

# Using Loss Surface Geometry for Practical Bayesian Deep Learning

Andrew Gordon Wilson

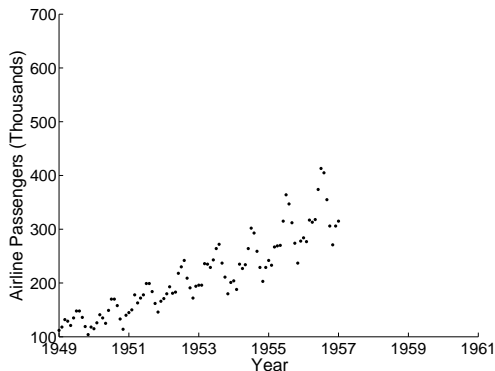
<https://cims.nyu.edu/~andrewgw>  
New York University

Bayesian Deep Learning Workshop  
Advances in Neural Information Processing Systems  
December 13, 2019

**Collaborators:**

**Pavel Izmailov, Wesley Maddox, Polina Kirichenko, Timur Garipov, Dmitry Vetrov**

# Model Selection



Which model should we choose?

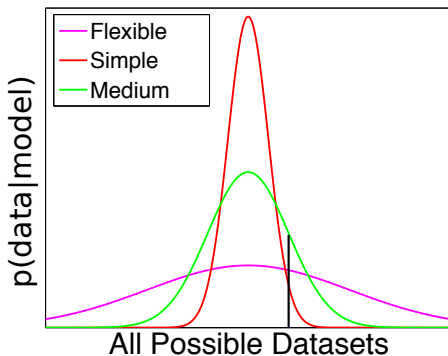
**(1):**  $f_1(x) = a_0 + a_1x$

**(2):**  $f_2(x) = \sum_{j=0}^3 a_j x^j$

**(3):**  $f_3(x) = \sum_{j=0}^{10^4} a_j x^j$

# How do we learn?

- ▶ The ability for a system to learn is determined by its *support* (which solutions are a priori possible) and *inductive biases* (which solutions are a priori likely).
- ▶ An influx of new *massive* datasets provide great opportunities to automatically learn rich statistical structure, leading to new scientific discoveries.



# Bayesian Deep Learning

## Why?

- ▶ A powerful framework for model construction and understanding generalization
- ▶ Uncertainty representation and calibration (crucial for decision making)
- ▶ *Better point estimates*
- ▶ Interpretably incorporate prior knowledge and domain expertise
- ▶ It was the most successful approach at the end of the second wave of neural networks (Neal, 1998).
- ▶ Neural nets are much less mysterious when viewed through the lens of probability theory.

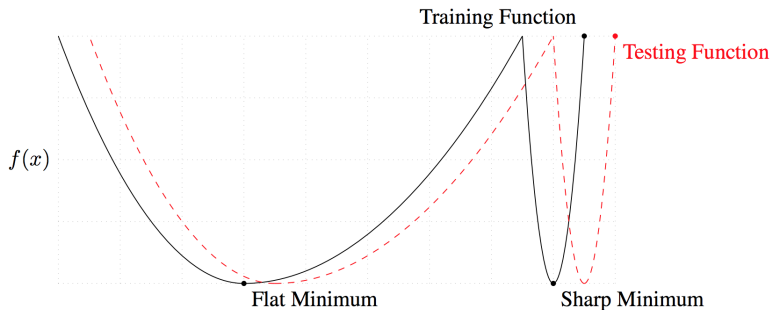
## Why not?

- ▶ Can be computationally intractable (but doesn't have to be).
- ▶ Can involve a lot of moving parts (but doesn't have to).

There has been exciting progress in the last year addressing these limitations.



# Wide Optima Generalize Better



*Keskar et. al (2017)*

- **Bayesian integration** will give very different predictions in deep learning especially!

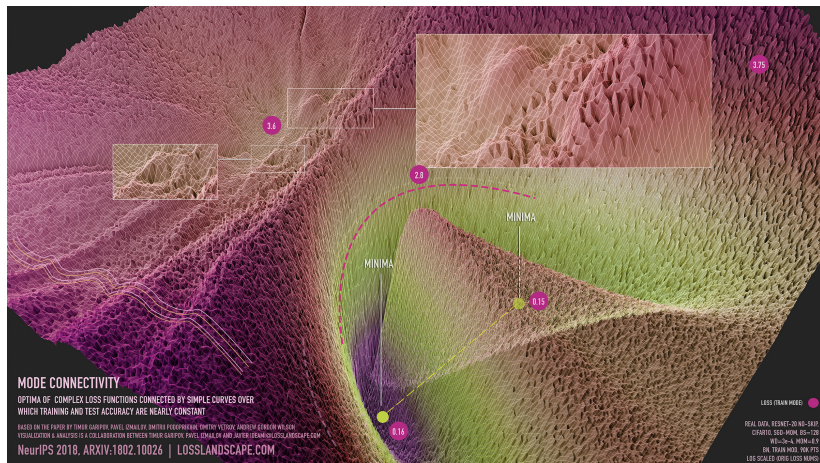
# Bayesian Deep Learning

**Sum rule:**  $p(x) = \sum_y p(x, y)$ . **Product rule:**  $p(x, y) = p(x|y)p(y) = p(y|x)p(x)$ .

$$p(y|x_*, \mathbf{y}, X) = \int p(y|x_*, \mathbf{w})p(\mathbf{w}|\mathbf{y}, X)d\mathbf{w} . \quad (1)$$

- ▶ Think of each setting of  $\mathbf{w}$  as a different model. Eq. (1) is a *Bayesian model average*, an average of infinitely many models weighted by their posterior probabilities.
- ▶ Automatically calibrated complexity even with highly flexible models.
- ▶ Can view classical training as using an approximate posterior  $q(\mathbf{w}|\mathbf{y}, X) = \delta(\mathbf{w} = \mathbf{w}_{\text{MAP}})$ .
- ▶ Typically more interested in the induced distribution over **functions** than in parameters  $\mathbf{w}$ . Can be hard to have intuitions for priors on  $p(\mathbf{w})$ .

# Mode Connectivity

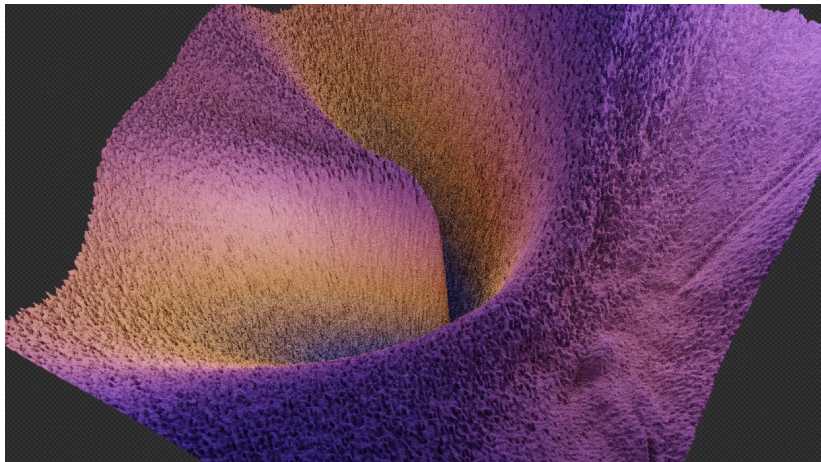


## Loss Surfaces, Mode Connectivity, and Fast Ensembling of DNNs

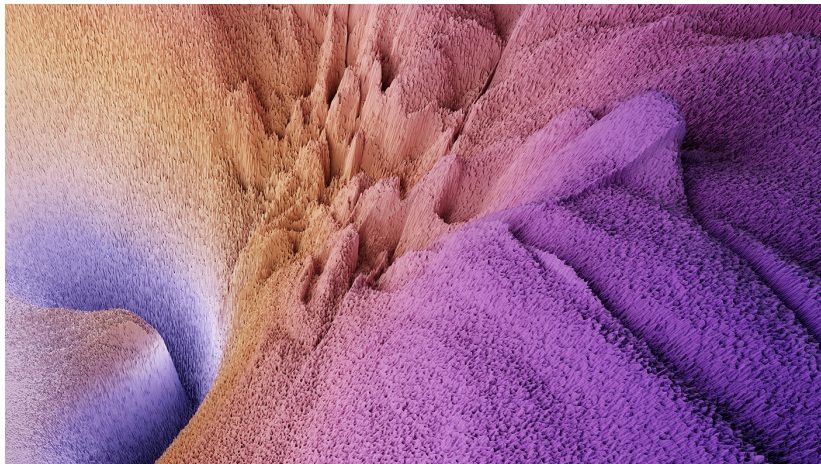
T. Garipov, P. Izmailov, D. Podoprikin, D. Vetrov, A.G. Wilson

NeurIPS 2018

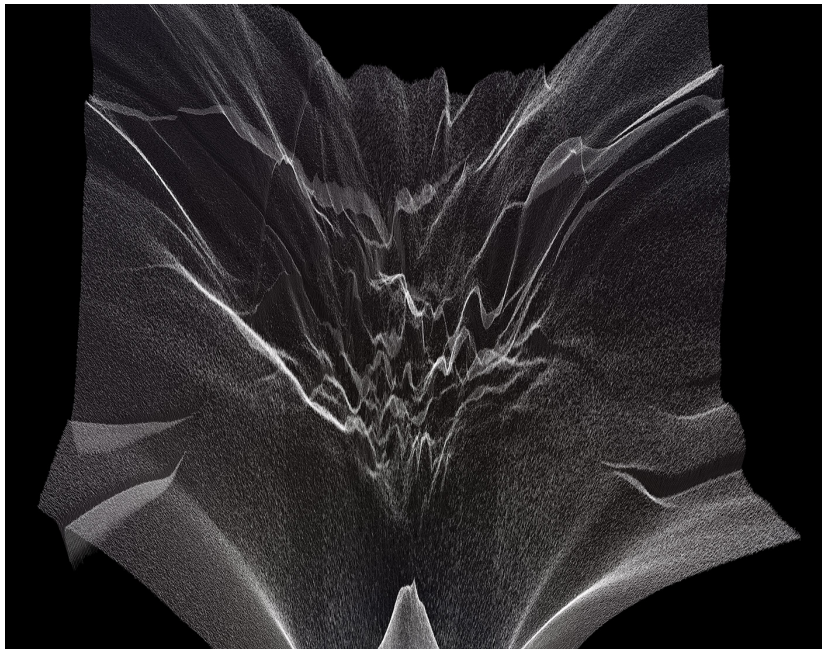
# Mode Connectivity



# Mode Connectivity

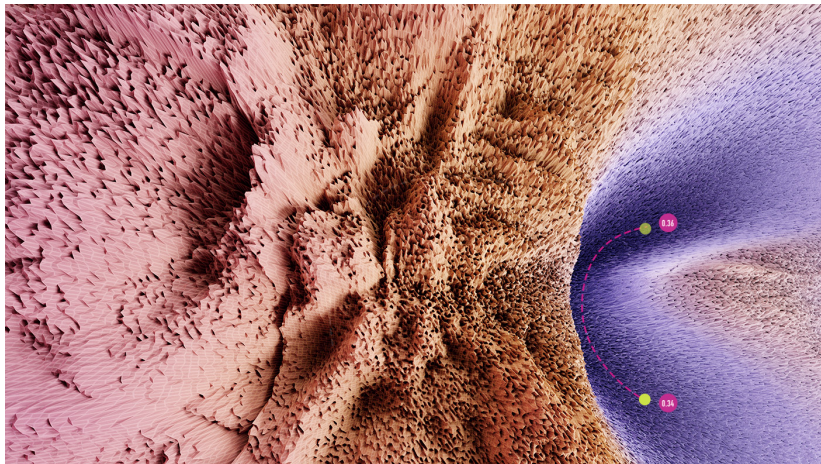


# Mode Connectivity



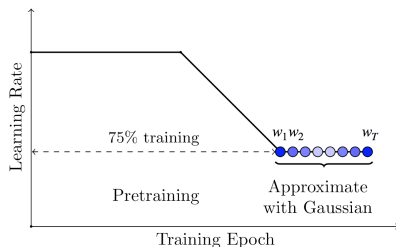


# Mode Connectivity



# Uncertainty Representation with SWAG

1. Leverage theory that shows SGD with a constant learning rate is approximately sampling from a Gaussian distribution.
2. Compute first *two* moments of SGD trajectory (SWA computes just the first).
3. Use these moments to construct a Gaussian approximation in weight space.
4. Sample from this Gaussian distribution, pass samples through predictive distribution, and form a Bayesian model average.



$$p(y_*|\mathcal{D}) \approx \frac{1}{J} \sum_{j=1}^J p(y_*|w_j), \quad w_j \sim q(w|\mathcal{D}), \quad q(w|\mathcal{D}) = \mathcal{N}(\bar{w}, K)$$

$$\bar{w} = \frac{1}{T} \sum_t w_t, \quad K = \frac{1}{2} \left( \frac{1}{T-1} \sum_t (w_t - \bar{w})(w_t - \bar{w})^T + \frac{1}{T-1} \sum_t \text{diag}(w_t - \bar{w})^2 \right)$$

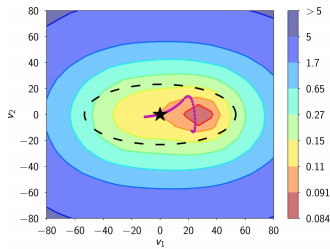
## A Simple Baseline for Bayesian Uncertainty in Deep Learning

W. Maddox, P. Izmailov, T. Garipov, D. Vetrov, A.G. Wilson

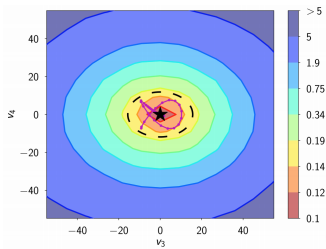
NeurIPS 2019



# Trajectory in PCA Subspace

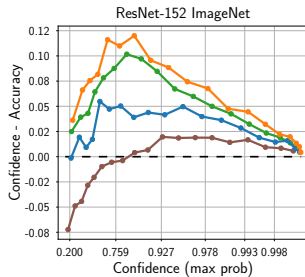
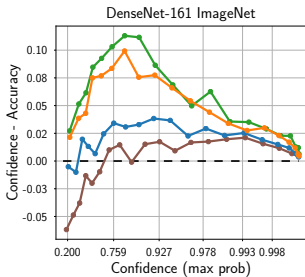
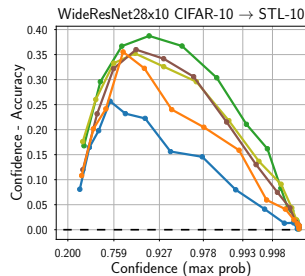
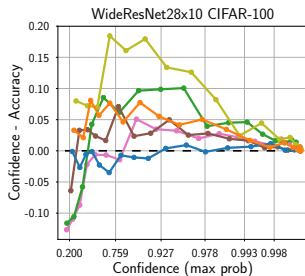


★ SWA      — Trajectory (proj)  
-- SWAG  $3\sigma$  region



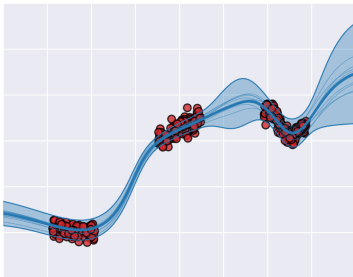
★ SWA      — Trajectory (proj)  
-- SWAG  $3\sigma$  region

# Uncertainty Calibration

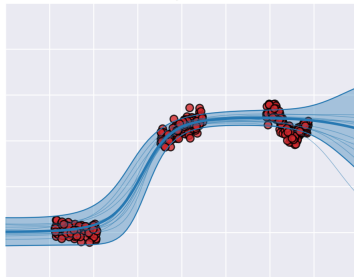


# SWAG Regression Uncertainty

SWAG



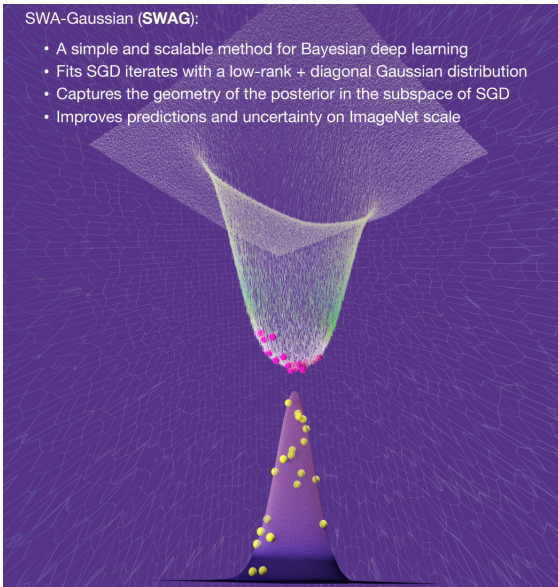
Full Space VI



# SWAG Visualization

SWA-Gaussian (**SWAG**):

- A simple and scalable method for Bayesian deep learning
- Fits SGD iterates with a low-rank + diagonal Gaussian distribution
- Captures the geometry of the posterior in the subspace of SGD
- Improves predictions and uncertainty on ImageNet scale



# Subspace Inference for Bayesian Deep Learning

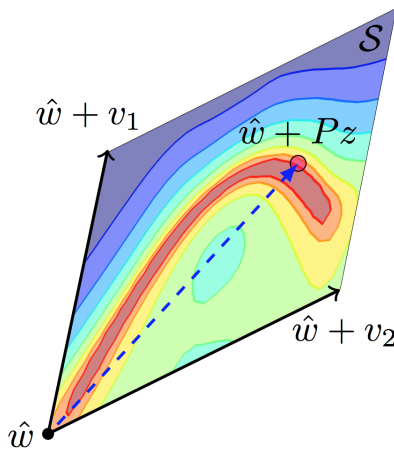
A modular approach:

- ▶ Construct a subspace of a network with a high dimensional parameter space
- ▶ Perform inference directly in the subspace
- ▶ Sample from approximate posterior for Bayesian model averaging

**We can approximate the posterior of a WideResNet with 36 million parameters in a 5D subspace and achieve state-of-the-art results!**

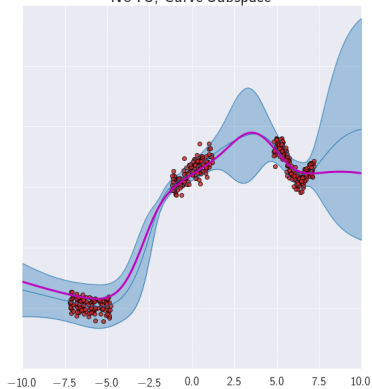
# Subspace Construction

- ▶ Choose shift  $\hat{w}$  and basis vectors  $\{d_1, \dots, d_k\}$ .
- ▶ Define subspace  $S = \{w | w = \hat{w} + z_1 d_1 + z_k d_k\}$ .
- ▶ Likelihood  $p(\mathcal{D}|z) = p_M(\mathcal{D}|w = \hat{w} + Pz)$ .
- ▶ Posterior inference  $p(z|\mathcal{D}) \propto p(\mathcal{D}|z)p(z)$ .

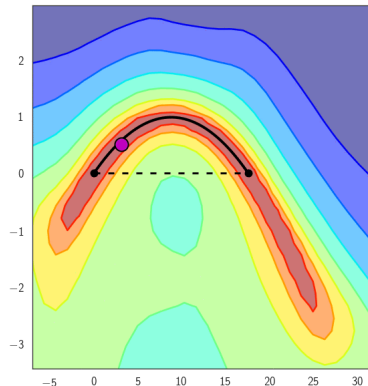


# Curve Subspace Traversal

Predictive Distribution  
NUTS, Curve Subspace

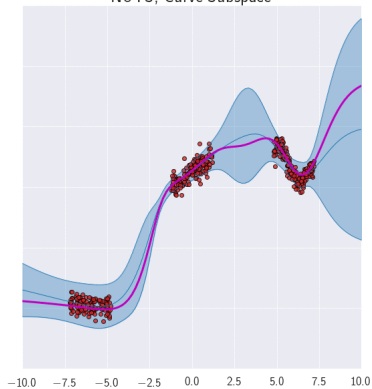


Posterior Log-Density  
NUTS, Curve Subspace

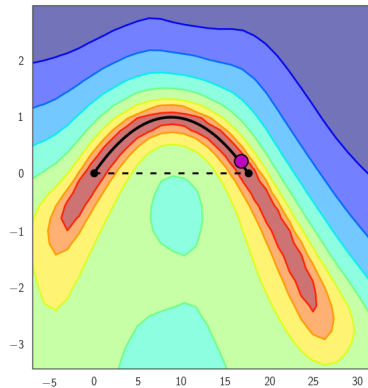


# Curve Subspace Traversal

Predictive Distribution  
NUTS, Curve Subspace



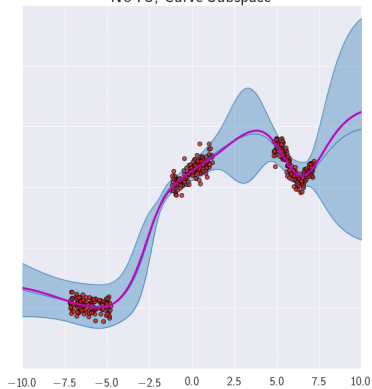
Posterior Log-Density  
NUTS, Curve Subspace



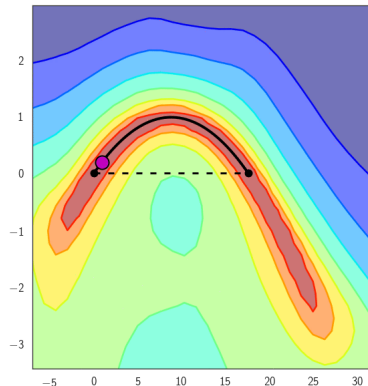


# Curve Subspace Traversal

Predictive Distribution  
NUTS, Curve Subspace

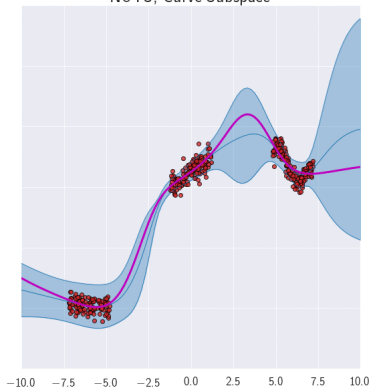


Posterior Log-Density  
NUTS, Curve Subspace

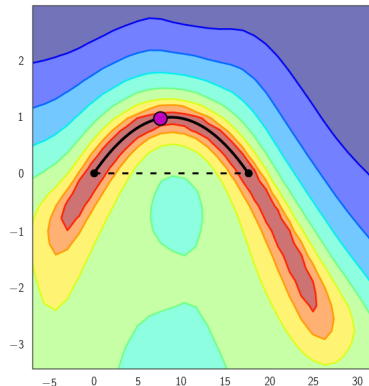


# Curve Subspace Traversal

Predictive Distribution  
NUTS, Curve Subspace

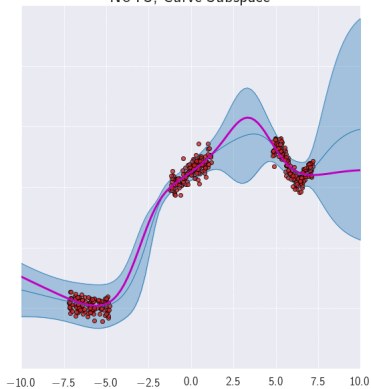


Posterior Log-Density  
NUTS, Curve Subspace

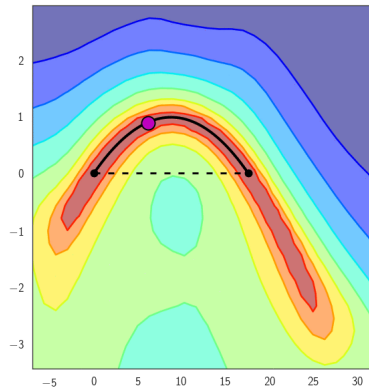


# Curve Subspace Traversal

Predictive Distribution  
NUTS, Curve Subspace

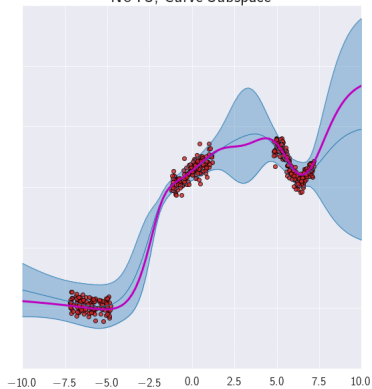


Posterior Log-Density  
NUTS, Curve Subspace

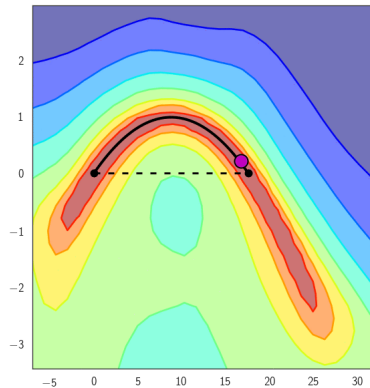


# Curve Subspace Traversal

Predictive Distribution  
NUTS, Curve Subspace

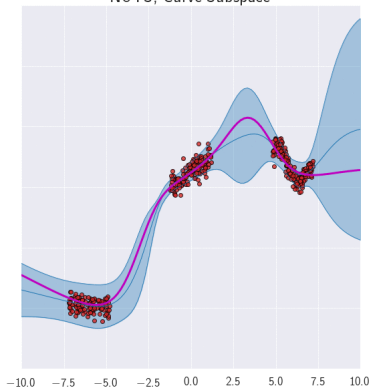


Posterior Log-Density  
NUTS, Curve Subspace

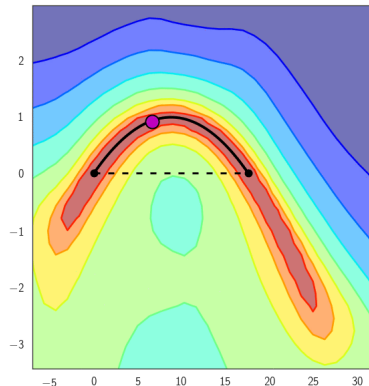


# Curve Subspace Traversal

Predictive Distribution  
NUTS, Curve Subspace

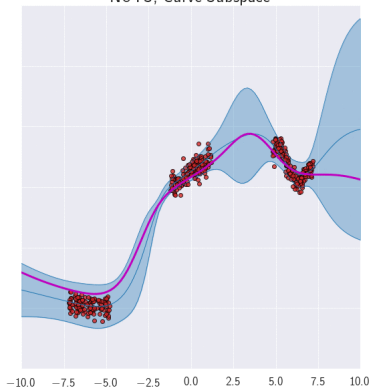


Posterior Log-Density  
NUTS, Curve Subspace

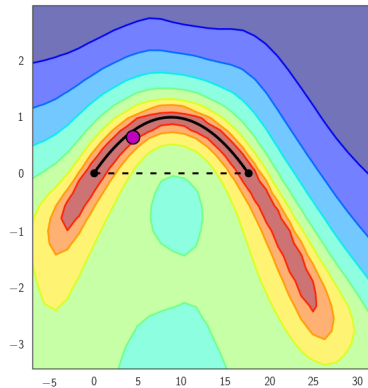


# Curve Subspace Traversal

Predictive Distribution  
NUTS, Curve Subspace

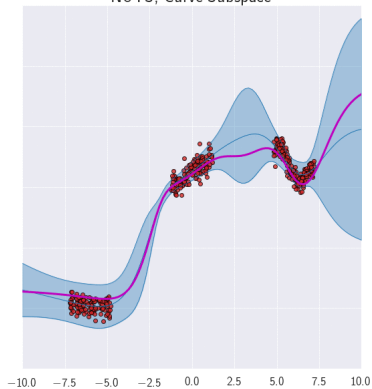


Posterior Log-Density  
NUTS, Curve Subspace

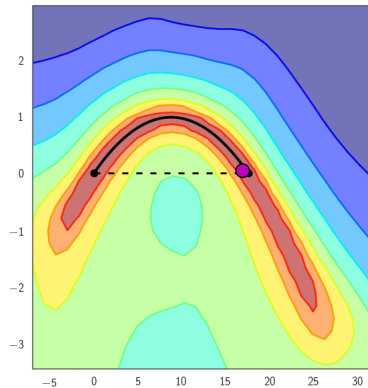


# Curve Subspace Traversal

Predictive Distribution  
NUTS, Curve Subspace

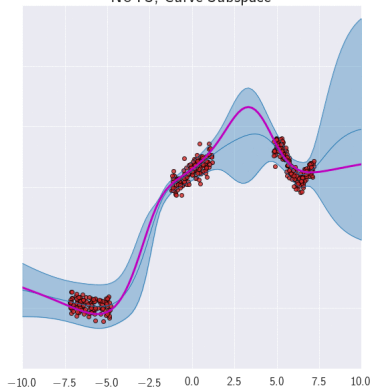


Posterior Log-Density  
NUTS, Curve Subspace

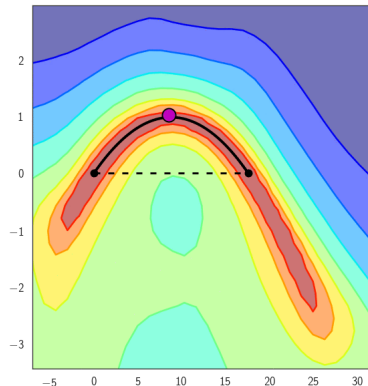


# Curve Subspace Traversal

Predictive Distribution  
NUTS, Curve Subspace



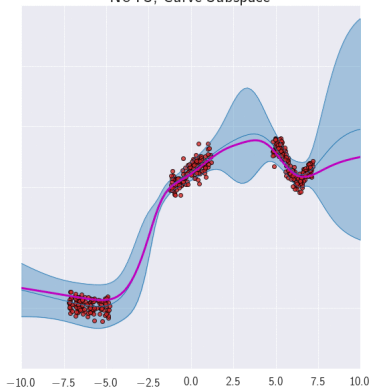
Posterior Log-Density  
NUTS, Curve Subspace



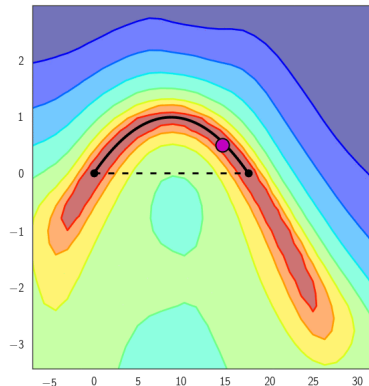


# Curve Subspace Traversal

Predictive Distribution  
NUTS, Curve Subspace

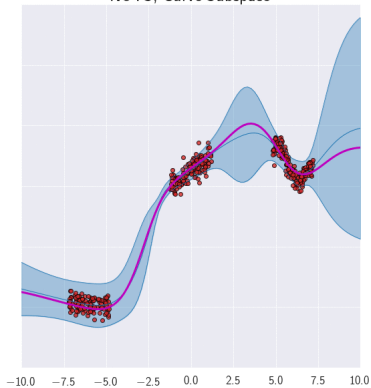


Posterior Log-Density  
NUTS, Curve Subspace

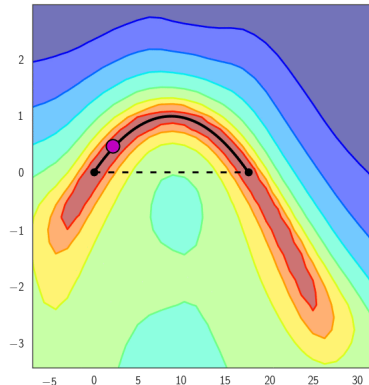


# Curve Subspace Traversal

Predictive Distribution  
NUTS, Curve Subspace

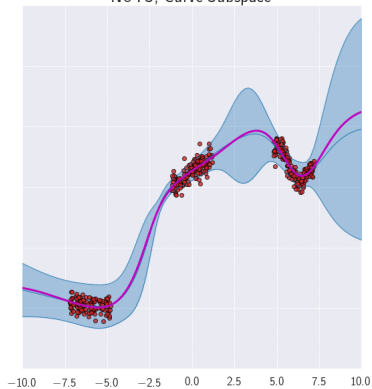


Posterior Log-Density  
NUTS, Curve Subspace

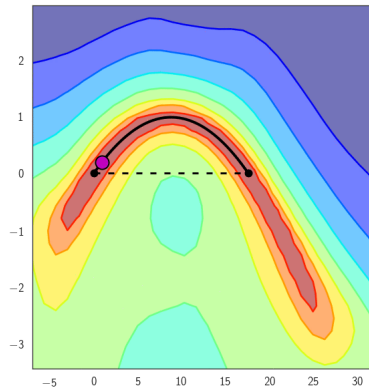


# Curve Subspace Traversal

Predictive Distribution  
NUTS, Curve Subspace

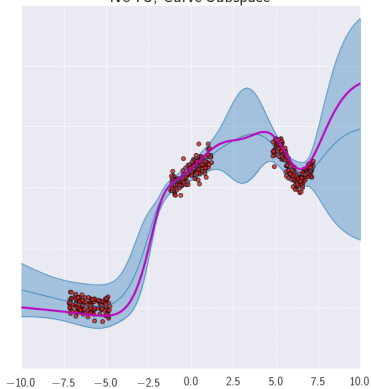


Posterior Log-Density  
NUTS, Curve Subspace

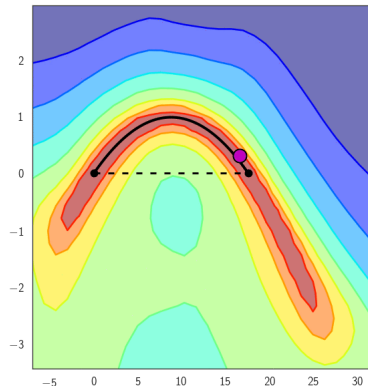


# Curve Subspace Traversal

Predictive Distribution  
NUTS, Curve Subspace

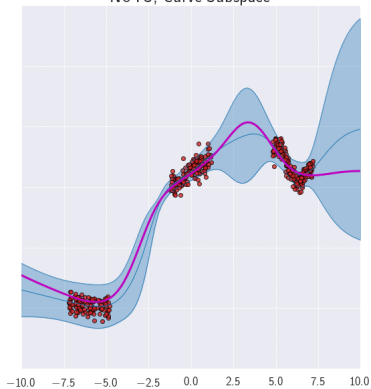


Posterior Log-Density  
NUTS, Curve Subspace

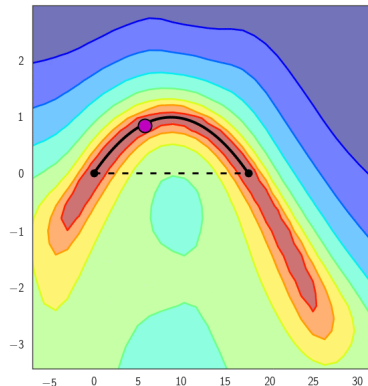


# Curve Subspace Traversal

Predictive Distribution  
NUTS, Curve Subspace

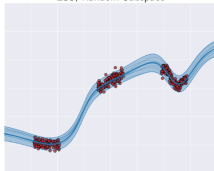


Posterior Log-Density  
NUTS, Curve Subspace

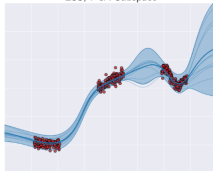


# Subspace Comparison (Regression)

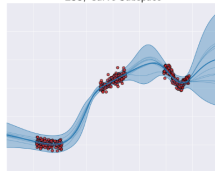
Predictive Distribution  
ESS, Random Subspace



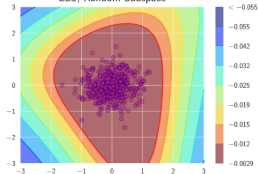
Predictive Distribution  
ESS, PCA Subspace



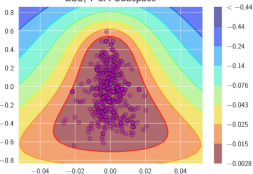
Predictive Distribution  
ESS, Curve Subspace



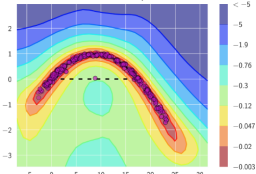
Posterior log-density  
ESS, Random Subspace



Posterior log-density  
ESS, PCA Subspace

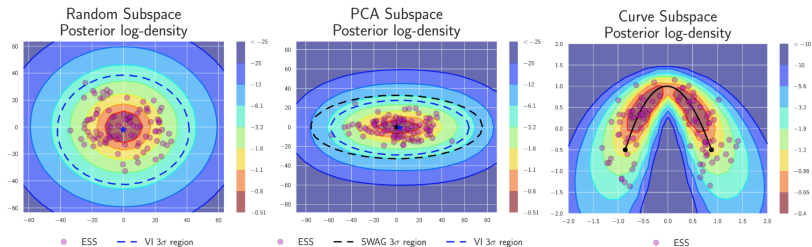


Posterior log-density  
ESS, Curve Subspace



# Subspace Comparison (Classification)

## Accuracy and NLL on CIFAR-100



	SGD	Random	PCA	Curve
NLL	$0.946 \pm 0.001$	$0.686 \pm 0.005$	$0.665 \pm 0.004$	0.646
Accuracy (%)	$78.50 \pm 0.32$	$80.17 \pm 0.03$	$80.54 \pm 0.13$	81.28

*Bayesian methods also lead to better point predictions in deep learning!*

## Subspace Inference for Bayesian Deep Learning

P. Izmailov, W. Maddox, P. Kirichenko, T. Garipov, D. Vetrov, A.G. Wilson

UAI 2019

# Conclusions

- ▶ Neural networks represent many compelling solutions to a given problem, and a very underspecified by the available data. This is the perfect situation for **Bayesian marginalization**.
- ▶ Even if we cannot perfectly express our priors, or perform full Bayesian inference, we can try our best and get much better point predictions as well as improved calibration. We can view standard training as an impoverished Bayesian approximation.
- ▶ By exploiting information about the loss geometry in training, we can scale Bayesian neural networks to ImageNet with improvements in accuracy and calibration, and essentially no runtime overhead.

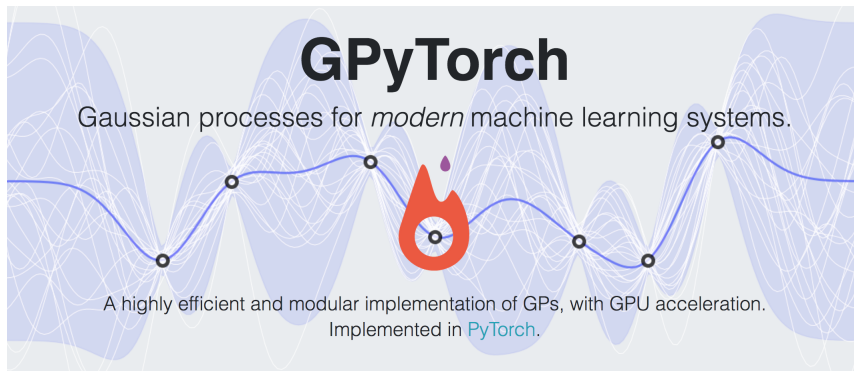


**There is a postdoc opening in my group!**

Join an energetic and ambitious team of scientists in New York City, looking to address big open questions in core machine learning.

# Scalable Gaussian Processes

- ▶ Run **exact** GPs on millions of points in minutes.
- ▶ Outperforms stand-alone deep neural networks by learning **deep kernels**.
- ▶ **Implemented in our new library GPyTorch:** `gpytorch.ai`

The background of the slide features a visualization of Gaussian Processes. It shows a solid blue line representing the mean function, with several black dots indicating observed data points. Surrounding this mean line are numerous light blue, wavy lines representing the uncertainty or predictive distribution at various points. The overall aesthetic is clean and technical, with a light blue and white color scheme.

## GPyTorch

Gaussian processes for *modern* machine learning systems.

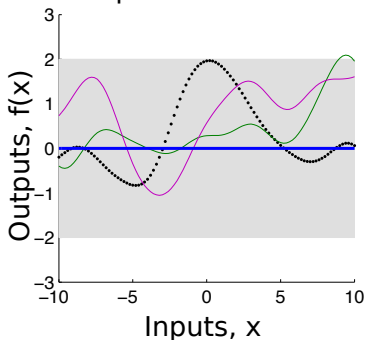
A highly efficient and modular implementation of GPs, with GPU acceleration.  
Implemented in [PyTorch](#).

# Gaussian processes: a function space view

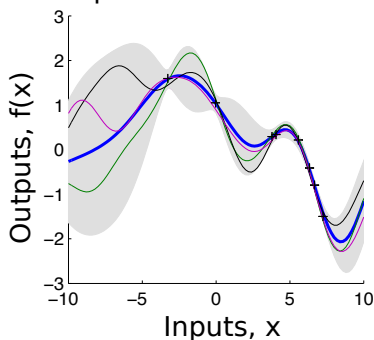
**Gaussian processes provide an intuitive function space perspective on learning and generalization.**

$$\overbrace{p(f(x)|\mathcal{D})}^{\text{GP posterior}} \propto \overbrace{p(\mathcal{D}|f(x))}^{\text{Likelihood}} \overbrace{p(f(x))}^{\text{GP prior}}$$

Sample Prior Functions



Sample Posterior Functions



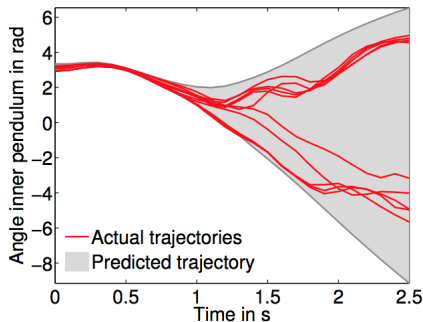
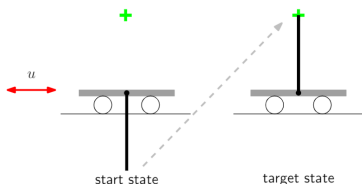
# BoTorch: Bayesian Optimization in PyTorch

- ▶ Probabilistic active learning
- ▶ Black box objectives, hyperparameter tuning, A/B testing, global optimization.



# Probabilistic Reinforcement Learning

**Robust, sample efficient online decision making under uncertainty.**



# References

- Stochastic Weight Averaging in PyTorch: <https://pytorch.org/blog/stochastic-weight-averaging-in-pytorch/>.
- Semi-supervised Learning with Normalizing Flows. *To appear*.
- W. Maddox, P. Izmailov, T. Garipov, D. Vetrov, A.G. Wilson. A Simple Baseline for Bayesian Uncertainty in Deep Learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
- K. A. Wang, G. Pleiss, J. Gardner, S. Tyree, K. Weinberger, A.G. Wilson. Exact Gaussian Processes on a Million Data Points. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
- P. Izmailov, W. Maddox, P. Kirichenko, T. Garipov, D. Vetrov, A.G. Wilson. Subspace Inference for Bayesian Deep Learning. *Uncertainty In Artificial Intelligence (UAI)*, 2019.
- G. Yang, T. Chen, P. Kirichenko, J. Bai, A.G. Wilson, C. de Sa. SWALP: Stochastic Weight Averaging in Low Precision Training. *International Conference on Machine Learning (ICML)*, 2019.
- W. Herlands, D.B. Neill, H. Nickisch, A.G. Wilson. Change Surfaces for Expressive Multidimensional Changepoints and Counterfactual Prediction. *Journal of Machine Learning Research (JMLR)*, 2019.
- B. Athiwaratkun, M. Finzi, P. Izmailov, A.G. Wilson. There are Many Consistent Explanations of Unlabeled Data: Why You Should Average. *International Conference on Learning Representations (ICLR)*, 2019.
- T. Garipov, P. Izmailov, D. Podoprikin, D. Vetrov, A.G. Wilson. Loss Surfaces, Mode Connectivity, and Fast Ensembling of DNNs. *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.
- J. Gardner, G. Pleiss, D. Bindel, K. Weinberger, A.G. Wilson. GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration. *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.
- C.E. Rasmussen and Z. Ghahramani. Occam's razor. *Advances in Neural Information Processing Systems (NeurIPS)*, 2001.
- D. MacKay. Information Theory, Inference, and Learning Algorithms. *Cambridge University Press*, 2003.
- C. Bishop. Pattern Recognition and Machine Learning. *Cambridge University Press*, 2006.
- P. Izmailov, D. Podoprikin, T. Garipov, D. Vetrov, A.G. Wilson. Averaging Weights Leads to Wider Optima and Better Generalization, *Uncertainty in Artificial Intelligence (UAI)*, 2018.

# References

- G. Pleiss, J. Gardner, K.Q. Weinberger, and A.G. Wilson. Constant time predictive distributions for Gaussian processes. *International Conference on Machine Learning (ICML)*, 2018.
- B. Athiwaratkun, A.G. Wilson, and A. Anandkumar. Probabilistic FastText. *Association for Computational Linguistics (ACL)*, 2018.
- B. Athiwaratkun and A.G. Wilson. Hierarchical Density Order Embeddings. *International Conference on Learning Representations (ICLR)*, 2018.
- Y. Saatchi and A.G. Wilson. Bayesian GAN. *Neural Information Processing Systems (NeurIPS)*, 2017.
- B. Athiwaratkun and A.G. Wilson. Multimodal Word Distributions. *Association for Computational Linguistics (ACL)*, 2017.
- M. Al-Shedivat, A.G. Wilson, Y. Saatchi, Z. Hu, and E.P. Xing. Learning Scalable Deep Kernels with Recurrent Structure. *Journal of Machine Learning Research (JMLR)*, 2017.
- A.G. Wilson, Z. Hu, R. Salakhutdinov, and E.P. Xing. Stochastic Variational Deep Kernel Learning. *Neural Information Processing Systems (NeurIPS)*, 2016.
- A.G. Wilson, Z. Hu, R. Salakhutdinov, and E.P. Xing. Deep kernel learning. *Artificial Intelligence and Statistics (AISTATS)*, 2016.
- A.G. Wilson, C. Dann, C.G. Lucas, and E.P. Xing. The human kernel. In *Neural Information Processing Systems (NeurIPS)*, 2015.
- A.G. Wilson and H. Nickisch. Kernel interpolation for scalable structured Gaussian processes (KISS-GP). *International Conference on Machine Learning (ICML)*, 2015. *International Conference on Machine Learning (ICML)*, 2015.
- A.G. Wilson. Covariance kernels for fast automatic pattern discovery and extrapolation with Gaussian processes. PhD Thesis, University of Cambridge. October 2014.