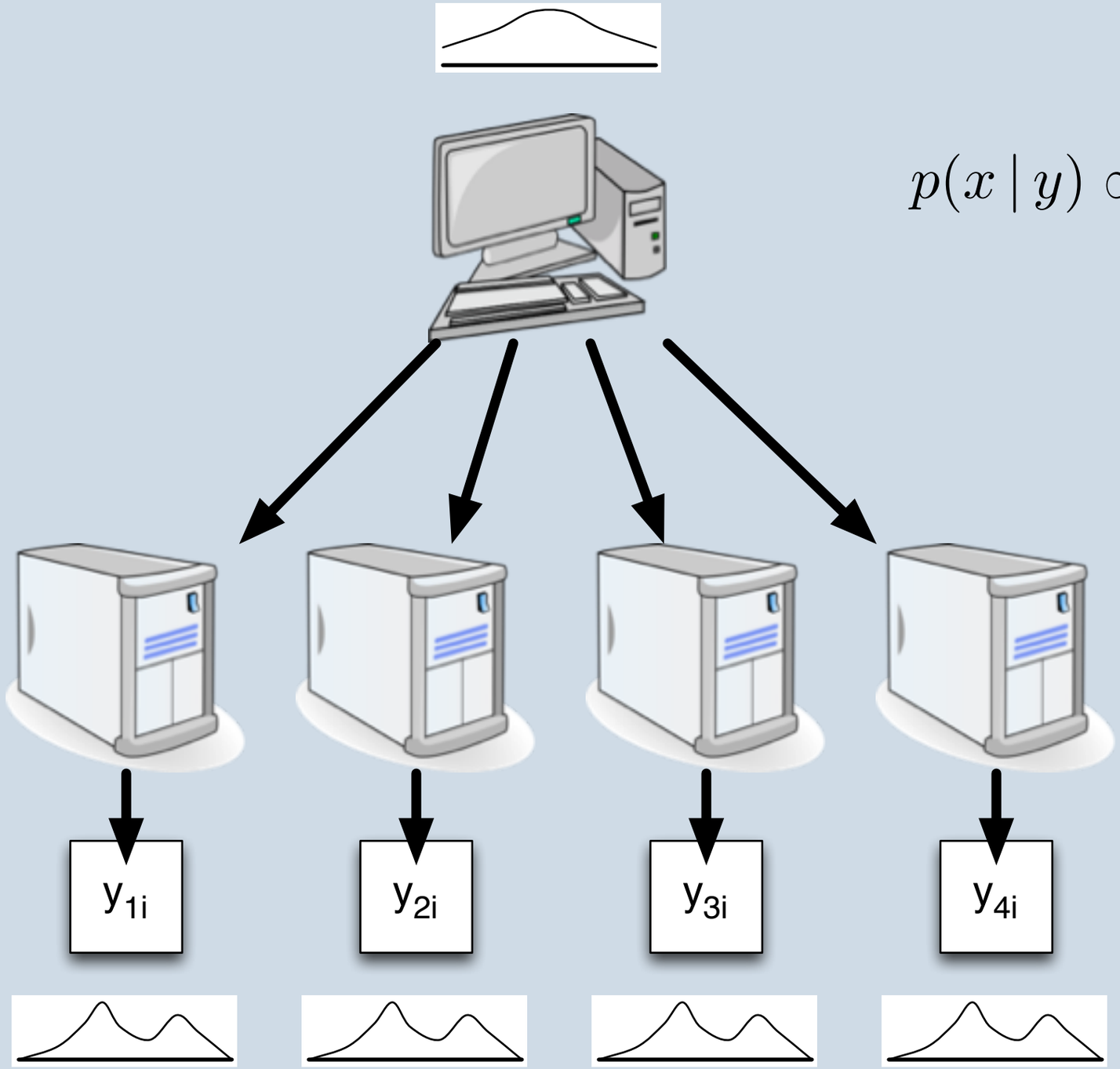
worker:
- $x_i = x$
- SGD updates to x_i'
- returns $\Delta x_i = x_i' - x_i$

Bayesian Learning



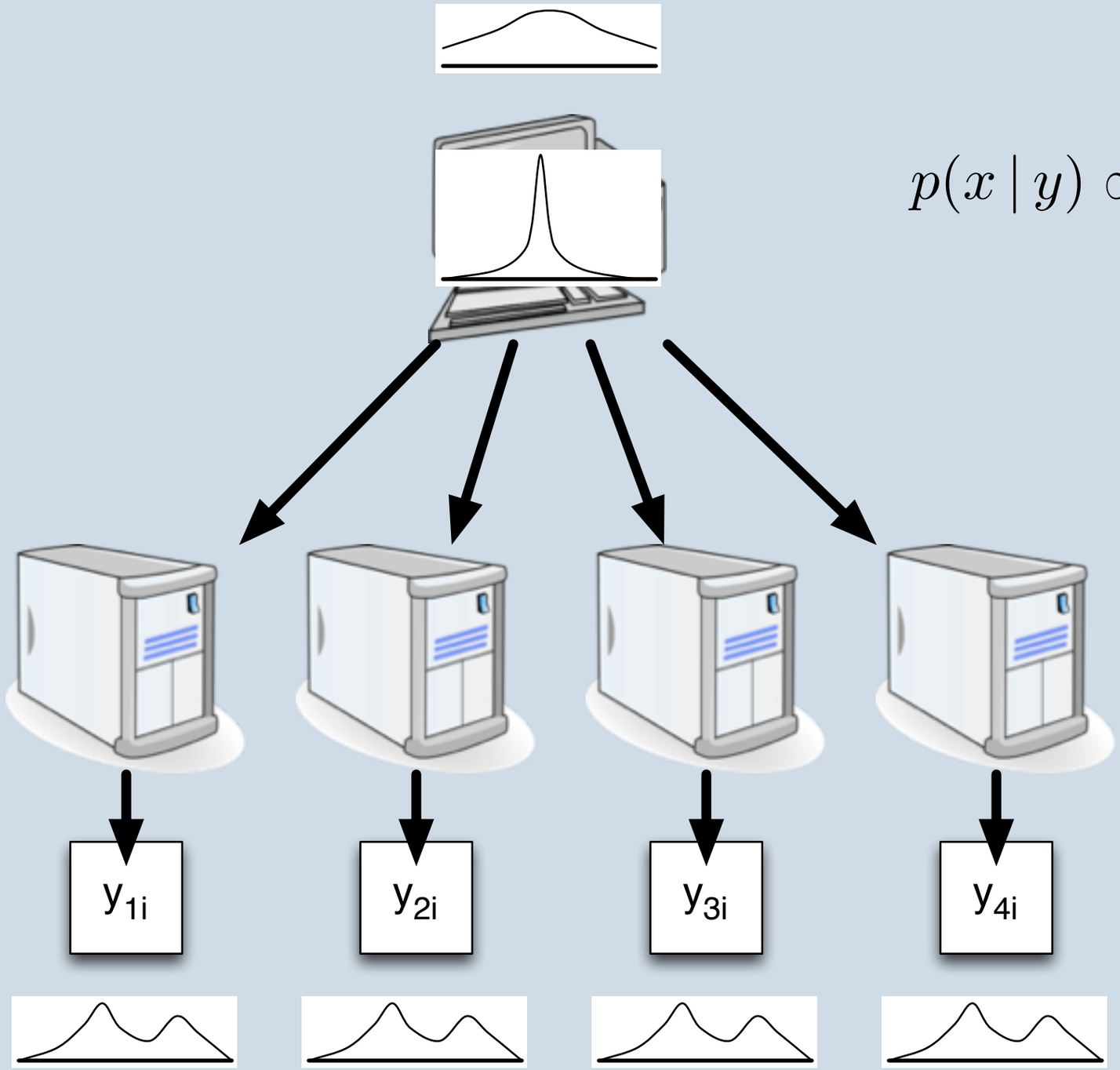
$$p(x \mid y) \propto p(x) \prod_{j=1}^{m} \prod_{i=1}^{I} p(y_{ji} \mid x)$$

Bayesian Learning



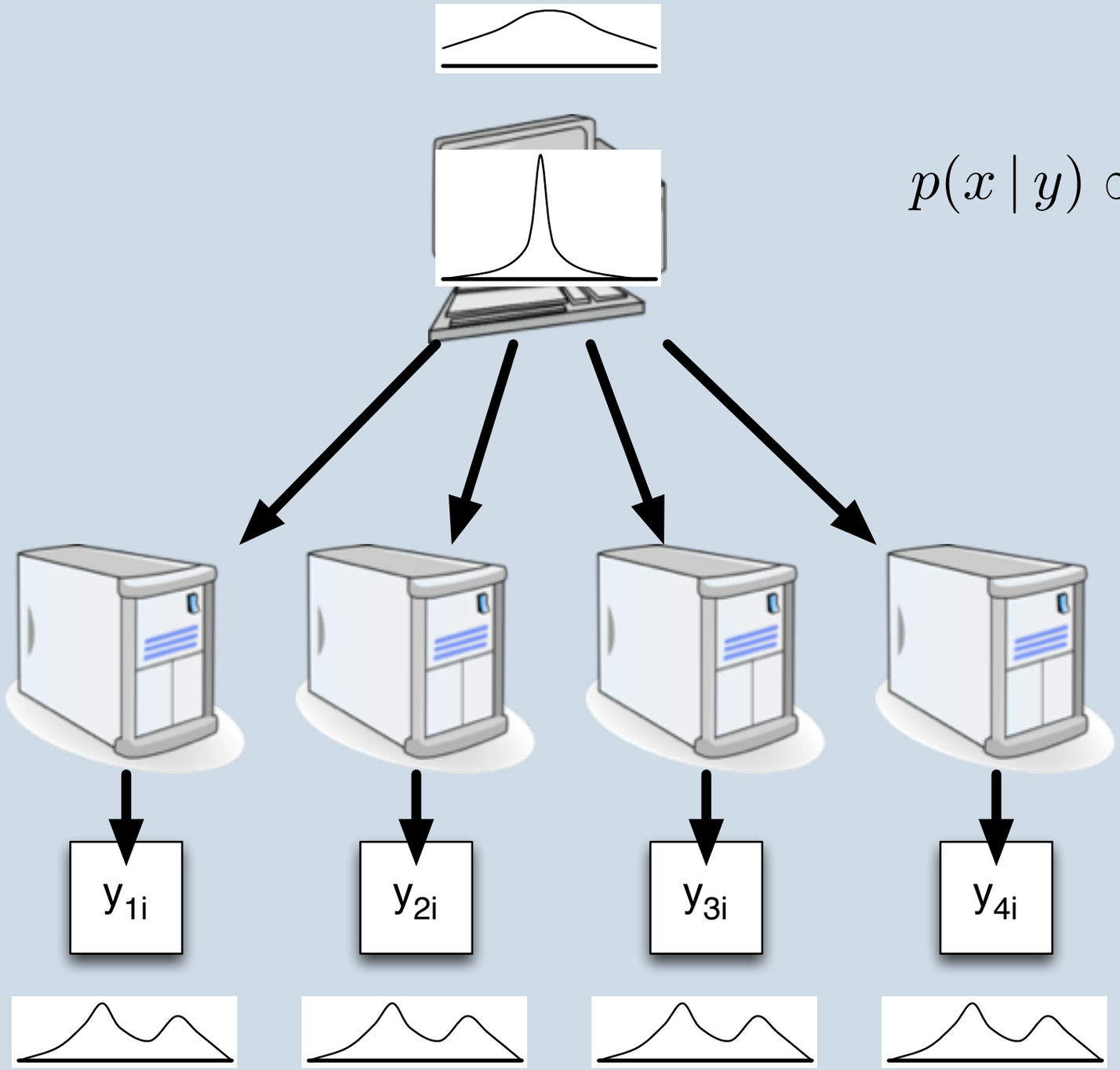
$$p(x \mid y) \propto p(x) \prod_{j=1}^{m} \prod_{i=1}^{I} p(y_{ji} \mid x)$$

Bayesian Learning



$$p(x \mid y) \propto p(x) \prod_{j=1}^{m} \prod_{i=1}^{I} p(y_{ji} \mid x)$$

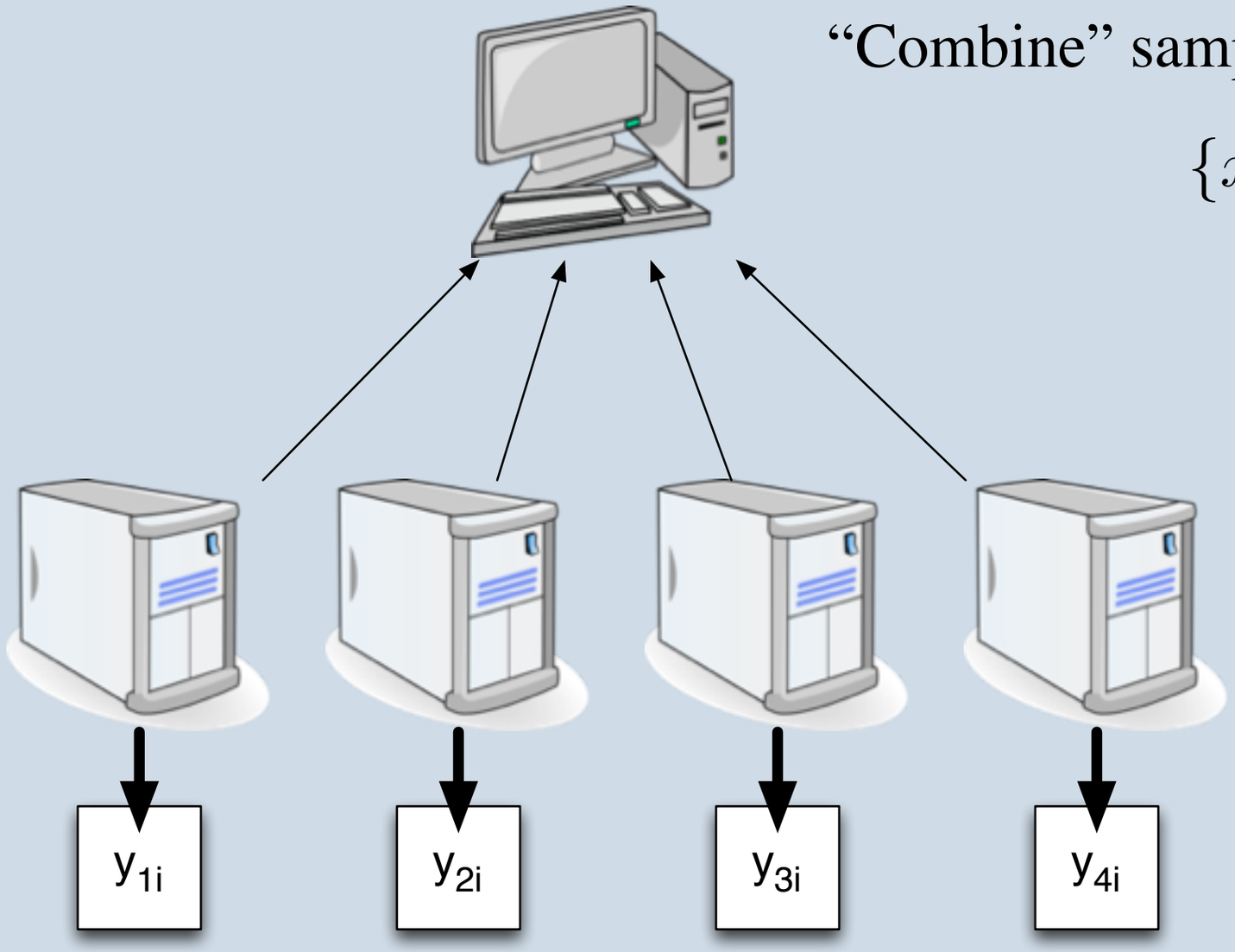
Bayesian Learning



$$p(x \mid y) \propto p(x) \prod_{j=1}^{m} \prod_{i=1}^{I} p(y_{ji} \mid x)$$

- Not feasible exactly.
- Approximations:
  - Monte Carlo sampling
  - Variational inference

Embarassingly Parallel MCMC Sampling



[Scott et al 2013, Neiswanger et al 2013, Wang & Dunson 2013, Stanislav et al 2014]

"Combine" samples together.

$$\{x_s\}_{s=1\ldots S}$$

Treat as independent inference problems. Collect samples.

$$\{x_{js}\}_{j=1\ldots m, s=1\ldots S}$$

- Only communication at the combination stage.

Consensus Monte Carlo

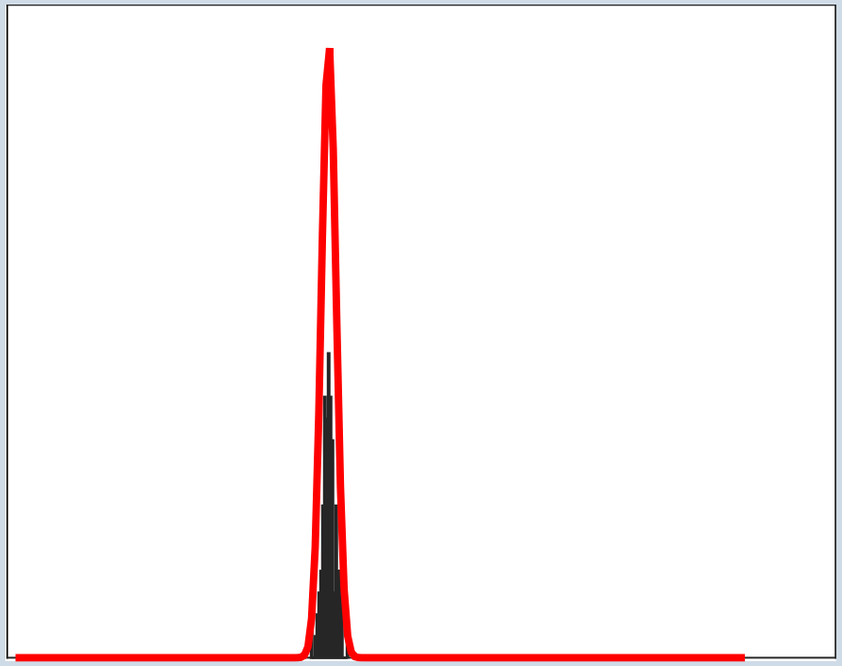
- Each worker machine j collects S samples $\{x_{js}\}$ from:

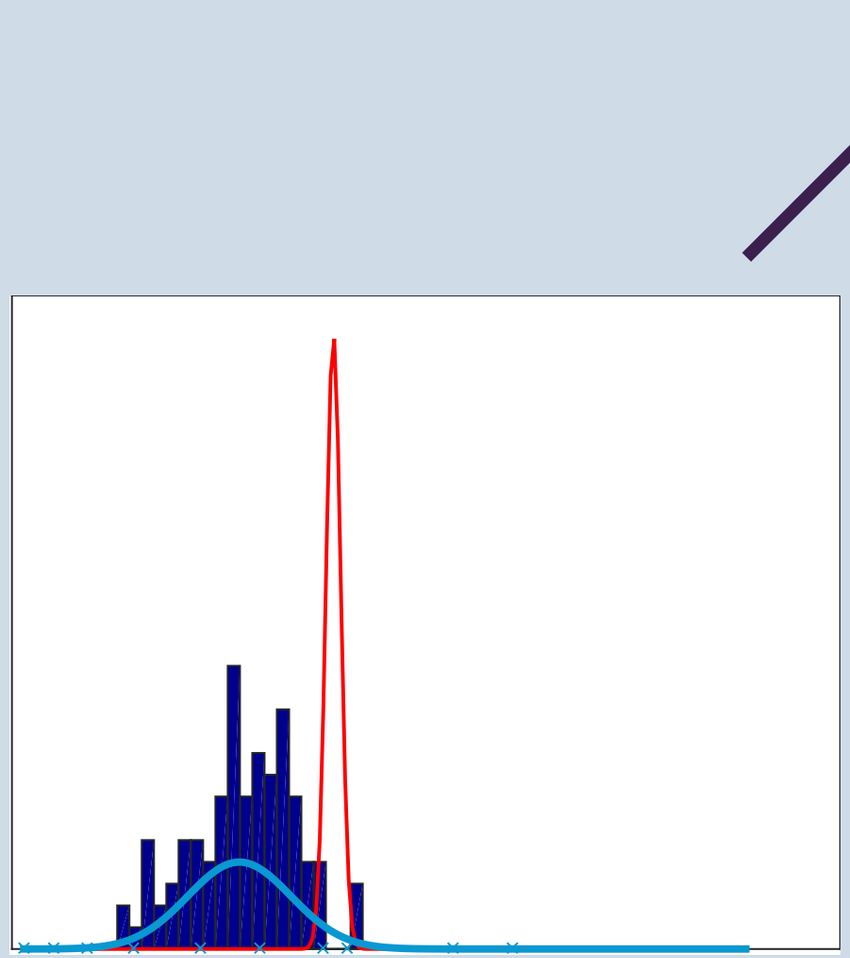
$$p_j(x \mid y_j) = p(x)^{1/m} \prod_{i=1}^{I} p(y_{ji}|x)$$

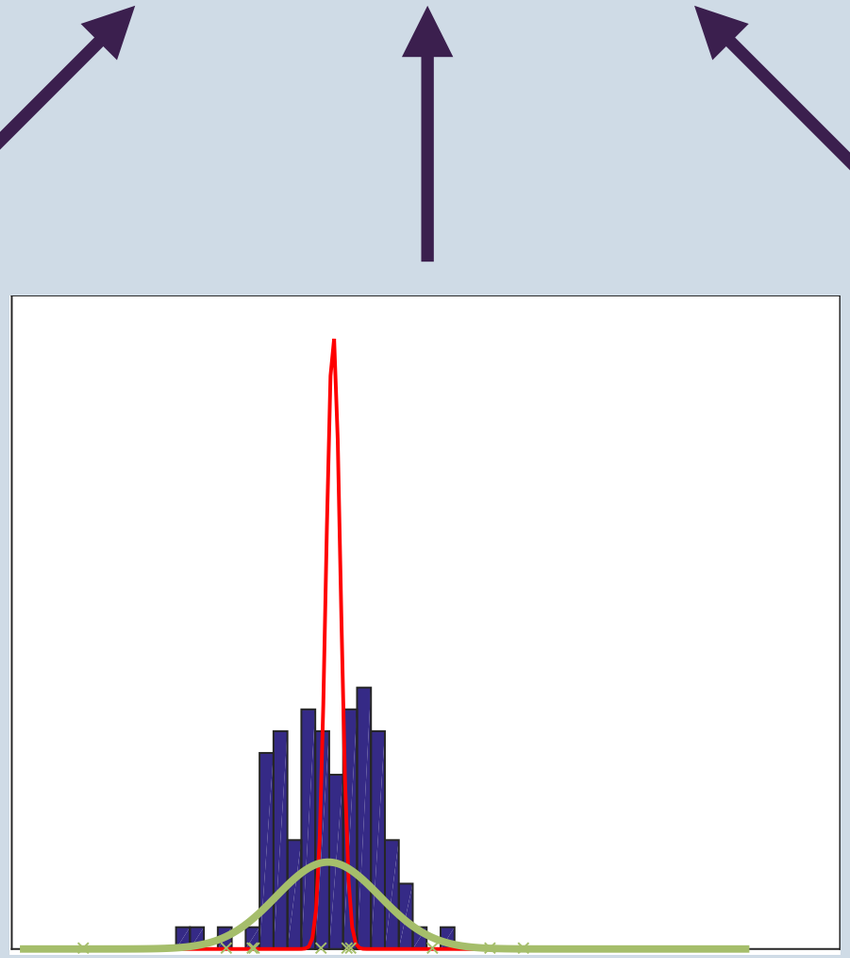
- Master machine combines samples by weighted average:

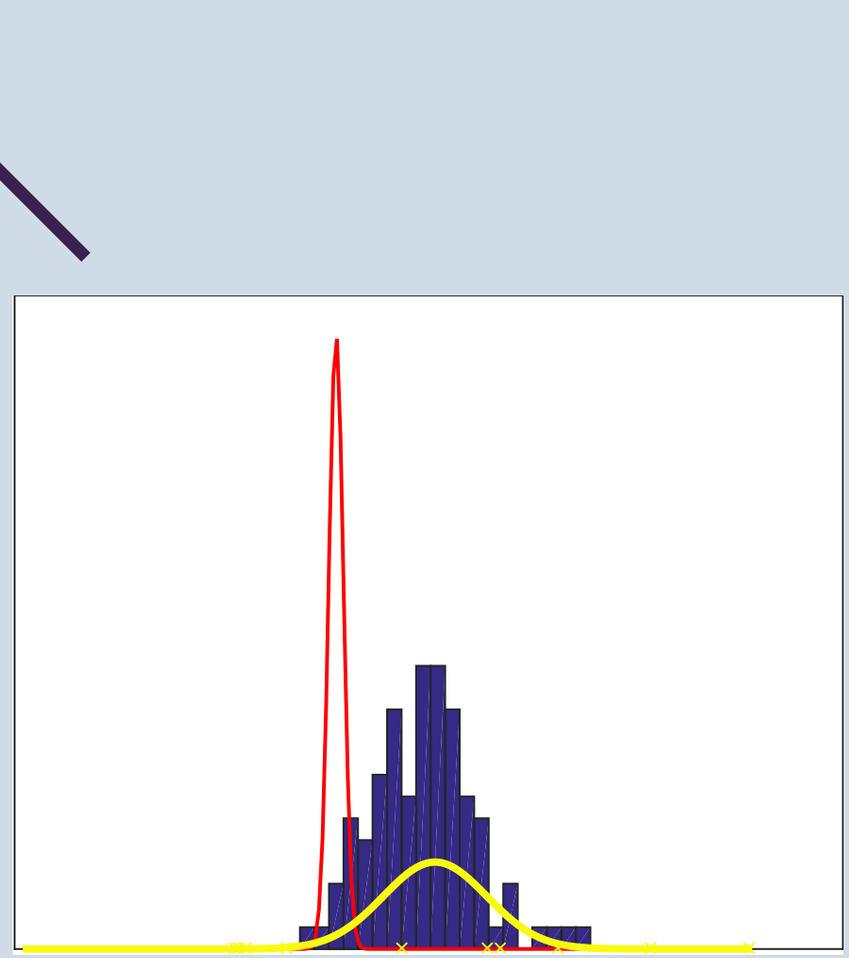
$$x_s = \left(\sum_{j=1}^{m} W_j \right)^{-1} \sum_{j=1}^{m} W_j x_{js}$$

[Scott et al 2013]









Consensus Monte Carlo

$$x_s = \left(\sum_{j=1}^{m} W_j \right)^{-1} \sum_{j=1}^{m} W_j x_{js}$$

- Combination is correct if local posteriors are Gaussian.
- Weights are local posterior precisions.
- If not Gaussian, unclear how this can work.

[Scott et al 2013]

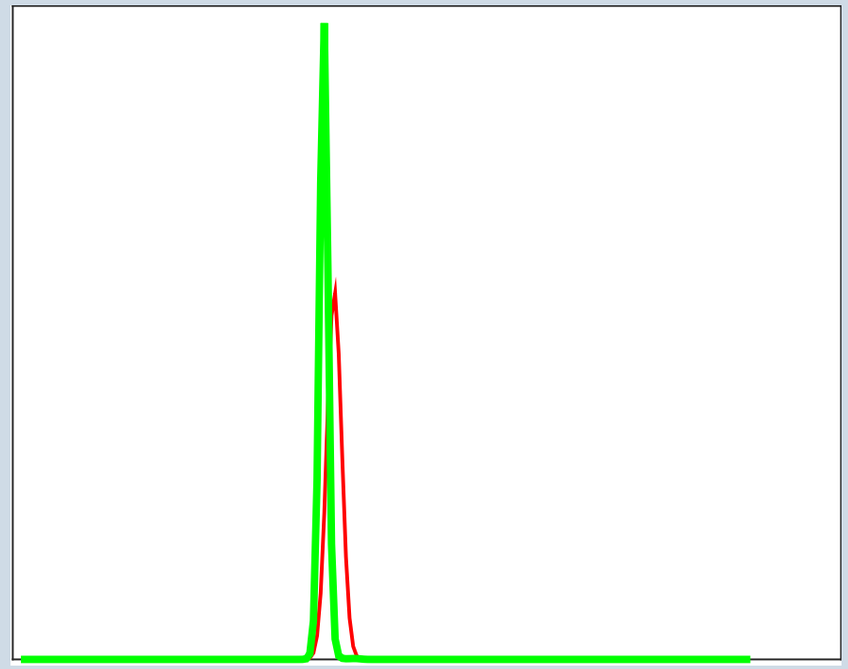
Approximating Local Posterior Densities

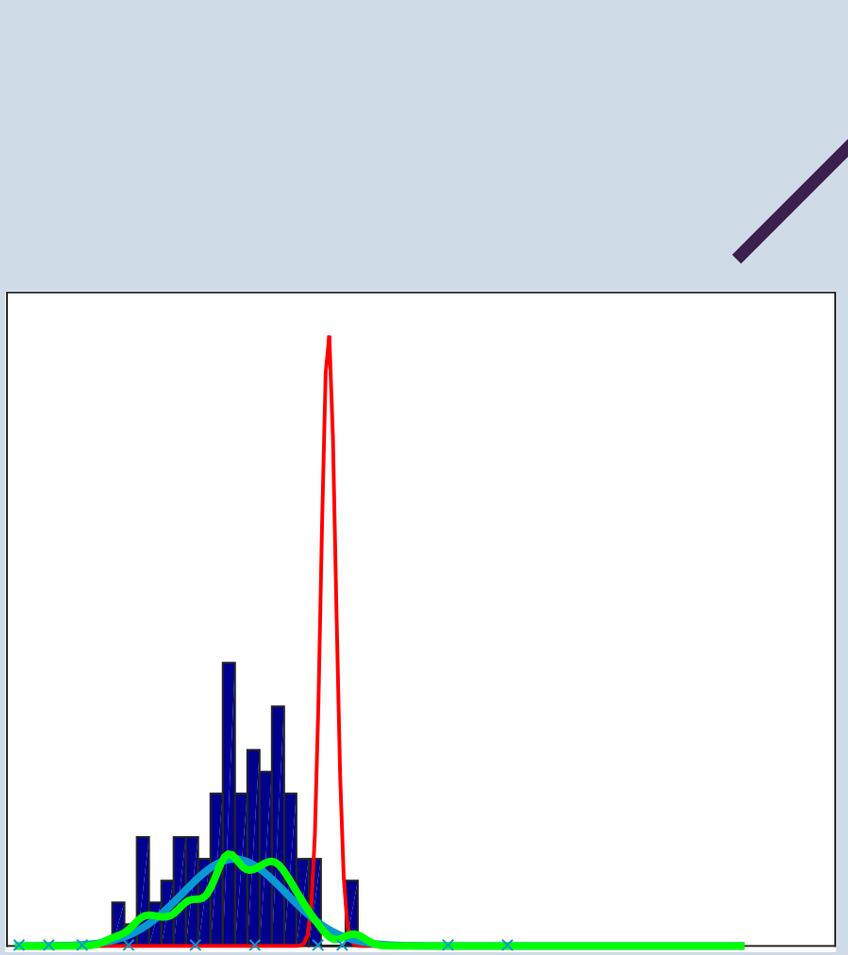
- [Neiswanger et al 2013] proposed methods to combine estimates of local posterior densities instead of samples:
  - Parametric: Gaussian approximation.
  - Nonparametric: kernel density estimation based on samples.
  - Semiparametric: Product of a parametric Gaussian approximation with a nonparametric KDE correction term.

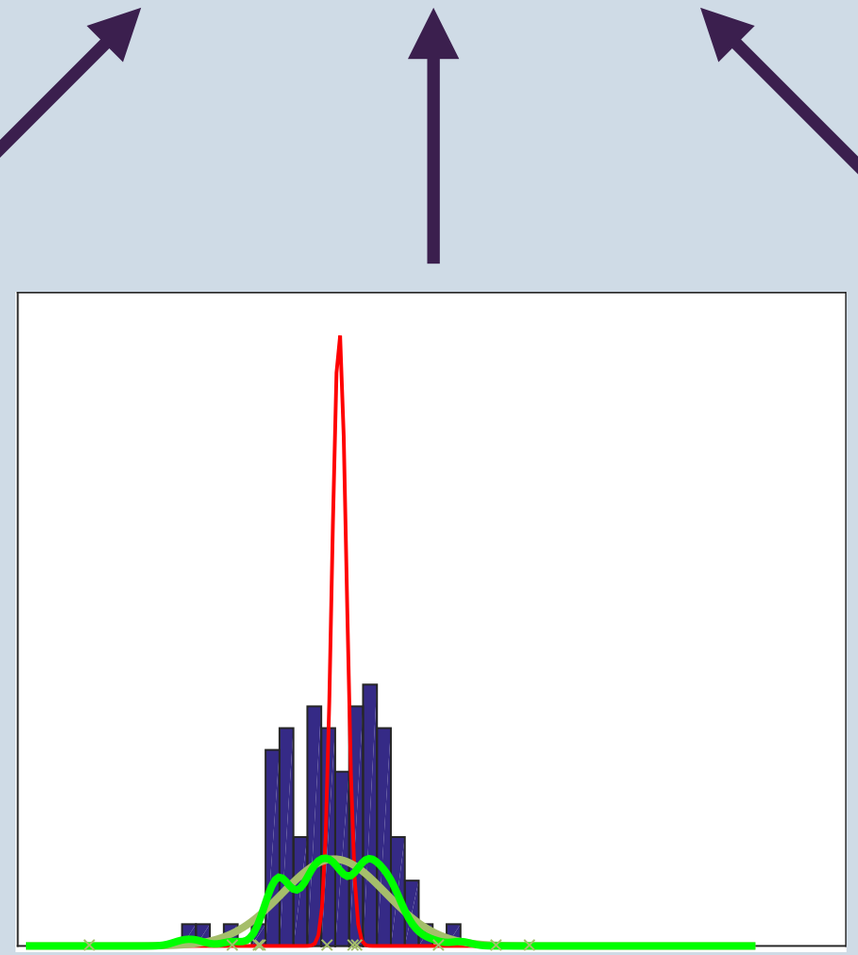
$$p(x \mid y) \propto \prod_{j=1}^{m} p_j(x \mid y_j) \approx \prod_{j=1}^{m} \frac{1}{S} \sum_{s=1}^{S} \mathcal{K}_{h_j}(x; x_{js})$$

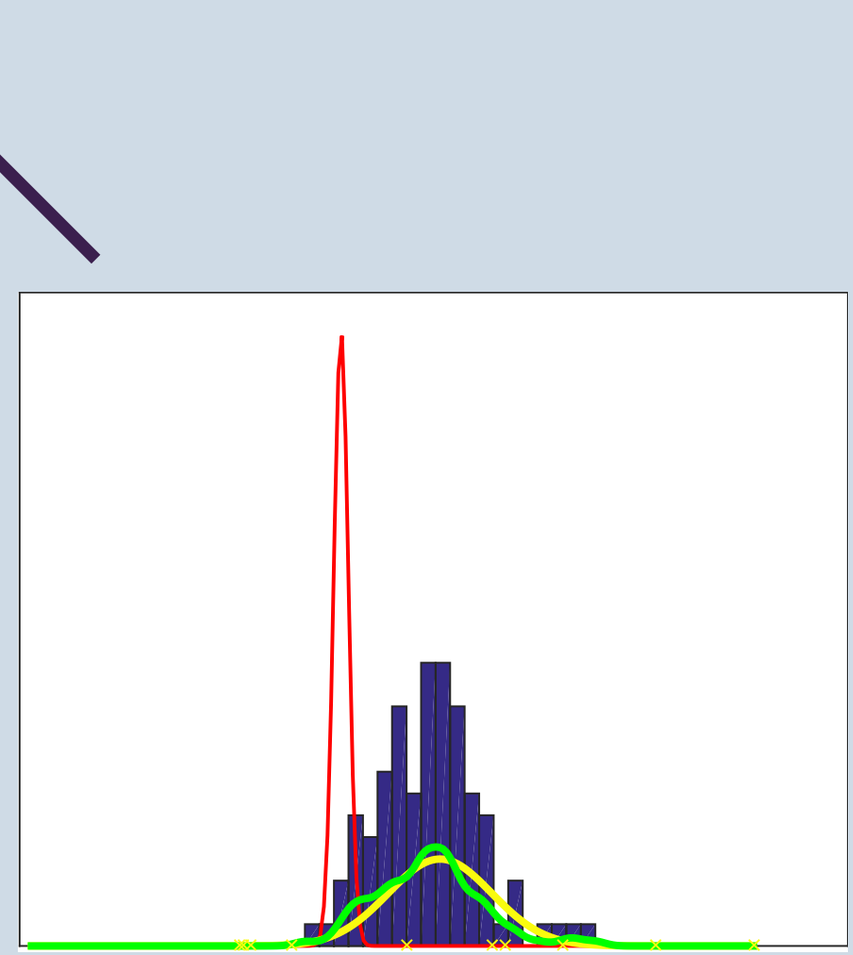
- Combination: Product of (approximate) densities.
- Sampling: Resort to Metropolis-within-Gibbs.
- [Wang & Dunson 2013]'s Weierstrass sampler is similar, using rejection sampling instead.

[Neiswanger et al 2013, Wang & Dunson 2013]









Embarassingly Parallel MCMC Sampling

- Unclear how to combine worker samples sensibly.
- Particularly if local posteriors on worker machines do not overlap.
- Combination at master involves:
  - weighted average of samples [Scott et al]
  - Gaussian approximation [Neiswanger et al]
  - KDE [Neiswanger, Wang & Dunson, Stanislav et al]

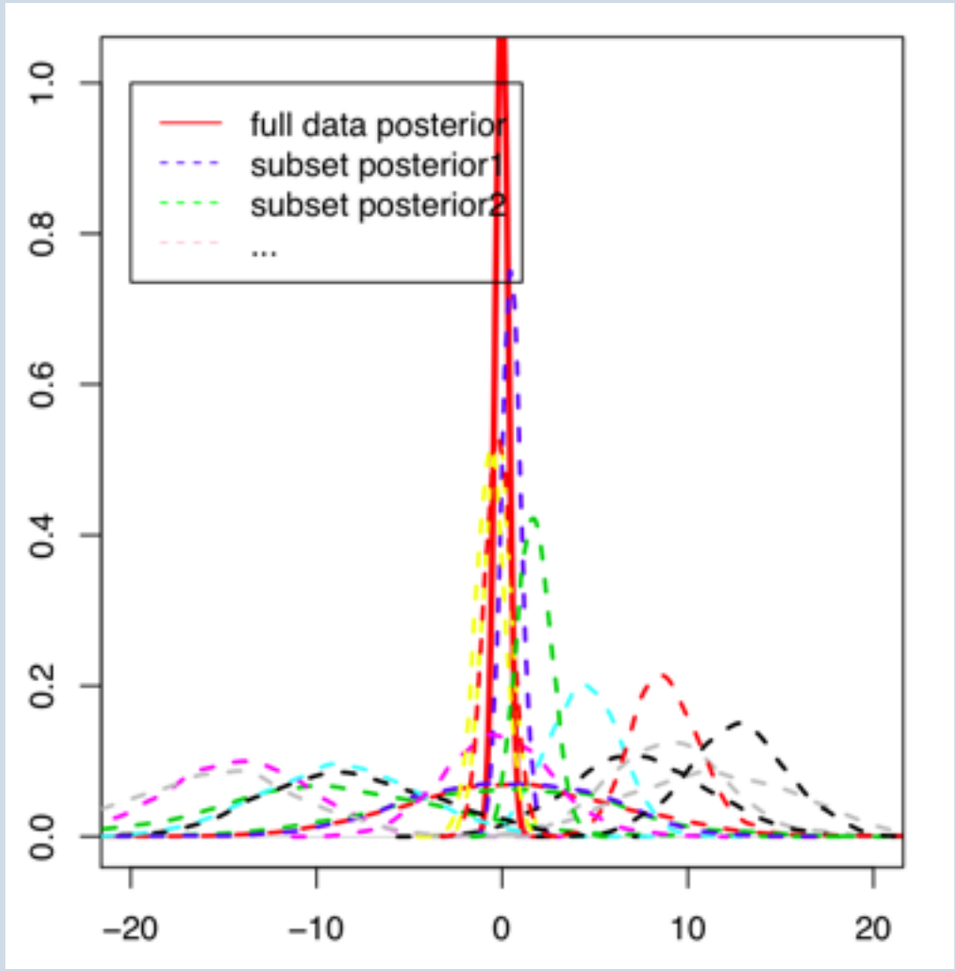


Figure from Wang & Dunson

Intuition and Desiderata

- Distributed system with independent MCMC sampling.
- Identify regions of high (global) posterior probability mass.
- Each local sampler is based on local data, but "concentrate on high probability regions".
- High probability regions found by identifying its moments using small amount of communication during learning.

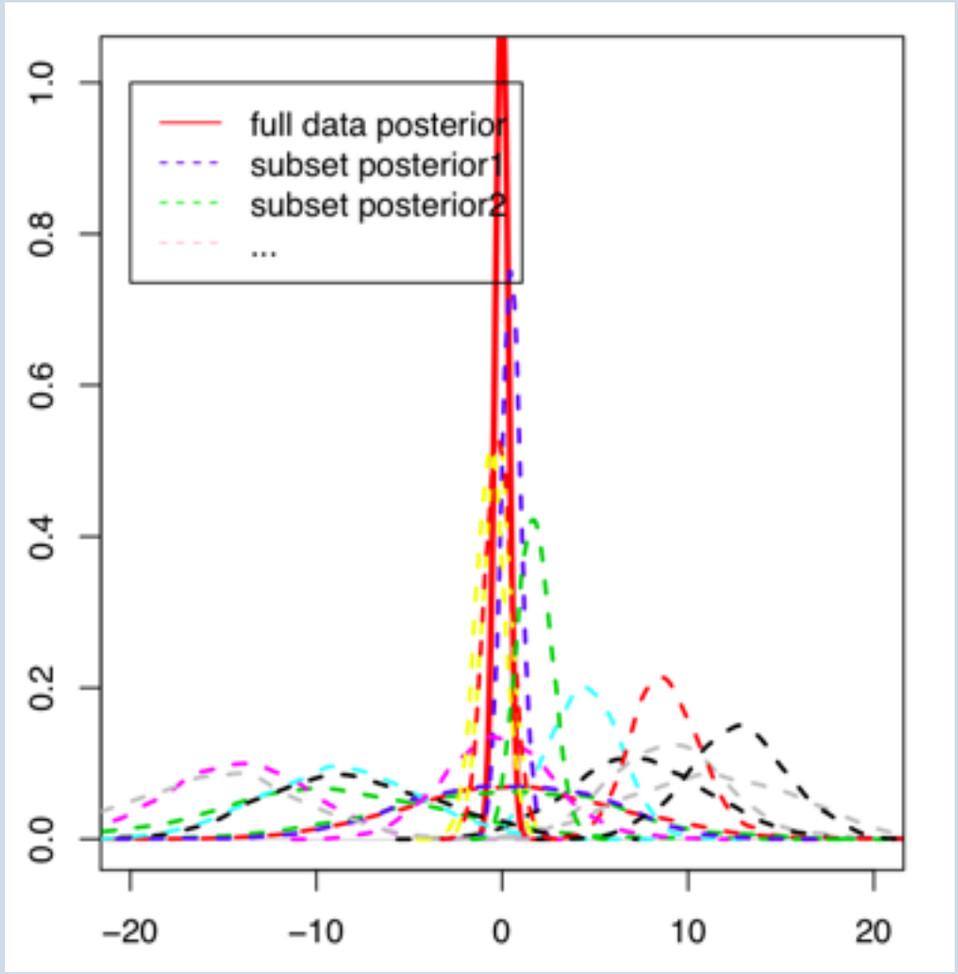


Figure from Wang & Dunson

Generic (Bayesian) Learning on Big Data

- Stochastic optimisation using mini-batches.
  - Stochastic gradient optimisation.
  - Stochastic variational inference [Hoffman et al 2013, Mimno et al 2012]
  - Stochastic Gradient MCMC [Welling & Teh 2011, Patterson & Teh 2013, Teh et al 2016, Chen et al 2014, Ma et al 2015, Din et al 2015, Leimkuhler & Shang 2015…]

- Distributed/parallel computations on cores/clusters/GPUs.
  - MapReduce, parameter server.
  - Embarassingly Parallel MCMC [[Scott et al 2013, Neiswanger et al 2013, Wang & Dunson 2013, Stanislav et al 2014]
  - **Sampling via Moment Sharing [Xu et al 2014]**
  - **Stochastic Natural-gradient EP and Posterior Server [Teh et al 2016]**

Local and Global Posteriors

- Each worker machine j has access only to its data subset.

$$p_j(x \mid y_j) = p_j(x) \prod_{i=1}^{I} p(y_{ji} \mid x)$$

where $p_j(x)$ is a local prior and $p_j(x \mid y_j)$ is local posterior.

- The (target) global posterior is

$$p(x \mid y) \propto p(x) \prod_{j=1}^{m} p(y_j \mid x) \propto p(x) \prod_{j=1}^{m} \frac{p_j(x \mid y_j)}{p_j(x)}$$

- Choose local priors $p_j(x)$ so that

$$\mathbb{E}_{p_j(x|y_j)}[s(x)] = s_0 \quad \forall j$$

- Use expectation propagation (EP) [Minka 2001] to find good local priors.

Expectation Propagation

- If N is large, the worker j likelihood term $p(y_j \mid x)$ should be well approximated by Gaussian

$$p(y_j \mid x) \approx q_j(x) = \mathcal{N}(x; \mu_j, \Sigma_j)$$

- Parameters fit iteratively to minimize KL divergence:

$$p(x \mid y) \approx p_j(x \mid y) \propto p(y_j \mid x)\, p(x) \underbrace{\prod_{k \neq j} q_k(x)}_{p_j(x)}$$

$$q_j^{\text{new}}(\cdot) = \arg\min_{\mathcal{N}(\cdot;\mu,\Sigma)} \mathrm{KL}\big(p_j(\cdot \mid y) \,\|\, \mathcal{N}(\cdot;\mu,\Sigma) p_j(\cdot)\big)$$

- Optimal q_j is such that first two moments of $\mathcal{N}(\cdot;\mu,\Sigma)p_j(\cdot)$ agree with $p_j(\cdot|y)$
- At convergence,

$$\mathbb{E}_{p_j(x|y_j)}[s(x)] = \mathbb{E}_{p(x)\prod_k q_k(x)}[s(x)] \quad \forall j$$

[Minka 2001]

Expectation Propagation

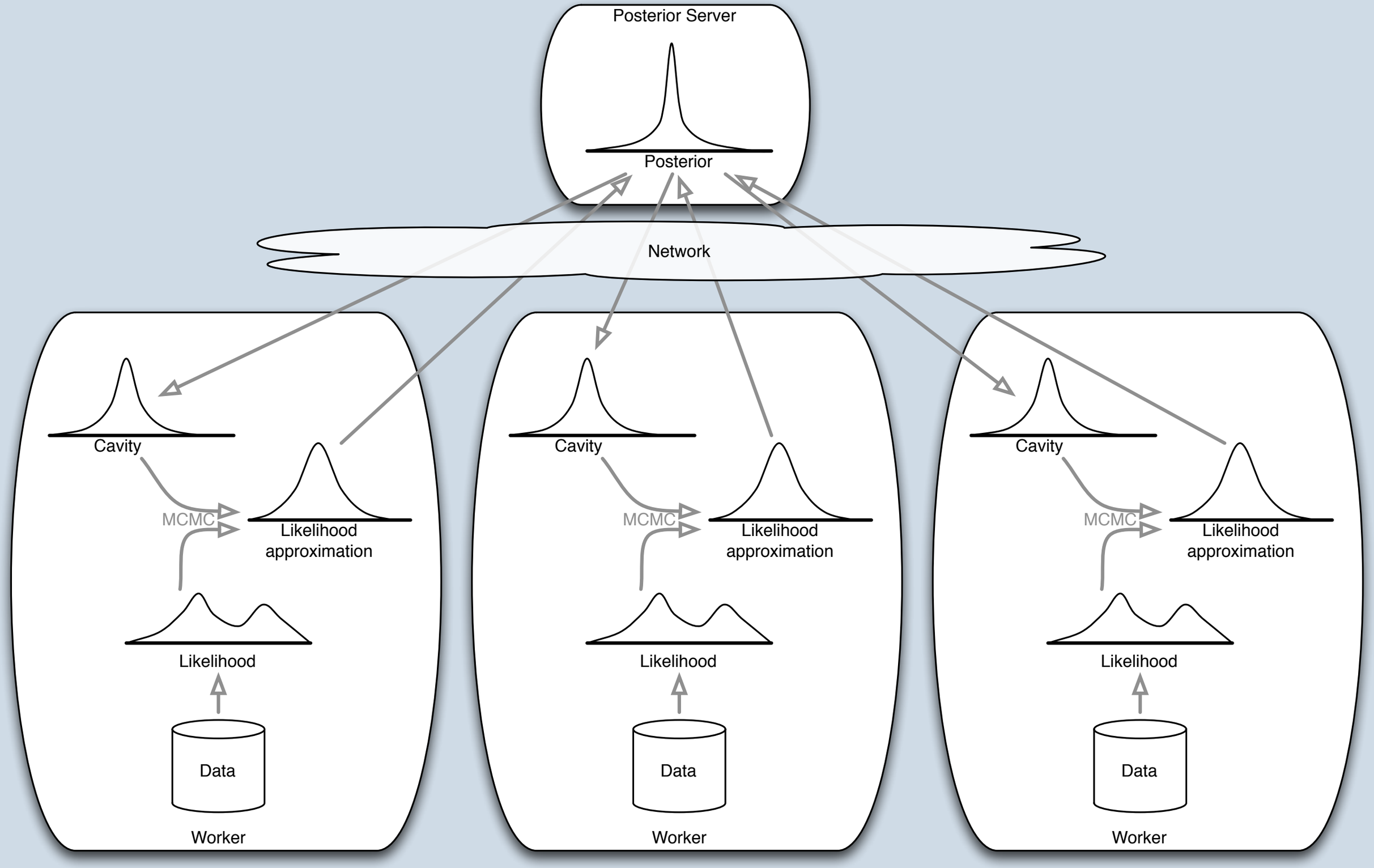
$$p(x \mid y) \approx p_j(x \mid y) \propto p(y_j \mid x)\, \underbrace{p(x) \prod_{k \neq j} q_k(x)}_{p_j(x)}$$

$$q_j^{\text{new}}(\cdot) = \arg \min_{\mathcal{N}(\cdot;\mu,\Sigma)} \mathrm{KL}\big(p_j(\cdot \mid y) \,\|\, \mathcal{N}(\cdot;\mu,\Sigma) p_j(\cdot)\big)$$

- Update performed as follows:
  - Compute (or estimate) first two moments μ^*, Σ^* of $p_j(x \mid y)$.
  - Compute μ_j, Σ_j so that $N(.; \mu_j, \Sigma_j)\, p_j(.)$ has moments μ^*, Σ^*.

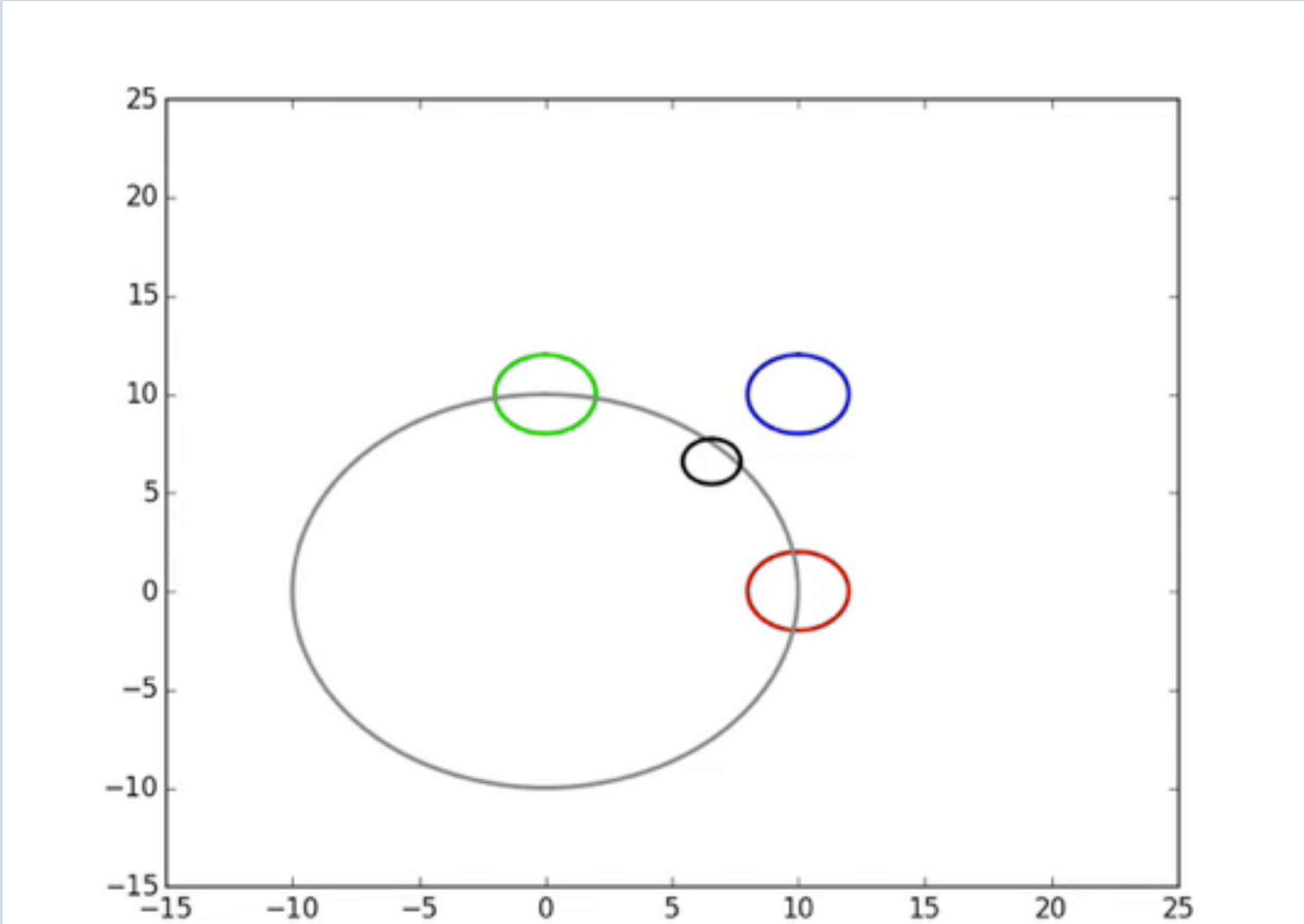
- In high-dimensions, can use diagonal covariances.
- Generalizes to other exponential families.
- EP tends to converge very quickly (when it does).
- At convergence, all local posteriors agree on their first two moments.

Big Picture


Posterior Server
Posterior
Network
Cavity
MCMC
Likelihood approximation
Likelihood
Data
Worker
Cavity
MCMC
Likelihood approximation
Likelihood
Data
Worker
Cavity
MCMC
Likelihood approximation
Likelihood
Data
Worker

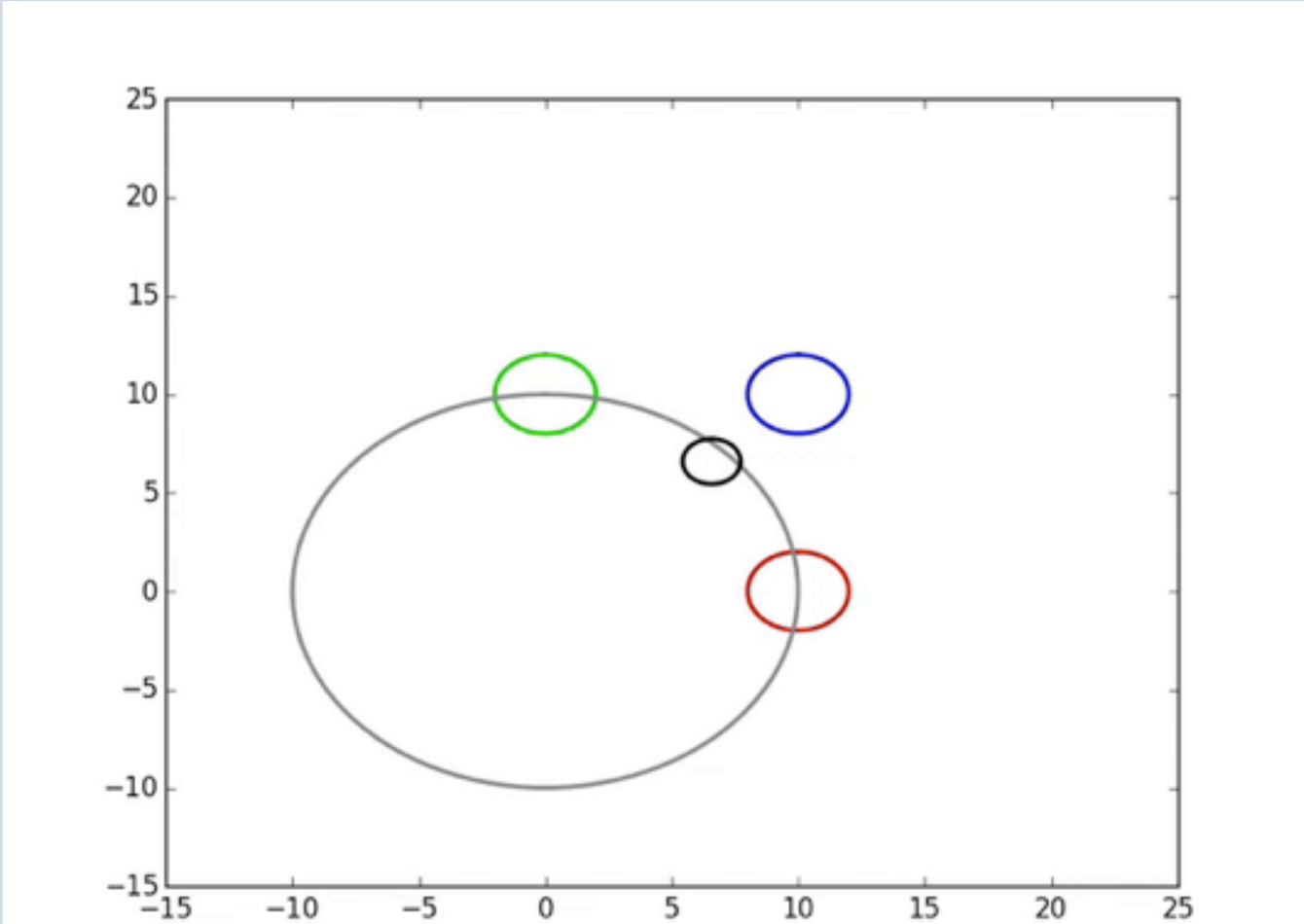
Demonstrative Example

- Simple 2D Gaussian example.
- 3 worker machines.
- 5000 MCMC samples used to estimate sufficient statistics per iteration.
- Each frame corresponds to 100 samples.

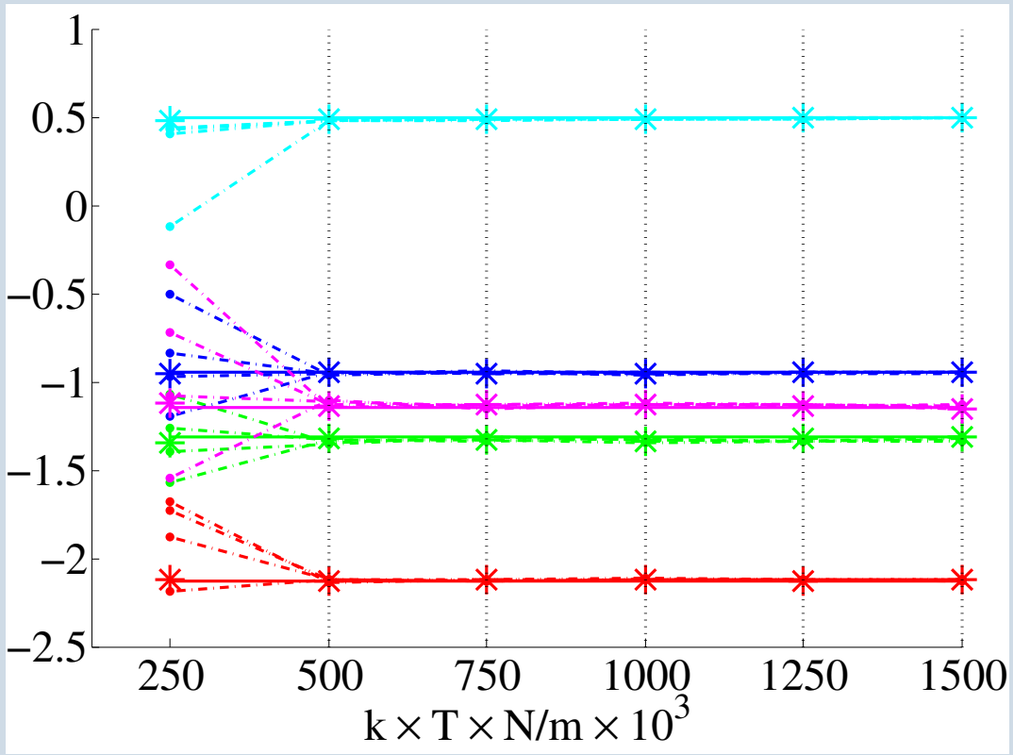


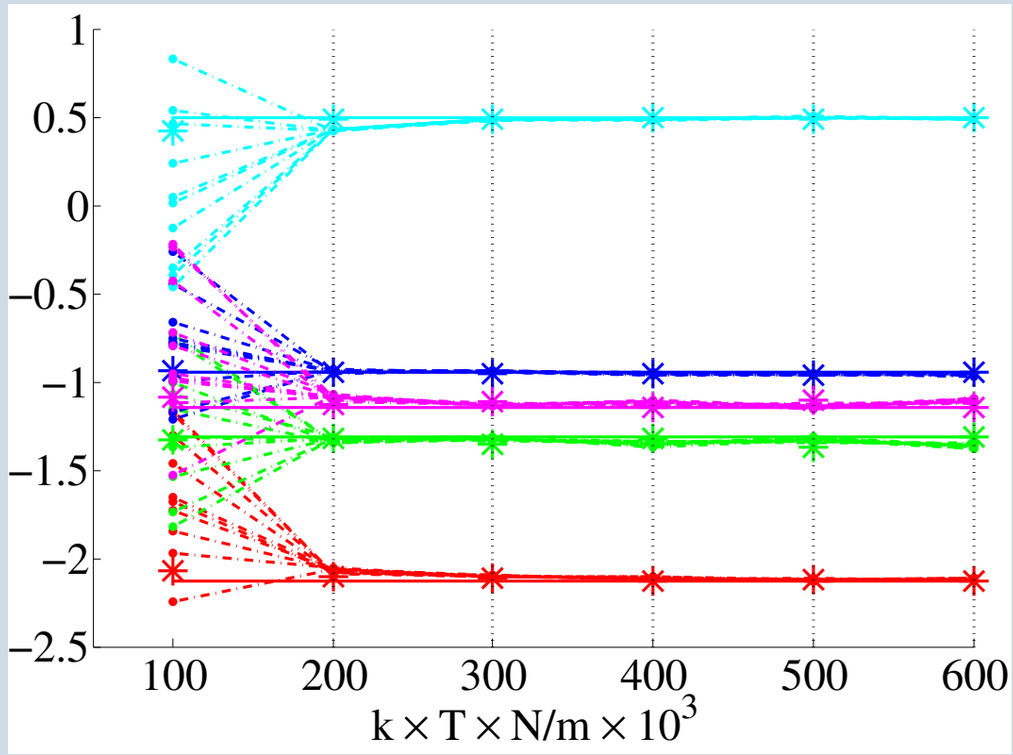
Demonstrative Example

- Simple 2D Gaussian example.
- 3 worker machines.
- 5000 MCMC samples used to estimate sufficient statistics per iteration.
- Each frame corresponds to 100 samples.

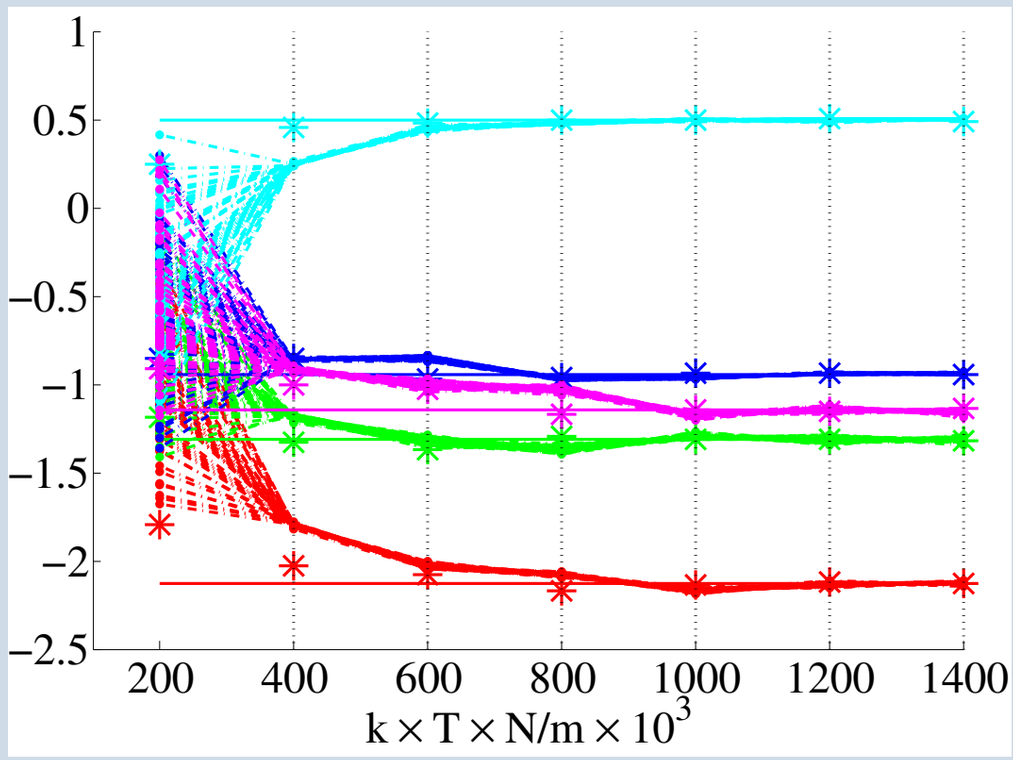


Bayesian Logistic Regression



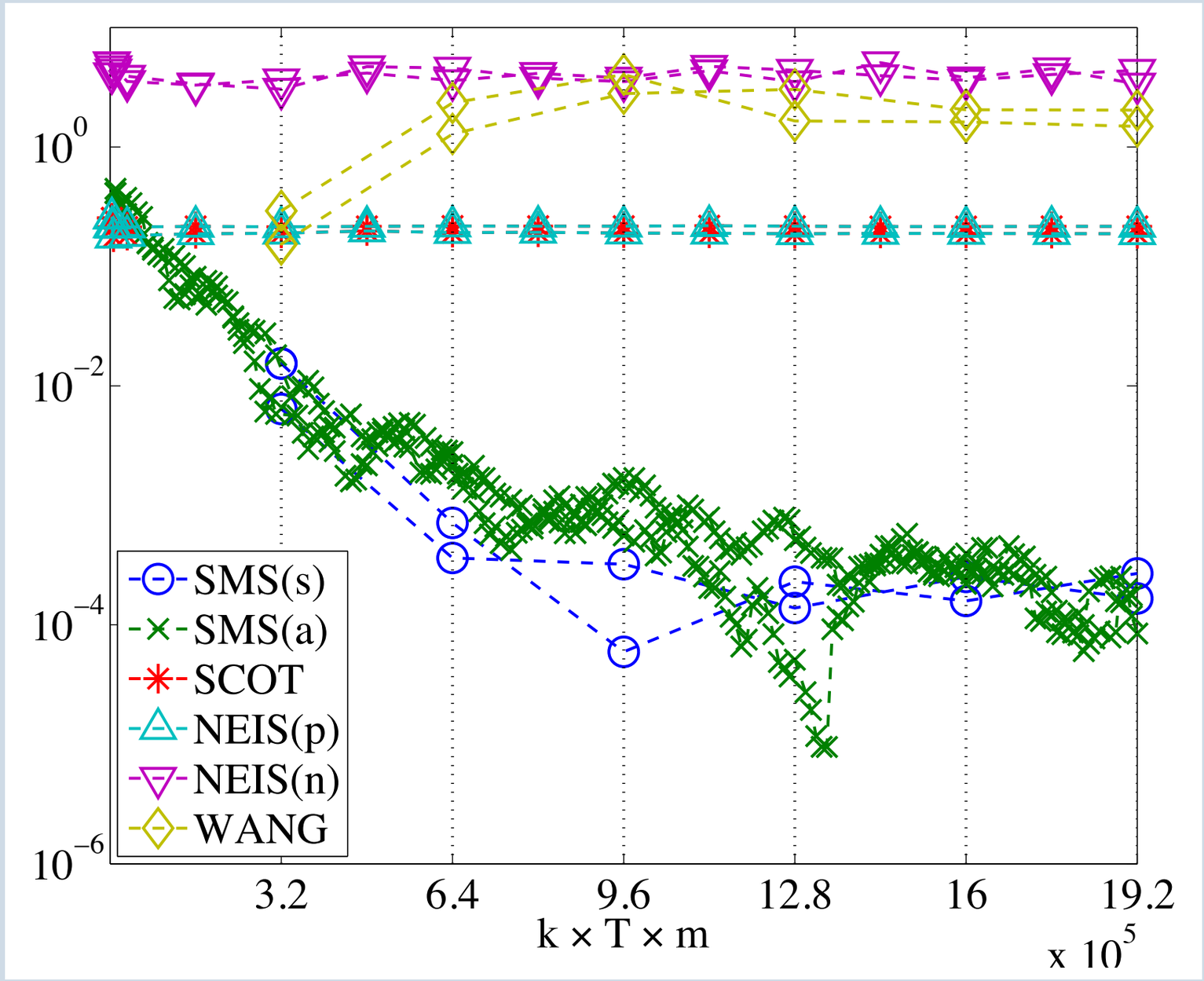


- Simulated dataset.
  - d=20, # data items N=1000.
- NUTS based sampler.
  - # workers m = 4,10,50.
  - # MCMC iters T = 1000,1000,10000.
- # EP iters k given as vertical lines.



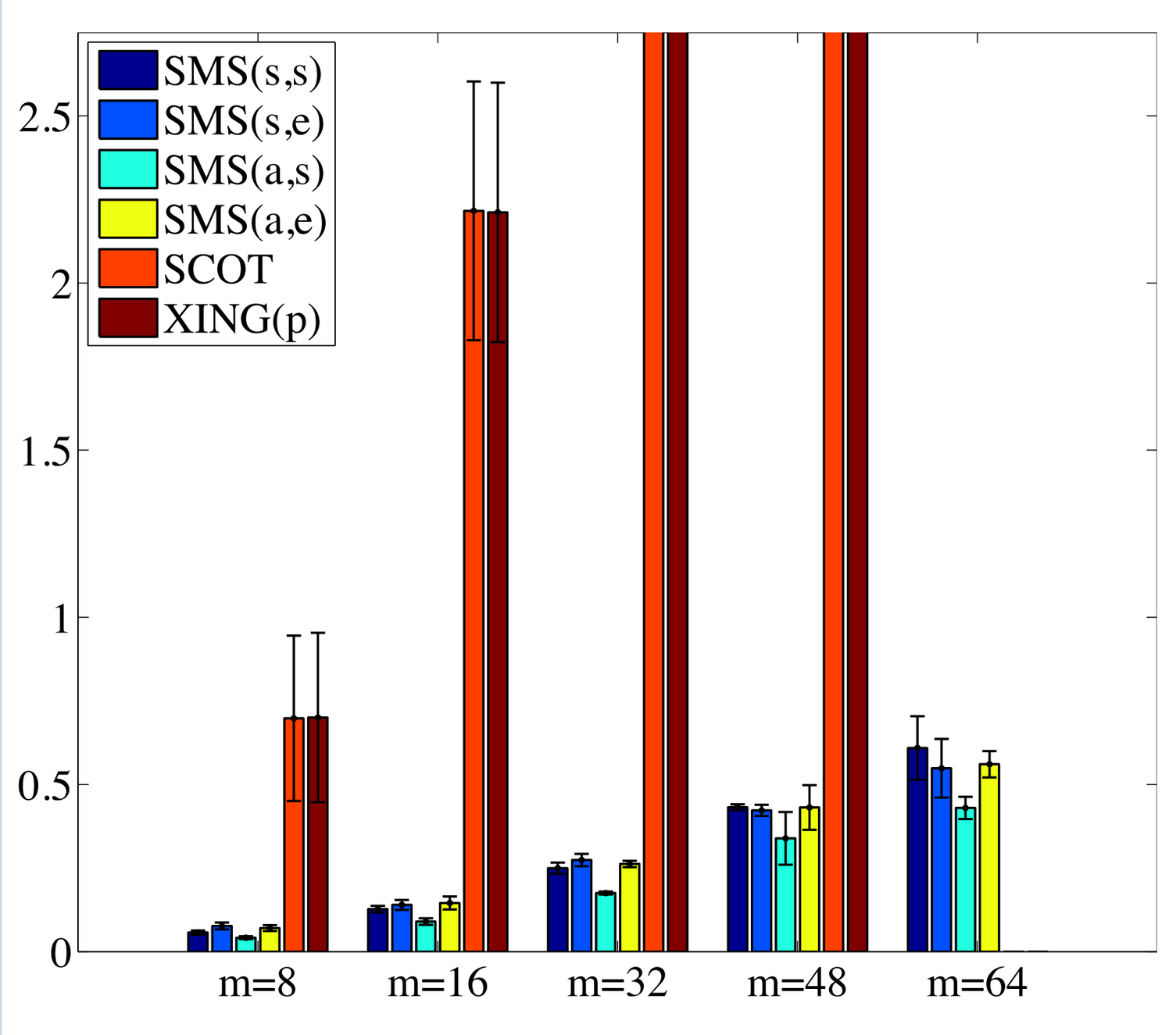
Bayesian Logistic Regression

- MSE of posterior mean, as function of total # iterations.



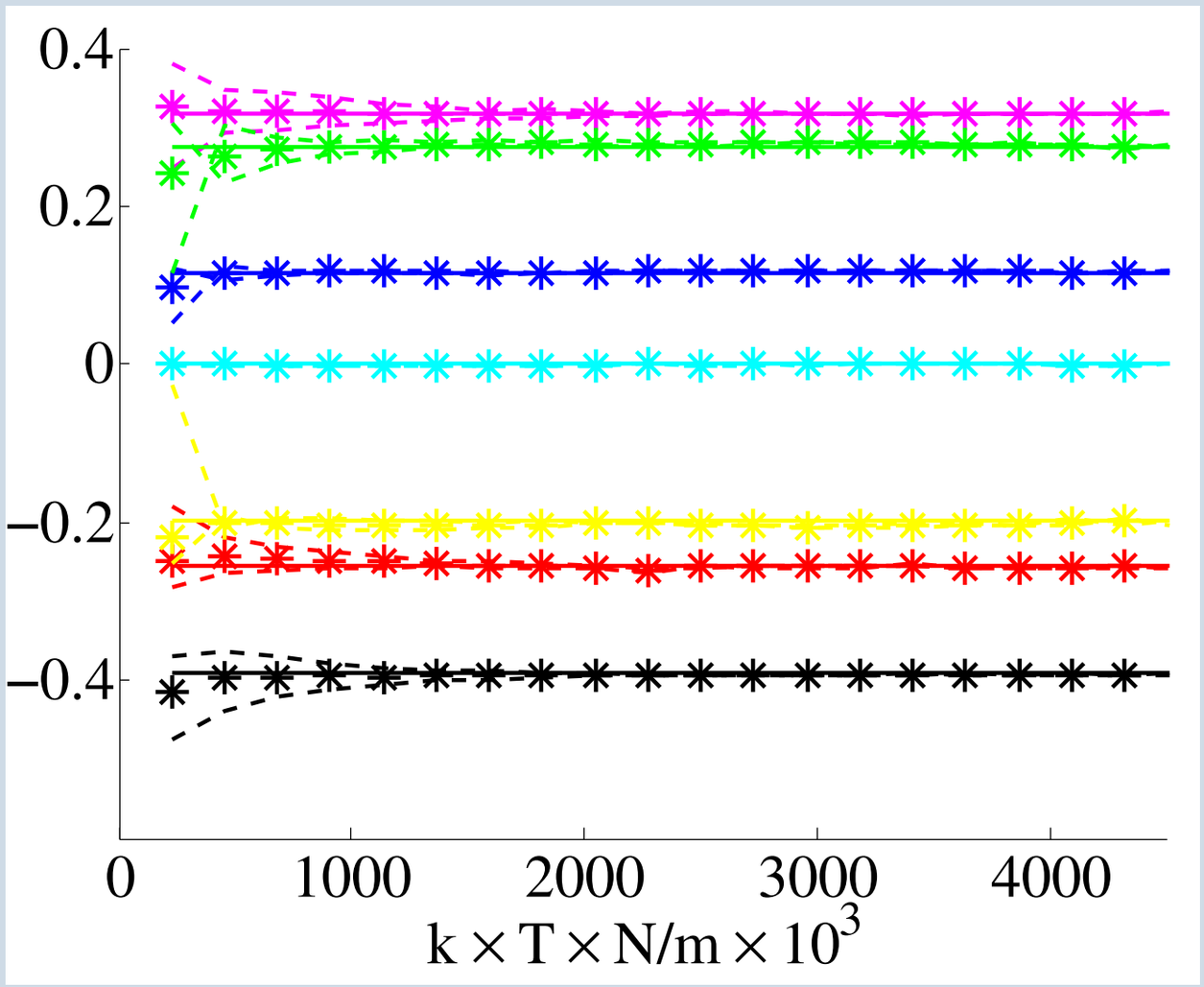
Bayesian Logistic Regression

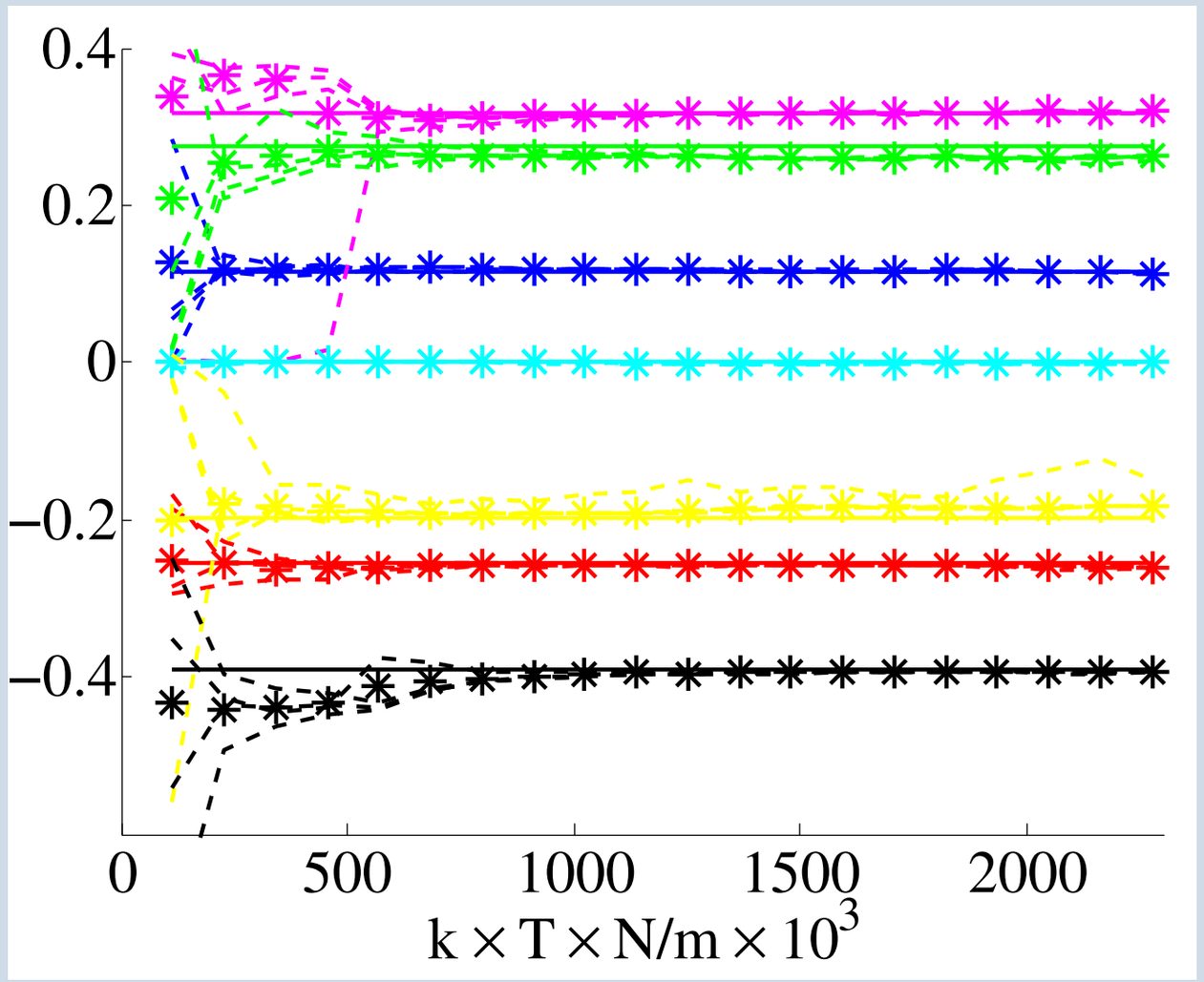
- Approximate KL as function of # nodes.



Spike-and-Slab Sparse Regression

- Posterior mean coefficients.





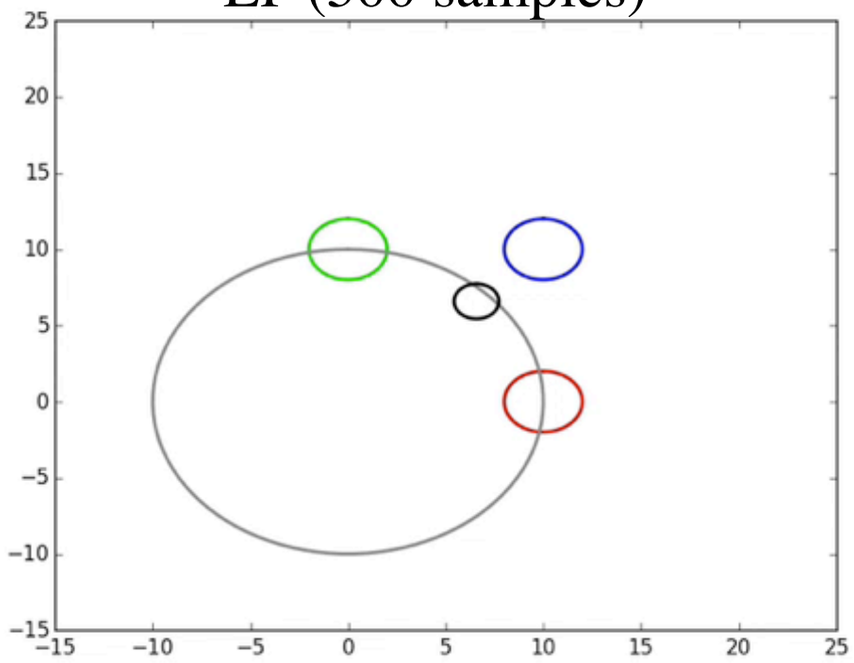
Stochastic Natural-gradient EP

- EP has no guarantee of convergence.
- EP technically cannot handle stochasticity in moment estimates.
- Long MCMC run needed for good moment estimates.
- No clear understanding of convergence and quality of approximation in stochastic case.
- Fails for neural nets and other complex high-dimensional models.

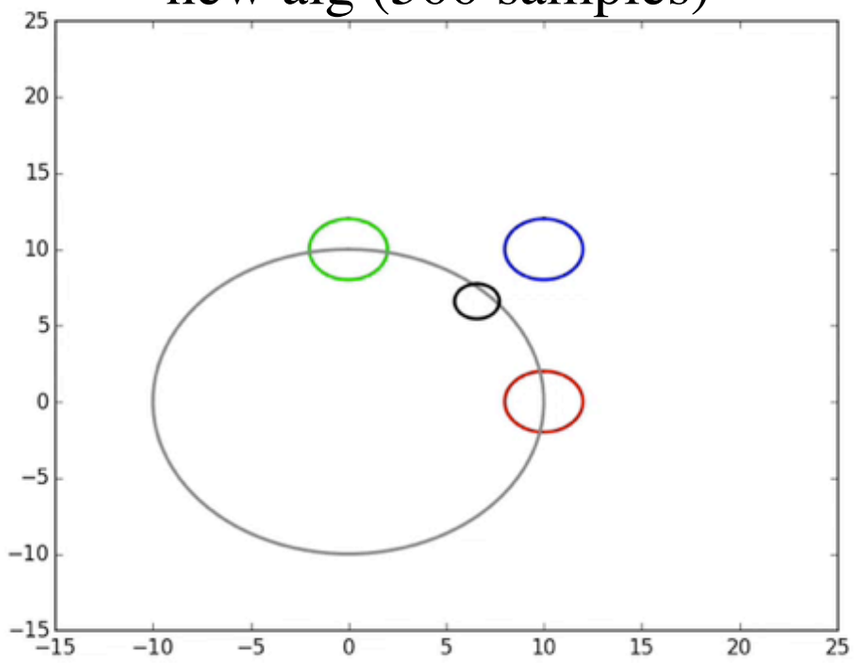
- Stochastic Natural-gradient EP (Teh et al 2015):
  - Alternative variational algorithm to EP.
  - Convergent, even with Monte Carlo estimates of moments.
  - Double-loop algorithm [Welling & Teh 2001, Yuille 2002, Heskes & Zoeter 2002]
  - Arxiv manuscript 1512.09327.

Demonstrative Example

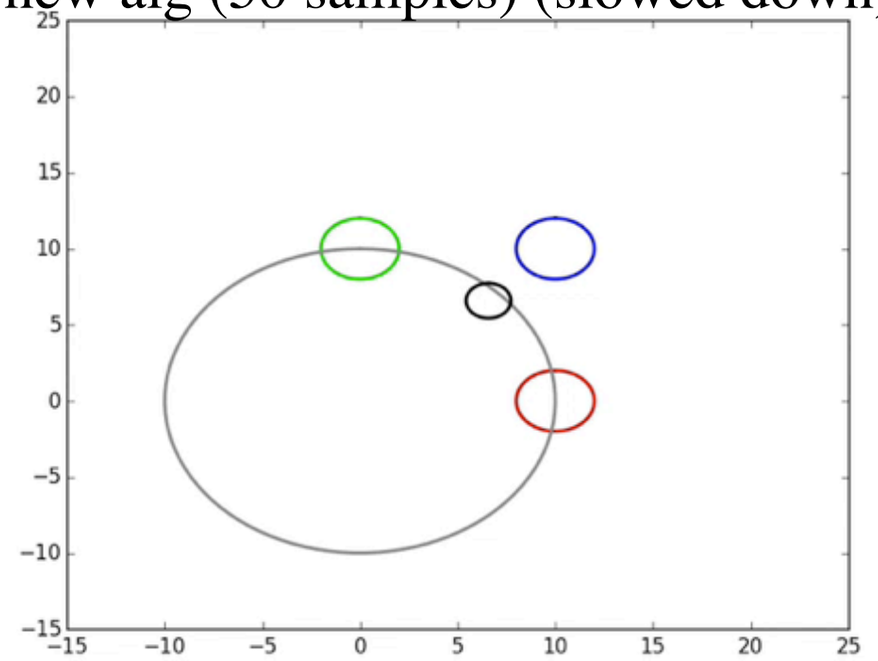
EP (500 samples)



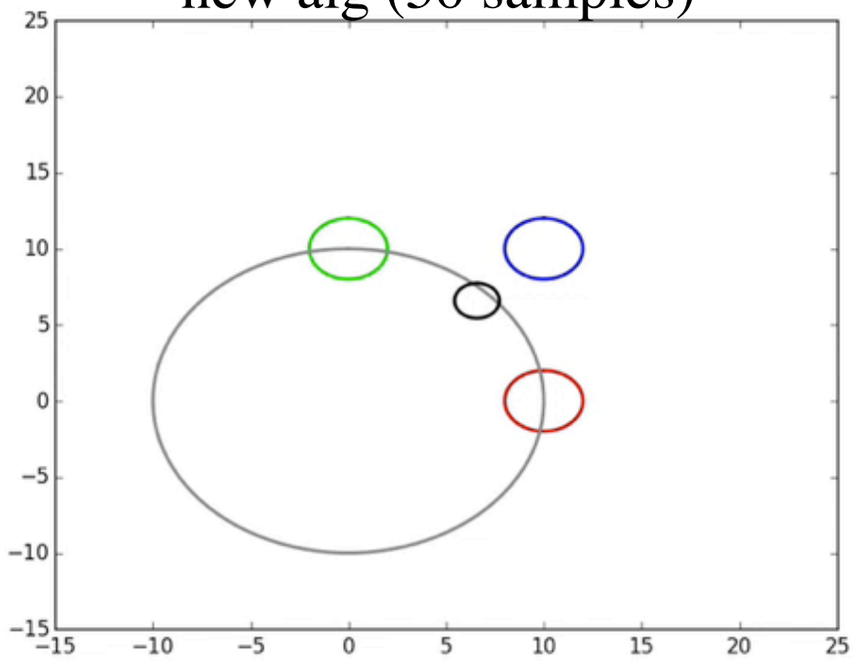
new alg (500 samples)



new alg (50 samples) (slowed down)

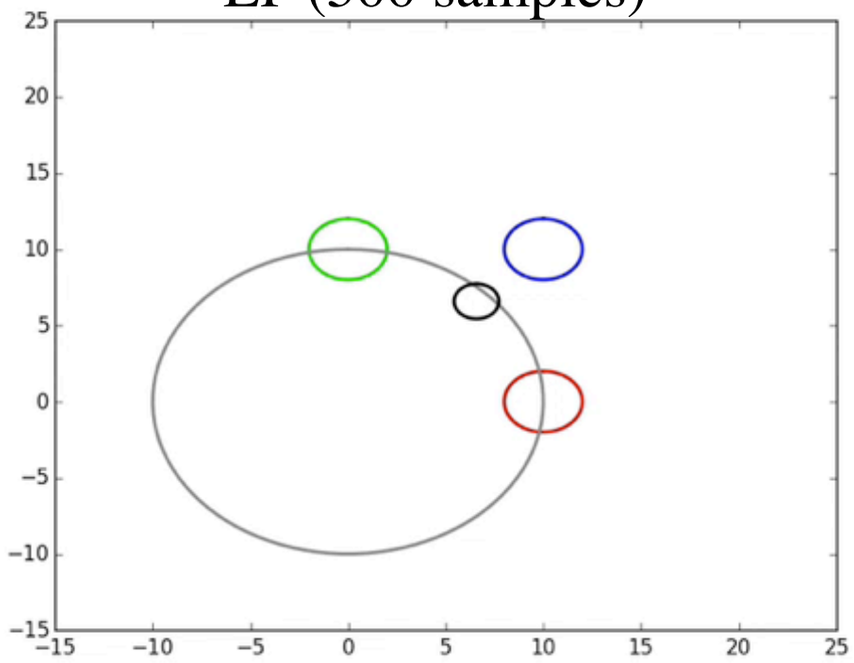


new alg (50 samples)

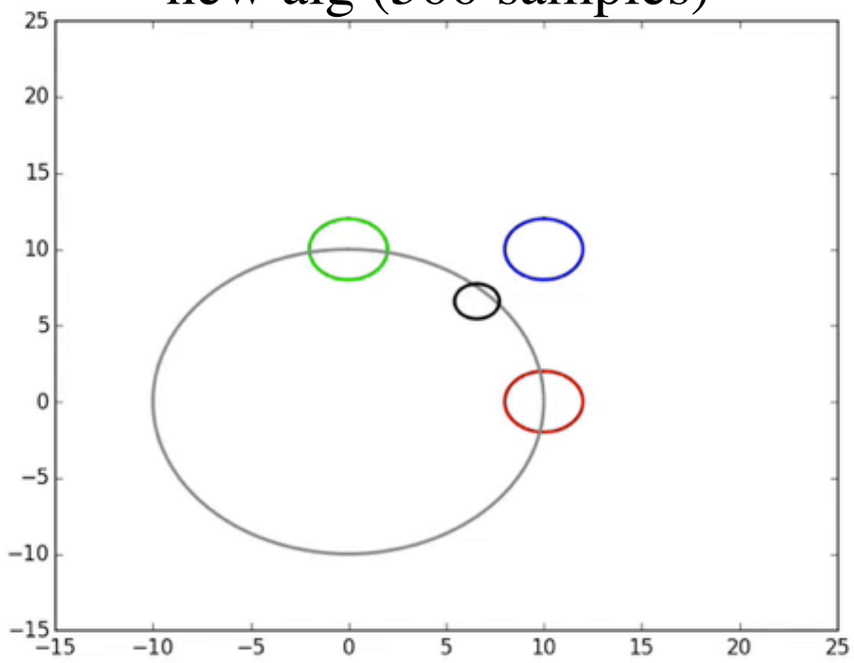


Demonstrative Example

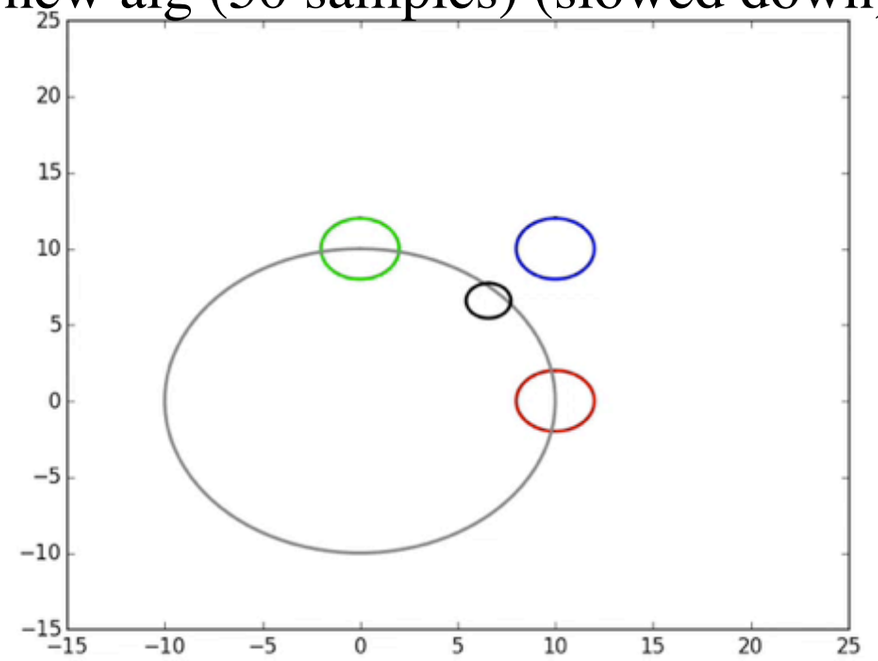
EP (500 samples)



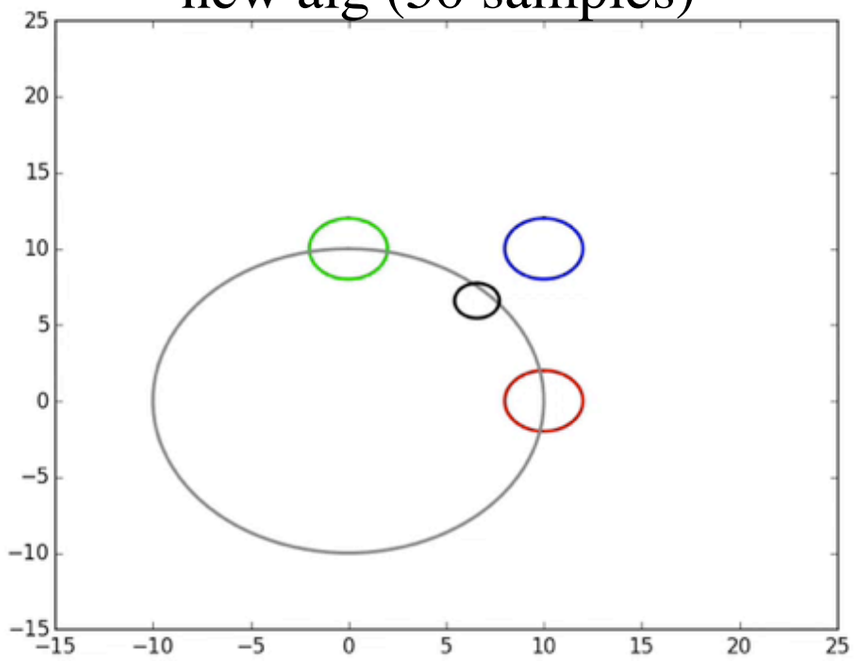
new alg (500 samples)



new alg (50 samples) (slowed down)

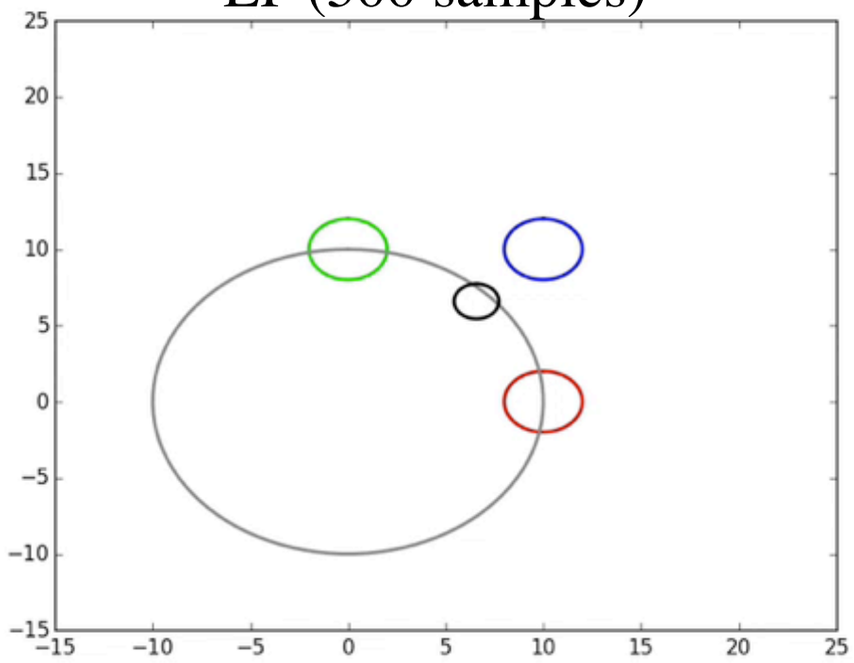


new alg (50 samples)

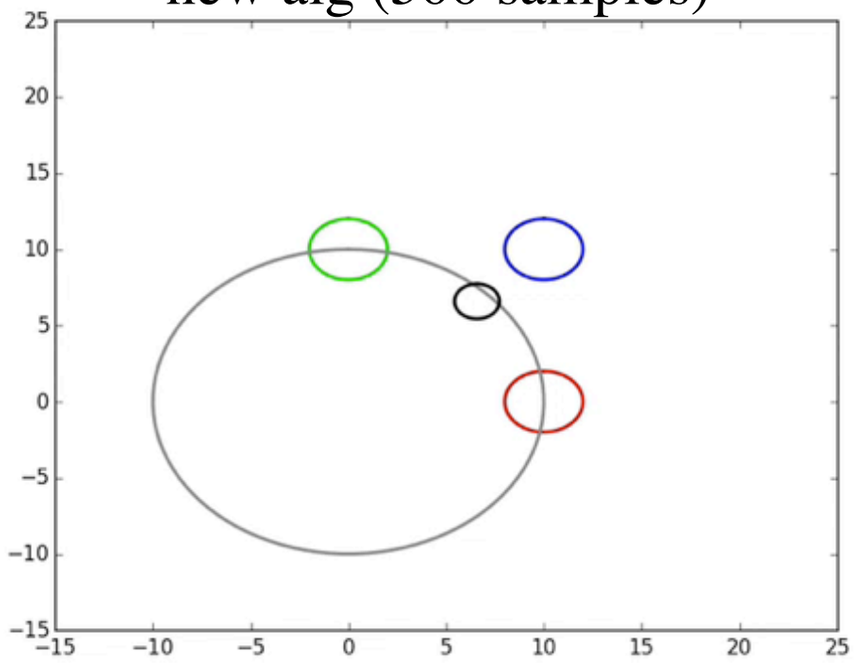


Demonstrative Example

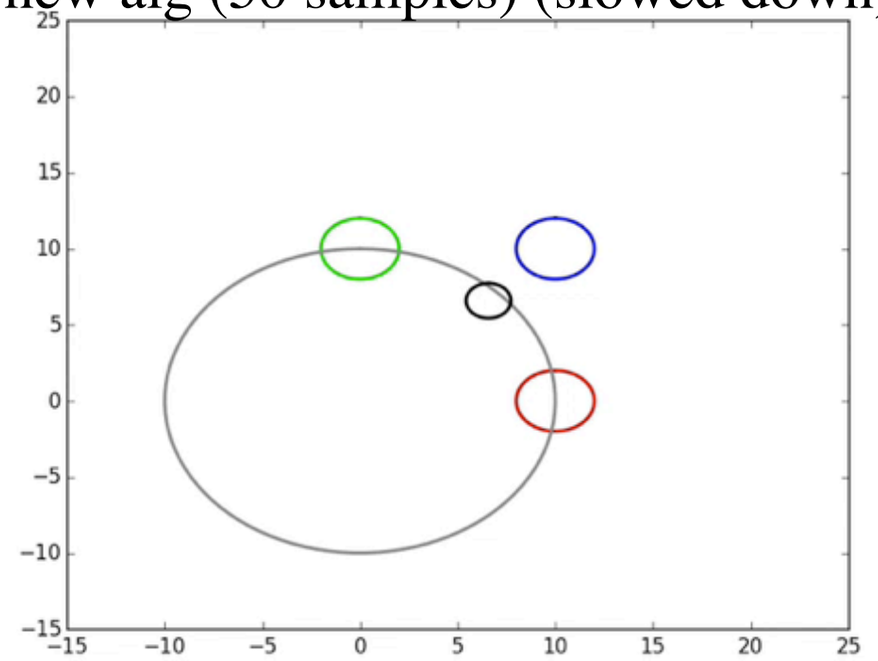
EP (500 samples)



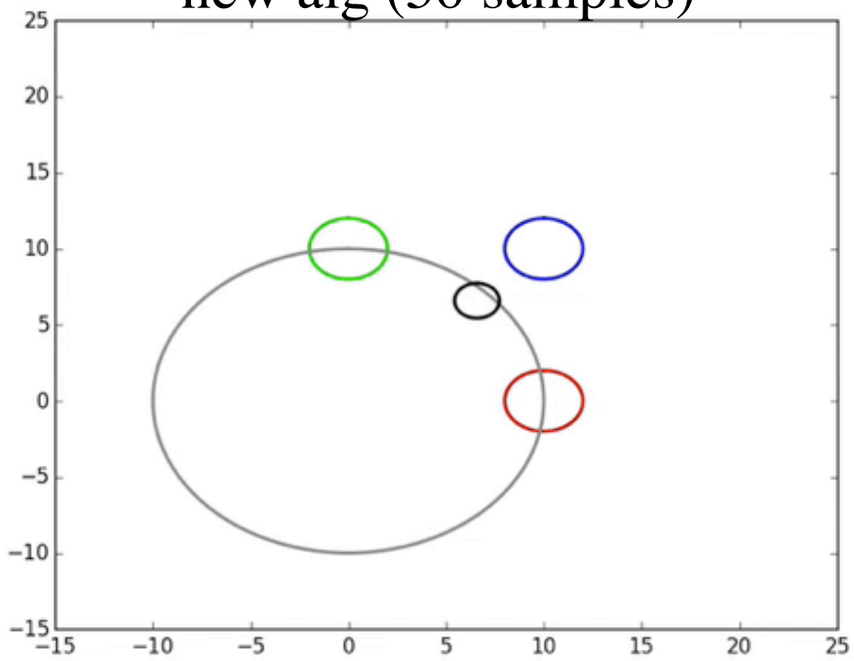
new alg (500 samples)



new alg (50 samples) (slowed down)

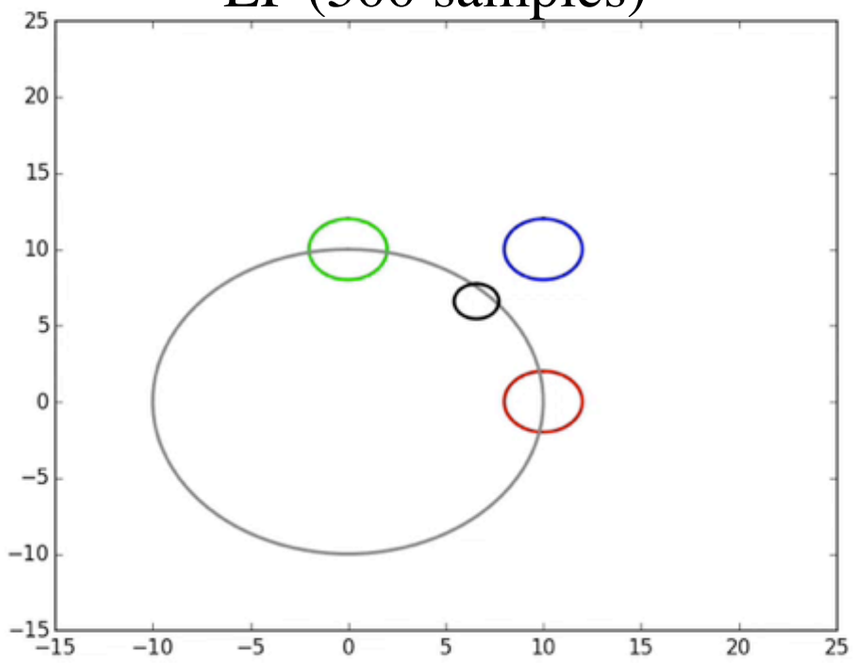


new alg (50 samples)

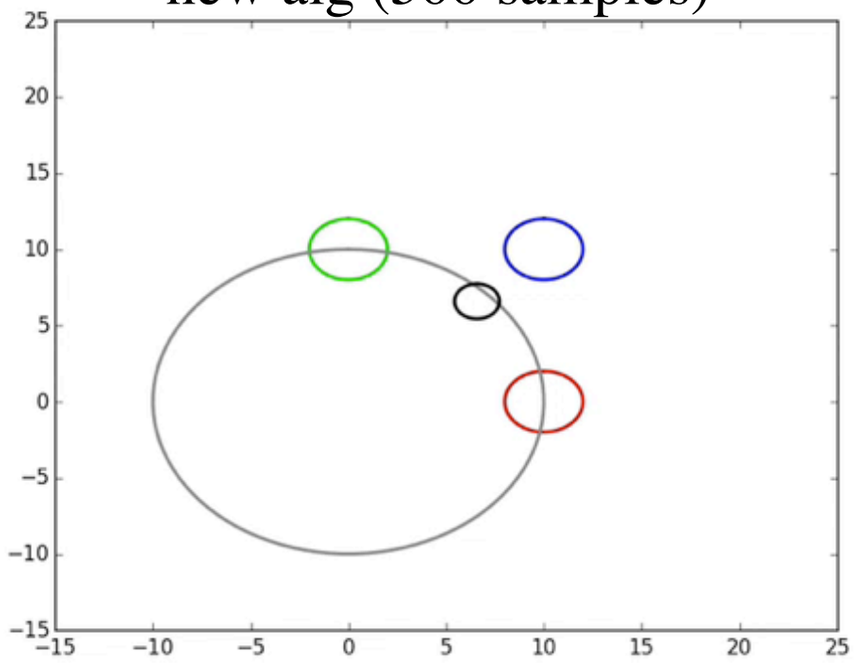


Demonstrative Example

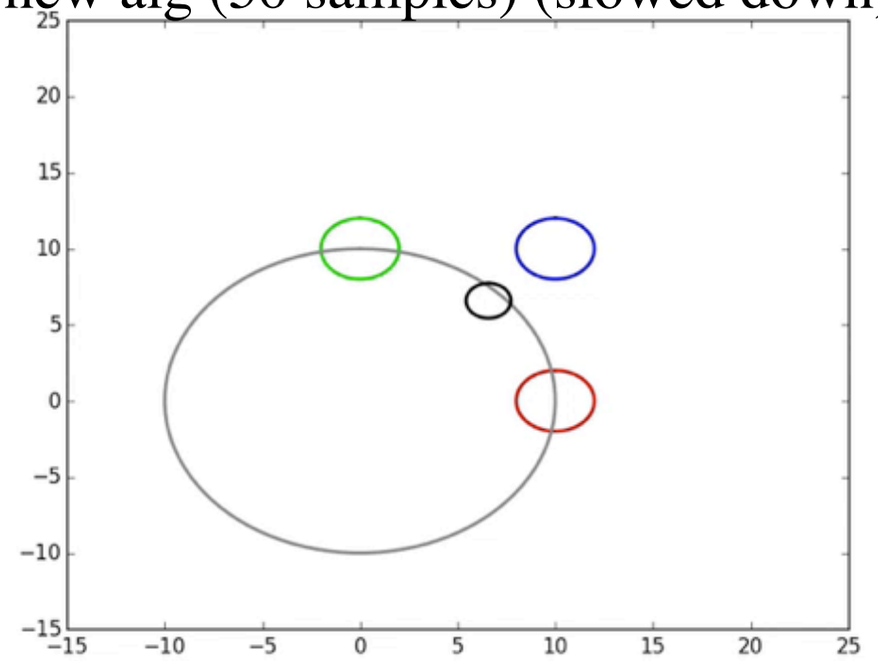
EP (500 samples)



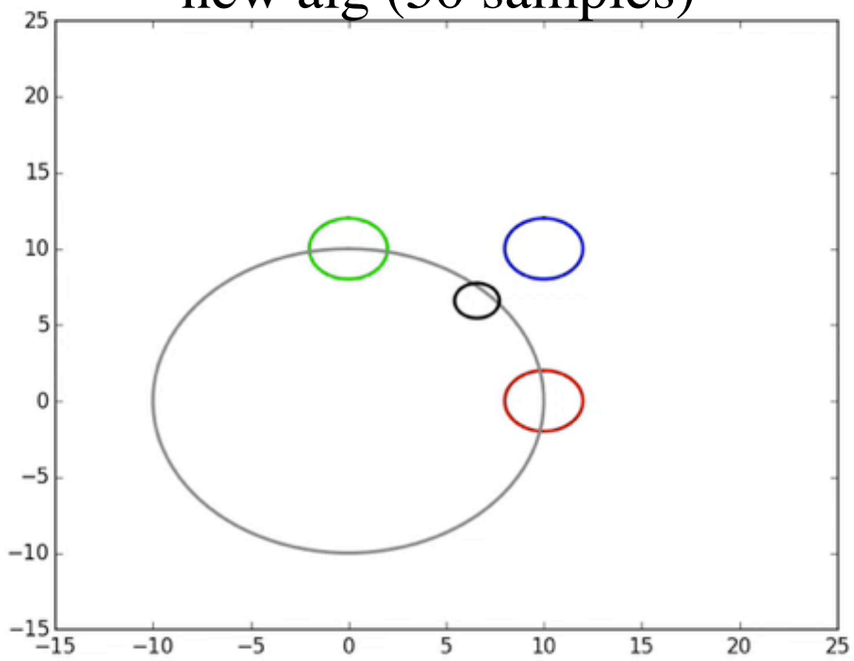
new alg (500 samples)



new alg (50 samples) (slowed down)



new alg (50 samples)



Exponential Families

$$p_\theta(x) = \exp\left(\theta^\top s(x) - A(\theta)\right)$$

- Sufficient statistics s, natural parameters θ, log partition function $A(\theta)$.

- Equivalent parameterisation as mean parameters μ:

$$\mu = \mathbb{E}_\theta[s(x)]$$

- $A(\theta)$ is convex with convex domain Θ.
- Convex conjugate is the negative entropy $A^*(\mu)$ with convex domain $\mathcal{M}$:

$$A^*(\mu) = \sup_\theta \theta^\top \mu - A(\theta) = \mathbb{E}_{\theta(\mu)}[\log p_{\theta(\mu)}(x)]$$

- Derivatives of A and A^* convert between natural and mean parameters:

$$\mu(\theta) = \nabla A(\theta) \qquad \theta(\mu) = \nabla A^*(\mu)$$

Arbitrary Model as Extended Exponential Family

- Prior $p_0(x)$ in exponential family, log likelihoods $l_j(x)$ for each worker j.

$$p(x|y) \propto \exp(\theta_0^\top s(x)) \prod_{j=1}^{m} \exp(l_j(x))$$

$$= \exp([\theta_0; 1 \dots 1]^\top [s(x); l_1(x) \dots l_m(x)])$$

$$p(x|y) = \exp(\tilde{\theta}^\top \tilde{s}(x) - \tilde{A}(\tilde{\theta}))$$

- Bayesian learning can now be posed as computing the mapping $\tilde{\theta} \mapsto \tilde{\mu}$:

$$\arg\max_{\tilde{\mu} \in \tilde{\mathcal{M}}} \tilde{\theta}^\top \tilde{\mu} - \tilde{A}^*(\tilde{\mu})$$

- Variational inference:
  - approximate negative entropy and
  - approximate mean domain.

Expectation Propagation as Variational Approximation

- Write mean parameters $\tilde{\mu} = [\mu; \nu_1 \dots \nu_m]$.
- Approximate entropy as sums of local entropies, approximate mean domain as intersections of local domains:

$$\tilde{A}^*([\mu, \nu_1, \dots, \nu_n]) \approx A^*(\mu) + \sum_{j=1}^{m} (A_j^*(\mu, \nu_j) - A^*(\mu)) \qquad \tilde{\mathcal{M}} \approx \bigcap_{j=1}^{m} \mathcal{M}_j$$

- Local entropies/domains are those associated with a single likelihood term (and the prior).
- Variational optimization problem:

$$\max_{\mu_0, [\mu_j, \nu_j]_{j=1}^m} \quad \theta_0^\top \mu_0 + \sum_{j=1}^{m} 1 \cdot \nu_j - A^*(\mu_0) - \sum_{j=1}^{m} (A_j^*(\mu_j, \nu_j) - A^*(\mu_j))$$

$$\begin{aligned} \text{subject to} \qquad \mu_0 &\in \mathcal{M} \\ [\mu_j, \nu_j] &\in \mathcal{M}_j \quad \text{for } j = 1, \dots, m \\ \mu_0 &= \mu_j \quad \text{for } j = 1, \dots, m \end{aligned}$$

Expectation Propagation as Variational Approximation

- Introducing Lagrange multipliers for the equality constraints,

$$\max_{\mu_0,[\mu_j,\nu_j]_{j=1}^m} \min_{[\lambda_j]_{j=1}^m} \theta_0^\top \mu_0 - A^*(\mu_0) + \sum_{j=1}^m \left(\nu_j - \lambda_j^\top(\mu_j - \mu_0) - A_j^*(\mu_j,\nu_j) + A^*(\mu_j)\right)$$

$$\text{subject to} \qquad \mu_0 \in \mathcal{M}$$
$$[\mu_j,\nu_j] \in \mathcal{M}_j \quad \text{for } j = 1,\ldots,m$$

- EP can now be derived as fixed-point equations:
  - KKT conditions (setting derivatives to zero).
- Problems:
  - Non-convex due to $+A^*(\mu_i)$ terms.
  - No guarantee of convergence.

Convergent Expectation Propagation

- Introduce additional parameters θ_j' and - KL terms

$$\max_{[\theta'_j]_{j=1}^m} \max_{\mu_0,[\mu_j,\nu_j]_{j=1}^m} \theta_0^\top \mu_0 + \sum_{j=1}^m \nu_j - A^*(\mu_0) - \sum_{j=1}^m (A_j^*(\mu_j,\nu_j) - A^*(\mu_j) + \mathrm{KL}(\mu_j \| \theta'_j))$$

$$\begin{aligned} \text{subject to} \quad \mu_0 &\in \mathcal{M} \\ [\mu_j,\nu_j] &\in \mathcal{M}_j \quad \text{for } j = 1,\ldots,m \\ \mu_0 &= \mu_j \quad \text{for } j = 1,\ldots,m \end{aligned}$$

- where the KL divergence is $\mathrm{KL}(\mu_j \| \theta'_j) = A^*(\mu_j) + A(\theta'_j) - \mu_j^\top \theta'_j$
- maximizing over θ_j' results in the original problem.

- Alternative interpretation of [Heskes & Zoeter 2002]'s convergent EP.
  - Different model structure.
  - Makes clear the interplay between the cost function and constraints.

Convergent Stochastic Approximation Algorithm

- Introduce Lagrange multipliers and simplifying,

$$\max_{[\theta'_j]} \max_{\mu_0,[\mu_j,\nu_j]} \min_{[\lambda_j]} \quad \theta_0^\top \mu_0 - A^*(\mu_0) + \sum_{j=1}^{m} \left(\nu_j - \lambda_j^\top (\mu_j - \mu_0) - A_j^*(\mu_j, \nu_j) + \mu_j^\top \theta'_j - A(\theta'_j)\right)$$

$$\text{subject to} \quad \mu_0 \in \mathcal{M}, \quad [\mu_j, \nu_j] \in \mathcal{M}_j \quad \text{for } j = 1, \ldots, m$$
$$\theta'_j \in \Theta \quad \text{for } j = 1, \ldots, m$$

- Noticing that cost function is concave in μ_0, μ_j, and ν_j, we can maximize over them (but not θ_j') and obtain the dual problem,

$$\max_{[\theta'_j]_{j=1}^m} \min_{[\lambda_j]_{j=1}^m} A\left(\theta_0 + \sum_{j=1}^{m} \lambda_j\right) + \sum_{j=1}^{m} \left(A_j\left(\theta'_j - \lambda_j, 1\right) - A(\theta'_j)\right)$$

$$\text{subject to} \quad \theta'_j \in \Theta \quad \text{for } j = 1, \ldots, m$$

- λ_j can be interpreted as natural parameters of an exponential family approximation to the likelihood at worker j.

Stochastic Natural-gradient EP (SNEP)

- Can be optimised using a double-loop algorithm.
  - Inner loop: stochastic natural gradient descent

$$\lambda_j^{(t)} = \nabla A\left(\nabla A^*(\lambda_j^{(t-1)}) + \epsilon_t\left(s(x_j^{(t)}) - \nabla A\left(\theta_0 + \sum_{k\neq j}\lambda_k + \lambda_j^{(t-1)}\right)\right)\right)$$

  - x_i are samples (we use SGLD with adaptive mass parameter) from the local posterior:

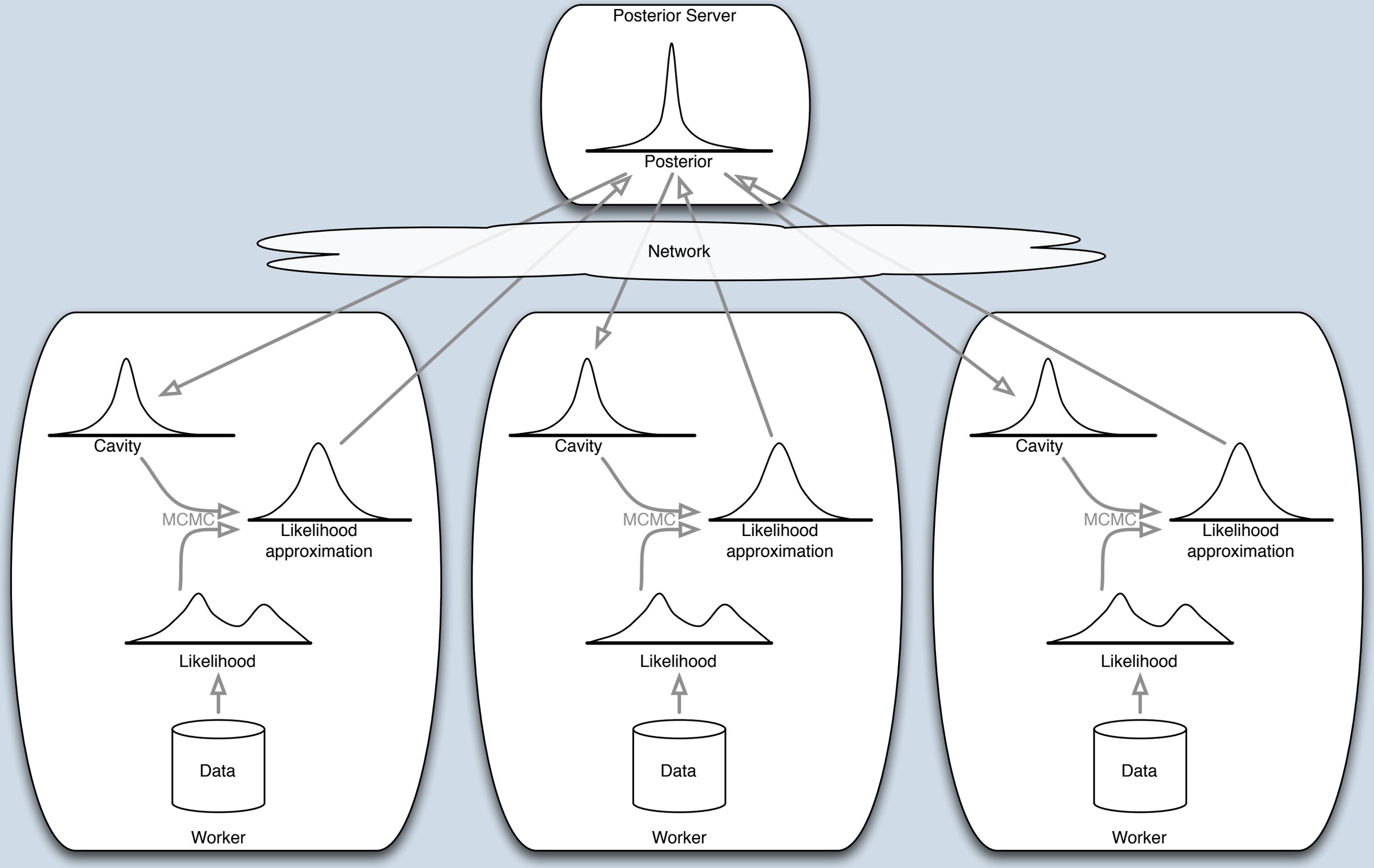
$$x_j \sim \exp\left(\left(\theta_0 + \textstyle\sum_{k\neq j}\lambda_k\right)^\top s(x_j) + l_j(x_j) - A\left(\theta_0 + \textstyle\sum_{k\neq j}\lambda_k, 1\right)\right)$$

  - Outer loop: update auxiliary variables

$$(\theta'_j)^{(t)} = \theta_0 + \sum_{j=1}^{m}\lambda_j^{(t-1)}$$

- Distributed learning:
  - Communicate with master for approximate conditional $\theta_0 + \Sigma_{k\neq j}\lambda_k$.

Posterior Server Architecture

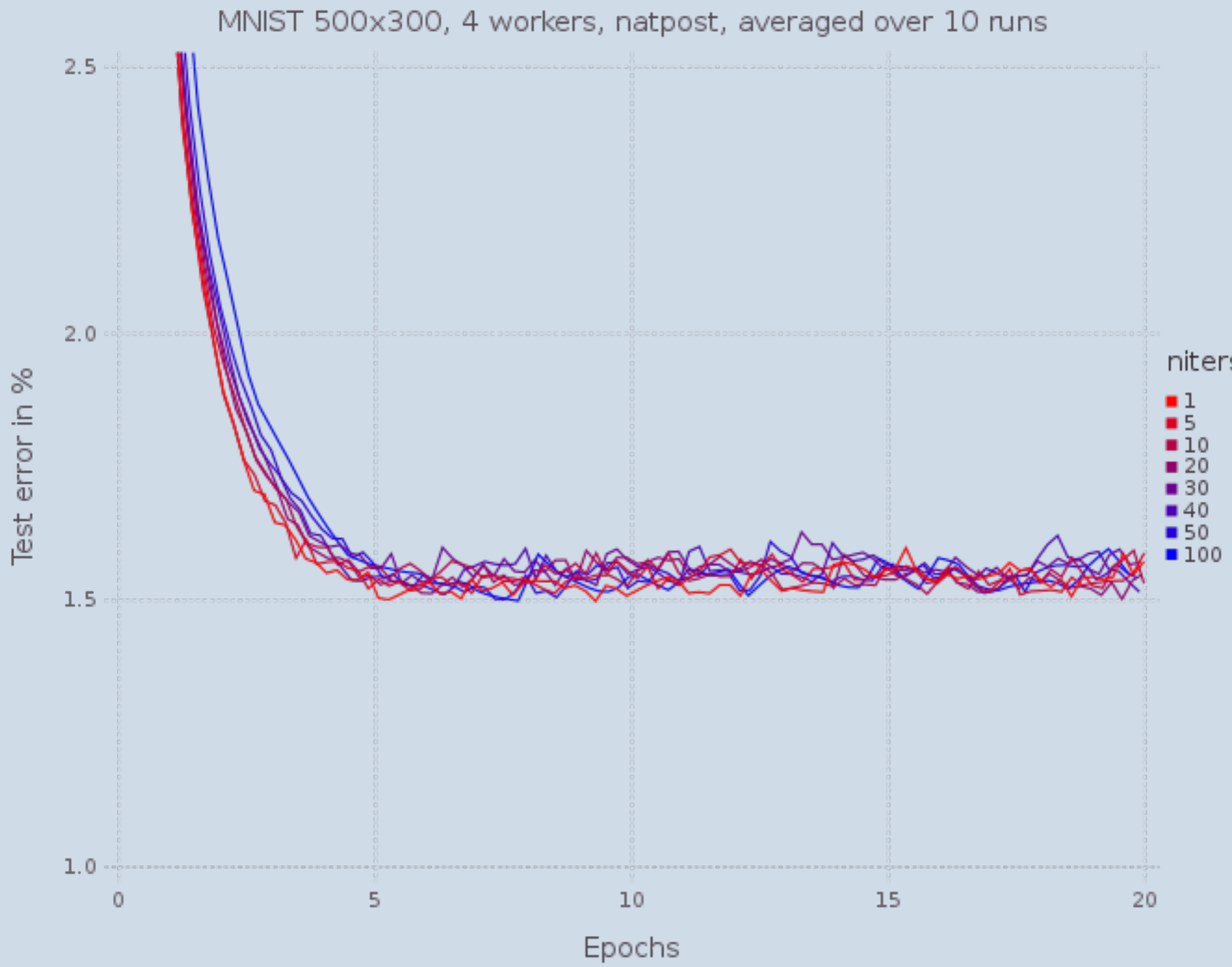


Experiments on Distributed Bayesian Neural Networks

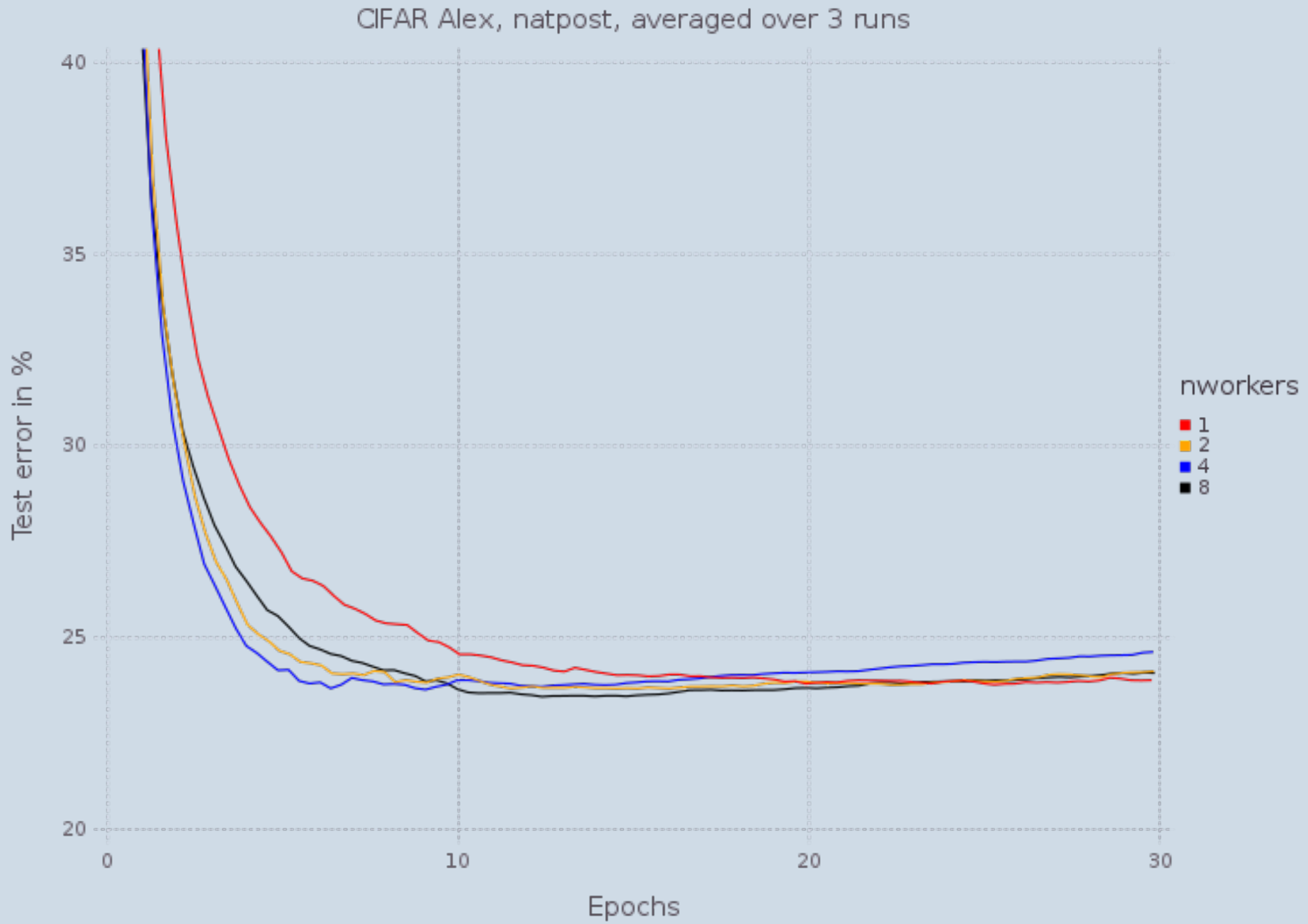
- Bayesian approach to learning neural network:
  - compute parameter posterior given complex neural network likelihood.
  - Diagonal covariance Gaussian prior and exponential-family approximation.
- Two datasets and architectures: MNIST fully-connected, CIFAR10 convnet.

- Implementation in Julia.
  - Workers are cores on a server.
  - SGLD sampler with adaptive mass parameter (preconditioner).
    - Adagrad [Duchi et al 2011]/RMSprop [Tieleman & Hinton 2012] type adaptation.
  - Evaluated on test accuracy.

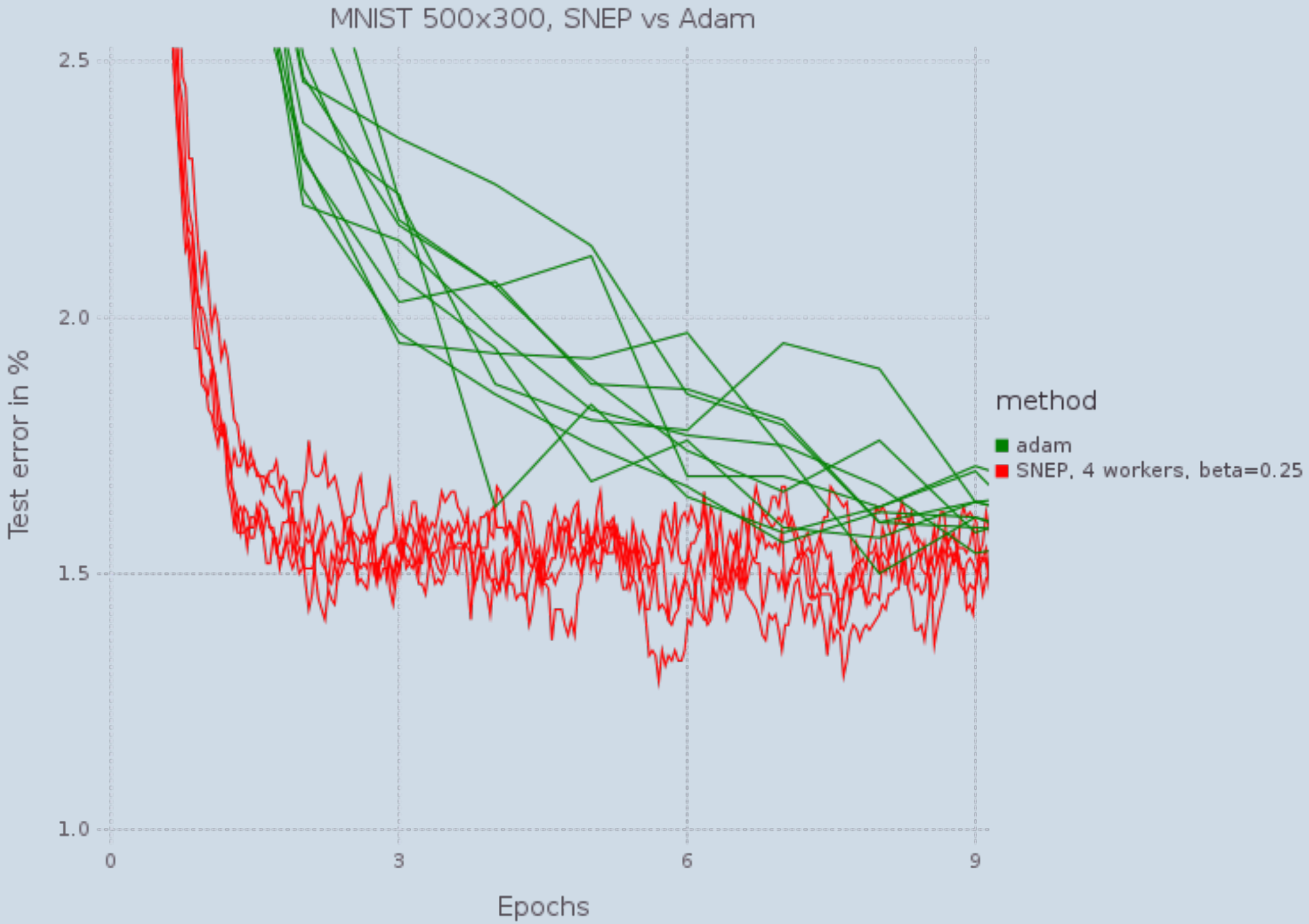
MNIST 500x300 Fully-Connected, Varying #synciters



CIFAR10 AlexNet, Varying #workers



MNIST 500x300 Fully-Connected, vs Adam



Concluding Remarks

- Bayesian framework should continue to be important in era of Big Data.

- Thank you!

Gatsby Charitable Foundation