SCALING AND GENERALIZING APPROXIMATE BAYESIAN INFERENCE

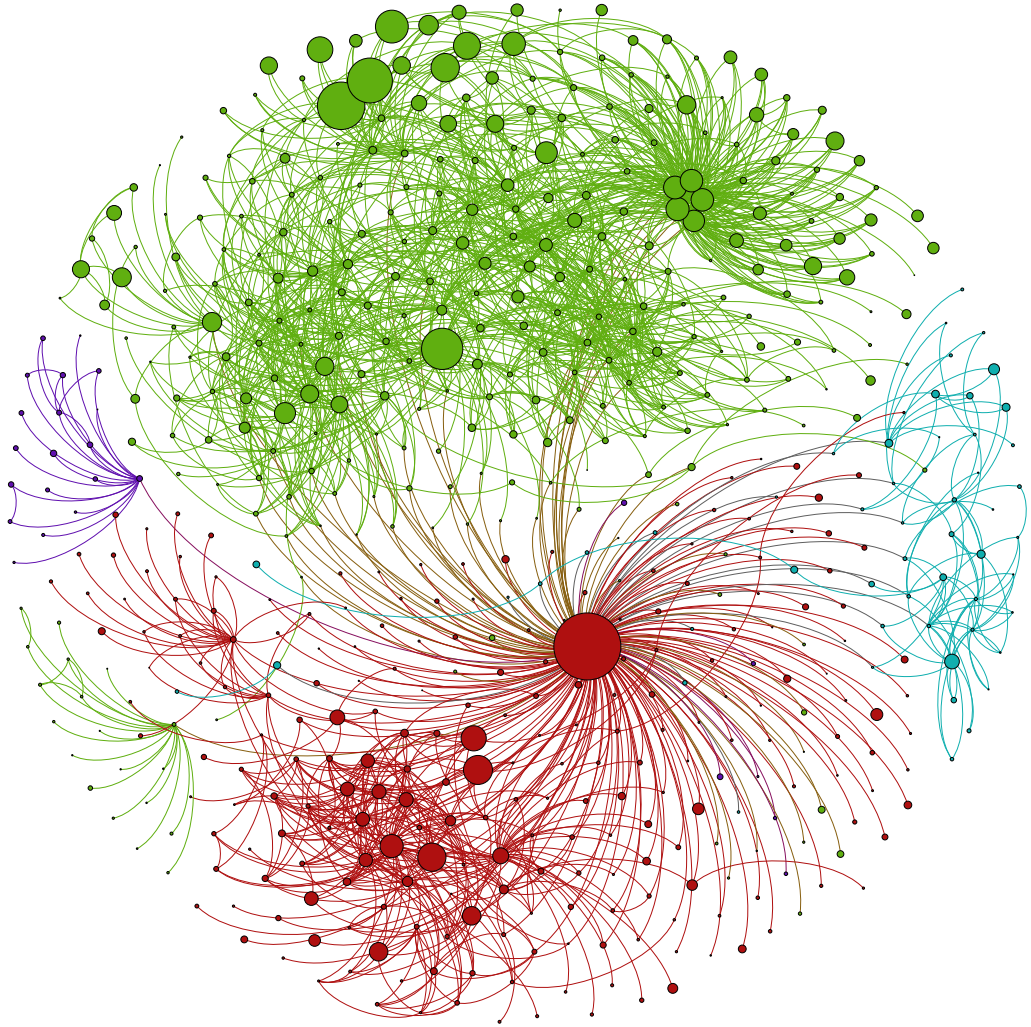
David M. Blei
Departments of Computer Science and Statistics
Columbia University

PROBABILISTIC MACHINE LEARNING

- ML methods that ***connect domain knowledge to data***.
- Provides a computational methodology for analyzing data
- Goal: A methodology that is ***expressive, scalable, easy to develop***

APPLIED BAYESIAN STATISTICS

- Statistical methods that ***connect domain knowledge to data***.
- Provides a computational methodology for analyzing data
- Goal: A methodology that is ***expressive, scalable, easy to develop***

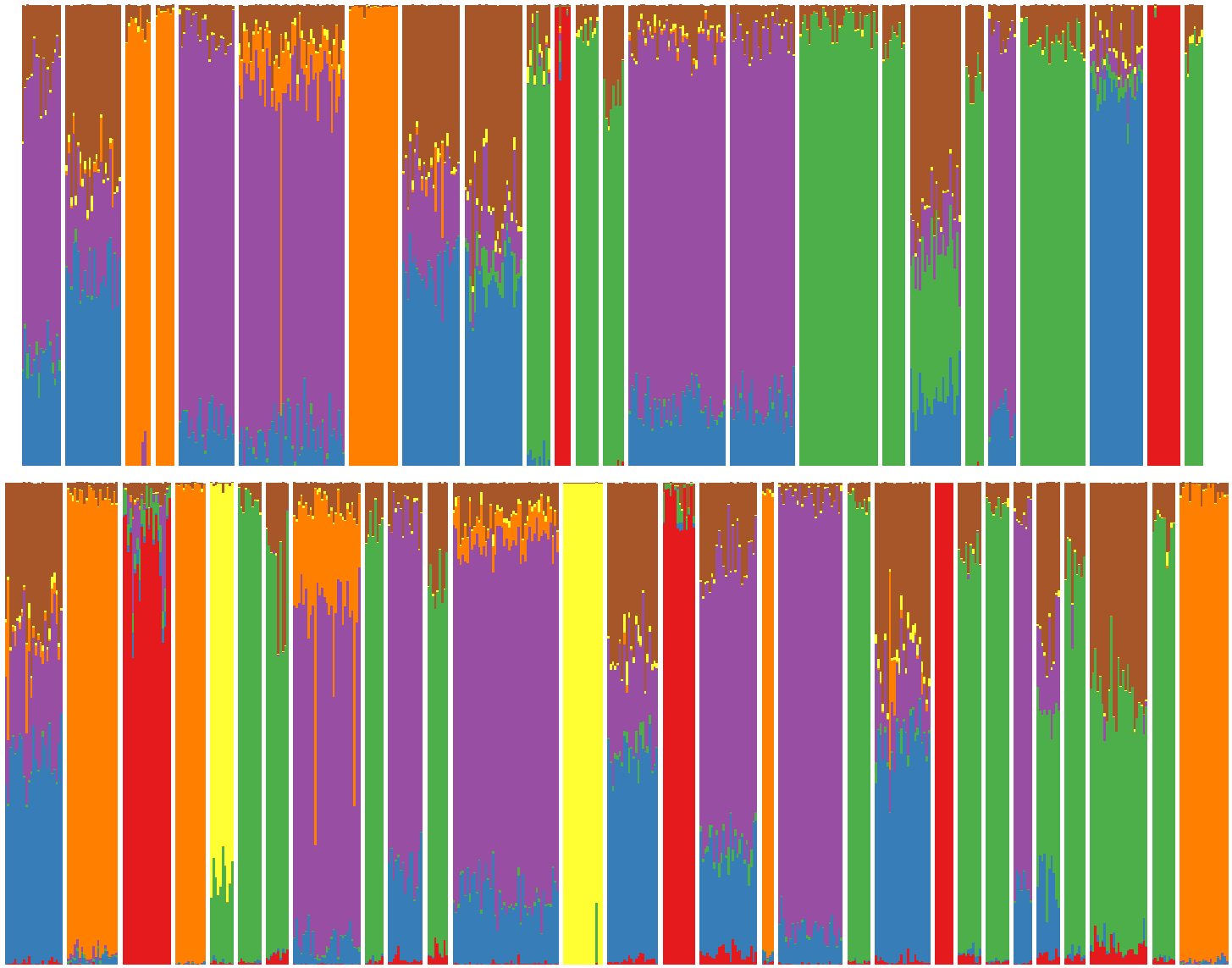


Communities discovered in a 3.7M node network of U.S. Patents

[Gopalan and Blei PNAS 2013]

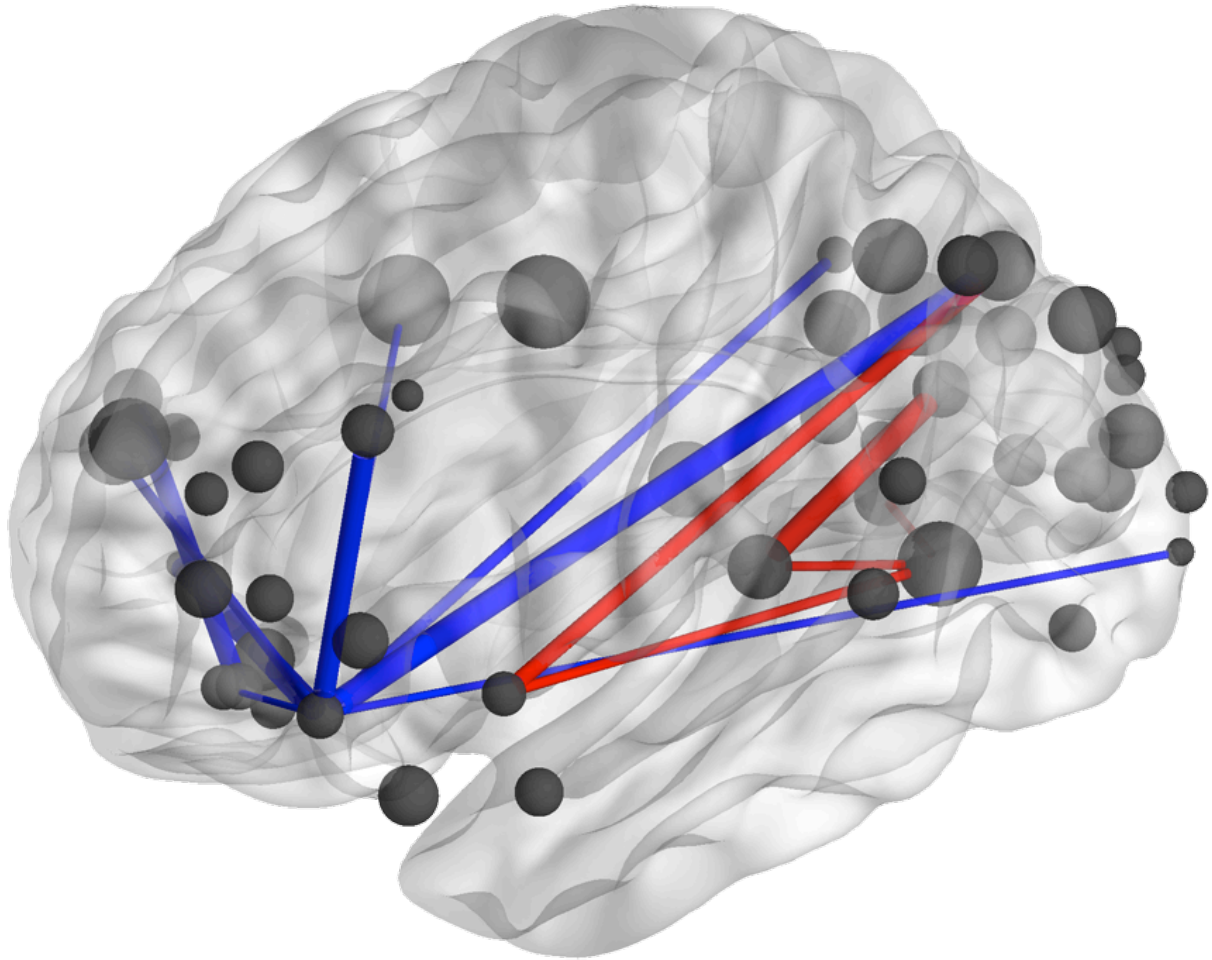
Topics found in 1.8M articles from the New York Times

[Hoffman+ JMLR 2013]



Population analysis of 2 billion genetic measurements

[Gopalan+ Nature Genetics 2016]

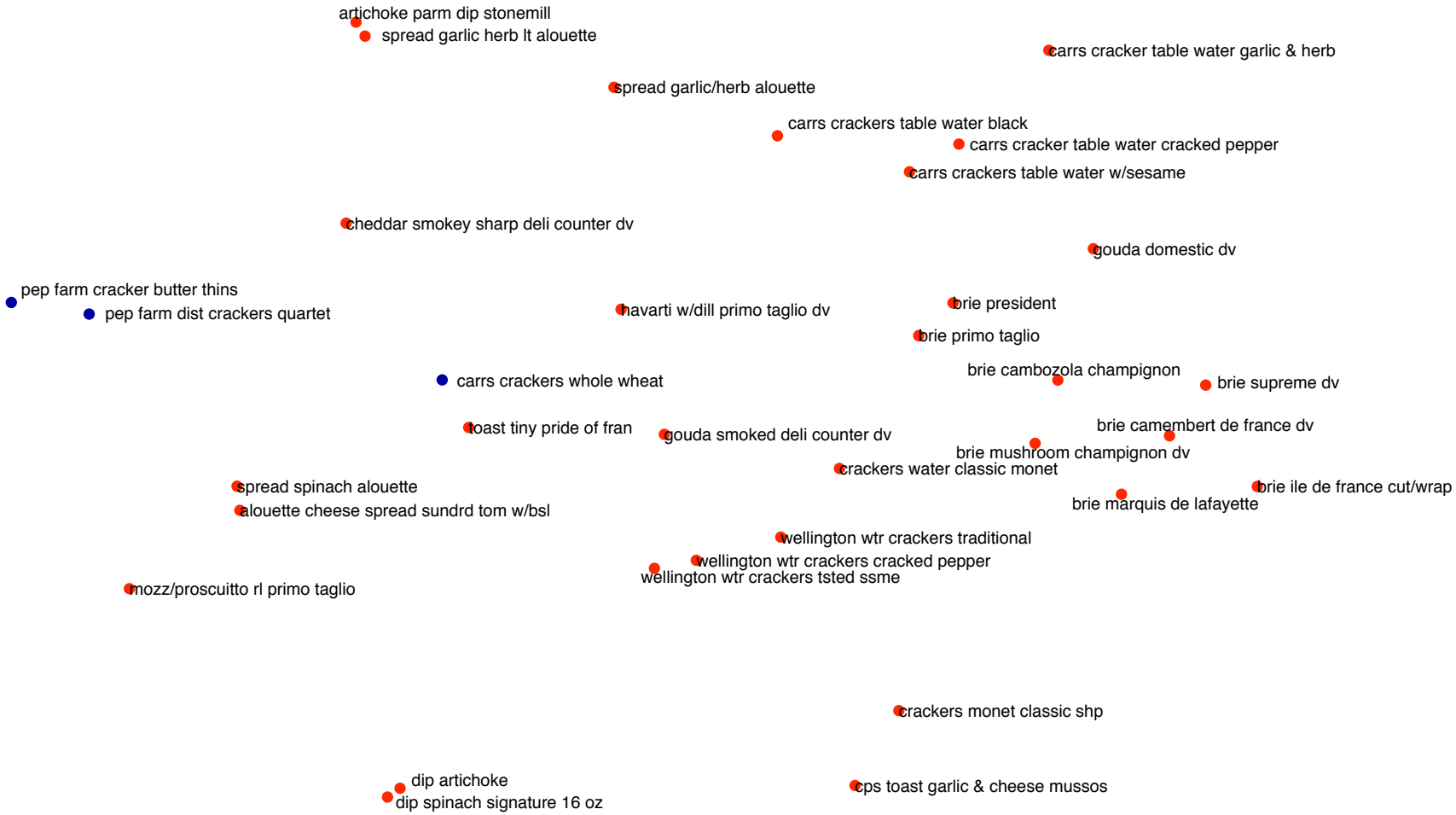


Neuroscience analysis of 220 million fMRI measurements

[Manning+ PLOS ONE 2014]

Analysis of 1.7M taxi trajectories, in Stan

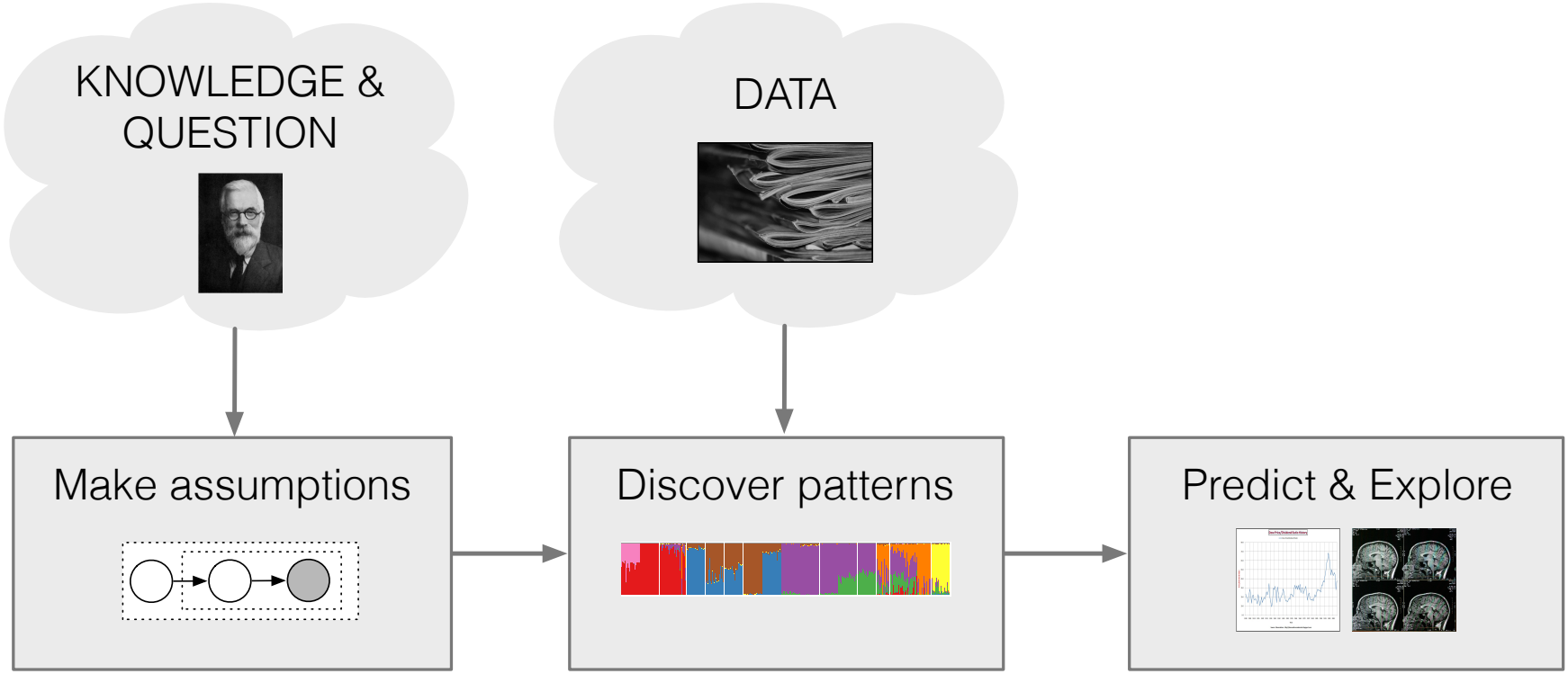
[Kucukelbir+ JMLR 2016]



(Fancy) discrete choice analysis of 5.7M purchases

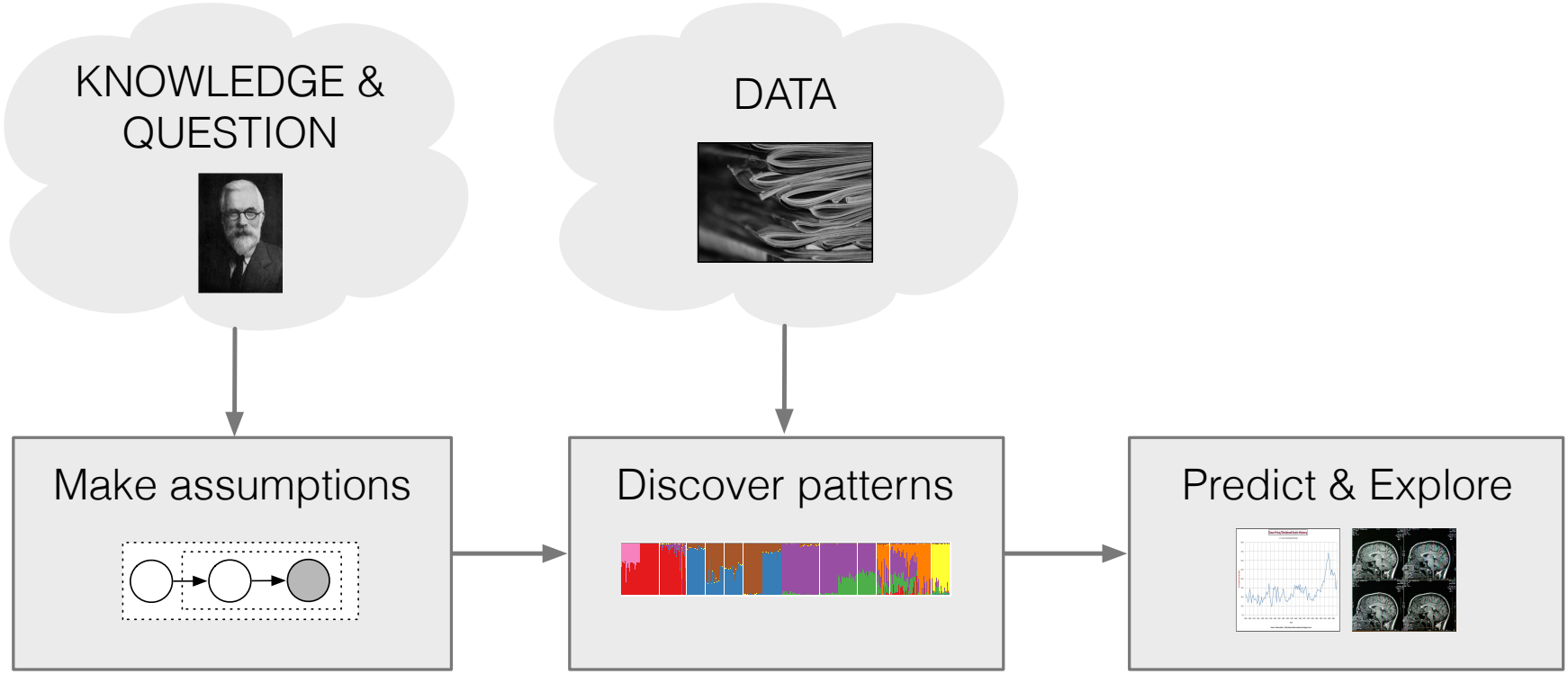
[Ruiz+ 2020]

The probabilistic pipeline

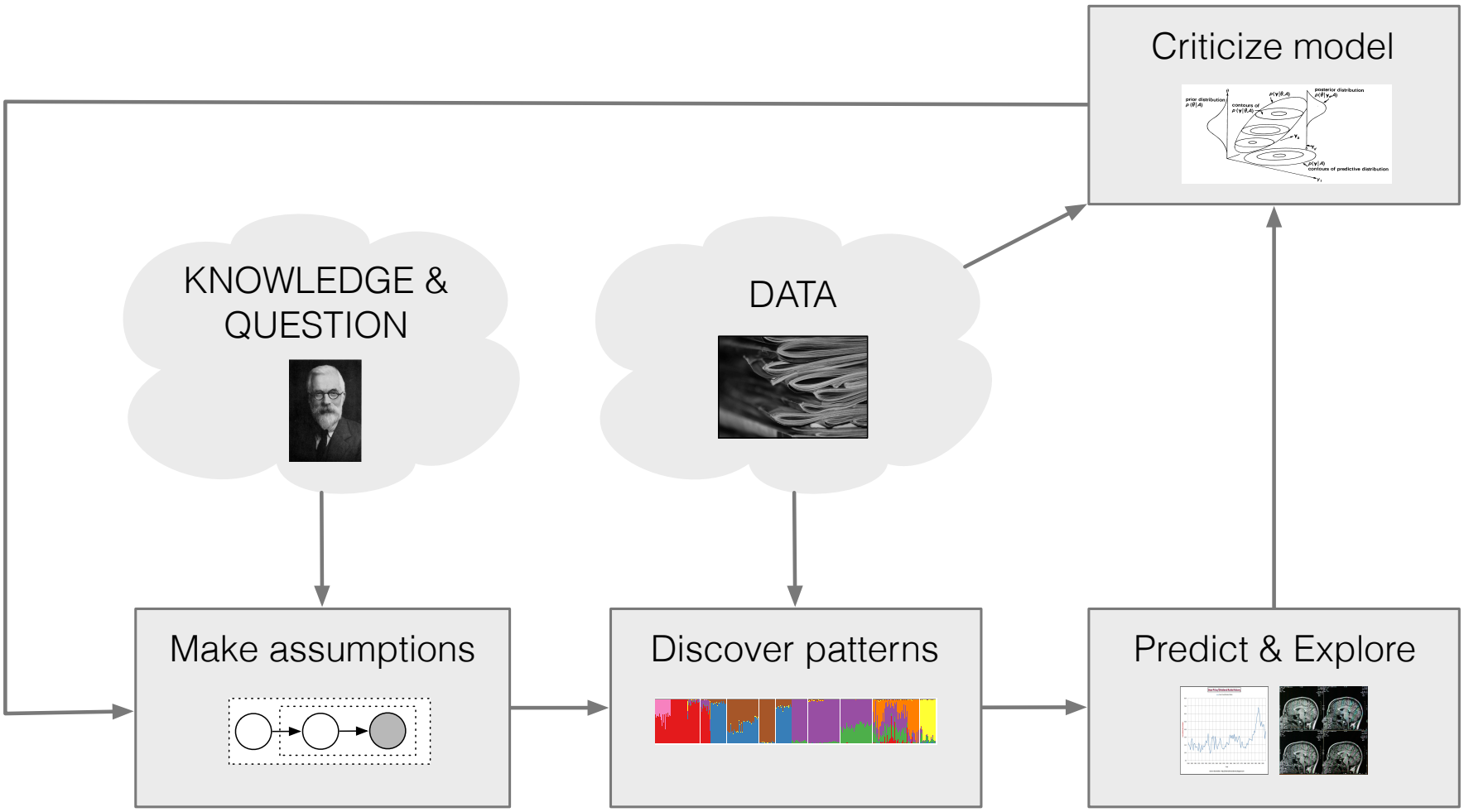


- Customized data analysis is important to many fields.
- Pipeline separates **assumptions**, **computation**, **application**
- Eases collaborative solutions to statistics problems

The probabilistic pipeline



- **Posterior inference** is the key algorithmic problem.
- Answers the question: What does this model say about this data?
- Today: **Scalable** and **general** approaches to posterior inference



[Box, 1980; Rubin, 1984; Gelman+ 1996; Blei, 2014]

Bayesian statistics / Probabilistic machine learning

- A probabilistic model is a joint distribution of hidden variables $\mathbf{z}$ and observed variables $\mathbf{x}$,

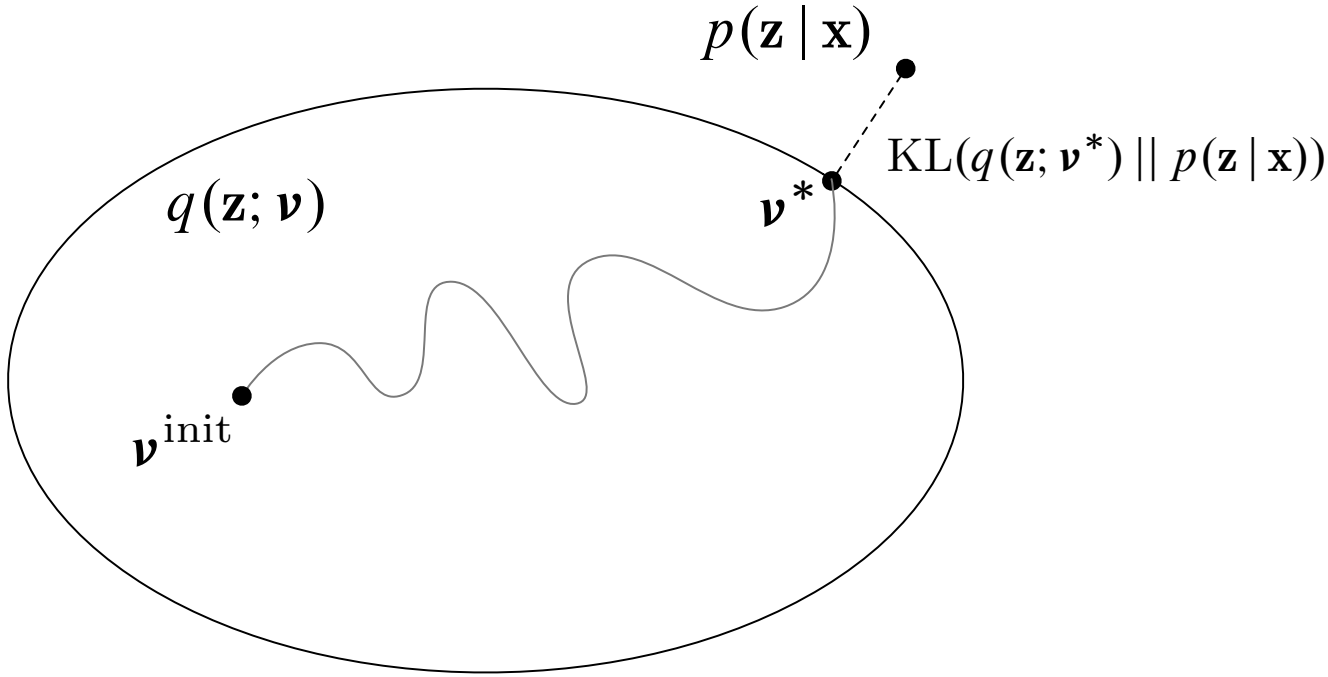
$$p(\mathbf{z}, \mathbf{x}).$$

- Inference about the unknowns is through the **posterior**, the conditional distribution of the hidden variables given the observations

$$p(\mathbf{z} \mid \mathbf{x}) = \frac{p(\mathbf{z}, \mathbf{x})}{p(\mathbf{x})}.$$

- For most interesting models, the denominator is not tractable. We appeal to **approximate posterior inference**.

Variational inference

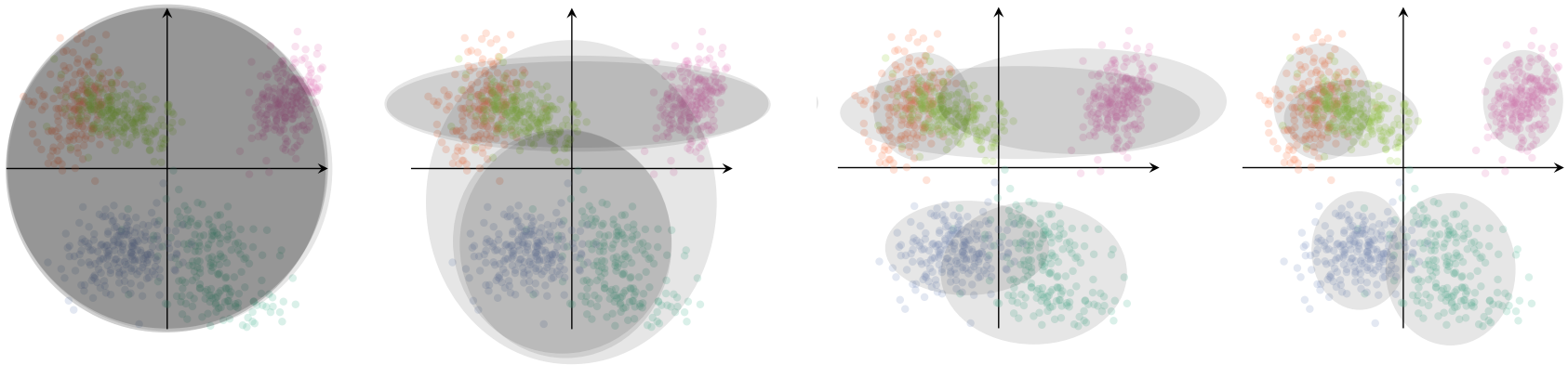


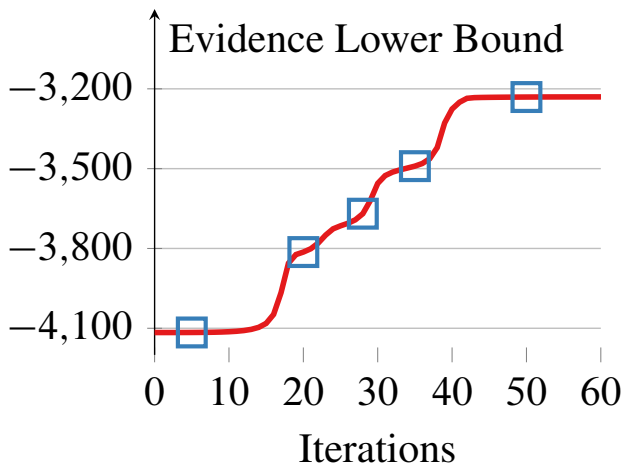
- VI solves **inference** with **optimization**.
  (Contrast this with MCMC.)
- Posit a **variational family** of distributions over the latent variables,

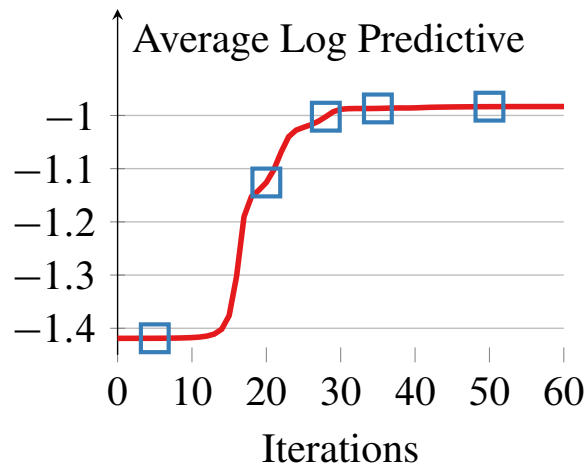
$$q(\mathbf{z}; \boldsymbol{\nu})$$

- Fit the **variational parameters** $\boldsymbol{\nu}$ to be close (in KL) to the exact posterior.

Example: Mixture of Gaussians

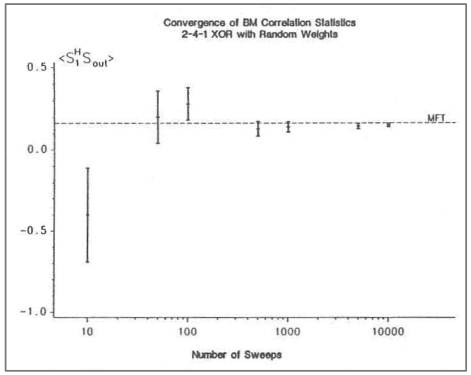




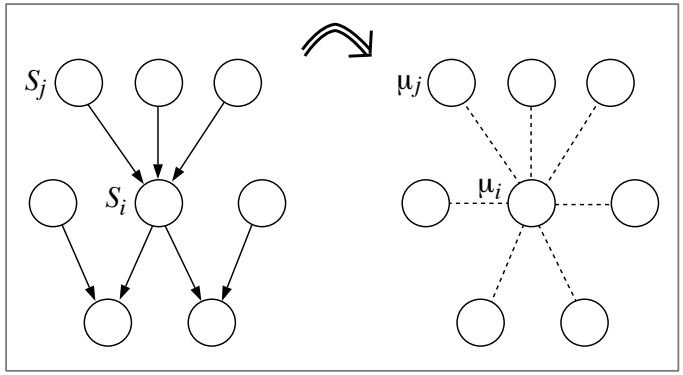


[images by Alp Kucukelbir; Blei+ 2017]

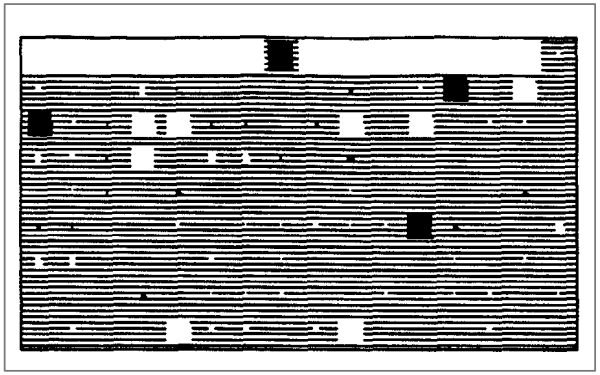
History



[Peterson and Anderson 1987]



[Jordan et al. 1999]

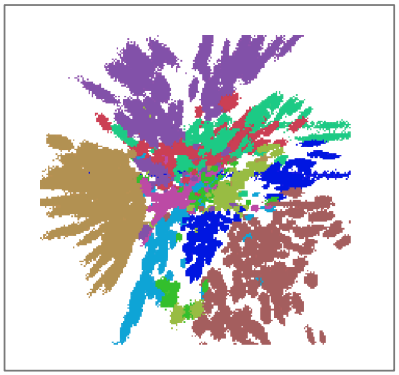


[Hinton and van Camp 1993]

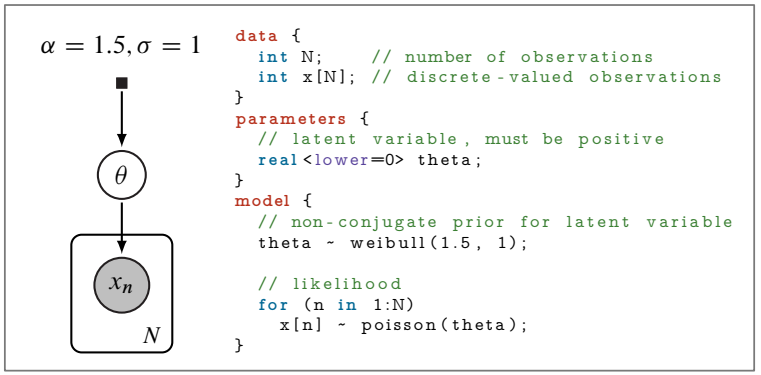
- Variational inference (VI) adapts **ideas from statistical physics** to probabilistic inference. Peterson and Anderson (1987) fits a neural network with mean-field methods.
- In the 1990s, M. Jordan, T. Jaakkola, L. Saul, and Z. Ghahramani **generalized it to many models**. (A review paper is Jordan+ 1999.)
- In parallel, Hinton and Van Camp (1993) developed **mean-field methods for neural networks**. Other applications included MoE (Waterhouse+ 1996), HMMs (MacKay, 1997), and more NN (Barber and Bishop, 1998).

Today

[Kingma and Welling 2013]



[Rezende et al. 2014]

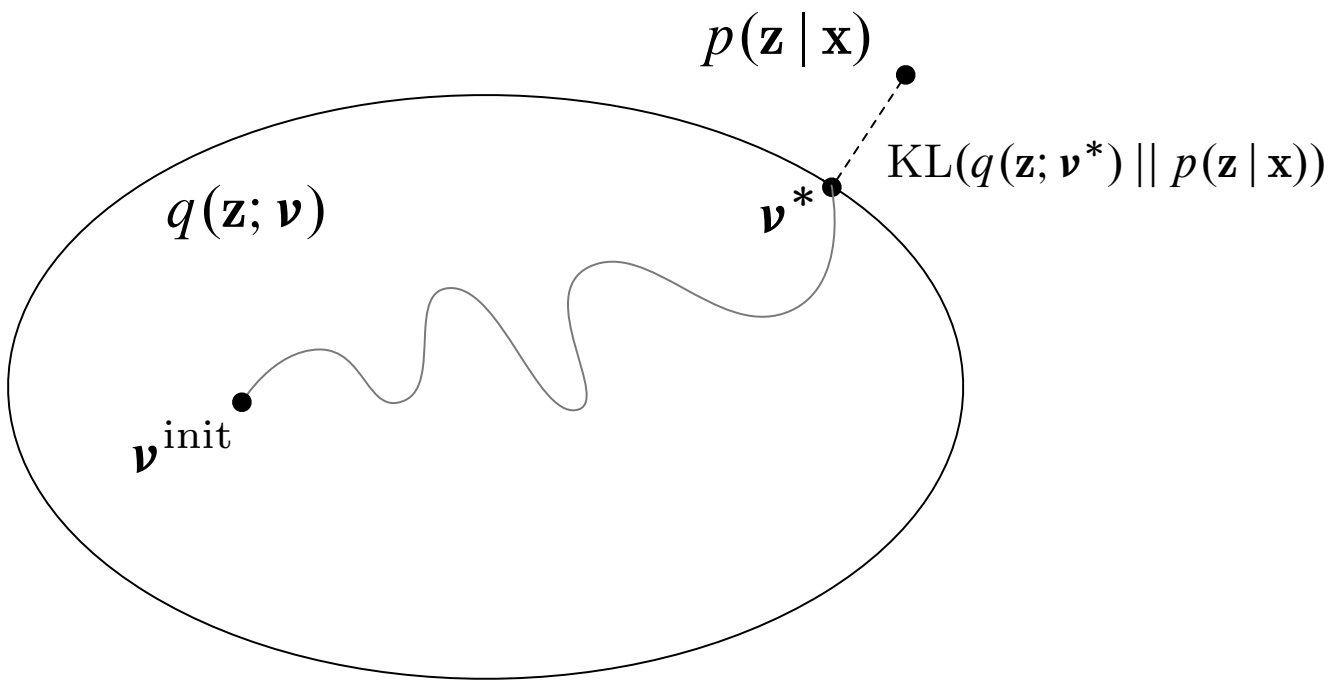


[Kucukelbir et al. 2015]

- There is now a flurry of new work on variational inference, making it **scalable**, **easier to derive**, **faster**, and **more accurate.**

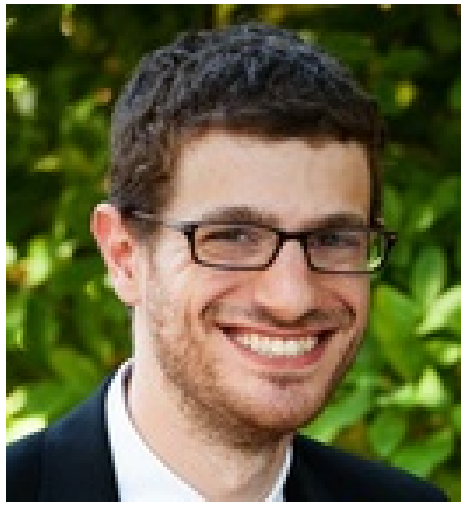
- VI touches many areas: probabilistic programming, reinforcement learning, neural networks, convex optimization, and Bayesian statistics.

Stochastic optimization makes VI better

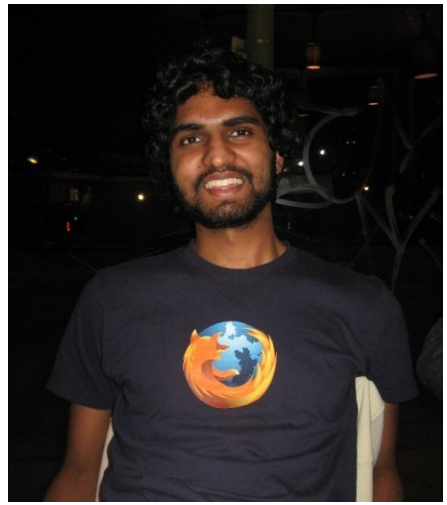


- **Stochastic VI** scales up VI to massive data. [Hoffman+ 2013]
- **Black box VI** generalizes VI to a wide class of models. [Ranganath+ 2014]

Collaborators



Matt Hoffman
(Google)

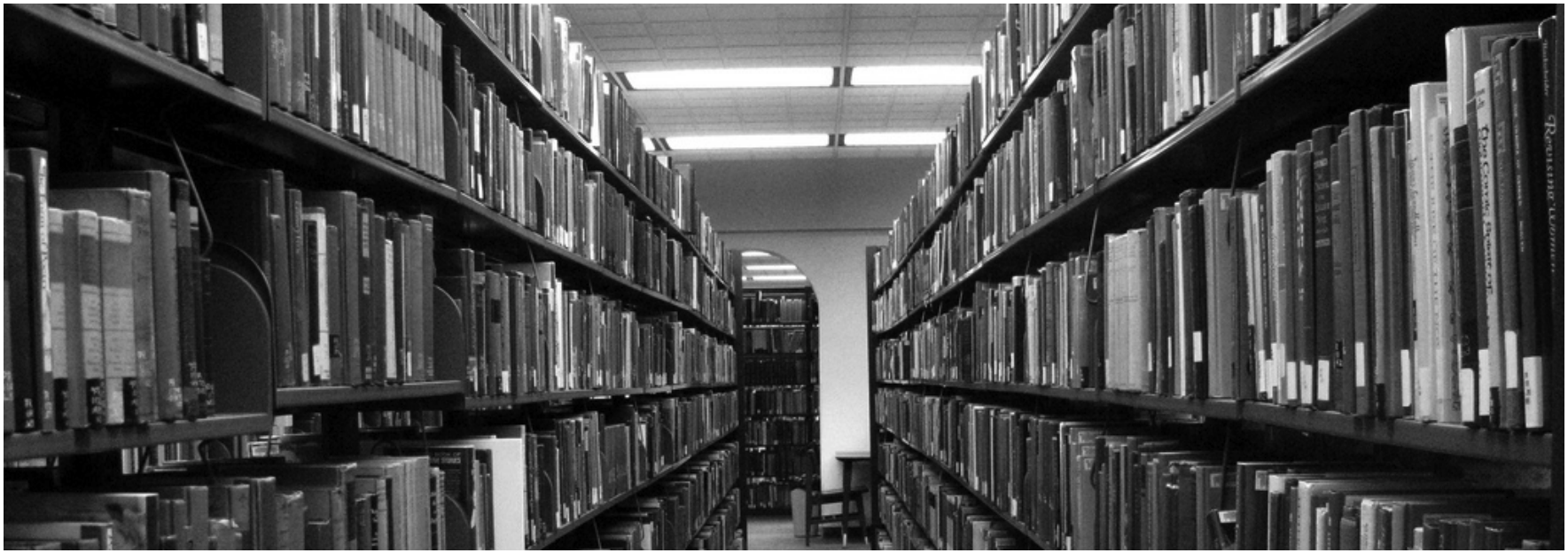


Rajesh Ranganath
(NYU)

Alp Kucukelbir
(Fero Labs)

Stochastic Variational Inference

Motivation: Topic Modeling



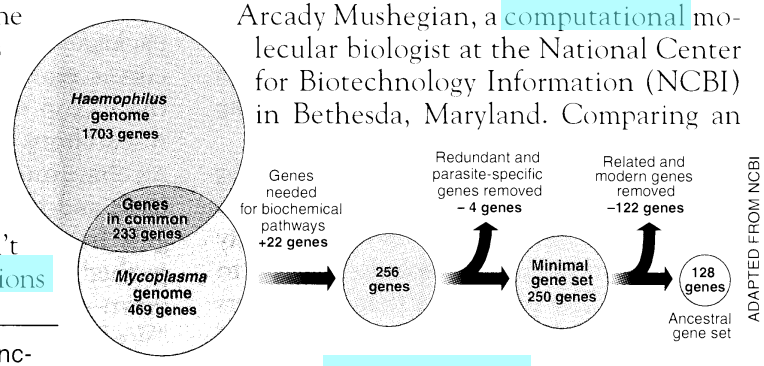
Topic models use posterior inference to discover the hidden thematic structure in a large collection of documents.

Model: Latent Dirichlet Allocation (LDA)

Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK—How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an



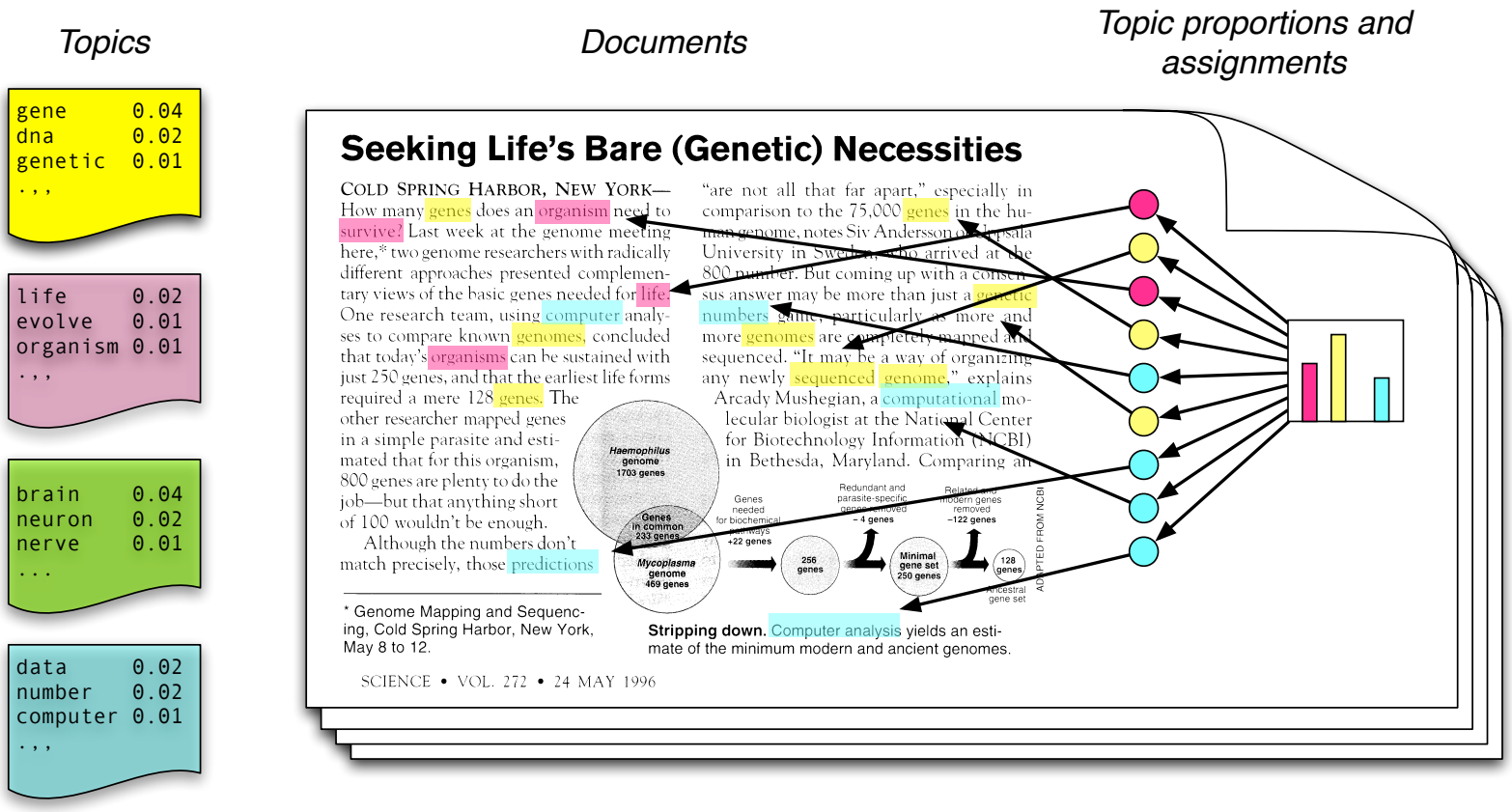
Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

SCIENCE • VOL. 272 • 24 MAY 1996

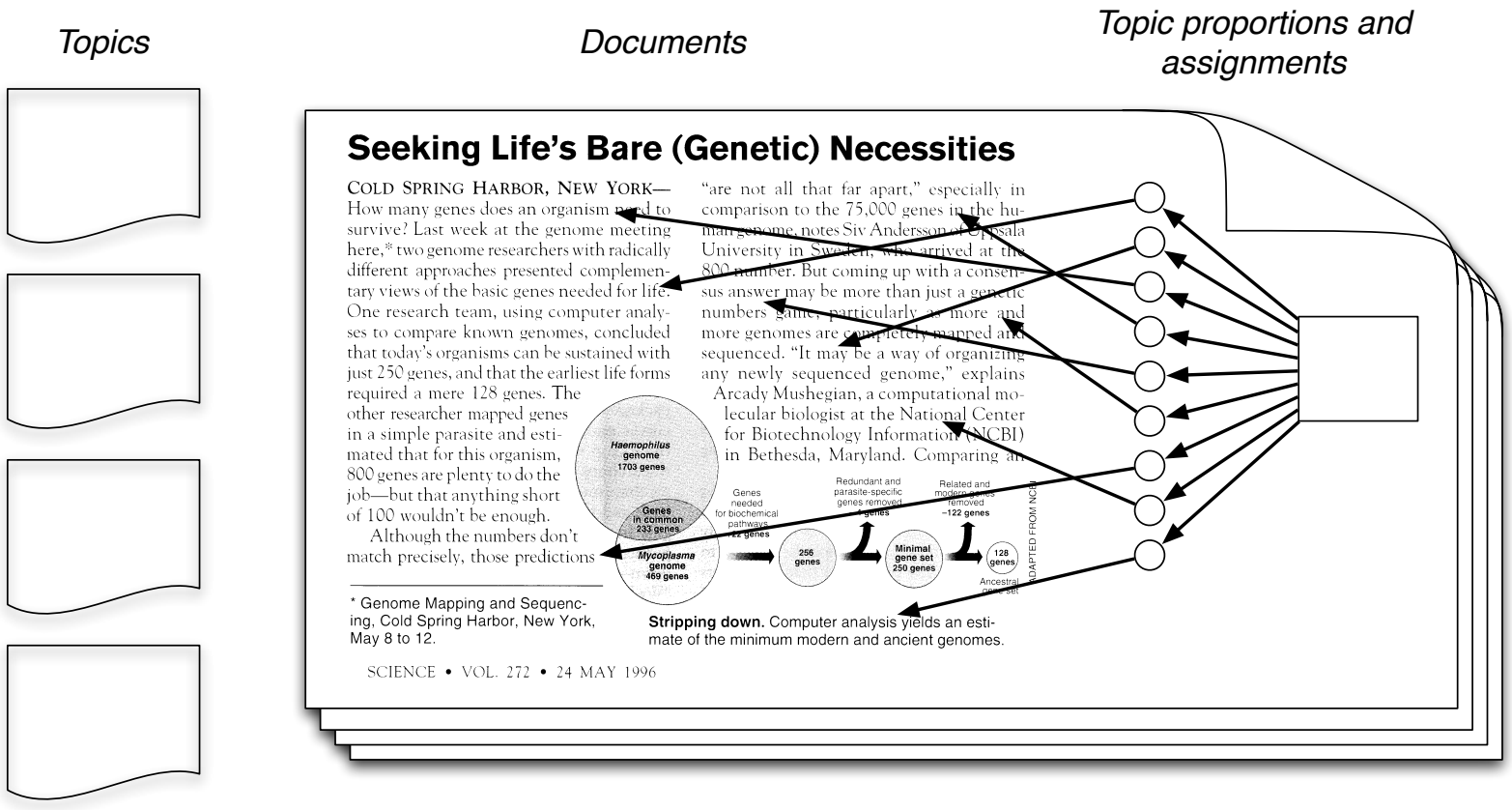
Documents exhibit multiple topics.

Latent Dirichlet Allocation (LDA)



- Each **topic** is a distribution over words
- Each **document** is a mixture of corpus-wide topics
- Each **word** is drawn from one of those topics

Latent Dirichlet Allocation (LDA)

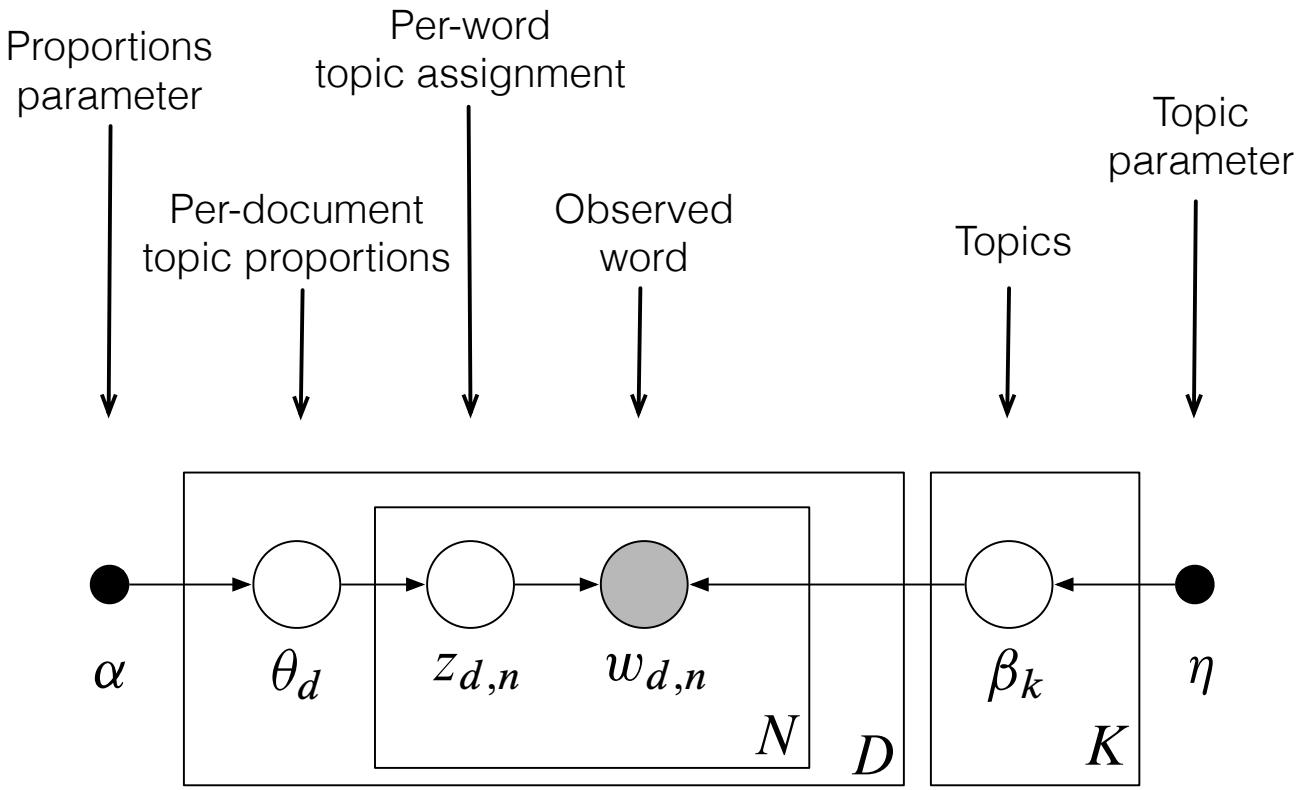


- But we only observe the documents; everything else is hidden.
- So we want to calculate the posterior

$$p(\text{topics, proportions, assignments} \mid \text{documents})$$

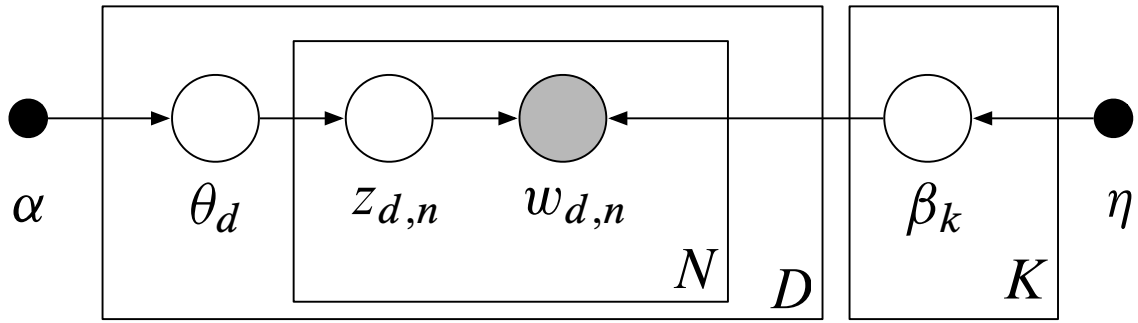
(Note: millions of documents; billions of latent variables)

LDA as a Graphical Model



- A schematic of the **generative process**
- Defines a **factorization of the joint distribution**
- Connects to **assumptions** and **algorithms**

Posterior Inference



- The posterior of the latent variables given the documents is

$$p(\beta, \boldsymbol{\theta}, \mathbf{z} \mid \mathbf{w}) = \frac{p(\beta, \boldsymbol{\theta}, \mathbf{z}, \mathbf{w})}{\int_\beta \int_{\boldsymbol{\theta}} \sum_{\mathbf{z}} p(\beta, \boldsymbol{\theta}, \mathbf{z}, \mathbf{w})}.$$

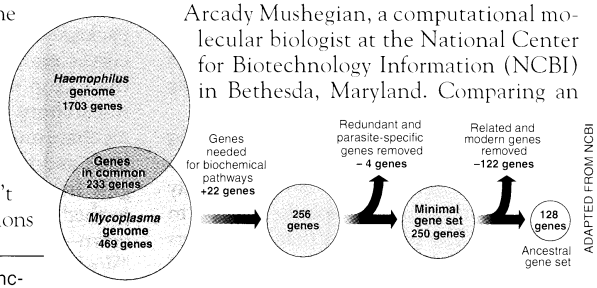
- We can't compute the denominator, the marginal $p(\mathbf{w})$.
- We use variational inference.

Mean-field variational inference for LDA

Seeking Life's Bare (Genetic) Necessities

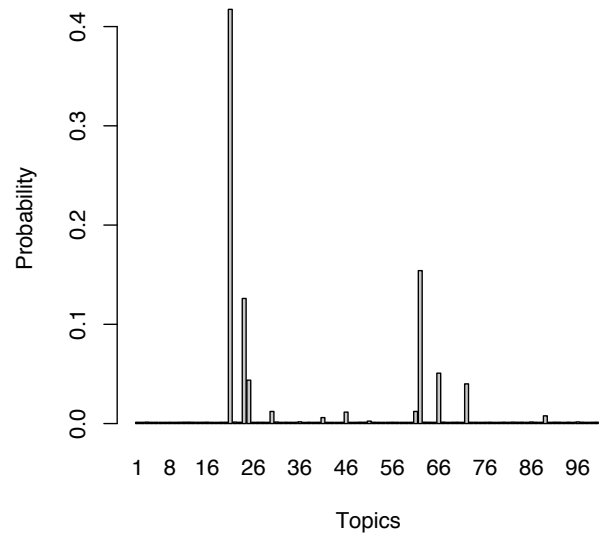
COLD SPRING HARBOR, NEW YORK—How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an



* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

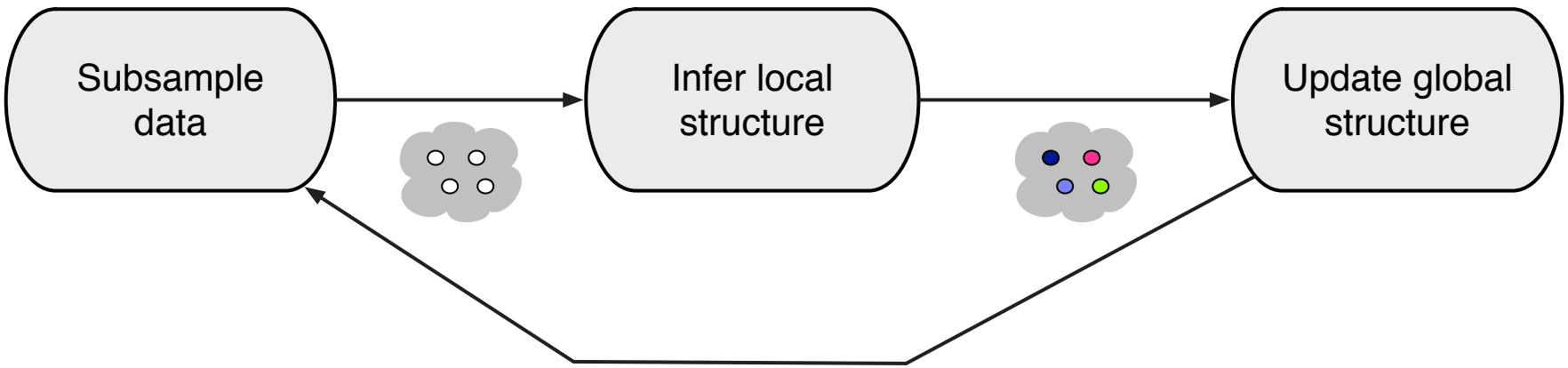


Mean-field variational inference for LDA

human	evolution	disease	computer
genome	evolutionary	host	models
dna	species	bacteria	information
genetic	organisms	diseases	data
genes	life	resistance	computers
sequence	origin	bacterial	system
gene	biology	new	network
molecular	groups	strains	systems
sequencing	phylogenetic	control	model
map	living	infectious	parallel
information	diversity	malaria	methods
genetics	group	parasite	networks
mapping	new	parasites	software
project	two	united	new
sequences	common	tuberculosis	simulations

Topics found in 1.8M articles from the New York Times

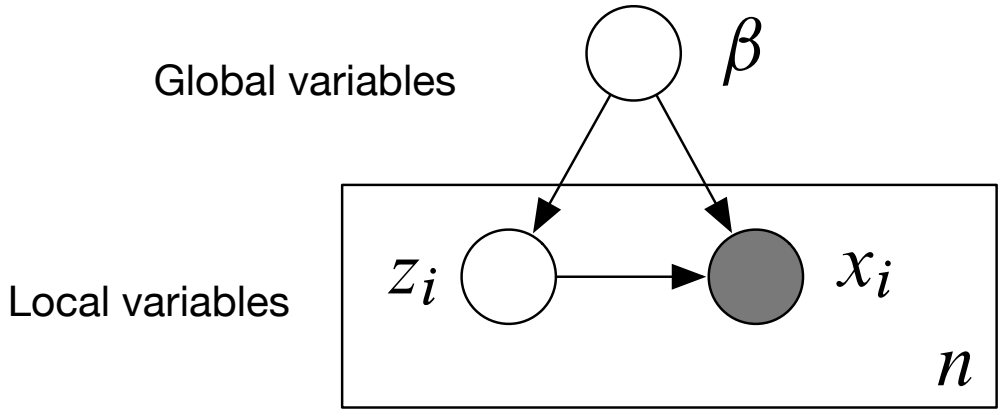
Mean-field VI and Stochastic VI



Road map:

- Define the generic class of conditionally conjugate models
- Derive classical mean-field VI
- Derive stochastic VI, which scales to massive data

Conditionally conjugate models

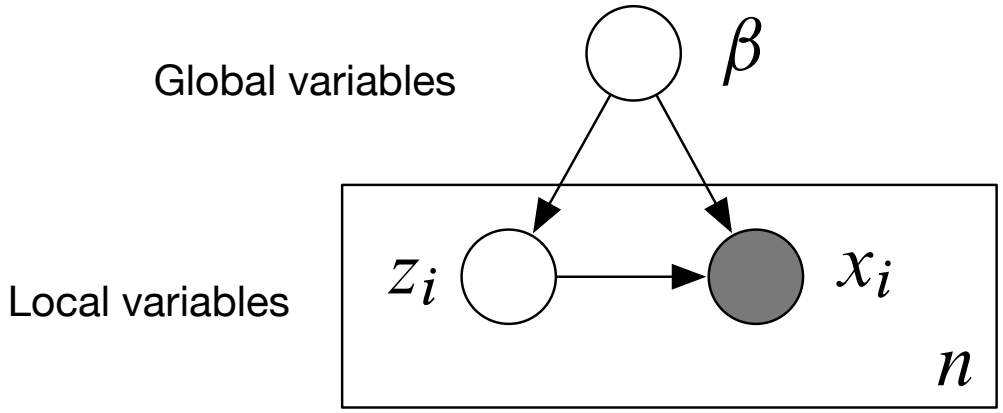


$$p(\beta, \mathbf{z}, \mathbf{x}) = p(\beta) \prod_{i=1}^{n} p(z_i, x_i \mid \beta)$$

- The observations are $\mathbf{x} = x_{1:n}$.
- The **local** variables are $\mathbf{z} = z_{1:n}$.
- The **global** variables are β.
- The ith data point x_i only depends on z_i and β.

Compute $p(\beta, \mathbf{z} \mid \mathbf{x})$.

Conditionally conjugate models



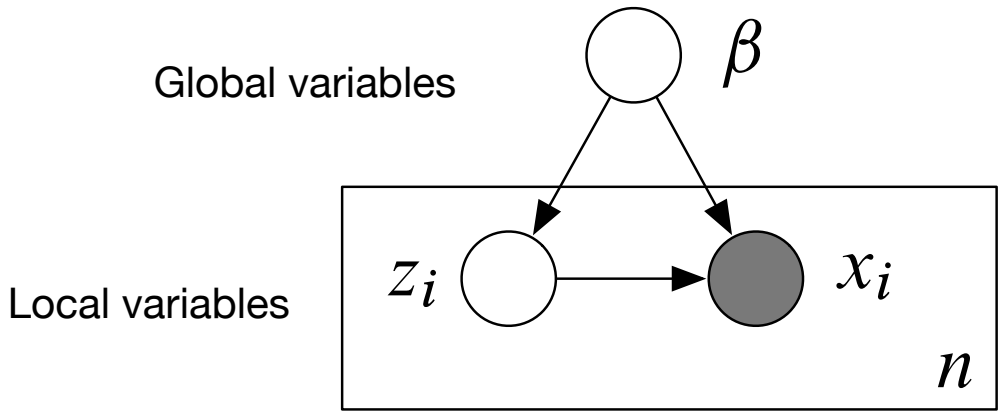
$$p(\beta, \mathbf{z}, \mathbf{x}) = p(\beta) \prod_{i=1}^{n} p(z_i, x_i \mid \beta)$$

- A **complete conditional** is the conditional of a latent variable given the observations and other latent variables.
- Assume each complete conditional is in the exponential family,

$$p(z_i \mid \beta, x_i) = \text{expfam}(z_i\,;\, \eta_\ell(\beta, x_i))$$
$$p(\beta \mid \mathbf{z}, \mathbf{x}) = \text{expfam}(\beta\,;\, \eta_g(\mathbf{z}, \mathbf{x})),$$

where $\text{expfam}(z\,;\,\eta) = h(z)\exp\{\eta^\top z - a(\eta)\}$.

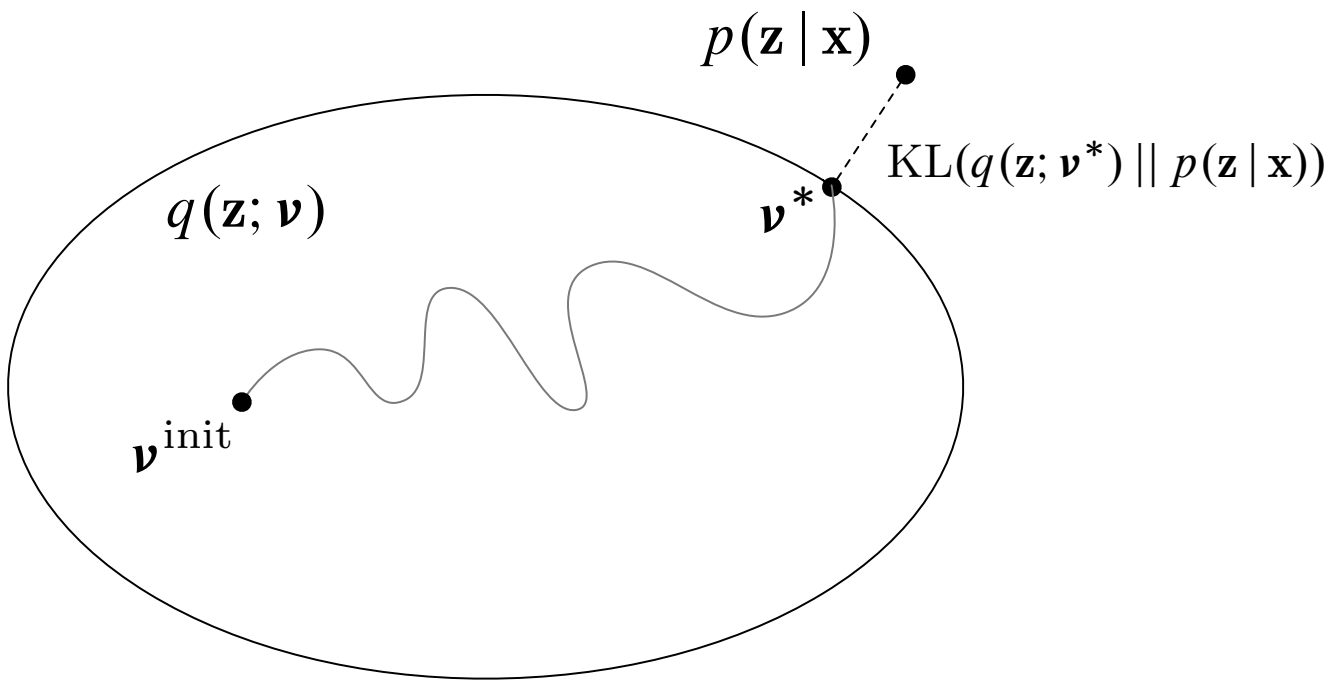
Conditionally conjugate models



$$p(\beta, \mathbf{z}, \mathbf{x}) = p(\beta) \prod_{i=1}^{n} p(z_i, x_i \mid \beta)$$

- Bayesian mixture models
- Time series models
  (HMMs, linear dynamic systems)
- Factorial models
- Matrix factorization
  (factor analysis, PCA, CCA)
- Dirichlet process mixtures, HDPs
- Multilevel regression
  (linear, probit, Poisson)
- Stochastic block models
- Mixed-membership models
  (LDA and some variants)

Variational inference



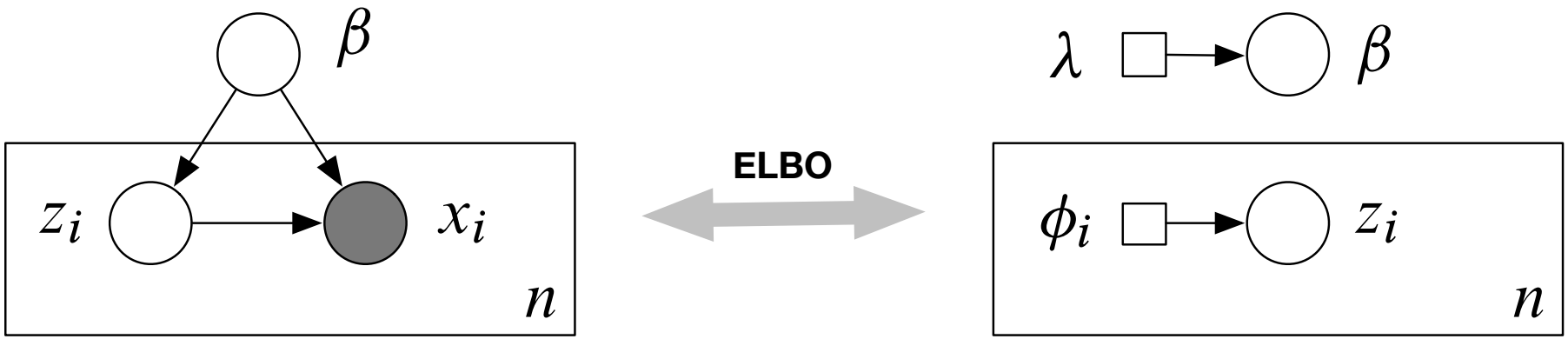
Minimize KL between $q(\beta, \mathbf{z}; \boldsymbol{\nu})$ and the posterior $p(\beta, \mathbf{z} | \mathbf{x})$.

The evidence lower bound

$$\mathcal{L}(\boldsymbol{\nu}) = \underbrace{\mathbb{E}_q\left[\log p(\beta, \mathbf{z}, \mathbf{x})\right]}_{\text{Expected complete log likelihood}} - \underbrace{\mathbb{E}_q\left[\log q(\beta, \mathbf{z}; \boldsymbol{\nu})\right]}_{\text{Negative entropy}}$$

- KL is intractable; VI optimizes the **evidence lower bound** (ELBO) instead.
  - It is a lower bound on $\log p(\mathbf{x})$.
  - Maximizing the ELBO is equivalent to minimizing the KL.
- The ELBO trades off two terms.
  - The first term prefers $q(\cdot)$ to place its mass on the MAP estimate.
  - The second term encourages $q(\cdot)$ to be diffuse.
- Caveat: The ELBO is not convex.

Mean-field variational inference



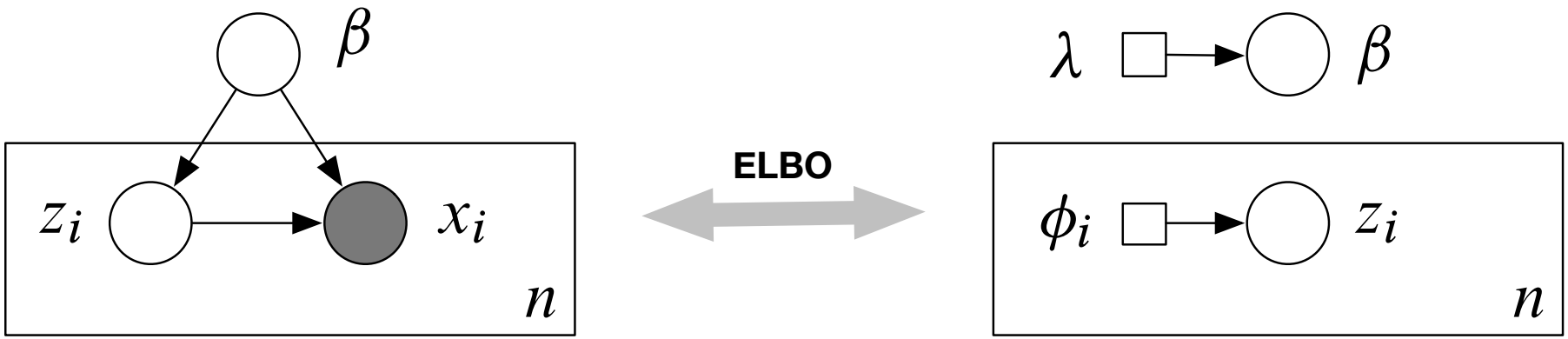
- The form of $q(\beta, \mathbf{z})$ defines the **variational family**.
- The **mean-field family** is fully factorized,

$$q(\beta, \mathbf{z}\,;\, \lambda, \boldsymbol{\phi}) = q(\beta\,;\, \lambda) \prod_{i=1}^{n} q(z_i\,;\, \phi_i).$$

- Each factor is the same family as the model's complete conditional.

$$p(\beta \mid \mathbf{z}, \mathbf{x}) = \text{expfam}(\beta\,;\, \eta_g(\mathbf{z}, \mathbf{x}))$$
$$q(\beta\,;\, \lambda) = \text{expfam}(\beta\,;\, \lambda)$$

Mean-field variational inference



- Optimize the ELBO,

$$\mathcal{L}(\lambda, \boldsymbol{\phi}) = \mathbb{E}_q[\log p(\beta, \mathbf{z}, \mathbf{x})] - \mathbb{E}_q[\log q(\beta, \mathbf{z})].$$

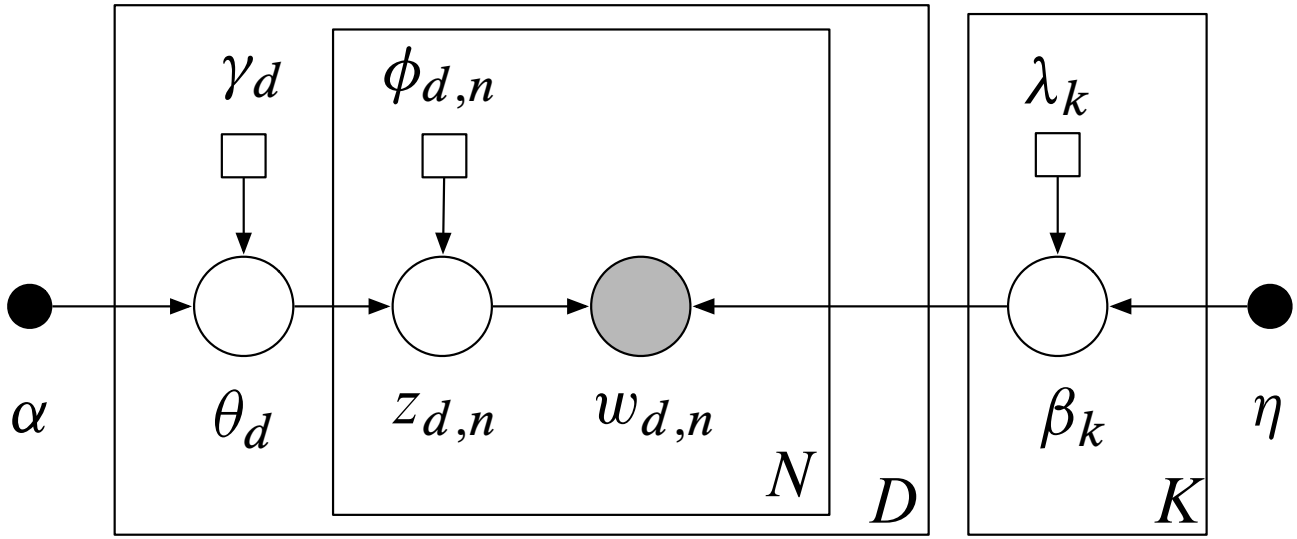
- Traditional VI uses coordinate ascent

$$\lambda^* = \mathbb{E}_\phi\left[\eta_g(\mathbf{z}, \mathbf{x})\right];\ \phi_i^* = \mathbb{E}_\lambda[\eta_\ell(\beta, x_i)]$$

  It iteratively updates each parameter [Ghahramani and Beal, 2001] .

- Notice the relationship to Gibbs sampling [Gelfand and Smith, 1990] .

Mean-field variational inference for LDA



- Local variables are per-document variables θ_d and $\mathbf{z}_d$.
  Global variables are the topics $\beta_1, \ldots, \beta_K$.
- The mean-field family is

$$q(\beta, \boldsymbol{\theta}, \mathbf{z}) = \prod_{k=1}^{K} q(\beta_k ; \lambda_k) \prod_{d=1}^{D} q(\theta_d ; \gamma_d) \prod_{n=1}^{N} q(z_{d,n} ; \phi_{d,n})$$

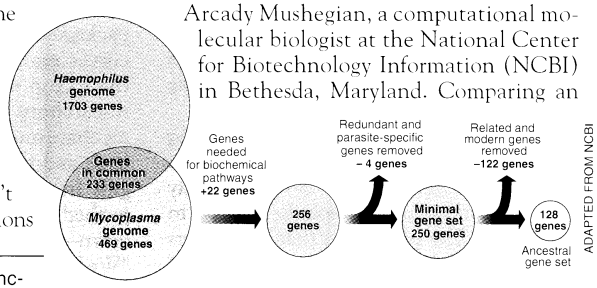
- Classical VI iteratively updates each variational parameter.

Mean-field variational inference for LDA

Seeking Life's Bare (Genetic) Necessities

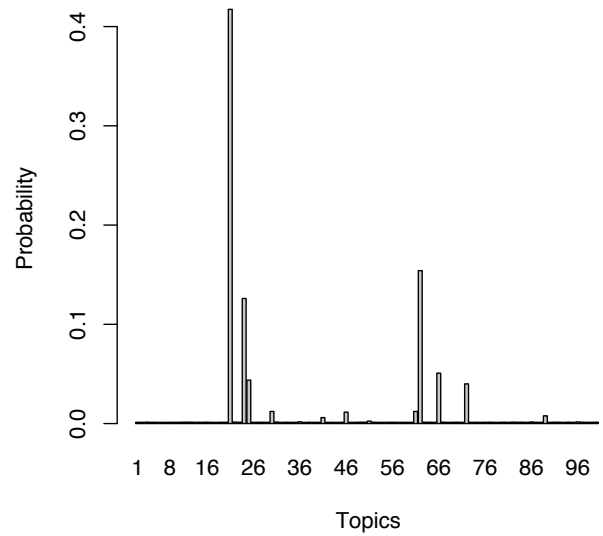
COLD SPRING HARBOR, NEW YORK—How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an



* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

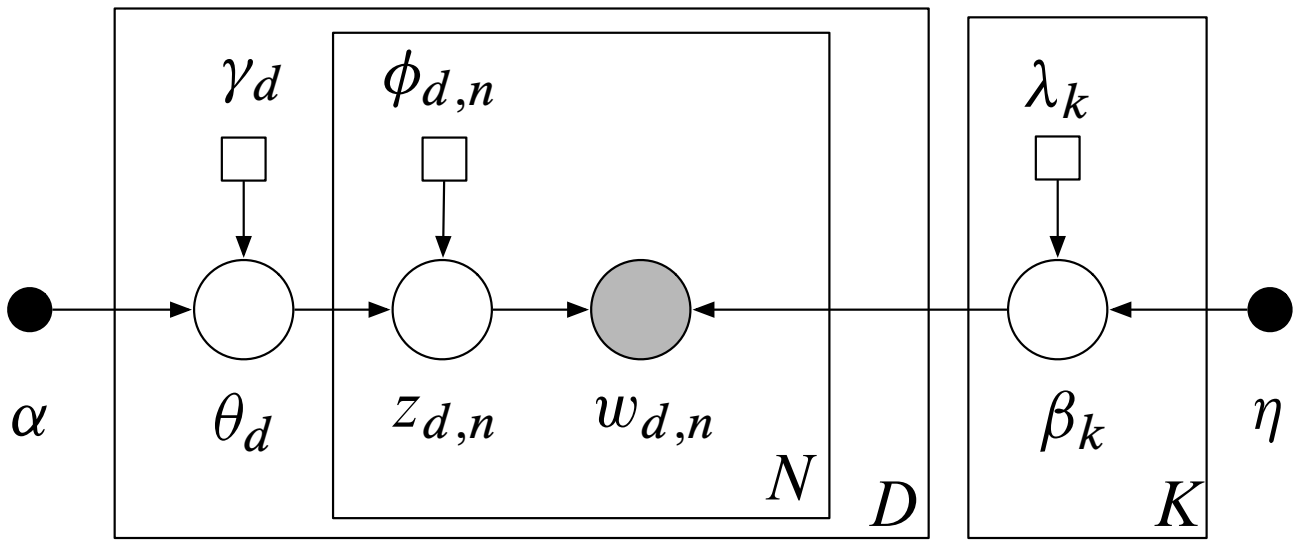
Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.



Mean-field variational inference for LDA

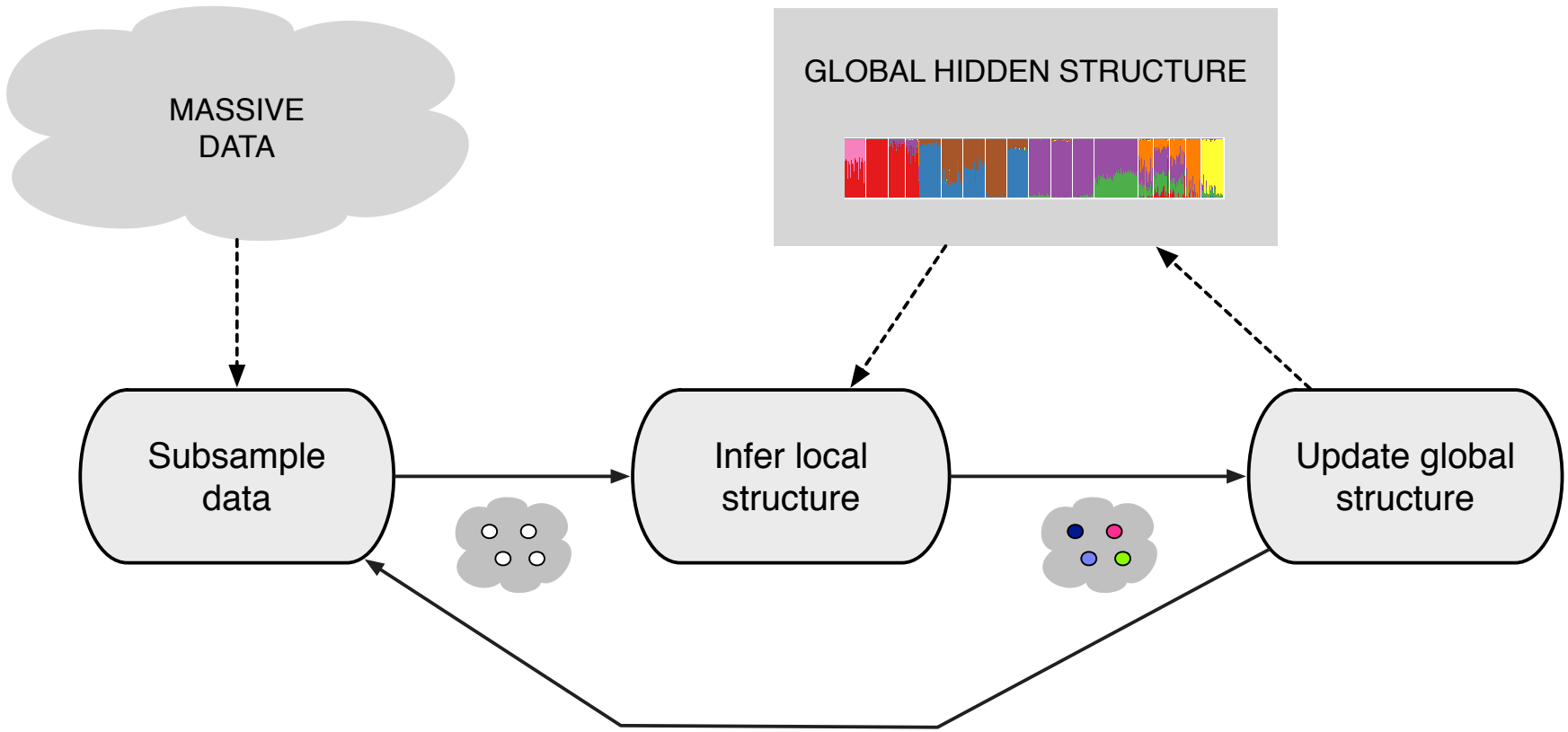
human	evolution	disease	computer
genome	evolutionary	host	models
dna	species	bacteria	information
genetic	organisms	diseases	data
genes	life	resistance	computers
sequence	origin	bacterial	system
gene	biology	new	network
molecular	groups	strains	systems
sequencing	phylogenetic	control	model
map	living	infectious	parallel
information	diversity	malaria	methods
genetics	group	parasite	networks
mapping	new	parasites	software
project	two	united	new
sequences	common	tuberculosis	simulations

Stochastic variational inference



- Classical VI is inefficient:
  - Do some local computation *for each data point*.
  - Aggregate these computations to re-estimate global structure.
  - Repeat.
- This cannot handle massive data.
- **Stochastic variational inference** (SVI) scales VI to massive data.

Stochastic variational inference



Stochastic optimization

A STOCHASTIC APPROXIMATION METHOD[1]

By Herbert Robbins and Sutton Monro

University of North Carolina

1. Summary. Let $M(x)$ denote the expected value at level x of the response to a certain experiment. $M(x)$ is assumed to be a monotone function of x but is unknown to the experimenter, and it is desired to find the solution $x = \theta$ of the equation $M(x) = \alpha$, where α is a given constant. We give a method for making successive experiments at levels $x_1, x_2, \cdots$ in such a way that x_n will tend to θ in probability.

- Replace the gradient with cheaper noisy estimates [Robbins and Monro, 1951]
- Guaranteed to converge to a local optimum [Bottou, 1996]
- ***This algorithm has enabled modern machine learning.***

Stochastic optimization

A STOCHASTIC APPROXIMATION METHOD[1]

By Herbert Robbins and Sutton Monro

University of North Carolina

1. Summary. Let $M(x)$ denote the expected value at level x of the response to a certain experiment. $M(x)$ is assumed to be a monotone function of x but is unknown to the experimenter, and it is desired to find the solution $x = \theta$ of the equation $M(x) = \alpha$, where α is a given constant. We give a method for making successive experiments at levels $x_1, x_2, \cdots$ in such a way that x_n will tend to θ in probability.

- Use noisy gradients to update

$$\nu_{t+1} = \nu_t + \rho_t \hat{\nabla}_\nu \mathcal{L}(\nu_t)$$

- Requires unbiased gradients $\mathbb{E}\left[\hat{\nabla}_\nu \mathcal{L}(\nu)\right] = \nabla_\nu \mathcal{L}(\nu)$
- Requires the step size sequence ρ_t follows Robbins-Monro conditions

The complete conditional of the global variable

- The complete conditional of the global variable is

$$p(\beta \mid \mathbf{z}, \mathbf{x}) = \text{expfam}(\beta\,;\, \eta_g(\mathbf{z}, \mathbf{x}))$$
$$\eta_g(\mathbf{z}, \mathbf{x}) = \alpha + \textstyle\sum_{i=1}^{n} t(z_i, x_i),$$

  where $t(\cdot, \cdot)$ is a function and α is the hyperparameter to the prior.
  (This is from classical theory of conjugate priors [Diaconis and Ylvisaker 1979].)

- The coordinate ascent update is

$$\lambda^* = \alpha + \textstyle\sum_{i=1}^{n} \mathbb{E}_{\phi_i}[t(Z_i, x_i)]$$

- For large datasets, this update is expensive.

Stochastic variational inference

- The **natural gradient** of the ELBO [Amari, 1998; Sato, 2001; Hoffman+ 2013] :

$$\nabla_\lambda^{\text{nat}} \mathcal{L}(\lambda) = \left(\alpha + \sum_{i=1}^n \mathbb{E}_{\phi_i^*}[t(Z_i, x_i)]\right) - \lambda.$$

- Construct a **noisy natural gradient**:

$$j \sim \text{Uniform}(1, \ldots, n)$$
$$\hat{\nabla}_\lambda^{\text{nat}} \mathcal{L}(\lambda) = \alpha + n\mathbb{E}_{\phi_j^*}[t(Z_j, x_j)] - \lambda.$$

- It is **good for stochastic optimization.**
  - Its expectation is the exact gradient (*unbiased*).
  - It only depends on optimized parameters of one data point (*cheap*).

Stochastic variational inference

Input: data $\mathbf{x}$, model $p(\beta, \mathbf{z}, \mathbf{x})$.

Stochastic variational inference

Input: data $\mathbf{x}$, model $p(\beta, \mathbf{z}, \mathbf{x})$.

Initialize λ randomly. Set ρ_t appropriately.

Stochastic variational inference

Input: data $\mathbf{x}$, model $p(\beta, \mathbf{z}, \mathbf{x})$.

Initialize λ randomly. Set ρ_t appropriately.

while *not converged* **do**

Stochastic variational inference

Input: data $\mathbf{x}$, model $p(\beta, \mathbf{z}, \mathbf{x})$.

Initialize λ randomly. Set ρ_t appropriately.

while *not converged* **do**

Sample $j \sim \text{Unif}(1, \ldots, n)$. Set local parameter

$$\phi \leftarrow \mathbb{E}_\lambda \left[\eta_\ell(\beta, x_j) \right].$$

Stochastic variational inference

Input: data $\mathbf{x}$, model $p(\beta, \mathbf{z}, \mathbf{x})$.

Initialize λ randomly. Set ρ_t appropriately.

while *not converged* **do**

Sample $j \sim \text{Unif}(1, \ldots, n)$. Set local parameter

$$\phi \leftarrow \mathbb{E}_\lambda \left[\eta_\ell(\beta, x_j) \right].$$

Set intermediate global parameter

$$\hat{\lambda} = \alpha + n\mathbb{E}_\phi[t(Z_j, x_j)].$$

Stochastic variational inference

Input: data $\mathbf{x}$, model $p(\beta, \mathbf{z}, \mathbf{x})$.

Initialize λ randomly. Set ρ_t appropriately.

while *not converged* **do**

Sample $j \sim \text{Unif}(1, \ldots, n)$. Set local parameter

$$\phi \leftarrow \mathbb{E}_\lambda \left[\eta_\ell(\beta, x_j) \right].$$

Set intermediate global parameter

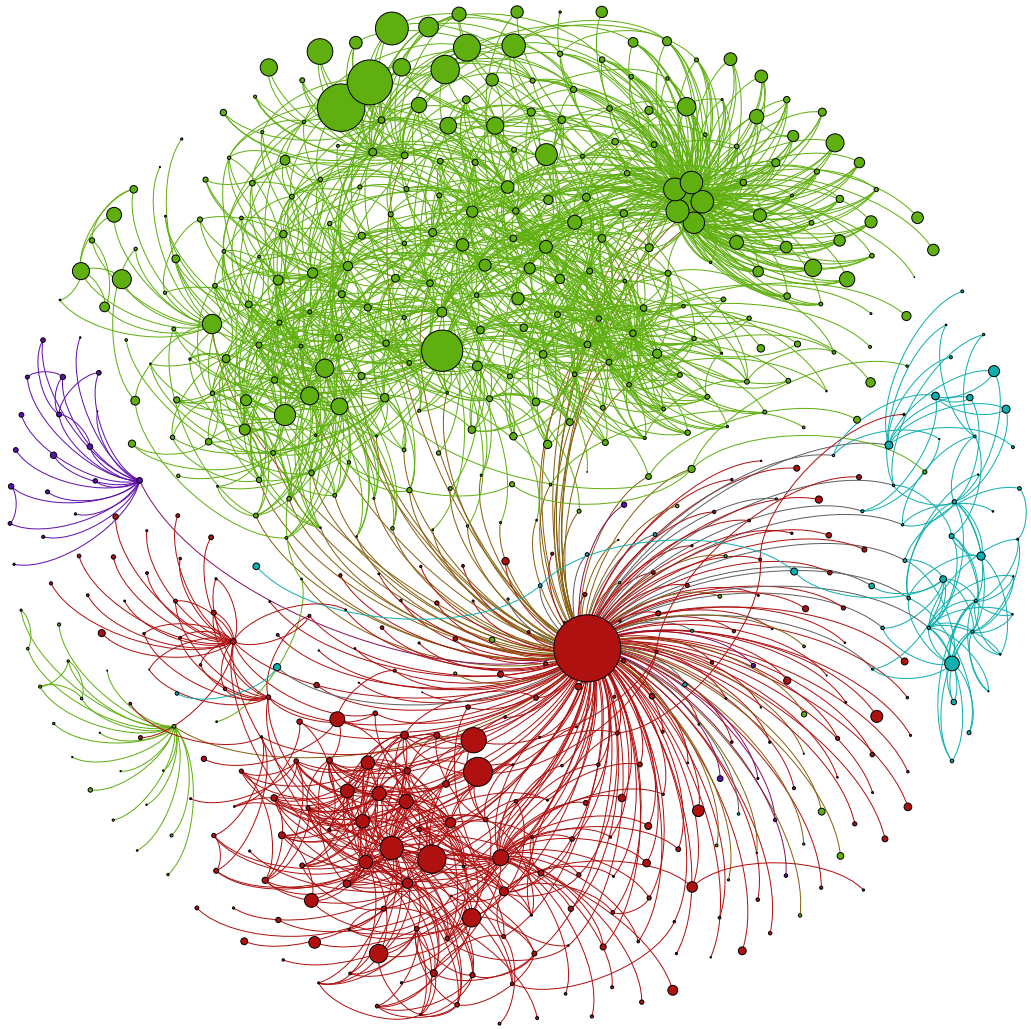
$$\hat{\lambda} = \alpha + n\mathbb{E}_\phi[t(Z_j, x_j)].$$

Set global parameter

$$\lambda = (1 - \rho_t)\,\lambda + \rho_t\,\hat{\lambda}.$$

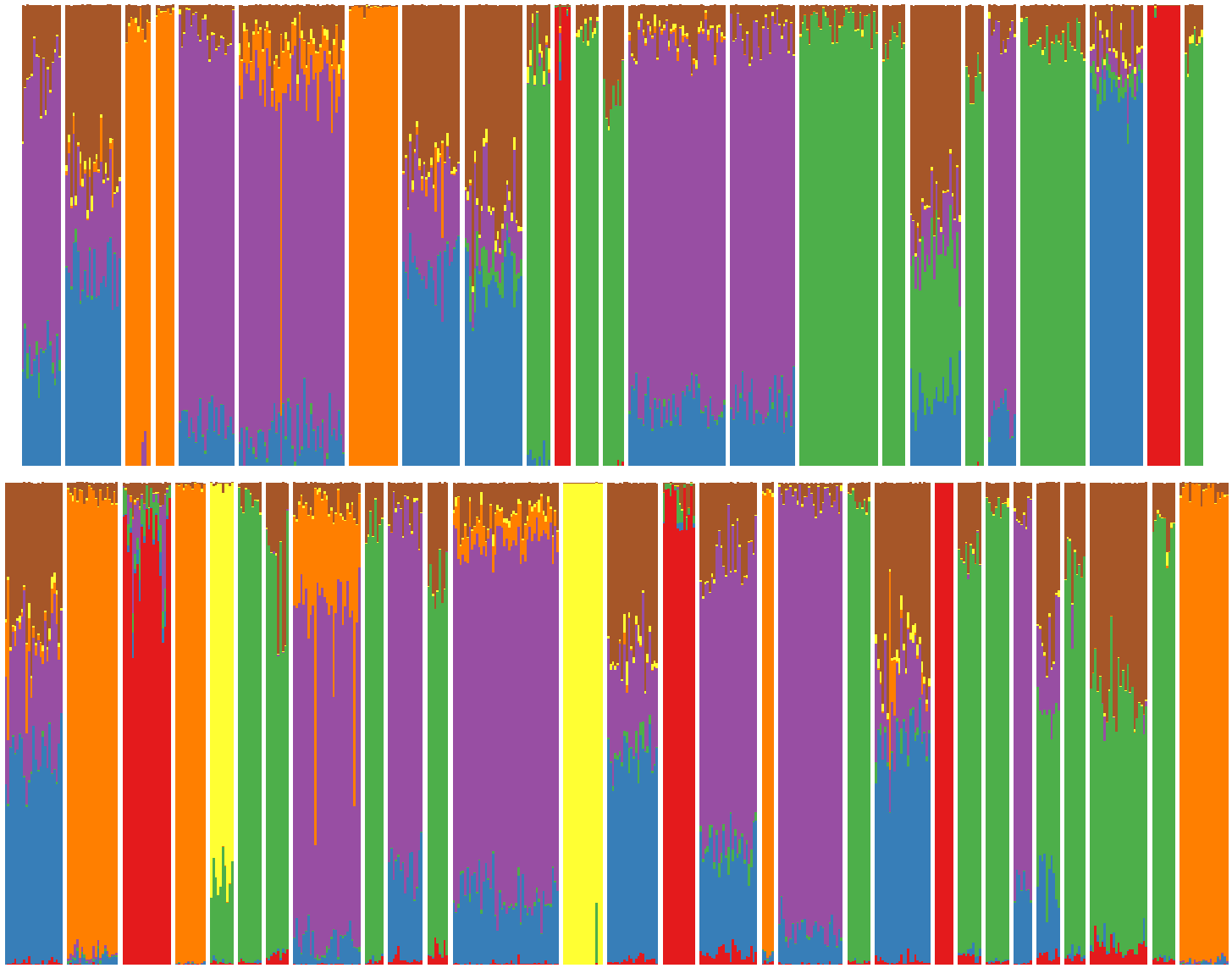
end

Topics using the HDP, found in 1.8M articles from the New York Times



Communities discovered in a 3.7M node network of U.S. Patents

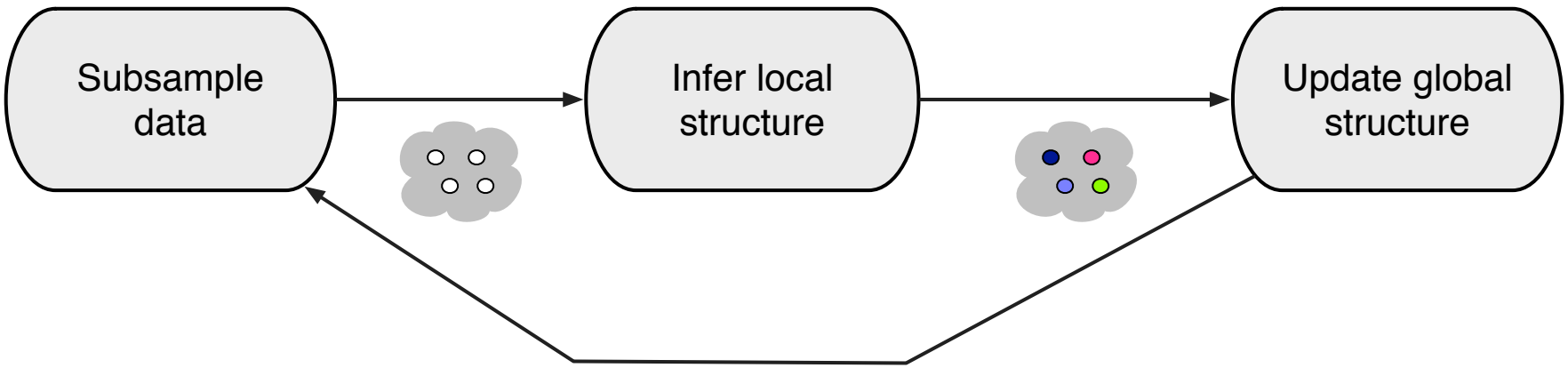
[Gopalan and Blei, PNAS 2013]



Population analysis of 2 billion genetic measurements

[Gopalan+ Nature Genetics 2016]

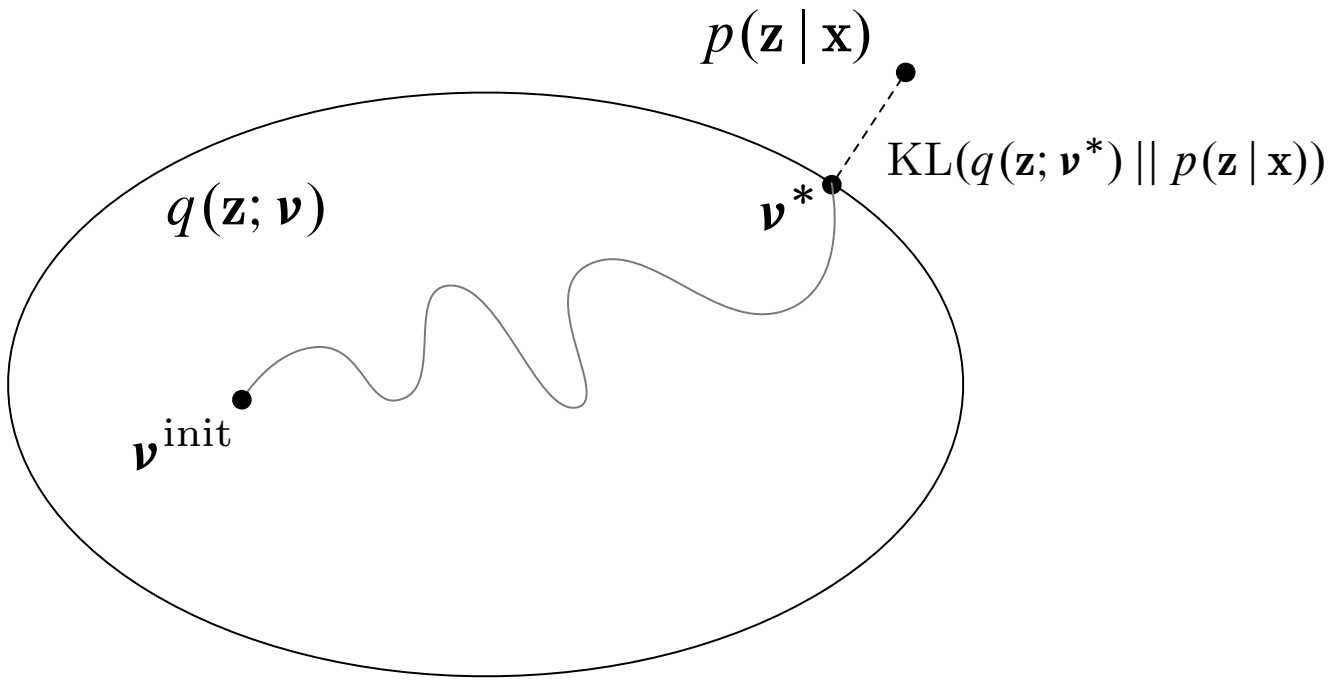
SVI scales many models



- Bayesian mixture models
- Time series models (HMMs, linear dynamic systems)
- Factorial models
- Matrix factorization (factor analysis, PCA, CCA)
- Dirichlet process mixtures, HDPs
- Multilevel regression (linear, probit, Poisson)
- Stochastic block models
- Mixed-membership models (LDA and some variants)

Black Box Variational Inference

Variational inference



- VI solves **inference** with **optimization**.
- Posit a **variational family** of distributions over the latent variables.
- Fit the **variational parameters** $\boldsymbol{\nu}$ to be close (in KL) to the exact posterior.

A.1 Computing $\mathrm{E}[\log(\theta_i \mid \alpha)]$

The need to compute the expected value of the log of a single probability component under the Dirichlet arises repeatedly in deriving the inference and parameter estimation procedures for LDA. This value can be easily computed from the natural parameterization of the exponential family representation of the Dirichlet distribution.

Recall that a distribution is in the exponential family if it can be written in the form:

$$p(x \mid \eta) = h(x) \exp\left\{\eta^T T(x) - A(\eta)\right\},$$

where η is the natural parameter, $T(x)$ is the sufficient statistic, and $A(\eta)$ is the log of the normalization factor.

We can write the Dirichlet in this form by exponentiating the log of Eq. (1):

$$p(\theta \mid \alpha) = \exp\left\{\left(\sum_{i=1}^k (\alpha_i - 1)\log\theta_i\right) + \log\Gamma\left(\sum_{i=1}^k \alpha_i\right) - \sum_{i=1}^k \log\Gamma(\alpha_i)\right\}.$$

From this form, we immediately see that the natural parameter of the Dirichlet is $\eta_i = \alpha_i - 1$ and the sufficient statistic is $T(\theta_i) = \log\theta_i$. Furthermore, using the general fact that the derivative of the log normalization factor with respect to the natural parameter is equal to the expectation of the sufficient statistic, we obtain:

$$\mathrm{E}[\log\theta_i \mid \alpha] = \Psi(\alpha_i) - \Psi\left(\sum_{j=1}^k \alpha_j\right)$$

where Ψ is the digamma function, the first derivative of the log Gamma function.

A.3.2 Variational Dirichlet

Next, we maximize Eq. (15) with respect to γ_i, the ith component of the posterior Dirichlet parameter. The terms containing γ_i are:

$$\begin{aligned} L_{[\gamma]} = &\sum_{i=1}^k (\alpha_i - 1)\left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^k \gamma_j\right)\right) + \sum_{n=1}^N \phi_{ni}\left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^k \gamma_j\right)\right) \\ &- \log\Gamma\left(\sum_{j=1}^k \gamma_j\right) + \log\Gamma(\gamma_i) - \sum_{i=1}^k (\gamma_i - 1)\left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^k \gamma_j\right)\right). \end{aligned}$$

This simplifies to:

$$L_{[\gamma]} = \sum_{i=1}^k \left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^k \gamma_j\right)\right)\left(\alpha_i + \sum_{n=1}^N \phi_{ni} - \gamma_i\right) - \log\Gamma\left(\sum_{j=1}^k \gamma_j\right) + \log\Gamma(\gamma_i).$$

We take the derivative with respect to γ_i:

$$\frac{\partial L}{\partial \gamma_i} = \Psi'(\gamma_i)\left(\alpha_i + \sum_{n=1}^N \phi_{ni} - \gamma_i\right) - \Psi'\left(\sum_{j=1}^k \gamma_j\right)\sum_{j=1}^k \left(\alpha_j + \sum_{n=1}^N \phi_{nj} - \gamma_j\right).$$

Setting this equation to zero yields a maximum at:

$$\gamma_i = \alpha_i + \sum_{n=1}^N \phi_{ni}. \tag{17}$$

Since Eq. (17) depends on the variational multinomial ϕ, full variational inference requires alternating between Eqs. (16) and (17) until the bound converges.

Finally, we expand Eq. (14) in terms of the model parameters (α, β) and the variational parameters (γ, ϕ). Each of the five lines below expands one of the five terms in the bound:

$$\begin{aligned} L(\gamma, \phi; \alpha, \beta) = &\log\Gamma\left(\sum_{j=1}^k \alpha_j\right) - \sum_{i=1}^k \log\Gamma(\alpha_i) + \sum_{i=1}^k (\alpha_i - 1)\left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^k \gamma_j\right)\right) \\ &+ \sum_{n=1}^N \sum_{i=1}^k \phi_{ni}\left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^k \gamma_j\right)\right) \\ &+ \sum_{n=1}^N \sum_{i=1}^k \sum_{j=1}^V \phi_{ni} w_n^j \log\beta_{ij} \\ &- \log\Gamma\left(\sum_{j=1}^k \gamma_j\right) + \sum_{i=1}^k \log\Gamma(\gamma_i) - \sum_{i=1}^k (\gamma_i - 1)\left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^k \gamma_j\right)\right) \\ &- \sum_{n=1}^N \sum_{i=1}^k \phi_{ni}\log\phi_{ni}, \end{aligned} \tag{15}$$

where we have made use of Eq. (8).

In the following two sections, we show how to maximize this lower bound with respect to the variational parameters ϕ and γ.

A.3.1 Variational multinomial

We first maximize Eq. (15) with respect to ϕ_{ni}, the probability that the nth word is generated by latent topic i. Observe that this is a constrained maximization since $\sum_{i=1}^k \phi_{ni} = 1$.

We form the Lagrangian by isolating the terms which contain ϕ_{ni} and adding the appropriate Lagrange multipliers. Let β_{iv} be $p(w_n^v = 1 \mid z^i = 1)$ for the appropriate v. (Recall that each w_n is a vector of size V with exactly one component equal to one; we can select the unique v such that $w_n^v = 1$):

$$L_{[\phi_{ni}]} = \phi_{ni}\left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^k \gamma_j\right)\right) + \phi_{ni}\log\beta_{iv} - \phi_{ni}\log\phi_{ni} + \lambda_n\left(\sum_{j=1}^k \phi_{ni} - 1\right),$$

where we have dropped the arguments of L for simplicity, and where the subscript ϕ_{ni} denotes that we have retained only those terms in L that are a function of ϕ_{ni}. Taking derivatives with respect to ϕ_{ni}, we obtain:

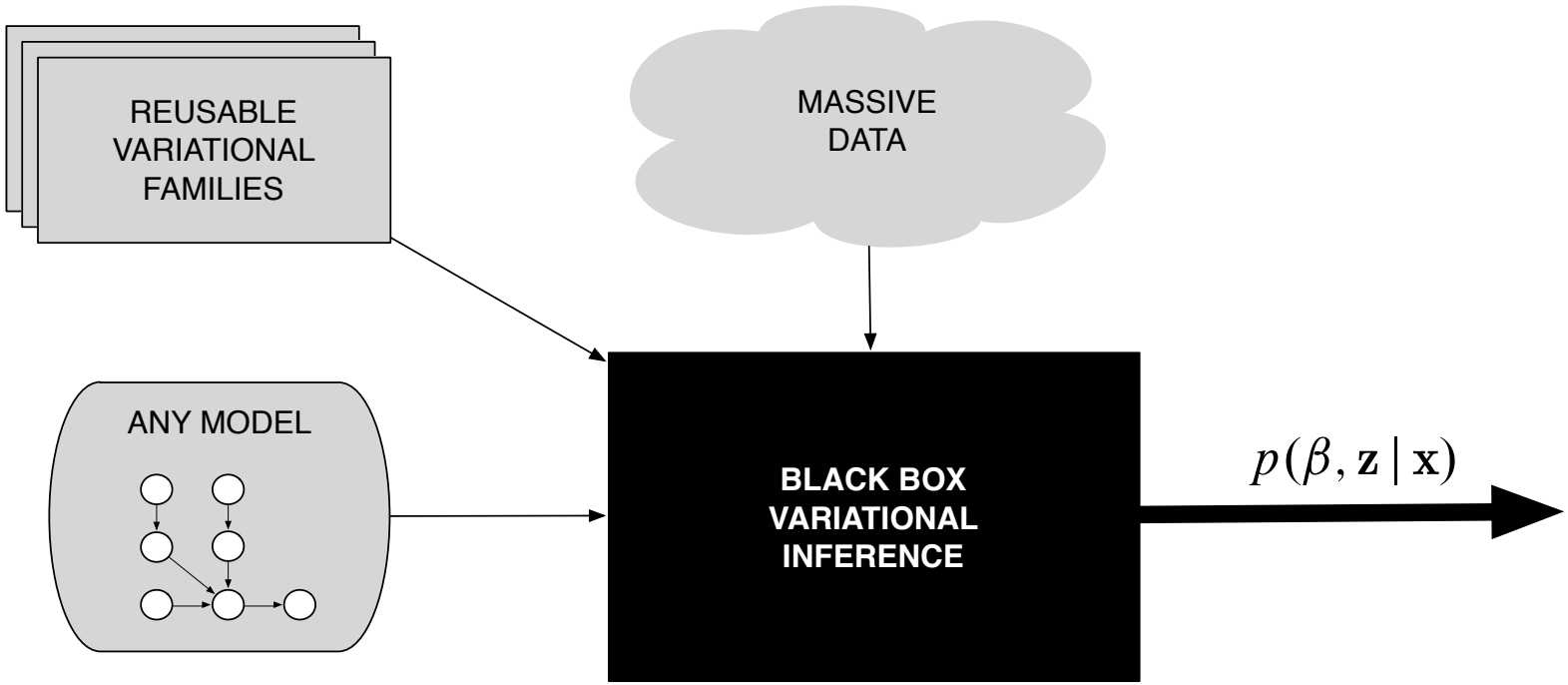
$$\frac{\partial L}{\partial \phi_{ni}} = \Psi(\gamma_i) - \Psi\left(\sum_{j=1}^k \gamma_j\right) + \log\beta_{iv} - \log\phi_{ni} - 1 + \lambda.$$

Setting this derivative to zero yields the maximizing value of the variational parameter ϕ_{ni} (cf. Eq. 6):

$$\phi_{ni} \propto \beta_{iv}\exp\left(\Psi(\gamma_i) - \Psi\left(\sum_{j=1}^k \gamma_j\right)\right). \tag{16}$$

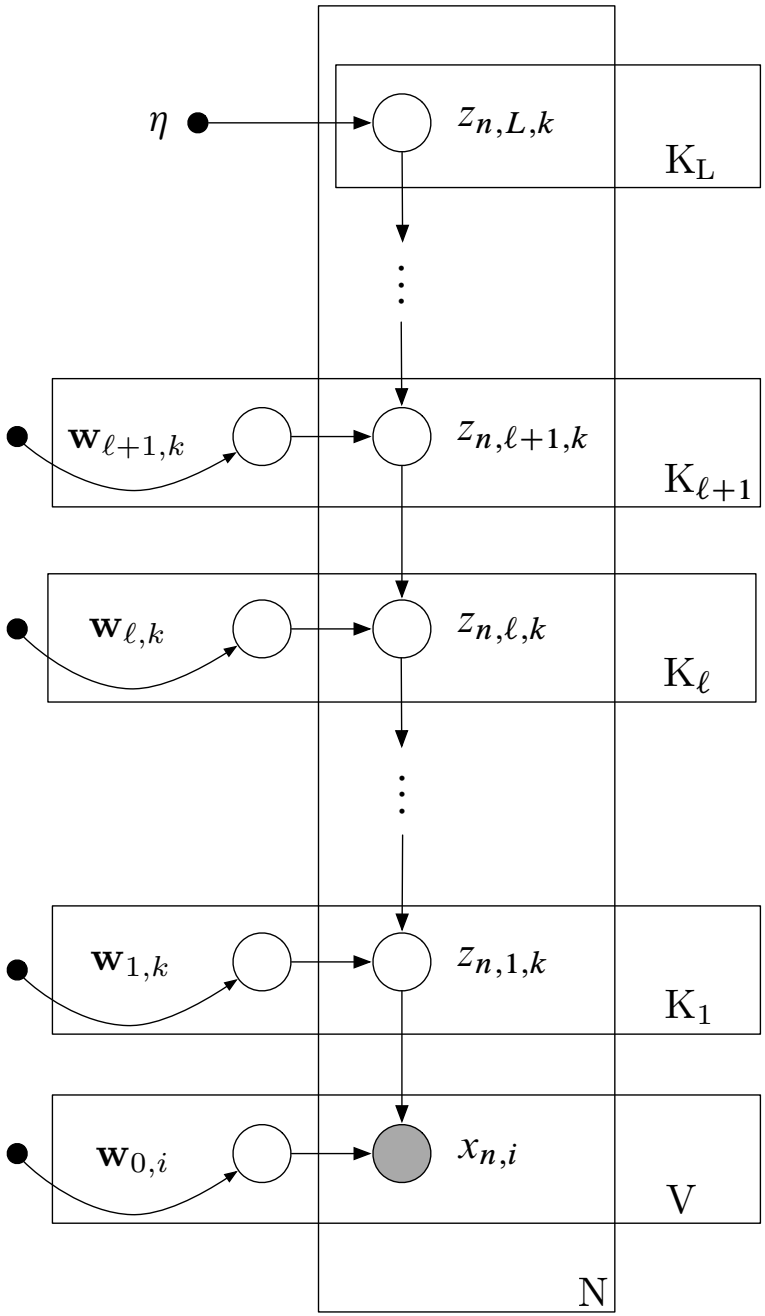
[from Blei+ 2003]

Black box variational inference



- Easily use variational inference with **any model**; no more appendices!
- Perform inference with **massive data**
- **No mathematical work** beyond specifying the model

Model: Deep exponential families

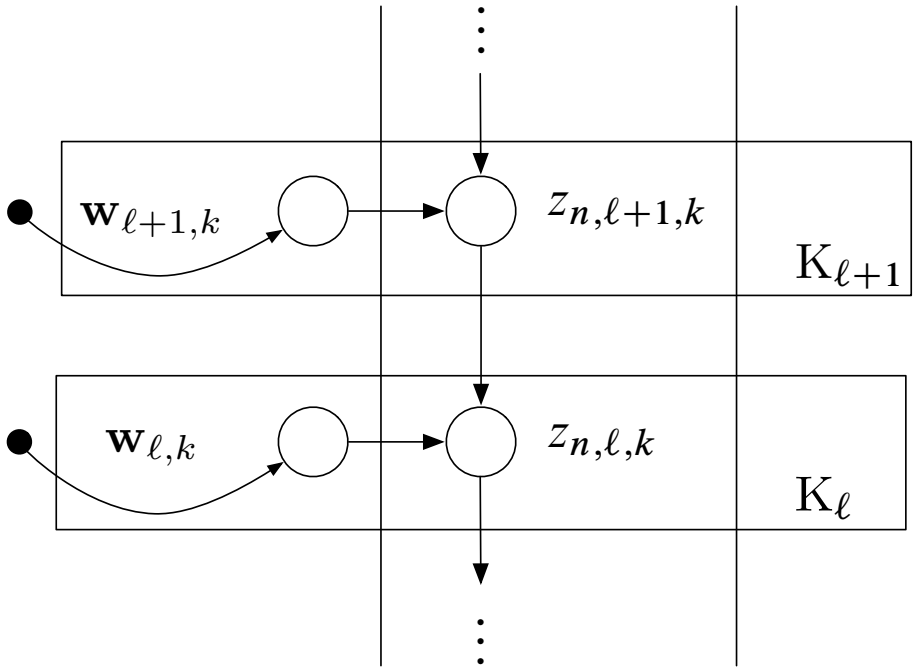


$z_{n,L,k} \sim \text{Exp-Fam}(\eta)$

$z_{n,\ell,k} \sim \text{Exp-Fam}(g(\mathbf{w}_{\ell,k}^\top \mathbf{z}_{n,\ell+1}))$

$x_{n,i} \sim \text{Exp-Fam}(g(\mathbf{w}_{0,i}^\top \mathbf{z}_{n,1}))$

Deep exponential families

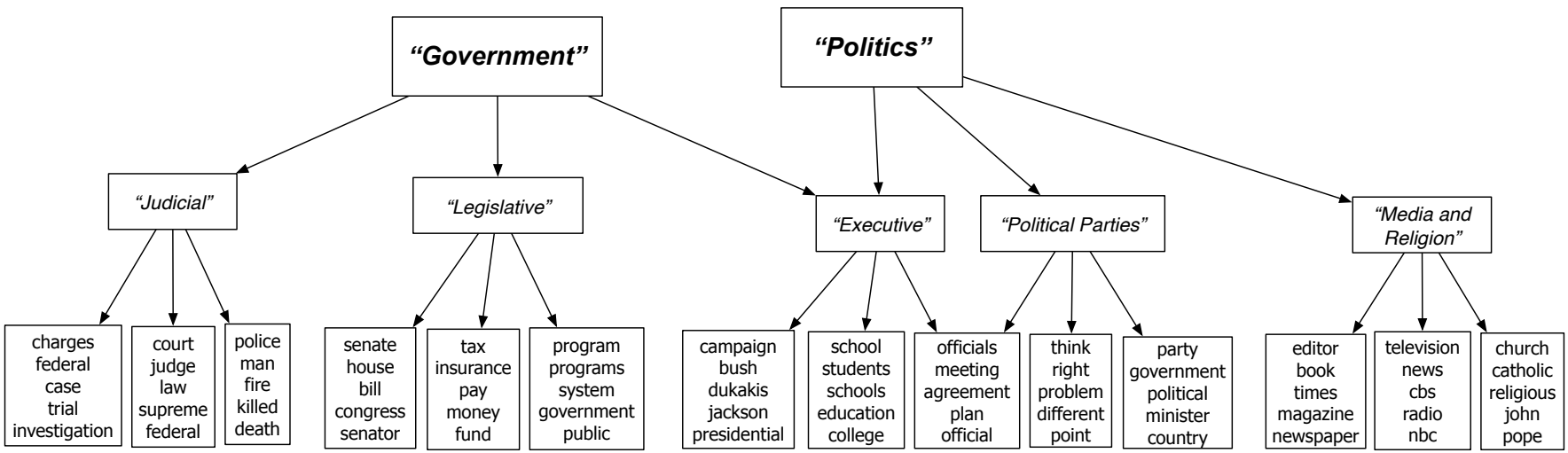


$$z_{n,\ell,k} \sim \textsc{Exp-Fam}(g(\mathbf{w}_{\ell,k}^{\top}\mathbf{z}_{n,\ell+1}))$$

Generalizes and expands probabilistic deep networks:

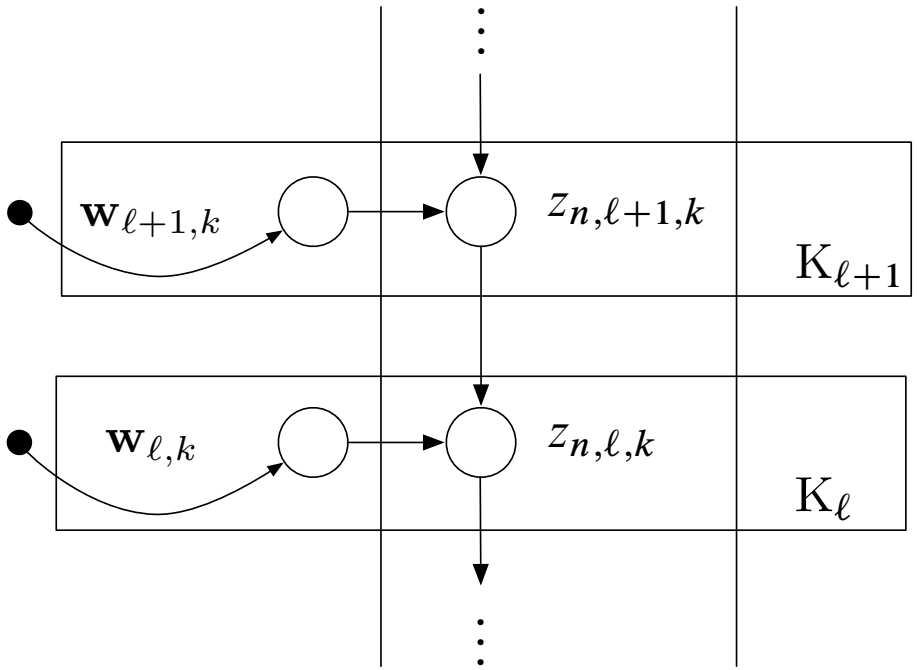
- *Bernoulli*, for binary representations [Sigmoid Belief Net, Neal 1992]
- *Gaussian*, for real representations [Deep Latent Gaussian Models, Rezende 2014]
- *Poisson*, for count representations
- *Gamma*, for positive (and sparse) representations

New York Times



- 160,000 documents; 8,500 vocabulary terms; 10M observed words
- The posterior weights provide topics and topics-of-topics.

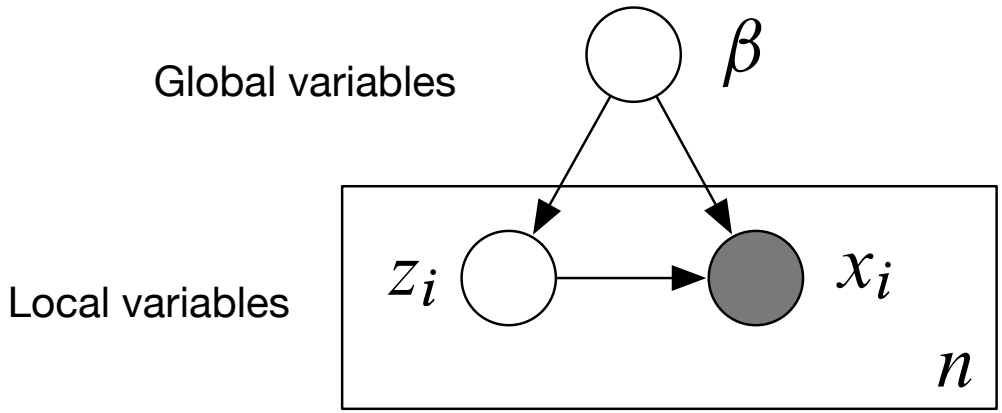
Deep exponential families



$$z_{n,\ell,k} \sim \textsc{Exp-Fam}(g(\mathbf{w}_{\ell,k}^\top \mathbf{z}_{n,\ell+1}))$$

- We want to try lots of types of DEFs. But how to do inference?
- DEFs contain cascades of latent variables.
- DEFs are ***not conditionally conjugate***.

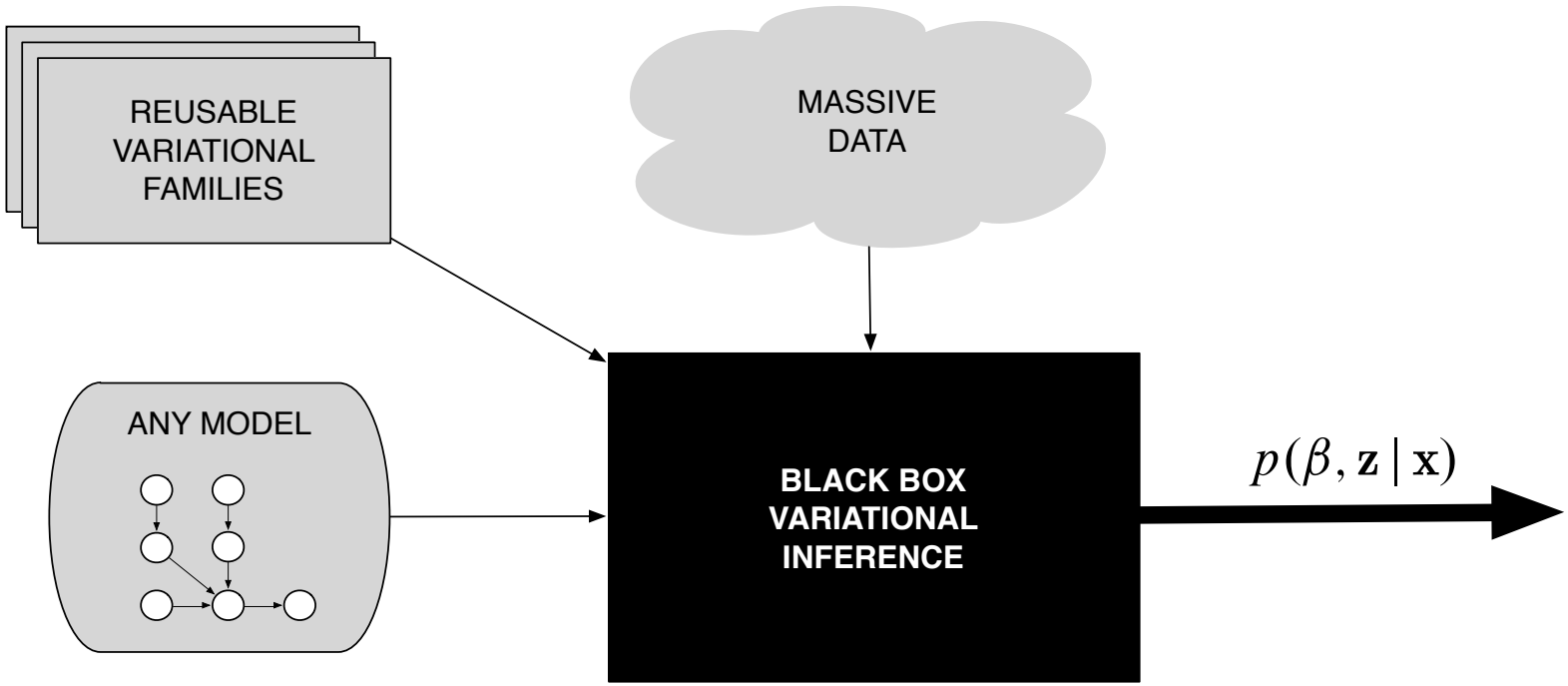
Nonconjugate models



$$p(\beta, \mathbf{z}, \mathbf{x}) = p(\beta) \prod_{i=1}^{n} p(z_i, x_i \mid \beta)$$

- Nonlinear time series models
- Deep latent Gaussian models
- Models with attention
- Generalized linear models
- Stochastic volatility models
- Discrete choice models
- Bayesian neural networks
- Deep exponential families
- Correlated topic models
- Sigmoid belief networks

Black box variational inference



- Easily use variational inference with **any model**
- Perform inference with **massive data**
- **No mathematical work** beyond specifying the model

Black box variational inference

$$\mathcal{L}(\boldsymbol{\nu}) = \underbrace{\mathbb{E}_q\left[\log p(\beta, \mathbf{z}, \mathbf{x})\right]}_{\text{Expected complete log likelihood}} - \underbrace{\mathbb{E}_q\left[\log q(\beta, \mathbf{z}; \boldsymbol{\nu})\right]}_{\text{Negative entropy}}$$

The main idea behind BBVI:

- write the **gradient of the ELBO as an expectation**
- sample from $q(\cdot)$ to form a **Monte Carlo estimate of the gradient**
- use the MC estimate in a **stochastic optimization**

Black box variational inference

$$\mathcal{L}(\boldsymbol{\nu}) = \underbrace{\mathbb{E}_q\left[\log p(\beta, \mathbf{z}, \mathbf{x})\right]}_{\text{Expected complete log likelihood}} - \underbrace{\mathbb{E}_q\left[\log q(\beta, \mathbf{z}; \boldsymbol{\nu})\right]}_{\text{Negative entropy}}$$

- Keep in mind the **black box criteria**.
- We should only need to:
  - sample from $q(\beta, \mathbf{z})$
  - evaluate things about $q(\beta, \mathbf{z})$
  - evaluate $\log p(\beta, \mathbf{z}, \mathbf{x})$
- These criteria let us perform approximate inference on many models.

Black box variational inference

$$\mathcal{L}(\boldsymbol{\nu}) = \underbrace{\mathbb{E}_q\left[\log p(\beta, \mathbf{z}, \mathbf{x})\right]}_{\text{Expected complete log likelihood}} - \underbrace{\mathbb{E}_q\left[\log q(\beta, \mathbf{z}; \boldsymbol{\nu})\right]}_{\text{Negative entropy}}$$

- Research in BBVI is about **how to write the gradient as an expectation**.
- While SVI uses stochastic optimization to overcome large datasets, BBVI uses it to overcome difficult objective functions.
- Two main strategies:
  - Score gradients (today)
  - Reparameterization gradients (used e.g., in the VAE)

The score gradient

$$\nabla_\nu \mathcal{L} = \mathbb{E}_{q(\mathbf{z};\nu)}[\underbrace{\nabla_\nu \log q(\mathbf{z};\nu)}_{\text{score function}} \underbrace{(\log p(\mathbf{x},\mathbf{z}) - \log q(\mathbf{z};\nu))}_{\text{instantaneous ELBO}}]$$

- Use the score function to write the gradient as an expectation.
  [Ji+ 2010; Paisley+ 2012; Wingate+ 2013; Ranganath+ 2014; Mnih+ 2014]
- Also called the likelihood ratio or REINFORCE gradient
  [Glynn 1990; Williams 1992]
- Pushes ν to give high probability on $\mathbf{z}$ with large instantaneous ELBO.

The score gradient

$$\nabla_\nu \mathscr{L} = \mathbb{E}_{q(\mathbf{z};\nu)}[\underbrace{\nabla_\nu \log q(\mathbf{z};\nu)}_{\text{score function}} \underbrace{(\log p(\mathbf{x},\mathbf{z}) - \log q(\mathbf{z};\nu))}_{\text{instantaneous ELBO}}]$$

Satisfies the **black box criteria** — no model-specific analysis needed.

- sample from $q(\mathbf{z};\nu)$
- evaluate $\nabla_\nu \log q(\mathbf{z};\nu)$
- evaluate $\log p(\mathbf{x},\mathbf{z})$ and $\log q(\mathbf{z})$

Black box variational inference

Input: data $\mathbf{x}$, model $p(\mathbf{z}, \mathbf{x})$.

Black box variational inference

Input: data $\mathbf{x}$, model $p(\mathbf{z}, \mathbf{x})$.

Initialize $\boldsymbol{\nu}$ randomly. Set ρ_j appropriately.

Black box variational inference

Input: data $\mathbf{x}$, model $p(\mathbf{z}, \mathbf{x})$.

Initialize $\boldsymbol{\nu}$ randomly. Set ρ_j appropriately.

while *not converged* **do**

Black box variational inference

Input: data $\mathbf{x}$, model $p(\mathbf{z}, \mathbf{x})$.

Initialize $\boldsymbol{\nu}$ randomly. Set ρ_j appropriately.

while *not converged* **do**

Take S samples from the variational distribution

$$\mathbf{z}[s] \sim q(\mathbf{z}; \boldsymbol{\nu}) \quad s = 1 \ldots S$$

Black box variational inference

Input: data $\mathbf{x}$, model $p(\mathbf{z}, \mathbf{x})$.

Initialize $\boldsymbol{\nu}$ randomly. Set ρ_j appropriately.

while *not converged* **do**

Take S samples from the variational distribution

$$\mathbf{z}[s] \sim q(\mathbf{z}; \boldsymbol{\nu}) \quad s = 1 \ldots S$$

Calculate the noisy score gradient

$$\tilde{g}_t = \frac{1}{S} \sum_{s=1}^{S} \nabla_{\boldsymbol{\nu}} \log q(\mathbf{z}[s]; \boldsymbol{\nu}_t)(\log p(\mathbf{x}, \mathbf{z}[s]) - \log q(\mathbf{z}[s]; \boldsymbol{\nu}_t))$$

Black box variational inference

Input: data $\mathbf{x}$, model $p(\mathbf{z}, \mathbf{x})$.

Initialize $\boldsymbol{\nu}$ randomly. Set ρ_j appropriately.

while *not converged* **do**

Take S samples from the variational distribution

$$\mathbf{z}[s] \sim q(\mathbf{z}; \boldsymbol{\nu}) \quad s = 1 \ldots S$$

Calculate the noisy score gradient

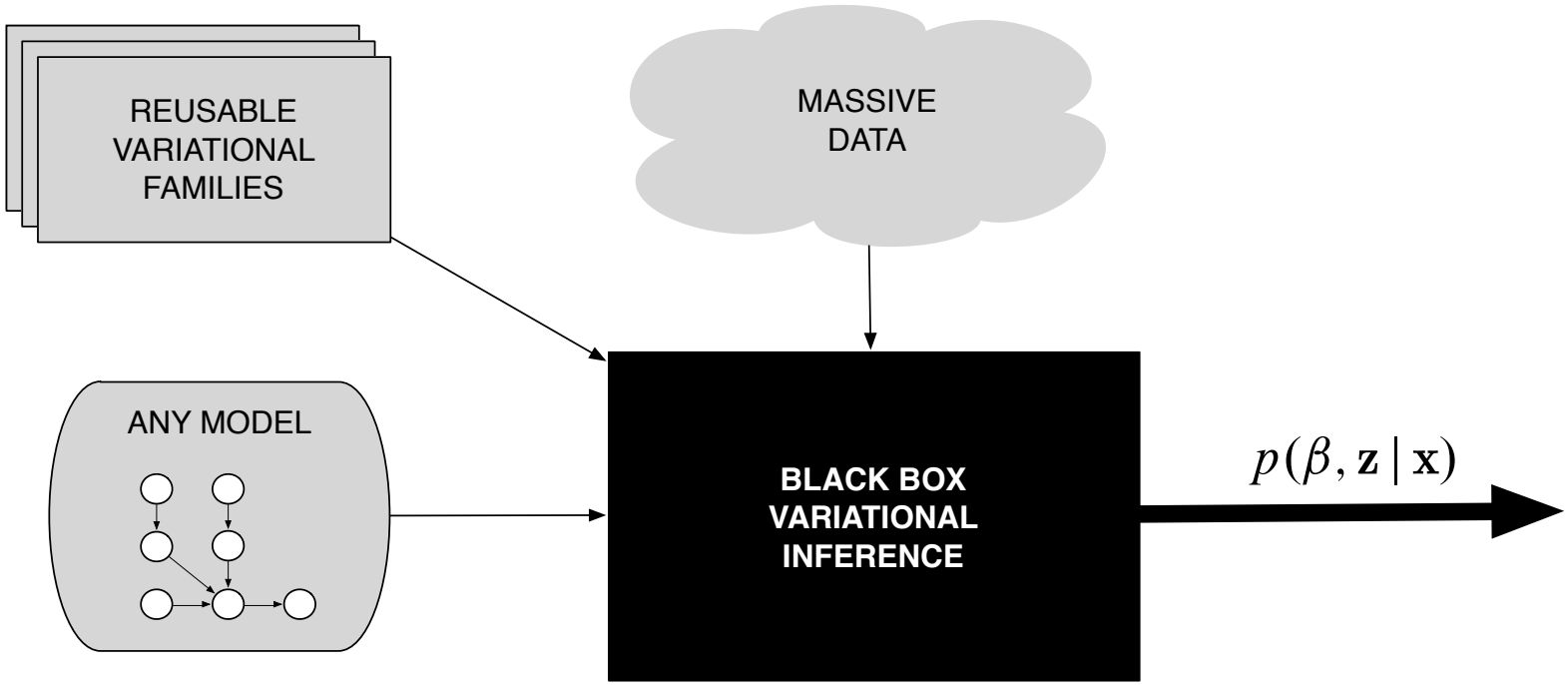
$$\tilde{g}_t = \frac{1}{S} \sum_{s=1}^{S} \nabla_{\boldsymbol{\nu}} \log q(\mathbf{z}[s]; \boldsymbol{\nu}_t)(\log p(\mathbf{x}, \mathbf{z}[s]) - \log q(\mathbf{z}[s]; \boldsymbol{\nu}_t))$$

Update the variational parameters

$$\boldsymbol{\nu}_{t+1} = \boldsymbol{\nu}_t + \rho_t \tilde{g}_t$$

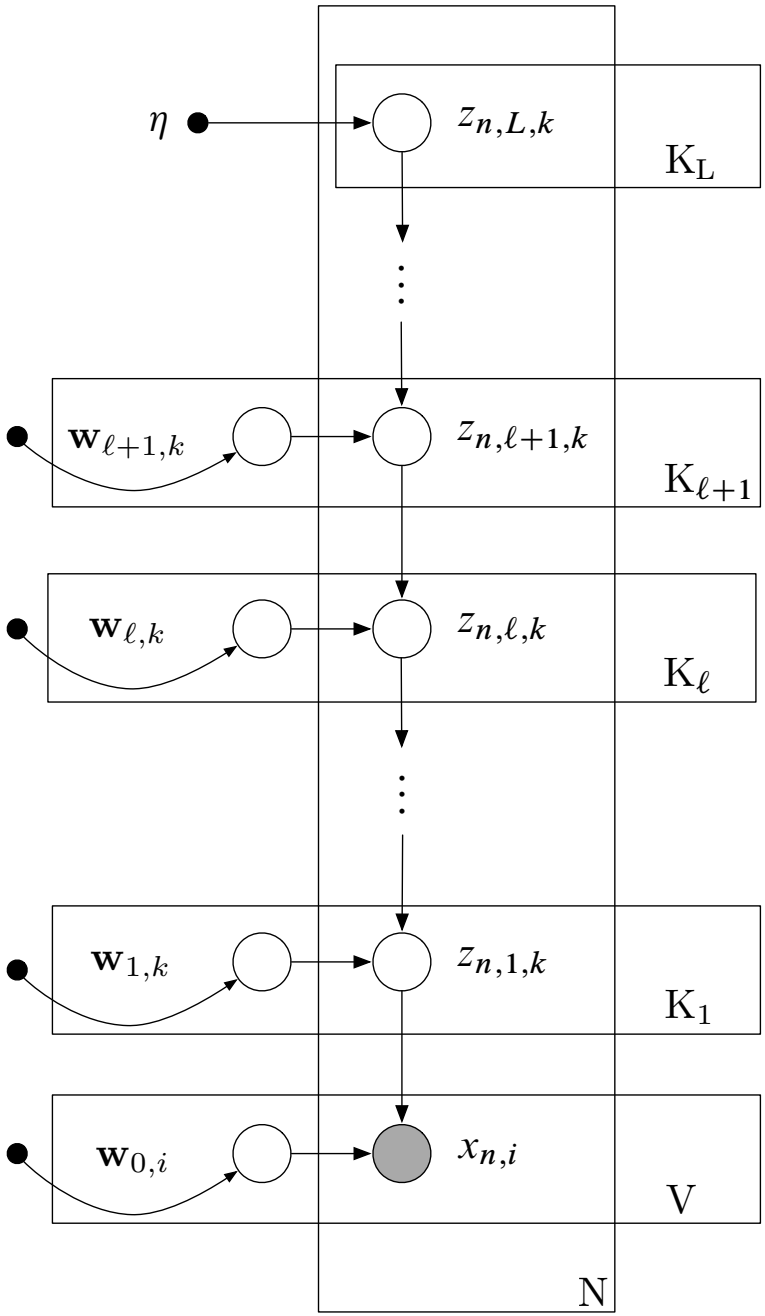
end

BBVI: Making it work



- Control the variance of the gradient [e.g., Paisley+ 2012; Ranganath+ 2014]
  - Rao-Blackwellization, control variates, importance sampling
- Adaptive step sizes [e.g., Duchi+ 2011; Kingma and Ba 2014; Kucukelbir+ 2016]
- SVI, for massive data [Hoffman+ 2013]

Deep exponential families

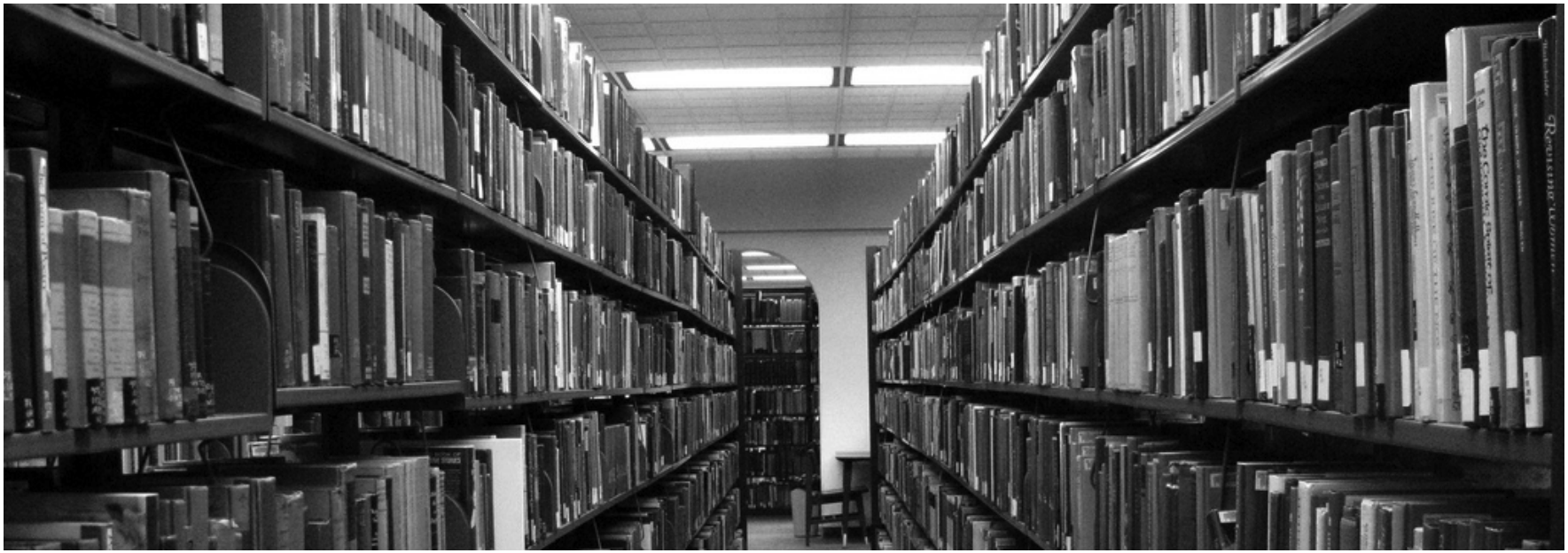


$z_{n,L,k} \sim \text{Exp-Fam}(\eta)$

$z_{n,\ell,k} \sim \text{Exp-Fam}(g(\mathbf{w}_{\ell,k}^\top \mathbf{z}_{n,\ell+1}))$

$x_{n,i} \sim \text{Exp-Fam}(g(\mathbf{w}_{0,i}^\top \mathbf{z}_{n,1}))$

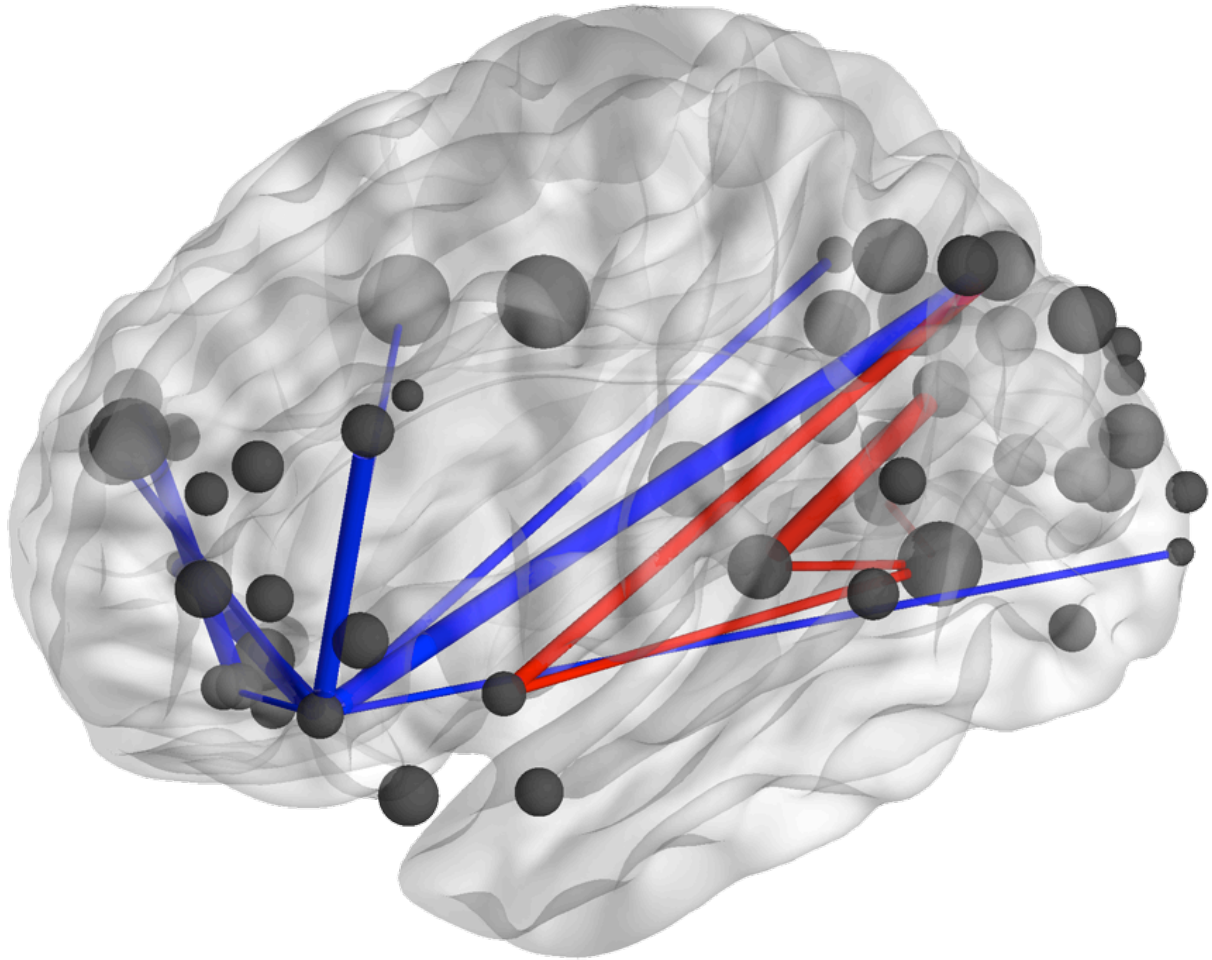
Empirical study of DEFs



- NYT and Science (about 150K documents in each, about 7K terms)
- Many models: adjusted depth, types of latents, priors, and link
- Held-out perplexity (lower is better) [Wallach+ 2009]

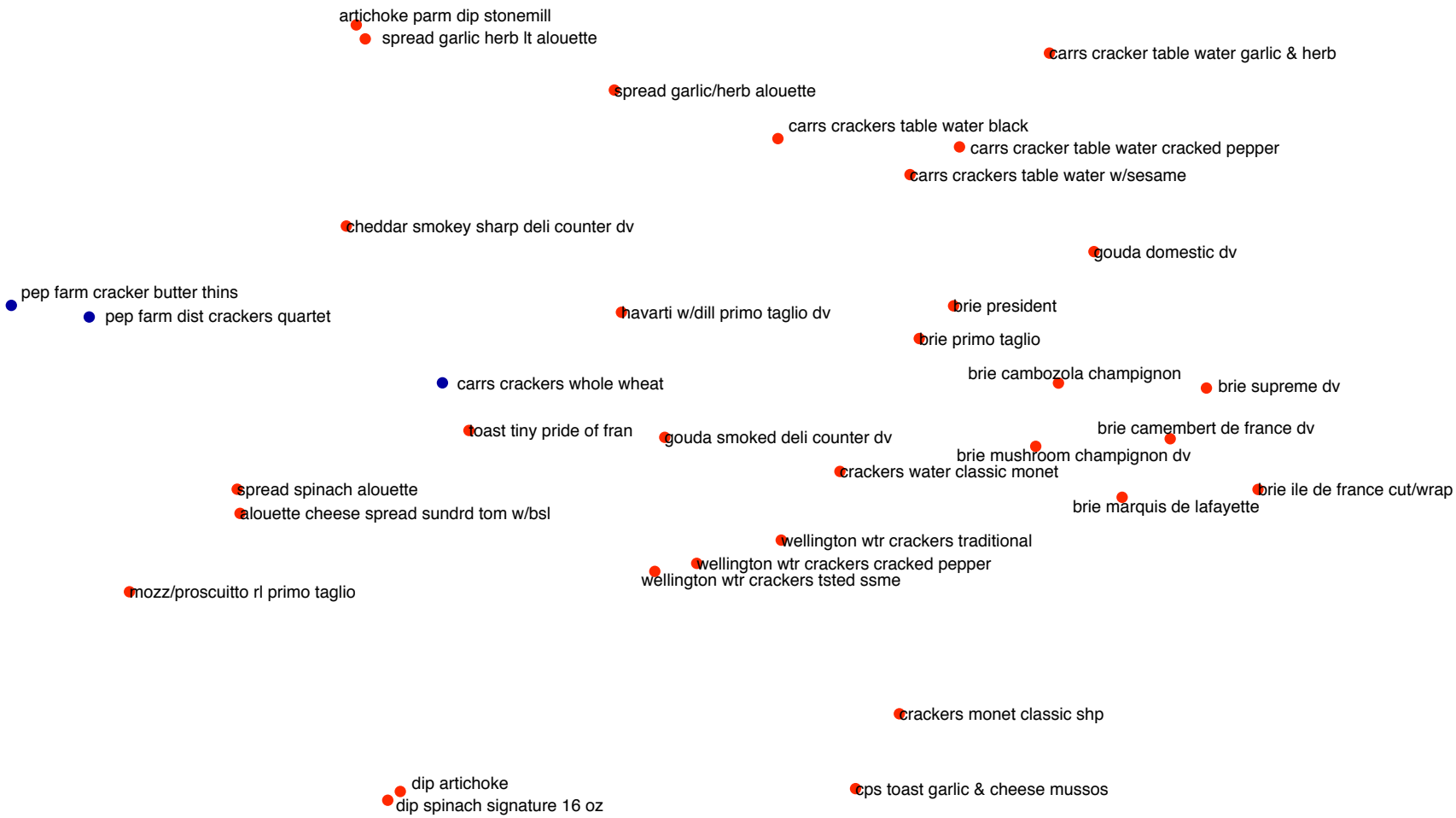
DEF evaluation

Model	$p(\mathbf{w})$	*NYT*	*Science*
LDA [Blei+ 2003]		2717	1711
DocNADE [Larochelle+ 2012]		2496	1725
Sparse Gamma 100	$\emptyset$	2525	1652
Sparse Gamma 100-30	Γ	2303	1539
Sparse Gamma 100-30-15	Γ	**2251**	1542
Sigmoid 100	$\emptyset$	2343	1633
Sigmoid 100-30	$\mathcal{N}$	2653	1665
Sigmoid 100-30-15	$\mathcal{N}$	2507	1653
Poisson 100	$\emptyset$	2590	1620
Poisson 100-30	$\mathcal{N}$	2423	1560
Poisson 100-30-15	$\mathcal{N}$	2416	1576
Poisson log-link 100-30	Γ	2288	**1523**
Poisson log-link 100-30-15	Γ	2366	1545



Neuroscience analysis of 220 million fMRI measurements

[Manning+ 2014]



Shopper on 5.7M purchases.

[Ruiz+ 2017]

Analysis of 1.7M taxi trajectories, in Stan

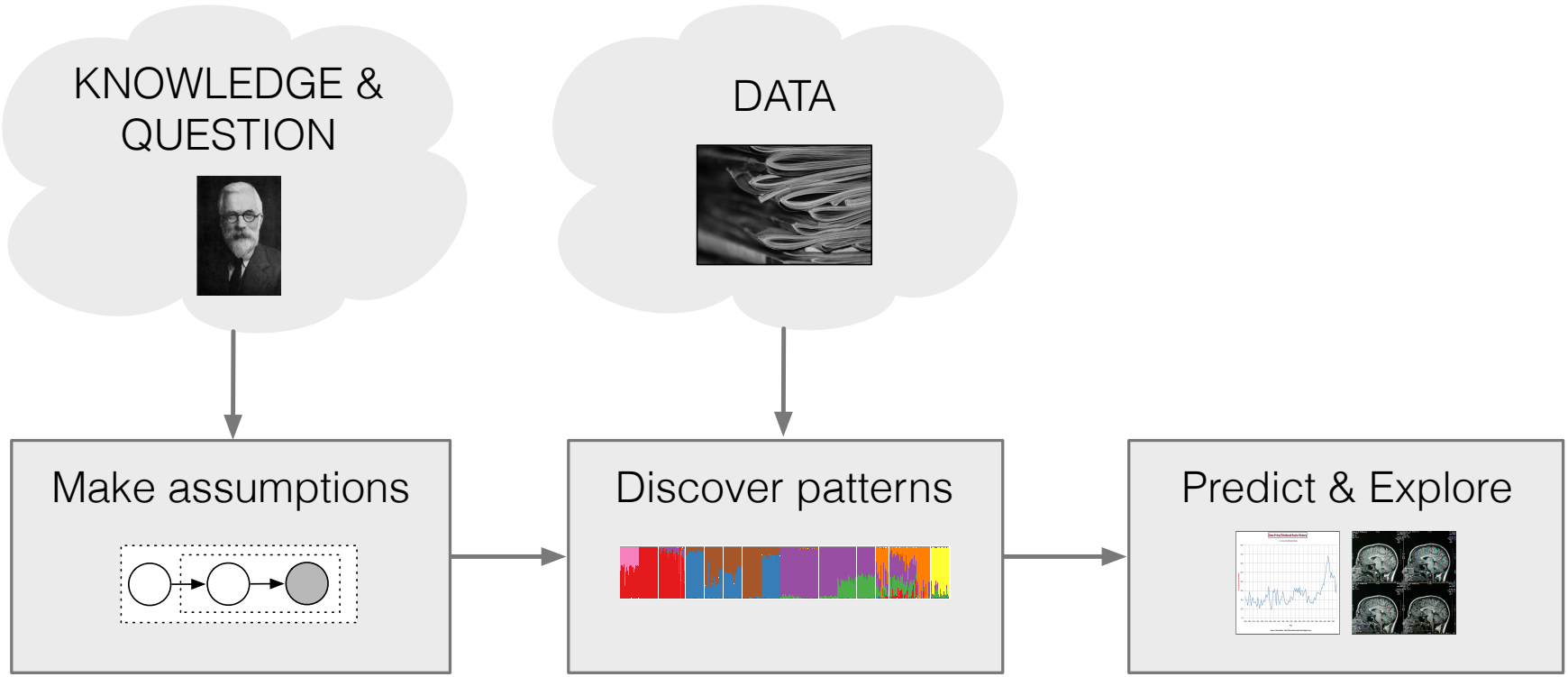
[Kucukelbir+ 2017]

A Tour of Variational Inference (with one picture)

PROBABILISTIC MACHINE LEARNING

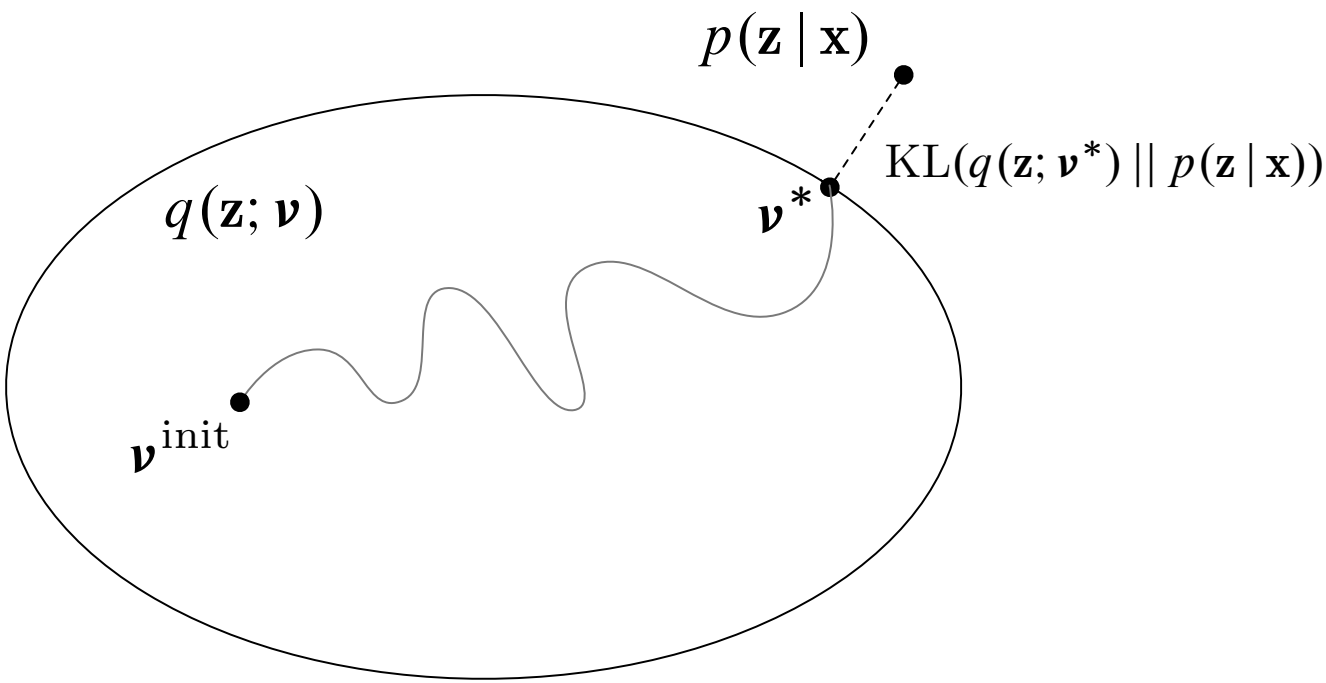
- ML methods that ***connect domain knowledge to data***.
- Provides a computational methodology for analyzing data
- Goal: A methodology that is ***expressive, scalable, easy to develop***

The probabilistic pipeline



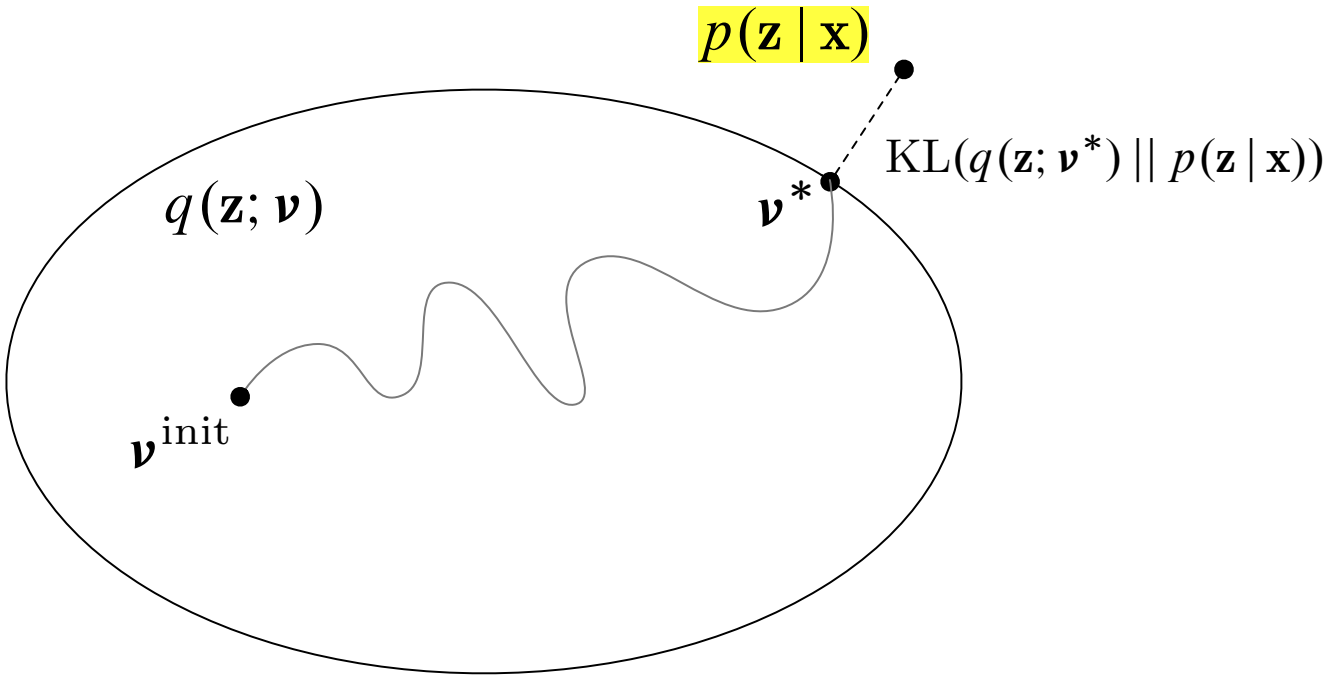
- **Posterior inference** is the key algorithmic problem.
- Answers the question: What does this model say about this data?
- VI provides **scalable** and **general** approaches to posterior inference

Stochastic optimization makes VI better



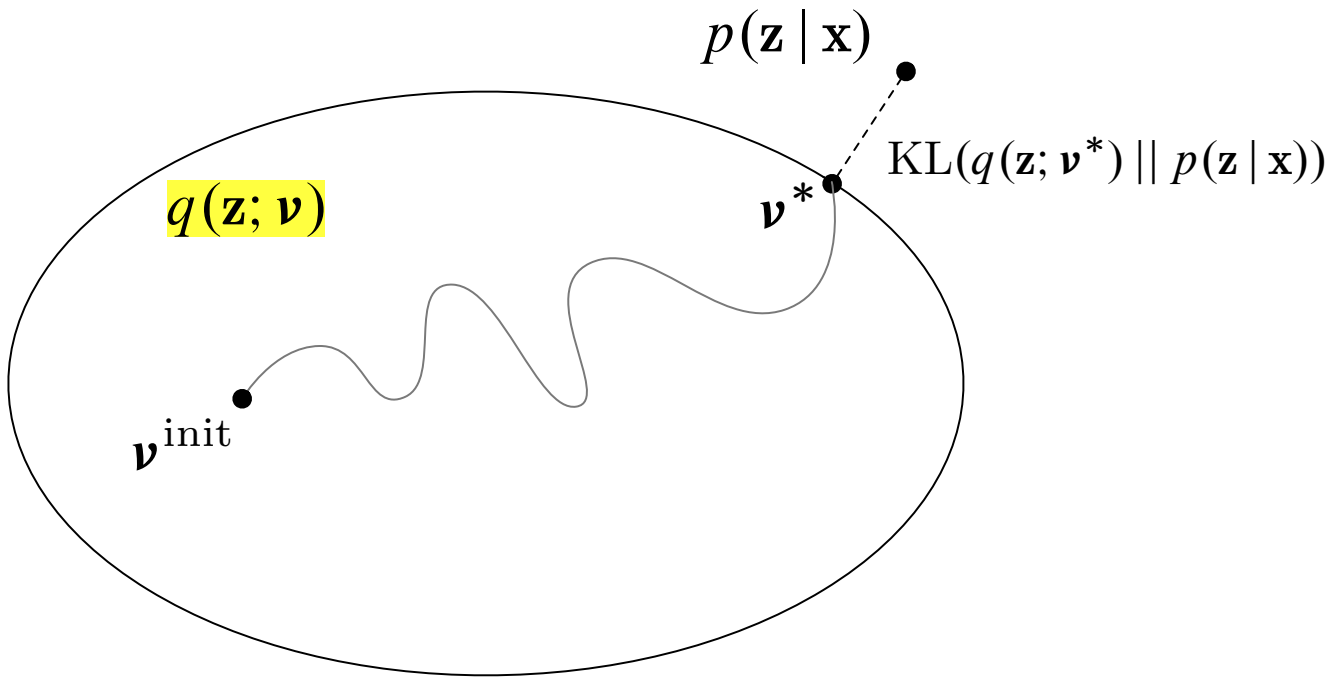
- **Stochastic VI** scales up VI to massive data.
- **Black box VI** generalizes VI to a wide class of models.

What classes of models can VI handle?



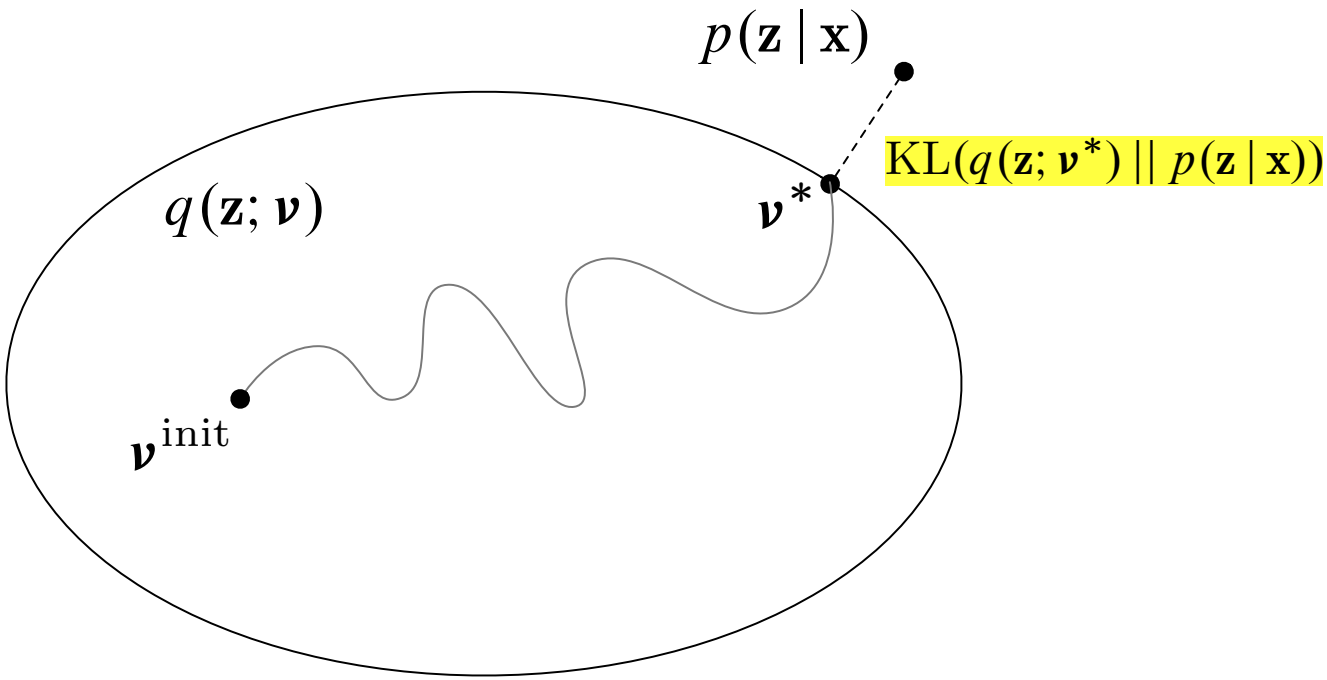
- Conditionally conjugate [Gharamani and Beal 2001; Hoffman+ 2013]
- Not ↑, but can differentiate the log likelihood [Kucukelbir+ 2015]
- Not ↑, but can calculate the log likelihood [Ranganath+ 2014]
- Not ↑, but can sample from the model [Ranganath+ 2017]

How can we expand the variational family?



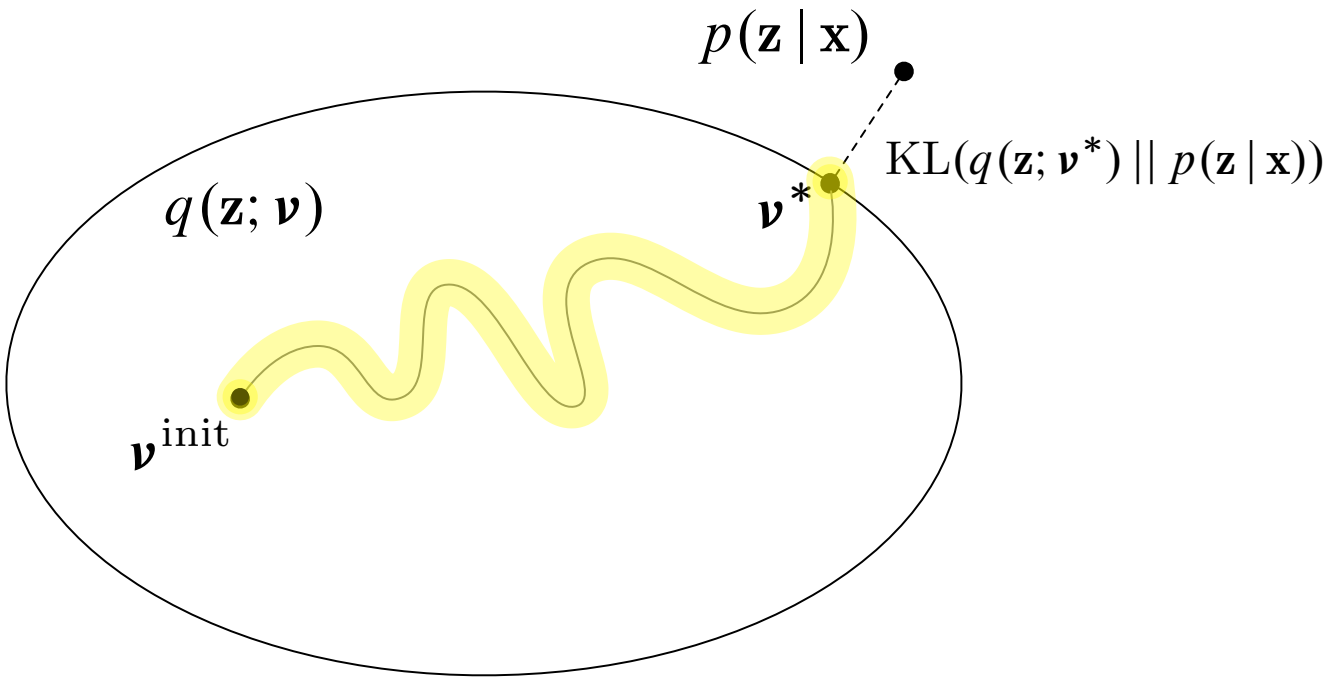
- Structured variational inference [Saul and Jordan 1996; Hoffman and Blei 2015]
- Variational models [Lawrence 2001; Ranganath+ 2015; Tran+ 2015]
- Amortized inference [Kingma and Welling 2014; Rezende+ 2014]
- Sequential Monte Carlo [Naesseth+ 2018; Maddison+ 2017; Le+ 2017]

Which distance should we use? How good is it?



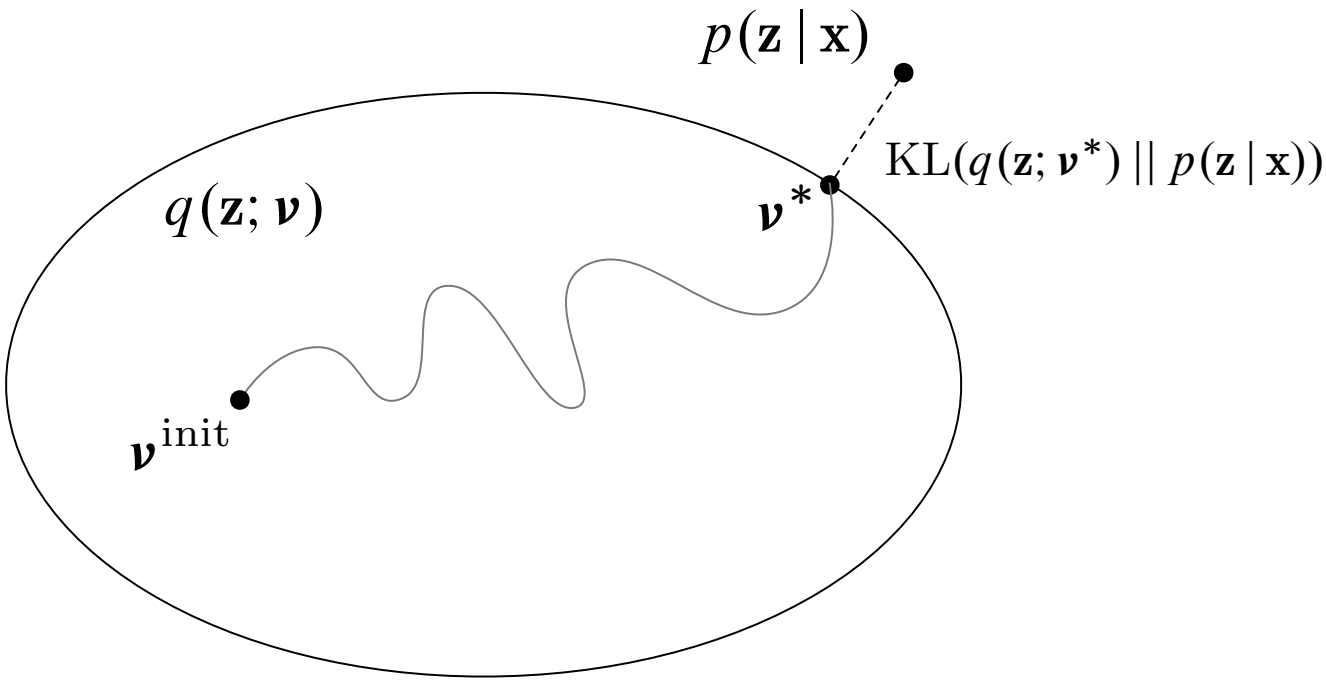
- Expectation propagation and inclusive KL [Minka 2001]
- Belief propagation [Yedidia 2001]
- Operator variational inference [Ranganath+ 2016]
- χ-variational inference [Dieng+ 2017]

Can we make the algorithm better?



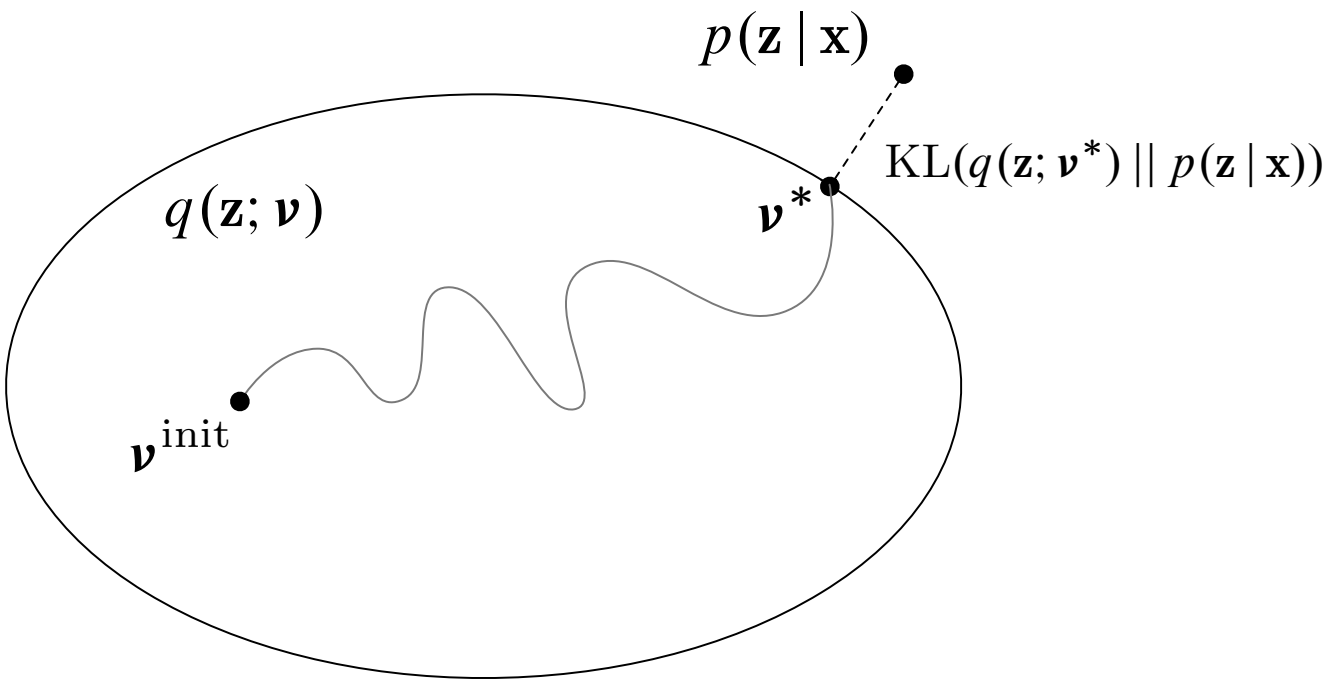
- SVI and structured SVI [Hoffman+ 2013; Hoffman and Blei 2015]
- Proximity VI [Altosaar+ 2018]
- SGD as VI [Mandt+ 2017]
- Adaptive rates, averaged and biased gradients, etc. [Many papers]

What is guaranteed about VI?



- Asymptotic normality of Gaussian approximations [Hall+ 2011]
- Risk bounds for VI [Pati+ 2017]
- Bernstein Von-Mises, model misspecification [Wang and Blei 2019, 2020]
- Convergence rates for VI [Alquier+ 2016, Zhang and Gao 2019]

How can we use VI in practice?



- Correct for VI's underestimates of the posterior variance [Giordano+ 2015]
- Software for VI [Minka 2014, Kucukelbir+ 2016]
- Best practices for finding good local optima
- How to check variational inferences

References (from our group)

- D. Blei, A. Kucukelbir, J. McAuliffe. **Variational inference: A review for statisticians**. Journal of American Statistical Association, 2017.

- M. Hoffman, D. Blei, C. Wang, J. Paisley. **Stochastic variational inference**. Journal of Machine Learning Research, 2013.

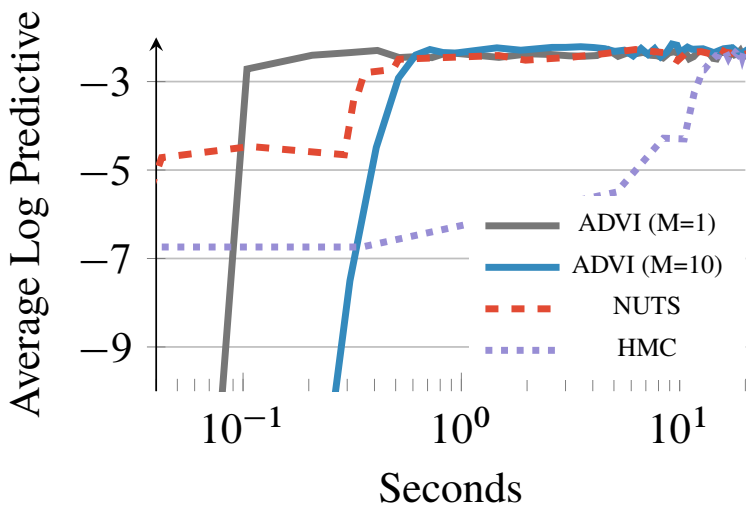
- R. Ranganath, S. Gerrish, D. Blei. **Black box variational inference**. Artificial Intelligence and Statistics, 2014.

- A. Kucukelbir, D. Tran, R. Ranganath, A. Gelman, D. Blei. **Automatic differentiation variational inference**. Journal of Machine Learning Research, 2017.

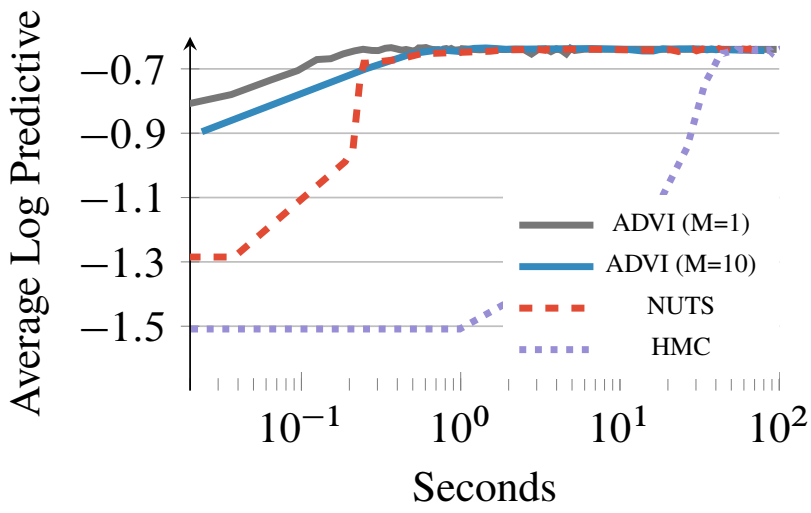
- Y. Wang and D. Blei. **Frequentist consistency of variational Bayes.** Journal of the American Statistical Association, 2019.

Extra slides

Should I be skeptical about variational inference?



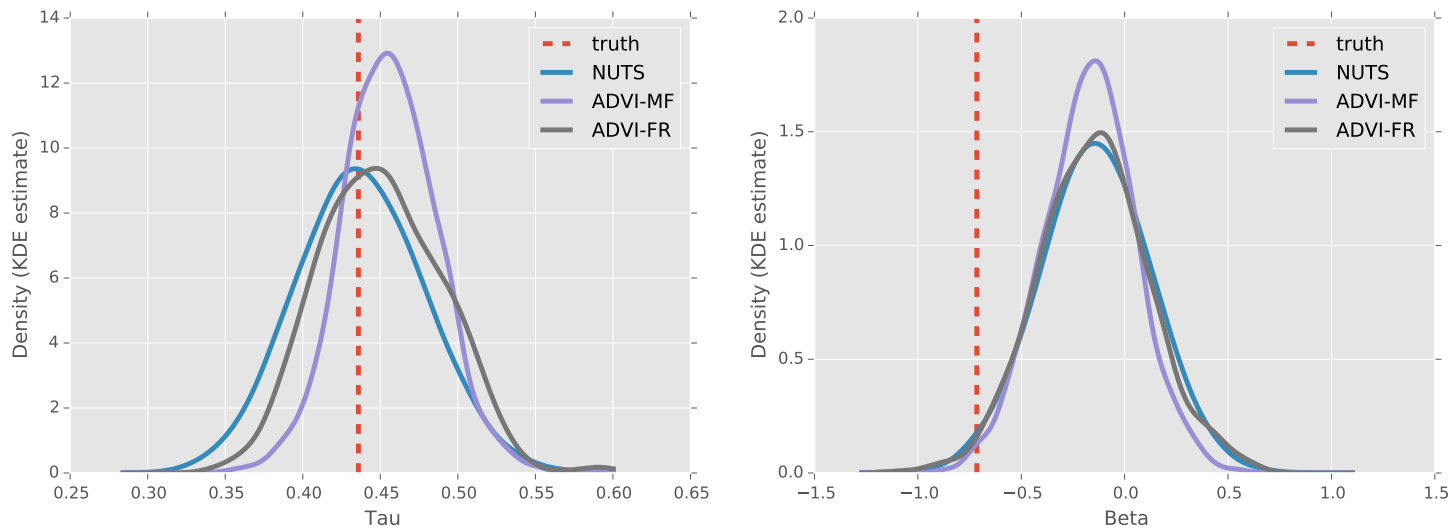
(a) Linear Regression with ARD



(b) Hierarchical Logistic Regression

- **MCMC enjoys theoretical guarantees.**
- But they usually get to the same place. [Kucukelbir+ 2016]
- We need more theory about variational inference ([E.g., Wang and Blei 2018]).

Should I be skeptical about variational inference?



- **Variational inference underestimates the variance of the posterior.**
- Relaxing the mean-field assumption can help.
- Here: A Poisson GLM [Giordano+ 2015]

Aside: The exponential family

$$p(x) = h(x) \exp\{\eta^\top t(x) - a(\eta)\}$$

Terminology:

η	the natural parameter
$t(x)$	the sufficient statistics
$a(\eta)$	the log normalizer
$h(x)$	the base density

Aside: The exponential family

$$p(x) = h(x) \exp\{\eta^\top t(x) - a(\eta)\}$$

- The log normalizer is

$$a(\eta) = \log \int \exp\{\eta^\top t(x)\} dx$$

- It ensures the density integrates to one.
- Its gradient calculates the expected sufficient statistics

$$\mathbb{E}[t(X)] = \nabla_\eta a(\eta).$$

Aside: The exponential family

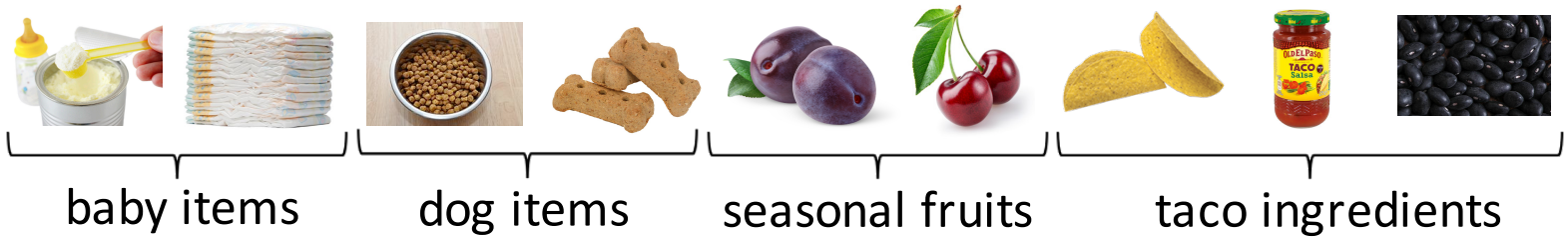
$$p(x) = h(x) \exp\{\eta^\top t(x) - a(\eta)\}$$

- Many common distributions are in the exponential family
  - Bernoulli, categorical, Gaussian, Poisson, Beta, Dirichlet, Gamma, etc.
- Outlines the theory around conjugate priors and corresponding posteriors
- Connects closely to variational inference [Wainwright and Jordan, 2008]

Model: Shopper

- Economists want to understand how people shop
- **Shopper** is a Bayesian model of consumer behavior [Ruiz+ 2017] .
- Use it to understand patterns of purchasing behavior and estimate the effects of interventions (e.g., on price)

Shopper



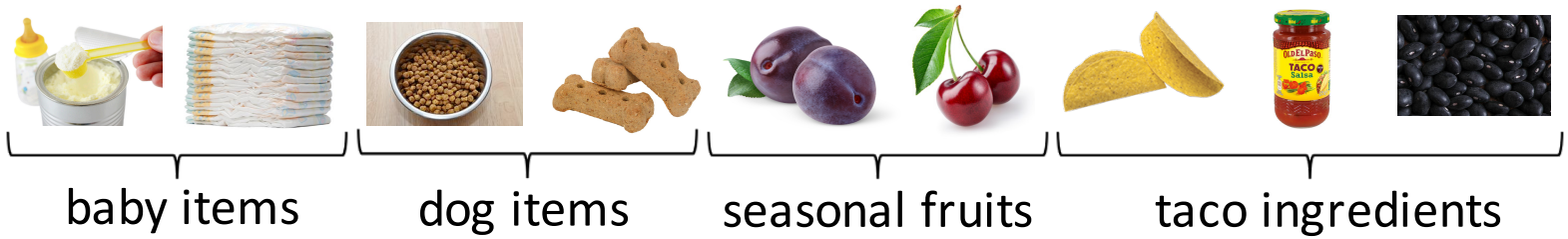
- Each customer walks into the store and sequentially chooses items, each time maximizing utility. This leads to a joint:

$$p(\mathbf{y}_t) = p(y_{t1})p(y_{t2}\,|\,y_{t1})\cdots p\left(y_{tn}\,|\,\mathbf{y}_t^{[n-1]}\right).$$

- The customer picks each item conditional on features of the other items. These features capture that, e.g.,
  - taco shells and beans go well together
  - a customer doesn't need to buy four different types of salsa
  - people who buy dog food also usually buy dog treats

- But these features are latent!

Shopper



- The conditional probability of picking item c is a log linear model

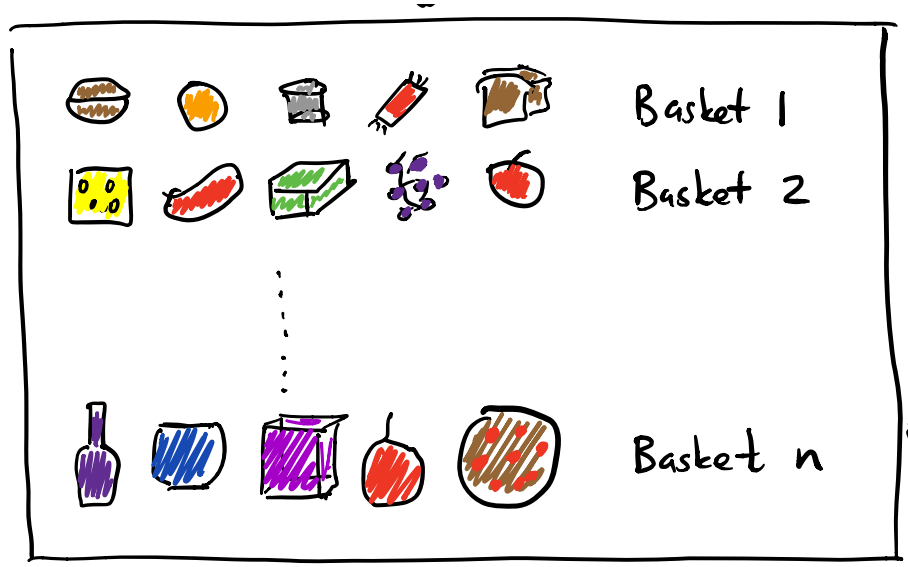
$$p(y_{ti} = c \,|\, \text{previously selected items}) \propto \exp\{\Psi_{tc}\}.$$

- The parameter is

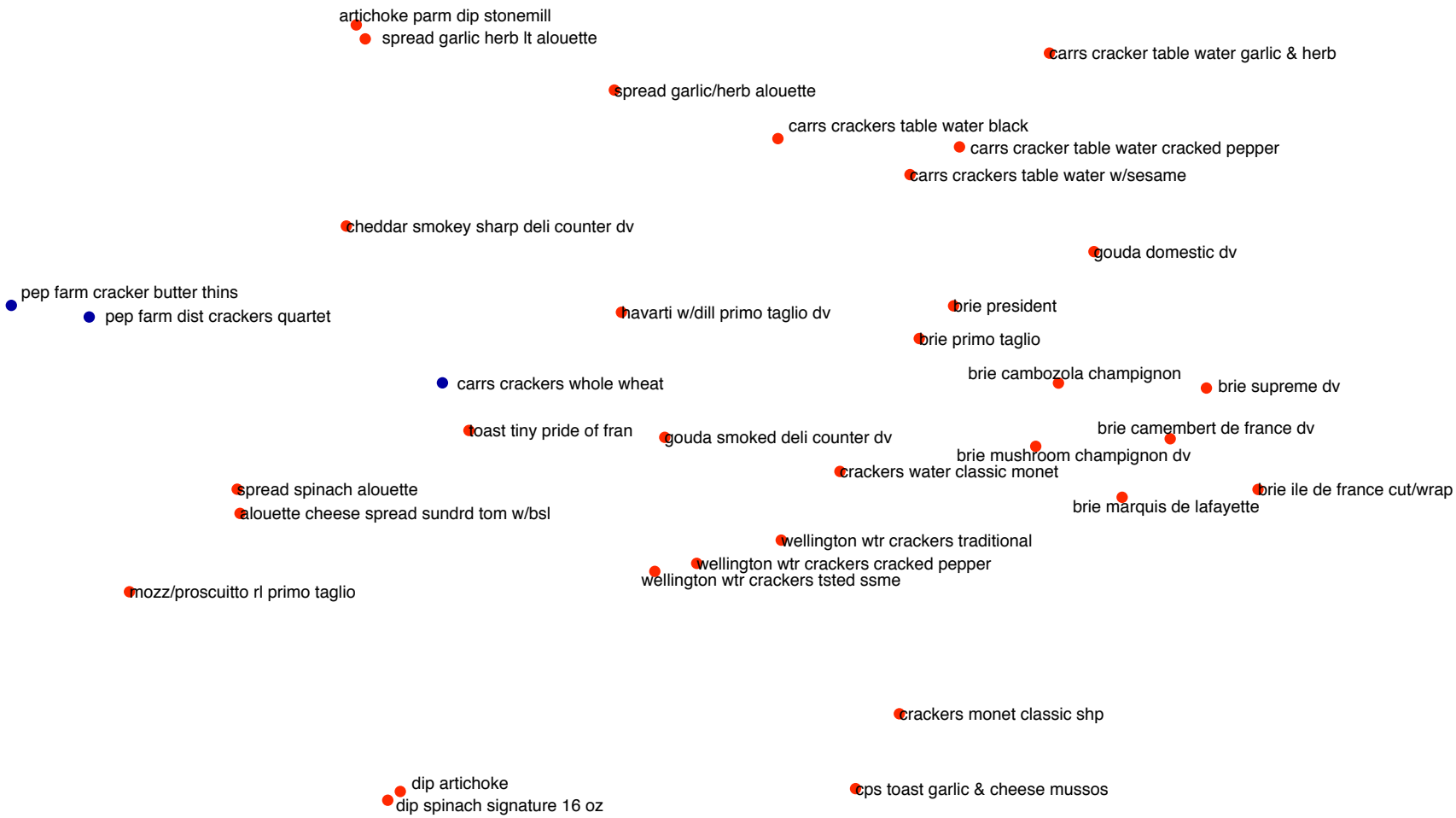
$$\Psi_{tc} = \rho_c^\top \left(\sum_{j=1}^{i-1} \alpha_{y_{tj}} \right)$$

- This is an *embedding method* [Bengio+ 2003, Rudolph+ 2016] .
  - α_{taco} : (latent) attributes of taco shells
  - ρ_{salsa} : attributes that go well with salsa

The Shopper posterior



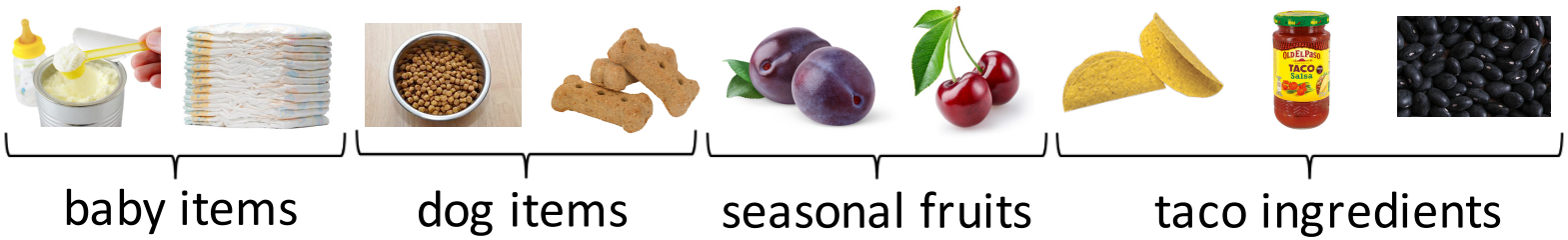
- From a dataset of shopping trips, infer the posterior $p(\boldsymbol{\alpha}, \boldsymbol{\rho} \mid \mathbf{x})$.
- Posterior of per-item attributes and per-item interaction coefficients
- 3,200 customers; 5,600 items; 570K trips; 5.7M purchased items



Shopper on 5.7M purchases.

[Ruiz+ 2017]

Shopper



- Shopper is not conditionally conjugate
- But we can evaluate $\log p(\boldsymbol{\alpha}, \boldsymbol{\rho}, \mathbf{x})$ and its gradient $\nabla_{\boldsymbol{\alpha},\boldsymbol{\rho}} \log p(\boldsymbol{\alpha}, \boldsymbol{\rho}, \mathbf{x})$.
- We can use BBVI with **the reparameterization gradient.**

Differentiable models

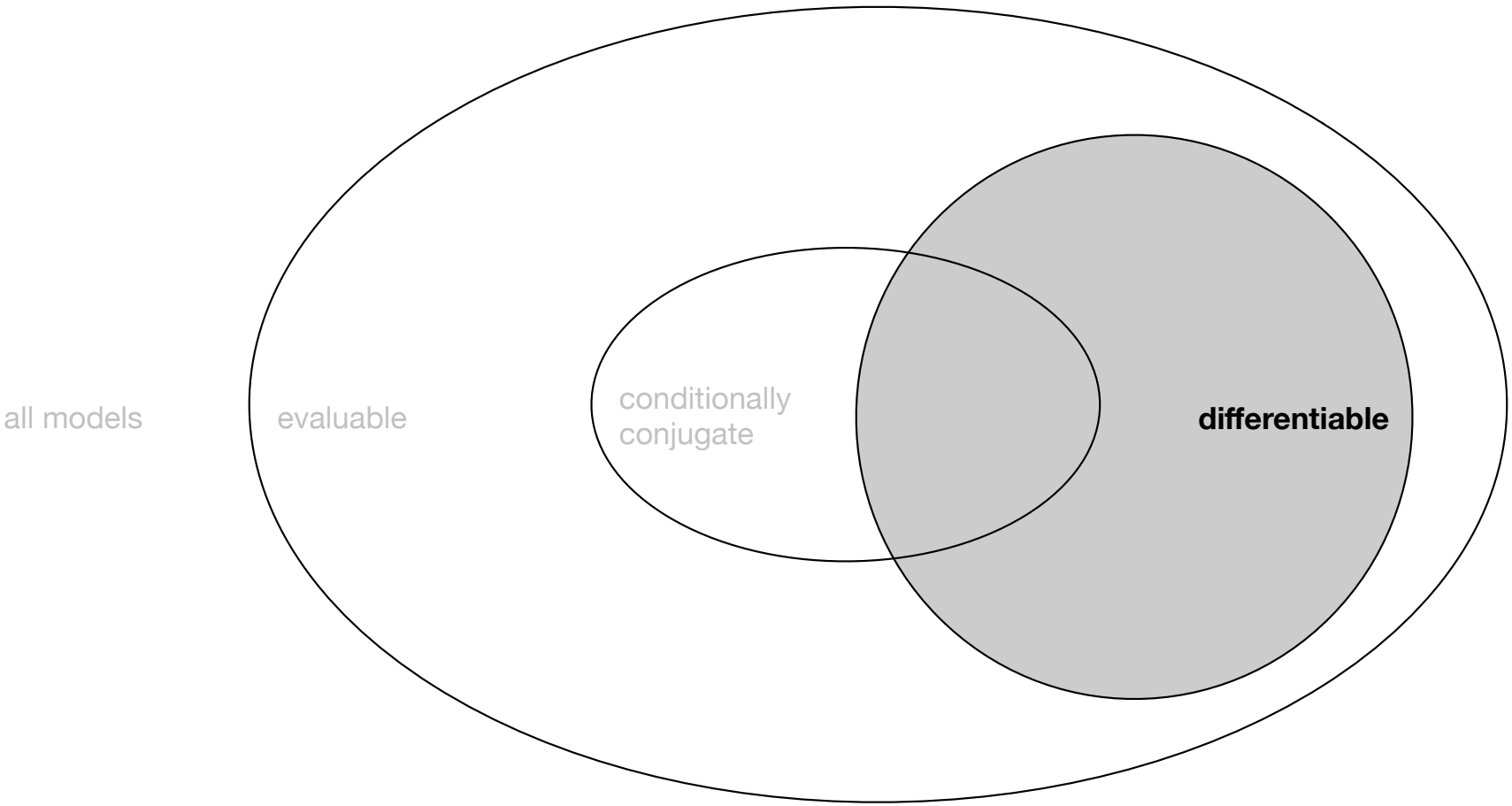
- Suppose $\log p(\mathbf{x}, \mathbf{z})$ and $\log q(\mathbf{z})$ are differentiable with respect to $\mathbf{z}$.
- Suppose the variational distribution can be written with a transformation,

$$\begin{aligned}
\boldsymbol{\epsilon} &\sim s(\boldsymbol{\epsilon}) \\
\mathbf{z} &= t(\boldsymbol{\epsilon}, \boldsymbol{\nu}) \\
&\rightarrow \mathbf{z} \sim q(\mathbf{z}; \boldsymbol{\nu}).
\end{aligned}$$

  For example,

$$\begin{aligned}
\epsilon &\sim \text{Normal}(0, 1) \\
z &= \epsilon\sigma + \mu \\
&\rightarrow z \sim \text{Normal}(\mu, \sigma^2).
\end{aligned}$$

- Note: The variational parameters are part of the transformation, but not the "noise" distribution.


all models
evaluable
conditionally
conjugate
differentiable

The reparameterization gradient

$$\nabla_{\nu}\mathcal{L} = \mathbb{E}_{s(\epsilon)}\left[\underbrace{\nabla_{\mathbf{z}}[\log p(\mathbf{x},\mathbf{z}) - \log q(\mathbf{z};\nu)]}_{\text{gradient of instananeous ELBO}} \quad \underbrace{\nabla_{\nu} t(\epsilon,\nu)}_{\text{gradient of transformation}}\right]$$

- This is the reparameterization gradient.
  [Glasserman 1991; Fu 2006; Kingma+ 2014; Rezende+ 2014; Titsias+ 2014]
- Can use autodifferentiation to take gradients (especially of the model)
- Can use and reuse different transformations [e.g., Naesseth+ 2017]

Algorithm 1: *

Input: data $\mathbf{x}$, model $p(\mathbf{z}, \mathbf{x})$.

Algorithm 2: *

Input: data $\mathbf{x}$, model $p(\mathbf{z}, \mathbf{x})$.

Initialize $\boldsymbol{\nu}$ randomly.
Set ρ_t appropriately.

Algorithm 3: *

Input: data $\mathbf{x}$, model $p(\mathbf{z}, \mathbf{x})$.

Initialize $\boldsymbol{\nu}$ randomly.
Set ρ_t appropriately.

while *not converged* **do**

Algorithm 4: *

Input: data $\mathbf{x}$, model $p(\mathbf{z}, \mathbf{x})$.

Initialize $\boldsymbol{\nu}$ randomly.
Set ρ_t appropriately.

while *not converged* **do**

Take S samples from the auxillary variable

$$\boldsymbol{\epsilon}_s \sim s(\epsilon) \quad s = 1 \ldots S$$

Algorithm 5: *

Input: data $\mathbf{x}$, model $p(\mathbf{z}, \mathbf{x})$.

Initialize $\boldsymbol{\nu}$ randomly.
Set ρ_t appropriately.

while *not converged* **do**

Take S samples from the auxillary variable

$$\boldsymbol{\epsilon}_s \sim s(\epsilon) \quad s = 1 \ldots S$$

Calculate the noisy gradient

$$\tilde{g}_t = \frac{1}{S} \sum_{s=1}^{S} \nabla_{\mathbf{z}} [\log p(\mathbf{x}, t(\boldsymbol{\epsilon}_s, \boldsymbol{\nu}_n)) - \log q(t(\boldsymbol{\epsilon}_s, \boldsymbol{\nu}_n); \boldsymbol{\nu}_n)] \nabla_{\boldsymbol{\nu}} t(\boldsymbol{\epsilon}_s, \boldsymbol{\nu}_n)$$

Algorithm 6: *

Input: data $\mathbf{x}$, model $p(\mathbf{z}, \mathbf{x})$.

Initialize $\boldsymbol{\nu}$ randomly.
Set ρ_t appropriately.

while *not converged* **do**

Take S samples from the auxillary variable

$$\boldsymbol{\epsilon}_s \sim s(\epsilon) \quad s = 1 \ldots S$$

Calculate the noisy gradient

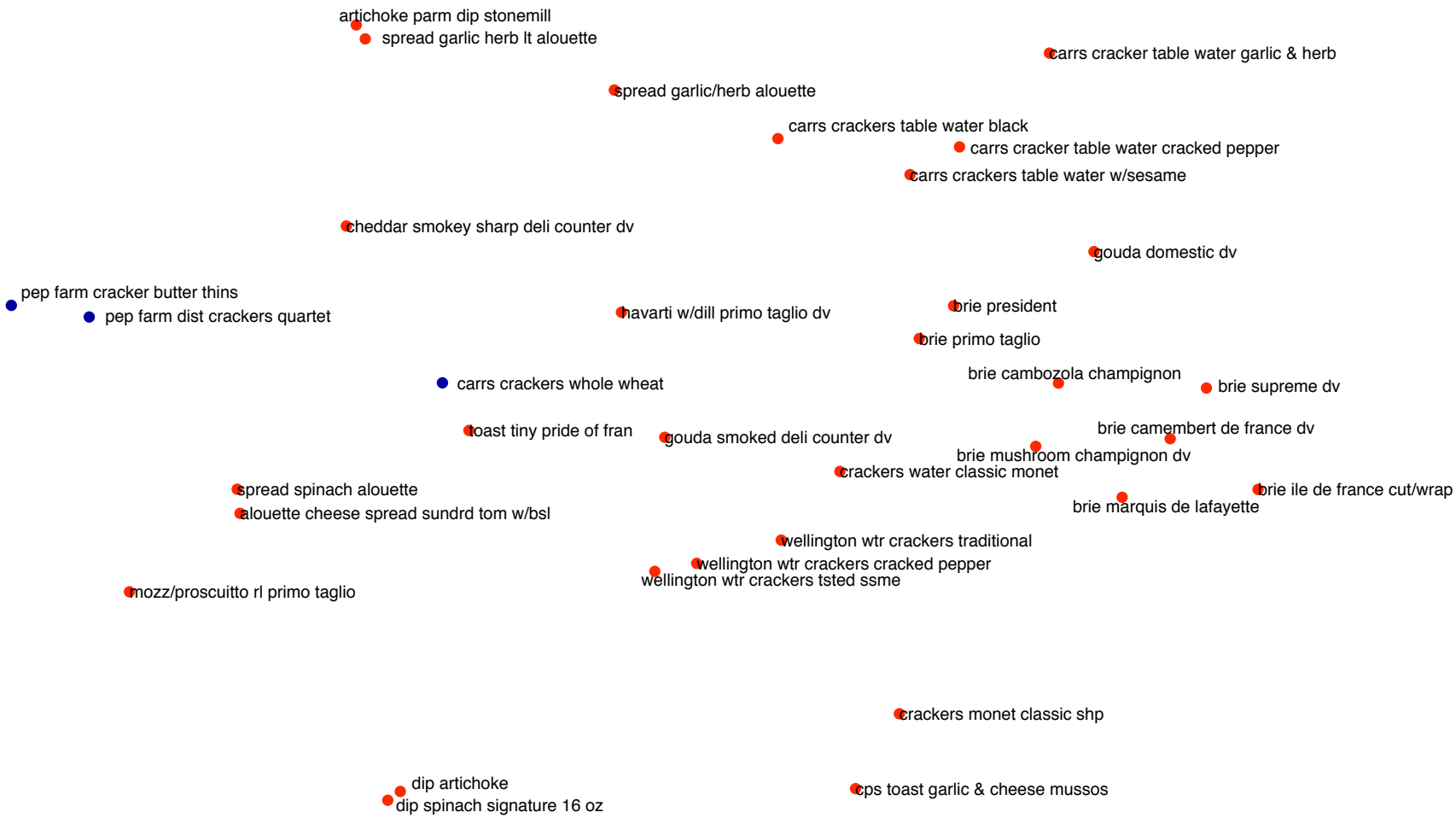
$$\tilde{g}_t = \frac{1}{S} \sum_{s=1}^{S} \nabla_{\mathbf{z}} [\log p(\mathbf{x}, t(\boldsymbol{\epsilon}_s, \boldsymbol{\nu}_n)) - \log q(t(\boldsymbol{\epsilon}_s, \boldsymbol{\nu}_n); \boldsymbol{\nu}_n)] \nabla_{\boldsymbol{\nu}} t(\boldsymbol{\epsilon}_s, \boldsymbol{\nu}_n)$$

Update the variational parameters

$$\boldsymbol{\nu}_{t+1} = \boldsymbol{\nu}_t + \rho_t \tilde{g}_t$$

end

Reparameterization Black Box Variational Inference



Shopper on 5.7M purchases.

[Ruiz+ 2017]

Analysis of 1.7M taxi trajectories, in Stan

[Kucukelbir+ 2017]

Score gradient

$$\mathbb{E}_{q(\mathbf{z};\boldsymbol{\nu})}[\nabla_{\boldsymbol{\nu}} \log q(\mathbf{z};\boldsymbol{\nu})(\log p(\mathbf{x},\mathbf{z}) - \log q(\mathbf{z};\boldsymbol{\nu}))]$$

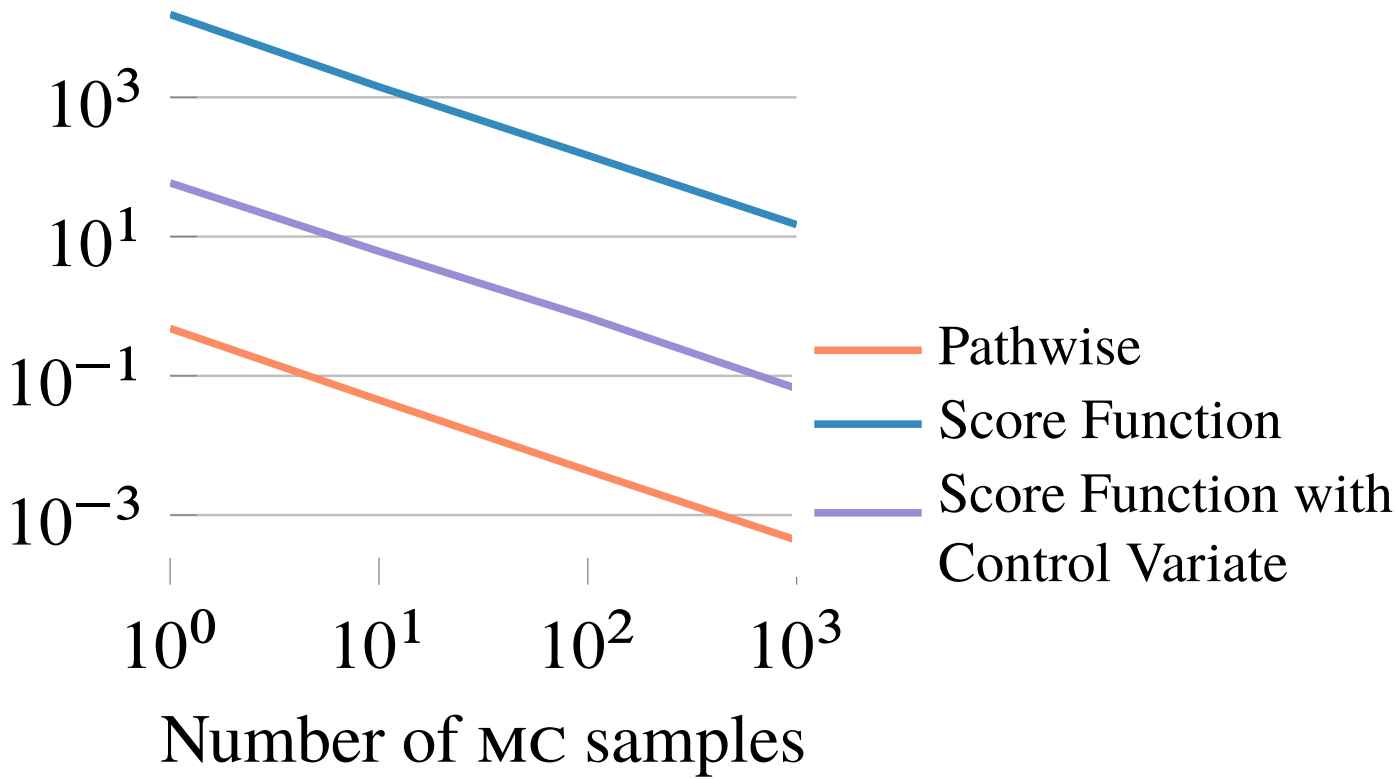
- Works for discrete and continuous models
- Works for a large class of variational approximations
- But the variance of the noisy gradient can be large

Reparameterization gradient

$$\mathbb{E}_{s(\boldsymbol{\epsilon})}[\nabla_{\mathbf{z}}[\log p(\mathbf{x},\mathbf{z}) - \log q(\mathbf{z};\boldsymbol{\nu})]\nabla_{\boldsymbol{\nu}} t(\boldsymbol{\epsilon},\boldsymbol{\nu})]$$

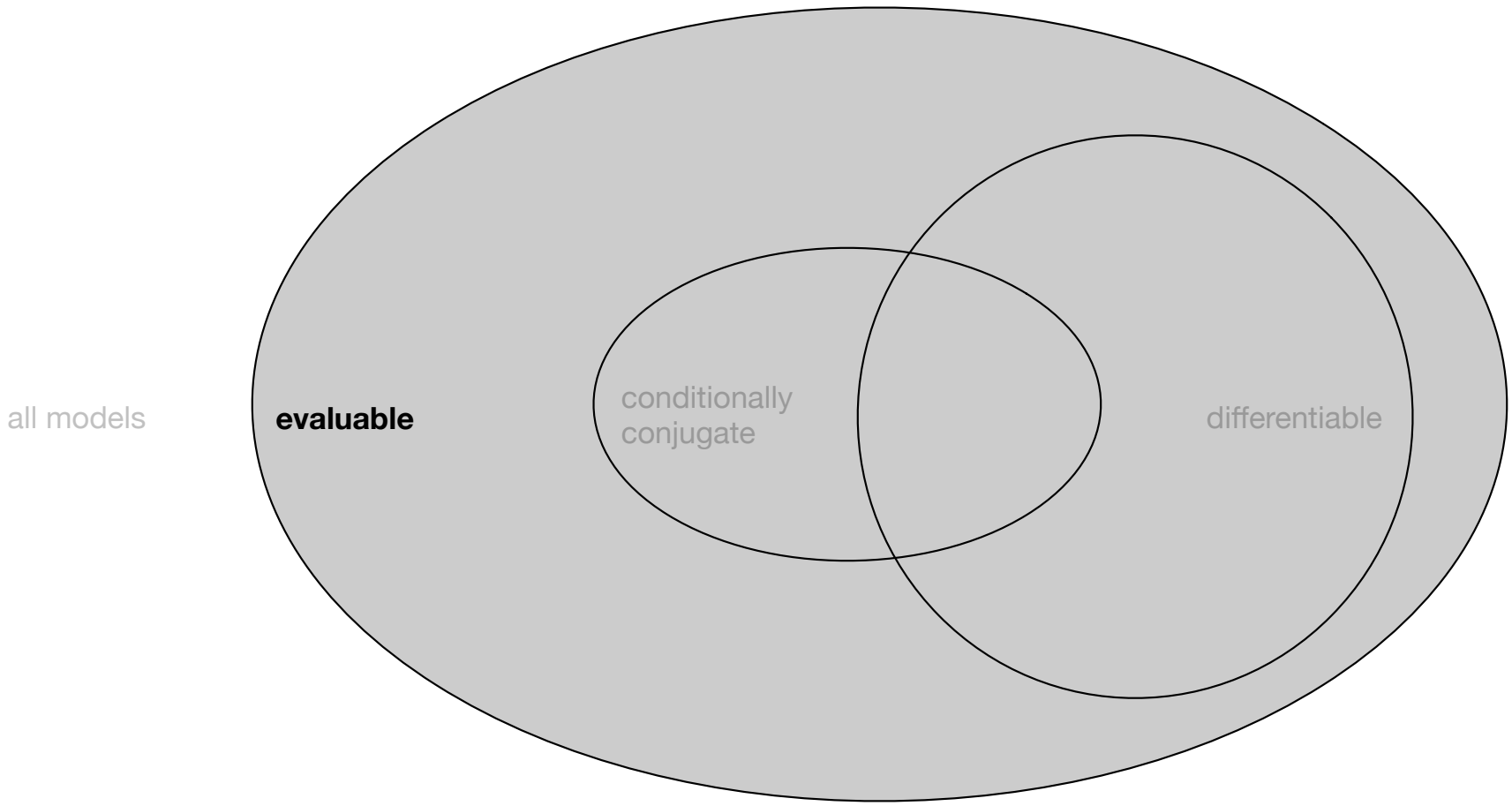
- Requires differentiable models, i.e., no discrete variables
- Requires variational approximation to have form $\mathbf{z} = t(\boldsymbol{\epsilon},\boldsymbol{\nu})$
- Better behaved variance

Variance comparison

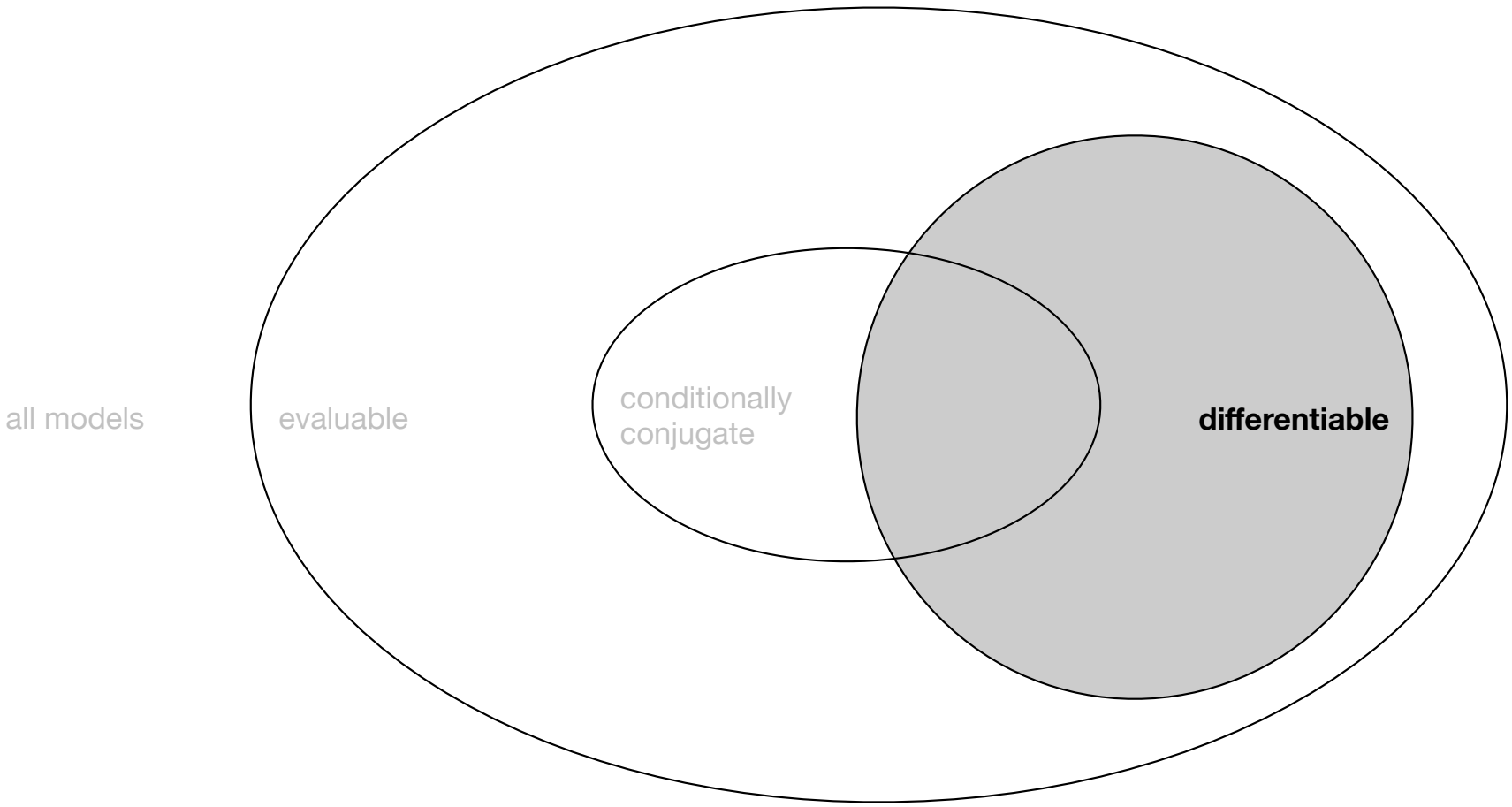


[Kucukelbir+ 2017]

Models that can use the score gradient

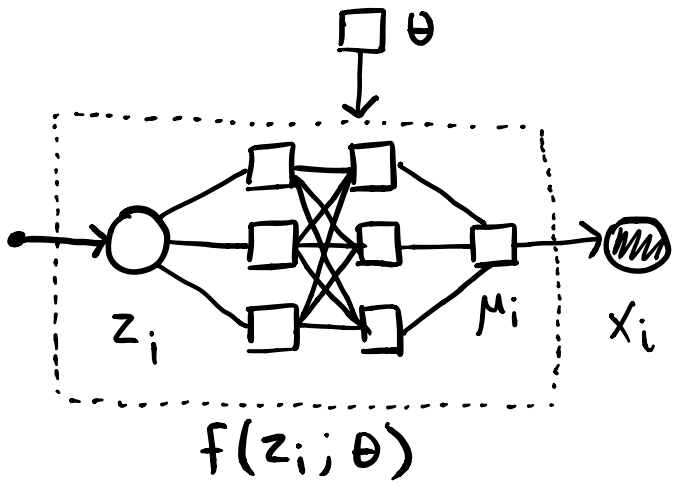


Models that can use the reparameterization gradient



What is a variational autoencoder?

Deep generative models



MODEL

$\leftarrow p(z_i, x_i \mid \theta)$

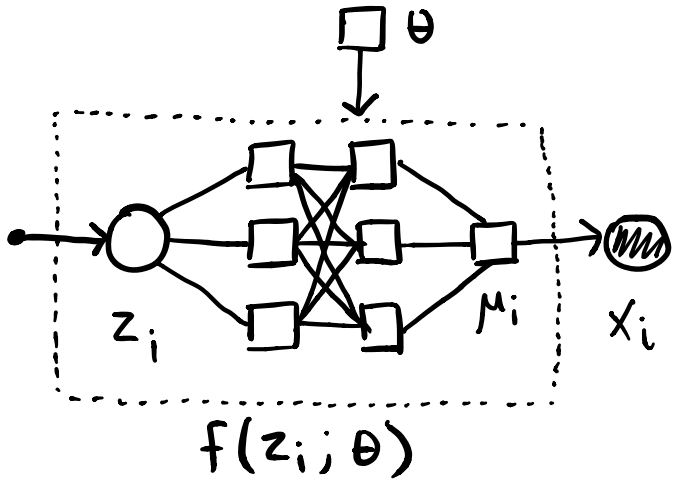
"decoder"

- Deep generative model [Kingma+ 2013; Rezende+ 2014]

$$z_i \sim \mathcal{N}(0, I)$$
$$x_i \sim \mathcal{N}(f(z_i; \theta), \sigma^2),$$

where $f(z_i; \theta)$ is a neural network with parameters θ and input z_i.

Deep generative models



MODEL

$\leftarrow p(z_i, x_i \mid \theta)$

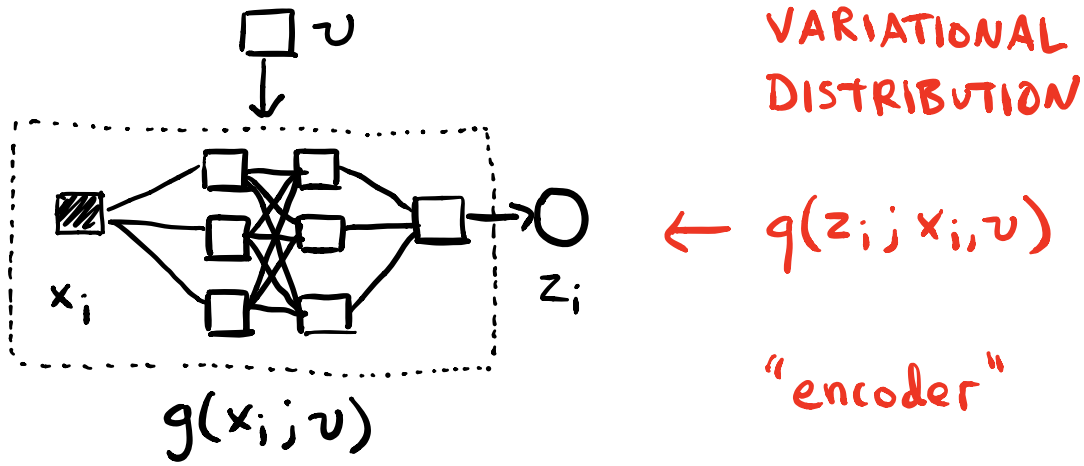
"decoder"

- Fix the global parameters θ. Infer the local variables z_i,

$$p(z_i \mid x_i, \theta) = \frac{p(z_i)p(x_i \mid z_i, \theta)}{\int p(z_i')p(x_i \mid z_i', \theta)\, \mathrm{d}z_i'}$$

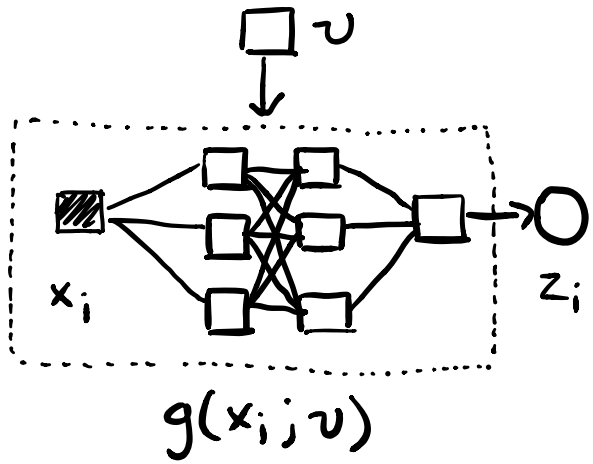
- This inference is intractable because of the integral in the denominator.

Amortized inference



- Use VI at the local level, aiming to fit a $q(z_i)$ that is close to the posterior.
- **Amortization**: the variational family $q(z_i; x_i, \nu)$ is a function of the input x_i and shared variational parameters ν [Gershman and Goodman 2014] .
- Let $q(z_i; x_i, \nu)$ be $\mathcal{N}(g(x_i; \nu), 1)$, where $g(x_i; \nu)$ is the **inference network**; it has parameters ν and input x_i.

The variational autoencoder



VARIATIONAL DISTRIBUTION

$\leftarrow q(z_i; x_i, v)$

"encoder"

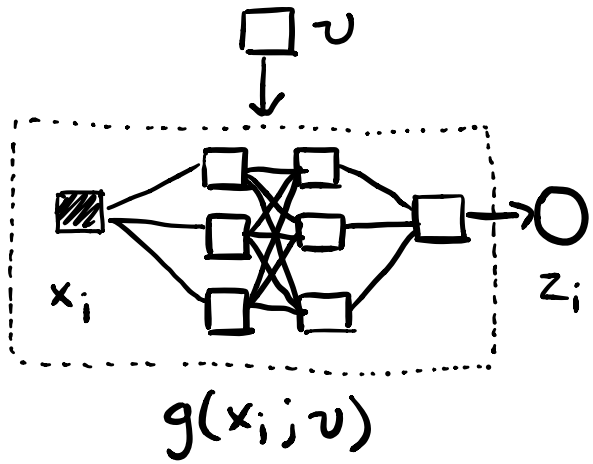
- Amortization ties together the ELBO for each z_i. The objective is

$$\mathcal{L}(v) = \sum_{i=1}^{n} \mathbb{E}_{q(z_i;x_i,v)} \left[\log p(z_i) + \log p(x_i \mid z_i, \theta) - \log q(z_i; x_i, v)\right].$$

- The expectation in the ith term uses $q(z_i; x_i, v)$.

- Amortization is about "learning to infer."
  (Open research: there seems to be more to this story.)

The variational autoencoder



VARIATIONAL DISTRIBUTION

$\leftarrow q(z_i; x_i, v)$

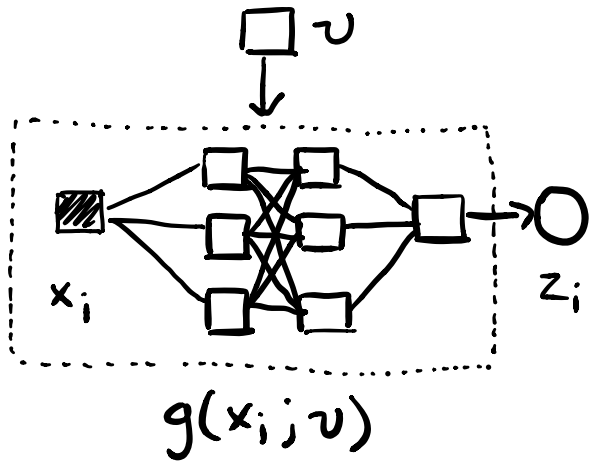
"encoder"

- Use the reparameterization gradient.
- First write z_i down as a transformation,

$$\varepsilon \sim \mathcal{N}(0, 1)$$
$$t(\varepsilon, x_i, v) = \varepsilon + g(x_i; v).$$

- This transformation involves variational parameters v and datapoint x_i.

The variational autoencoder



VARIATIONAL DISTRIBUTION

$\leftarrow q(z_i; x_i, v)$

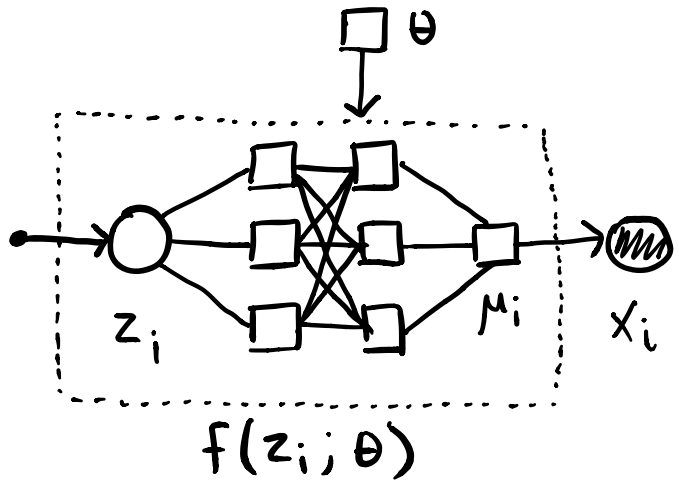
"encoder"

- With the amortized family, the reparameterization gradient is

$$\nabla_v \mathcal{L} = \sum_{i=1}^{n} \mathbb{E}_{s(\varepsilon)} \left[\nabla_{z_i} (\log p(z_i) + \log p(x_i \mid z_i, \theta) - \log q(z_i; x_i, v)) \nabla_v t(\varepsilon, v, x_i) \right].$$

- We can calculate this gradient with Monte Carlo.
- The gradients involved—of the log likelihood, log variational factor, and transformation—involve standard NN calculations (i.e., backprop) of either the model's NN (θ) or the variational NN (v).

Fitting the model



MODEL

$\leftarrow p(z_i, x_i \mid \theta)$

"decoder"

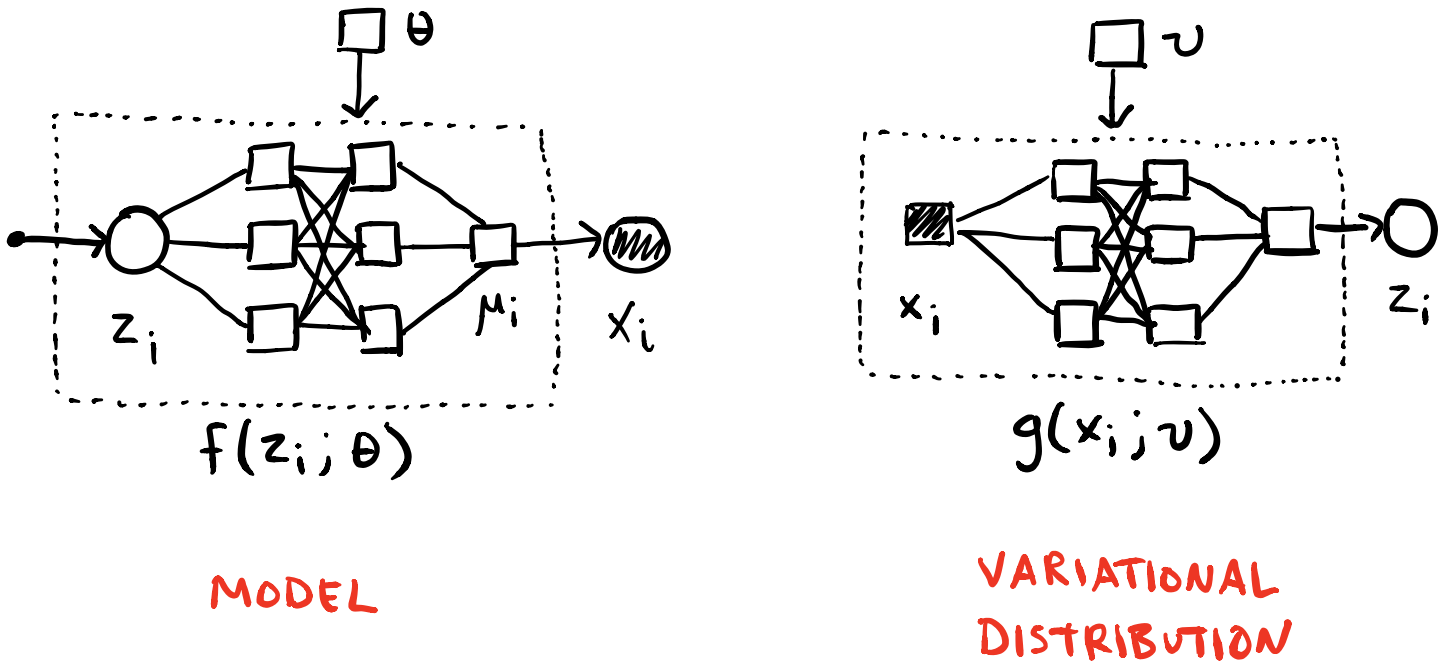
- The ELBO is a bound on the log likelihood $\log p(\mathbf{x}; \theta)$.
- Fit the model by following its gradient with respect to θ,

$$\nabla_\theta \mathscr{L} = \sum_{i=1}^{n} \mathbb{E}_{s(\varepsilon)} [\nabla_\theta \log p(x \mid z_i, \theta)].$$

Here again

$$z_i = t(\varepsilon, x_i, \nu).$$

So what is a VAE?



- Simultaneously optimize the variational family ν and model θ.
- A VAE
  - samples ε_i for each datapoint and calculates z_i.
  - uses these samples to calculate noisy gradients with respect to ν and θ.
  - follows those gradients in a stochastic optimization
- (VAEs are implemented as a stochastic computational graph of the MC approximation of the ELBO; backprop takes care of the rest.)

Algorithm 7: *

Input: data $\mathbf{x}$.

Algorithm 8: *

Input: data $\mathbf{x}$.

Initialize $\boldsymbol{\nu}$ randomly; set ρ_t appropriately.

Algorithm 9: *

Input: data $\mathbf{x}$.

Initialize $\boldsymbol{\nu}$ randomly; set ρ_t appropriately.

while *not converged* **do**

Algorithm 10: *

Input: data **x**.

Initialize $\boldsymbol{\nu}$ randomly; set ρ_t appropriately.

while *not converged* **do**

  for *each datapoint i* **do**

    Draw the noise variable and calculate the latent variable

$$\varepsilon_i \sim \mathcal{N}(0,1); \quad z_i = t(\varepsilon_i, x_i, \nu)$$

  end

Algorithm 11: *

Input: data **x**.

Initialize $\boldsymbol{\nu}$ randomly; set ρ_t appropriately.

while *not converged* **do**

  for *each datapoint i* **do**

    Draw the noise variable and calculate the latent variable

$$\varepsilon_i \sim \mathcal{N}(0,1); \quad z_i = t(\varepsilon_i, x_i, \nu)$$

  end

  Calculate the noisy gradients

$$\tilde{g}_\nu = \sum_{i=1}^{n} \nabla_{z_i}(\log p(z_i) + \log p(x_i \mid z_i, \theta_t) - \log q(z_i; x_i, \nu_t)) \nabla_\nu t(\varepsilon_i, x_i, \nu_t)$$
$$\tilde{g}_\theta = \sum_{i=1}^{n} \nabla_\theta \log p(x_i \mid z_i, \theta_t)$$

Algorithm 12: *

Input: data $\mathbf{x}$.

Initialize $\boldsymbol{\nu}$ randomly; set ρ_t appropriately.

while *not converged* **do**

  for *each datapoint i* **do**

    Draw the noise variable and calculate the latent variable

$$\varepsilon_i \sim \mathcal{N}(0,1); \quad z_i = t(\varepsilon_i, x_i, \nu)$$

  end

  Calculate the noisy gradients

$$\tilde{g}_\nu = \sum_{i=1}^{n} \nabla_{z_i}(\log p(z_i) + \log p(x_i \mid z_i, \theta_t) - \log q(z_i; x_i, \nu_t)) \nabla_\nu t(\varepsilon_i, x_i, \nu_t)$$
$$\tilde{g}_\theta = \sum_{i=1}^{n} \nabla_\theta \log p(x_i \mid z_i, \theta_t)$$

  Update the variational parameters and model parameters

$$\nu_{t+1} = \nu_t + \rho_t \tilde{g}_\nu$$
$$\theta_{t+1} = \theta_t + \rho_t \tilde{g}_\theta$$

end

The variational autoencoder

Let's derive BBVI!

Why do we need black box variational inference?

- Here is a recipe for variational inference
  - Posit a model
  - Choose a variational family
  - Integrate (calculate the ELBO)
  - Take derivatives
  - Optimize
- What can go wrong?

A simple failure

- Take the simplest machine learning model, Bayesian logistic regression.
- Data are pairs (x_i, y_i)
  - x_i is a covariate
  - $y_i \in \{0, 1\}$ is a binary label
  - z are the regression coefficients
- Conditional on covariates, Bayesian LR posits a generative process of labels

$$z \sim N(0, 1)$$
$$y_i \,|\, x_i, z \sim \text{Bernoulli}(\sigma(z x_i)),$$

  where $\sigma(\cdot)$ is the logistic function, mapping reals to $(0, 1)$.

- Consider just one data point (x, y). Set $y = 1$, so the datapoint is $(x, 1)$.
- The goal is to approximate the posterior coefficient $p(z \mid x, y)$.
- The variational family $q(z; \boldsymbol{\nu})$ is a normal; $\boldsymbol{\nu} = (\mu, \sigma^2)$. The ELBO is

$$\mathcal{L}(\mu, \sigma^2) = \mathbb{E}_q[\log p(z) + \log p(y \mid x, z) - \log q(z)]$$

- Try to calculate the ELBO:

$$\mathscr{L}(\mu, \sigma^2) \quad = \quad \mathbb{E}_q[\log p(z) - \log q(z) + \log p(y \mid x, z)]$$

- Try to calculate the ELBO:

$$\begin{aligned}\mathcal{L}(\mu, \sigma^2) &= \mathbb{E}_q[\log p(z) - \log q(z) + \log p(y \mid x, z)] \\ &= -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log \sigma^2 + \mathbb{E}_q[\log p(y \mid x, z)] + C\end{aligned}$$

- Try to calculate the ELBO:

$$\begin{aligned}
\mathscr{L}(\mu, \sigma^2) &= \mathbb{E}_q[\log p(z) - \log q(z) + \log p(y \mid x, z)] \\
&= -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log \sigma^2 + \mathbb{E}_q[\log p(y \mid x, z)] + C \\
&= -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log \sigma^2 + \mathbb{E}_q[yxz - \log(1 + \exp(xz))]
\end{aligned}$$

- Try to calculate the ELBO:

$$
\begin{aligned}
\mathcal{L}(\mu, \sigma^2) &= \mathbb{E}_q[\log p(z) - \log q(z) + \log p(y \mid x, z)] \\
&= -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log \sigma^2 + \mathbb{E}_q[\log p(y \mid x, z)] + C \\
&= -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log \sigma^2 + \mathbb{E}_q[yxz - \log(1 + \exp(xz))] \\
&= -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log \sigma^2 + yx\mu - \mathbb{E}_q[\log(1 + \exp(xz))]
\end{aligned}
$$

- Try to calculate the ELBO:

$$\begin{aligned}
\mathscr{L}(\mu, \sigma^2) &= \mathbb{E}_q[\log p(z) - \log q(z) + \log p(y \mid x, z)] \\
&= -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log \sigma^2 + \mathbb{E}_q[\log p(y \mid x, z)] + C \\
&= -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log \sigma^2 + \mathbb{E}_q[yxz - \log(1 + \exp(xz))] \\
&= -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log \sigma^2 + yx\mu - \mathbb{E}_q[\log(1 + \exp(xz))]
\end{aligned}$$

- We are stuck—we cannot analytically take the expectation.

- Try to calculate the ELBO:

$$\begin{aligned}
\mathscr{L}(\mu, \sigma^2) &= \mathbb{E}_q[\log p(z) - \log q(z) + \log p(y \mid x, z)] \\
&= -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log \sigma^2 + \mathbb{E}_q[\log p(y \mid x, z)] + C \\
&= -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log \sigma^2 + \mathbb{E}_q[yxz - \log(1 + \exp(xz))] \\
&= -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2}\log \sigma^2 + yx\mu - \mathbb{E}_q[\log(1 + \exp(xz))]
\end{aligned}$$

- We are stuck—we cannot analytically take the expectation.

- Q: Why not take gradients of MC estimates of the ELBO?

  A: It's complicated to take gradients when the samples depend on the variable you are optimizing, here the variational parameters

Options?

- Derive a model-specific bound
  [Jordan and Jaakola 1996], [Braun and McAuliffe 2008], others
- Use other approximations (that require model-specific analysis)
  [Wang and Blei 2013], [Knowles and Minka 2011]
- But neither satisfies the *black box criteria*.

Let's derive BBVI

- Define the **instantaneous ELBO**

$$g(\mathbf{z}, \boldsymbol{\nu}) = \log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}; \boldsymbol{\nu}).$$

  The ELBO is its expectation

$$\mathscr{L} = \mathbb{E}_q \left[g(\mathbf{z}, \boldsymbol{\nu}) \right] = \int q(\mathbf{z}; \boldsymbol{\nu}) g(\mathbf{z}, \boldsymbol{\nu}) d\mathbf{z}$$

Let's derive BBVI

- Define the **instantaneous ELBO**

$$g(\mathbf{z}, \boldsymbol{\nu}) = \log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}; \boldsymbol{\nu}).$$

  The ELBO is its expectation

$$\mathcal{L} = \mathbb{E}_q \left[g(\mathbf{z}, \boldsymbol{\nu}) \right] = \int q(\mathbf{z}; \boldsymbol{\nu}) g(\mathbf{z}, \boldsymbol{\nu}) d\mathbf{z}$$

- We want to calculate $\nabla_{\boldsymbol{\nu}} \mathcal{L}$ **as an expectation**.
  (Then we can use Monte Carlo and stochastic gradients.)

Let's derive BBVI

- Define the **instantaneous ELBO**

$$g(\mathbf{z}, \boldsymbol{\nu}) = \log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}; \boldsymbol{\nu}).$$

  The ELBO is its expectation

$$\mathcal{L} = \mathbb{E}_q \left[g(\mathbf{z}, \boldsymbol{\nu}) \right] = \int q(\mathbf{z}; \boldsymbol{\nu}) g(\mathbf{z}, \boldsymbol{\nu}) d\mathbf{z}$$

- We want to calculate $\nabla_{\boldsymbol{\nu}} \mathcal{L}$ **as an expectation**.
  (Then we can use Monte Carlo and stochastic gradients.)

- Fact:

$$\nabla_{\boldsymbol{\nu}} q(\mathbf{z}; \boldsymbol{\nu}) = q(\mathbf{z}; \boldsymbol{\nu}) \nabla_{\boldsymbol{\nu}} \log q(\mathbf{z}; \boldsymbol{\nu}).$$

  You can confirm it in your mind.

Derive the score gradient

- With this fact,

$$\nabla_{\nu}\mathcal{L} \quad = \quad \nabla_{\nu}\int q(\mathbf{z};\boldsymbol{\nu})g(\mathbf{z},\boldsymbol{\nu})d\mathbf{z}$$

Derive the score gradient

- With this fact,

$$\begin{aligned}\nabla_{\nu}\mathcal{L} &= \nabla_{\nu}\int q(\mathbf{z};\boldsymbol{\nu})g(\mathbf{z},\boldsymbol{\nu})d\mathbf{z} \\ &= \int \nabla_{\nu}q(\mathbf{z};\boldsymbol{\nu})g(\mathbf{z},\boldsymbol{\nu}) + q(\mathbf{z};\boldsymbol{\nu})\nabla_{\nu}g(\mathbf{z},\boldsymbol{\nu})d\mathbf{z}\end{aligned}$$

Derive the score gradient

- With this fact,

$$\begin{aligned}
\nabla_{\nu}\mathcal{L} &= \nabla_{\nu}\int q(\mathbf{z};\boldsymbol{\nu})g(\mathbf{z},\boldsymbol{\nu})d\mathbf{z} \\
&= \int \nabla_{\nu} q(\mathbf{z};\boldsymbol{\nu})g(\mathbf{z},\boldsymbol{\nu}) + q(\mathbf{z};\boldsymbol{\nu})\nabla_{\nu} g(\mathbf{z},\boldsymbol{\nu})d\mathbf{z} \\
&= \int q(\mathbf{z};\boldsymbol{\nu})\nabla_{\nu}\log q(\mathbf{z};\boldsymbol{\nu})g(\mathbf{z},\boldsymbol{\nu}) + q(\mathbf{z};\boldsymbol{\nu})\nabla_{\nu} g(\mathbf{z},\boldsymbol{\nu})d\mathbf{z}
\end{aligned}$$

Derive the score gradient

- With this fact,

$$
\begin{aligned}
\nabla_{\nu}\mathscr{L} &= \nabla_{\nu}\int q(\mathbf{z};\boldsymbol{\nu})g(\mathbf{z},\boldsymbol{\nu})d\mathbf{z} \\
&= \int \nabla_{\nu} q(\mathbf{z};\boldsymbol{\nu})g(\mathbf{z},\boldsymbol{\nu}) + q(\mathbf{z};\boldsymbol{\nu})\nabla_{\nu} g(\mathbf{z},\boldsymbol{\nu})d\mathbf{z} \\
&= \int q(\mathbf{z};\boldsymbol{\nu})\nabla_{\nu}\log q(\mathbf{z};\boldsymbol{\nu})g(\mathbf{z},\boldsymbol{\nu}) + q(\mathbf{z};\boldsymbol{\nu})\nabla_{\nu} g(\mathbf{z},\boldsymbol{\nu})d\mathbf{z} \\
&= \mathbb{E}_{q(\mathbf{z};\boldsymbol{\nu})}[\nabla_{\nu}\log q(\mathbf{z};\boldsymbol{\nu})g(\mathbf{z},\boldsymbol{\nu}) + \nabla_{\nu} g(\mathbf{z},\boldsymbol{\nu})]
\end{aligned}
$$

Derive the score gradient

- With this fact,

$$
\begin{aligned}
\nabla_{\nu}\mathscr{L} &= \nabla_{\nu}\int q(\mathbf{z};\boldsymbol{\nu})g(\mathbf{z},\boldsymbol{\nu})d\mathbf{z} \\
&= \int \nabla_{\nu} q(\mathbf{z};\boldsymbol{\nu})g(\mathbf{z},\boldsymbol{\nu}) + q(\mathbf{z};\boldsymbol{\nu})\nabla_{\nu} g(\mathbf{z},\boldsymbol{\nu})d\mathbf{z} \\
&= \int q(\mathbf{z};\boldsymbol{\nu})\nabla_{\nu}\log q(\mathbf{z};\boldsymbol{\nu})g(\mathbf{z},\boldsymbol{\nu}) + q(\mathbf{z};\boldsymbol{\nu})\nabla_{\nu} g(\mathbf{z},\boldsymbol{\nu})d\mathbf{z} \\
&= \mathbb{E}_{q(\mathbf{z};\boldsymbol{\nu})}[\nabla_{\nu}\log q(\mathbf{z};\boldsymbol{\nu})g(\mathbf{z},\boldsymbol{\nu}) + \nabla_{\nu} g(\mathbf{z},\boldsymbol{\nu})]
\end{aligned}
$$

- **We have written the gradient as an expectation.**

- The second term vanishes,

$$\mathbb{E}_q[\nabla_{\boldsymbol{\nu}} g(\mathbf{z}, \boldsymbol{\nu})] = -\mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\mathbf{z}; \boldsymbol{\nu})] = 0.$$

- The second term vanishes,

$$\mathbb{E}_q[\nabla_{\boldsymbol{\nu}} g(\mathbf{z}, \boldsymbol{\nu})] = -\mathbb{E}_q[\nabla_{\boldsymbol{\nu}} \log q(\mathbf{z}; \boldsymbol{\nu})] = 0.$$

- What's left is the score gradient,

$$\nabla_{\boldsymbol{\nu}} \mathcal{L} = \mathbb{E}_{q(\mathbf{z};\boldsymbol{\nu})}[\nabla_{\boldsymbol{\nu}} \log q(\mathbf{z}; \boldsymbol{\nu})(\log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}; \boldsymbol{\nu}))].$$

- The second term vanishes,

$$\mathbb{E}_q[\nabla_\nu g(\mathbf{z}, \nu)] = -\mathbb{E}_q[\nabla_\nu \log q(\mathbf{z}; \nu)] = 0.$$

- What's left is the score gradient,

$$\nabla_\nu \mathcal{L} = \mathbb{E}_{q(\mathbf{z};\nu)}[\nabla_\nu \log q(\mathbf{z}; \nu)(\log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}; \nu))].$$

- Aside: Why is the expectation of the score function equal to zero?

$$\begin{aligned}
\mathbb{E}_q[\nabla_\nu \log q(\mathbf{z}; \nu)] &= \int q(\mathbf{z}; \nu) \nabla_\nu \log q(\mathbf{z}; \nu) \mathrm{d}\mathbf{z} \\
&= \int \nabla_\nu q(\mathbf{z}; \nu) \mathrm{d}\mathbf{z} \\
&= \nabla_\nu \int q(\mathbf{z}; \nu) \mathrm{d}\mathbf{z} = \nabla_\nu 1 = 0.
\end{aligned}$$

Derive the reparameterization gradient

- Assume $\log p(\mathbf{x}, \mathbf{z})$ and $\log q(\mathbf{z})$ are differentiable with respect to $\mathbf{z}$.
- Also assume that we can express the variational distribution with a transformation, where

$$\begin{aligned} \boldsymbol{\epsilon} &\sim s(\boldsymbol{\epsilon}) \\ \mathbf{z} &= t(\boldsymbol{\epsilon}, \boldsymbol{\nu}) \\ &\rightarrow \mathbf{z} \sim q(\mathbf{z}; \boldsymbol{\nu}) \end{aligned}$$

- Rewrite the ELBO using $\mathbf{z} = t(\boldsymbol{\epsilon}, \boldsymbol{\nu})$,

$$\mathcal{L} = \mathbb{E}_{s(\boldsymbol{\epsilon})}[g(t(\boldsymbol{\epsilon}, \boldsymbol{\nu}), \boldsymbol{\nu})].$$

- Now take the gradient of the ELBO with respect to $\boldsymbol{\nu}$.

- Now take the gradient of the ELBO with respect to $\boldsymbol{\nu}$.

- The gradient easily goes into the expectation. Then use the chain rule,

$$\nabla_{\boldsymbol{\nu}} \mathscr{L} = \mathbb{E}_{s(\boldsymbol{\varepsilon})}[\nabla_{\mathbf{z}}(\log p(\mathbf{z}, \mathbf{x}) - \log q(\mathbf{z}; \boldsymbol{\nu}))\nabla_{\boldsymbol{\nu}} t(\boldsymbol{\varepsilon}, \boldsymbol{\nu}) - \nabla_{\boldsymbol{\nu}} \log q(\mathbf{z}; \boldsymbol{\nu}))].$$

  Here we expanded the instantaneous ELBO and used chain rule for functions of two variables,

$$\frac{\mathrm{d}f(x(t), y(t))}{\mathrm{d}t} = \frac{\mathrm{d}f}{\mathrm{d}x}\frac{\mathrm{d}f}{\mathrm{d}t} + \frac{\mathrm{d}f}{\mathrm{d}y}\frac{\mathrm{d}y}{\mathrm{d}t}.$$

- Now take the gradient of the ELBO with respect to $\boldsymbol{\nu}$.

- The gradient easily goes into the expectation. Then use the chain rule,

$$\nabla_{\boldsymbol{\nu}} \mathcal{L} = \mathbb{E}_{s(\boldsymbol{\varepsilon})}[\nabla_{\mathbf{z}}(\log p(\mathbf{z}, \mathbf{x}) - \log q(\mathbf{z}; \boldsymbol{\nu}))\nabla_{\boldsymbol{\nu}} t(\boldsymbol{\varepsilon}, \boldsymbol{\nu}) - \nabla_{\boldsymbol{\nu}} \log q(\mathbf{z}; \boldsymbol{\nu}))].$$

  Here we expanded the instantaneous ELBO and used chain rule for functions of two variables,

$$\frac{\mathrm{d}f(x(t), y(t))}{\mathrm{d}t} = \frac{\mathrm{d}f}{\mathrm{d}x}\frac{\mathrm{d}f}{\mathrm{d}t} + \frac{\mathrm{d}f}{\mathrm{d}y}\frac{\mathrm{d}y}{\mathrm{d}t}.$$

- The second term vanishes as above, $-\mathbb{E}[\nabla_{\boldsymbol{\nu}} \log q(\mathbf{z}, \boldsymbol{\nu})] = 0$.

  The first terms provide the reparameterization gradient,

$$\nabla_{\boldsymbol{\nu}} \mathcal{L} = \mathbb{E}_{s(\boldsymbol{\varepsilon})}[\nabla_{\mathbf{z}}(\log p(\mathbf{z}, \mathbf{x}) - \log q(\mathbf{z}; \boldsymbol{\nu}))\nabla_{\boldsymbol{\nu}} t(\boldsymbol{\varepsilon}, \boldsymbol{\nu})].$$