# Online Learning for Large Scale Mixture Models

Mohammad Pasande

School of Electrical and Computer Engineering

University of Tehran

Aug, 2022

# Outline

# Preliminaries

# Gaussian Mixture (GM)

*'Many years ago I called the Laplace–Gaussian curve the normal curve, which name, while it avoids an international question of priority, has the disadvantage of leading people to believe that all other distributions of frequency are in one sense or another "abnormal".'*

Karl Pearson

# Gaussian Mixture (GM)

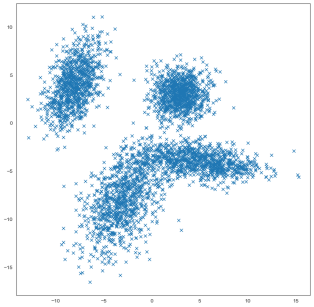
$$\mathcal{N}(x|\mu,\Sigma^{-1}) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\Sigma|^{1/2}} exp\Big(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\Big) \tag{1}$$

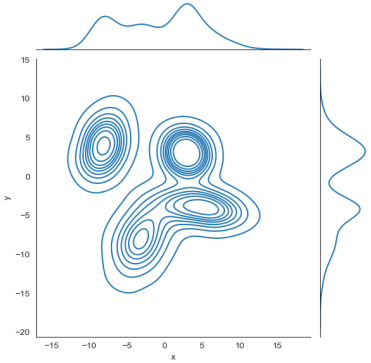
$$p(x) = \sum_{k=1}^{K}\pi_k\mathcal{N}(x|\mu_k,\Sigma_k^{-1})$$

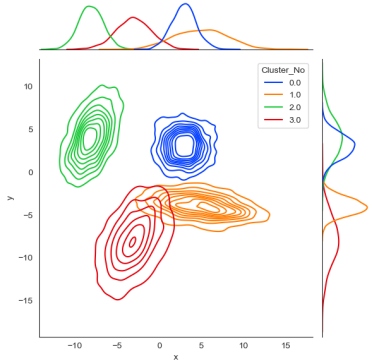
$$\text{s. t.} \quad \pi_k \geq 0 \qquad \& \qquad \sum_{k=1}^{K}\pi_k = 1.$$

$$\zeta^T\Sigma^{-1}\zeta \geq 0 \qquad (\forall\zeta\in\mathbb{R}^D).$$

# Gaussian Mixture Model (GMM)

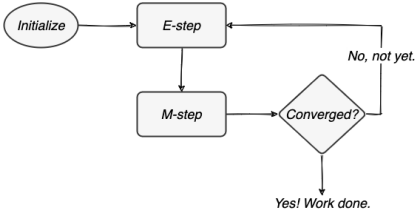




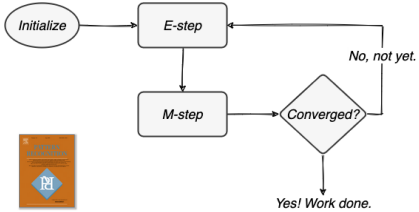
# Parameter Estimation

- Expectation Maximization (EM, CEM, Incr-EM, ...)



# Parameter Estimation

- Expectation Maximization (EM, CEM, Incr-EM, ...)




Pattern Recognition
Volume 114, June 2021, 107836



A new EM algorithm for flexibly tied GMMs with large number of components

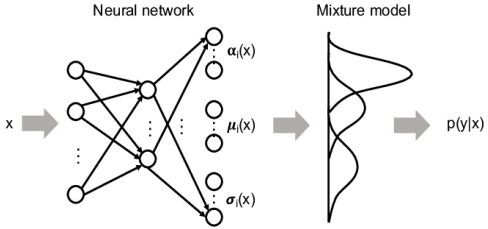
Hadi Asheri, Reshad Hosseini ✉, Babak Nadjar Araabi

The Proposed Hybrid Algorithm outperforms others in both convergence and time performance.[Asheri et al., 2021]

| GMM Type | Learning Algorithm | BSDS500 (64) | CIFAR10 (89) | CIFAR100 (98) | MAGIC (10) | MNIST (154) | STL (101) | SVHN (100) | USPS (65) | WAVE (21) | YEAR (90) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ***basic*** | EM | 48.24 | 116.39 | 127.95 | 9.87 | 139.75 | 114.37 | 87.32 | 52.7 | 29.16 | 100.89 |
| | CD-EM | **15.95** | 113.93 | 125.38 | **9.71** | 130.36 | 95.59 | 70.06 | **49.82** | 29.06 | 95.28 |
| ***flexibly tied*** | FNMR-EM | 16.73 | 114.46 | 125.01 | 9.72 | 131.86 | 96.04 | 70.10 | 49.94 | 29.68 | 96.16 |
| | FNMR-CD-EM | **15.95** | **113.79** | **124.84** | **9.71** | **130.29** | **95.56** | **70.01** | **49.82** | **29.04** | **95.26** |

# Parameter Estimation

- Expectation Maximization (EM, CEM, Incr-EM, ...)
- Neural Networks (Mixture Density Networks)



# Parameter Estimation

- Expectation Maximization (EM, CEM, Incr-EM, ...)
- Neural Networks (Mixture Density Networks)
- **Numerical Optimization**

$$loss := \mathcal{L}(\pi_k, \mu_k, \Sigma_k^{-1}) = -\sum_{i=1}^{N} \log \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(x_i|\mu_k, \Sigma_k) \right\} \quad NLL \tag{2}$$

$$\begin{aligned} \underset{\theta}{\text{minimize}} \quad & \mathcal{L}(\pi_k, \mu_k, \Sigma_k^{-1}) \\ \text{subject to} \quad & \pi_k \geq 0 \ \& \ \sum_{k=1}^{K} \pi_k = 1. \\ & \Sigma_k^{-1} \succeq 0. \end{aligned} \tag{3}$$

# The Passion Of Constraints!

- $\pi_k$ Constraint:

$$\pi_j = \frac{\exp(\eta_j)}{\sum_{k=1}^{K} \exp(\eta_k)}$$

AKA: SoftMax.

Dobby is FREE!

# The Passion Of Constraints!

- $\pi_k$ Constraint:

$$\pi_j = \frac{\exp(\eta_j)}{\sum_{k=1}^{K} \exp(\eta_k)}$$ AKA: SoftMax.

- $\Sigma_k^{-1}$ Constraint:
  - Cholesky Decomposition:

$$\Sigma^{-1} = LDL^T$$

# The Passion Of Constraints!

- $\pi_k$ Constraint:

$$\pi_j = \frac{\exp(\eta_j)}{\sum_{k=1}^{K} \exp(\eta_k)} \quad \text{AKA: SoftMax.}$$

- $\Sigma_k^{-1}$ Constraint:
  - Reformulation Trick [Hosseini and Sra, 2015] :

$$S_k = \begin{pmatrix} \Sigma_k^{-1} + \mu_k \mu_k^T & \mu_k \\ \mu_k^T & 1 \end{pmatrix}; y = \begin{pmatrix} x & 1 \end{pmatrix}^T$$

# The Passion Of Constraints!

- $\pi_k$ Constraint:

$$\pi_j = \frac{\exp(\eta_j)}{\sum_{k=1}^{K} \exp(\eta_k)}$$ AKA: SoftMax.

- $\Sigma_k^{-1}$ Constraint:
  - Semi-Tied [Gales, 1999] :

$$\Sigma_k^{-1} = U D_k U^T$$

# How About $UDU^T$?

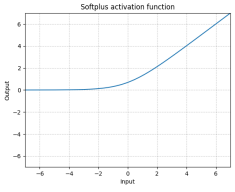
Simplify the $\Sigma^{-1}$ as much as you can!

- put the orthogonal constraint on $U$:

$$UU^T = I_D \quad \Rightarrow \quad det(\Sigma_k^{-1}) = \prod_{j=1}^{D} d_k^{(j)}$$

- consider each $d_k^{(j)}$ is a SoftPlus output:

$$d_k^{(j)} = \frac{1}{\beta} \log\left(1 + \exp(\beta \hat{d}_k^{(j)})\right) \quad \Rightarrow \quad D_k \succeq 0$$




- inject more flexibility using component-wise scaler:

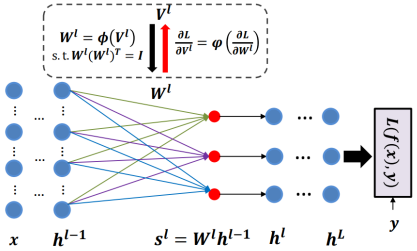
$$UD_kU^T \quad \rightarrow \quad \lambda_k UD_kU^T$$

# Optimization

# Orthonormality

$$O(n) = \{X \in \mathbb{R}^{n \times n} : X^T X = I_n\} \rightarrow SO(n) = \{X \in O(n) : det(X) = +1\}$$

[Huang et al., 2018]

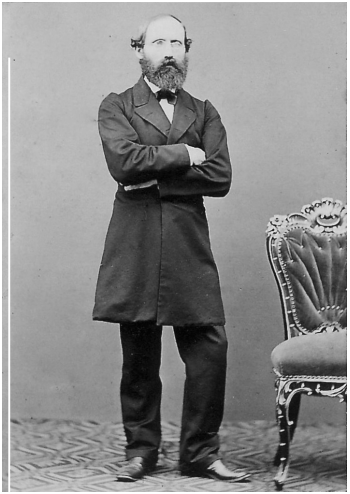



[Bansal et al., 2018]

| | |
|---|---|
| Double Soft Orthogonality | $\lambda(\|W^T W - I\|_F + \|WW^T - I\|_F)$ |
| Mutual Coherence | $\lambda(\|W^T W - I\|_\infty)$ |
| Spectral Restricted Isometry Property | $\lambda(Sup_{z \in \mathbb{R}^n, z \neq 0} \|\frac{\|(Wz\|}{\|z\|} - 1\|)$ |

# Riemannian Manifold

You might want to sit down for this



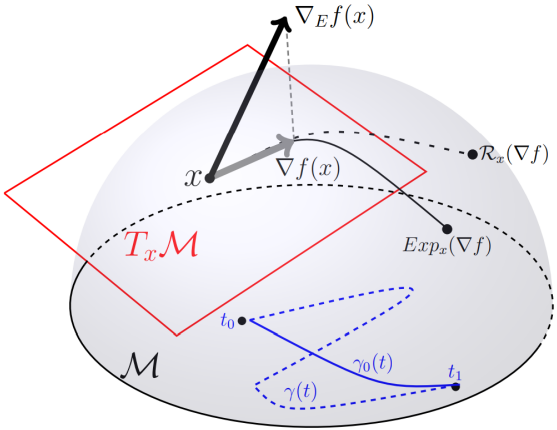
# Riemannian Manifold

**Riemannian Metric** := $< ., . >_x$

- Bilinear
- Symmetric
- Positive Definite

Better

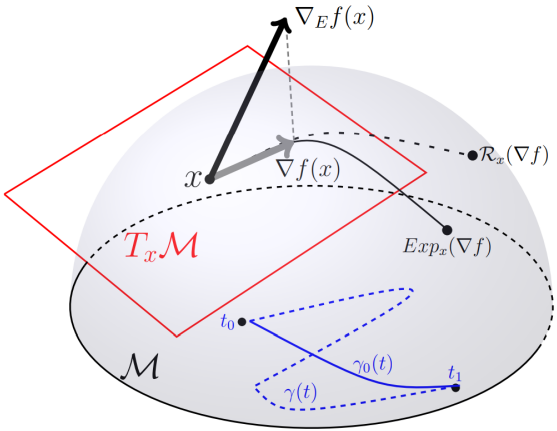
# Riemannian Manifold




- $\{x \in \mathbb{R}^{n+1} \mid \|x\|_2 = 1\}$
- $g : T_x\mathcal{M} \times T_x\mathcal{M} \to \mathbb{R}$
- $\gamma_0(t)$ : Geodesic
- $\nabla_E f$ and $\nabla f$ indicate the Edulidian gradient and Riemannian gradient
- $Exp_x(\cdot)$ : Exponential Map
- $\mathcal{R}_x(\cdot)$ : Retraction function

# Riemannian Manifold (Let There be Light)

*' To deal with hyper-planes in a 14-dimensional space, visualize a 3-D space and say "fourteen" to yourself very loudly.* Everyone does it.' Geoffrey E. Hinton




- $\{x \in \mathbb{R}^{n+1} \mid \|x\|_2 = 1\}$
- $g : T_x\mathcal{M} \times T_x\mathcal{M} \to \mathbb{R}$
- $\gamma_0(t)$ : Geodesic
- $\nabla_E f$ and $\nabla f$ indicate the Edulidian gradient and Riemannian gradient
- $Exp_x(\cdot)$ : Exponential Map
- $\mathcal{R}_x(\cdot)$ : Retraction function

# Stochastic Gradient Descent on GMM [Hosseini and Sra, 2020]

$$f_i(\eta_k, S_k) := \log\left\{\sum_{k=1}^{K} \frac{\exp(\eta_k)}{\sum_{j=1}^{K}\exp(\eta_j)}\mathcal{N}(y_i; 0, S_k)\right\} + \frac{1}{n}\left(\sum_{k=1}^{K}\psi(S_k; \Psi) + \phi(\{\eta_k\}_{k=1}^{K-1}; \zeta)\right),$$

$$\left\{\{S_k \succ 0\}_{k=1}^{K}, \{\eta_k\}_{k=1}^{K-1}\right\} \leftarrow Ret\left(\eta_t \nabla f_i\left(\{S_k \succ 0\}_{k=1}^{K}, \{\eta_k\}_{k=1}^{K-1}\right)\right). \tag{4}$$

**Note.**

In case of $S_k$, $\left(\nabla f_i(.)\right)$ refers to the Riemannian gradient which can be achieved by mapping the Euclidean gradient $\left(\nabla_E f_i(.)\right)$ on the manifold of Symmetric Positive Definite (SPD) matrices.

# Proposed Method

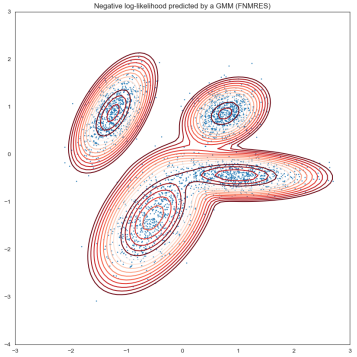
# General SGD for GMM

**Algorithm 1** SGD GMM using $UD_kU^T$

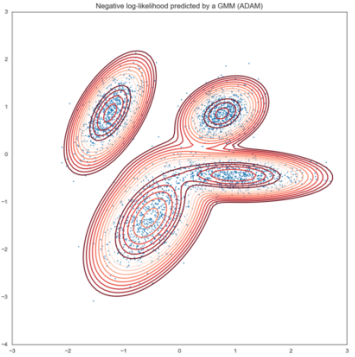
1: Initial values of $U$, $\hat{D}_k$, $\eta_k$ and $\mu_k$;
2: **while** *epoch* $<$ *MaxNo.Epochs* **do**
3:  **for** each **Train** iteration **do**
4:  $\nabla_E \mathcal{L}_i$ : Compute Stochastic Euclidean Gradient;
5:  Take a Stochastic Gradient step;
6:  Compute the negative log-likelihood;
7:  **end for**
8: **end while**

# Is it working?

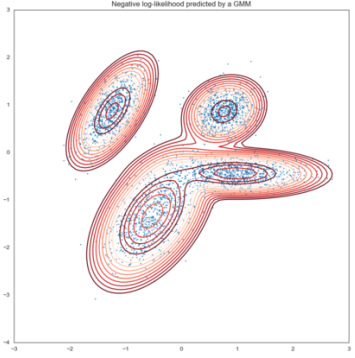
Using CD-FNMRES EM




Using ADAM




Using vanilla EM (Kmeans initialization)



# Adaptive Coordinate-wise CLIPping SGD with Orthogonality Constraint

**Algorithm 2** ACClipping Gradient Descent with orthogonality constraint (using Retraction)

$M_t, \tau_t \leftarrow 0, 0$;{Initialize}
**for** t = 1, . . . , T **do**
  $M_{t+1} \leftarrow \beta_1 M_t + (1-\beta_1)\mathcal{G}_t$ ; {$\mathcal{G}$ as euclidean gradient}
  $\tau_{t+1}^{\alpha} \leftarrow \beta_2 \tau_t^{\alpha} + (1-\beta_2)|\mathcal{G}_t|^{\alpha}$
  $\hat{\mathcal{G}} = min(\frac{\tau_{t+1}}{|M_{t+1}|+\epsilon}, 1) M_{t+1}$ ; {Adaptive Coordinate-wise CLIPping}
  $M_{t+1} \leftarrow \mathcal{P}roj(\hat{\mathcal{G}})$ {Project onto the tangent space}
  $X_{t+1} \leftarrow \mathcal{R}_{X_t}(-\alpha M_{t+1})$ {Retraction, $\alpha$ is the learning rate}
**end for**

# Controlled Datasets

- Separation:

$$\forall\, i \neq j \; \|\mu_i - \mu_j\| \geq c \max_{i,j}\{tr(\Sigma_i), tr(\Sigma_i)\}$$

High Sep. : $c = 10$, Mid Sep. : $c = 1$, Low Sep. : $c = 0.1$

Smallest Eigenvalue : $e = 10$

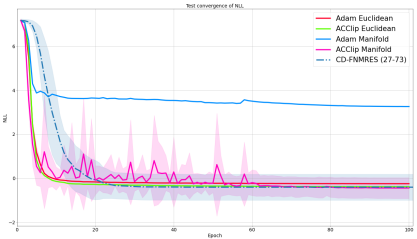
No. of Clusters : $K = 5$ No. of Dimensions : $d = 5$

No. of Data Points : $10d^2$, $100d^2$, $1000d^2$

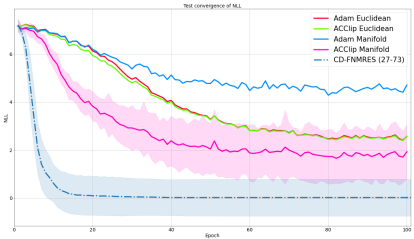
No. of Tests : $N = 10$

- Errors:
  - Averaged NLL over N Tests
  - Frobenius norm of differences for Covariance matrices
  - Cosine similarity distance of differences for Mean Vectors
  - difference of L2 Norms for weight Vectors

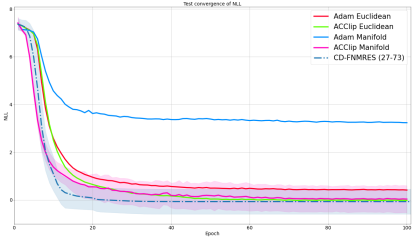
# High Sep. NLL




25000 datapoints

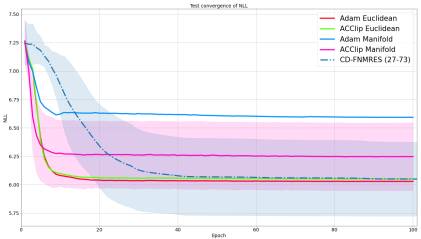



2500 datapoints

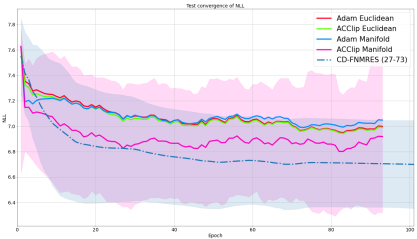



250 datapoints

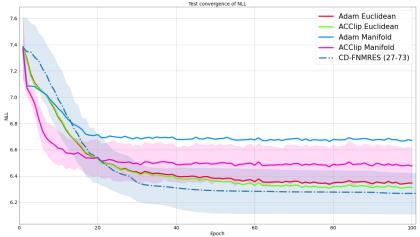
# Mid Sep. NLL




25000 datapoints

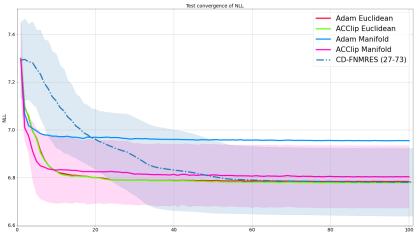



2500 datapoints

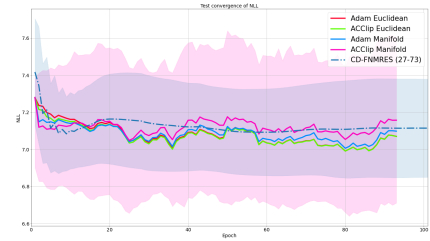



250 datapoints

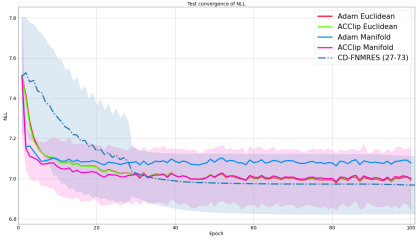
# Low Sep. NLL




25000 datapoints




2500 datapoints




250 datapoints

# Experiments

# Experiment Setup

| DataSet | No. Sample | Dimension | Description |
|---|---|---|---|
| WAVE | 5000 | 21 | Waveform Database Generator Generator generating 3 classes of waves. Each class is generated from a combination of 2 of 3 "base" waves. |
| SVHN | 99289 | $32 \times 32$ | The dataset is obtained from house numbers in Google Street View images. There are 531,131 additional samples that we do not use. |
| USPS | 7291 | $16 \times 16$ | The dataset contains normalized handwritten digits, automatically scanned from envelopes by the U.S. Postal Service. |
| YEAR | 515345* | 90 | The dataset are audio features from different songs. It has been gathered to be utilized to predict the release year of a song. |

Table: DataSet Description

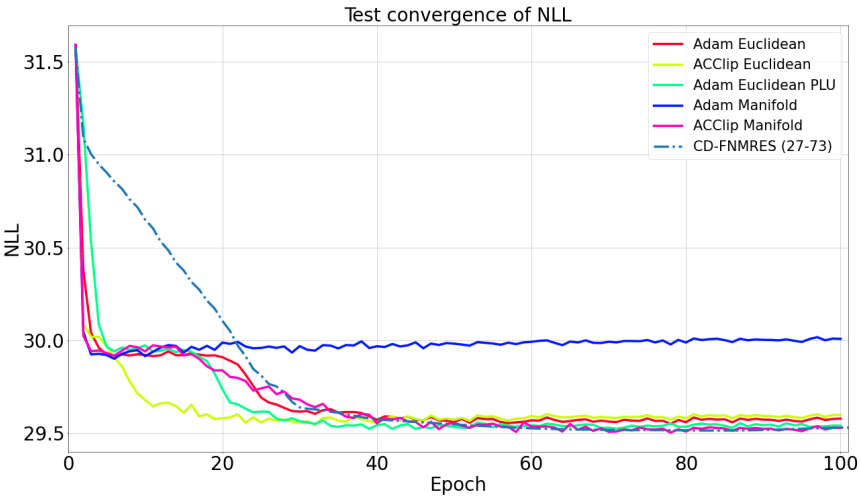
* 125000 of that had been used in tests

# Experiment Setup

- The proportion is 80% Train set and 20% Test set in all Datasets.
- Dimension reduction with explained variance consideration is applied to data in some cases.
- Data had been whitened before entering the procedure.
- Number of components is fixed to dimension of dataset.
- Batch Size is always 128
- Single CPU Core was used
- Cosine Annealing was used for step-size decay

* 125000 of that had been used in tests

# Experiment Results (WAVE[21])

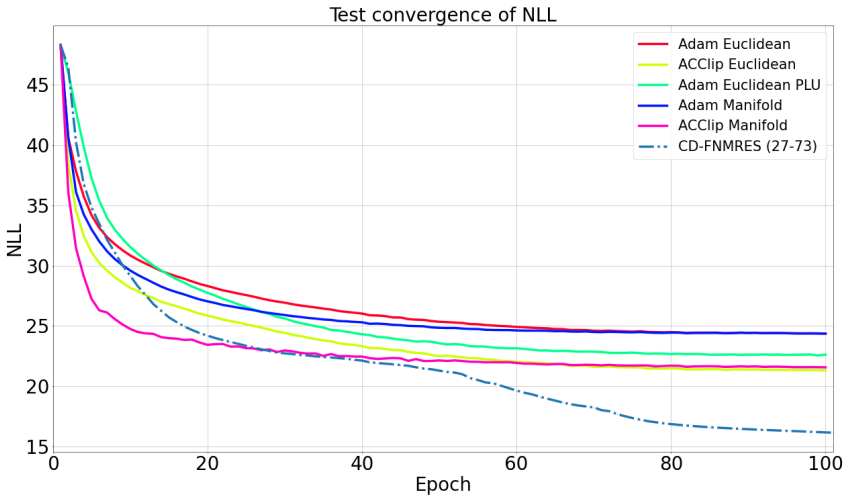


# Experiment Results (WAVE[21])

| Method | NLL | Time per epoch | Comments |
|---|---|---|---|
| Adam Euclidean | 29.57 | $0.49 \pm 0.01s$ | |
| ACClip Euclidean | 29.58 | $0.57 \pm 0.01s$ | |
| Adam Euclidean PLU | 29.54 | $0.72 \pm 0.04s$ | |
| Adam Manifold | 29.99 | $0.43 \pm 0.01s$ | qr Ret |
| ACClip Manifold | 29.51 | $0.65 \pm 0.02s$ | qr Ret |
| CD-FNMRES | 29.53 | $0.23 \pm 0.03s$ | 27-73 |

Table: Time performance

# Experiment Results (SVHN[30])

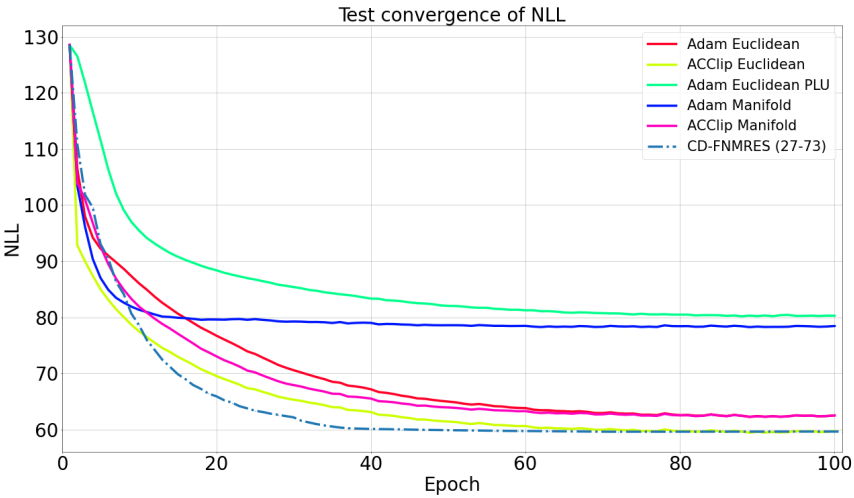


# Experiment Results (SVHN[30])

| Method | NLL | Time per epoch | Comments |
|---|---|---|---|
| Adam Euclidean | 24.37 | $15.45 \pm 0.45s$ | |
| ACClip Euclidean | 21.33 | $16.33 \pm 0.57s$ | |
| Adam Euclidean PLU | 25.31 | $24.18 \pm 1.51s$ | |
| Adam Manifold | 24.36 | $12.72 \pm 0.39s$ | qr Ret |
| ACClip Manifold | 21.57 | $14.71 \pm 0.37s$ | qr Ret |
| CD-FNMRES | 29.53 | $4.55 \pm 0.57s$ | 27-73 |

Table: Time performance

# Experiment Results (USPS[65])

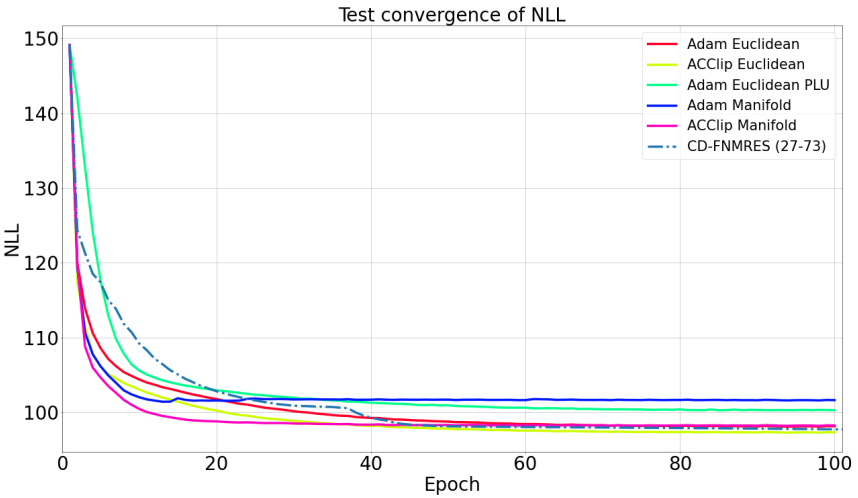


# Experiment Results (USPS[65])

| Method | NLL | Time per epoch | Comments |
|---|---|---|---|
| Adam Euclidean | 62.51 | $4.43 \pm 0.44s$ | |
| ACClip Euclidean | 59.73 | $4.75 \pm 0.37s$ | |
| Adam Euclidean PLU | 80.24 | $8.95 \pm 0.94s$ | |
| Adam Manifold | 78.47 | $3.77 \pm 0.40s$ | qr Ret |
| ACClip Manifold | 62.52 | $5.25 \pm 0.37s$ | qr Ret |
| CD-FNMRES | 59.66 | $15.75 \pm 8.37s$ | 27-73 |

Table: Time performance

# Experiment Results (YEAR[90])

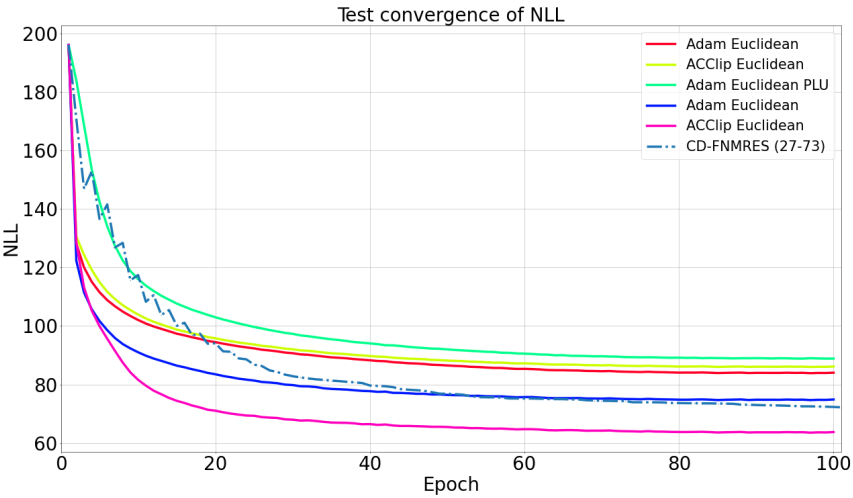


# Experiment Results (YEAR[90])

| Method | NLL | Time per epoch | Comments |
|---|---|---|---|
| Adam Euclidean | 98.16 | $94.63 \pm 5.53s$ | |
| ACClip Euclidean | 97.31 | $103.34 \pm 5.87s$ | |
| Adam Euclidean PLU | 100.31 | $195.47 \pm 16.31s$ | |
| Adam Manifold | 101.61 | $78.13 \pm 4.07s$ | qr Ret |
| ACClip Manifold | 98.22 | $97.37 \pm 3.72s$ | qr Ret |
| CD-FNMRES | 97.74 | $141.06 \pm 60.20s$ | 27-73 |

Table: Time performance

# Experiment Results (SVHN[100])

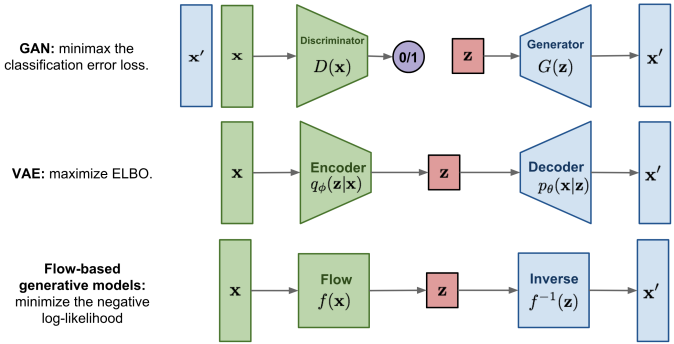


# Experiment Results (SVHN[100])

| Method | NLL | Time per epoch | Comments |
|---|---|---|---|
| Adam Euclidean | 83.90 | $83.37 \pm 6.49s$ | |
| ACClip Euclidean | 85.95 | $90.41 \pm 6.54s$ | |
| Adam Euclidean PLU | 88.82 | $185.96 \pm 19.33s$ | |
| Adam Manifold | 85.54 | $86.04 \pm 3.19s$ | qr Ret |
| ACClip Manifold | 63.57 | $85.01 \pm 13.34s$ | qr Ret |
| CD-FNMRES | 72.17 | $216.30 \pm 107.42s$ | 27-73 |

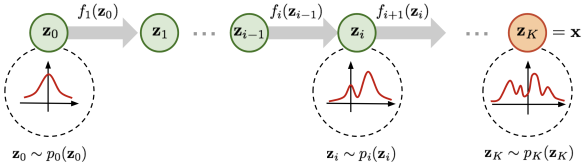
Table: Time performance

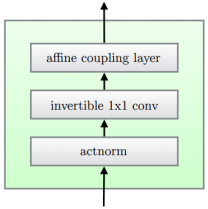
# Flow-based Deep Generative Models

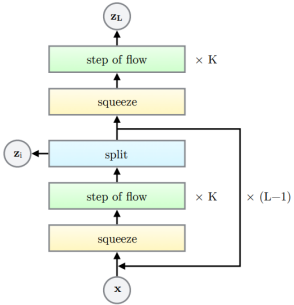


# Flow-based Deep Generative Models

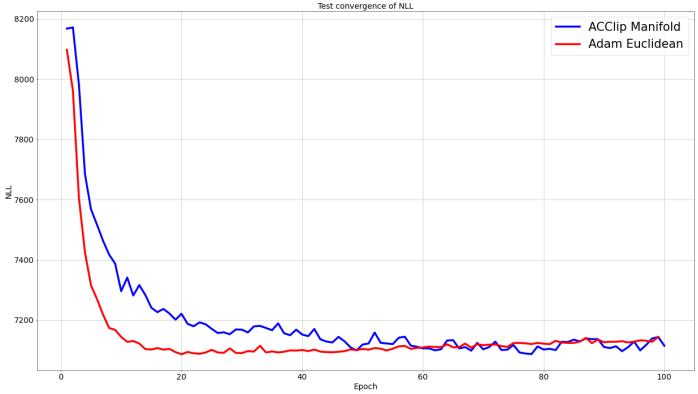
[Kingma and Dhariwal, 2018]



# GLOW Results



# GLOW Results

Last 3 Epochs of ADAM

# GLOW Results

Last 3 Epochs of ADAM

Last 3 Epochs of Ours

# GLOW Results

Last 3 Epochs of ADAM



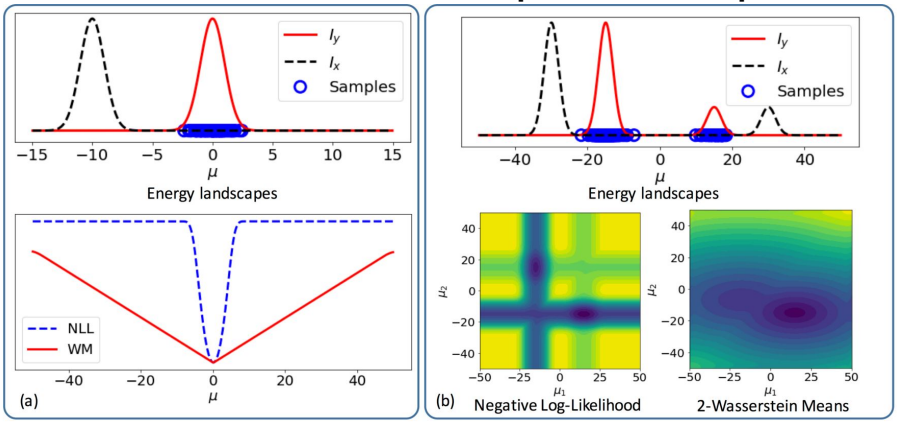
Last 3 Epochs of Ours



# Conclusion

# Discussion

Parameter Estimation is HARD! [Kolouri et al., 2018]



# Discussion

- We have investigated the online numerical solution for parameter estimation off GMM in sharing parameter scheme.
- An online optimization algorithm on SO(n) is proposed which has been tested on both GMM and GLOW.
- The PLU factorization was shown to be not suitable for GMM parameter estimation.

# References I



Asheri, H., Hosseini, R., and Araabi, B. N. (2021).
A new em algorithm for flexibly tied gmms with large number of components.
*Pattern Recognition*, 114:107836.


Bansal, N., Chen, X., and Wang, Z. (2018).
Can we gain more from orthogonality regularizations in training deep networks?
*Advances in Neural Information Processing Systems*, 31.


Gales, M. J. (1999).
Semi-tied covariance matrices for hidden markov models.
*IEEE transactions on speech and audio processing*, 7(3):272–281.


Hosseini, R. and Sra, S. (2015).
Matrix manifold optimization for gaussian mixtures.
*Advances in Neural Information Processing Systems*, 28:910–918.

# References II


Hosseini, R. and Sra, S. (2020).
An alternative to em for gaussian mixture models: batch and stochastic riemannian optimization.
*Mathematical Programming*, 181(1):187–223.

Huang, L., Liu, X., Lang, B., Yu, A., Wang, Y., and Li, B. (2018).
Orthogonal weight normalization: Solution to optimization over multiple dependent stiefel manifolds in deep neural networks.
In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Kingma, D. P. and Dhariwal, P. (2018).
Glow: Generative flow with invertible 1x1 convolutions.
*Advances in neural information processing systems*, 31.

# References III


Kolouri, S., Rohde, G. K., and Hoffmann, H. (2018).
Sliced wasserstein distance for learning gaussian mixture models.
In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3427–3436.

# Thanks for Your Attention

Please feel free to share comments or ask questions!

*mohammad.pasande@ut.ac.ir*