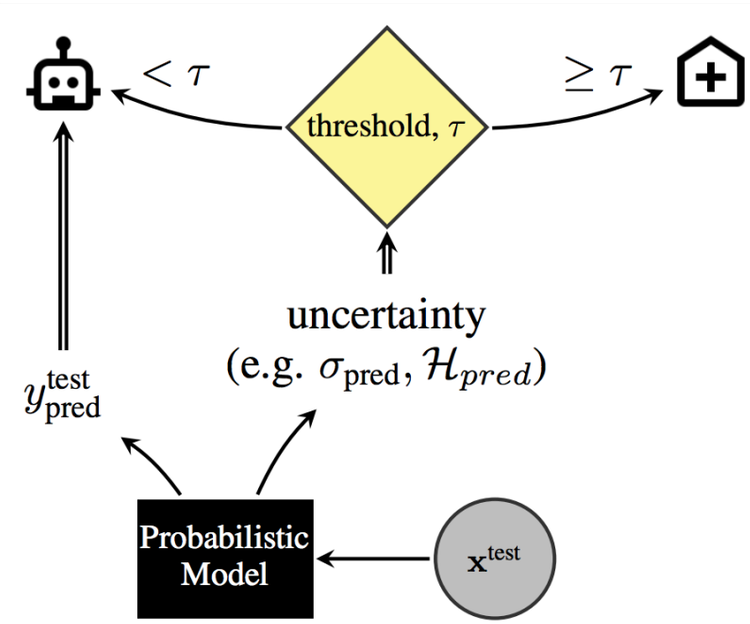
GAUSSIAN PROCESSES AND BAYESIAN NEURAL NETWORKS FOR DECISION-MAKING

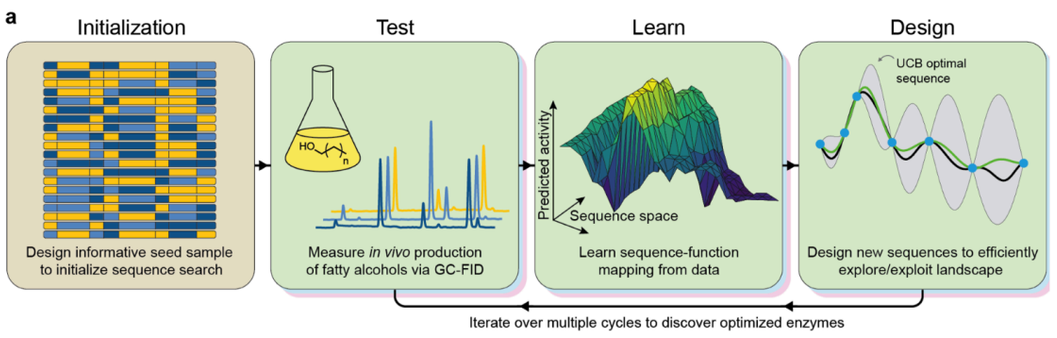
WESLEY MADDOX

ORGANIZATION

- Two motivating themes:
  - "How do we develop practical Bayesian neural networks?"
  - "How do we use Bayesian optimization to solve real problems?"



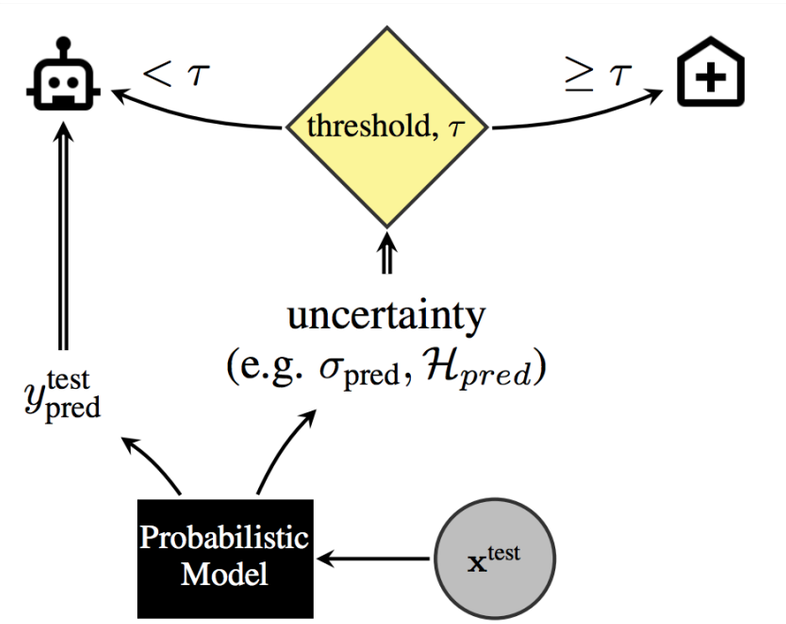
Filos et al, '20



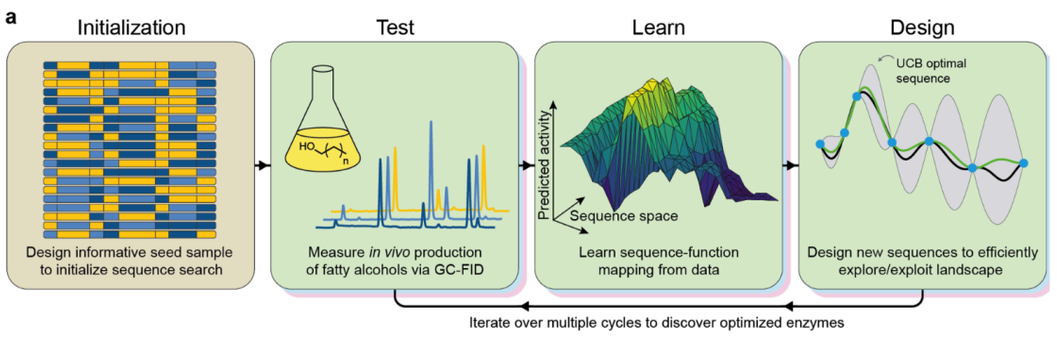
Greenhalgh et al, '21

ORGANIZATION

- Two motivating themes:
  - **"How do we develop practical Bayesian neural networks?"**
  - "How do we use Bayesian optimization to solve real problems?"



Filos et al, '20



Greenhalgh et al, '21

BAYESIAN INFERENCE

- Likelihood $p(\mathcal{D}|w) = p(y|f(x;w))$
- Prior $p(w)$ (possibly implicit)
- Posterior $p(w|\mathcal{D}) = \dfrac{p(\mathcal{D}|w)p(w)}{p(\mathcal{D})} \approx q(w|\mathcal{D})$
- Inference (Bayesian model averaging)

$$p(y^*|\mathcal{D}) = \mathbb{E}_{p(w|\mathcal{D})} p(y^*|w) \approx \frac{1}{K}\sum_{k=1}^{K} p(y^*|w_k) \quad w_k \sim q(w|\mathcal{D})$$

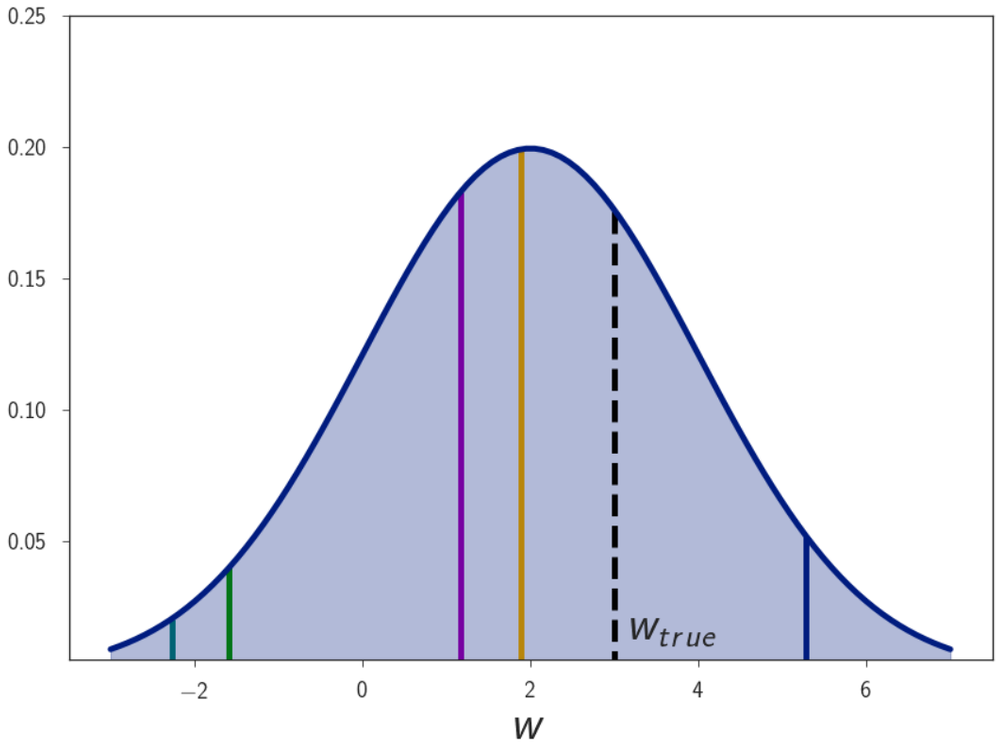
- Bayesian model comparison (marginal likelihood)

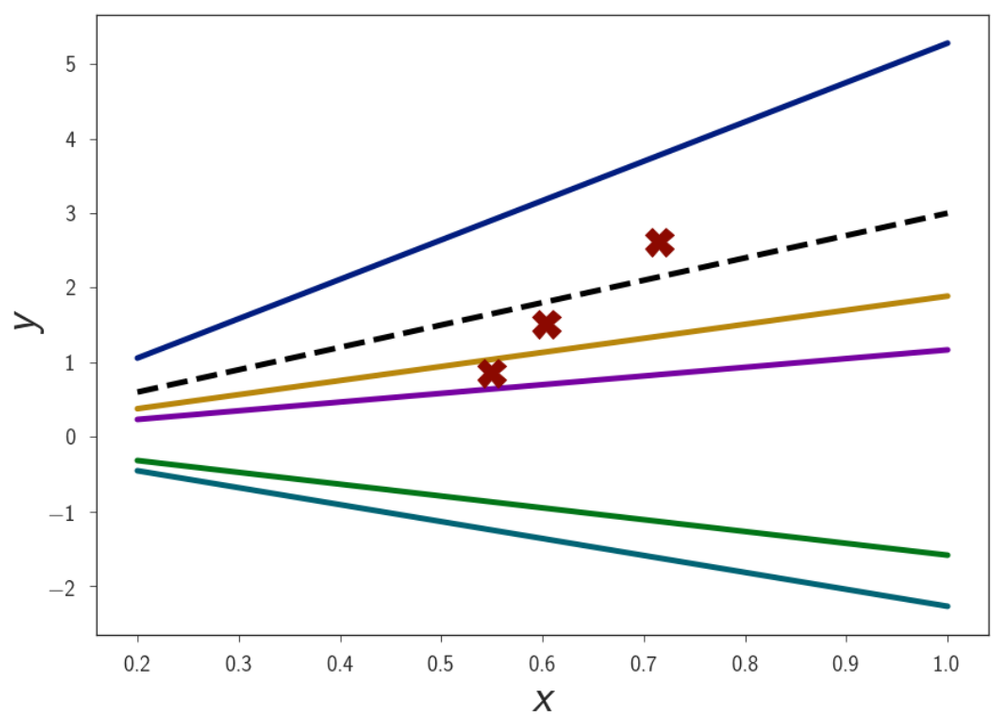
$$p(\mathcal{D}|\mathcal{M}) = \int p(\mathcal{D}|w,\mathcal{M})p(w)dw$$

BAYESIAN LEARNING

$$y = wx + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2)$$

Prior distribution $p(w)$ over parameters induces a range of potential functions



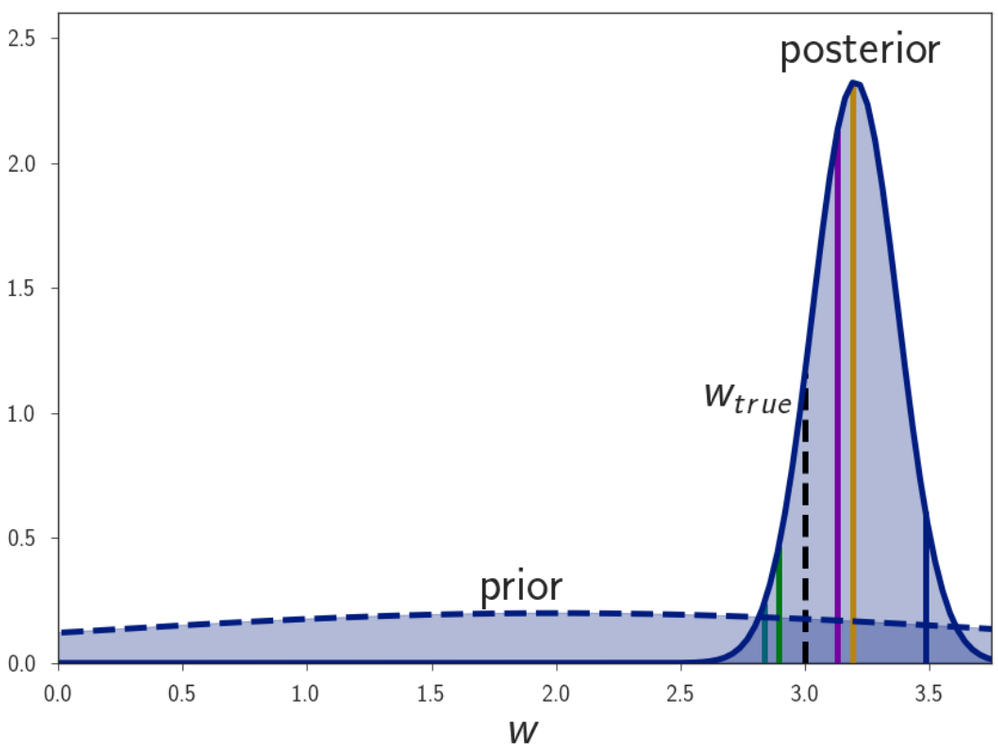


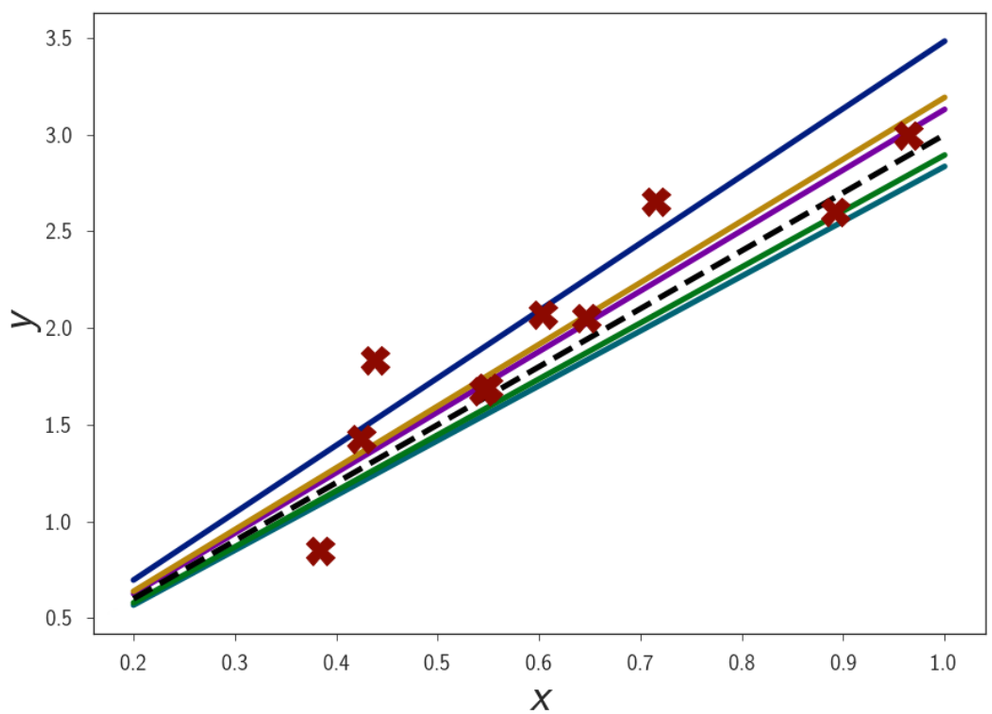
BAYESIAN LEARNING

Posterior distribution: $$p(w|D) = \frac{p(D|w)p(w)}{p(D)}$$

Computing the posterior in weight space produces a posterior over functions that is more confident about the data.

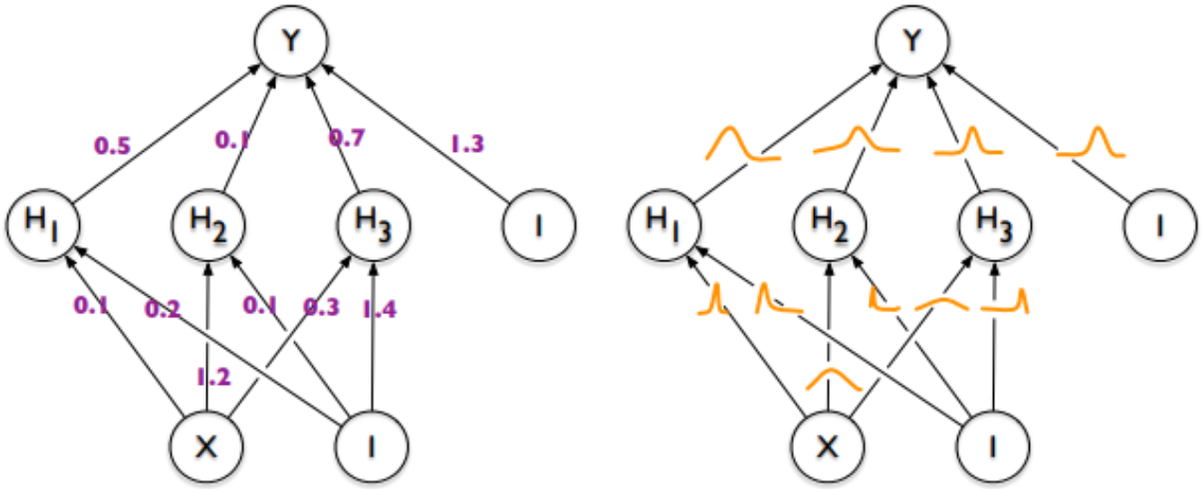
We can use the posterior over functions to **make decisions** (on new data, about trends, etc.).





BAYESIAN DEEP LEARNING

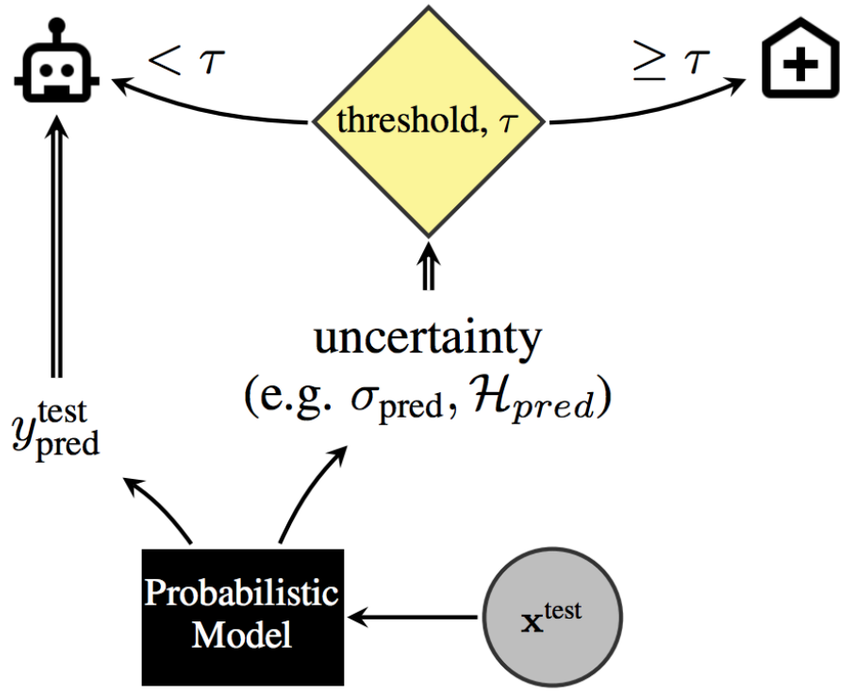
- In Bayesian deep learning we model posterior distribution over the weights of neural networks
- Often leads to better prediction and well-calibrated uncertainty



Standard DNN

Bayesian DNN

From Blundell et al, ‘15



From Filos et al, ‘19

BAYESIAN DEEP LEARNING: CHALLENGES

Bayesian inference for deep neural networks is extremely challenging

- Posterior is intractable
- Millions of parameters
- Large datasets
- Unclear which priors to use

$$p(w|D) = \frac{p(D|w)p(w)}{p(D)} = \frac{p(D|w)p(w)}{\int_{w'} p(D|w')p(w')dw'}$$

BAYESIAN DEEP LEARNING: CHALLENGES

Bayesian inference for deep neural networks is extremely challenging

- Posterior is intractable — Is the likelihood correct?
- Millions of parameters — What do these parameters mean?
- Large datasets — Can we run MCMC for 1 million steps on ImageNet??
- Unclear which priors to use — Is the prior correct?

$$p(w|D) = \frac{p(D|w)p(w)}{p(D)} = \frac{p(D|w)p(w)}{\int_{w'} p(D|w')p(w')dw'}$$

BAYESIAN DEEP LEARNING: CHALLENGES

Bayesian inference for deep neural networks is extremely challenging

- Posterior is intractable
- Millions of parameters
- Large datasets
- Unclear which priors to use

Is the likelihood correct? *Maybe**

What do these parameters mean?

Care about functions instead

Can we run MCMC for 1 million steps on ImageNet??

We don't need to

Is the prior correct?

Probably

$$p(w|D) = \frac{p(D|w)p(w)}{p(D)} = \frac{p(D|w)p(w)}{\int_{w'} p(D|w')p(w')dw'}$$

BAYESIAN DEEP LEARNING: CHALLENGES

Bayesian inference for deep neural networks is extremely challenging

- Posterior is intractable
- Millions of parameters
- Large datasets
- Unclear which priors to use

Is the likelihood correct? *Maybe**

What do these parameters mean?

Care about functions instead

Can we run MCMC for 1 million steps on ImageNet??

We don't need to

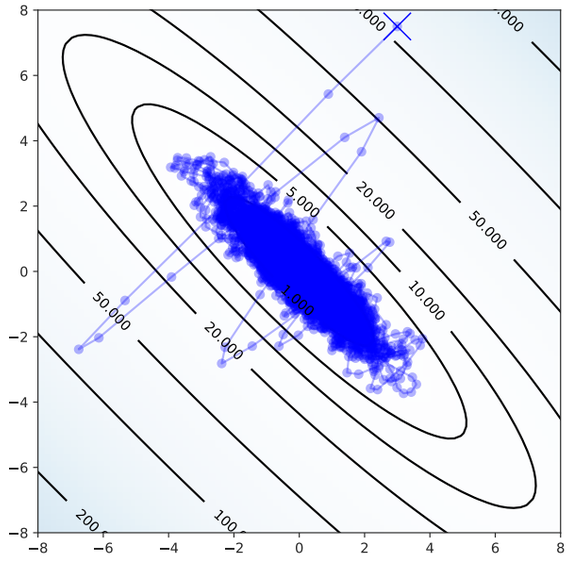
Is the prior correct?

Probably

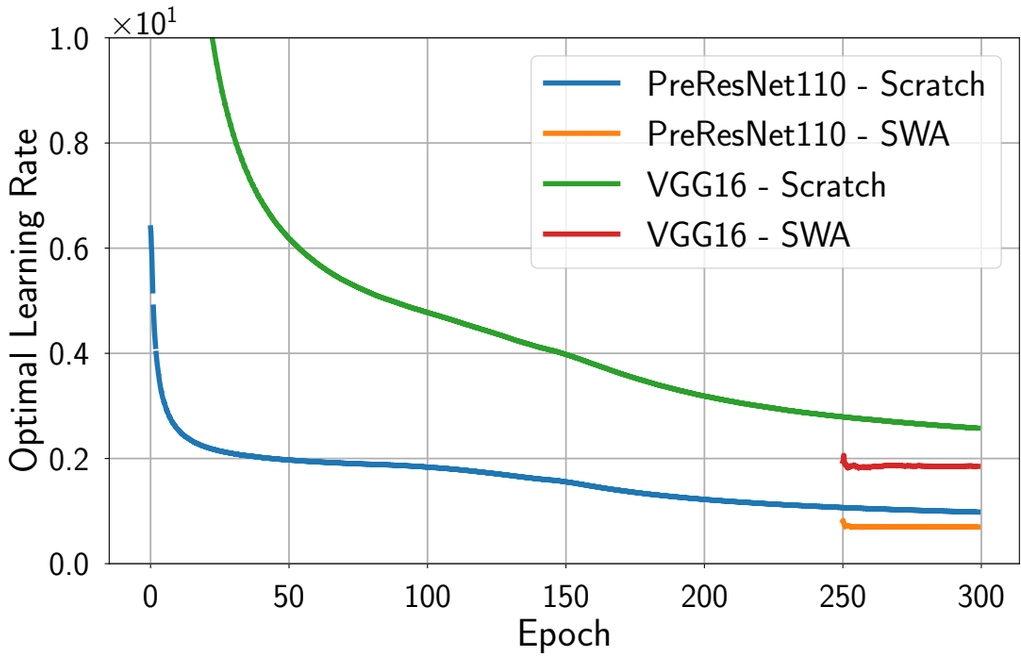
$$p(w|D) = \frac{p(D|w)p(w)}{p(D)} = \frac{p(D|w)p(w)}{\int_{w'} p(D|w')p(w')dw'}$$

SGD AS APPROXIMATE BAYESIAN INFERENCE – MANDT, ET AL, '17

- SGD with isotropic noise follows the shape of the posterior

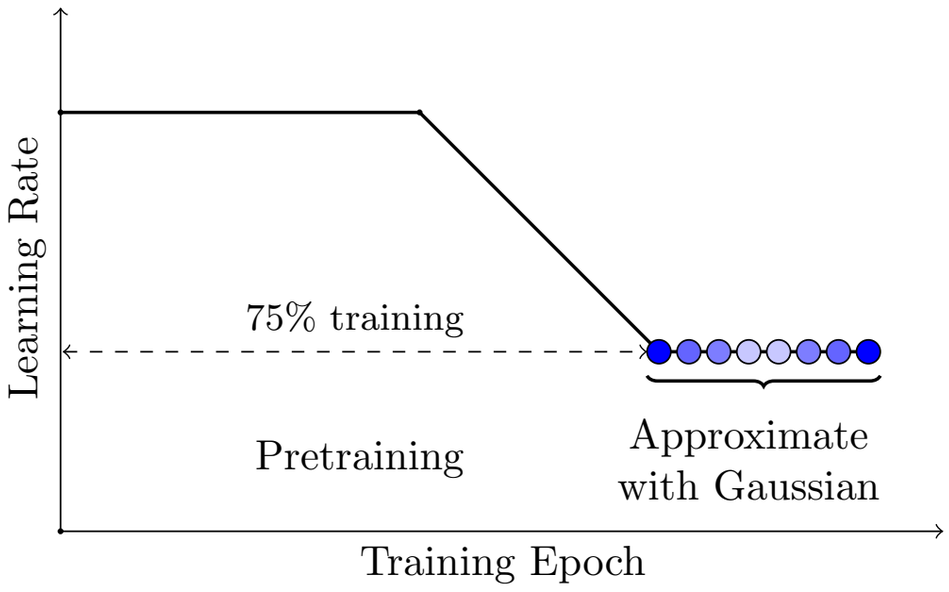


- Assumptions of analysis don't quite hold for DNNs
- But… we can use the same idea to approximate the posterior for DNNs

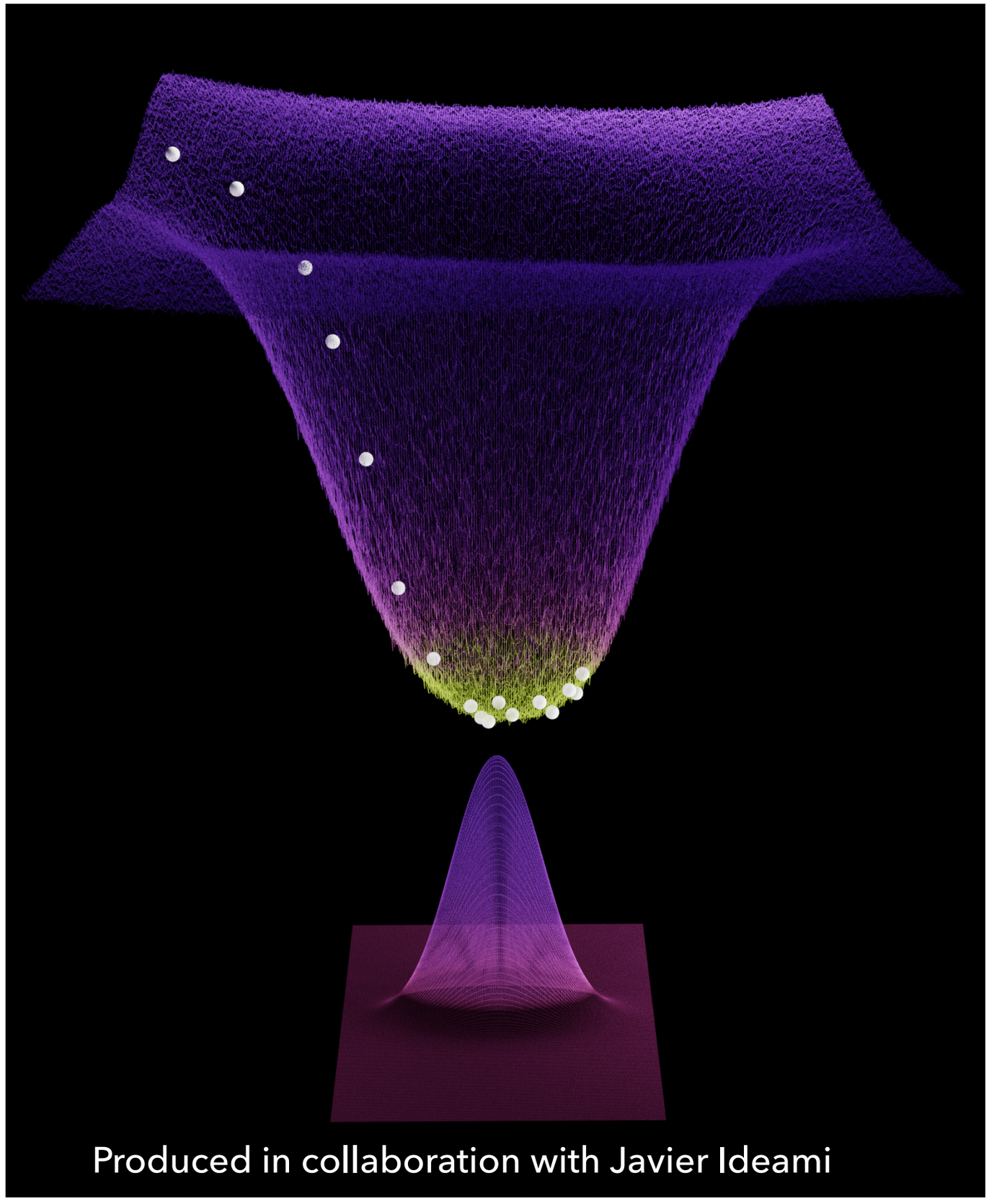


STOCHASTIC WEIGHT AVERAGING GAUSSIAN (SWAG)

During training

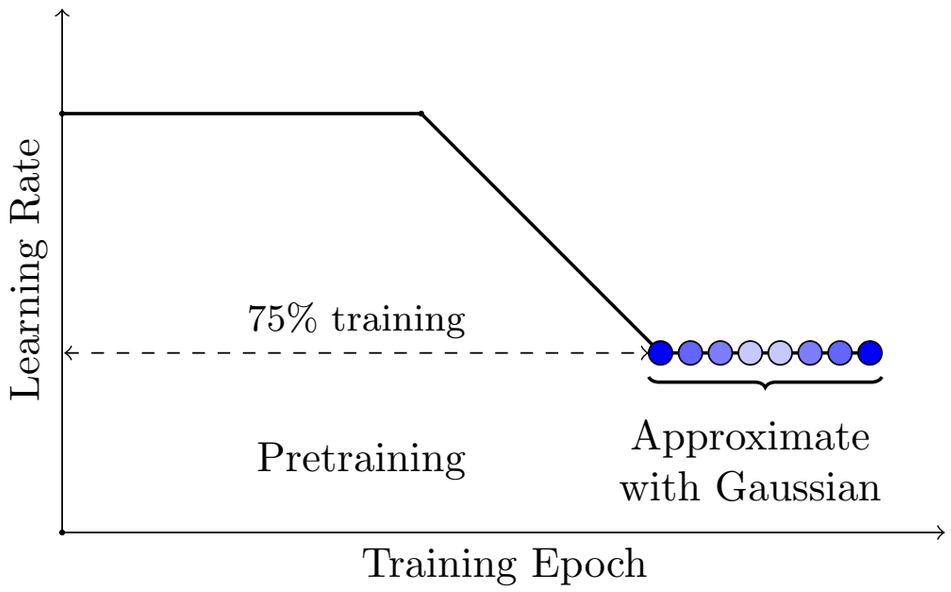


Work with Pavel Izmailov, Timur Garipov, Dmitry Vetrov, Andrew Gordon Wilson

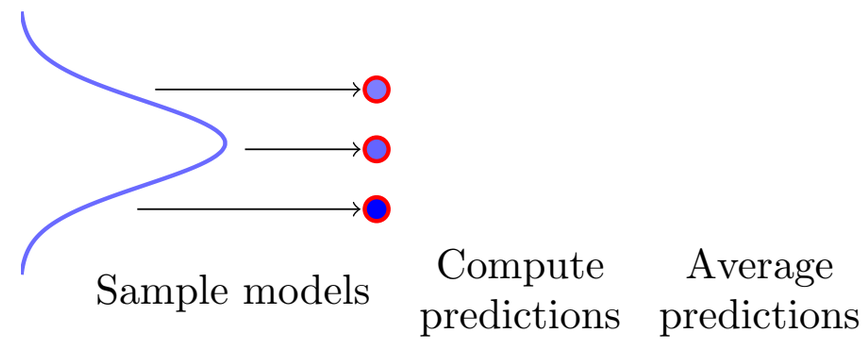


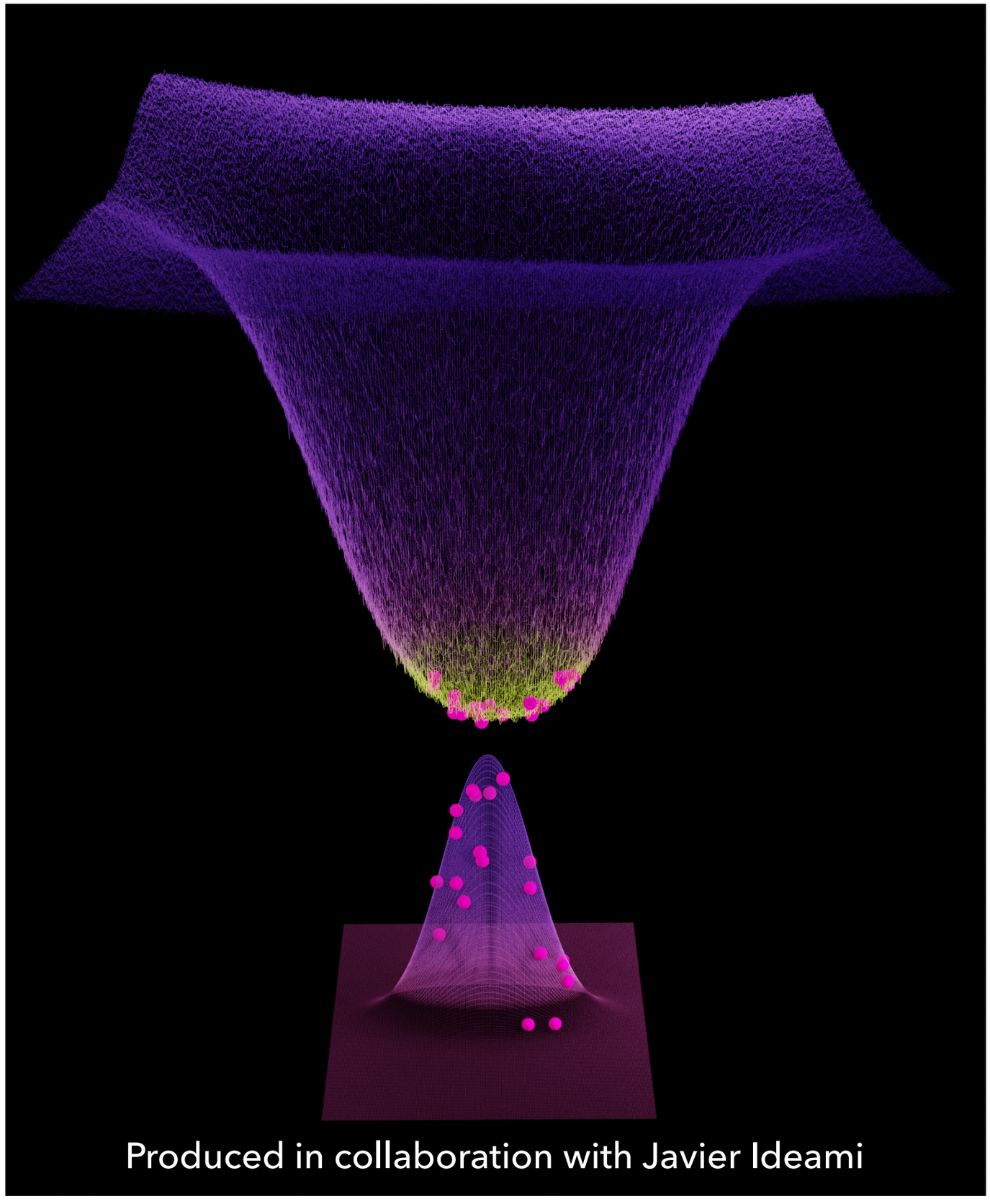
STOCHASTIC WEIGHT AVERAGING GAUSSIAN (SWAG)

During training

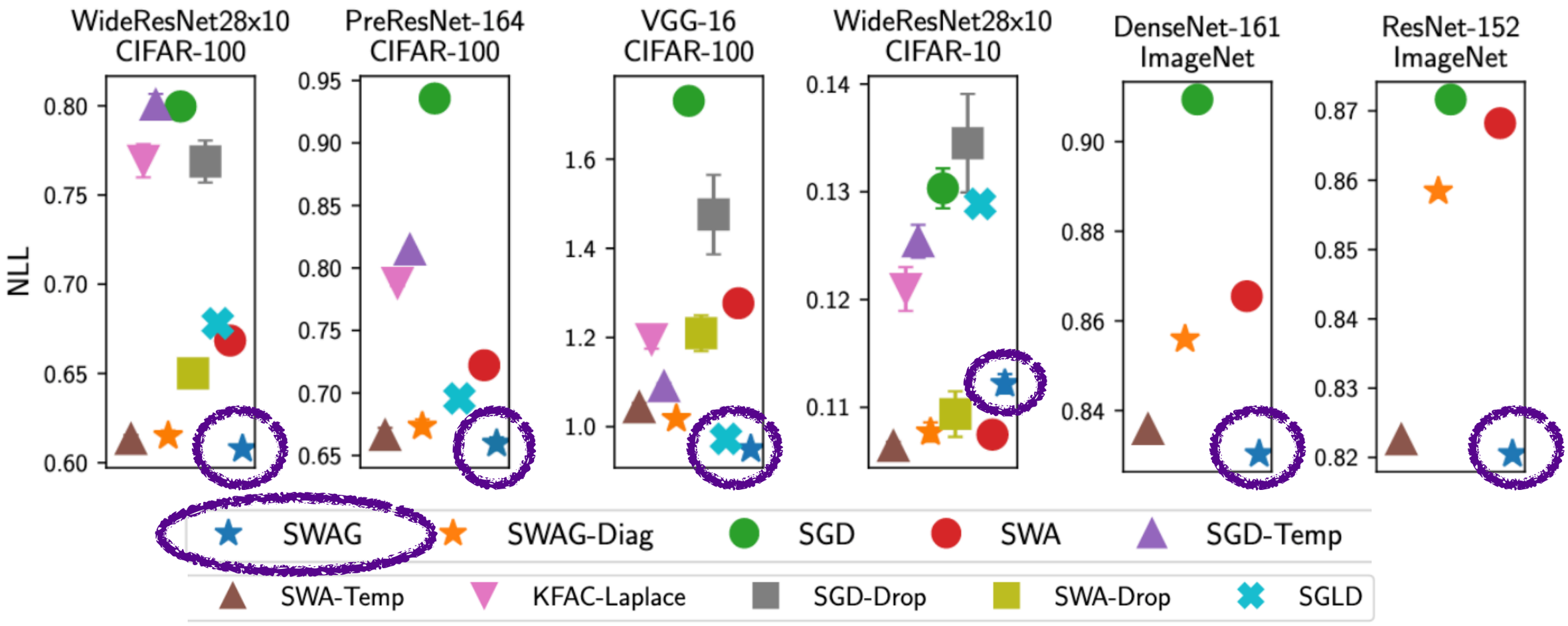


At test time

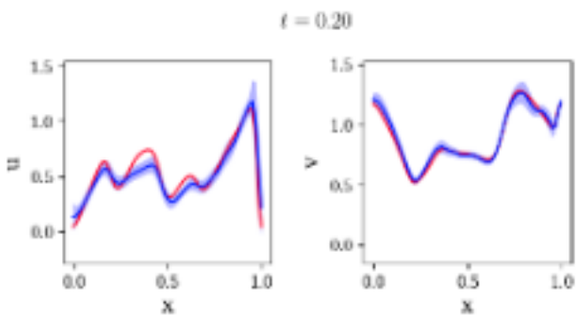


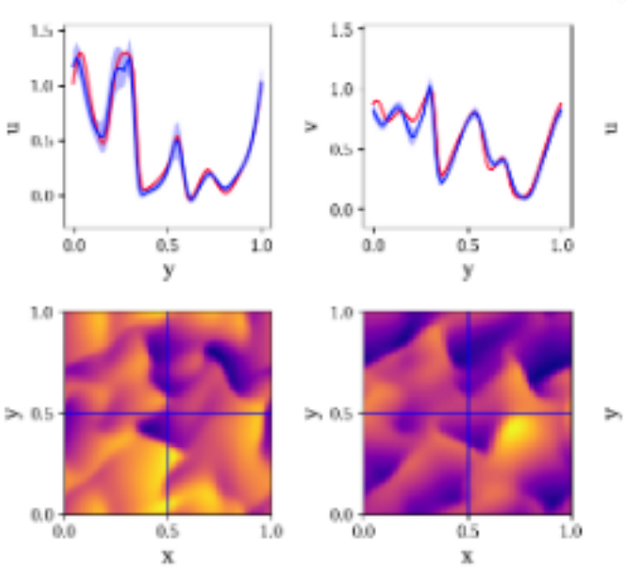


SWAG – BAYESIAN MODEL AVERAGING



APPLICATIONS OF SWAG

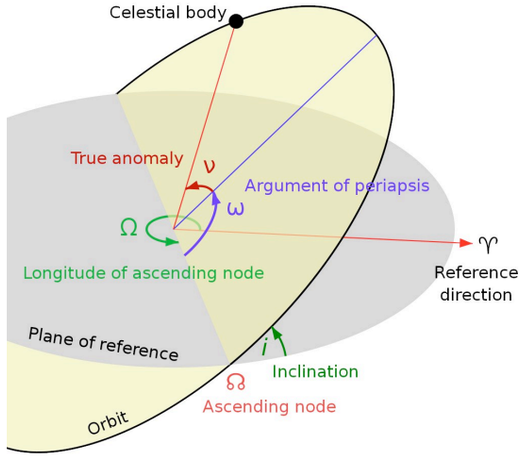




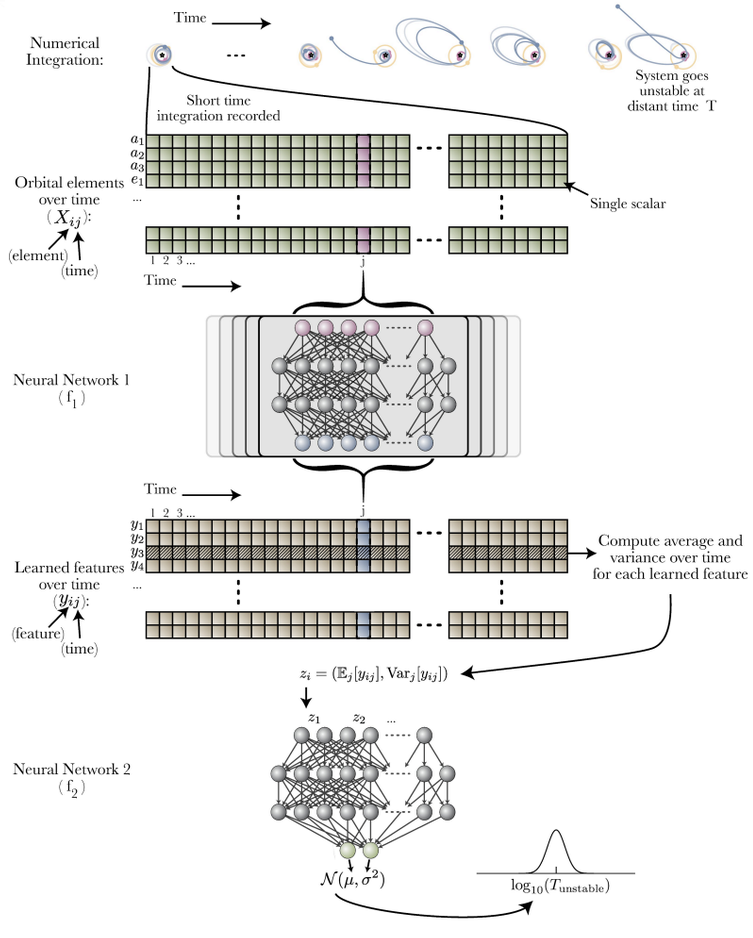
Modelling PDE Dynamics, Geneva & Zabaras, ‘19

Federated learning, Chen & Chao, ‘21

Image:https://blog.ml.cmu.edu/2019/11/12/federated-learning-challenges-methods-and-future-directions/

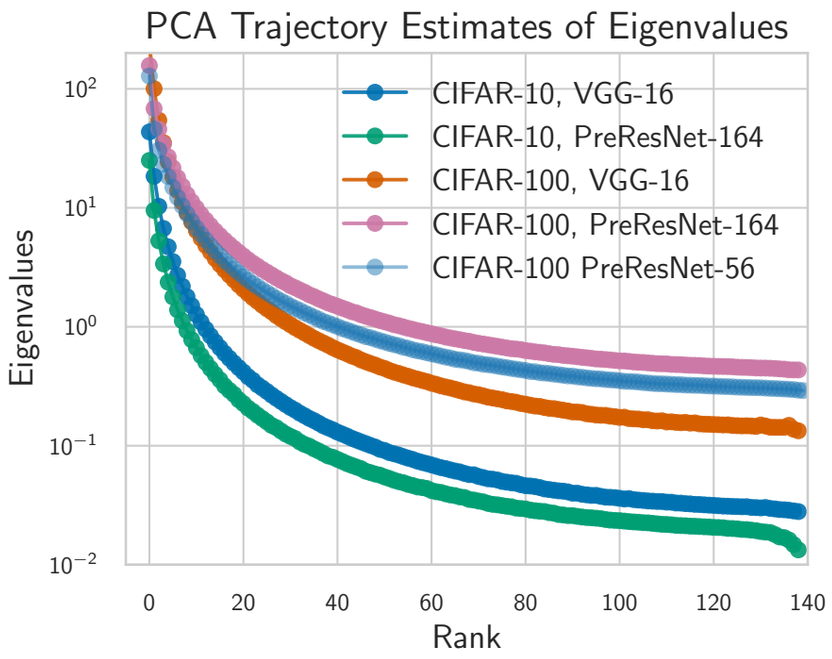


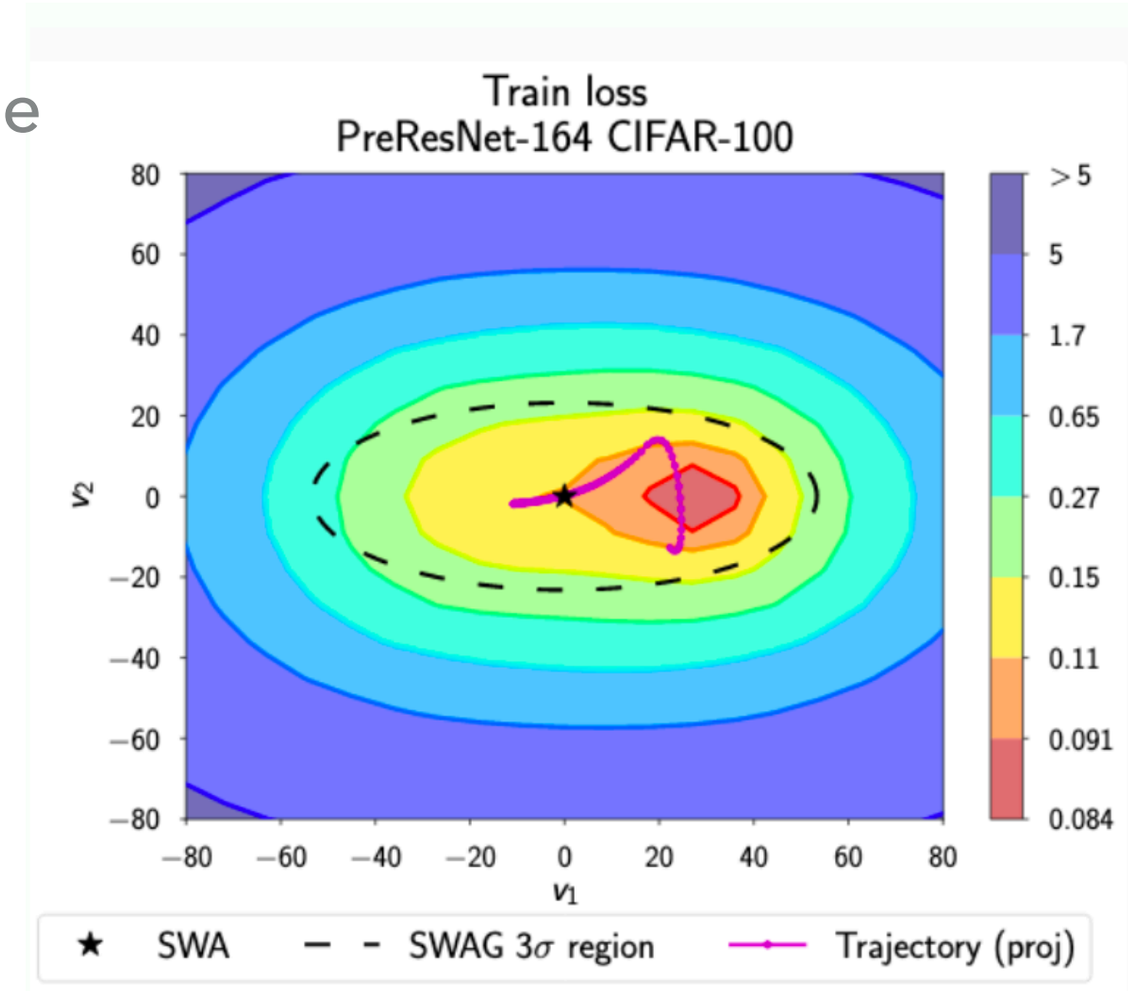
Planetary dynamics, Cranmer et al, ‘21



EMPIRICAL FINDINGS DURING SWAG EXPERIMENTS

- SGD trajectory happens in a very small subspace
  - Summarize the information from the trajectory in very low dimensions
- Also seen in Gur-Ari, et al, ‘19



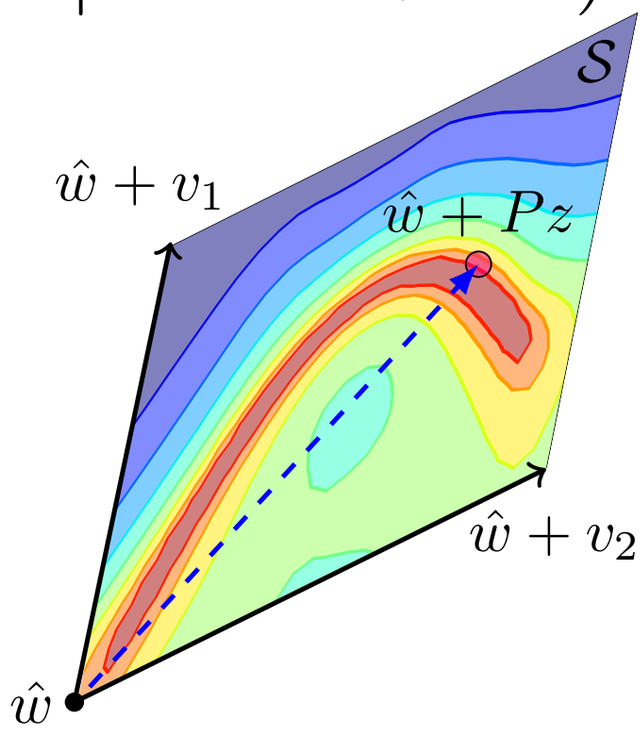


$$\psi(t_1, t_2) = \mathcal{L}(\theta_{\text{SWA}} + t_1 \cdot \frac{v_i}{\|v_i\|} + t_2 \cdot \frac{v_j}{\|v_j\|})$$

CREATING THE SUBSPACE

- Choose shift $\hat{w}$ and basis vectors $\{d_1, \cdots, d_K\}$
- Define subspace $\mathcal{S} = \{w | w = \hat{w} + \underbrace{d_1 z_1 + \cdots d_K z_K}_{Pz}\}$
- Likelihood

$$p(\mathcal{D}|z) = p_{\mathcal{M}}(\mathcal{D}|w = \hat{w} + Pz)^{1/T}$$

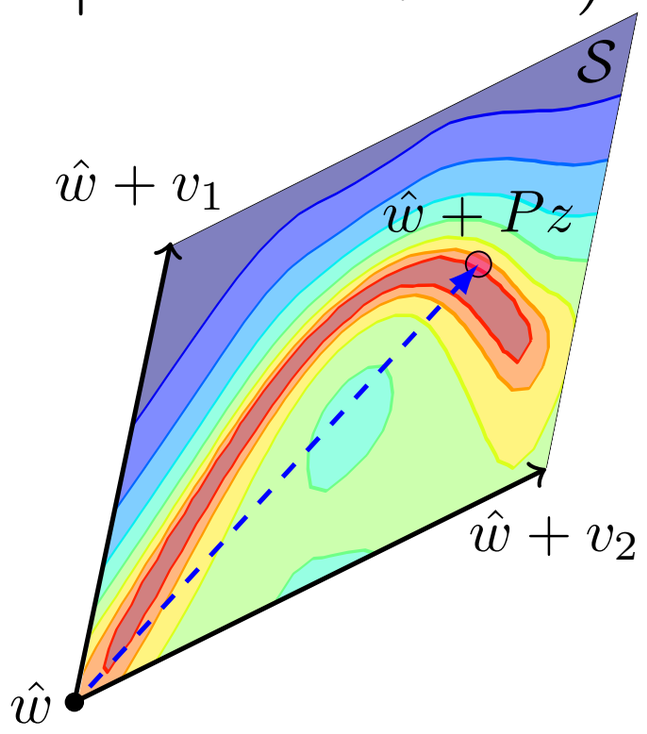


Work with Pavel Izmailov, Polina Kirichenko, Timur Garipov, Dmitry Vetrov, Andrew Gordon Wilson

CREATING THE SUBSPACE

- Choose shift $\hat{w}$ and basis vectors $\{d_1, \cdots, d_K\}$
- Define subspace $\mathcal{S} = \{w | w = \hat{w} + \underbrace{d_1 z_1 + \cdots d_K z_K}_{Pz}\}$
- Likelihood

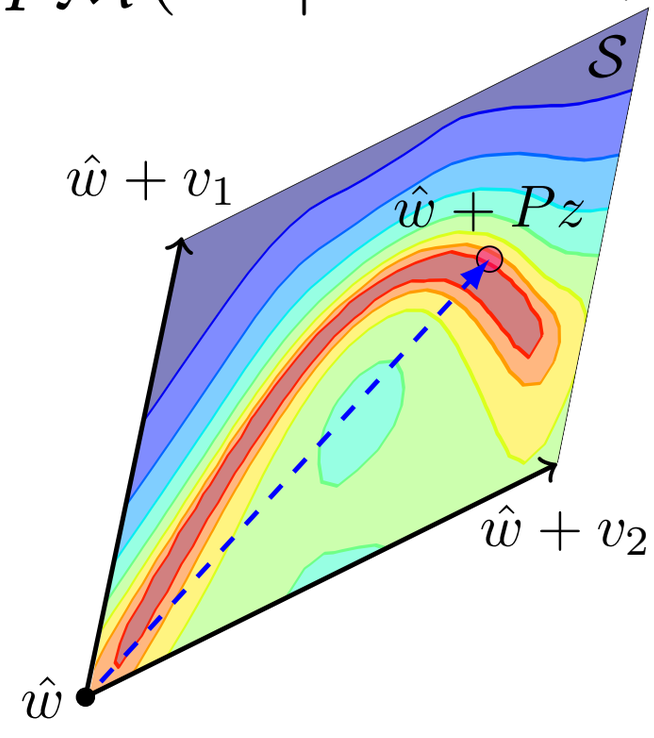
$$p(\mathcal{D}|z) = p_{\mathcal{M}}(\mathcal{D}|w = \hat{w} + Pz)^{1/T}$$



T>>1: to increase prior dependency & reduce effect of likelihood

INFERENCE IN THE SUBSPACE

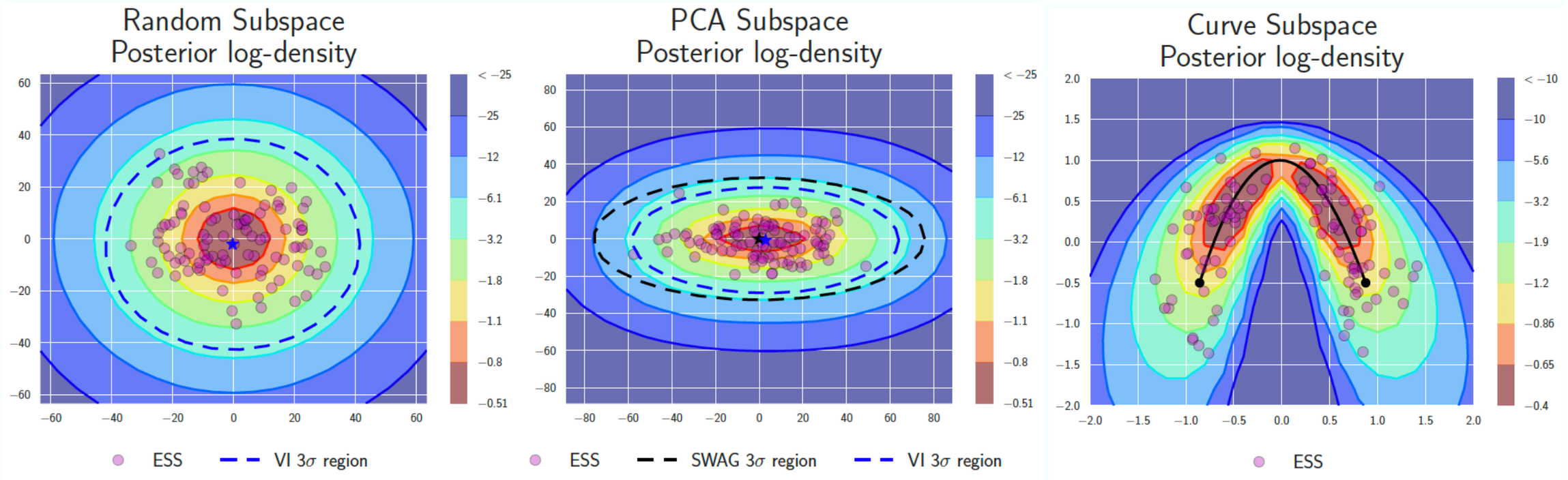
- Approximate inference over parameters
  - MCMC, VI, Normalizing Flows, …
- Bayesian model averaging at test time
- $p(\mathcal{D}^*|\mathcal{D}) = \frac{1}{J}\sum_{j=1}^{J} p_{\mathcal{M}}(\mathcal{D}^*|w = \hat{w} + P\tilde{z}_j), \tilde{z}_j \sim q(\tilde{z}|\mathcal{D})$



RESULTS (PRERESNET164, CIFAR100)

PCA of the SGD trajectory

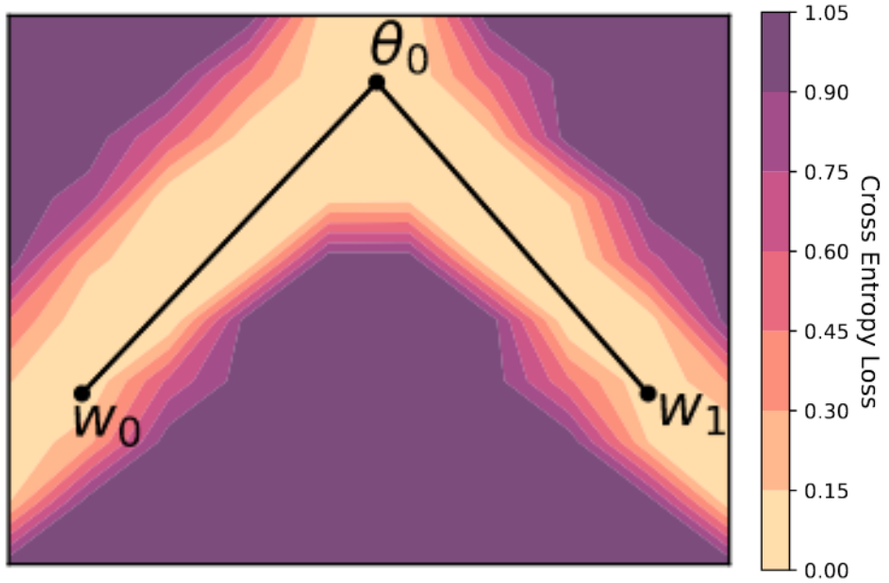
Curves: chaining ind. models (Garipov, et al, '18)



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

BUILDING THE SUBSPACE OF LOW LOSS ITERATIVELY: SPRO

- Rather than constructing a posterior over the subspace, let's build subspaces that represent regions of low training loss for NNs



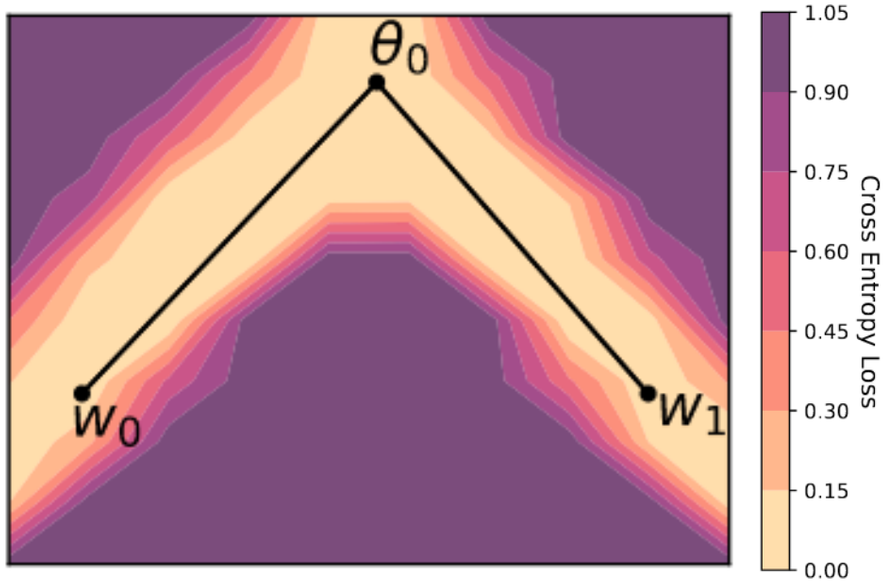
Mode connecting path like Garipov et al, '18

Work with Greg Benton, Sanae Lofti, Andrew Gordon Wilson

w_i: independent network, θ_i: connector

BUILDING THE SUBSPACE OF LOW LOSS ITERATIVELY: SPRO

- Rather than constructing a posterior over the subspace, let's build subspaces that represent regions of low training loss for NNs



Mode connecting path like Garipov et al, '18

Continue training an entire simplex of low loss while also pushing each vertex away from each other

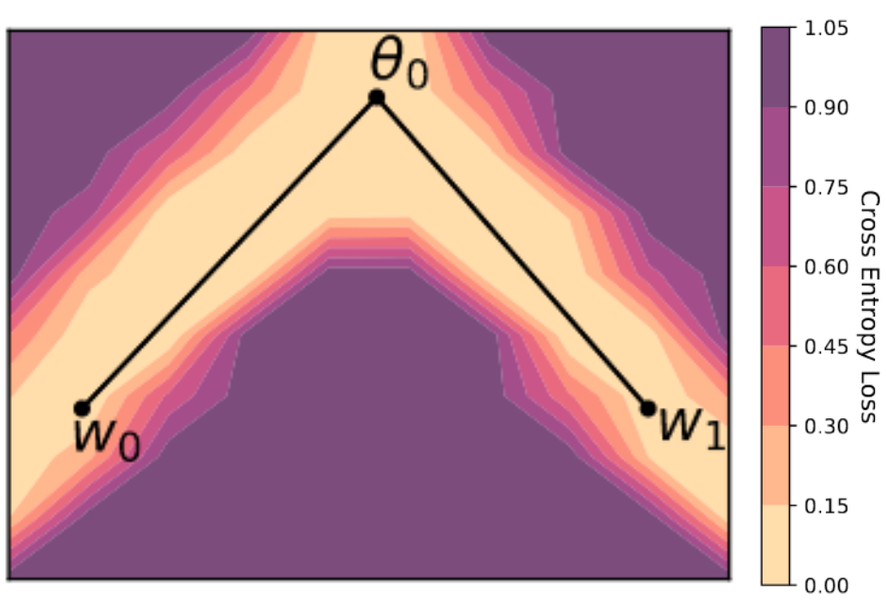
$$\mathcal{L}_{reg}(\mathcal{D}, S_{(w_j,\theta_{j,0},\ldots,\theta_{j,k})}) = \frac{1}{H}\sum_{\phi_h \sim S} \mathcal{L}(\mathcal{D}, \phi_h) - \lambda_i \log(\mathrm{V}(S_{(w_j,\theta_{j,0},\ldots,\theta_{j,k})})).$$

Simplicial pointwise random optimization (SPRO)

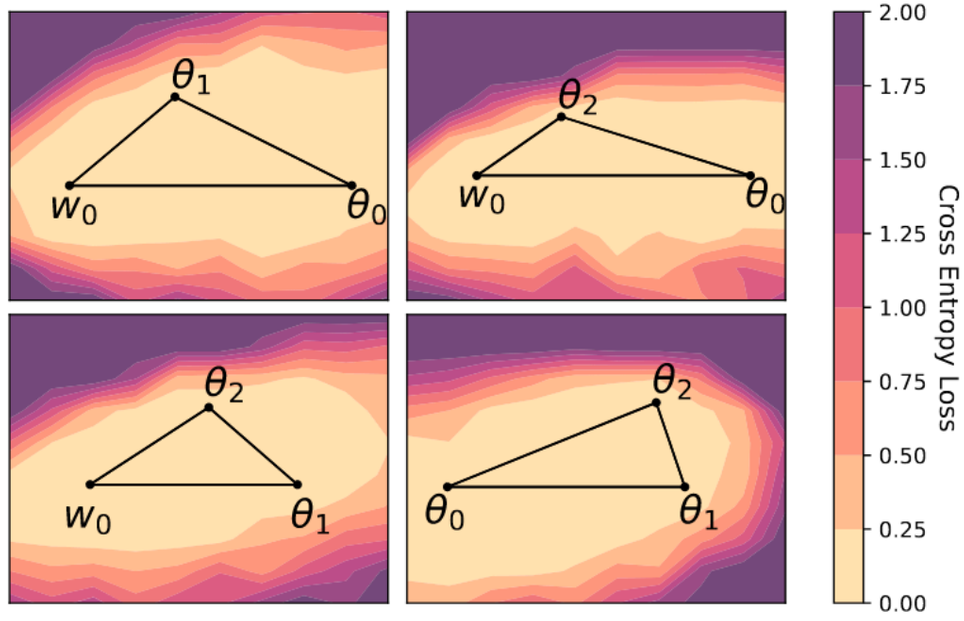
w_i: independent network, θ_i: connector

BUILDING THE SUBSPACE OF LOW LOSS ITERATIVELY: ESPRO

- Rather than constructing a posterior over the subspace, let's build subspaces that represent regions of low training loss for NNs



Mode connecting path like Garipov et al, '18

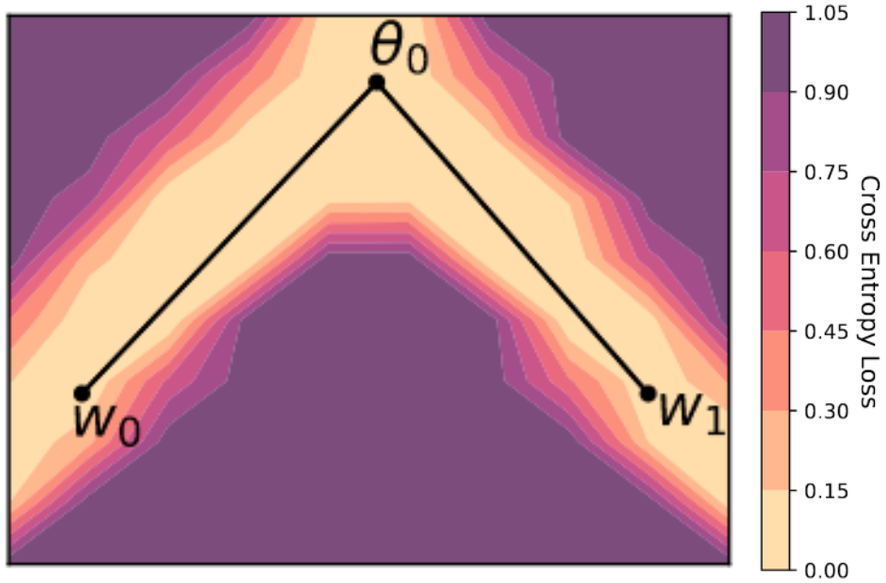


Different faces of a mode connecting simplexes, which has low loss.

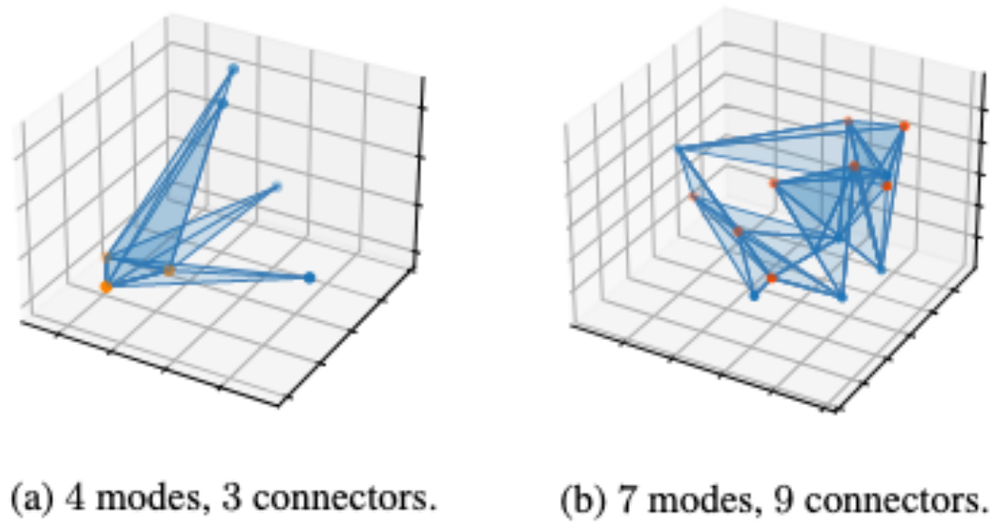
w_i: independent network, θ_i: connector

BUILDING THE SUBSPACE OF LOW LOSS ITERATIVELY: ESPRO

- Rather than constructing a posterior over the subspace, let's build subspaces that represent regions of low training loss for NNs



Mode connecting path like Garipov et al, '18



(a) 4 modes, 3 connectors. (b) 7 modes, 9 connectors.

Visualization of a single simplex.

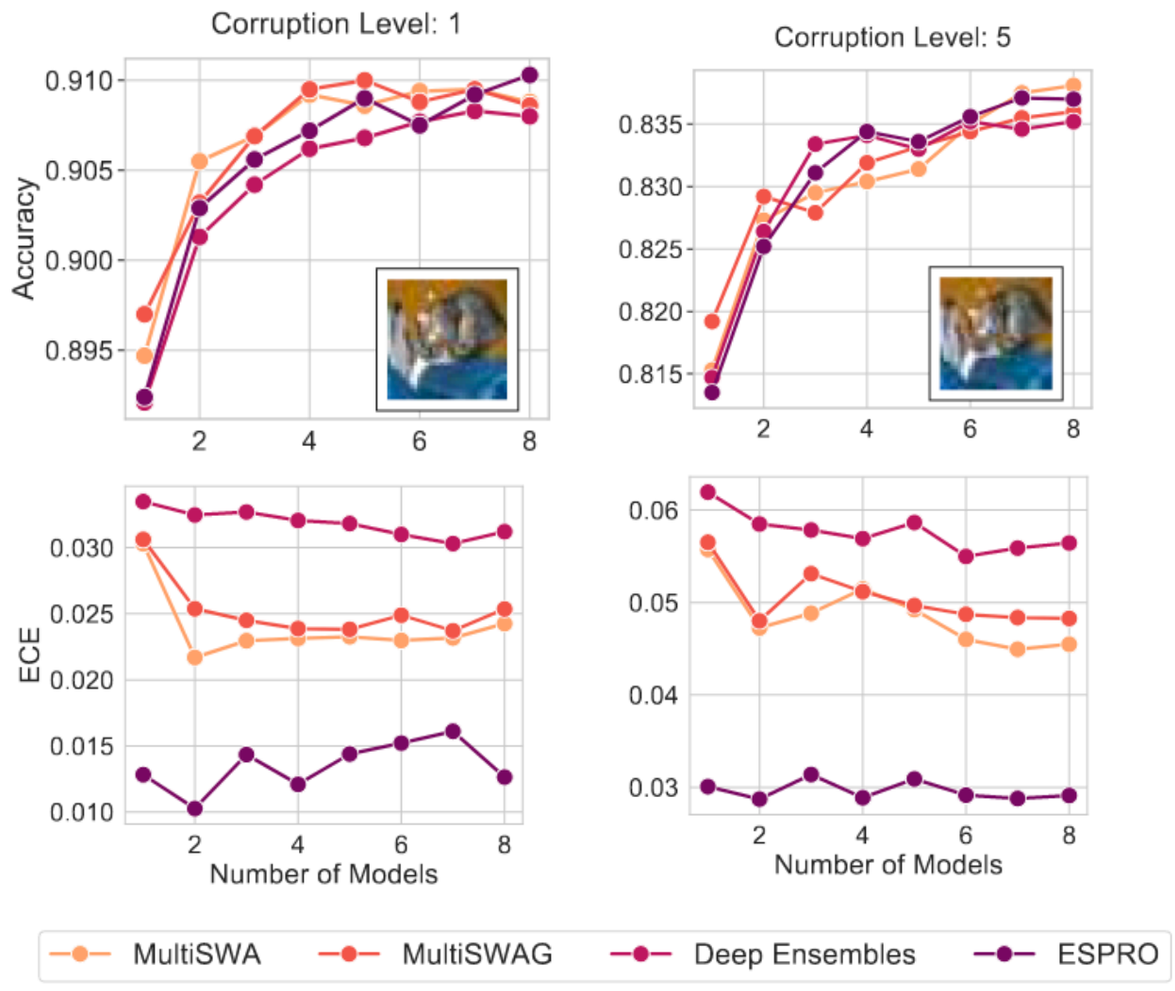
w_i: independent network, θ_i: connector

BUILDING THE SUBSPACE OF LOW LOSS ITERATIVELY: ESPRO

At test time, marginalize over the simplex via MC integration (smells like a Bayesian model average).

$$\hat{y} = \frac{1}{J} \sum_{\phi_j \sim \mathcal{K}} f(x, \phi_j) \approx \int_{\mathcal{K}} f(x, \phi_j) d\phi_j$$

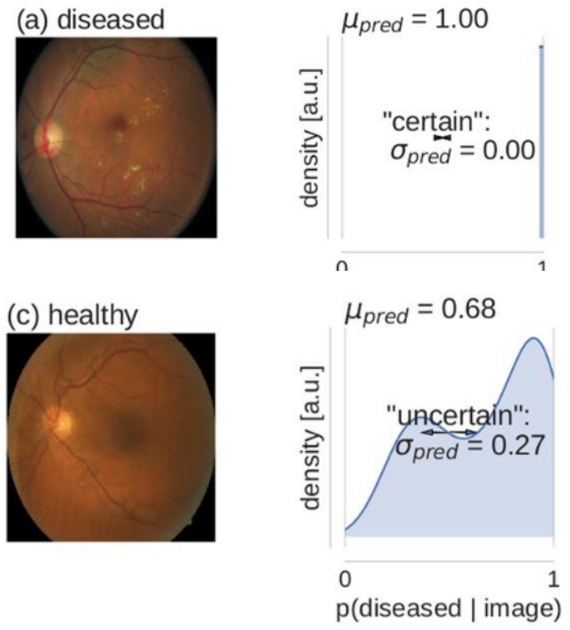
Gaussian blur corruptions of CIFAR-10 test set



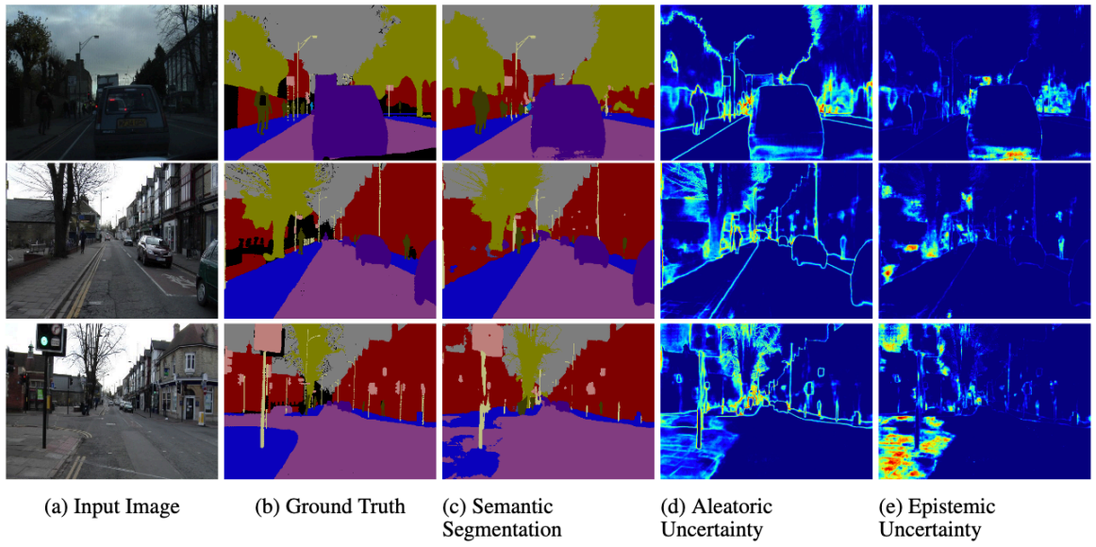
COMPARISONS OF APPROXIMATE INFERENCE METHODS

Method	Accuracy	Calibration	Train time	Test time	Code
Ensembles (Lakshminarayanan et al, '17)	**Best**	Often more overconfident	K times standard training	K times slower	**Train K models**
SWAG (Maddox et al, '19)	Slightly better than MAP	**Less overconfident**	**1.5x standard training**	K times slower	**Store models at train time**
SPRO (Benton et al, '21)	**Better than MAP**	Needs temperature scaling	**1.5x standard training**	K times slower	**Modify loss function.**
SGMCMC (Welling & Teh '11, Chen et al, '16, Zhang et al, '20)	**Better than MAP**	**Less overconfident**	2-5x standard training	K times slower	**Modify SGD. run longer**
Dropout (Gal & Gharamani, '16)	About the same as MAP	Slightly less overconfident	**Standard training**	K times slower	**Apply dropout at test time**
VOGN (Osawa et al, '19)	Worse than MAP	Less overconfident	2x standard training	K times slower	Modify Adam

OTHER APPLICATIONS OF BAYESIAN DEEP LEARNING

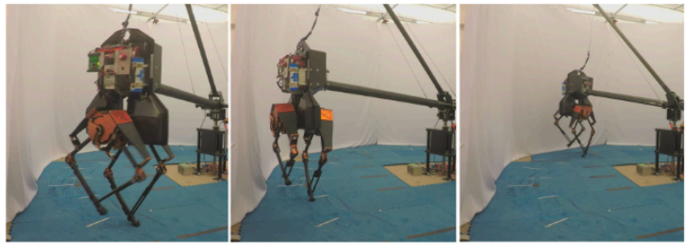


Diabetic retinopathy detection, Leibig et al, '17; Filos et al, '19

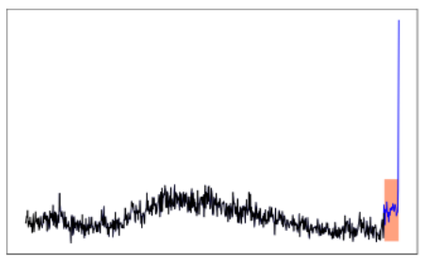


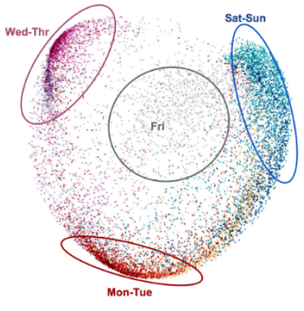
Semantic segmentation for autonomous driving, Kendall & Gal, '17; Kendall & Cipolla, '15

Figure 5: ATRIAS during BO with *asymNN* kernel.



Robotics, Akrour et al, 17; Antonova et al,'17

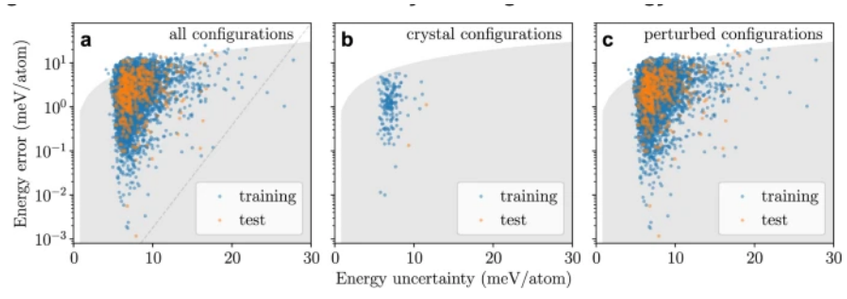




Uber demand prediction, Zhu & Laptev, '17

Click-through rate prediction at Tencent, Liu et al, '17

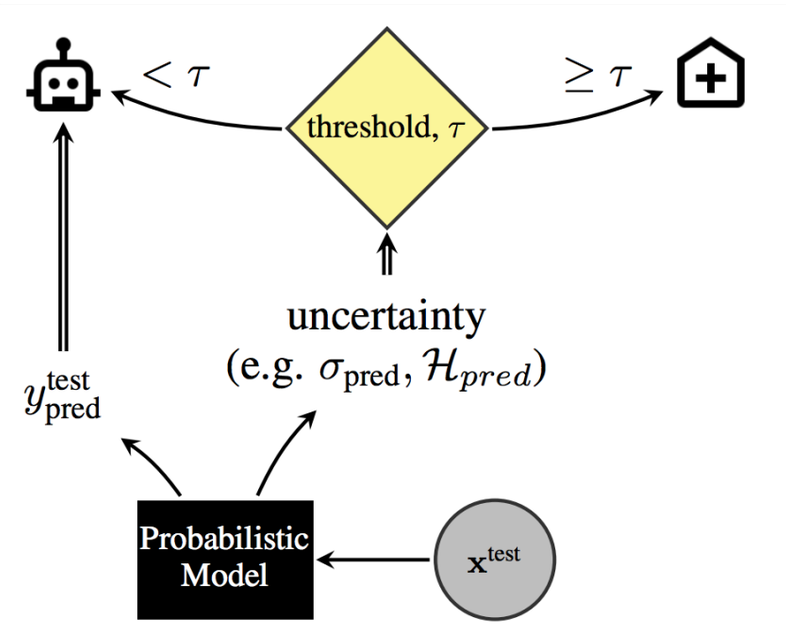
https://towardsdatascience.com/predicting-click-through-rate-for-a-website-7cd2a892d26e



Molecular dynamics, Wen & Tadmor, '20

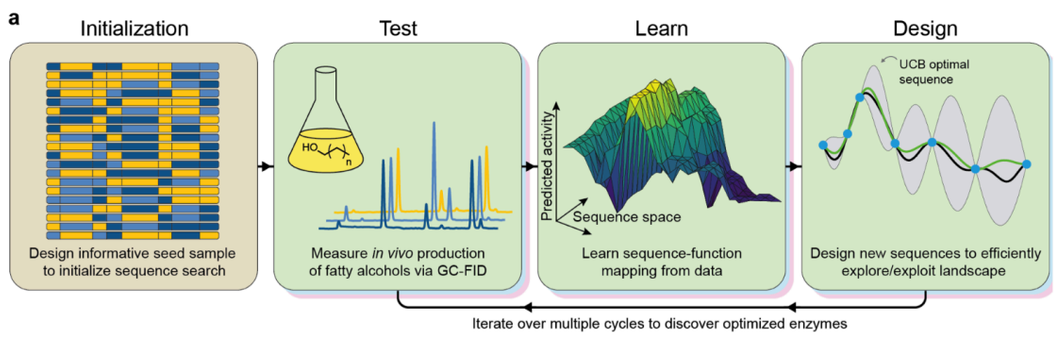
ORGANIZATION

- Two motivating themes:
  - "How do we develop practical Bayesian neural networks?"



Filos et al, '20

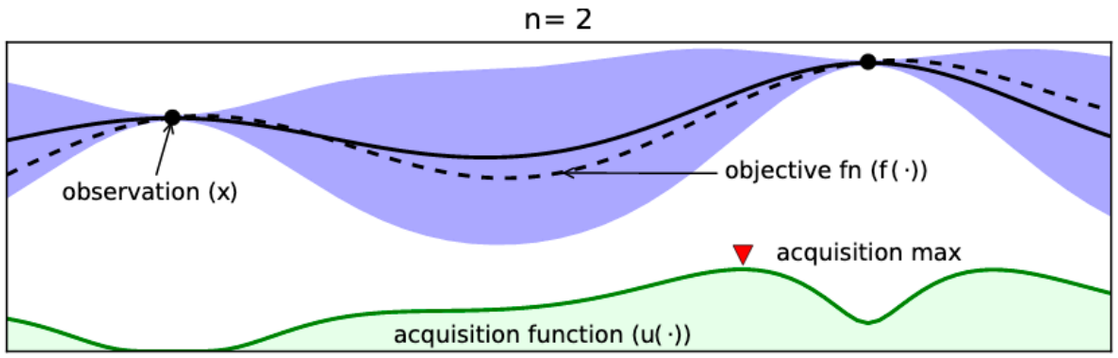
- **"How do we use Bayesian optimization to solve real problems?"**



Greenhalgh et al, '21

BAYESIAN OPTIMIZATION INTRO

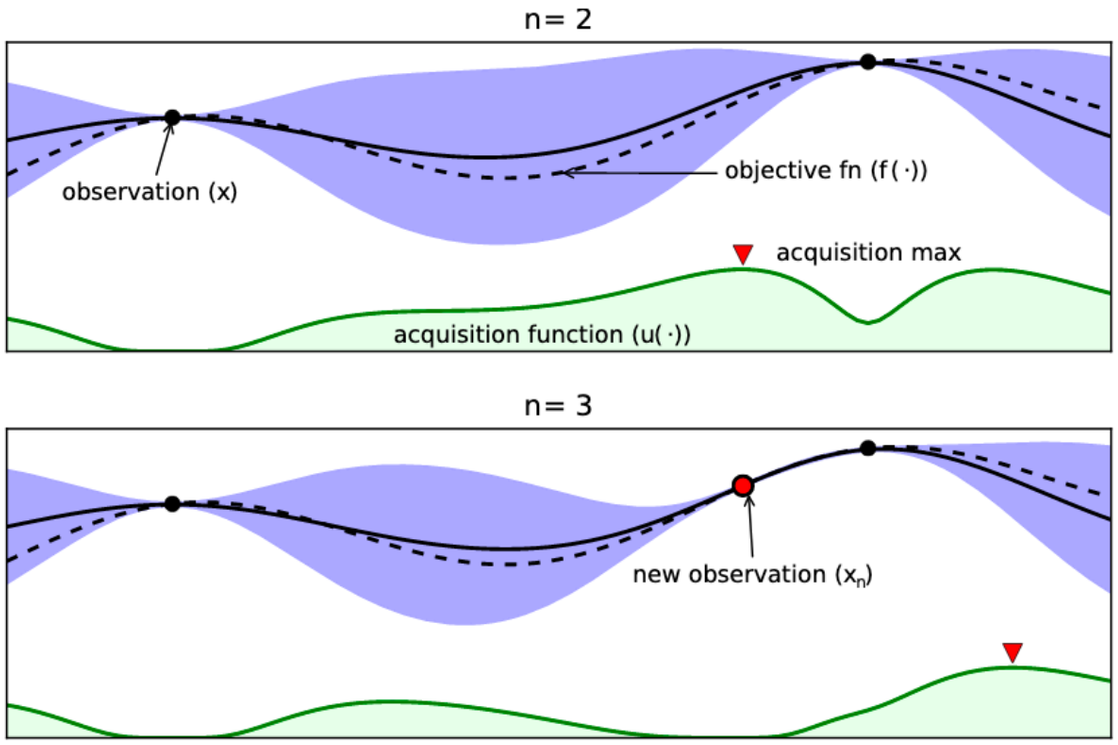
- Goal: $\max_x f(x)$
  - f is costly to evaluate
  - Make minimal assumptions about problem
    - X is low-dimensional
- Approach:
  - Build a probabilistic *surrogate* model
  - Suggest new points by optimizing an *acquisition function* on the surrogate



From Shahriari et al, '16

BAYESIAN OPTIMIZATION INTRO

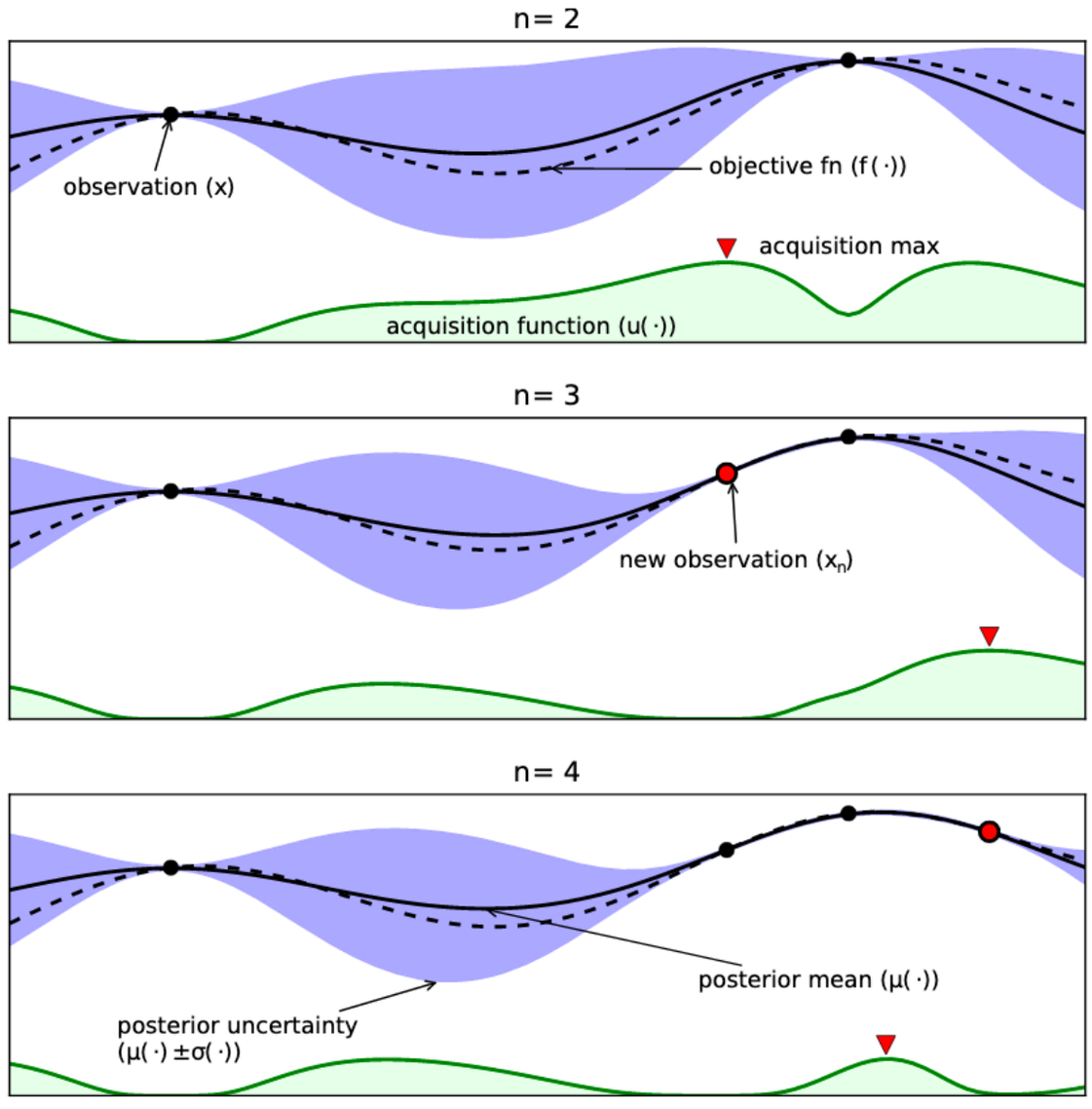
- Goal: $\max_x f(x)$
  - f is costly to evaluate
  - Make minimal assumptions about problem
  - X is low-dimensional
- Approach:
  - Build a probabilistic *surrogate* model
  - Suggest new points by optimizing an *acquisition function* on the surrogate



From Shahriari et al, ‘16

BAYESIAN OPTIMIZATION INTRO

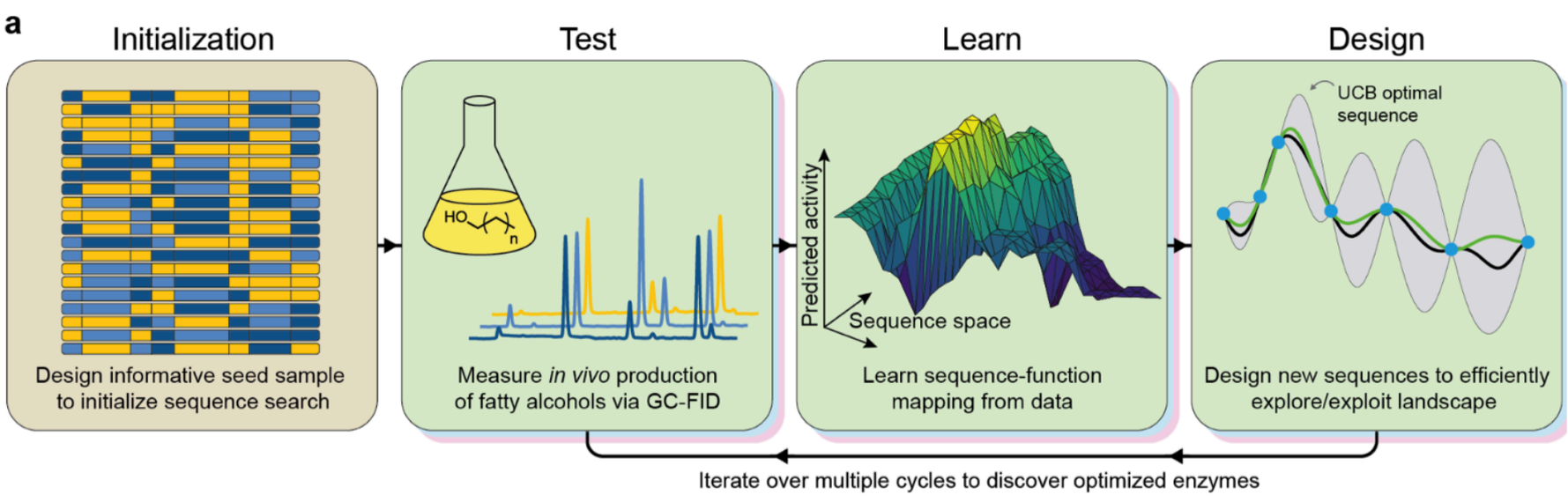
- Goal: $\max_x f(x)$
  - f is costly to evaluate
  - Make minimal assumptions about problem
    - X is low-dimensional
- Approach:
  - Build a probabilistic *surrogate* model
  - Suggest new points by optimizing an *acquisition function* on the surrogate



From Shahriari et al, ‘16

BAYESIAN OPTIMIZATION LOOPS IN SCIENCE

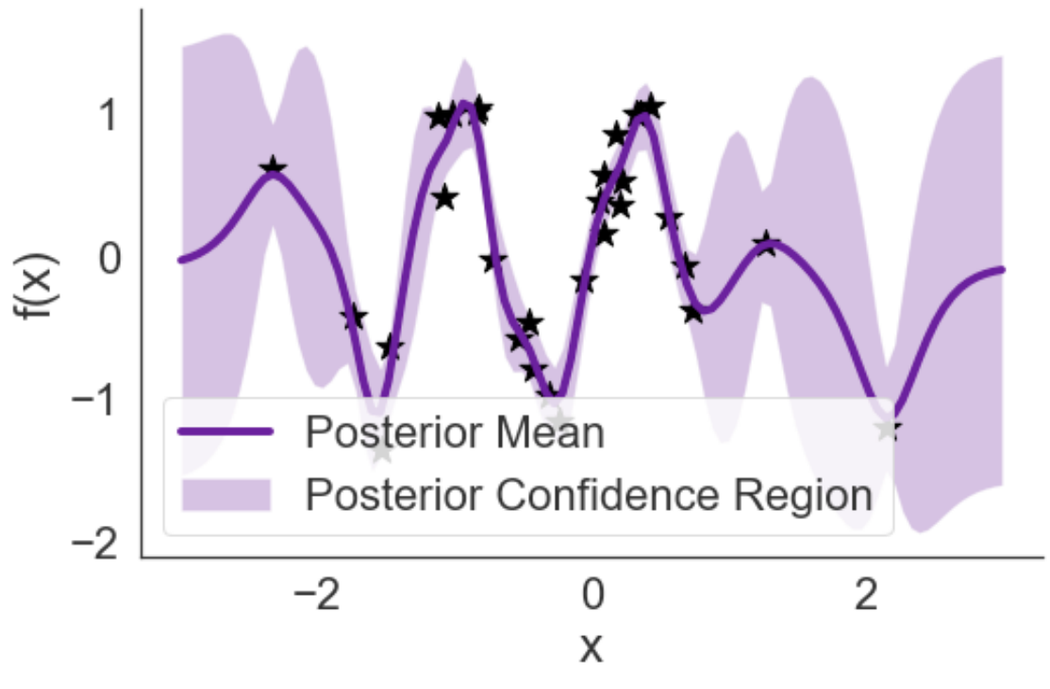
- Bayesian optimization loops look a lot like this.



Enzyme generation: Greenhalgh et al, '21 (https://www.biorxiv.org/content/10.1101/2021.05.21.445192v1)

GAUSSIAN PROCESSES

- Nonparametric models over functions
  - Extend multivariate gaussians to function spaces
  - Latent function $f \sim \mathcal{GP}(\mu_\theta(x), k_\theta(x, x'))$ $y \sim \mathcal{N}(f, \sigma^2 I)$
- Predictive distribution is closed form (for regression)



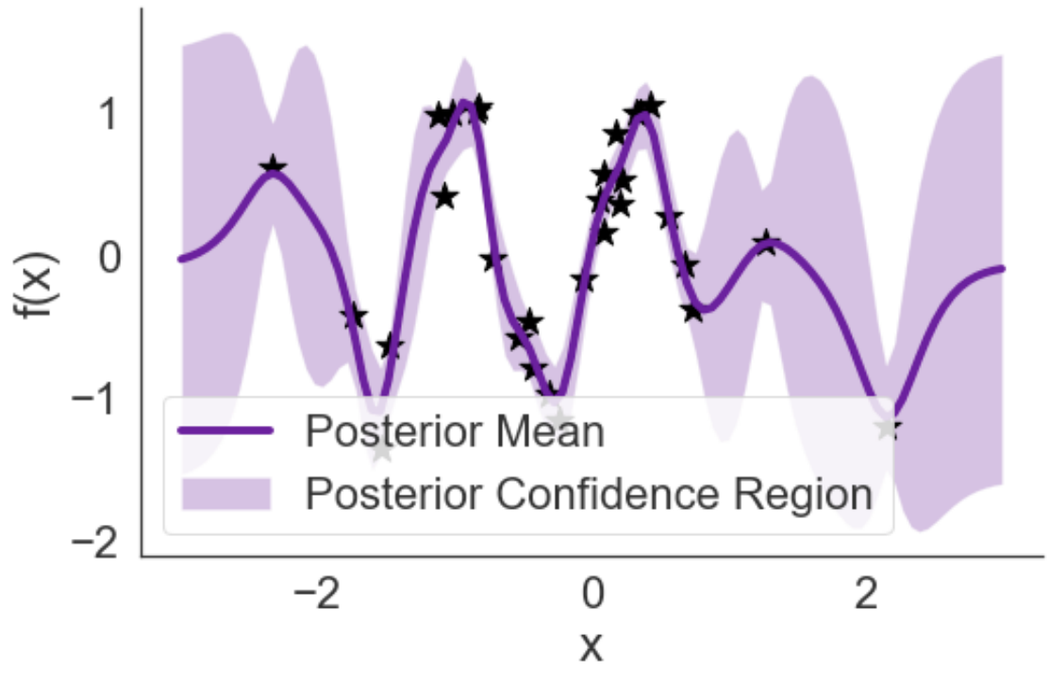
GAUSSIAN PROCESSES: PREDICTION

- The predictive distribution is given by:

$$p(f^*|X^*, X, y) = \mathcal{N}(\mu_{f|\mathcal{D}}, \Sigma_{f|\mathcal{D}})$$

$$\mu_{f|\mathcal{D}} = K_{\mathbf{x}^* X}(K_{XX} + \sigma^2 I)^{-1}\mathbf{y},$$

$$\Sigma_{f|\mathcal{D}} = K_{\mathbf{x}^*\mathbf{x}^*} - K_{\mathbf{x}^* X}(K_{XX} + \sigma^2 I)^{-1}K_{X\mathbf{x}^*}.$$



GAUSSIAN PROCESSES: UPDATING THE PREDICTIVE

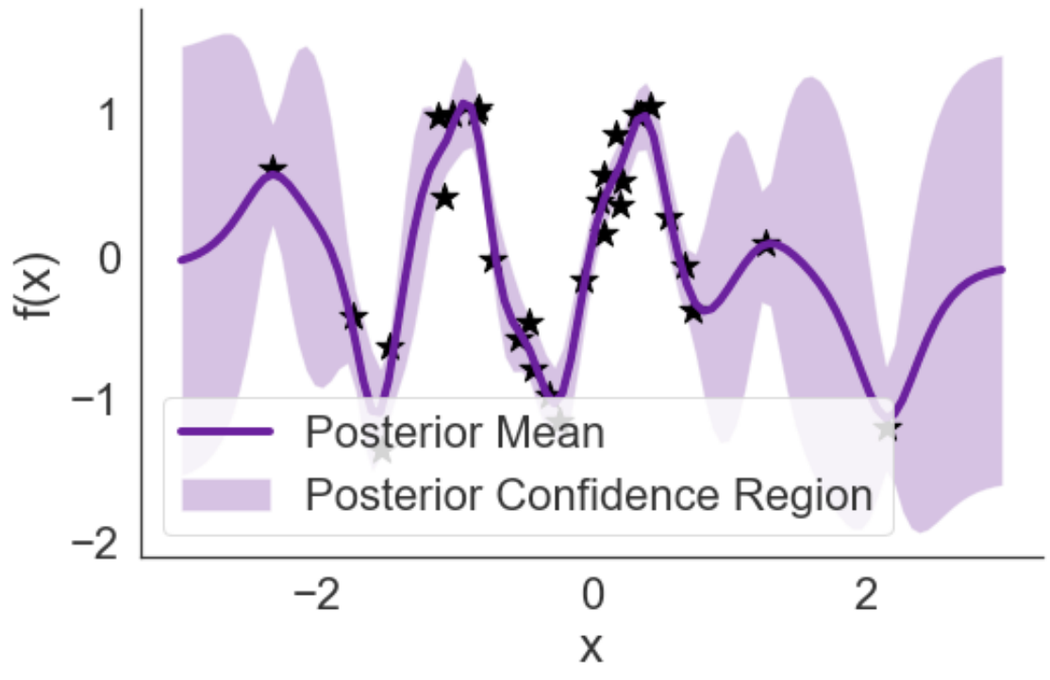
- The predictive distribution is given by:

$$p(f^*|X^*, X, y) = \mathcal{N}(\mu_{f|\mathcal{D}}, \Sigma_{f|\mathcal{D}})$$

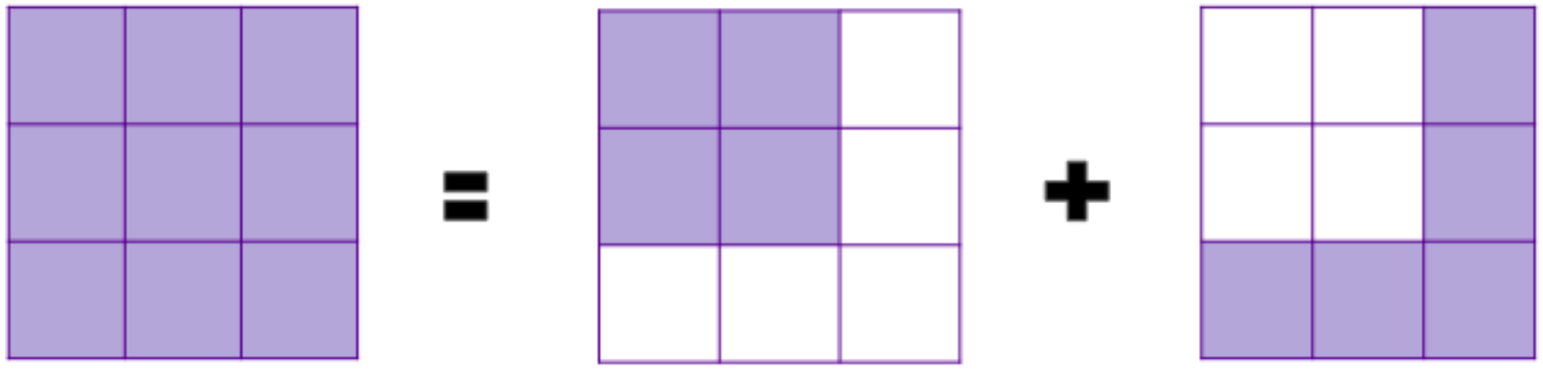
We need to update these terms

$$\mu_{f|\mathcal{D}} = K_{\mathbf{x}^* X}(K_{XX} + \sigma^2 I)^{-1}\mathbf{y},$$

$$\Sigma_{f|\mathcal{D}} = K_{\mathbf{x}^*\mathbf{x}^*} - K_{\mathbf{x}^* X}(K_{XX} + \sigma^2 I)^{-1}K_{X\mathbf{x}^*}.$$



UPDATING THE PREDICTIVE



New kernel

$$(\mathbf{K}_{X'X'} + \sigma^2 I)$$

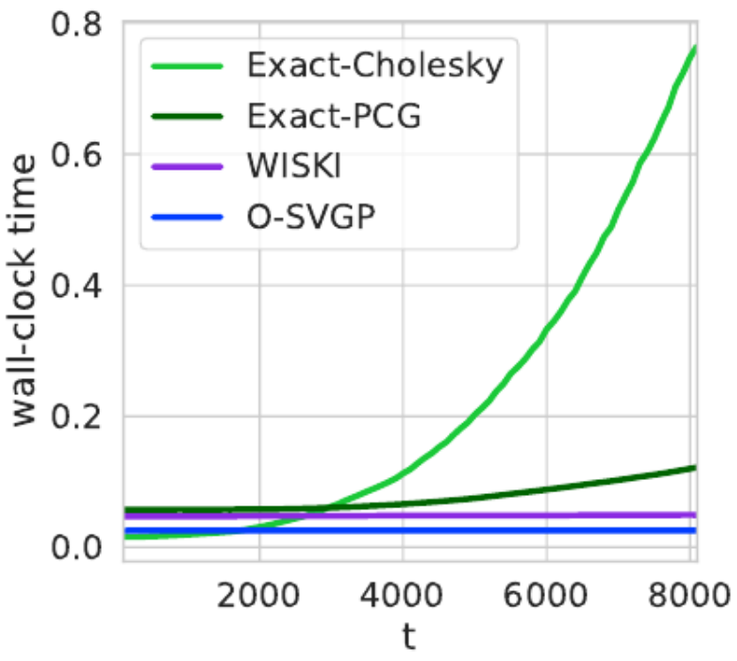
Old kernel

$$(\mathbf{K}_{XX} + \sigma^2 I)$$

Cross-terms

$$\begin{bmatrix} \mathbf{0} & k(X, \mathbf{x}') \\ k(\mathbf{x}', X) & k(\mathbf{x}', \mathbf{x}') + \sigma^2 \end{bmatrix}$$

Low-rank updates cost at least **O(kn)** even if using low-rank decompositions (e.g. Lanczos)



Model gets slower as we see more data!

Work with Sam Stanton, Ian Delbridge, Andrew Gordon Wilson

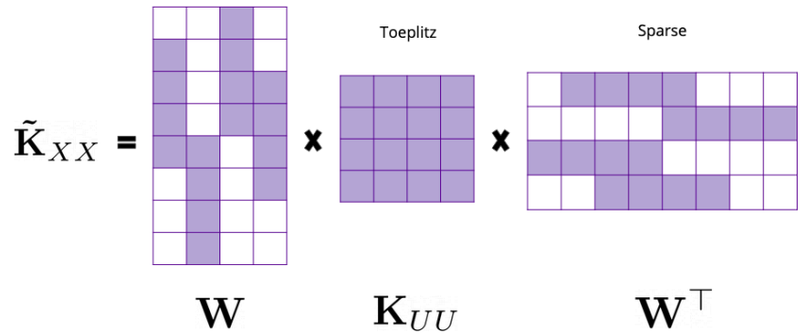
USING SKI FOR CONSTANT TIME PREDICTION: WISKI

Wilson & Nickisch '15

- We can use Woodbury's matrix identity to invert the structured kernel interpolation (SKI, Wilson & Nickisch, '15) kernel in **constant time**

$$(\tilde{K}_{XX} + \sigma^2 I)^{-1} = (\mathbf{W}\mathbf{K}_{UU}\mathbf{W}^\top + \sigma^2 I)^{-1}$$

$$(\tilde{K}_{XX} + \sigma^2 I)^{-1} = \sigma^{-2} I - \sigma^{-4}\mathbf{W}\underbrace{(\mathbf{K}_{UU}^{-1} + \sigma^{-2}\mathbf{W}^\top\mathbf{W})^{-1}}_{\mathbf{M}}\mathbf{W}^\top$$

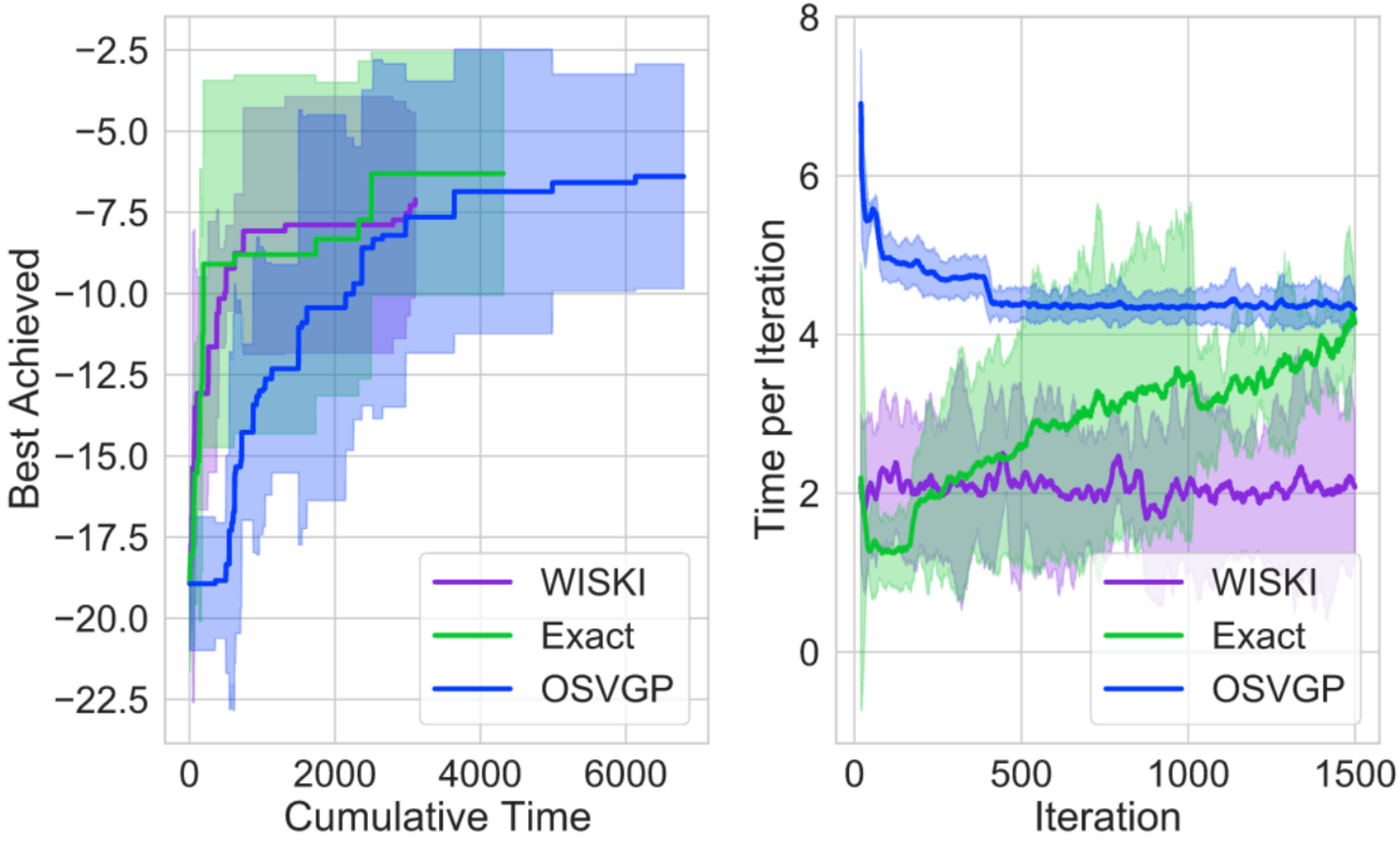


All updates are O(1)

$$\mu(\mathbf{x}^*) = \mathbf{w}_{\mathbf{x}^*}^\top \mathbf{M}\mathbf{W}^\top\mathbf{y} \qquad \Sigma(\mathbf{x}^*) = \sigma^2\mathbf{w}_{\mathbf{x}^*}^\top \mathbf{M}\mathbf{w}_{\mathbf{x}^*}$$

m x m — cached — m x m

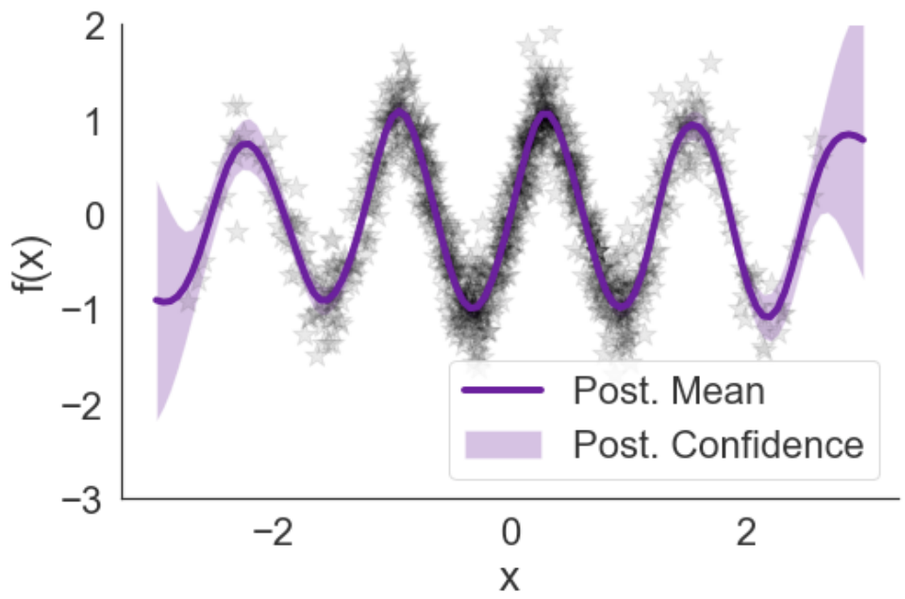
THE ADVANTAGES OF EXACT INFERENCE W/ INDUCING POINTS

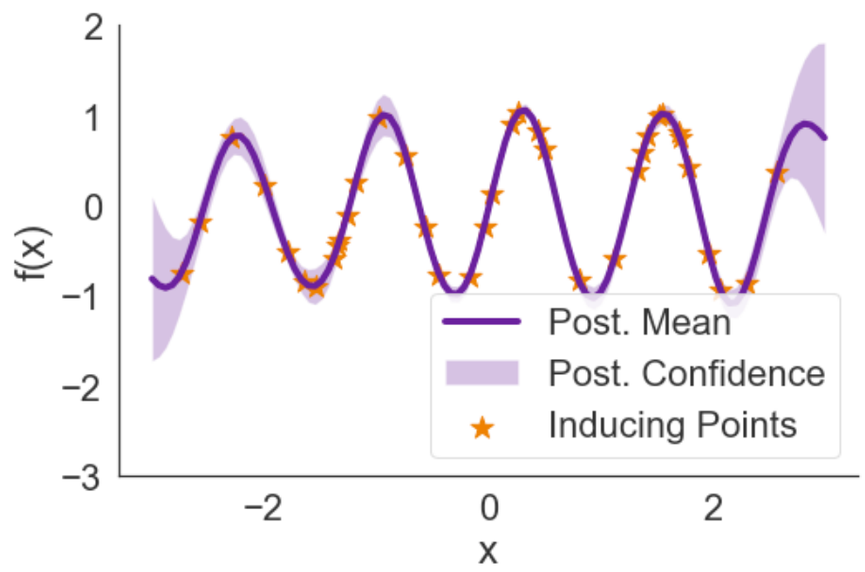


Strong performance on bayesian optimization tasks, while being significantly faster

Noisy Levy-3 test problem

STOCHASTIC VARIATIONAL GAUSSIAN PROCESSES (SVGPS)





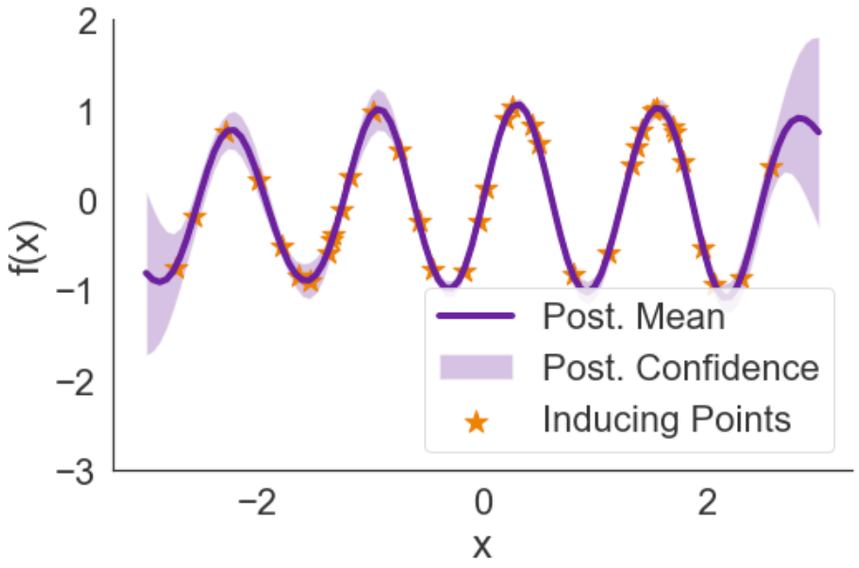
Stochastic variational GPs (Titsias, '09, Hensman et al, '13, '15) condition the GP on **"inducing points"** and optimize the ELBO wrt to the inducing points and their corresponding variational distribution, $\phi(\mathbf{u}) = \mathcal{N}(\mathbf{m}, \mathbf{S})$.

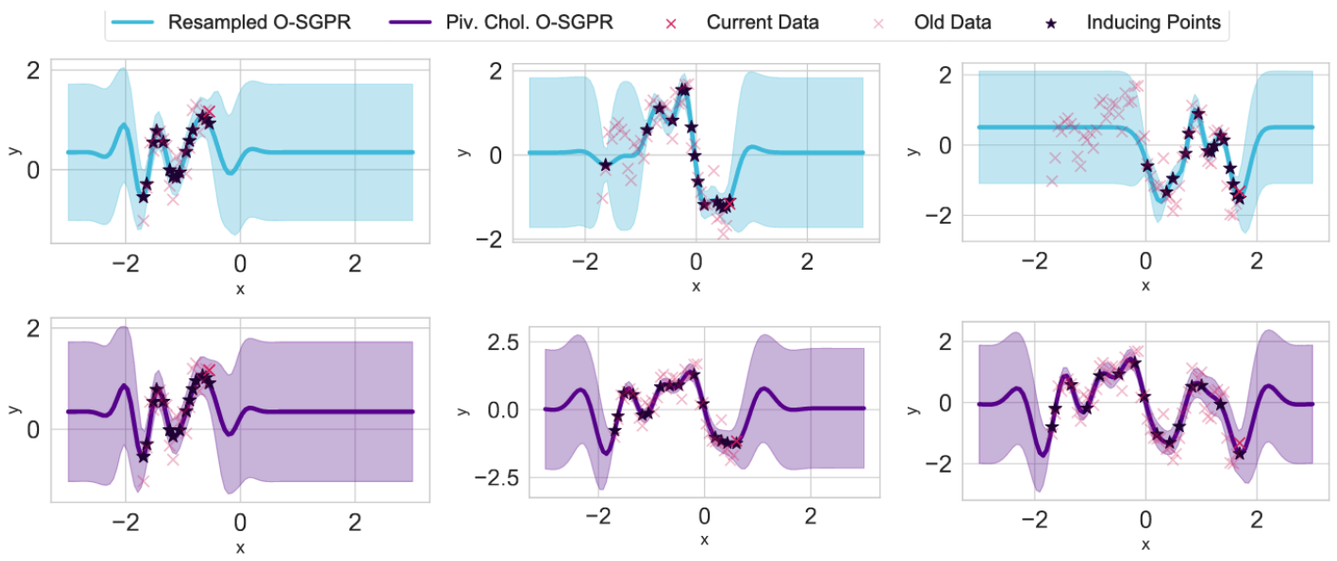
Instead of optimizing the marginal log likelihood, we optimize the evidence lower bound (ELBO)

$$\mathcal{F}(\theta, Z, \mathbf{m_u}, S_\mathbf{u}) := \mathbb{E}_{q(f)} \log p(y|f) - \mathbf{KL}(\phi(\mathbf{u})||p(\mathbf{u})),$$

$$q(f) = \int p(f|\mathbf{u})\phi(\mathbf{u})d\mathbf{u}.$$

ONLINE VARIATIONAL CONDITIONING





(a) $t = 20$ (b) $t = 40$ (c) $t = 60$

Inducing points are really “pseudo-data”

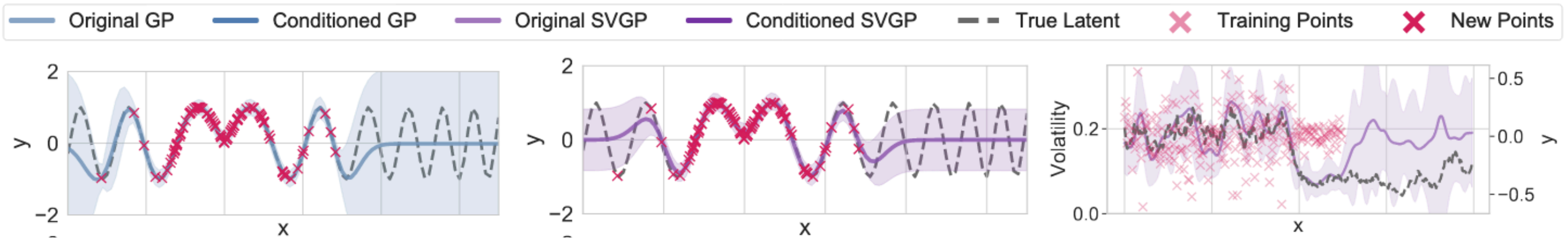
Only need to derive their noise model

Also need to move the inducing points in response to new data

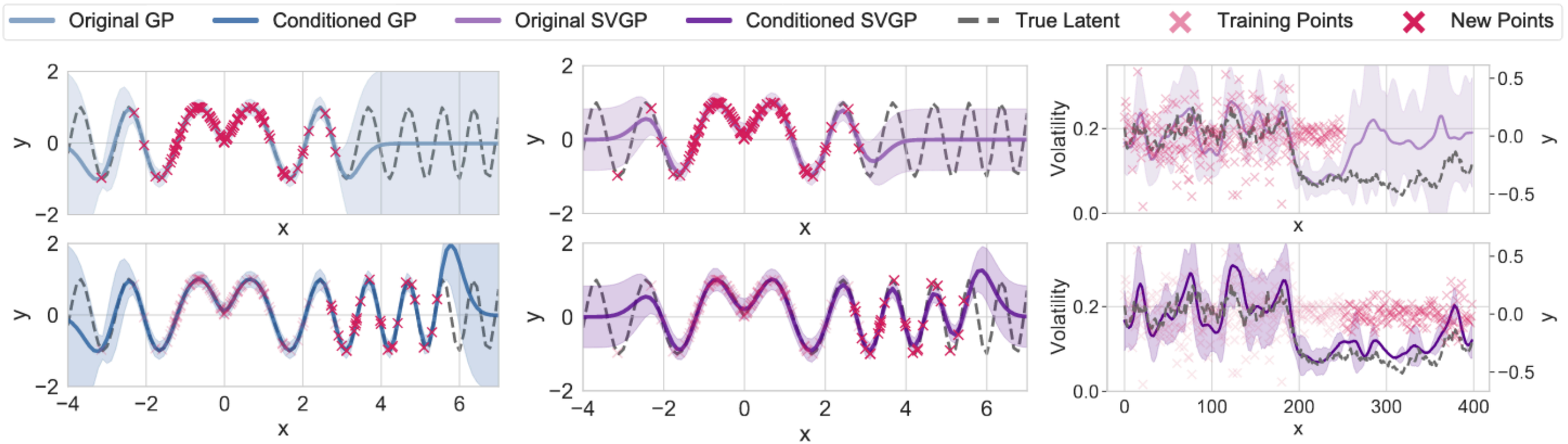
- Enables closed form updates to the parameters of the variational distribution

Work with Sam Stanton,
Andrew Gordon Wilson

CONDITIONED MODEL



CONDITIONED MODEL



(a) Exact GP

(b) SVGP + OVC

(c) SVGP + OVC volatility model

ENTROPY REDUCTION SEARCH FOR ACTIVE LEARNING

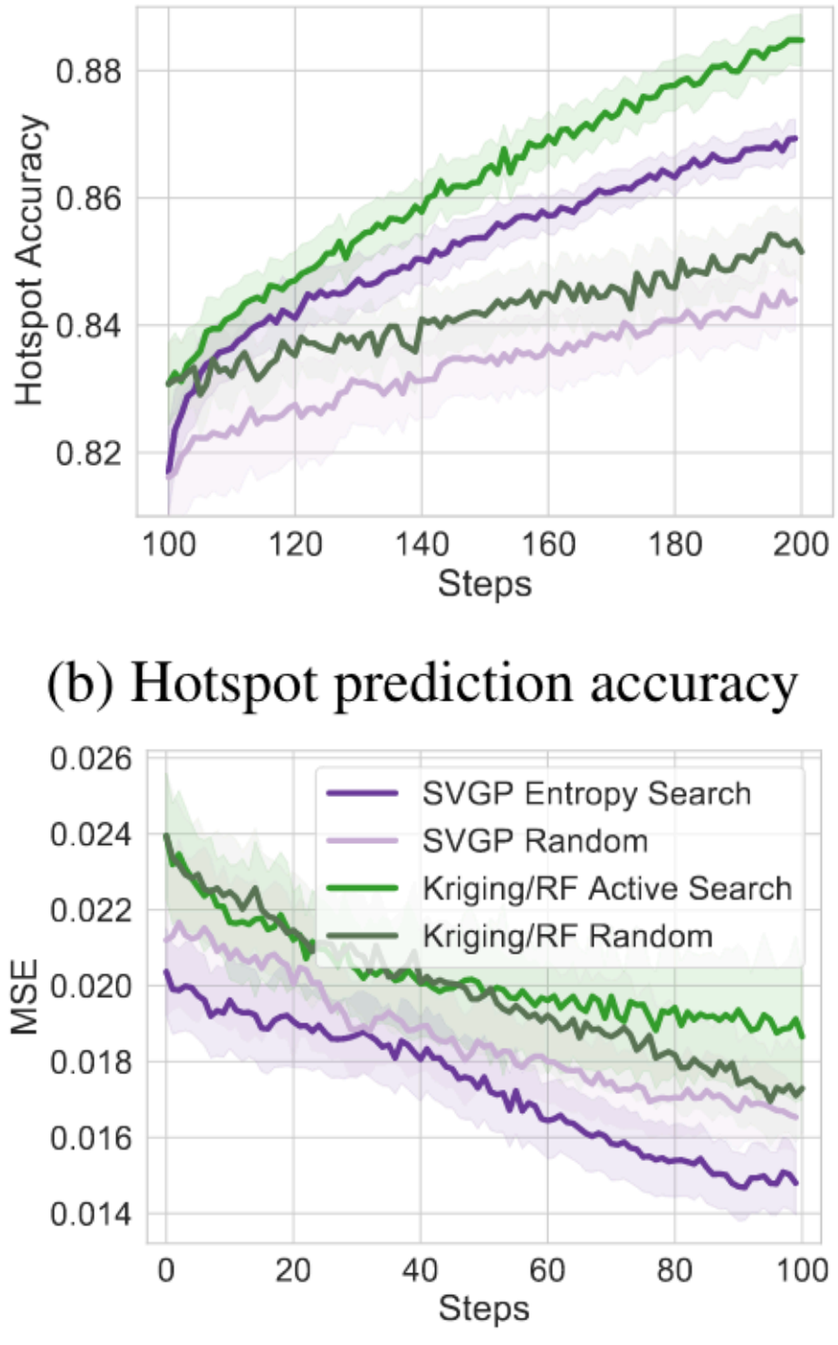
- Want to choose the (batch of) point(s) that most highly reduces the posterior entropy across the entire domain

- But responses are Binomial!

$$a_\tau(x, \mathcal{D}) := \int_{x' \in \mathcal{X}} \big(h_\tau(x', \mathcal{D}_{+x}) - h_\tau(x', \mathcal{D})\big) dx'.$$

$$h_\tau(x, \mathcal{D}) := \mathbb{E}_{p(f|\mathcal{D})}(\mathbb{H}(\text{Bernoulli}(f > \text{logit}(\tau))))$$

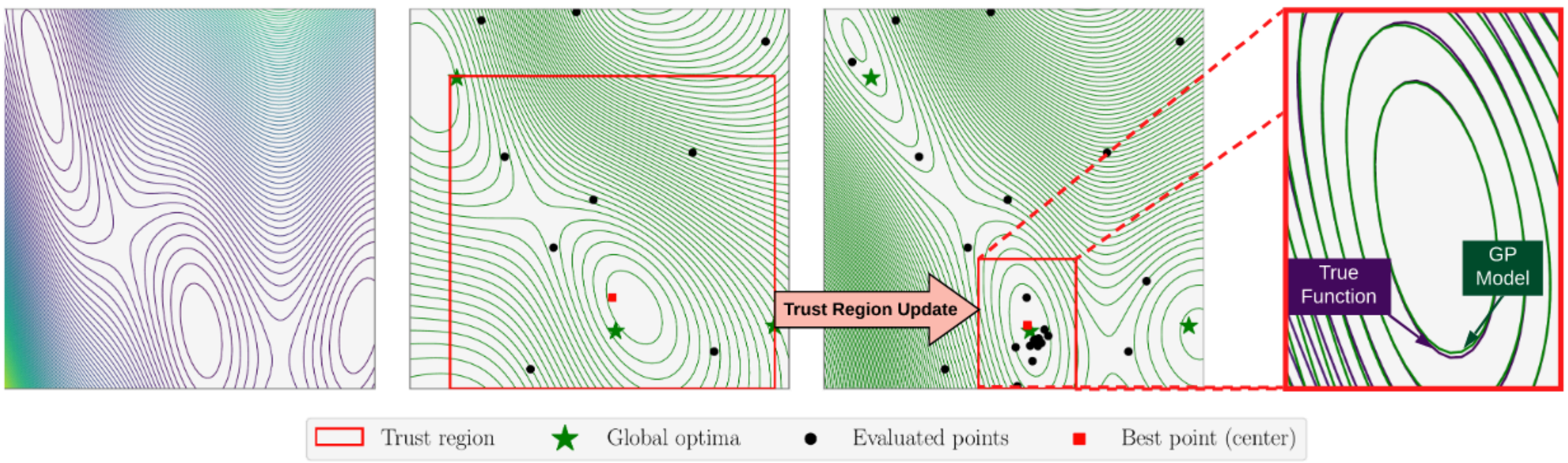
$$\approx \frac{1}{K} \sum_{i=1}^{K} 1_{f > \text{logit}(\tau)} \mathbb{H}(\text{Bernoulli}(f)),$$



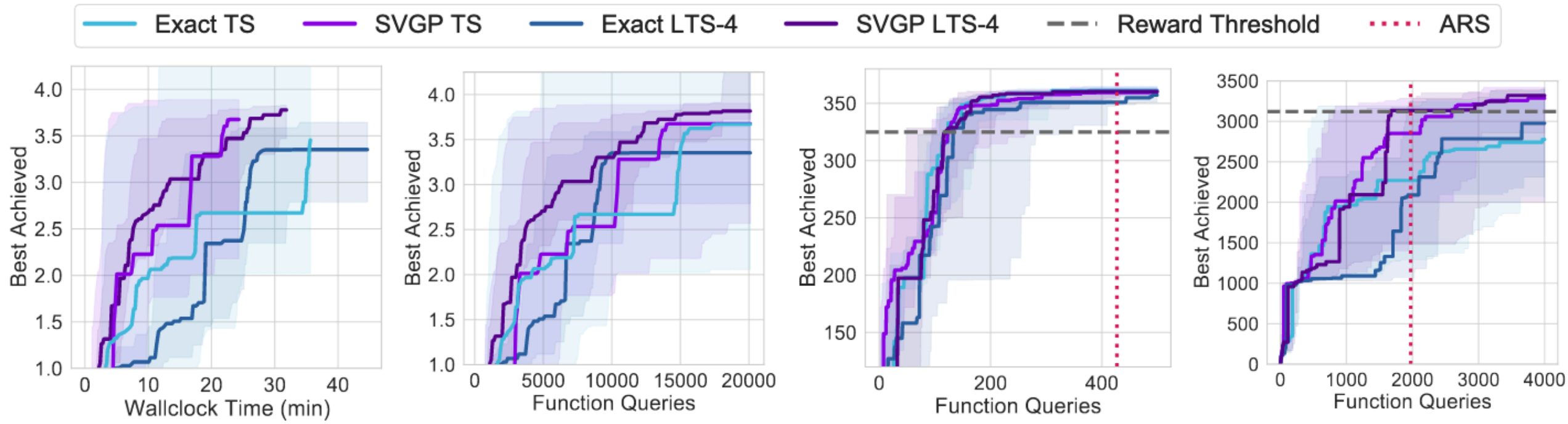
(Simulated) prevalence of schistomiatosis in Cote d'Ivoire, from Andrade-Pacheco et al, '20

TO LARGE SCALE BAYESIAN OPTIMIZATION

- Acquisition function optimization takes a long time and GPs (generally) slow down as we see new data
- Use local models and Thompson sampling (Eriksson et al, '19) ==> approach is called TrBO (trust region BO)



HIGH DIMENSIONAL BO / REINFORCEMENT LEARNING



(a) Wall-Clock time, rover, $d = 60$

(b) rover, $d = 60$

(c) `swimmer`, $d = 16$

(d) `hopper`, $d = 33$

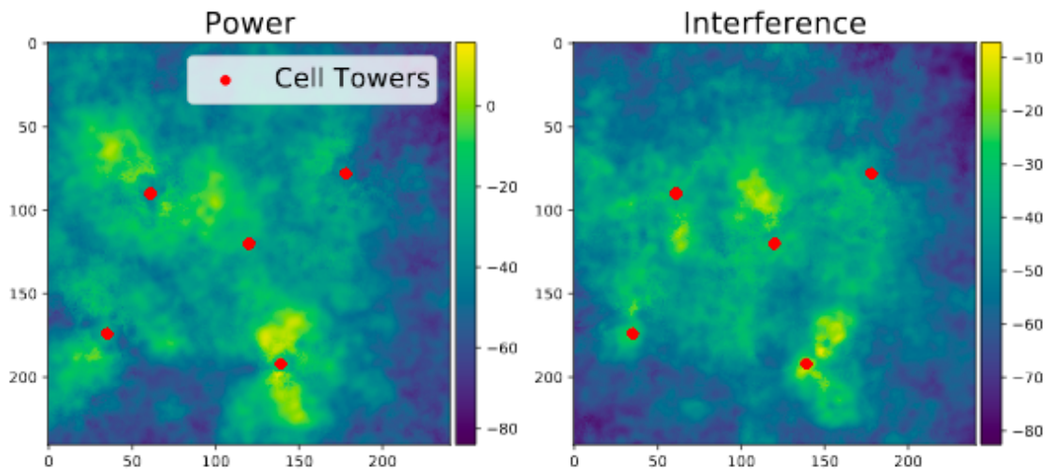
OVC is faster than sub-sampled exact GPs

Even on mujoco problems!

And performs better due to numerical stability

Optimization methods: TrBO (Eriksson et al, NeurIPS, ’19) for rover, LaMCTS + TrBO (Wang et al, NeurIPS, ’20) for swimmer / hopper.

MULTI TASK GAUSSIAN PROCESSES: PREDICTION



Want to model each pixel jointly

Prior assumption: $\mathbf{vec}(y) \sim \mathcal{N}(0, K_{XX} \otimes K_T).$

Predictive distribution is closed form: $p(f^*|X^*, X, y) = \mathcal{N}(\mu^*, \Sigma^*)$

$$\mu^* = (K_{x^*,X} \otimes K_T)(K_{XX} \otimes K_T + \sigma^2 I_{nT})^{-1} y$$

$$\Sigma^* = (K_{x^*,x^*} \otimes K_T) - (K_{x^*,X} \otimes K_T)(K_{XX} \otimes K_T + \sigma^2 I_{nT})^{-1}(K_{x^*,X}^\top \otimes K_T)$$

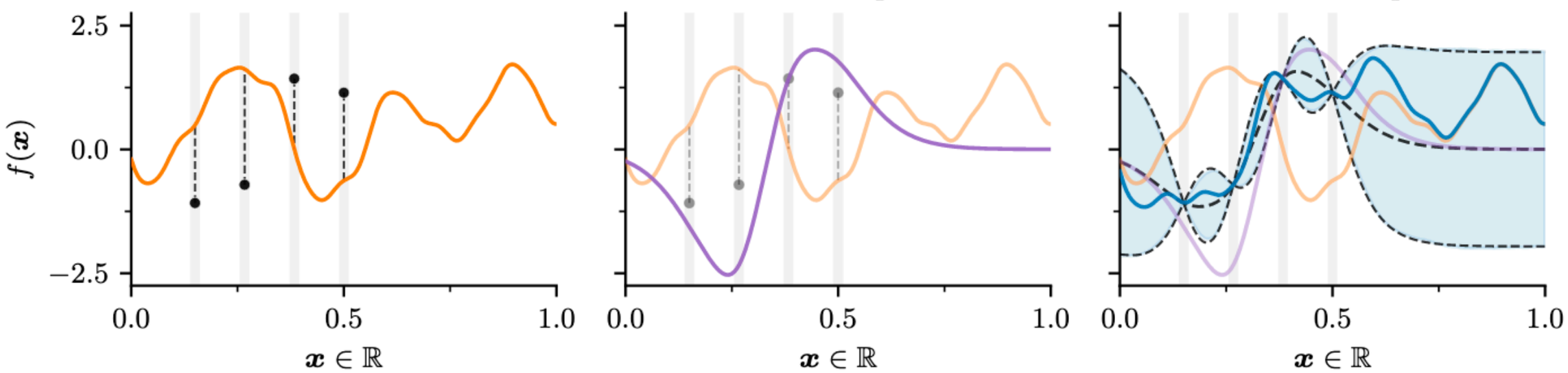
This matrix is no longer Kronecker structured, and it gets really big!

50 data points. 5000 outputs ==> Σ^* is (50*5000) x (50*5000)

MATHERON'S RULE FOR SAMPLING GAUSSIAN PROCESS POSTERIORS

Can sample from conditional Gaussian random variables via Matheron's rule (1970s)

$$f^*|(Y = y) \overset{d}{=} f^* + K_{x_{\text{test}}X}(K_{XX} + \sigma^2 I)^{-1}(y - Y - \epsilon)$$



From "Efficiently sampling functions from Gaussian Process posteriors," Wilson et al, ICML, 2020

Work with Max Balandat, Eytan Bakshy, Andrew Gordon Wilson

MATHERON'S RULE: MULTITASK SETTING

Can sample from conditional Gaussian random variables via Matheron's rule (1970s)

$$f^*|(Y = y) \overset{d}{=} f^* + K_{x_{\text{test}}X}(K_{XX} + \sigma^2 I)^{-1}(y - Y - \epsilon)$$

Prior function comes from $(f^*, Y) \sim \mathcal{N}(0, K_{(x_{\text{test}}X),(x_{\text{test}}X)}),$

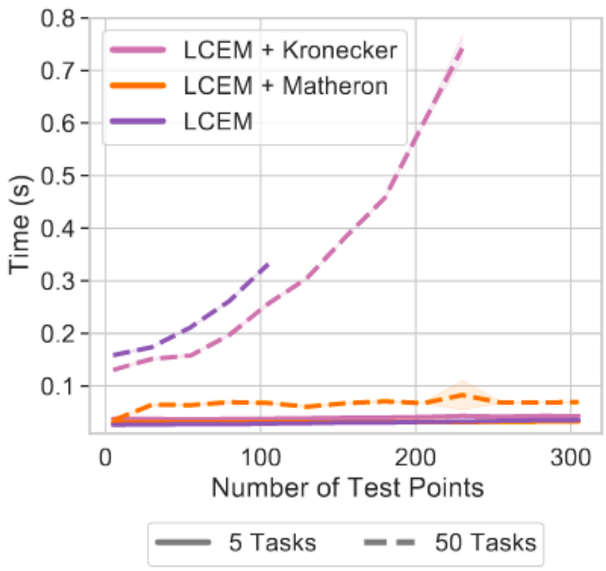
Which is structured (e.g. efficient sampling)

$$\begin{aligned} K_{\text{mt},(x_{\text{test}}X),(x_{\text{test}}X)} &= K_{(x_{\text{test}}X),(x_{\text{test}}X)} \otimes K_T = \tilde{R}\tilde{R}^\top \otimes LL^\top \\ &= (\tilde{R} \otimes L)(\tilde{R} \otimes L)^\top \end{aligned}$$

Pathwise update is a structured solve and a Kronecker MVM.

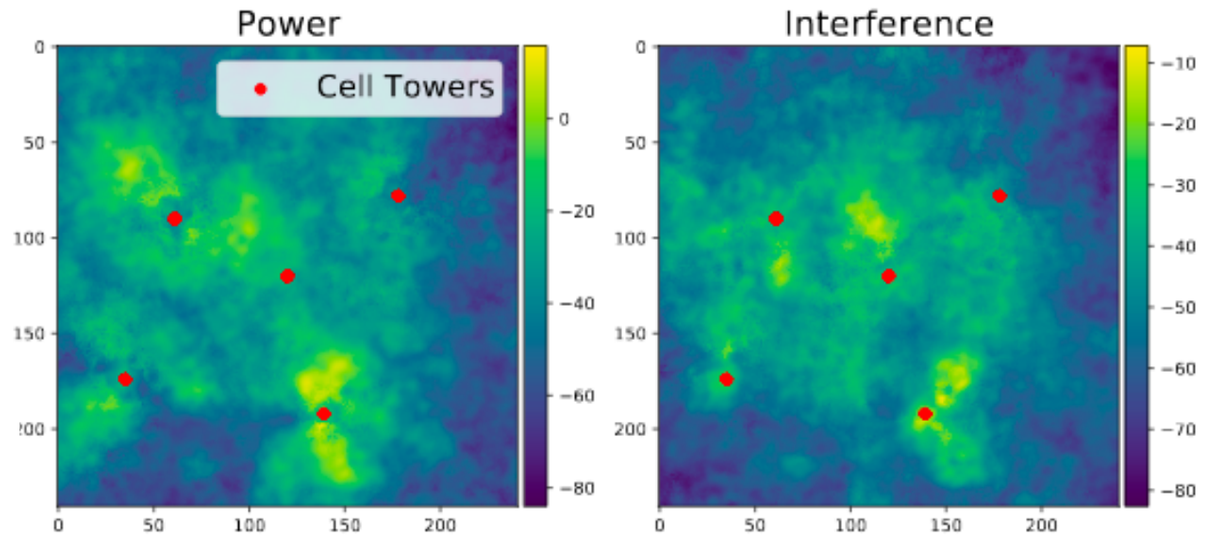
$$(\otimes_{i=}^{d} K_i + \sigma^2 I)^{-1} z = \otimes_{i=1}^{d} Q_i (\otimes_{i=1}^{d} \Lambda_i + \sigma^2 I)^{-1} \otimes_{i=1}^{d} Q_i^\top z.$$

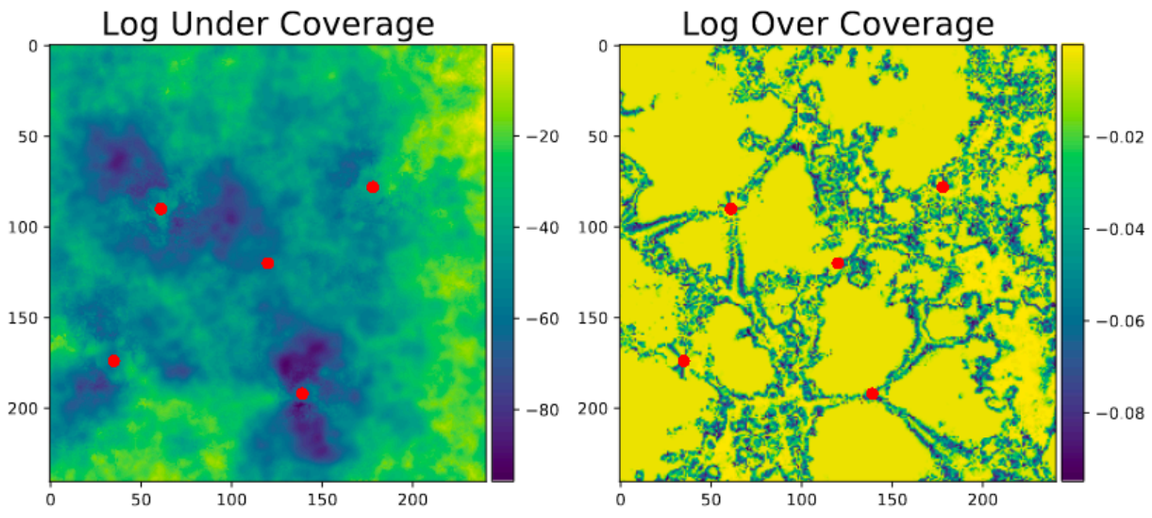
Posterior sampling is O(n^3 + t^3) time rather than O($n^3 t^3$) time.



LARGE OUTPUT COMPOSITE BAYESIAN OPTIMIZATION

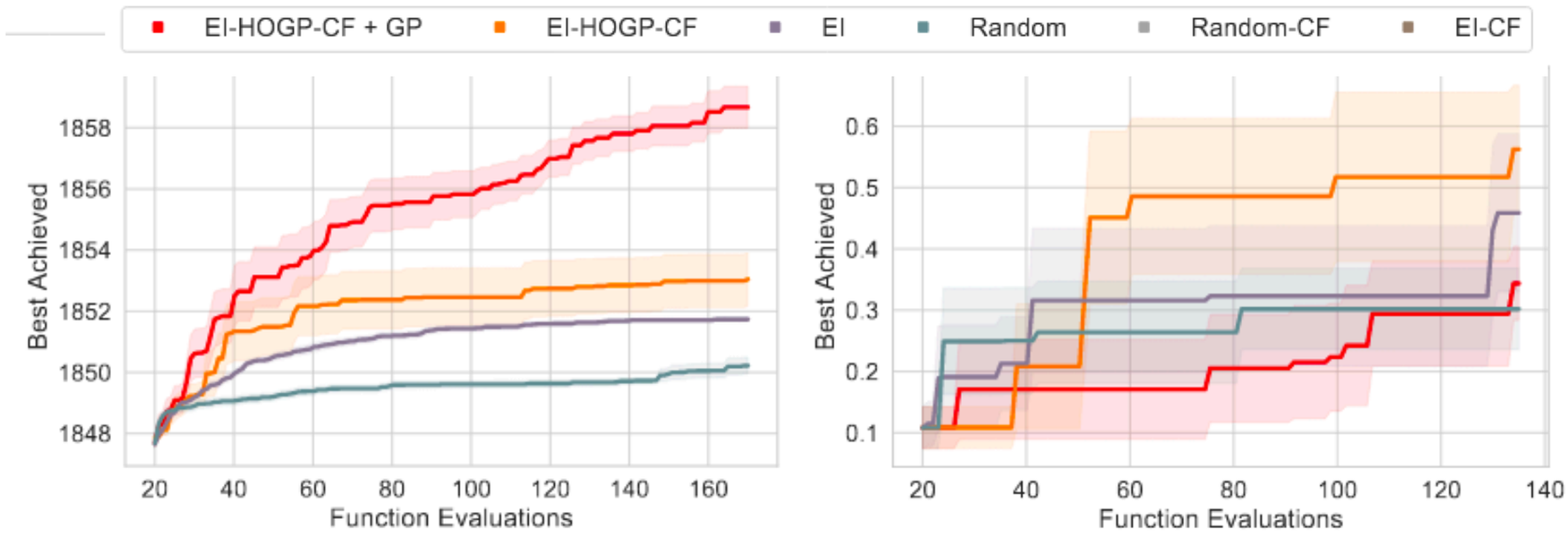
Objective is a nonlinear function of the responses:





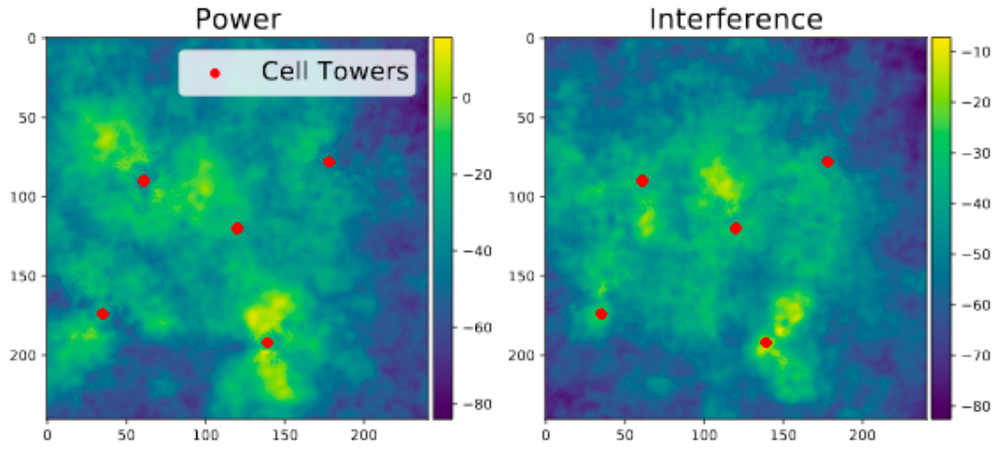
Final objective value

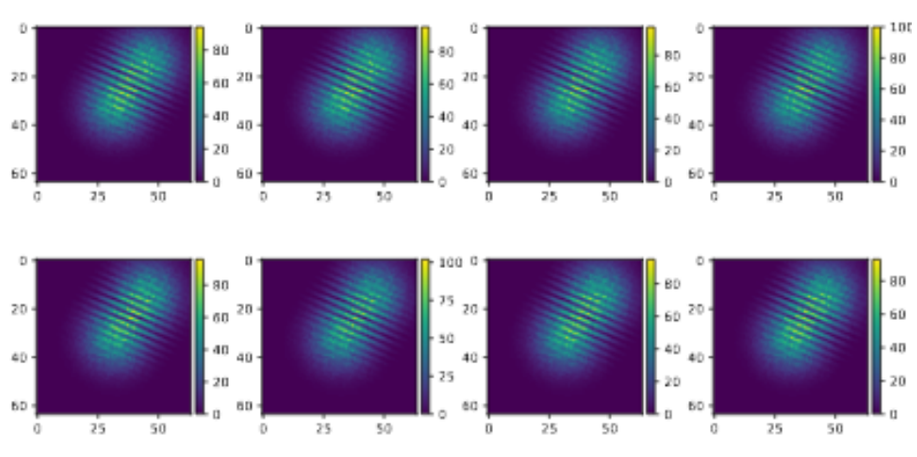
LARGE OUTPUT COMPOSITE BAYESIAN OPTIMIZATION



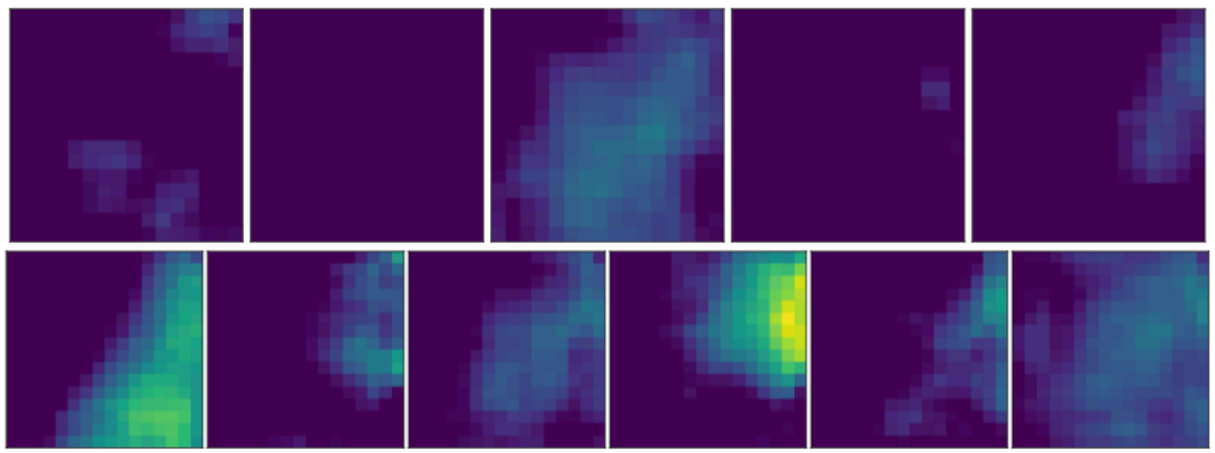
(c) Cell-tower coverage ($t = 2 \times 50 \times 50, d = 30$).

(d) Optics ($t = 16 \times 64 \times 64, d = 4$).

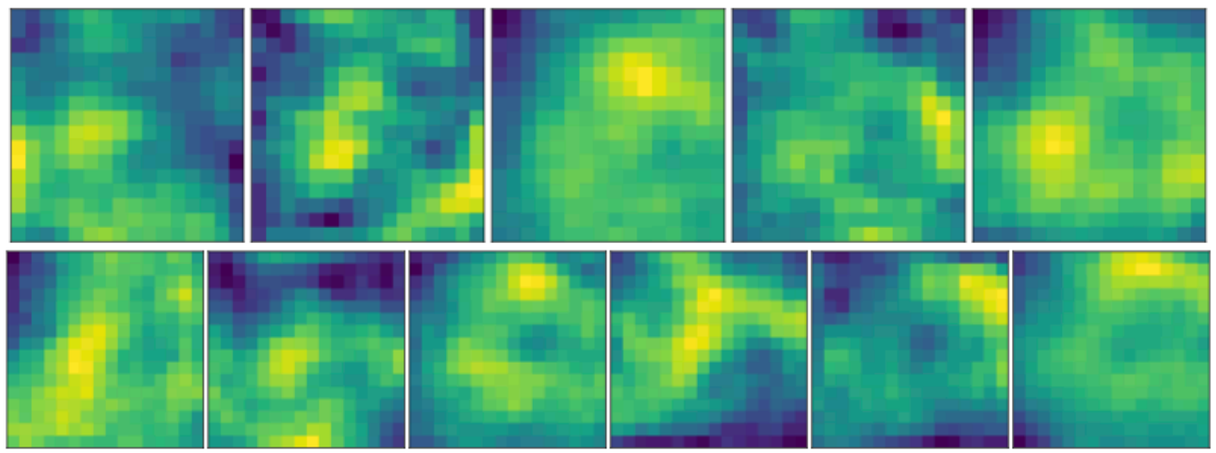




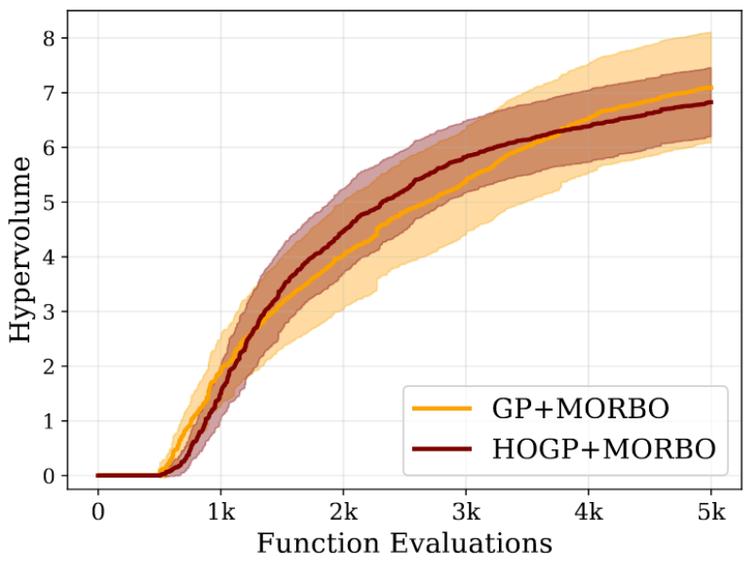
COMBINING TRBO WITH COMPOSITE BAYESIAN OPTIMIZATION

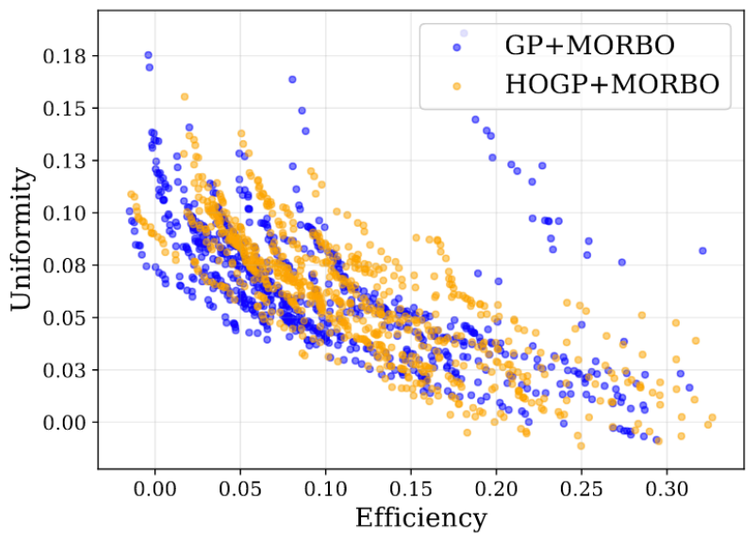


(a) Example image outputs from simulations.



(b) Example optimized image outputs from simulations.



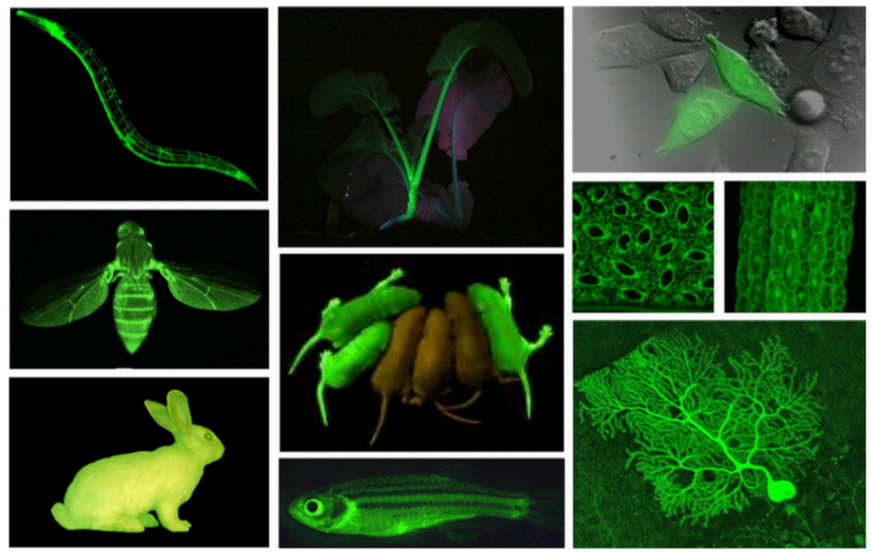


Optical design problem: t = 11 x 11 x 16, d = 104, q = 50

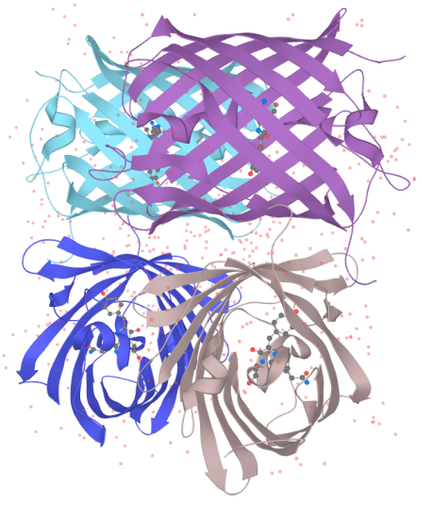
Using MORBO (multi-objective TrBO extension), Daulton et al, ‘21

BAYESIAN OPTIMIZATION FOR PROTEIN DESIGN

- Goal is to optimize red fluorescent proteins with respect to their emission intensity and wavelength (aka make them glow brighter)
- Proxy tasks: minimize change in free Gibbs energy due to folding (stability) and maximize solvent-accessible surface area (SASA).
- Collaboration w/ BigHat Biosciences.



Transgenic animals w/ green FP

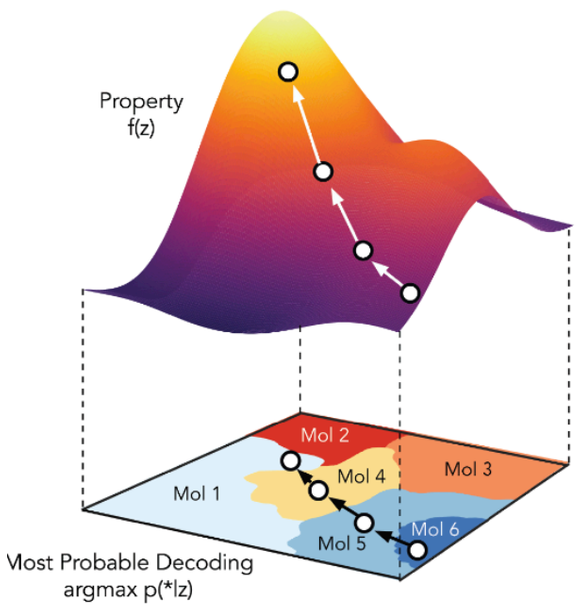


Un-optimized dsRed structure

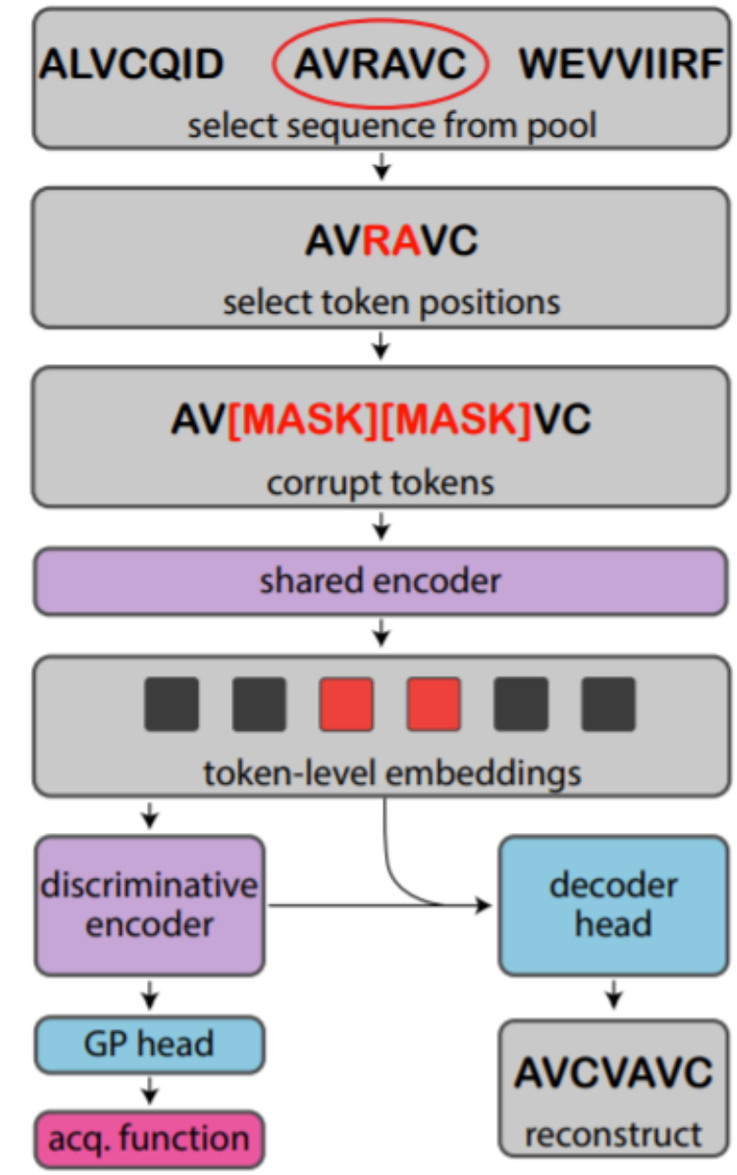
Also w/ Sam Stanton, Nate Gruver, Andrew Gordon Wilson

LAMBO: DISCRETE BAYESIAN OPTIMIZATION W/ DEEP MODELS

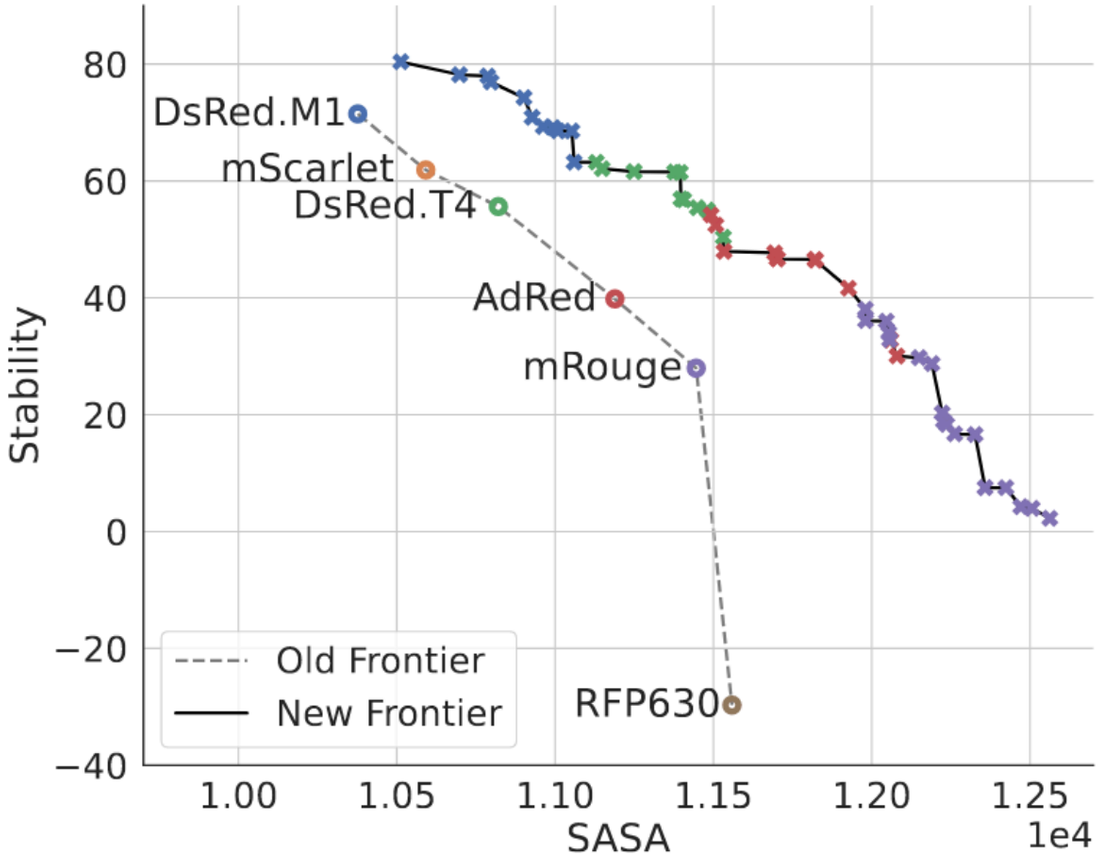
- Uses a denoising auto encoder (DAE) with a multi-task GP head
- Then optimizes in latent space to and decodes back to generate new sequences



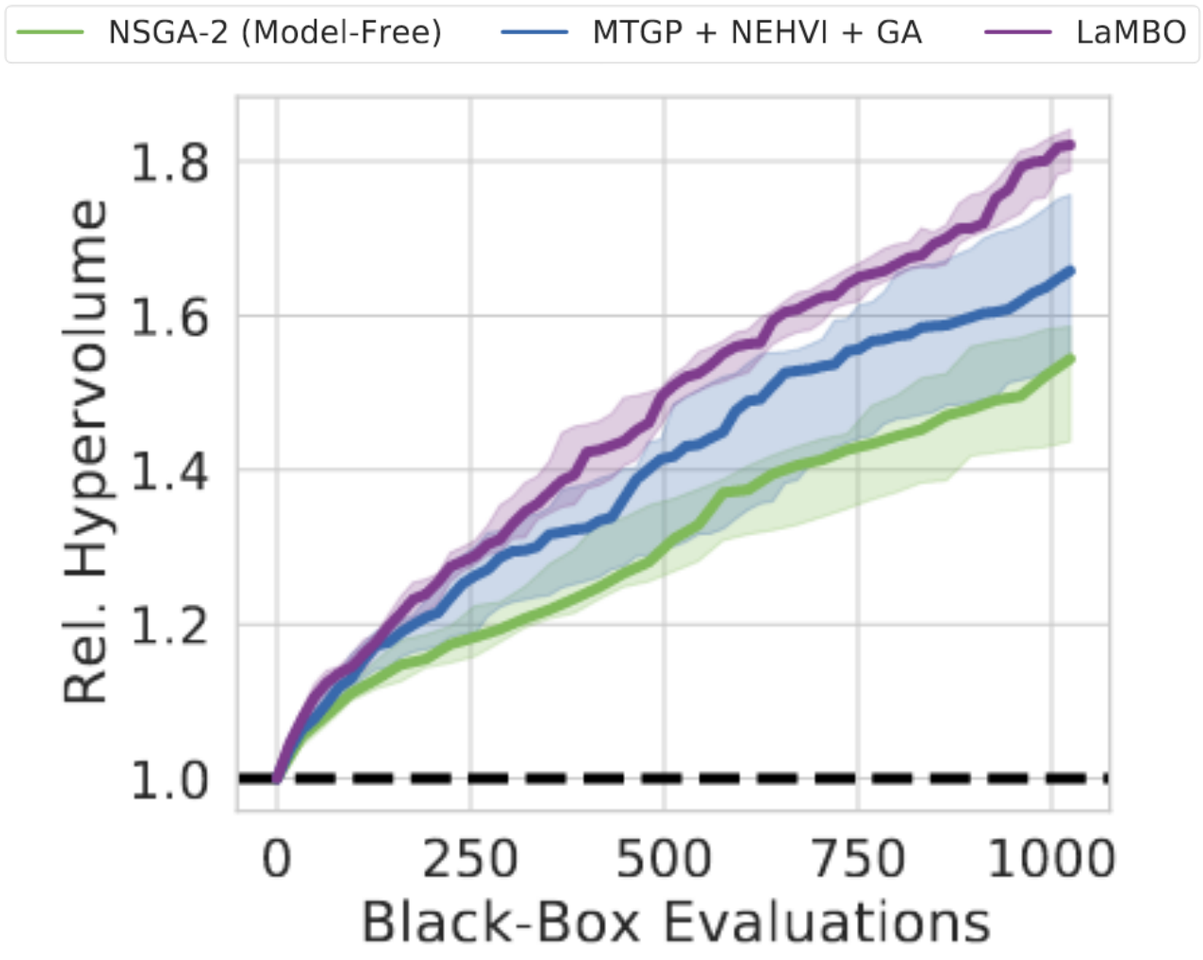
Schematic from Gomez-Bombarelli et al, ‘17



LAMBO: DISCRETE BAYESIAN OPTIMIZATION W/ DEEP MODELS



New Pareto frontier after optimization.



Optimization results on the RFP task.

Lab validation is hopefully coming in the near future as well.

A FUTURE PERSPECTIVE

- The use of big models will only become more widespread.
- GPs and Bayesian methods are not going away.
  - Need the data efficiency and predictive variances for many bespoke tasks.
  - Combining approaches (GPs + DNNs) will probably become more popular.
- Domain knowledge seems unlikely to go away either.
- *We're only at the beginning of ML decision-making loops.*

THANKS TO

- Andrew Wilson
- Pavel Izmailov, Polina Kirichenko, Timur Garipov, Dmitry Vetrov (SWAG, Subspace Inference)
- Greg Benton, Sanae Lofti (SPRO)
- Sam Stanton (OVC, LaMBO), Ian Delbridge (WISKI)
- Max Balandat, Eytan Bakshy, Qing Feng, Dominic Meiser (MTGPs)
- Nate Gruver, Phil Maffetone, Emily Delaney, Peyton Greenside (LaMBO)
- The rest of the lab as well

QUESTIONS?

Slides at https://wjmaddox.github.io soon.