Statistical Machine Learning Methods for High-dimensional Neural Population Data Analysis

Yuanjun Gao

Department of Statistics
Columbia University

Overview

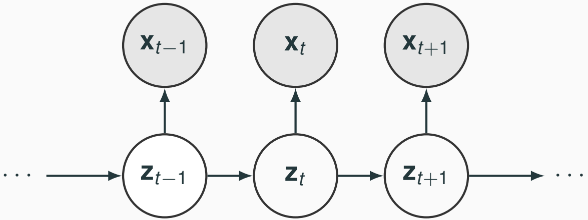
TODO: add a diagram for statistical criticizing

TODO: add a page for spike train

Table of Contents

Table of Contents

State space models

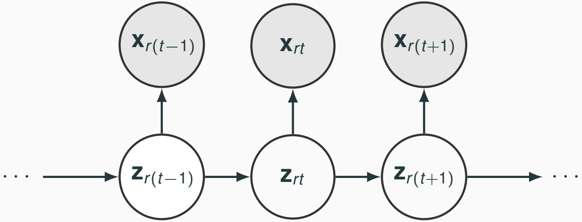


- $\mathbf{x}_t \in \mathbb{N}^n$: spike counts; $\mathbf{z}_t \in \mathbb{R}^m$: latent variables
- Joint distribution

$$p(\mathbf{x}, \mathbf{z}) = \underbrace{p(\mathbf{z}_1)}_{\text{Initial distribution}} \underbrace{\prod_{t=1}^{T-1} p(\mathbf{z}_{t+1}|\mathbf{z}_t)}_{\text{Transition model}} \underbrace{\prod_{t=1}^{T} p(\mathbf{x}_t|\mathbf{z}_t)}_{\text{Observation model}}$$

- Common input; Dynamical view of motor data (TODO: elaborate this line)

State space models: multiple trials



- $r = 1, ..., R$: trial number
- $\mathbf{x}_{rt} \in \mathbb{N}^n$: spike counts; $\mathbf{z}_{rt} \in \mathbb{R}^m$: latent variables
- Joint distribution

$$p(\mathbf{x}, \mathbf{z}) = \prod_{r=1}^{R} \left[\underbrace{p(\mathbf{z}_{r1})}_{\text{Initial distribution}} \underbrace{\prod_{t=1}^{T-1} p(\mathbf{z}_{r(t+1)} | \mathbf{z}_{rt})}_{\text{Transition model}} \underbrace{\prod_{t=1}^{T} p(\mathbf{x}_{rt} | \mathbf{z}_{rt})}_{\text{Observation model}} \right]$$

Common parameterization and our extensions

- Common assumptions for latent dynamics: linear Gaussian dynamical system (LDS)

$$\mathbf{z}_1 \sim \mathcal{N}(\mu_1, Q_1)$$
$$\mathbf{z}_{t+1}|\mathbf{z}_t \sim \mathcal{N}(A\mathbf{z}_t, Q)$$

- Common observation models:

$$\mathbf{x}_t|\mathbf{z}_t \sim \underbrace{\underbrace{\mathcal{N}(C\mathbf{z}_t + d, \Sigma)}_{\text{model mismatch}} \text{ or } \underbrace{\text{Poisson}\left(\exp(C\mathbf{z}_t + d)\right)}_{\text{equal dispersion}}}_{\text{stringent assumptions}}$$

- Our extensions for observation model:
  - Generalized count distribution (GCLDS) (Gao et al. 2015)
  - Flexible nonlinear observation (fLDS) (Gao et al. 2016)

Table of Contents

Motivation

- Doubly stochastic Poisson model implies overdispersion

$$\left.\begin{aligned} \mathbf{z} &\sim p(\mathbf{z}) \\ \mathbf{x} &\sim \text{Poisson}(f(\mathbf{z})) \end{aligned}\right\} \Rightarrow \text{var}(\mathbf{x}) \geq E(\mathbf{x})$$

- Need a more flexible distribution to separate firing rate variability with noise variability.

$$\text{var}(\mathbf{x}) = \underbrace{\text{var}\left(E(\mathbf{x}|\mathbf{z})\right)}_{\text{firing rate variability}} + \underbrace{E\left(\text{var}(\mathbf{x}|\mathbf{z})\right)}_{\text{noise variability}}$$

Generalized count distribution family

- Generalized count (GC) distribution family

$$p_{\text{Poisson}}(x;\lambda) \propto \frac{\exp\{\log\lambda \cdot x\}}{x!}, \quad x \in \mathbb{N}$$

$$\Downarrow$$

$$p_{\mathcal{GC}}(x;\theta, g(\cdot)) \propto \frac{\exp(\theta \cdot x + g(x))}{x!}, \quad x \in \mathbb{N}$$

  where $\theta \in \mathbb{R}$, $g(\cdot) : \mathbb{N} \to \mathbb{R}$.

- Parameterizes all the count distributions redundantly.
- Given $g(\cdot)$, θ controls the expectation.
- $g(\cdot)$ controls the "shape" of the distribution.
  Convex/concave $g(\cdot)$ implies over/under-dispersion.

Model formulation

- Linear dynamical systems with generalized count observation

$$
\begin{aligned}
\mathbf{z}_{r1} &\sim \mathcal{N}(\mu_1, Q_1) \\
\mathbf{z}_{r(t+1)}|\mathbf{z}_{rt} &\sim \mathcal{N}(A\mathbf{z}_{rt}, Q) \\
x_{rti} &\sim \mathcal{GC}(c_i^T\mathbf{z}_{rt}, g_i(\cdot)), i = 1, ..., n
\end{aligned}
$$

- Practical considerations
  - Set $g_i(k) = -\infty$ for $k > K$ to facilitate computation;
  - Ridge penalty on the 2nd difference of $g_i(\cdot)$ to avoid overfitting;
  - Set $g_i(0) = 0$ without loss of generality.

Variational Bayes Expectation Maximization (VBEM)

- $\mathbf{x}$: data, $\mathbf{z}$: latent variables, θ: model parameters,
- Often hard to compute $p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}, \mathbf{z}) d\mathbf{z}$ and $p_\theta(\mathbf{z}|\mathbf{x})$.
- Approximate the posterior by a tractable distribution family.

$$p_\theta(\mathbf{z}|\mathbf{x}) \approx q(\mathbf{z}) \in \mathcal{Q}$$

- Optimize a lower bound of log likelihood, or ELBO

$$\begin{aligned}\mathrm{ELBO}(\theta, q) &= \int \left[\log p_\theta(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z})\right] q(\mathbf{z}) d\mathbf{z} \\ &= \log p_\theta(\mathbf{x}) - \mathrm{KL}(q(\mathbf{z}) || p_\theta(\mathbf{z}|\mathbf{x})) \leq \log p_\theta(\mathbf{x})\end{aligned}$$

Variational Bayes Expectation Maximization (VBEM)

- VBEM: Optimize ELBO$(\theta, q) \leq \log p_\theta(\mathbf{x})$ iteratively
  - E-step: For a fixed θ, optimize q
  - M-step: For a fixed q, optimize θ
- VBEM for GCLDS
  - We set q to be multivariate Gaussian
  - We derive a looser but tractable ELBO
  - E-step: fast Laplace approximation initialization + dual optimization
  - M-step: convex optimization + analytical solution

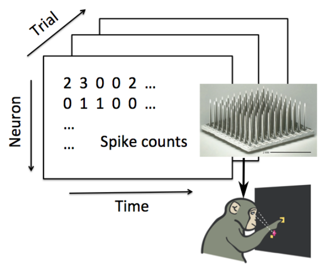
Experiments

- For both simulated and real dataset, we compare GCLDS with PLDS (Poisson observation model)

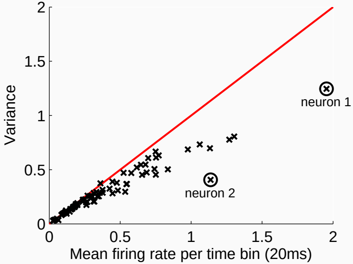
	Mean	Variance	Likelihood
PLDS	✓	✗	✗
GCLDS	✓	✓	✓

Real data analysis: data

Data



Variance and mean of spike counts

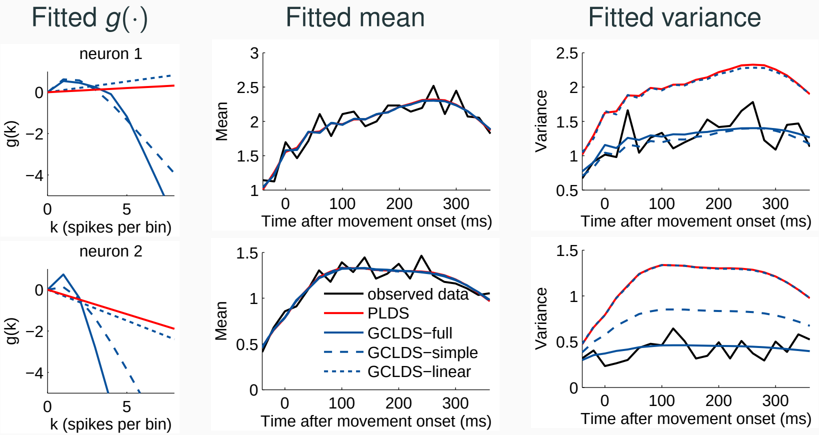


- Center-out reaching experiments
- Multi-electrode array recording
- Strong under-dispersion

Real data analysis: algorithms

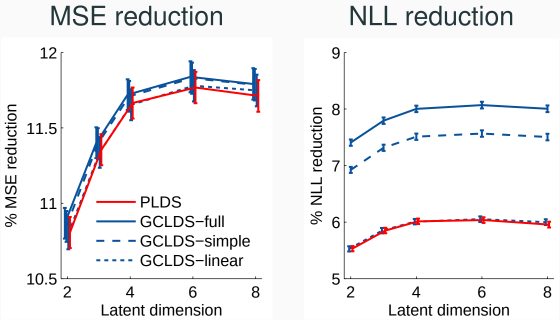
- Main algorithms to be compared
    - PLDS: Poisson observation
    - GCLDS-full: Generalized count observation, individual $g(\cdot)$ across neurons
- Two control cases for GCLDS
    - GCLDS-linear: truncated linear $g(\cdot)$ (truncated Poisson)
    - GCLDS-simple: $g(\cdot)$ shared across neurons (up to a linear function)

Real data analysis: single neuron fit



Real data analysis: population fit

- Leave-one-neuron-out prediction



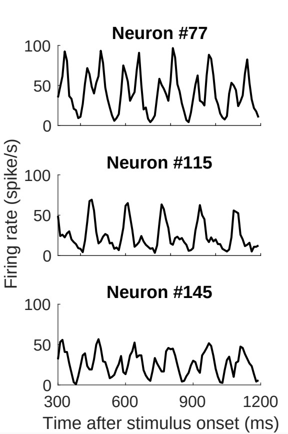
Conclusion and discussion

- Summary
  - Incorporated generalized count family into state space models.
  - Developed VBEM algorithm.
  - Observed superior fitted result on real neural data.
- Extensions
  - $g(\cdot)$ vary across time?
  - Share information of $g(\cdot)$ across neurons? (hierarchical model?)
  - Generative models for under-dispersion?

Table of Contents

Motivation

- Neural activities lie in a low-dimensional nonlinear manifold rather than a linear subspace
- Flexible observation model makes the state space model more expressive



Model formulation: fLDS

- Linear dynamical systems with nonlinear link and count observation

$$\begin{aligned} \mathbf{z}_{r1} &\sim \mathcal{N}(\mu_1, Q_1) \\ \mathbf{z}_{r(t+1)} | \mathbf{z}_{rt} &\sim \mathcal{N}(A\mathbf{z}_{rt}, Q) \\ x_{rti} &\sim \text{Poisson}(f_i(\mathbf{z}_{rt})) \text{ (PfLDS)} \\ &\quad \text{or } \mathcal{GC}(f_i(\mathbf{z}_{rt}), g_i(\cdot)) \text{ (GCfLDS)} \end{aligned}$$

  where f_i is a nonlinear function parameterized by a neural network
- Linear dynamics: simple, tractable, interpretable
- Nonlinear observation: flexibility

Inference algorithm: AEVB (high level idea)

- Auto-encoding Variational Bayes (AEVB)
- Learn a mapping (recognition model) from data to the approximate posterior distribution of latent variable.
- Jointly optimize the generative model parameters and recognition model parameters.
- Naturally incorporate stochastic optimization to handle large datasets.

Inference algorithm: AEVB (algorithm)

- Decompose ELBO by trials

$$\text{ELBO}(\theta, q) = \sum_{r=1}^{R} \int \left[\log p_\theta(\mathbf{x}_r, \mathbf{z}_r) - \log q(\mathbf{z}_r)\right] q(\mathbf{z}_r) d\mathbf{z}_r$$

- Map data $\mathbf{x}_r$ to $q(\mathbf{z}_r)$ by a parameterized function

$$q(\mathbf{z}_r) = q_\phi(\mathbf{z}_r; \mathbf{x}_r) = \mathcal{N}\left(\mu_\phi(\mathbf{x}_r), \Sigma_\phi(\mathbf{x}_r)\right)$$

- Learn both θ and ϕ by optimizing ELBO

$$\text{ELBO}(\theta, \phi) = \sum_{r=1}^{R} \int \left[\log p_\theta(\mathbf{x}_r, \mathbf{z}_r) - \log q_\phi(\mathbf{z}_r; \mathbf{x}_r)\right] q_\phi(\mathbf{z}_r; \mathbf{x}_r) d\mathbf{z}_r$$

- Do stochastic optimization with gradient of a single trial

Inference algorithm: AEVB (important details)

- Specific parameterization of the recognition model

$$q(\mathbf{z}_r) = q_\phi(\mathbf{z}_r; \mathbf{x}_r) = \mathcal{N}\left(\mu_\phi(\mathbf{x}_r), \Sigma_\phi(\mathbf{x}_r)\right)$$

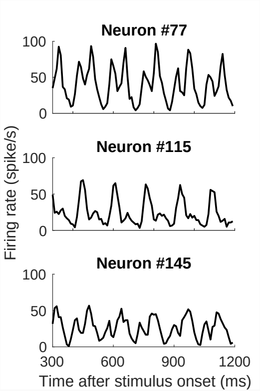
  - Block tri-diagonal precision matrix that agrees with Markovian structure
  - Potentially useful to perform filtering in an online fashion
- Reparameterization trick for stochastic optimization
  - Easy implementation
  - Low variance

Experiments

	Mean	Variance	Likelihood	Concise representation
PLDS	✓	✗	✗	✗
GCLDS	✓	✓	✓	✗
PfLDS	✓	✗	✗	✓
GCfLDS	✓	✓	✓	✓

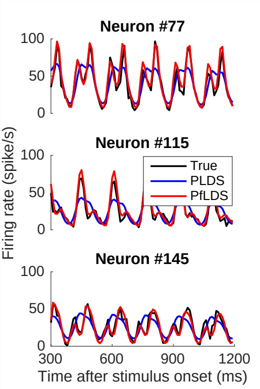
Real data analysis: primate visual cortex

Firing rate



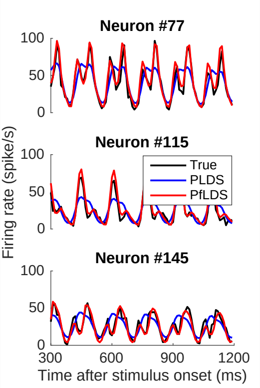
Real data analysis: primate visual cortex

Firing rate



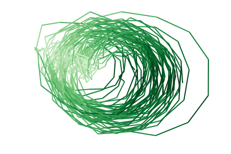
Real data analysis: primate visual cortex

Firing rate

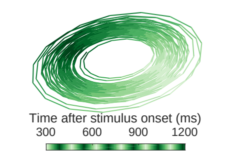


Latent projection

PLDS

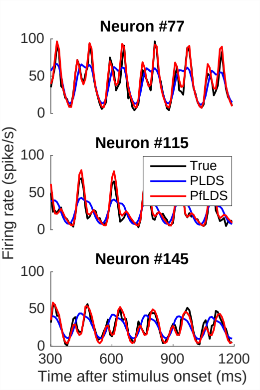


PfLDS



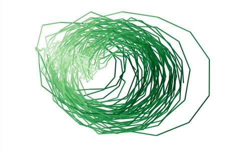
Real data analysis: primate visual cortex

Firing rate

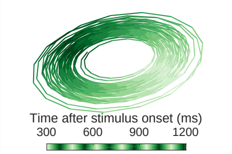


Latent projection

PLDS

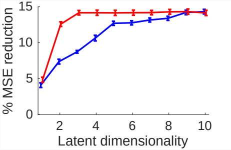


PfLDS

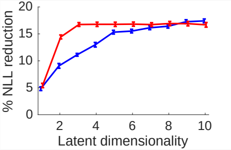


1-step-ahead prediction

MSE reduction

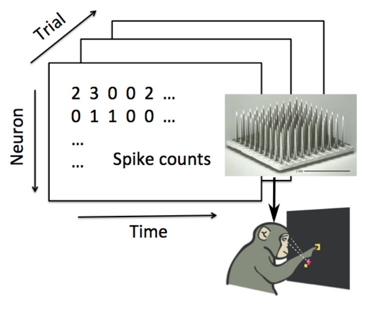


NLL reduction

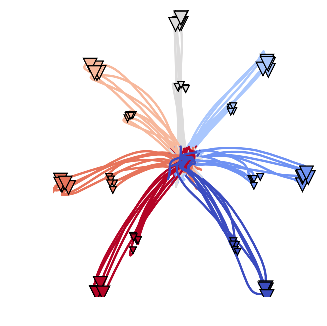


Real data analysis: Primate motor cortex

Data

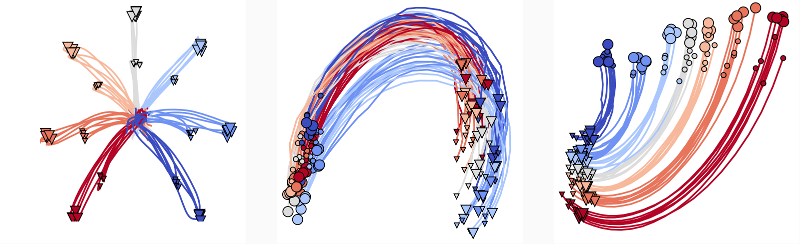


Reaching trajectory



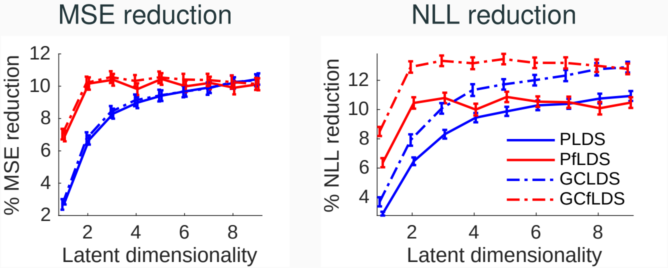
- Latent projection with 2 latent dimensions

Reaching trajectory | PLDS | PfLDS



Real data analysis: Primate motor cortex

- One-step-ahead predictive performance



Conclusion and discussion

- Summary
  - Incorporated nonlinear observation into state space models.
  - Developed AEVB algorithm (flexible and scalable).
  - Obtain concise latent representation.
- Future work
  - Better stochastic optimization scheme
  - Interpretable nonlinearity
  - Application on more complex datasets

Table of Contents

Introduction: calcium imaging data

TODO: incorporate a video?

Model formulation: idea

- $X \in \mathbb{R}^{N \times T}$ represents the calcium imaging data, where each column is a (vectorized) frame that contains N pixels
- Decompose X into a product of K spatial component and temporal component

$$X = D \cdot A^T + \text{noise}$$

  - $D = [D_1, ..., D_K] \in \mathbb{R}^{N \times K}$ represents the neuron shapes
  - $A = [A_1, ..., A_K] \in \mathbb{R}^{T \times K}$ is the neural activities
- Further exploit structure of the components (localized neuron shapes)

Model formulation: objective

- Structured matrix factorization

$$
\begin{aligned}
\underset{D,A}{\text{minimize}} \quad & \|X - DA^T\|_2^2 + f_D(D), \\
\text{subject to} \quad & D_k \in \mathcal{D}_w^+; k = 1, \ldots, K, \\
& \|A_k\|_2 \leq c_k,
\end{aligned}
$$

- $\mathcal{D}_w^+$: non-negative vectors whose nonzero values is within a $w \times w$ window
- $f_D(D)$ regularizes the neuron shape (discussed later)
- $\|A_k\|_2 \leq c_k$ avoids degenerate solution

Greedy algorithm

- Scan the each frame of the video with a small Gaussian kernel
- At iteration k, given the current residue (unexplained by existing ROI)
    - Greedy identification: Identify the location p_k where the Gaussian kernel explains most of the data (across time)
    - Shape fine tuning: Locally optimize the spatial and temporal component
    - Residue update: Subtract the newly identified ROI

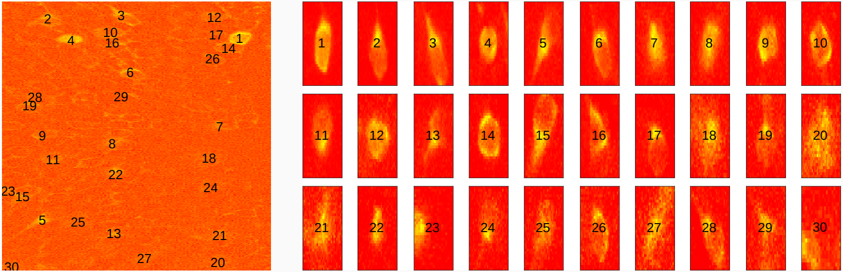
Shape fine tuning

- Given current residue R, an identified center pixel p_k, denote S_k as a $w \times w$ window centered at p_k

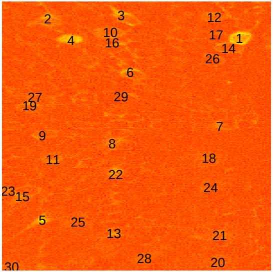
$$
\begin{aligned}
\underset{D_k, A_k}{\text{minimize}} \quad & \|R - D_k A_k^T\|^2 + f(D_k), \\
\text{subject to} \quad & D_{kp} \geq 0, p \in S_k, \\
& D_{kp} = 0, p \notin S_k, \\
& \|A_k\|_2 \leq c_k,
\end{aligned}
$$

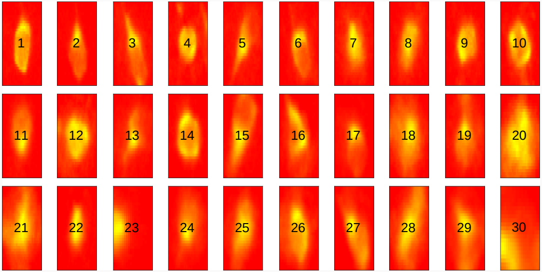
- $f(D_k) = \sum_{i=1}^{3} \lambda_i f_i(D_k)$
    - $f_1(D_k) = \sum_p \tau_{(p,p_k)} |D_{kp}|$ encourages sparsity
    - $f_2(D_k) = \sum_p (D_{kp} - G_{p_k})^2$ encourage Gaussian shape
    - $f_3(D_k) = \sum_{p_1 \text{ and } p_2 \text{ are neighbors}} (D_{kp_1} - D_{kp_2})^2$ encourages smoothness
- Optimize D_k and A_k by block coordinate descent

Real data analysis: sample patch, no shape regularization



Real data analysis: sample patch, shape regularization





Conclusion and discussion

- Summary
  - Formulating calcium imaging ROI detection as a structure matrix factorization problem
  - Greedy algorithm with shape regularization
  - Fast ROI detection algorithm
- Future work
  - More spatial and temporal structure
  - Overlapping neuron
  - Online ROI detection
  - Motion correction, background elimination

Table of Contents

Maximum entropy principle

- Entropy: for a continuous distribution with density $p(\mathbf{z})$ where $\mathbf{z} \in \mathbb{R}^d$, the entropy is defined as

$$H(p) = -\int p(\mathbf{z}) \log p(\mathbf{z}) d\mathbf{z}.$$

  A popular measure of diversity and information content.
- Maximum entropy principle: Subject to some given prior knowledge (moment/support constraints), the distribution that makes minimal additional assumptions is that which has the largest entropy of any distribution obeying those constraints

Maximum entropy problem

- Maximum entropy problem

$$\begin{aligned} p^* \quad = \quad & \text{maximize} \quad H(p) \\ & \text{subject to} \quad E_{\mathbf{Z}\sim p}[T(\mathbf{Z})] = 0 \\ & \qquad\qquad\quad \text{supp}(p) = \mathcal{Z}, \end{aligned}$$

where $T(\mathbf{z}) = (T_1(\mathbf{z}), ..., T_m(\mathbf{z})) : \mathcal{Z} \to \mathbb{R}^m$ is the vector of known statistics, and $\mathcal{Z}$ is the given support.

Application example: neuroscience

Application example: texture modeling

Gibbs distribution

- Under standard regularity conditions, the maximum entropy problem can be solved by Lagrange multipliers, yielding an exponential family p^* of the form (Gibbs distribution):

$$p^*(\mathbf{z}) \propto e^{<\eta, T(\mathbf{z})>} \mathbb{1}(\mathbf{z} \in \mathcal{Z})$$

- Identifying $\eta \in \mathbb{R}^m$ can be hard in high-dimensional setting.
- Sampling from the distribution can be hard. MCMC methods can take long to mix.
- Question: is there a better way to do this?

Idea: normalizing flow

- Considering a family of smooth and invertible transformation (normalizing flow)

$$\mathcal{F} = \{f_\phi : \mathbb{R}^d \to \mathbb{R}^d, \phi \in \mathbb{R}^q\}$$

- Identify a transformation $f_{\phi^*} \in \mathcal{F}$ that transforms a simple distribution p_0 to approximate the maximum entropy distribution.

$$\begin{aligned} \phi^* \quad = \quad & \text{maximize} \quad H(p_\phi) \\ & \text{subject to} \quad E_{\mathbf{Z}_0 \sim p_0}[T(f_\phi(\mathbf{Z}_0))] = 0 \\ & \qquad\qquad\quad \operatorname{supp}(p_\phi) = \mathcal{Z}. \end{aligned}$$

where $p_\phi(\mathbf{z})$ is the distribution of $f_\phi(Z_0)$ for $Z_0 \sim p_0$.

$$p_\phi(\mathbf{z}) = p_0(f_\phi^{-1}(\mathbf{z}))|\det(J_\phi(\mathbf{z}))|^{-1}$$

Augmented Lagrangian method

- Denote $R(\phi) = E\left(T(f_\phi(\mathbf{Z}_0))\right) \in \mathbb{R}^m$, augmented Lagrangian method minimizes the objective

$$L(\phi; \lambda, c) = -H(p_\phi) + \lambda^\top R(\phi) + \frac{c}{2}||R(\phi)||^2$$

for a sequence of $\lambda \in \mathbb{R}^m$ and $c \geq 0$.

- Update rule: at iteration k, given λ_k and c_k, suppose ϕ_k optimizes $L(\phi; \lambda_k, c_k)$, update λ and c by

$$\begin{aligned} \lambda_{k+1} =& \lambda_k + c_k R(\phi_k) \\ c_{k+1} =& \begin{cases} \beta c_k & ||R(\phi_k)|| > \gamma ||R(\phi_{k-1})|| \\ c_k & \text{otherwise} \end{cases} \end{aligned}$$

for some $\gamma \in (0, 1)$, $\beta > 1$

Augmented Lagrangian method in stochastic setting

- Denote $R(\phi) = E\left(T(f_\phi(\mathbf{Z}_0))\right) \in \mathbb{R}^m$, augmented Lagrangian method minimizes the objective

$$L(\phi; \lambda, c) = -H(p_\phi) + \lambda^\top R(\phi) + \frac{c}{2}||R(\phi)||^2$$

for a sequence of $\lambda \in \mathbb{R}^m$ and $c \geq 0$.

- Here $R(\phi) = E\left(T(f_\phi(\mathbf{Z}_0))\right) \in \mathbb{R}^m$ is intractable, but we can approximate with a sampled version.

$$R(\phi) \approx \frac{1}{n}\sum_{i=1}^{n} T(f_\phi(\mathbf{z}^{(i)})), \mathbf{z}^{(i)} \sim p_0$$

We can then optimize the objective by stochastic gradient descent.

Simulation: Dirichlet

Application: Texture modeling

Conclusion and discussion

- Summary
    - Solve maximum entropy problem by optimizing a normalizing flow
    - Combining augmented Lagrangian optimization with stochastic optimization
    - Promising result on simulation and real data
- Future work
    - Normalizing flow structure
    - Better constrained stochastic optimization algorithm

Table of Contents

fLDS: AEVB form (backup)

MEFN: Normalizing flow structures (backup)