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, Blei, Wang, Paisley, JMLR 2013]
Scenes, concepts and control.

[Eslami et al., 2016, Lake et al. 2015]
Population analysis of 2 billion genetic measurements

[Gopalan, Hao, Blei, Storey, Nature Genetics (in press)]
Neuroscience analysis of 220 million fMRI measurements

[Manning et al., PLOS ONE 2014]
Compression and content generation.

[Van den Oord et al., 2016, Gregor et al., 2016]
Analysis of 1.7M taxi trajectories, in Stan

[Kucukelbir et al., 2016]
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

- **Inference** is the key algorithmic problem.

- Answers the question: What does this model say about this data?

- Our goal: **General** and **scalable** approaches to inference
Figure S2: Population structure inferred from the TGP data set using the TeraStructure algorithm at three values for the number of populations $K$. The visualization of the ✓'s in the Figure shows patterns consistent with the major geographical regions. Some of the clusters identify a specific region (e.g. red for Africa) while others represent admixture between regions (e.g. green for Europeans and Central/South Americans). The presence of clusters that are shared between different regions demonstrates the more continuous nature of the structure. The new cluster from $K=7$ to $K=8$ matches structure differentiating between American groups. For $K=9$, the new cluster is unpopulated.

[Box, 1980; Rubin, 1984; Gelman et al., 1996; Blei, 2014]
PART I

Main ideas and historical context
A probabilistic model is a joint distribution of hidden variables \( z \) and observed variables \( x \),

\[
p(z, x).
\]

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

\[
p(z | x) = \frac{p(z, x)}{p(x)}.
\]

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

- VI turns inference into optimization.
- Posit a variational family of distributions over the latent variables, $q(z; \nu)$
- Fit the variational parameters $\nu$ to be close (in KL) to the exact posterior.

(There are alternative divergences, which connect to algorithms like EP, BP, and others.)
Example: Mixture of Gaussians

![Diagram showing the evolution of a mixture of Gaussians through iterations.](images by Alp Kucukelbir)
History

- Variational inference adapts **ideas from statistical physics** to probabilistic inference. Arguably, it began in the late eighties with Peterson and Anderson (1987), who used mean-field methods to fit a neural network.

- This idea was picked up by Jordan’s lab in the early 1990s—Tommi Jaakkola, Lawrence Saul, Zoubin Gharamani—who **generalized it to many probabilistic models**. (A review paper is Jordan et al., 1999.)

- In parallel, Hinton and Van Camp (1993) also **developed mean-field for neural networks**. Neal and Hinton (1993) connected this idea to the EM algorithm, which lead to further variational methods for mixtures of experts (Waterhouse et al., 1996) and HMMs (MacKay, 1997).
Today

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

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

- Our goal today is to teach you the basics, explain some of the newer ideas, and to suggest open areas of new research.
Variational Inference: Foundations and Modern Methods

Part II: Mean-field VI and stochastic VI

Jordan+, Introduction to Variational Methods for Graphical Models, 1999
Ghahramani and Beal, Propagation Algorithms for Variational Bayesian Learning, 2001
Hoffman+, Stochastic Variational Inference, 2013

Part III: Stochastic gradients of the ELBO

Kingma and Welling, Auto-Encoding Variational Bayes, 2014
Ranganath+, Black Box Variational Inference, 2014
Rezende+, Stochastic Backpropagation and Approximate Inference in Deep Generative Models, 2014

Part IV: Beyond the mean field

Agakov and Barber, An Auxiliary Variational Method, 2004
Gregor+, DRAW: A recurrent neural network for image generation, 2015
Rezende+, Variational Inference with Normalizing Flows, 2015
Ranganath+, Hierarchical Variational Models, 2015
Maaløe+, Auxiliary Deep Generative Models, 2016
VI approximates difficult quantities from complex models.

With **stochastic optimization** we can

- scale up VI to massive data
- enable VI on a wide class of difficult models
- enable VI with elaborate and flexible families of approximations
PART II

Mean-field variational inference
and stochastic variational inference
Motivation: Topic Modeling

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

---


SCIENCE • VOL. 272 • 24 MAY 1996

Documents exhibit multiple topics.
Example: 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
Example: Latent Dirichlet Allocation (LDA)

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

\[ p(\text{topics, proportions, assignments} | \text{documents}) \]

(Note: millions of documents; billions of latent variables)
LDA as a Graphical Model

- Encodes **assumptions** about data with a factorization of the joint
- Connects assumptions to **algorithms** for computing with data
- Defines the **posterior** (through the joint)
The posterior of the latent variables given the documents is

\[ p(\beta, \theta, z | w) = \frac{p(\beta, \theta, z, w)}{\int_\beta \int_\theta \sum_z p(\beta, \theta, z, w)}. \]

- We can’t compute the denominator, the marginal \( p(w) \).
- We use approximate inference.
| Figure 5 | Topics found in 1.8M articles from the New York Times | Figure 4 | c illustrates the graphical model. This model is closely related to a linear factor model, except that each cell's distribution is determined by hidden variables that depend on the cell's row and column. The overlapping plates show how the observations at the nth row share its embedding but use different variables γm for each column. Similarly, the observations in the mth column share its embedding γm but use different variables wn for each row. Casting matrix factorization... |
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
A Generic Class of Models

The observations are $x = x_{1:n}$.

The local variables are $z = z_{1:n}$.

The global variables are $\beta$.

The $i$th data point $x_i$ only depends on $z_i$ and $\beta$.

Compute $p(\beta, z | x)$. 

$$p(\beta, z, x) = p(\beta) \prod_{i=1}^{n} p(z_i, x_i | \beta)$$
A Generic Class of Models

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,

\[ p(z_i | \beta, x_i) = h(z_i) \exp \{ \eta_{\ell}(\beta, x_i)^\top z_i - a(\eta_{\ell}(\beta, x_i)) \} \]

\[ p(\beta | z, x) = h(\beta) \exp \{ \eta_{g}(z, x)^\top \beta - a(\eta_{g}(z, x)) \} . \]
A Generic Class of Models

\[ p(\beta, z, x) = p(\beta) \prod_{i=1}^{n} p(z_i, x_i | \beta) \]

- A **complete conditional** is the conditional of a latent variable given the observations and other latent variable.

- The global parameter comes from conjugacy [Bernardo and Smith, 1994]

\[ \eta_g(z, x) = \alpha + \sum_{i=1}^{n} t(z_i, x_i), \]

where \( \alpha \) is a hyperparameter and \( t(\cdot) \) are sufficient statistics for \( [z_i, x_i] \).
A Generic Class of Models

Global variables

Local variables

\[ p(\beta, z, x) = p(\beta) \prod_{i=1}^{n} p(z_i, x_i | \beta) \]

- 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)
Minimize KL between $q(\beta, z; \nu)$ and the posterior $p(\beta, z | x)$. 
The Evidence Lower Bound

\[ \mathcal{L}(\nu) = \mathbb{E}_q[\log p(\beta, z, x)] - \mathbb{E}_q[\log q(\beta, z; \nu)] \]

- KL is intractable; VI optimizes the evidence lower bound (ELBO) instead.
  - It is a lower bound on \(\log p(x)\).
  - 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

- We need to specify the form of $q(\beta, z)$.
- The **mean-field family** is fully factorized,

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

- Each factor is the same family as the model’s complete conditional,

  $$p(\beta | z, x) = h(\beta) \exp\{\eta_g(z, x)^\top \beta - a(\eta_g(z, x))\}$$
  $$q(\beta; \lambda) = h(\beta) \exp\{\lambda^\top \beta - a(\lambda)\}.$$
Mean-field Variational Inference

- Optimize the ELBO,
  \[ \mathcal{L}(\lambda, \phi) = \mathbb{E}_q[\log p(\beta, z, x)] - \mathbb{E}_q[\log q(\beta, z)]. \]
- Traditional VI uses coordinate ascent [Ghahramani and Beal, 2001]
  \[ \lambda^* = \mathbb{E}_\phi[\eta_g(z, x)]; \phi_i^* = \mathbb{E}_\lambda[\eta_\ell(\beta, x_i)]. \]
- Iteratively update each parameter, holding others fixed.
  - Notice the relationship to Gibbs sampling [Gelfand and Smith, 1990].
  - Caveat: The ELBO is not convex.
Mean-field Variational Inference for LDA

- The local variables are the per-document variables $\theta_d$ and $z_{d,n}$.
- The global variables are the topics $\beta_1, \ldots, \beta_K$.
- The variational distribution is

$$q(\beta, \theta, z) = \prod_{k=1}^{K} q(\beta_k; \lambda_k) \prod_{d=1}^{D} q(\theta_d; \gamma_d) \prod_{n=1}^{N} q(z_{d,n}; \phi_{d,n})$$
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 Stw. Anderson 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.
Mean-field Variational Inference for LDA

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

**Input:** data \( x \), model \( p(\beta, z, x) \).

Initialize \( \lambda \) randomly.

**repeat**

**for each data point \( i \) do**

Set local parameter \( \phi_i \leftarrow \mathbb{E}_\lambda [\eta_\ell(\beta, x_i)] \).

**end**

Set global parameter

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

**until the ELBO has converged**
A Generic Class of Models

\[ p(\beta, z, x) = p(\beta) \prod_{i=1}^{n} p(z_i, x_i | \beta) \]

- 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)
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

Figure S2: Population structure inferred from the TGP data set using the TeraStructure algorithm at three values for the number of populations $K$. The visualization of the $\checkmark$'s in the Figure shows patterns consistent with the major geographical regions. Some of the clusters identify a specific region (e.g. red for Africa) while others represent admixture between regions (e.g. green for Europeans and Central/South Americans). The presence of clusters that are shared between different regions demonstrates the more continuous nature of the structure. The new cluster from $K=7$ to $K=8$ matches structure differentiating between American groups. For $K=9$, the new cluster is unpopulated.
Stochastic Optimization

A STOCHASTIC APPROXIMATION METHOD

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 $\alpha$ 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 $\theta$ in probability.

- Replace the gradient with cheaper noisy estimates [Robbins and Monro, 1951]
- Guaranteed to converge to a local optimum [Bottou, 1996]
- Has enabled modern machine learning
With noisy gradients, update

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

Requires unbiased gradients, \( \mathbb{E}[\hat{\nabla}_\nu \mathcal{L}(\nu)] = \nabla_\nu \mathcal{L}(\nu) \)

Requires the step size sequence \( \rho_t \) follows the Robbins-Monro conditions
Stochastic Variational Inference

- The natural gradient of the ELBO [Amari, 1998; Sato, 2001]

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

- Construct a noisy natural gradient,

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

- This is a good noisy gradient.
  - Its expectation is the exact gradient (unbiased).
  - It only depends on optimized parameters of one data point (cheap).
**Stochastic Variational Inference**

**Input:** data $x$, model $p(\beta, z, x)$.

Initialize $\lambda$ randomly. Set $\rho_t$ appropriately.

repeat
  Sample $j \sim \text{Unif}(1, \ldots, n)$.
  Set local parameter $\phi \leftarrow \mathbb{E}_\lambda[\eta(\beta, x_j)]$.
  Set intermediate global parameter
  $$\hat{\lambda} = \alpha + n\mathbb{E}_\phi[t(Z_j, x_j)].$$
  Set global parameter
  $$\lambda = (1 - \rho_t)\lambda + \rho_t\hat{\lambda}.$$ 
until forever
Stochastic Variational Inference

GLOBAL HIDDEN STRUCTURE

Subsample data
Infer local structure
Update global structure

MASSIVE DATA

Figure S2: Population structure inferred from the TGP data set using the TeraStructure algorithm at three values for the number of populations $K$. The visualization of the $\checkmark$'s in the Figure shows patterns consistent with the major geographical regions. Some of the clusters identify a specific region (e.g. red for Africa) while others represent admixture between regions (e.g. green for Europeans and Central/South Americans). The presence of clusters that are shared between different regions demonstrates the more continuous nature of the structure. The new cluster from $K=7$ to $K=8$ matches structure differentiating between American groups. For $K=9$, the new cluster is unpopulated.
Stochastic Variational Inference in LDA

- Sample a document
- Estimate the local variational parameters using the current topics
- Form intermediate topics from those local parameters
- Update topics as a weighted average of intermediate and current topics
Stochastic Variational Inference in LDA

Documents analyzed

<table>
<thead>
<tr>
<th>Documents analyzed</th>
<th>2048</th>
<th>4096</th>
<th>8192</th>
<th>12288</th>
<th>16384</th>
<th>32768</th>
<th>49152</th>
<th>65536</th>
</tr>
</thead>
<tbody>
<tr>
<td>systems</td>
<td>road</td>
<td>made</td>
<td>service</td>
<td>announced</td>
<td>national</td>
<td>west</td>
<td>language</td>
<td>systems</td>
</tr>
</tbody>
</table>

Top eight words

[Hoffman et al., 2010]
Topics using the HDP, found in 1.8M articles from the New York Times
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)
PART III

Stochastic Gradients of the ELBO
Review: The Promise

- Realized for conditionally conjugate models
- What about the general case?
The Variational Inference Recipe

Start with a model:

\[ p(z, x) \]
The Variational Inference Recipe

Choose a variational approximation:

\[ q(z; \nu) \]
The Variational Inference Recipe

Write down the ELBO:

\[
\mathcal{L}(\nu) = \mathbb{E}_{q(z; \nu)}[\log p(x, z) - \log q(z; \nu)]
\]
The Variational Inference Recipe

Compute the expectation (integral):

Example: $\mathcal{L}(\nu) = x\nu^2 + \log \nu$
The Variational Inference Recipe

Take derivatives:

Example: $\nabla_\nu \mathcal{L}(\nu) = 2x\nu + \frac{1}{\nu}$
The Variational Inference Recipe

Optimize:

\[ \nu_{t+1} = \nu_t + \rho_t \nabla_\nu \mathcal{L} \]
The Variational Inference Recipe

\[ p(x, z) \]
\[ q(z; \nu) \]
\[ \int (\cdots) q(z; \nu) \, dz \]
\[ \nabla_\nu \]
\[ q(z; \nu) \]
Example: Bayesian Logistic Regression

- Data pairs $y_i, x_i$
- $x_i$ are covariates
- $y_i$ are label
- $z$ is the regression coefficient
- Generative process

$$p(z) \sim N(0, 1)$$
$$p(y_i | x_i, z) \sim \text{Bernoulli}(\sigma(zx_i))$$
VI for Bayesian Logistic Regression

Assume:
- We have one data point \((y, x)\)
- \(x\) is a scalar
- The approximating family \(q\) is the normal; \(\nu = (\mu, \sigma^2)\)

The ELBO is

\[
\mathcal{L}(\mu, \sigma^2) = \mathbb{E}_q[\log p(z) + \log p(y | x, z) - \log q(z)]
\]
VI for Bayesian Logistic Regression

\[ \mathcal{L}(\mu, \sigma^2) = \mathbb{E}_q[\log p(z) - \log q(z) + \log p(y|x,z)] \]
VI for Bayesian Logistic Regression

\[ \mathcal{L}(\mu, \sigma^2) \]

\[ = \mathbb{E}_q[\log p(z) - \log q(z) + \log p(y | x, z)] \]

\[ = -\frac{1}{2} (\mu^2 + \sigma^2) + \frac{1}{2} \log \sigma^2 + \mathbb{E}_q[\log p(y | x, z)] + C \]
VI for Bayesian Logistic Regression

\[ \mathcal{L}(\mu, \sigma^2) \]

\[ = \mathbb{E}_q[\log p(z) - \log q(z) + \log p(y|x, z)] \]

\[ = -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2} \log \sigma^2 + \mathbb{E}_q[\log p(y|x, z)] + C \]

\[ = -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2} \log \sigma^2 + \mathbb{E}_q[\mu xz - \log(1 + \exp(xz))] \]
III for Bayesian Logistic Regression

\[ \mathcal{L}(\mu, \sigma^2) \]

\[ = \mathbb{E}_q[\log p(z) - \log q(z) + \log p(y|x, z)] \]

\[ = -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2} \log \sigma^2 + \mathbb{E}_q[\log p(y|x, z)] + C \]

\[ = -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2} \log \sigma^2 + \mathbb{E}_q[yxz - \log(1 + \exp(xz))] \]

\[ = -\frac{1}{2}(\mu^2 + \sigma^2) + \frac{1}{2} \log \sigma^2 + yx\mu - \mathbb{E}_q[\log(1 + \exp(xz))] \]
VI for Bayesian Logistic Regression

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

\[ = -\frac{1}{2} (\mu^2 + \sigma^2) + \frac{1}{2} \log \sigma^2 + \mathbb{E}_q[\log p(y|x, z)] + C \]

\[ = -\frac{1}{2} (\mu^2 + \sigma^2) + \frac{1}{2} \log \sigma^2 + \mathbb{E}_q[yx - \log(1 + \exp(xz))] \]

\[ = -\frac{1}{2} (\mu^2 + \sigma^2) + \frac{1}{2} \log \sigma^2 + yx\mu - \mathbb{E}_q[\log(1 + \exp(xz))] \]

We are stuck.

1. We cannot analytically take that expectation.

2. The expectation hides the objectives dependence on the variational parameters. This makes it hard to directly optimize.
Options?

- Derive a model specific bound:
  [Jordan and Jaakola; 1996], [Braun and McAuliffe; 2008], others

- More general approximations that require model-specific analysis:
  [Wang and Blei; 2013], [Knowles and Minka; 2011]
Nonconjugate Models

- Nonlinear Time series Models
- Deep Latent Gaussian Models
- Models with Attention (such as DRAW)
- Generalized Linear Models (Poisson Regression)
- Stochastic Volatility Models
- Discrete Choice Models
- Bayesian Neural Networks
- Deep Exponential Families (e.g. Sparse Gamma or Poisson)
- Correlated Topic Model (including nonparametric variants)
- Sigmoid Belief Network

*We need a solution that does not entail model specific work*
Black Box Variational Inference (BBVI)

- Sample from $q$.
- Form noisy gradients without model-specific computation
- Use stochastic optimization

ANY MODEL

REUSABLE VARIATIONAL FAMILIES

MASSIVE DATA

BLACK BOX VARIATIONAL INFERENCE

$p(\beta, z | x)$
The Problem in the Classical VI Recipe

\[ p(x, z) \]
\[ q(z; \nu) \]
\[ \int (\cdots) q(z; \nu) dz \]
\[ \nabla \nu \]
\[ q(z; \nu) \]
The New VI Recipe

Use stochastic optimization!
Computing Gradients of Expectations

- Define

\[ g(z, \nu) = \log p(x, z) - \log q(z; \nu) \]

- What is \( \nabla_\nu \mathcal{L} \)

\[
\nabla_\nu \mathcal{L} = \nabla_\nu \int q(z; \nu)g(z, \nu)dz
\]

\[
= \int \nabla_\nu q(z; \nu)g(z, \nu) + q(z; \nu)\nabla_\nu g(z, \nu)dz
\]

\[
= \int q(z; \nu)\nabla_\nu \log q(z; \nu)g(z, \nu) + q(z; \nu)\nabla_\nu g(z, \nu)dz
\]

\[
= \mathbb{E}_{q(z; \nu)}[\nabla_\nu \log q(z; \nu)g(z, \nu) + \nabla_\nu g(z, \nu)]
\]

Using \( \nabla_\nu \log q = \frac{\nabla_\nu q}{q} \)
Roadmap

- Score Function Gradients
- Pathwise Gradients
- Amortized Inference
Score Function Gradients of the ELBO
Score Function Estimator

Recall

\[ \nabla_{\nu} \mathcal{L} = \mathbb{E}_{q(z; \nu)}[\nabla_{\nu} \log q(z; \nu)g(z, \nu) + \nabla_{\nu} g(z, \nu)] \]

Simplify:

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

Gives the gradient:

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

Sometimes called likelihood ratio or REINFORCE gradients

[Glynn 1990; Williams, 1992; Wingate+ 2013; Ranganath+ 2014; Mnih+ 2014]
Noisy Unbiased Gradients

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

Noisy unbiased gradients with Monte Carlo!

\[
\frac{1}{S} \sum_{s=1}^{S} \nabla_\nu \log q(z_s; \nu)(\log p(x, z_s) - \log q(z_s; \nu)),
\]

where \( z_s \sim q(z; \nu) \)
Basic BBVI

**Algorithm 1: Basic Black Box Variational Inference**

**Input**: Model $\log p(x, z)$, Variational approximation $q(z; \nu)$

**Output**: Variational Parameters: $\nu$

```
while not converged do
    z[s] ∼ q // Draw S samples from q
    ρ = t-th value of a Robbins Monro sequence
    \nu = \nu + \rho \frac{1}{S} \sum_{s=1}^{S} \nabla_{\nu} \log q(z[s]; \nu)(\log p(x, z[s]) − \log q(z[s]; \nu))
    t = t + 1
end
```
The requirements for inference

The noisy gradient:

\[
\frac{1}{S} \sum_{s=1}^{S} \nabla_\nu \log q(z_s; \nu)(\log p(x, z_s) - \log q(z_s; \nu)),
\]

where \( z_s \sim q(z; \nu) \)

To compute the noisy gradient of the ELBO we need

- Sampling from \( q(z) \)
- Evaluating \( \nabla_\nu \log q(z; \nu) \)
- Evaluating \( \log p(x, z) \) and \( \log q(z) \)

There is no model specific work: black box criteria are satisfied
Black Box Variational Inference

- Sample from $q$.
- Form noisy gradients without model-specific computation.
- Use stochastic optimization.

$\mathbb{P}(\beta, z | x)$
Problem: Basic BBVI doesn’t work

Variance of the gradient can be a problem

\[ \text{Var}_{q(z; \nu)} = \mathbb{E}_{q(z; \nu)}[(\nabla_{\nu} \log q(z; \nu)(\log p(x, z) - \log q(z; \nu)) - \nabla_{\nu} L)^2]. \]

Intuition:
Sampling rare values can lead to large scores and thus high variance
Solution: Control Variates

Replace with $f$ with $\hat{f}$ where $\mathbb{E}[\hat{f}(z)] = \mathbb{E}[f(z)]$. General such class:

$$\hat{f}(z) \triangleq f(z) - a(h(z) - \mathbb{E}[h(z)])$$

- $h$ is a function of our choice
- $a$ is chosen to minimize the variance
- Good $h$ have high correlation with the original function $f$
Solution: Control Variates

Replace with \( f \) with \( \hat{f} \) where \( \mathbb{E}[\hat{f}(z)] = \mathbb{E}[f(z)] \). General such class:

\[
\hat{f}(z) \doteq f(z) - a(h(z) - \mathbb{E}[h(z)])
\]

- For variational inference we need functions with known \( q \) expectation
- Set \( h \) as \( \nabla_\nu \log q(z; \nu) \)
- Simple as \( \mathbb{E}_q[\nabla_\nu \log q(z; \nu)] = 0 \) for any \( q \)
Solution: Control Variates

Replace with $f$ with $\hat{f}$ where $\mathbb{E}[\hat{f}(z)] = \mathbb{E}[f(z)]$. General such class:

$$\hat{f}(z) \triangleq f(z) - a(h(z) - \mathbb{E}[h(z)])$$

Many of the other techniques from Monte Carlo can help:

- Importance Sampling, Quasi Monte Carlo, Rao-Blackwellization

[Ruiz+ 2016; Ranganath+2014; Titsias+2015; Mnih+2016]
Nonconjugate Models

- Nonlinear Time series Models
- Deep Latent Gaussian Models
- Models with Attention (such as DRAW)
- Generalized Linear Models (Poisson Regression)
- Stochastic Volatility Models
- Discrete Choice Models
- Bayesian Neural Networks
- Deep Exponential Families (e.g. Sparse Gamma or Poisson)
- Correlated Topic Model (including nonparametric variants)
- Sigmoid Belief Network

We can design models based on data rather than inference.
More Assumptions?

The current black box criteria

- Sampling from $q(z)$
- Evaluating $\nabla_\nu \log q(z; \nu)$
- Evaluating $\log p(x, z)$ and $\log q(z)$

Can we make additional assumptions that are not too restrictive?
Pathwise Gradients of the ELBO
Pathwise Estimator

Assume

1. $z = t(\epsilon, \nu)$ for $\epsilon \sim s(\epsilon)$ implies $z \sim q(z; \nu)$

   Example:

   $\epsilon \sim \text{Normal}(0, 1)$

   $z = \epsilon \sigma + \mu$

   $\rightarrow z \sim \text{Normal}(\mu, \sigma^2)$

2. $\log p(x, z)$ and $\log q(z)$ are differentiable with respect to $z$
Pathwise Estimator

Recall

\[ \nabla_{\nu} \mathcal{L} = \mathbb{E}_{q(z; \nu)}[\nabla_{\nu} \log q(z; \nu)g(z, \nu) + \nabla_{\nu}g(z, \nu)] \]

Rewrite using using \( z = t(\epsilon, \nu) \)

\[ \nabla_{\nu} \mathcal{L} = \mathbb{E}_{s(\epsilon)}[\nabla_{\nu} \log s(\epsilon)g(t(\epsilon, \nu), \nu) + \nabla_{\nu}g(t(\epsilon, \nu), \nu)] \]

To differentiate:

\[
\nabla \mathcal{L}(\nu) = \mathbb{E}_{s(\epsilon)}[\nabla_{\nu}g(t(\epsilon, \nu), \nu)] \\
= \mathbb{E}_{s(\epsilon)}[\nabla_{z}[\log p(x, z) - \log q(z; \nu)]\nabla_{\nu}t(\epsilon, \nu) - \nabla_{\nu}\log q(z; \nu)] \\
= \mathbb{E}_{s(\epsilon)}[\nabla_{z}[\log p(x, z) - \log q(z; \nu)]\nabla_{\nu}t(\epsilon, \nu)]
\]

This is also known as the reparameterization gradient.

[Glasserman 1991; Fu 2006; Kingma+ 2014; Rezende+ 2014; Titsias+ 2014]
is not the only way to compute Monte Carlo approximations of the gradient of the.
Black box variational inference (Kucukelbir et al., 2016) takes a different approach (Ranganath et al., 2014). The gradient estimator uses the gradient of the variational approximation and avoids using the gradient of the model. For example, the following estimator
\[
r \cdot \log q(x \mid \theta) - \log p(x; \theta) - \log Z
\]
and the gradient estimator in Equation (7) both lead to unbiased estimates of the exact gradient.

While is more general—it does not require the gradient of the model and thus applies to more settings—its gradients can suffer from high variance.

Figure 8: Comparison of gradient estimator variances. The gradient estimator exhibits lower variance than the estimator. Moreover, it does not require control variate variance reduction, which is not available in univariate situations.

Figure 8a shows the variance of both gradient estimators for a simple univariate model, where the posterior is a Gamma. We estimate the variance using ten thousand recalculations of the gradient across an increasing number of samples. The gradient has lower variance; in practice, a single sample succeeds. (See the experiments in Section 4.)

Figure 8b shows the same calculation for a 100-dimensional nonlinear regression model with likelihood \( \mathcal{N}(y_j \mid \tanh(x_j), \mathbf{\theta}) \) and a Gaussian prior on the regression coefficients \( \mathbf{\theta} \). Because this is a multivariate example, we also show the gradient with a variance reduction scheme using control variates described in Ranganath et al. (2014). In both cases, the gradients is computationally more efficient.

3.3 Sensitivity to Transformations uses a transformation \( T \) from the unconstrained space to the constrained space. We now study how the choice of this transformation affects the non-Gaussian posterior approximation in the original latent variable space.

Consider a posterior density in the Gamma family, with support over \( \mathbb{R}^+ \). Figure 9 shows three configurations of the Gamma, ranging from Gamma(1; 2), which places most of its mass close to \( \theta \leq 0 \), to Gamma(10; 10), which is centered at \( \theta = 1 \). Consider two transformations \( T_1 \) and \( T_2 \):
\[
T_1 \left( \log \theta \right)
\]
and
\[
T_2 \left( \log \theta \right) - \frac{1}{\theta};
\]

[Kucukelbir+ 2016]
Score Function Estimator vs. Pathwise Estimator

Score Function
- Differentiates the density $\nabla_\nu q(z; \nu)$
- Works for discrete and continuous models
- Works for large class of variational approximations
- Variance can be a big problem

Pathwise
- Differentiates the function $\nabla_z [\log p(x, z) - \log q(z; \nu)]$
- Requires differentiable models
- Requires variational approximation to have form $z = t(\epsilon, \nu)$
- Generally better behaved variance
Amortized Inference
Hierarchical Models

\[ p(\beta, z, x) = p(\beta) \prod_{i=1}^{n} p(z_i, x_i | \beta) \]
Mean Field Variational Approximation
SVI: Revisited

**Input:** data $x$, model $p(\beta, z, x)$.

Initialize $\lambda$ randomly. Set $\rho_t$ appropriately.

**repeat**

- 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
  \[
  \hat{\lambda} = \alpha + n \mathbb{E}_\phi \left[ t(Z_j, x_j) \right].
  \]
- Set global parameter
  \[
  \lambda = (1 - \rho_t)\lambda + \rho_t \hat{\lambda}.
  \]

**until** forever
SVI: The problem

**Input:** data \( x \), model \( p(\beta, z, x) \).

Initialize \( \lambda \) randomly. Set \( \rho_t \) appropriately.

**repeat**
- Sample \( j \sim \text{Unif}(1, \ldots, n) \).
- Set local parameter \( \phi \leftarrow \mathbb{E}_\lambda [\eta_\ell(\beta, x_j)] \).
- Set intermediate global parameter
  \[
  \hat{\lambda} = \alpha + n\mathbb{E}_\phi [t(Z_j, x_j)].
  \]
- Set global parameter
  \[
  \lambda = (1 - \rho_t)\lambda + \rho_t\hat{\lambda}.
  \]

**until** forever
- These expectations are no longer tractable
- Inner stochastic optimization needed for each data point.
**SVI: The problem**

**Input:** data $x$, model $p(\beta, z, x)$.

Initialize $\lambda$ randomly. Set $\rho_t$ appropriately.

**repeat**

1. Sample $j \sim \text{Unif}(1, \ldots, n)$.
2. Set local parameter $\phi \leftarrow \mathbb{E}_\lambda [\eta_\ell(\beta, x_j)]$.
3. Set intermediate global parameter
   \[
   \hat{\lambda} = \alpha + n\mathbb{E}_\phi [t(Z_j, x_j)].
   \]
4. Set global parameter
   \[
   \lambda = (1 - \rho_t)\lambda + \rho_t\hat{\lambda}.
   \]

**until** forever

**Idea:** Learn a mapping $f$ from $x_i$ to $\phi_i$
Amortizing Inference

ELBO:

$$\mathcal{L}(\lambda, \phi_{1...n}) = \mathbb{E}_q[\log p(\beta, z, x)] - \mathbb{E}_q \left[ \log q(\beta; \lambda) + \sum_{i=1}^{n} q(z_i; \phi_i) \right]$$

Amortizing the ELBO with inference network $f$:

$$\mathcal{L}(\lambda, \theta) = \mathbb{E}_q[\log p(\beta, z, x)] - \mathbb{E}_q \left[ \log q(\beta; \lambda) + \sum_{i=1}^{n} q(z_i | x_i; \phi_i = f(\theta(x_i))) \right]$$

[Dayan+ 1995; Heess+ 2013; Gershman+ 2014, many others]
Amortized SVI

**Input:** data \( x \), model \( p(\beta, z, x) \).

Initialize \( \lambda \) randomly. Set \( \rho_t \) appropriately.

**repeat**

Sample \( \beta \sim q(\beta; \lambda) \).

Sample \( j \sim \text{Unif}(1, \ldots, n) \).

Sample \( z_j \sim q(z_j | x_j; \phi_\theta(x_j)) \).

Compute stochastic gradients

\[
\hat{\nabla}_\lambda \mathcal{L} = \nabla_\lambda \log q(\beta; \lambda) (\log p(\beta) + n \log p(x_j, z_j | \beta) - \log q(\beta))
\]

\[
\hat{\nabla}_\theta \mathcal{L} = n \nabla_\theta \log q(z_j | x_j; \theta) (\log p(x_j, z_j | \beta) - \log q(z_j | x_k; \theta))
\]

**Update**

\[
\lambda = \lambda + \rho_t \hat{\nabla}_\lambda
\]

\[
\theta = \theta + \rho_t \hat{\nabla}_\theta.
\]

**until** forever
A computational-statistical tradeoff

- Amortized inference is faster, but admits a smaller class of approximations
- The size of the smaller class depends on the flexibility of $f$
Example: Variational Autoencoder (VAE)

\[ p(z) = \text{Normal}(0, 1) \]

\[ p(x|z) = \text{Normal}(\mu_\beta(z), \sigma_\beta^2(z)) \]

\( \mu \) and \( \sigma^2 \) are deep networks with parameters \( \beta \).

[Kingma+ 2014; Rezende+ 2014]
Example: Variational Autoencoder (VAE)

\[ q(z | x) = \text{Normal}(f_{\theta}^\mu(x), f_{\theta}^\sigma^2(x)) \]

All functions are deep networks
Example: Variational Autoencoder (VAE)
Rules of Thumb for a New Model

If \( \log p(x, z) \) is \( z \) differentiable

- Try out an approximation \( q \) that is reparameterizable

If \( \log p(x, z) \) is not \( z \) differentiable

- Use score function estimator with control variates
- Add further variance reductions based on experimental evidence
Rules of Thumb for a New Model

If \( \log p(x, z) \) is \( z \) differentiable
- Try out an approximation \( q \) that is reparameterizable

If \( \log p(x, z) \) is not \( z \) differentiable
- Use score function estimator with control variates
- Add further variance reductions based on experimental evidence

General Advice:
- Use coordinate specific learning rates (e.g. RMSProp, AdaGrad)
- Annealing + Tempering
- Consider parallelizing across samples from \( q \)
Software

Systems with Variational Inference:
- Venture, WebPPL, Edward, Stan, PyMC3, Infer.net, Anglican

Good for trying out lots of models

Differentiation Tools:
- Theano, Torch, Tensorflow, Stan Math, Caffe

Can lead to more scalable implementations of individual models
PART IV

Beyond the Mean Field
Review: Variational Bound and Optimisation

- Probabilistic modelling and variational inference.
- Scalable inference through stochastic optimisation.
- Black-box variational inference: Non-conjugate models, Monte Carlo gradient estimators and amortised inference.

*These advances empower us with new way to design more flexible approximate posterior distributions* $q(z)$
Mean-field Approximations

Key part of algorithm is the choice of approximate posterior $q(z)$.

$$\log p(x) \geq \mathcal{L} = \mathbb{E}_{q(z|x)}[\log p(x, z)] - \mathbb{E}_{q(z|x)}[\log q(z|x)]$$

- **Expected likelihood**
- **Entropy**
Mean-Field Posterior Approximations

Deep Latent Gaussian Model

Mean-field or fully-factorised posterior is usually not sufficient
Real-world Posterior Distributions

Deep Latent Gaussian Model

Latent variable model $p(x,z)$

$p(z)$

$z$

$x$

$p(x|z)$

Complex dependencies • Non-Gaussian distributions • Multiple modes
Families of Approximate Posteriors

Two high-level goals:

- Build richer approximate posterior distributions.

- Maintain computational efficiency and scalability.
Families of Approximate Posteriors

Two high-level goals:

- Build richer approximate posterior distributions.
- Maintain computational efficiency and scalability.

\[
q^* (z \mid x) \propto p(x \mid z)p(z)
\]

\[
q_{MF} (z \mid x) = \prod_k q(z_k)
\]
Families of Approximate Posteriors

Two high-level goals:

- Build richer approximate posterior distributions.
- Maintain computational efficiency and scalability.

Same as the problem of specifying a model of the data itself.
Structured Posterior Approximations

True Posterior

Structured Approx.

Fully-factorised

Structured mean field: Introduce any form of dependency to provide a richer approximating class of distributions.

[Saul and Jordan, 1996.]
Gaussian Approximate Posteriors

Use a correlated Gaussian:

\[ q_G(z; \nu) = \mathcal{N}(z|\mu, \Sigma) \]

Variational parameters \( \nu = \{\mu, \Sigma\} \)
Gaussian Approximate Posteriors

Use a correlated Gaussian:

\[ q_G(z; \nu) = \mathcal{N}(z | \mu, \Sigma) \]

Variational parameters \( \nu = \{\mu, \Sigma\} \)

**Covariance models:** Structure of covariance \( \Sigma \) describes dependency. Full covariance is richest, but computationally expensive.

\[
\text{Mean-field} \quad \text{diag}(\alpha_1, \ldots, \alpha_K) \\
+ \sum_j u_j u_j^T \\
\text{Rank-1} \quad \text{diag}(\alpha_1, \ldots, \alpha_K) \\
+ uu^T \\
\text{Rank-J} \quad \text{diag}(\alpha_1, \ldots, \alpha_K) \\
+ \sum_j u_j u_j^T \\
\text{Full} \quad \text{diag}(\alpha_1, \ldots, \alpha_K) \\
+ uu^T \\
+ \ldots + \\
\]

Test neg. marginal likelihood

<table>
<thead>
<tr>
<th>Rank-J</th>
<th>Diag</th>
<th>Wake-Sleep</th>
<th>FA</th>
</tr>
</thead>
<tbody>
<tr>
<td>84</td>
<td>88</td>
<td>92</td>
<td>96</td>
</tr>
<tr>
<td>104</td>
<td>100</td>
<td>99</td>
<td>98</td>
</tr>
</tbody>
</table>
Gaussian Approximate Posteriors

Use a correlated Gaussian:

\[ q_G(z; \nu) = \mathcal{N}(z|\mu, \Sigma) \]

Variational parameters \( \nu = \{\mu, \Sigma\} \)

**Covariance models:** Structure of covariance \( \Sigma \) describes dependency. Full covariance is richest, but computationally expensive.

\[
\text{Mean-field} + \text{diag}(\alpha_1, \ldots, \alpha_K) + uu^T + \sum_j u_j u_j^T + \ldots + \text{Full} + \text{diag}(\alpha_1, \ldots, \alpha_K) + UU^T
\]

Approximate posterior is always Gaussian.
Beyond Gaussian Approximations

Autoregressive distributions: Impose an ordering and non-linear dependency on all preceding variables.

\[ q_{AR}(\mathbf{z}; \nu) = \prod_{k} q_{k}(z_{k}|z_{<k}; \nu_{k}) \]
Beyond Gaussian Approximations

**Autoregressive distributions:** Impose an ordering and non-linear dependency on all preceding variables.

\[ q_{AR}(z; \nu) = \prod_k q_k(z_k | z_{<k}; \nu_k) \]

**Compare DLGMs:** Using Gaussian mean field (VAE) vs. auto-regressive posterior (DRAW) in fully-connected DLGMs on CIFAR10.

\[ \leq 86.6 \quad \leq 80.9 \]

[Gregor et al., 2015]
Beyond Gaussian Approximations

**Autoregressive distributions:** Impose an ordering and non-linear dependency on all preceding variables.

\[ q_{AR}(z; \nu) = \prod_k q_k(z_k | z_{<k}; \nu_k) \]

**Compare DLGMs:** Using Gaussian mean field (VAE) vs. auto-regressive posterior (DRAW) in fully-connected DLGMs on CIFAR10.

Joint-distribution non-Gaussian, although conditionals are.
More Structured Posteriors

Mixture model

\[ q_{mm}(z; \nu) = \sum_r \rho_r q_r(z_r|\nu_r) \]

Linking functions

\[ q_{lm}(z; \nu) = \left( \prod_k q_k(z_k|\nu_k) \right) C(z; \nu_{k+1}) \]

[Saul and Jordan, 1996, Tran et al., 2016]
More Structured Posteriors

Mixture model

\[
q_{mm}(\mathbf{z}; \nu) = \sum_{r} \rho_r q_r(\mathbf{z}_r | \nu_r)
\]

Linking functions

\[
q_{lm}(\mathbf{z}; \nu) = \left( \prod_{k} q_k(\mathbf{z}_k | \nu_k) \right) C(\mathbf{z}; \nu_{k+1})
\]

[Saul and Jordan, 1996, Tran et al., 2016]

Suggests a general way to improve posterior approximations:

Introduce additional variables that induce dependencies, but that remain tractable and efficient.
Designing Richer Posteriors

1. **Introduce new variables** \( \omega \) **that help to form a**
richer approximate posterior distribution.

\[
q(z; \nu) = \int q(z, \omega; \nu) d\omega
\]
Designing Richer Posteriors

1. **Introduce new variables** $\omega$ that help to form a richer approximate posterior distribution.

   \[ q(z; \nu) = \int q(z, \omega; \nu)d\omega \]

2. **Adapt bound** to compute entropy or a bound.

   \[ \log p(x) \geq \mathcal{L} = \mathbb{E}_{q(z|x)}[\log p(x, z)] - \mathbb{E}_{q(z|x)}[\log q(z|x)] \]

   - **Expected likelihood**
   - **Entropy**
Designing Richer Posteriors

1. **Introduce new variables** $\omega$ that help to form a richer approximate posterior distribution.

$$q(z; \nu) = \int q(z, \omega; \nu) d\omega$$

2. **Adapt bound** to compute entropy or a bound.

$$\log p(x) \geq \mathcal{L} = \mathbb{E}_{q(z|x)}[\log p(x, z)] - \mathbb{E}_{q(z|x)}[\log q(z|x)]$$

3. **Maintain computational efficiency**: linear in number of latent variables.
Designing Richer Posteriors

1. **Introduce new variables** $\omega$ that help to form a richer approximate posterior distribution.

   $$ q(z; \nu) = \int q(z, \omega; \nu) d\omega $$

2. **Adapt bound** to compute entropy or a bound.

   $$ \log p(x) \geq \mathcal{L} = \mathbb{E}_{q(z|x)}[\log p(x, z)] - \mathbb{E}_{q(z|x)}[\log q(z|x)] $$

   Expected likelihood $-$ Entropy

3. Maintain **computational efficiency**: linear in number of latent variables.

---

*Look at two different approaches*

- **Change-of-variables**: Normalising flows and invertible transforms.
- **Auxiliary variables**: Entropy bounds, Monte Carlo sampling.
Approximations using Change-of-variables

Exploit the rule for change of variables for random variables:

- Begin with an initial distribution $q_0(z_0|x)$.
- Apply a sequence of $K$ invertible functions $f_k$. 

\[ z_0 = f_K \ldots f_2 f_1 (z_0) \]

Distribution flows through a sequence of invertible transforms \cite{rezende2015variational}.
Approximations using Change-of-variables

Exploit the rule for change of variables for random variables:

- Begin with an initial distribution $q_0(z_0|x)$.
- Apply a sequence of $K$ invertible functions $f_k$.

Sampling and Entropy

\[ z_K = f_K \circ \ldots \circ f_2 \circ f_1(z_0) \]
\[ \log q_K(z_K) = \log q_0(z_0) - \sum_{k=1}^{K} \log \det \left| \frac{\partial f_k}{\partial z_k} \right| \]

Distribution flows through a sequence of invertible transforms

[Rezende and Mohamed, 2015]
Normalising Flows

<table>
<thead>
<tr>
<th>$q_0$</th>
<th>K=1</th>
<th>Planar</th>
<th>K=2</th>
<th>K=10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit Gaussian</td>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="image4.png" alt="Image" /></td>
</tr>
<tr>
<td>Uniform</td>
<td><img src="image5.png" alt="Image" /></td>
<td><img src="image6.png" alt="Image" /></td>
<td><img src="image7.png" alt="Image" /></td>
<td><img src="image8.png" alt="Image" /></td>
</tr>
</tbody>
</table>
Normalising Flows

K = 2

K = 8

K = 32
Choice of Transformation Function

\[
\mathcal{L} = \mathbb{E}_{q_0(z_0)}[\log p(x, z_K)] - \mathbb{E}_{q_0(z_0)}[\log q_0(z_0)] - \mathbb{E}_{q_0(z_0)} \left[ \sum_{k=1}^{K} \log \det \left| \frac{\partial f_k}{\partial z_k} \right| \right]
\]
Choice of Transformation Function

\[ \mathcal{L} = \mathbb{E}_{q_0(z)}[\log p(x, z_K)] - \mathbb{E}_{q_0(z_0)}[\log q_0(z_0)] - \mathbb{E}_{q_0(z_0)} \left[ \sum_{k=1}^{K} \log \det \left| \frac{\partial f_k}{\partial z_k} \right| \right] \]

- Begin with a fully-factorised Gaussian and improve by change of variables.
- Triangular Jacobians allow for computational efficiency.
**Choice of Transformation Function**

\[
\mathcal{L} = \mathbb{E}_{q_0(z_0)}[\log p(x, z_K)] - \mathbb{E}_{q_0(z_0)}[\log q_0(z_0)] - \mathbb{E}_{q_0(z_0)} \left[ \sum_{k=1}^{K} \log \det \left| \frac{\partial f_k}{\partial z_k} \right| \right]
\]

- Begin with a fully-factorised Gaussian and improve by change of variables.
- Triangular Jacobians allow for computational efficiency.

---

[Rezende and Mohamed, 2016; Dinh et al., 2016; Kingma et al., 2016]
Choice of Transformation Function

\[ \mathcal{L} = \mathbb{E}_{q_0(z_0)}[\log p(x, z_K)] - \mathbb{E}_{q_0(z_0)}[\log q_0(z_0)] - \mathbb{E}_{q_0(z_0)} \left[ \sum_{k=1}^{K} \log \det \left| \frac{\partial f_k}{\partial z_k} \right| \right] \]

- Begin with a fully-factorised Gaussian and improve by change of variables.
- Triangular Jacobians allow for computational efficiency.

**Planar Flow**

\[ z_k = z_{k-1} + uh(w^\top z_{k-1} + b) \]

**Real NVP**

\[ y_{1:d} = z_{k-1,1:d} \]
\[ y_{d+1:D} = t(z_{k-1,1:d}) + z_{d+1:D} \odot \exp(s(z_{k-1,1:d})) \]

**Inverse AR Flow**

\[ z_k = \frac{z_{k-1} - \mu_k(z_{<k}, x)}{\sigma_k(z_{<k}, x)} \]

[Rezende and Mohamed, 2016; Dinh et al., 2016; Kingma et al., 2016]

Linear time computation of the determinant and its gradient.
Modelling Improvements

VAE-type algorithms on the MNIST benchmark

<table>
<thead>
<tr>
<th>Method</th>
<th>Bits/dim</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAE</td>
<td>≤86.6</td>
</tr>
<tr>
<td>DRAW</td>
<td>≤80.9</td>
</tr>
<tr>
<td>IAF</td>
<td>≥79.1</td>
</tr>
</tbody>
</table>

Figure 4: Random samples from generative ResNet trained on the CIFAR-10 dataset of natural image patches.

Table 2: Our results with ResNet VAEs on CIFAR-10 images, compared to earlier results, in average number of bits per data dimension on the test set. The number for convolutional DRAW is an upper bound, while the ResNet VAE log-likelihood was estimated using importance sampling.

We empirically demonstrated the usefulness of inverse autoregressive flow for variational inference by training a novel deep architecture of variational auto-encoders. In experiments we demonstrated that autoregressive flow leads to significant performance gains compared to similar models with factorized Gaussian approximate posteriors, and we report the best results on CIFAR-10 for latent-variable models so far.

Acknowledgements

We thank Jascha Sohl-Dickstein, Karol Gregor, and many others at Google Deepmind for interesting discussions. We thank Harri Valpola for referring us to Gustavo Deco's very relevant pioneering work on a form of inverse autoregressive flow applied to nonlinear independent component analysis.

References


Modelling Improvements

VAE-type algorithms on the MNIST benchmark

<table>
<thead>
<tr>
<th>Method</th>
<th>Bits/dim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uniform distribution (van den Oord et al., 2016b)</td>
<td>8.00</td>
</tr>
<tr>
<td>Multivariate Gaussian (van den Oord et al., 2016b)</td>
<td>4.70</td>
</tr>
<tr>
<td>NICE (Dinh et al., 2014)</td>
<td>4.48</td>
</tr>
<tr>
<td>Deep GMMs (van den Oord and Schrauwen, 2014)</td>
<td>4.00</td>
</tr>
<tr>
<td>Real NVP (Dinh et al., 2016)</td>
<td>3.49</td>
</tr>
<tr>
<td>PixelRNN (van den Oord et al., 2016b)</td>
<td>3.00</td>
</tr>
<tr>
<td>Gated PixelCNN (van den Oord et al., 2016c)</td>
<td>3.03</td>
</tr>
</tbody>
</table>

Results with variationally trained latent-variable models:

<table>
<thead>
<tr>
<th>Method</th>
<th>Bits/dim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deep Diffusion (Sohl-Dickstein et al., 2015)</td>
<td>5.4</td>
</tr>
<tr>
<td>Convolutional DRAW (Gregor et al., 2016)</td>
<td>3.58 (Var. Bound)</td>
</tr>
<tr>
<td>Ours (ResNet V AE with IAF)</td>
<td>3.11</td>
</tr>
</tbody>
</table>

We empirically demonstrated the usefulness of inverse autoregressive flow for variational inference by training a novel deep architecture of variational auto-encoders. In experiments we demonstrated that autoregressive flow leads to significant performance gains compared to similar models with factorized Gaussian approximate posteriors, and we report the best results on CIFAR-10 for latent-variable models so far.

Samples generated from model on CIFAR10 images
Hierarchical Approximate Posteriors

We can use ‘latent variables’ $\omega$ to enrich the approximate posterior distribution, like we do for our density models.

$$q(z|x) = \int q(z|\omega, x)q(\omega|x)d\omega$$
Hierarchical Approximate Posteriors

We can use ‘latent variables’ $\omega$ to enrich the approximate posterior distribution, like we do for our density models.

$$q(z|x) = \int q(z|\omega, x)q(\omega|x)d\omega$$

- Use a **hierarchical model** for the approximate posterior.

- **Stochastic variables** $\omega$ rather than deterministic in the change-of-variables approach.

- **Both continuous and discrete** latent variables can be modelled.

[Ranganath et al., 2016]
Auxiliary-variable Methods

Modify the model to include $\omega = (z_0, \ldots, z_{K-1})$.

Latent variable model $p(x,z)$

$\begin{align*}
- p(z) \\
\to & \quad x \\
& \quad p(x|z)
\end{align*}$

Auxiliary latent variable model $p(x,z,\omega)$

$\begin{align*}
- p(z) \\
\to & \quad x \\
\to & \quad \omega \\
& \quad p(x|z) \\
& \quad r(\omega|x,z)
\end{align*}$

[Agakov and Barber, 2004; Maaløe et al., 2016]
Auxiliary-variable Methods

Modify the model to include $\omega = (z_0, \ldots, z_{K-1})$.

- **Auxiliary variables** leave the original model unchanged.
- They capture structure of correlated variables because they turn the posterior into a mixture of distributions $q(z|x, \omega)$.

[Agakov and Barber, 2004; Maaløe et al., 2016]
Auxiliary Variational Lower Bounds

Standard bound: \( \log p(x) \geq \mathcal{L} = E_{q(z|x)}[\log p(x, z)] - E_{q(z|x)}[\log q(z|x)] \)

- **Expected likelihood**
- **Entropy**

Auxiliary latent variable model \( p(x,z,\omega) \)

\[
p(z) \\
\downarrow \\
x \quad \omega \\
p(x|z) \quad r(\omega|x, z)
\]

Inference model \( q(z, \omega) \)

\[
q(z|x, \omega) \\
\downarrow \\
x \quad \omega \\
q(\omega|x)
\]

Salimans et al., 2015; Ranganath et al., 2016; Maaløe et al., 2016
Auxiliary Variational Lower Bounds

Standard bound: \( \log p(x) \geq \mathcal{L} = \mathbb{E}_{q(z|x)}[\log p(x, z)] - \mathbb{E}_{q(z|x)}[\log q(z|x)] \)

Expected likelihood \hspace{1cm} Entropy

\[ \mathbb{E}_{q(z|x)}[\log p(x, z)] - \mathbb{E}_{q(z|x)}[\log q(z|x)] \]

Auxiliary latent variable model \( p(x, z, \omega) \)

Inference model \( q(z, \omega) \)

Auxiliary variational bound: Bound the entropy for tractability.

\[ \log p(x) \geq \mathbb{E}_{q(\omega, z|x)}[\log p(x, z) + \log r(\omega|z, x)] - \mathbb{E}_{q(\omega, z|x)}[\log q(z, \omega|x)] \]
\[ \geq \mathcal{L} - \mathbb{E}_{q(z|x)}[\text{KL}[q(\omega|z, x) || r(\omega|z, x)]] \]

[Salimans et al., 2015; Ranganath et al., 2016; Maaløe et al., 2016]
Auxiliary Variational Methods

Choose an auxiliary prior $r(\omega|z,x)$ and auxiliary posterior $q(\omega|x,z)$
Auxiliary Variational Methods

Choose an auxiliary prior \( r(\omega|z,x) \) and auxiliary posterior \( q(\omega|x,z) \)

- Hamiltonian flow: \( r(\omega) = \mathcal{N}(\omega|0,M) \)
- Input-dependent Gaussian: \( r(\omega|x,z) \)
- Auto-regressive: \( r(\omega|x,z) = \prod_t r(\omega_t|f_\theta(\omega_{<t}, x)) \)

- \( q(\omega|x,z) \) can be a mixture model, normalising flow, Gaussian process.

\[ \begin{align*}
\text{Auxiliary latent variable model } & p(x,z,\omega) \\
\text{Inference model } & q(z,\omega) \\
\text{Hamiltonian flow: } & r(\omega) = \mathcal{N}(\omega|0,M) \\
\text{Input-dependent Gaussian: } & r(\omega|x,z) \\
\text{Auto-regressive: } & r(\omega|x,z) = \prod_t r(\omega_t|f_\theta(\omega_{<t}, x)) \\
\text{q(\omega|x,z) can be a mixture model, normalising flow, Gaussian process.}
\end{align*} \]
Auxiliary Variational Methods

Choose an auxiliary prior $r(\omega|z, x)$ and auxiliary posterior $q(\omega|x, z)$

- Hamiltonian flow: $r(\omega) = \mathcal{N}(\omega|0, M)$
- Input-dependent Gaussian: $r(\omega|x, z)$
- Auto-regressive: $r(\omega|x, z) = \prod_t r(\omega_t|f_\theta(\omega_{<t}, x))$

$q(\omega|x, z)$ can be a mixture model, normalising flow, Gaussian process.

Easy sampling, evaluation of bound and gradients.
Summary

True Posterior

Families of Posterior Approximations

Fully-factorised

Most Expressive

$\mathbf{q}^*(z|x) \propto p(x|z)p(z)$

Least Expressive

$q_{MF}(z|x) = \prod_k q(z_k)$
Choosing your Approximation

KNOWLEDGE & QUESTION

Make assumptions

DATA

Discover patterns

Predict & Explore

Criticize model

Revise

Figure S2: Population structure inferred from the TGP data set using the TeraStructure algorithm at three values for the number of populations K. The visualization of the ✓'s in the Figure shows patterns consistent with the major geographical regions. Some of the clusters identify a specific region (e.g. red for Africa) while others represent admixture between regions (e.g. green for Europeans and Central/South Americans). The presence of clusters that are shared between different regions demonstrates the more continuous nature of the structure. The new cluster from K = 7 to K = 8 matches structure differentiating between American groups. For K = 9, the new cluster is unpopulated.
Summary
Variational Inference:
Foundations and Modern Methods

VI approximates difficult quantities from complex models.

With **stochastic optimization** we can

- scale up VI to massive data
- enable VI on a wide class of difficult models
- enable VI with elaborate and flexible families of approximations
Introductory Variational Inference


Applications of Variational Inference


Bibliography

Monte Carlo Gradient Estimation

- Pierre L’Ecuyer, Note: On the interchange of derivative and expectation for likelihood ratio derivative estimators, Management Science, 1995
- Peter W Glynn, Likelihood ratio gradient estimation for stochastic systems, Communications of the ACM, 1990
- Michael C Fu, Gradient estimation, Handbooks in operations research and management science, 2006
- Ronald J Williams, Simple statistical gradient-following algorithms for connectionist reinforcement learning, Machine learning, 1992
- Paul Glasserman, Monte Carlo methods in financial engineering, 2003
- Michael C Fu, Gradient estimation, Handbooks in operations research and management science, 2006
Monte Carlo Gradient Estimation (cont.)

Bibliography

**Amortized Inference**


Bibliography

Structured Mean Field


Bibliography

Change-of-variables and Normalising Flows

Auxiliary Variational Methods

Related Variational Objectives

Discrete Latent Variable Models and Posterior Approximations


