Learning Structural Weight Uncertainty for Sequential Decision-Making: Supplementary Material

Ruiyi Zhang[1] **Chunyuan Li**[1] **Changyou Chen**[2] **Lawrence Carin**[1]
[1]Duke University [2]University at Buffalo
ryzhang@cs.duke.edu, cl319@duke.edu, cchangyou@gmail.com, lcarin@duke.edu

A Proof of Proposition 3

Proof. Let a MVG distributed matrix $\mathbf{W}$ be

$$\mathbf{W} \sim \mathcal{MN}(\mathbf{W}; \mathbf{M}, \mathbf{U}, \mathbf{V}), \tag{20}$$

Since the covariance matrices $\mathbf{U}$ and $\mathbf{V}$ are positive definite, we can decompose them as

$$\mathbf{U} = \mathbf{P}\mathbf{\Lambda}_1\mathbf{\Lambda}_1\mathbf{P}^\top, \tag{21}$$

$$\mathbf{V} = \mathbf{Q}\mathbf{\Lambda}_2\mathbf{\Lambda}_2\mathbf{Q}^\top, \tag{22}$$

where $\mathbf{P}$ and $\mathbf{Q}$ are the corresponding orthogonal matrices, *i.e.*, $\mathbf{P}\mathbf{P}^\top = \mathbf{I}$, $\mathbf{Q}\mathbf{Q}^\top = \mathbf{I}$.

According to Lemma 2, we have,

$$\mathbf{P}^\top\mathbf{W} \sim \mathcal{MN}(\mathbf{P}^\top\mathbf{W}; \mathbf{M}, \mathbf{P}^\top\mathbf{U}\mathbf{P}, \mathbf{V}), \tag{23}$$

Since $\mathbf{U} = \mathbf{P}\mathbf{\Lambda}_1\mathbf{\Lambda}_1\mathbf{P}^\top$, and we have:

$$\mathbf{P}^\top\mathbf{W} \sim \mathcal{MN}(\mathbf{P}^\top\mathbf{W}; \mathbf{M}, \mathbf{P}^\top\mathbf{P}\mathbf{\Lambda}_1\mathbf{\Lambda}_1\mathbf{P}^\top\mathbf{P}, \mathbf{V}). \tag{24}$$

Then,

$$\mathbf{P}^\top\mathbf{W} \sim \mathcal{MN}(\mathbf{P}^\top\mathbf{W}; \mathbf{M}, \mathbf{\Lambda}_1\mathbf{\Lambda}_1, \mathbf{V}), \tag{25}$$

Similarly, we have,

$$\mathbf{P}^\top\mathbf{W}\mathbf{Q} \sim \mathcal{MN}(\mathbf{P}^\top\mathbf{W}\mathbf{Q}; \mathbf{M}, \mathbf{\Lambda}_1\mathbf{\Lambda}_1, \mathbf{\Lambda}_2\mathbf{\Lambda}_2). \tag{26}$$

Further,

$$\mathbf{\Lambda}_1^{-1}\mathbf{P}^\top\mathbf{W}\mathbf{Q}\mathbf{\Lambda}_2^{-1} \sim \mathcal{MN}(\mathbf{\Lambda}_1^{-1}\mathbf{P}^\top\mathbf{W}\mathbf{Q}\mathbf{\Lambda}_2^{-1}; \mathbf{0}, \mathbf{I}, \mathbf{I}), \tag{27}$$

Define $\mathbf{C} = \mathbf{\Lambda}_1^{-1}\mathbf{P}^\top\mathbf{W}\mathbf{Q}\mathbf{\Lambda}_2^{-1}$, then $\mathbf{C}$ follows an independent Gaussian distribution:

$$\mathbf{C} \sim \mathcal{MN}(\mathbf{C}; \mathbf{P}^\top\mathbf{\Lambda}_1^{-1}\mathbf{M}\mathbf{\Lambda}_2^{-1}\mathbf{Q}, \mathbf{I}, \mathbf{I}), \tag{28}$$

(28) can also be expressed as:

$$\mathsf{vec}(\mathbf{C}) \sim \mathcal{N}(\mathsf{vec}(\mathbf{C}); \mathbf{P}^\top\mathbf{\Lambda}_1^{-1}\mathbf{M}\mathbf{\Lambda}_2^{-1}\mathbf{Q}, \mathbf{I}), \tag{29}$$

showing that vectorized elements form of $\mathbf{C}$ follows an isotropic Gaussian distribution. Finally, since $\mathbf{\Lambda}_1\mathbf{C}\mathbf{\Lambda}_2 = \mathbf{P}^\top\mathbf{W}\mathbf{Q}$, we have:

$$\mathbf{W} = \mathbf{P}\mathbf{\Lambda}_1\mathbf{C}\mathbf{\Lambda}_2\mathbf{Q}^\top \ . \tag{30}$$

□

B Properties of Householder Flow

B.1 The Upperbound of Orthogonal Degree

Lemma 5 ([Sun and Bischof, 1995] The Basis-Kernel Representation of Orthogonal Matrices)**.** *For any $m \times m$ orthogonal matrix $\mathbf{Q}$, there exist a full-rank $m \times k$ matrix $\mathbf{Y}$ and a nonsingular $k \times k$ matrix $\mathbf{S}$, $k \leq m$, such that:*

$$\mathbf{Q} \triangleq \mathbf{Q}(\mathbf{Y}, \mathbf{S}) = \mathbf{I} - \mathbf{Y}\mathbf{S}\mathbf{Y}^\top \tag{31}$$

Definition 6 ([Sun and Bischof, 1995] Active Subspace)**.** *Orthogonal matrix $\mathbf{Q}$ acts on the space $\mathcal{R}(\mathbf{Y})^\perp$ as the identity and changes every nonzero vector in $\mathcal{R}(\mathbf{Y})$, and $\mathcal{R}(\mathbf{Y})^\perp$ is the active space of $\mathbf{Q}$*

In basis-kernel representation, $\mathbf{S}$ is the kernel, and $\mathbf{Y}$ is the basis. The *degree of an orthogonal matrix* is defined as the dimension of its active subspace. Specifically, Householder matrix is an orthogonal matrix of degree 1. With the introduced definitions and Lemma 5, the degree of an orthogonal matrix is bounded by Lemma 7.

Lemma 7 ([Sun and Bischof, 1995])**.** *Let $\mathbf{A}$ and $\mathbf{B}$ be two m-by-k matrices, $k < m$. If $\mathbf{B} = \mathbf{Q}\mathbf{A}$ for some orthogonal matrix $\mathbf{Q}$, then $\mathbf{Q}$ is either of degree no greater than k or can be replaced by an orthogonal factor of its own with degree no greater than k.*

Since the degree of orthogonal matrix is bounded by size of the matrix (k), the number of Householder transformations needed for Householder flow is also bounded.

B.2 Intuitive Explanation

In our setting, the orthogonal matrix works as a rotation matrix. The Householder transformation reflects the weights by a hyperplane orthogonal to its corresponding Householder vector. Hence, Householder flows applies a series of reflections to the original weights. In another word, it rotates the weight matrix, equivalent to the effects of the rotation matrix.

C Computational Trick

Assume $\boldsymbol{v}$ is a Householder vector, and its corresponding Householder matrix $\mathbf{H} \triangleq \mathbf{I} - 2\frac{\boldsymbol{v}\boldsymbol{v}^\top}{\|\boldsymbol{v}\|^2}$. For a feature vector $\boldsymbol{x}$, defining $\hat{\boldsymbol{v}} \triangleq \boldsymbol{v}/\|\boldsymbol{v}\|$, then it is easy to show:

$$\mathbf{H}\boldsymbol{x} = \boldsymbol{x} - 2\frac{\boldsymbol{v}\boldsymbol{v}^\top \boldsymbol{x}}{\|\boldsymbol{v}\|^2} \tag{32}$$

$$= \boldsymbol{x} - 2\boldsymbol{v}^\top \boldsymbol{x}\frac{\boldsymbol{v}}{\|\boldsymbol{v}\|^2} \tag{33}$$

$$= \boldsymbol{x} - 2\langle \boldsymbol{x}, \hat{\boldsymbol{v}}\rangle \hat{\boldsymbol{v}} \ . \tag{34}$$

To efficiently compute the Householder transformation, we do not need to revert the Householder matrix $\mathbf{H}$ and apply $\mathbf{H}\boldsymbol{x}$ to complete one Householder transformation. (32) can be used to drastically reduce the computational cost and thus make the Householder flows more efficient.

D Experimental Results

We follow the toy example in [Ghosh et al., 2016] to consider the binary classification task, in which we uniformly sample 10 data points in seperate distributions:$[-3,-1]\times[-3,-1]$ and $[1,3]\times[1,3]$. A one-layer BNNs with 30 hidden units is employed, we use at most 200 epochs for PBP_MV, SVGD and S^{2}VGD. The posterior prediction density is plotted in Figure D. The S^{2}VGD performs better than other models, because it is more similar to the ground truth and has a balanced posterior density. PBP_MV employed the structure information but is a little unbalanced. The SVGD is more unbalanced compared with PBP_MV. For non-linear regression, we follow the experiment setup of [Louizos and Welling, 2016, Sun et al., 2017]. We randomly generated 20 points as input x_n, in which 12 points are sampled from Uniform(0,0.6), and 8 points are sampled from Uniform(0.8,1). The output y_n is corresponding to x_n, and $y_n = x_n + \epsilon_n + \sin(4(x_n+\epsilon_n)) + sin(13(x_n+\epsilon_n))$, where $\epsilon_n \sim \mathcal{N}(0, 0.0009)$. We fit a one-layer ReLU neural network with 100 hidden units. We run PBP_MV, SVGD and S^{2}VGD for at most 1500 epochs.

Compared with other methods, S^{2}VGD captures uncertainty on two-sides using its variance, but other methods only capture part of the uncertainty.

E Experimental setting

We discuss the hyper-parameter settings for S^{2}VGD, and then provide their values for our experiments. All experiments are conducted on a single TITAN X GPU.

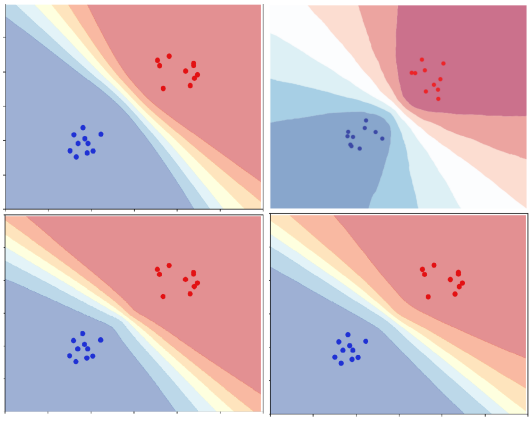

Figure 5: Binary classification on the synthetic dataset, Top left is SVGD, top right is the ground truth, bottom left is PBP_MV and bottom right is ours. The dots indicate both training and testing data points, different colors illustrates the prediction density, and the higher confidence the model has, the deeper the color will be.

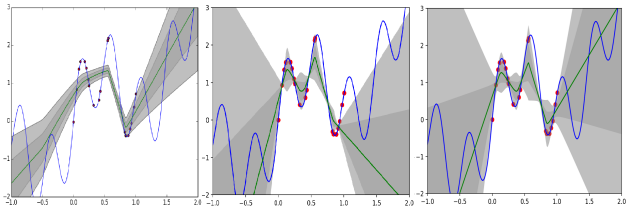

Figure 6: Regression on synthetic dataset. From the left to the right are PBP_MV, SVGD, S^{2}VGD. The blue line is the ground truth. The light grey area shows the ± 3 standard derivation confidence intervals; The green line is the mean of predictions.

E.1 Discussion of Hyper-parameters

Step Size The step size ϵ for SVGD and S^{2}VGD corresponds to the optimization counterparts. A block decay strategy is used on several datasets, it decreases by the stepsize by half every L epochs.

Mini-batch Size The gradient at step t is evaluated on a batch of data $\mathcal{S}_t$. For small datasets, the batch size can be set to the training sample size $|\mathcal{S}_t| = N$, giving the true gradient for each step. For large datasets, a stochastic gradient evaluated from a mini-batch of size $|\mathcal{S}_t| < N$ is used to

Variance of Gaussian Prior The prior distributions on the parameterized weights of DNNs are Gaussian, with mean 0 and variance σ^2. The variance of this Gaussian distribution determines the prior belief of how strongly these weights should concentrate on 0. This setting depends on user perception of the amount variability existing in the data. A larger variance in the prior leads to a wider range of weight choices, thus higher uncertainty. The weight decay value of ℓ_2 reg-

Table 3: Hyper-parameter settings for policy gradient experiments.

Datasets	`CartpoleSwingup`	`DoublePendulum`	`Cartpole`
Batch Size	5000	5000	5000
Step Size	5×10^{-3}	5×10^{-3}	5×10^{-3}
#Episodes	1000	1000	100
Discount	0.99	0.99	0.99
Temperature	[7,8,9,10,11]	[7,8,9,10,11]	[7,8,9,10,11]
Network	[25, 10]	[25, 10]	[25, 10]
Variance in prior	0.01	0.01	0.01

ularization in stochastic optimization is related to the prior variance in SVGD and S^2VGD.

Number of Particles The number of particles M, is employed to approximate the posterior. SVGD and S^2VGD represents the posterior approximately in terms of a set of particles (samples), and is endowed with guarantees on the approximation accuracy when the number of particles is exactly infinity [Liu, 2017]. The number of particles balanced posterior approximation accuracy and computational cost.

Number of Householder Transformations Instead of maintaining a full orthogonal matrix, we approximate it employing householder flows containing K Householder transformations. According to Lemma 4, the orthogonal matrix can be **exactly** represented by a series of Householder transformations, when K is the degree. See details in Supplementary D. In experiments, the number of Householder transformations balanced orthogonal matrix approximation accuracy and computational cost.

E.2 Settings in Our Experiments

The hyper-parameter settings of SVGD and S^2VGD on each dataset is specified in Table 5 for MNIST, Table 4 for contextual bandits and Table 3 for reinforcement learning.

Table 4: Hyper-parameter settings for contextual bandits.

Datasets	`Mushroom`	`Yahoo!Today`
Batch Size	64	100
Step Size	10^{-3}	10^{-3}
#Trial	5×10^{4}	4.5×10^{7}
Network (hidden layers)	[50]	[50]
Variance in prior	1	1

Table 5: Hyper-parameter settings for MNIST.

Datasets	`MNIST`	
Batch Size	100	100
Step Size	5×10^{-4}	5×10^{-4}
#Epoch	150	150
RMSProp	0.99	0.99
Network (hidden layers)	[400, 400]	[800, 800]
Variance in prior	1	1