Statistical inference in theoretical models of cognition

Sean, Agostina, Alex, Chunyu, Francesca, Srdjan, Carlos, Ken, and John

1 Abstract

Theoretical neuroscientists often design circuit models of neural activity with the hopes of producing some emergent property observed in data. However, the standard inference toolkit is designed to condition on data points collected in an experiment, rather than the abstract notion of an emergent property. We introduce a novel machine learning methodology called degenerate solution networks (DSNs), which learn a distribution of generative model parameterizations that produces the emergent property of interest, and is otherwise as random as possible. We use DSNs to advance theoretical understanding of the stomatogastric ganglion (STG), primary visual cortex (V1), superior colliculus (SC), and low rank recurrent neural networks (RNNs). As models continue to increase in complexity and become less tractable in terms of conventional analytic and theoretical approaches, DSNs will be a useful tool for theorists to interrogate their models

2 Introduction

Developing a theory of neural computation requires a parameterized model of the brain area(s) executing the computation, a mathematical definition of the emergent properties of the model signifying the computation, and finally, a characterization of the model parameters that produce these emergent properties. Since the advent of theoretical neuroscience, parameterized models of neural systems have become ubiquitous in neuroscience [1]. In idealized practice of theory, scientists analytically derive parametric solutions to their model that produce the emergent property of interest. Historical examples of this idealized approach include derivations of memory capacity in associative neural networks [2], chaos and autocorrelation timescales in random neural networks [3], and the paradoxical effect in E/I networks [5]. Unfortunately, as biological realism is introduced into neural circuit models, theory through analytic derivation becomes infeasible.

Theorists design neural circuit models in the context of misaligned, competing incentives. Models are kept simplistic, and given structured assumptions (e.g. symmetry, gaussianity) conducive of reaching the idealized analytic derivations of emergent properties. On the other hand, complexity is introduced to more closely represent biological reality at the cost analytic tractability. When biological realism is the focus of a study, standard practice is to examine simulated activity from the model [4] (cite a bunch here). Visualizations and regression are used to provide insights on how parameters govern system activity, however theorists strive for a more formalized understanding of these complex systems.

A viable, formalized alternative to standard practices can come from Bayesian inference. Research in neural data analysis using modern advancements like convex optimization theory and deep learning has resulted in careful posterior inference procedures (the modern "inference engine") on a host of valuable phenomenlogical models (cite all from dsn talk) (Fig. 1A, top row). This is juxtaposed with the standard practices of the theoretical neuroscience community, which focuses on carefully modeling neural circuits, and have rather practical, ad hoc methods of inferring the effect of parameters on model behavior (Fig. 1A, bottom row). Where theorists are careful in modeling, statisticians are practical in designing phenomenlogical generative models in which statistical inference is possible.

Classically, Bayesian inference is designed to condition on an experimentally observed set of data points. There is a common misconception proliferated throughout neurosciece that contributes to the inadequate degree of overlap between neural data analysis and theoretical neuroscience research. Contrary to the common narrative, theoreticians rarely attempt to directly reproduce experimental data. They work with abstracted mathematical definitions of the requisite neural circuit properties for computation (albeit, these are often motivated by experimental findings). Thus, theoretists, who want to understand their complex models, really want to condition their interpretable neural circuit models on mathematically defined emergent properties of computation, *not* a data set.

Here, we use the state of the art Bayesian inference engines, to learn formal proababilistic relationships between carefully modeled neural circuit parameters and their emergent properties. We present a novel machine learning method, degenerate solution networks (DSNs), which learn distributions of theoretical model parameterizations that produce the emerget properties of interest. In this study, we use DSNs to provide novel theoretical insights using biological models of the stomatogastric ganglion (STG), primary visual cortex (V1), superior colliculus (SC), and low-rank recurrent neural networks (RNNs) (Fig. 1B).

3 Results

3.1 Degenerate solution networks

In order to translate the achievements of research in neural data analysis to advancements in theoretical neuroscience, there are two key steps that must be taken. First, we must be able to learn parameter distributions on the biologically realistic models designed by theorists. While phenomenological models have been used to identify important structure in neural data sets, theorists are left without formal inference methods for their biological models. Second, we must be able to contextualize these parameter distributions by the emergent properties of computation relevant to theorists. Bayesian data scientists will argue that the scientific objective of modeling is to optimally explain scientific observations. However, when working in a creative, exploratory modeling setting, theorists actually want to flexibly condition their models on various mathematically defined statistics, and various levels of these statistics.

We introduce a novel machine learning method, degenerate solution networks (DSNs), which make these necessary steps for bridging methodology in these disconnected subfields of computational neuroscience. DSNs combine ideas from maximum entropy flow networks (cite Gabe) and likelihood-free variational inference (cite Dustin) to learn a deep probability distribution of theoretical model parameterizations z that produce the emergent properties of interest $T(x)$ (see Appendix). Deep probability distributions, a central component of the modern inference engine, are used to learn flexible approximations to distributions $q_\theta(z)$ by optimizing the weights and biases θ of a deep neural network. Once trained, these deep probability distributions emit fast samples via a sequence of deterministic transformations $z = f_l(f_{l-1}(...(f_1(\omega)))$ of a simple random variable $\omega \sim \mathcal{N}(0, I)$. DSNs are deep probability distributions of theoretical model parameters, which are optimized to be maximally random (maximum entropy) while producing the specified value of emergent properties in expectation:

$$\begin{aligned} q_\theta^*(z) &= \underset{q_\theta \in Q}{\text{argmax}}\, H(q_\theta(z)) \\ \text{s.t. } & E_{z \sim q_\theta}\left[E_{x \sim p(x|z)}\left[T(x)\right]\right] = \mu \end{aligned} \tag{1}$$

For example, consider a model of the stomatogastric ganglion (STG) circuit of the crustacean, which is involved in generating multiple rhythmic muscle activation patterns for digestion (Fig. 2B, cite Gutierrez). The two fast neurons ($f1$ and $f2$) mutually inhibit one another, and oscillate at a faster frequency than the slow neurons ($s1$ and $s2$), which also mutually inhibit each other.

Electrical and synaptic conductance parameters g_{el} and g_{synA} govern the coupling of the hub neuron to the fast or slow oscillator. There is an interesting regime of circuit activity, in which all neurons phase lock and oscillate at an intermediate synced frequency network syncing. For our choices of STG as model and network syncing as emergent property, we use a DSN to learn a distribution on STG conductance parameters that produces network syncing. Once a DSN is trained, we can immediately identify dimensions of parameter space that are sensitive (or degenerate) with respect to network syncing by evaluating the Hessian of the DSN distribution at the mode (Fig. S1). So long as we have a differentiable theoretical model simulator $x \sim p(x \mid z)$ and emergent property statistic calculation $T(x)$, we can train the DSN to maximize entropy of the deep probability distribution while satisfying the emergent property constraints. An equivalent conceptualization is that DSNs execute Bayesian inference conditioned on emergent property values (see Appendix). DSNs enable a new class of exploratory analyses of neural circuit models, which readily produce novel insights.

3.2 Exploratory analysis of a theoretical model

Dynamical models with two populations (excitatory (E) and inhibitory (I) neurons) of visual processing have been used to reproduce a host of experimentally documented phenomena in V1. When an inhibition stabilized network (ISN, the I population stabilizes an otherwise unstable E population), these models exhibit the paradoxical effect [5], selective amplification [6], surround suppression [7], and sensory integrative properties [8]. Since I neurons mostly fall into one of three classes (parvalbumin (P)-, somatostatin (S)-, and vasointestinal peptide (V)-expressing neurons) [9, 10], theorists look to extend these dynamical models to four populations [11]. A current challenge in theoretical neuroscience is understanding the distributed role of inhibition stabilization across these subtypes.

These four populations exhibit neuron-type specific connectivity (Fig. 1A) [12], in which some populations do not project to others. Since S and V are the only populations that mutually inhibit each other, a popular conceptualization is that S and V have winner-take-all dynamics. In fact, evidence in mice suggests that V silences S when presented with large stimuli, and S silences V for small stimuli [13]. Here, we use DSNs to understand the possible sources of inhibition stabilization in this V1 model, when either S or V is inactive. The behavior of the DSN distributed models are constrained to produce two things: 1.) a mean-zero distribution of ISN coefficients $\gamma(W) = 1 - f^{'}(f^{-1}(r_E(W)))W_{EE}$ with some variance, and 2.) α-population silencing $r_\alpha(W) = 0$, for $\alpha \in \{S, V\}$ (Fig. 1B). When $\gamma < 0$ the network is ISN, and not ISN otherwise. Constraining the DSN behavior to a zero-mean distribution of ISN coefficients gives us samples of both ISN and non-ISN networks, optimizied to have greatest variety of stabilization motifs.

When optimized to produce a variety of stabilization motifs, there are informative differences between S-silenced and V-silenced DSN posteriors. The marginal posteriors for each weight matrix element (W_{EE} is fixed to 1.0, and W_{*E} is one parameter), are visualized by their location in the dynamics matrix (Fig. 1C). Low-variance marginals, like $q_\theta(W_{PP} \mid \mathcal{B}_{S=0})$, $q_\theta(W_{VP} \mid \mathcal{B}_{S=0})$, and $q_\theta(W_{SV} \mid \mathcal{B}_{S=0})$, indicate that either the $\gamma(W)$, S-silencing, or both are sensitive to changes in such parameters. Whereas, $q_\theta(W_{PP} \mid \mathcal{B}_{V=0})$ and $q_\theta(W_{PS} \mid \mathcal{B}_{V=0})$ have high variance indicating degeneracy with respect to $\gamma(W)$ and V-silencing.

As with the STG circuit, we evaluate the Hessian of the DSN posterior at $\gamma(W) = 0$, and visualize the eigendecompositions ordered by eigenvalues (Fig. 1D). In accordance with the marginals, W_{PP}, W_{VP}, and W_{SV} are pronounced in the Hessian eigenvectors with the greatest magnitude eigenvalues. The low magnitude eigenvalues indicate degenerate dimensions of the weight matrix w.r.t. $\mathcal{B}_{\alpha=0}$.

Having a distribution optimized to be as random as possible allows us to see the variety of way in which populations are silenced across ISN regimes. We show how E- and V-input to a silenced S population and P- and S- input to a silenced V population change with ISN regime (Fig. 1E).

3.3 Identifying sufficient mechanisms of task learning

In behavioral neuroscience, model organisms are studied while performing tasks in order to investigate the underlying neural computation. As the animal is trained, its accuracy improves via a learning process in the brain. A central challenge for theoreticians is to describe sufficient changes of model parameters that drive task performance, since such changes may indicate how the learning brain adapts. We show that when a data-motivated, restricted dynamical model is proposed, we can use DSNs to clearly identify sufficient changes in network connectivity for task learning.

In a rapid task switching experiment, where rats are to respond right (R) or left (L) to the side of a light stimulus in the pro (P) task, and oppositely in the anti (A) task predicated by an auditory cue, neural recordings exhibited two population of neurons in each hemisphere of superior colliculus (SC) that simultaneously represented both task condition and motor response: the pro/contra and anti/ipsi neurons [14]. We trained five DSNs on a 4-neuron model of SC proposed by Duan et al. (Fig. 1A, see Methods), constraining the task performance in both the pro and anti tasks to an accuracy p with some allowed variance fixed across chosen p. We constrained the network to emit Bernoulli responses (approximately 0.0 or 1.0 on a given trial), and have winner-take-all behavior between the pro neuron populations of each hemisphere. Altogether, these DSNs, optimized to be expansive and unbiased, learned posteriors of SC model weight matrix parameters, $z = W$, conditioned on different regimes of rapid task switching performance denoted by $\mathcal{B}(p)$ (see Methods).

A convenient property of this dynamical model is that the weight matrix always has the same Schur modes (Fig. 1B), albeit variable eigenvalues for each mode. These Schur modes have intuitive roles with respect to processing in this task, and are accordingly named the *all*, *side*, *task*, and *diag* modes. The degree of amplification of each processing mode in a task performance regime can be examined via $q(S_\alpha(z) \mid \mathcal{B}(p))$, where $S_\alpha(z)$, $\alpha \in \{\text{all}, \text{side}, \text{task}, \text{diag}\}$, is the eigenvalue of the matching Schur mode (Fig. 1C).

As learning progresses, the task mode is increasingly amplified, indicating the criticality of a distributed task representation at the time of stimulus presentation, (Fig. 1D, purple). Stepping from task-naive 50% networks to task-performing 60% networks, there is a switch from amplified (pos. eigs.) to suppressed side mode (neg. eigs.) (Fig. 1D, orange). Side mode suppression is also found in the regimes of greater accuracy, revealing the importance of side mode suppression in allowing a distributed task representation to exist. Across all learning regimes, the diag mode is amplified (Fig. 1D, cyan), and the all mode is suppressed (Fig. 1D, black), which can be seen as signatures of Bernoulli winner-take-all networks. We can conclude that side mode suppression allows rapid task switching, and that greater task-mode representation increases accuracy.

3.4 Conditioning on computation with interpretable models of RNNs

4 Discussion

Still need to write this.

References

[1] Larry F Abbott. Theoretical neuroscience rising. *Neuron*, 60(3):489–495, 2008.

[2] John J Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the national academy of sciences*, 81(10):3088–3092, 1984.

[3] Haim Sompolinsky, Andrea Crisanti, and Hans-Jurgen Sommers. Chaos in random neural networks. *Physical review letters*, 61(3):259, 1988.

[4] Gabrielle J Gutierrez, Timothy OLeary, and Eve Marder. Multiple mechanisms switch an electrically coupled, synaptically inhibited neuron between competing rhythmic oscillators. *Neuron*, 77(5):845–858, 2013.

[5] Misha V Tsodyks, William E Skaggs, Terrence J Sejnowski, and Bruce L McNaughton. Paradoxical effects of external modulation of inhibitory interneurons. *Journal of neuroscience*, 17(11):4382–4388, 1997.

[6] Brendan K Murphy and Kenneth D Miller. Balanced amplification: a new mechanism of selective amplification of neural activity patterns. *Neuron*, 61(4):635–648, 2009.

[7] Hirofumi Ozeki, Ian M Finn, Evan S Schaffer, Kenneth D Miller, and David Ferster. Inhibitory stabilization of the cortical network underlies visual surround suppression. *Neuron*, 62(4):578–592, 2009.

[8] Daniel B Rubin, Stephen D Van Hooser, and Kenneth D Miller. The stabilized supralinear network: a unifying circuit motif underlying multi-input integration in sensory cortex. *Neuron*, 85(2):402–417, 2015.

[9] Henry Markram, Maria Toledo-Rodriguez, Yun Wang, Anirudh Gupta, Gilad Silberberg, and Caizhi Wu. Interneurons of the neocortical inhibitory system. *Nature reviews neuroscience*, 5(10):793, 2004.

[10] Bernardo Rudy, Gordon Fishell, SooHyun Lee, and Jens Hjerling-Leffler. Three groups of interneurons account for nearly 100% of neocortical gabaergic neurons. *Developmental neurobiology*, 71(1):45–61, 2011.

[11] Ashok Litwin-Kumar, Robert Rosenbaum, and Brent Doiron. Inhibitory stabilization and visual coding in cortical circuits with multiple interneuron subtypes. *Journal of neurophysiology*, 115(3):1399–1409, 2016.

[12] Carsten K Pfeffer, Mingshan Xue, Miao He, Z Josh Huang, and Massimo Scanziani. Inhibition of inhibition in visual cortex: the logic of connections between molecularly distinct interneurons. *Nature neuroscience*, 16(8):1068, 2013.

[13] Mario Dipoppa, Adam Ranson, Michael Krumin, Marius Pachitariu, Matteo Carandini, and Kenneth D Harris. Vision and locomotion shape the interactions between neuron types in mouse visual cortex. *Neuron*, 98(3):602–615, 2018.

[14] Chunyu A Duan, Marino Pagan, Alex T Piet, Charles D Kopec, Athena Akrami, Alexander J Riordan, Jeffrey C Erlich, and Carlos D Brody. Collicular circuits for flexible sensorimotor routing. *bioRxiv*, page 245613, 2018.

A Methods

A.1 Degenerate solution networks (DSNs)

DSNs learn distributions of theoretical model parameters that produce emergent properties of interest. They combine ideas from likelihood-free variational inference (cite Dustin) and maximum entropy flow networks (cite Gabe). A maximum entropy flow network is used as a deep probability distribution for the parameters, while these samples are passed through a differentiable model/dynamics simulator, which can lack a tractable likelihood function.

Consider model parameterization z and data x generated from some theoretical model simulator represented as $p(x \mid z)$, which may be deterministic or stochastic. Neural circuit models usually have known sampling procedures for simulating activity given a circuit parameterization, yet often lack an explicit likelihood function for the neural activity due to having nonlinear dynamics. DSNs learn a distribution on parameters z, that yields a behavior of interest $\mathcal{B}$,

$$\mathcal{B} : E_{z \sim q_\theta} \left[E_{x \sim p(x|z)} \left[T(x) \right] \right] = \mu \tag{2}$$

by making an approximation $q_\theta(z)$ to $p(z \mid \mathcal{B})$. So, over the degenerate solution distribution $q_\theta(z)$ of model $p(x \mid z)$ for behavior $\mathcal{B}$, the emergent propertiesT(x) are constrained in expectation to μ.

In deep probability distributions, a simple random variable $w \sim p_0$ is mapped deterministically via a function f_θ parameterized by a neural network to the support of the distribution of interest where $z = f_\theta(\omega) = f_l(..f_1(\omega))$. Given a theoretical model $p(x \mid z)$ and some behavior of interest $\mathcal{B}$, DSNs are trained by optimizing the neural network parameters θ to find the optimal approximation q_θ^*

within the deep variational family Q to

$$p(z \mid \mathcal{B})$$

.

In most settings (especially those relevant to theoretical neuroscience) the likelihood of the behavior with respect to the model parameters $p(T(x) \mid z)$ is unknown or intractable, requiring an alternative to stochastic gradient variational bayes (cite Kingma Welling 2013) or black box variational inference (cite Ranganeth 2014). These types of methods called likelihood-free variational inference (LFVI, cite Tran) skate around the intractable likelihood function in situations where there is a differentiable simulator. Akin to LFVI, DSNs are optimized with the following objective for a given generative model and statistical constraints on its produced activity:

$$\begin{aligned} q_\theta^*(z) &= \underset{q_\theta \in Q}{\operatorname{argmax}}\, H(q_\theta(z)) \\ \text{s.t. } & E_{z \sim q_\theta}\left[E_{x \sim p(x|z)}\left[T(x)\right]\right] = \mu \end{aligned} \tag{3}$$

A.1.1 Example: 2D LDS

To gain intuition for DSNs, consider two-dimensional linear dynamical systems, $\tau \dot{x} = Ax$ with

$$A = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}$$

that produce a band of oscillations. To train a DSN to learn the maximally entropic distribution of real entries of the dynamics matrix $z = [a_1, a_2, a_3, a_4]$, $(\tau = 1,)$ that yield a band of oscillations, $T(x)$ is chosen to contain the first- and second-moments of the oscillatory frequency ω and the the primary growth/decay factor c of the oscillating system. To learn the distribution of real entries of A that yield a c around zero with variance 1.0, and oscillations at 1 Hz with variance 1.0, the behavior of DSN would be constrained to:

$$\mu = E \begin{bmatrix} c \\ \omega \\ c^2 \\ \omega^2 \end{bmatrix} = \begin{bmatrix} 0.0 \\ 1.0 \\ 1.0 \\ 1.025 \end{bmatrix} \tag{4}$$

We could simuilate system activity x from z for some finite number of time steps, and estimate ω by e.g. taking the peak of the Discrete Fourier series. Instead, the sufficient statistics for this oscillating behavior are computed through a closed form function $g(z)$ by taking the eigendecomposition of the dynamics matrix

$$g(z) = E_{x\sim p(x|z)}\left[T(x)\right] = \begin{bmatrix} \text{real}(\lambda_1) \\ \frac{\text{imag}(\lambda_1)}{2\pi} \\ \text{real}(\lambda_1)^2 \\ (\frac{\text{imag}(\lambda_1)}{2\pi})^2 \end{bmatrix} \tag{5}$$

$$\lambda = \frac{(\frac{a_1+a_4}{\tau}) \pm \sqrt{(\frac{a_1+a_4}{\tau})^2 + 4(\frac{a_2a_3-a_1a_4}{\tau})}}{2} \tag{6}$$

where λ_1 is the eigenvalue of $\frac{1}{\tau}A$ with greatest real part. Even though $E_{x\sim p(x|z)}\left[T(x)\right]$ is calculable directly via $g(z)$, we cannot derive the distribution q_θ^*, since the backward mapping fromt the mean parameters μ to the natural parameters η of his exponential family is unknown. Instead, we can train a DSN to learn the degenerate linear system parameterization (Fig. S2B). Even this relatively simple system has nontrivial (though intuitively sensible) structure in the parameter distribution.

A.1.2 Augmented Lagrangian optimization

To optimize $q_\theta(z)$ in equation 1, the constrained optimization is performed using the augmented Lagrangian method. The following objective is minimized:

$$L(\theta; \lambda, c) = -H(q_\theta) + \lambda^\top R(\theta) + \frac{c}{2}||R(\theta)||^2 \tag{7}$$

where $R(\theta) = E_{z\sim q_\theta}\left[E_{x\sim p(x|z)}\left[T(x) - \mu\right]\right]$, $\lambda \in \mathcal{R}^m$ are the Lagrange multipliers and c is the penalty coefficient. For a fixed (λ, c), θ is optimized with stochastic gradient descent. A low value of c is used initially, and increased during each augmented Lagrangian epoch. Similarly, λ is tuned each epoch based on the constraint violations. For the linear 2-dimensional system (Fig. S2C) optimization hyperparameters are initialized to $c_1 = 10^{-4}$ and $\lambda_1 = 0$. The penalty coefficient is updated based on a hypothesis test regarding the reduction in constraint violation. The p-value of $E[||R(\theta_{k+1})||] > \gamma E[||R(\theta_k)||]$ is computed, and c_{k+1} is updated to βc_k with probability $1-p$. Throughout the project, $\beta = 4.0$ and $\gamma = 0.25$ is used. The other update rule is $\lambda_{k+1} = \lambda_k + c_k \frac{1}{n}\sum_{i=1}^n (T(x^{(i)}) - \mu)$. Each augmented Lagrangian epoch runs for 5,000 iterations. We consider the optimization to have converged when a null hypothesis test of constraint violations being zero is accepted for all constraints at a significance threshold 0.05. This is the dotted line on the plots below depicting the optimization cutoff of the DSN optimization for the 2-dimensional linear system. If the optimization is left to continue running, entropy may decrease, and structural pathologies in the distribution may be introduced.

The intention is that c and λ start at values encouraging entropic growth early in optimization. Then, as they increase in magnitude with each training epoch, the constraint satisfaction terms are increasingly weighted, resulting in a decrease in entropy. Rather than using a naive initialization, before the DSN training, we otpimize the density network parameters to generate samples of an isotropic gaussian of a selected variance, such as 1.0 for the 2D LDS example. This provides a convenient starting point, whose level of entropy is controlled by the user.

A.1.3 Normalizing flows

Since we are optimizing parameters θ of our deep probability distribution with respect to the entropy, we will need to take gradients with respect to the log-density of samples from the DSN.

$$H(q_\theta(z)) = \int -q_\theta(z) \log(q_\theta(z)) dz = E_{z \sim q_\theta} \left[-\log(q_\theta(z)) \right] = E_{w \sim q_0} \left[-\log(q_\theta(f_\theta(w))) \right] \tag{8}$$

$$\nabla_\theta H(q_\theta(z)) = E_{w \sim q_0} \left[-\nabla_\theta \log(q_\theta(f_\theta(w))) \right] \tag{9}$$

Deep probability models typically consist of several layers of fully connected neural networks. When each neural network layer is restricted to be a bijective function, the sample density can be calculated using the change of variables formula at each layer of the network. For $z' = f(z)$,

$$q(z') = q(f^{-1}(z')) \left| \det \frac{\partial f^{-1}(z')}{\partial z'} \right| = q(z) \left| \det \frac{\partial f(z)}{\partial z} \right|^{-1} \tag{10}$$

However, this computation has cubic complexity in dimensionality for fully connected layers. By restricting our layers to normalizimg flows (cite Rez and Mo) – bijective functions with fast log determinant jacobian computations, we can tractably optimize deep generative models with objectives that are a function of sample density, like entropy. Most of our analyses use real NVP (cite Dinh), which have proven effective in our architecture searches, and have the advantageous features of fast sampling and fast density evaluation. For more background on normalizing flows, see (cite kobyzev).

A.1.4 Related work

Working on this.

- Talk about ABC and synthetic likelihoods.
- Obviously there's MCMC
- Talk about Leukmann/Makarios approaches.
- Talk about likelihood ratio estimation.
- introduce Dustin/Dave's stuff.
- Talk about how APT uses this same idea.
- Set up the DSNs as VI in an EF derivation.

A.1.5 DSNs as VI in an exponential family conditioned on μ

I'll clean this up.

Exponential family for the posterior distribution.

$$\begin{aligned} p(z \mid x) = h(z) \exp\left(\eta(x)^\top T(z) - A(\eta(x))\right) &= \exp\left(\begin{bmatrix}\eta(x) \\ 1\end{bmatrix}^\top \begin{bmatrix}T(z) \\ h(z)\end{bmatrix} - A(\eta(x))\right) \\ &= \exp\left(\tilde{\eta(x)}^\top \tilde{T}(z) - A(\eta(x))\right) \end{aligned} \tag{11}$$

Doing VI looks like this.

$$q_\theta^* = \underset{q_\theta \in Q}{\operatorname{argmin}} \, KL(q_\theta \mid\mid p(z \mid x)) \tag{12}$$

$$KL(q_\theta \mid\mid p(z \mid x)) = E_{z \sim q_\theta}\left[\log(q_\theta(z))\right] - E_{z \sim q_\theta}\left[\log(p(z \mid x))\right] \tag{13}$$

$$= -H(q_\theta) - E_{z \sim q_\theta}\left[\tilde{\eta}(x)^\top \tilde{T}(z) - A(\eta(x))\right] \tag{14}$$

$$\underset{q_\theta \in Q}{\operatorname{argmin}} \, KL(q_\theta \mid\mid p(z \mid x)) = \underset{q_\theta \in Q}{\operatorname{argmin}} -H(q_\theta) - E_{z \sim q_\theta}\left[\tilde{\eta}(x)^\top \tilde{T}(z)\right] \tag{15}$$

$$= \underset{q_\theta \in Q}{\operatorname{argmin}} -H(q_\theta) - E_{z \sim q_\theta}\left[\tilde{\eta}(x)^\top \left(\tilde{T}(z) - \mu\right)\right] + \tilde{\eta(x)}^\top \mu \tag{16}$$

$$= \underset{q_\theta \in Q}{\operatorname{argmin}} -H(q_\theta) - E_{z \sim q_\theta}\left[\tilde{\eta}(x)^\top \left(\tilde{T}(z) - \mu\right)\right] \tag{17}$$

With DSNs we're doing this:

$$q_\theta^*(z)y = \underset{q_\theta \in Q}{\operatorname{argmax}} \, H(q_\theta(z)), \text{ s.t. } E_{z \sim q_\theta}\left[E_{x \sim p(x|z)}\left[T(x)\right]\right] = \mu \tag{18}$$

We use an augmented lagrangian objective:

$$q_\theta^* = \underset{q_\theta \in Q}{\operatorname{argmin}} -H(q_\theta) + \lambda^\top \left(E_{z \sim q_\theta}\left[T(z)\right] - \mu\right) \tag{19}$$

- λ should converge to $\tilde{\eta}$
- Really $\tilde{\eta}(x) = \tilde{\eta}(\mu)$, the backward mapping.
- Since deterministic, we should replace $p(z \mid x)$ with $p(z \mid \mu)$.