Karlsruher Institut für Technologie

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

CLUSTER OF EXCELLENCE
QUANTUM UNIVERSE

DASHH

Estimating Uncertainties for Trained Neural Networks

Sebastian Bieringer[1], Gregor Kasieczka, Maximilian F. Steffen, Mathias Trabs

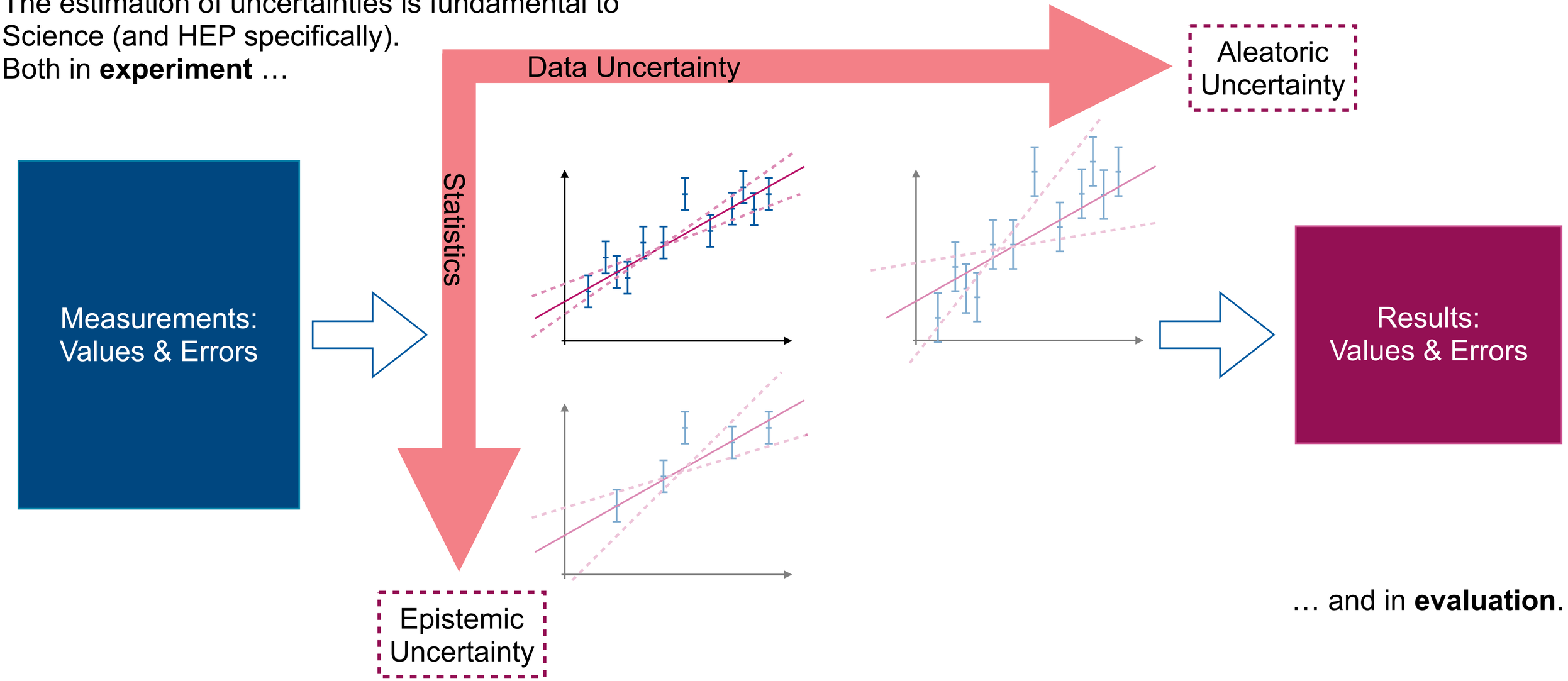
[1]Institut für Experimentalphysik, Universität Hamburg, Germany

sebastian.guido.bieringer@uni-hamburg.de

ML4jets 2022

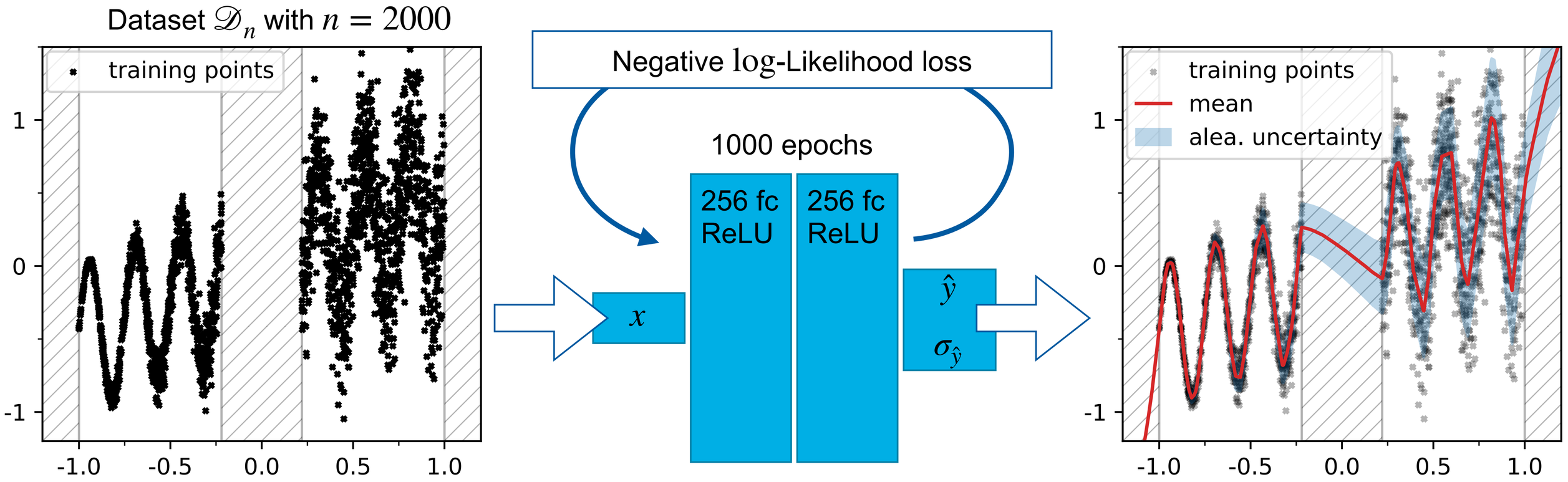
Introduction

The estimation of uncertainties is fundamental to Science (and HEP specifically).
Both in **experiment** …



… and in **evaluation**.

Introduction



What about the uncertainty due to **lack of knowledge**, i.e. the epistemic uncertainty?

Bayesian formula $\Rightarrow$ Encoded in posterior $p(\theta \,|\, \mathscr{D}_n)$

Deep Ensembles

Description (1612.01474):

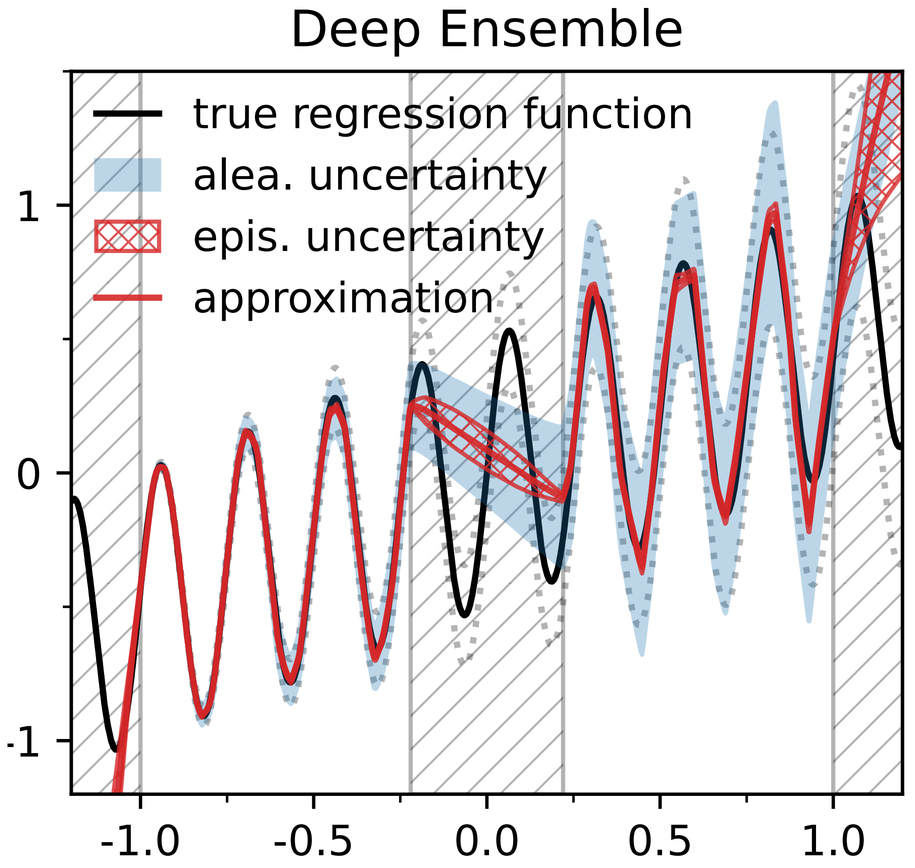
- Train multiple networks, here 50
- Sample from the minima of the loss function rather than the posterior $p(\theta \,|\, \mathscr{D})$

Pros & Cons:

+ Stable
- Very slow
- Does not sample the posterior

Adaptations:

- Change hyperparameters and network architecture (2006.08573)
- Combine the ensemble in one network (2210.09767)



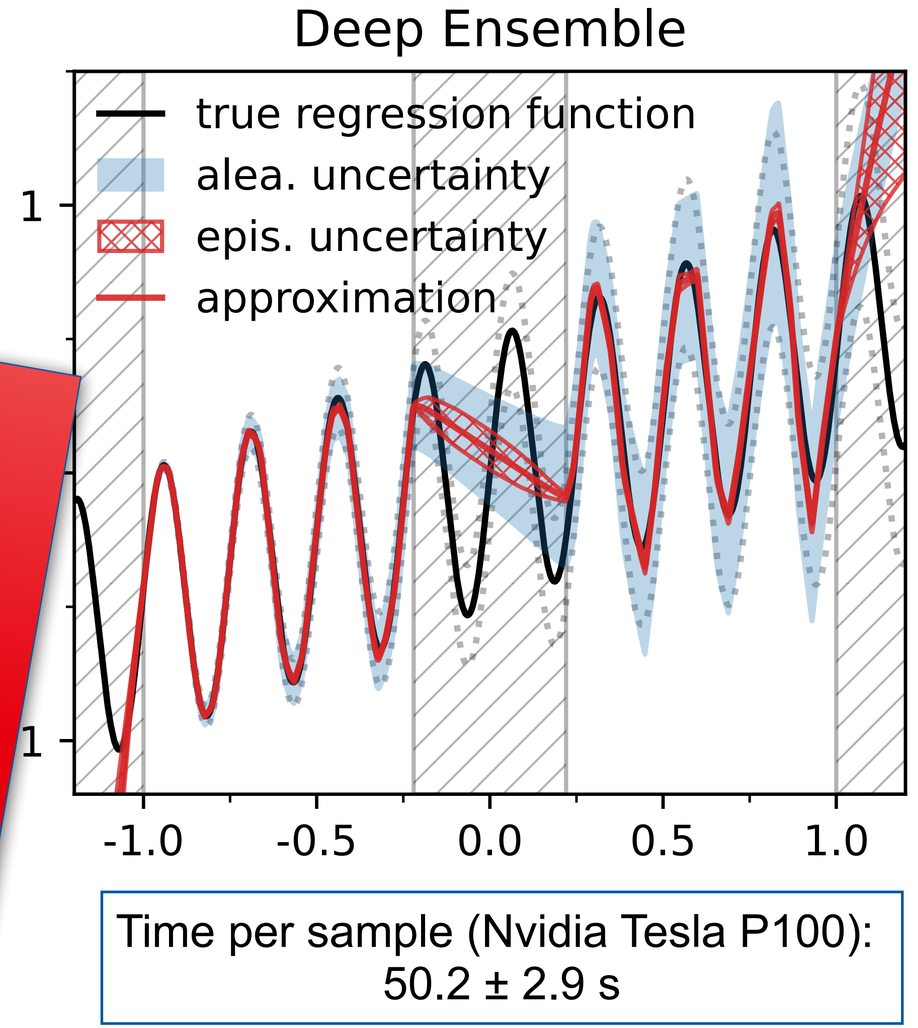
Time per sample (Nvidia Tesla P100):
50.2 ± 2.9 s

Deep Ensembles

Description (1612.01474):

- networks, here 50
- loss function rather than the
- Combine the ensem

Reliable
Slow



Time per sample (Nvidia Tesla P100):
50.2 ± 2.9 s

„Bayesian Neural Networks“

Mean Field Gaussian Variational Inference

Description (1505.05424):

- Estimate the posterior $p(\theta \mid \mathscr{D})$ with a simpler distribution $q(\theta)$
- Infer with gradient descent:

$$L_n(\hat{f}_\vartheta; \mathscr{D}_n) = \mathrm{KL}(p(\theta \mid \mathscr{D}_n) \mid q(\theta)) = -\int d\theta\, q(\theta)\, \log p(\mathscr{D}_n \mid \theta) + \mathrm{KL}(q(\theta) \mid p(\theta))$$

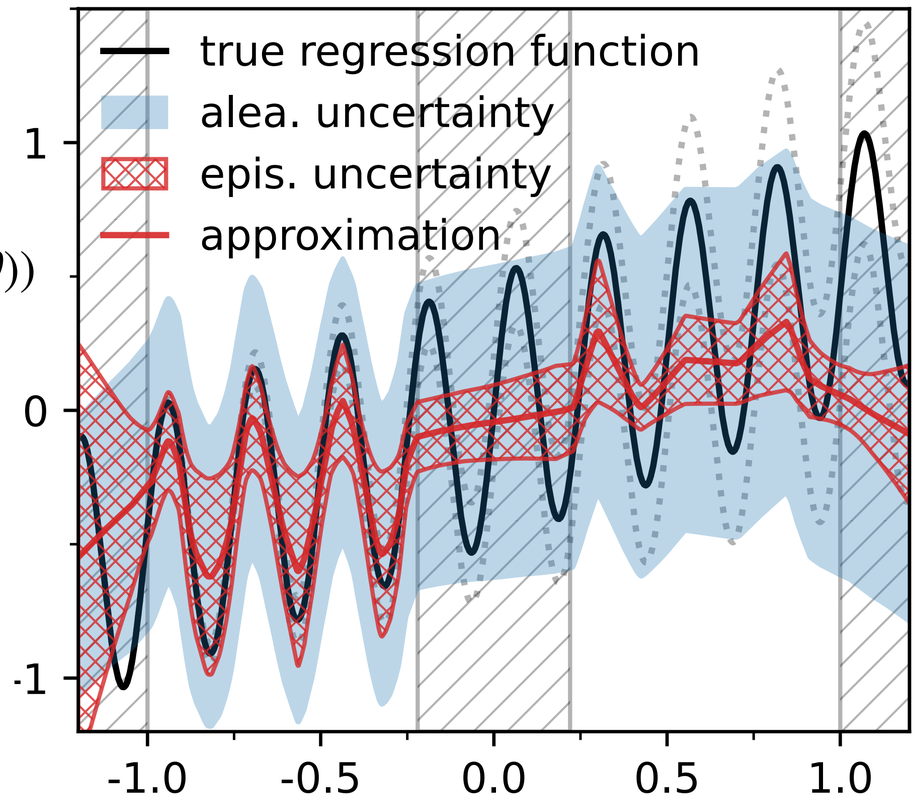
Pros & Cons:

+ Fast posterior sampling, active learning possible
- Additional loss term with high variance → influences performance
- Assumption: Posterior has uncorrelated Gaussian shape
- Doubles the number of parameters

Adaptations:

- Noise-Contrastive Priors (1807.09289), Flipout Layers (1803.04386)
- … next talk?

VI



Time per training (Nvidia Tesla P100):
216 ± 15 s
Time per sample (Nvidia Tesla P100):
2.0 ± 0.1 ms

„Bayesian Neural Networks“

Mean Field Gaussian Variational Inference

Description (1505.05424):

- Estimate the posterior $p(\theta \mid \mathscr{D})$ with a simpler distribution $q(\theta)$
- Infer with gradient descent:

$$L_n(\hat{f}_\vartheta; \mathscr{D}_n) = \mathrm{KL}(p(\theta \mid \mathscr{D}_n) \mid q(\theta)) = -\int d\theta\, q(\theta)\, \log p(\mathscr{D}_n \mid \theta) + \mathrm{KL}(q(\theta) \mid p(\theta))$$

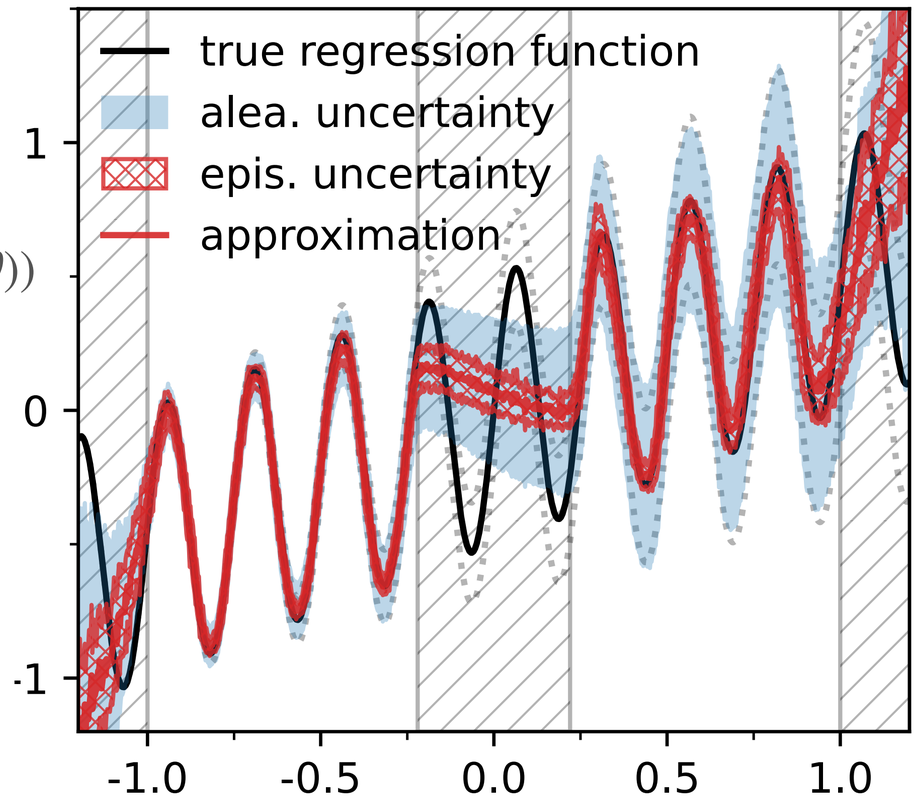
Pros & Cons:

+ Fast posterior sampling, active learning possible
- Additional loss term with high variance → influences performance
- Assumption: Posterior has uncorrelated Gaussian shape
- Doubles the number of parameters

Adaptations:

- Noise-Contrastive Priors (1807.09289), **Flipout Layers** (1803.04386)
- … next talk?

VI + Flipout



Time per training (Nvidia Tesla P100):
133 ± 2 s
Time per sample (Nvidia Tesla P100):
1.5 ± 0.1 ms

„Bayesian Neural Networks“

Mean Field Gaussian Variational Inference

Description (1505.05424):

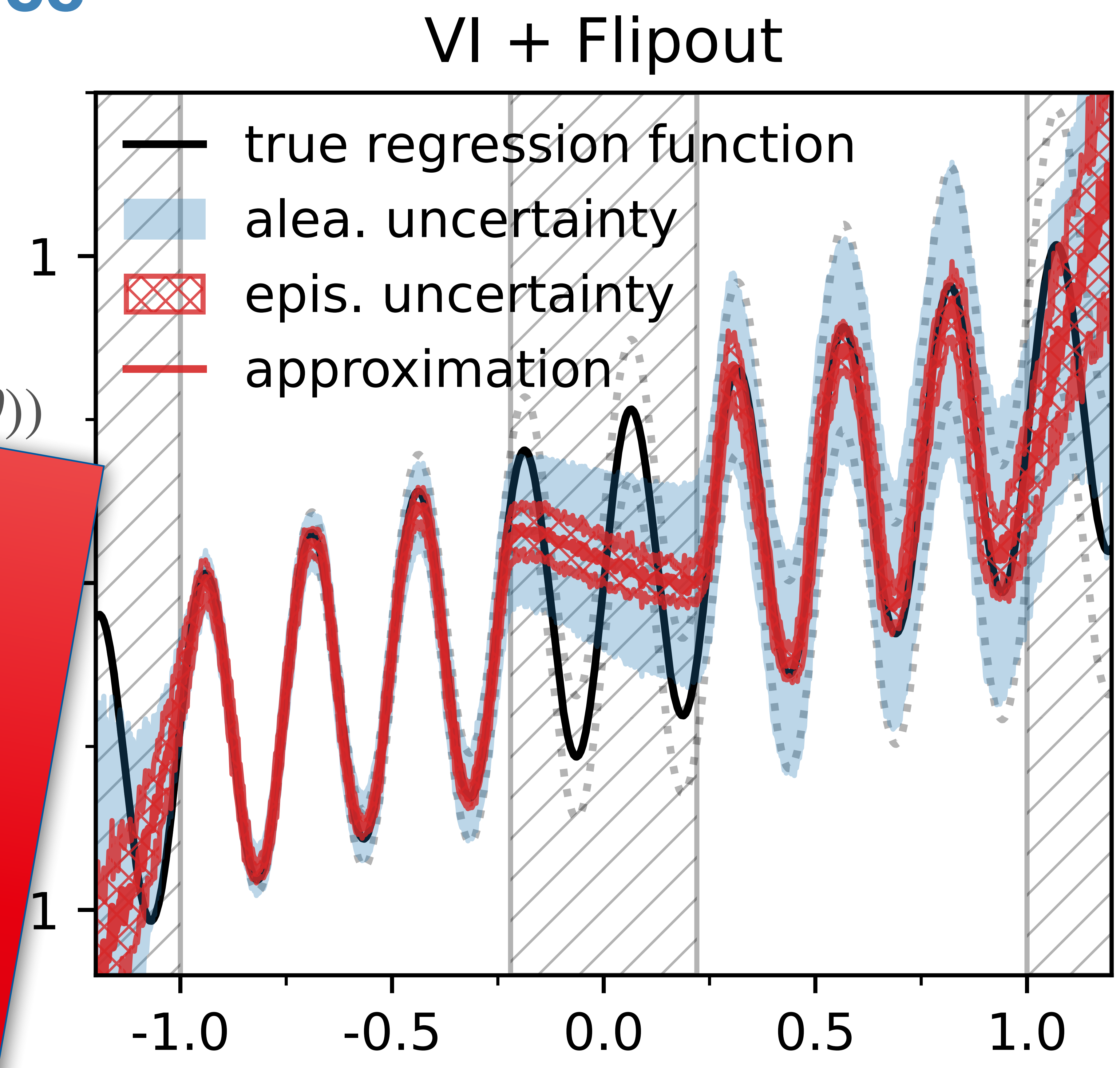
sterior $p(\theta|\mathscr{D})$ with a simpler distribution $q(\theta)$

$\mathrm{KL}(q(\theta)|p(\theta))$

Slow

Influences the fitting

- Noise-Contrastive (1803.04386)
- … next talk?



Time per training (Nvidia Tesla P100):
133 ± 2 s
Time per sample (Nvidia Tesla P100):
1.5 ± 0.1 ms

Laplace Approximation

Description (2106.14806):

- Estimate optimal network parmeters $\theta^\star$
- LA of the posterior $p(\theta \,|\, \mathscr{D})$ using the Hessian matrix

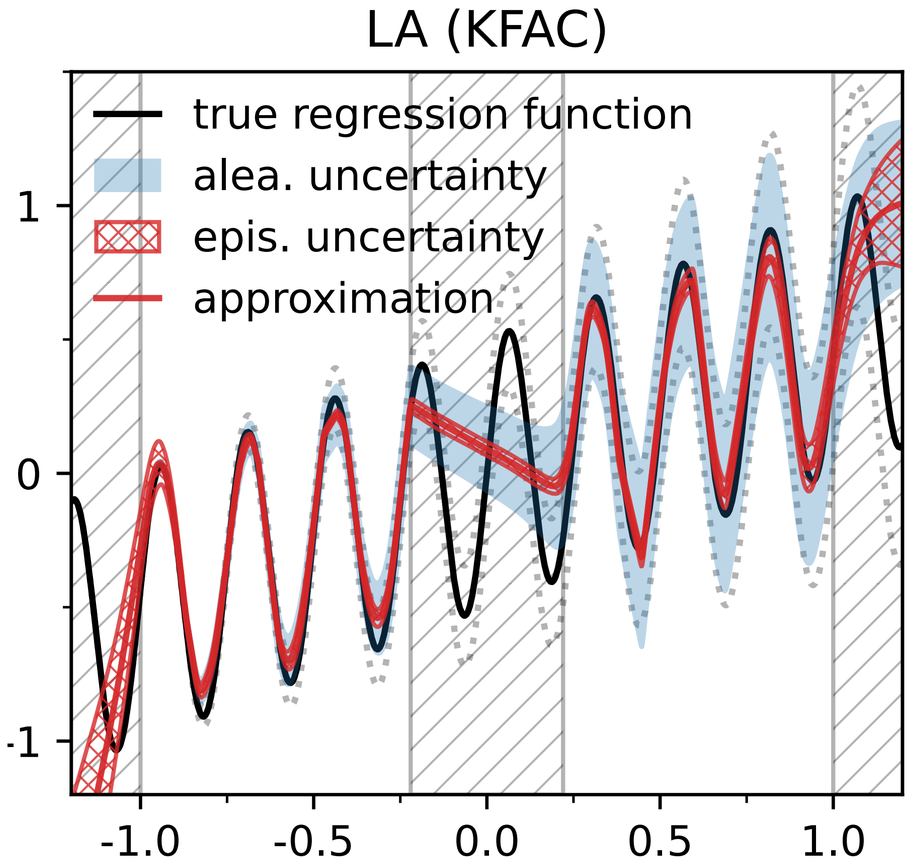
$$p(\theta \,|\, \mathscr{D}) \approx \mathcal{N}(\theta^\star, \Sigma) \text{ with } \Sigma = (\nabla^2_\theta L_{\text{NLL}}(\hat{f}_\theta; \mathscr{D})\,|_{\theta^\star})^{-1}$$

Pros & Cons:

+ **FAST** posterior sampling, active learning possible
- Scales badly with network size depending on chosen approximation
- Assumption: Posterior has correlated Gaussian shape

Adaptations:

- Kronecker-factored Approximate Curvature (1503.05671)
- Other low rank approximations of the Hessian (2006.11631)



Time per LA (Nvidia Tesla P100):
84.1 ± 0.2 ms
Time per sample (Nvidia Tesla P100):
1.2 ± 0.2 ms

Laplace Approximation

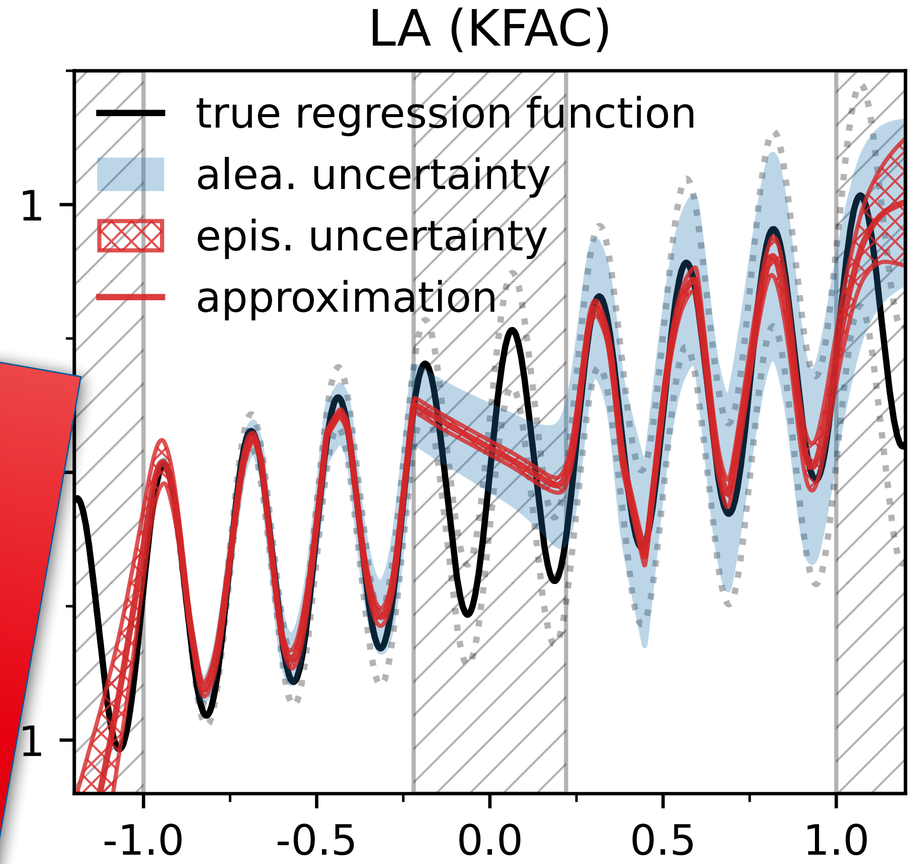
Description (2106.14806):

- network parmeters $\theta^\star$
- Hessian matrix

Fast
Available on trained networks

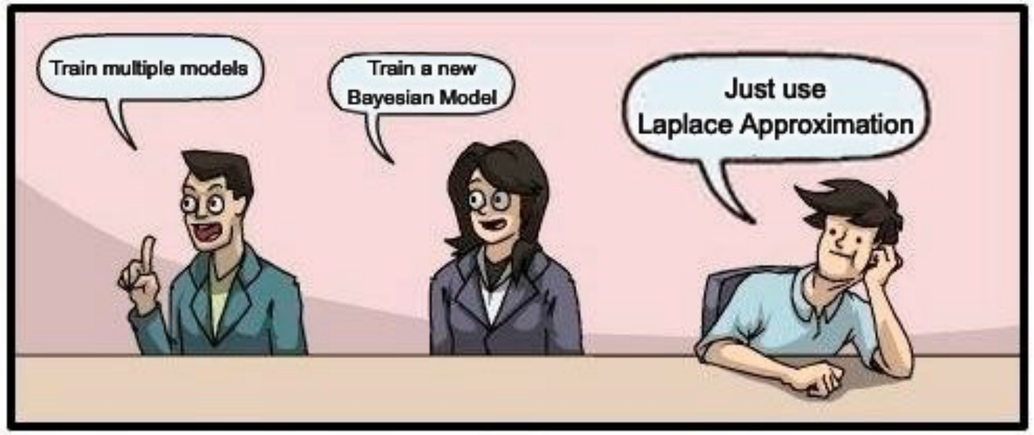
Adaptations:

- Kronecker-factored Approximate Cu
- Other low rank approximations of the Hessian (2006.11631)



Time per LA (Nvidia Tesla P100):
84.1 ± 0.2 ms
Time per sample (Nvidia Tesla P100):
1.2 ± 0.2 ms


We need uncertainty estimations, stat!
Train multiple models
Train a new Bayesian Model
Just use Laplace Approximation
imgflip.com



approximate posterior

sample from posterior

Cyclic sgLD

Description (1902.03932):

- (Pretrain to optimal parameters $\theta^{(0)} = \theta^{\star}$)
- Construct a Markov-Chain with invariant distribution

$$p(\theta \,|\, \mathscr{D}) = \exp\left(-\lambda_{\mathrm{LD}} L_{\mathrm{NLL}}(\hat{f}_{\theta}; \mathscr{D})\right)$$

- Stochastic Gradient Langevin Dynamics (sgLD):

$$\theta^{(k+1)} = \theta^{(k)} - \eta_k \nabla_{\theta} L_{\mathrm{NLL},n}(\theta^{(k)}) + \sqrt{\frac{2\eta_k}{\lambda_{\mathrm{LD}}}}\epsilon_k \text{ with } \epsilon_k \sim \mathcal{N}(0,1)$$

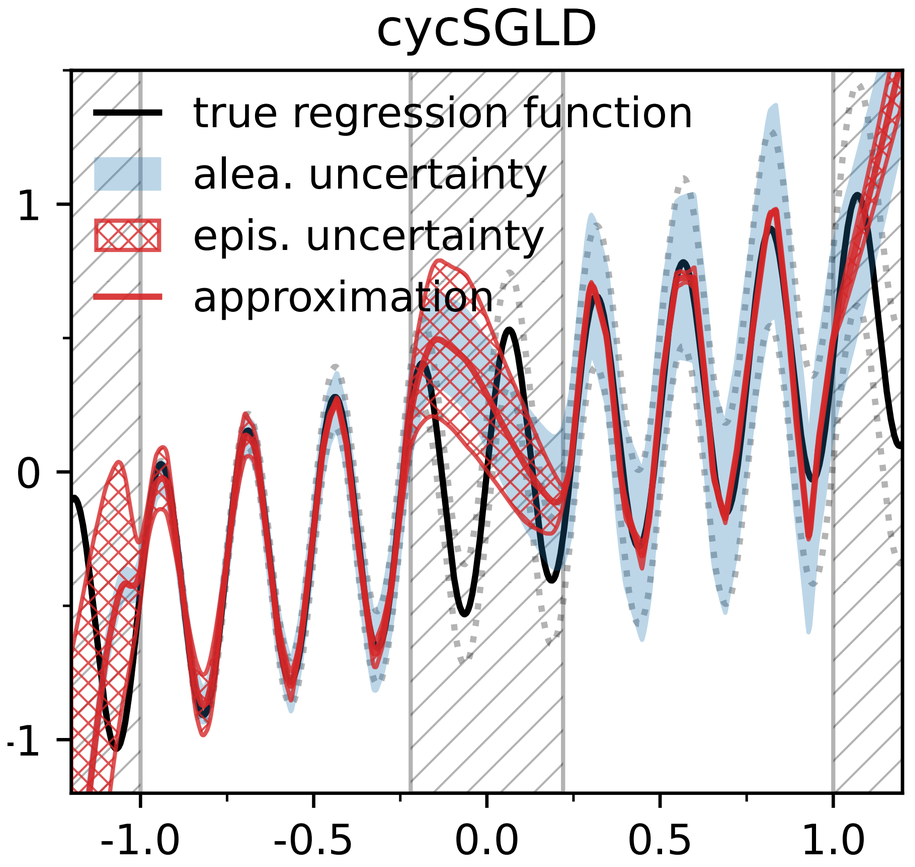
- Cyclic scheduling of stepsize η_k

Pros & Cons:

+ Exact sampling from the posterior
+ Good out-of-distribution detection
- Slow mixing rates
- Strongly dependent on the scheduling parameters

Adaptations:

- Hamiltonian Monte-Carlo (HMC) (1902.03932)



Time per sample (Nvidia Tesla P100):
112 ± 8 s

Cyclic sgLD

Description (1902.03932):

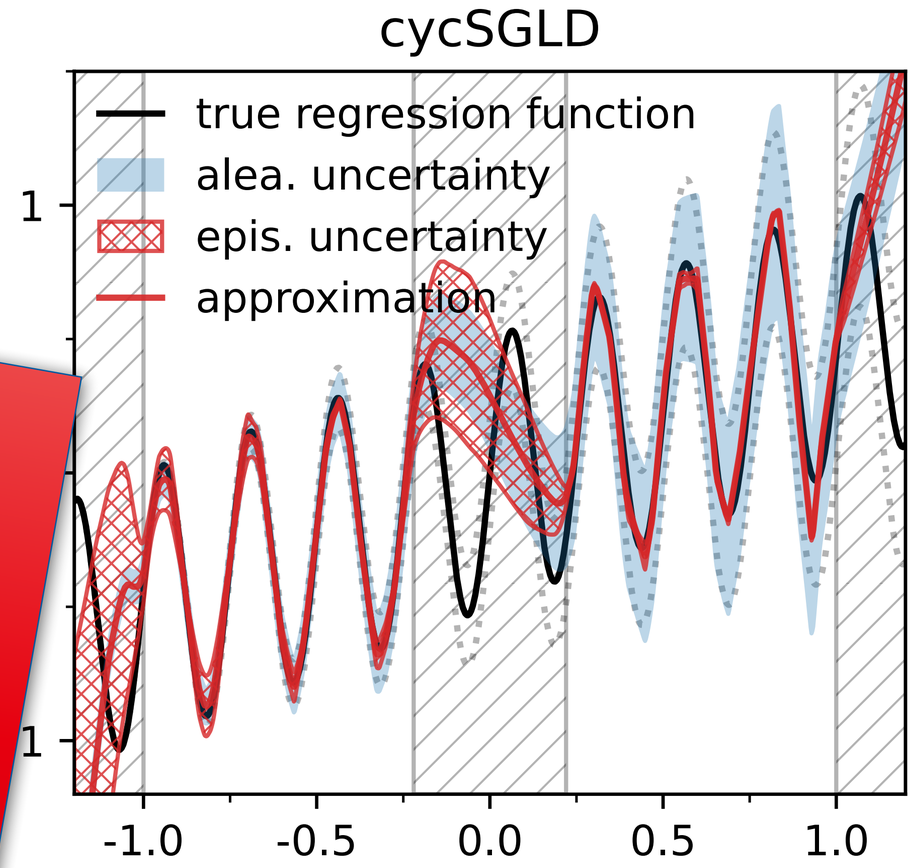
- timal parameters $\theta^{(0)} = \theta^{\star}$)
- ariant distribution

Slow
Available on trained networks
Good epistemic uncertainties

- Slow mixing rate
- Strongly dependent on the scheduling

Adaptations:

- Hamiltonian Monte-Carlo (HMC) (1902.03932)



Time per sample (Nvidia Tesla P100):
112 ± 8 s

Cyclic Stochastic Gradient Metropolis-Hastings

Description (2204.12392):

- Accept $\theta^{(k+1)} = \tau^{(k)}$ where

$$\tau^{(k)} \sim q(\,\cdot\,|\,\theta^{(k)}) = \mathcal{N}\left(\theta^{(k)} - \eta_k \nabla_\theta L_{\mathrm{NLL},n}(\theta^{(k)}), \sqrt{\frac{2\eta_k}{\lambda_{\mathrm{LD}}}}\right)$$

  with (stochastic) acceptance probability

$$\alpha(\tau^{(k)}, \theta^{(k)}) = \min\left(\frac{\exp\left(-\lambda L_{\mathrm{NLL},n}(\tau^{(k)})\right) q(\theta^{(k)}\,|\,\tau^{(k)})}{\exp\left(-\lambda L_{\mathrm{NLL},n}(\theta^{(k)})\right) q(\tau^{(k)}\,|\,\theta^{(k)})}, 1\right)$$

  to correct the invariant distribution

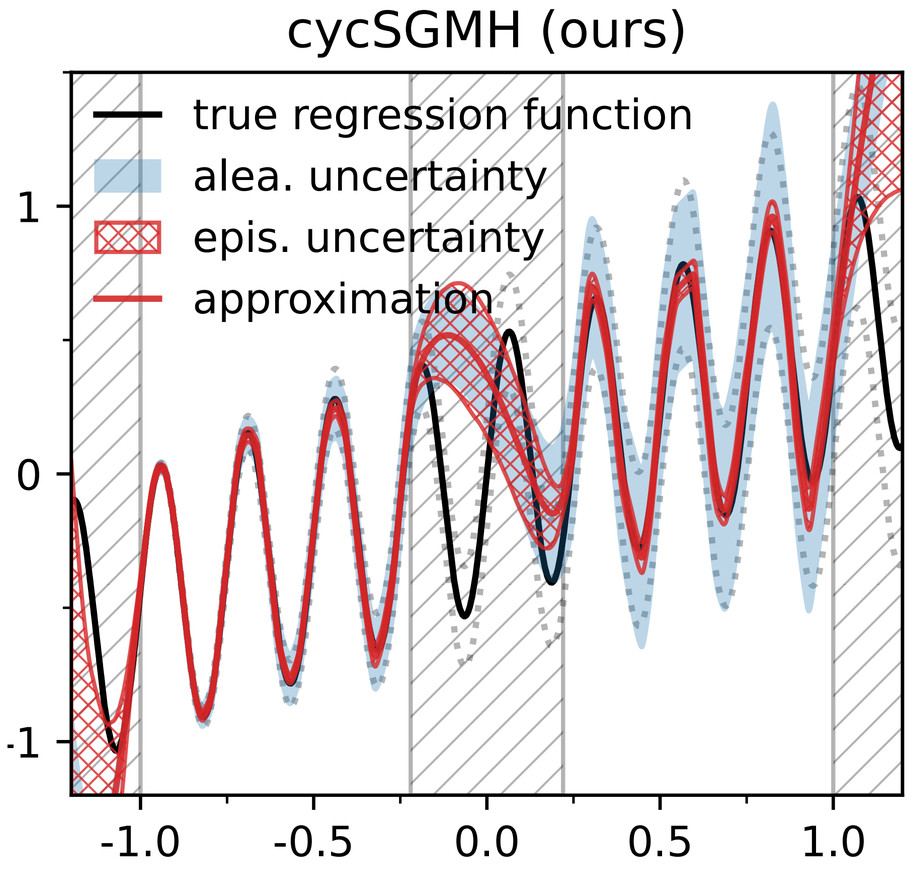
- Allows the use of ADAM in $q(\,\cdot\,|\,\theta^{(k)})$

Pros & Cons:

+ Exact sampling from the posterior
+ Good out-of-distribution detection
+ Fast and stable uncertainty quantification on existing NN

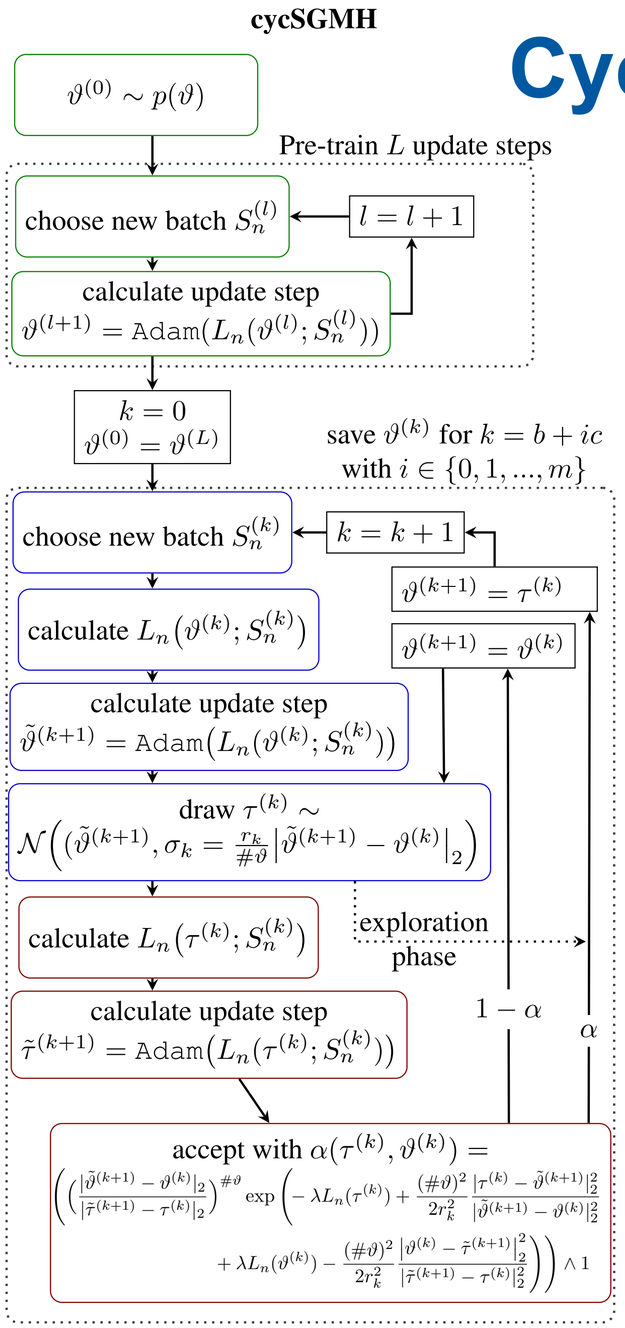
Adaptations:

- Full-loss, other optimizers, etc.



Time per sample (Nvidia Tesla P100):
16.7 ± 1.1 s

Cyclic Stochastic Gradient Metropolis-Hastings

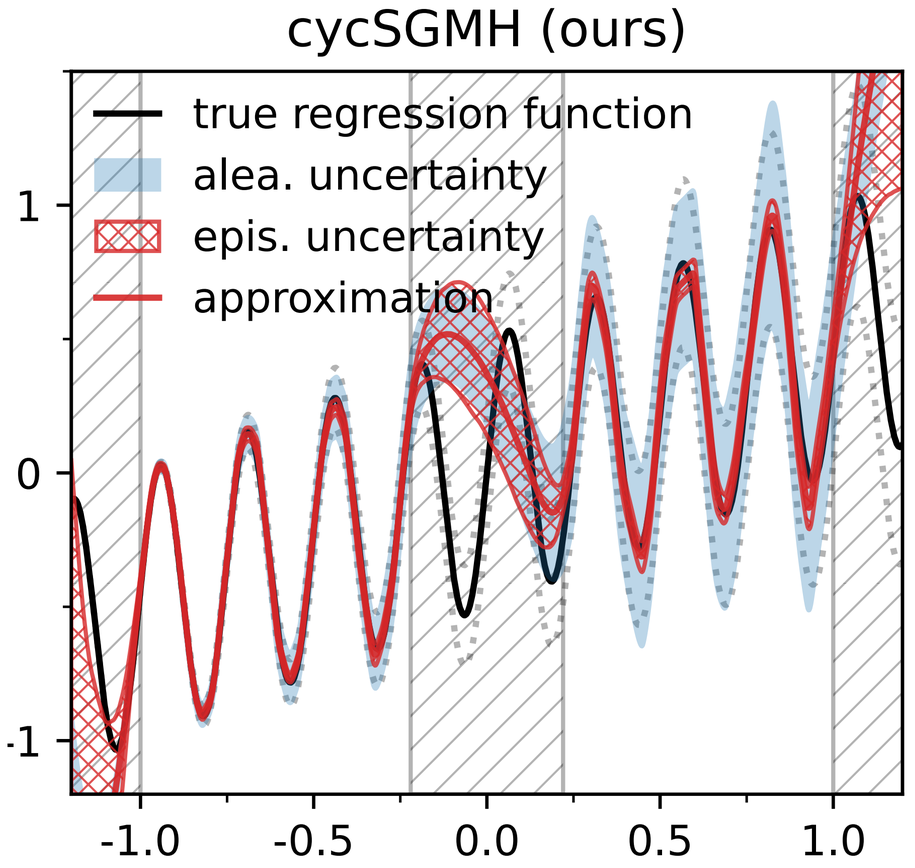


Details:

- Scale noise with update step length and time
- Omit acceptance step at high noise levels

Technical aspects:

- Stochastic acceptance step
- Invariant distribution of the Monte-Carlo chain



Time per sample (Nvidia Tesla P100):
16.7 ± 1.1 s

Cyclic Stochastic Gradient Metropolis-Hastings

Description (2204.12392):

$\tau^{(k+1)} = \tau^{(k)}$ where

$$\left(\theta^{(k)}, \sqrt{\frac{2\eta_k}{\lambda_D}}\right)$$

Faster

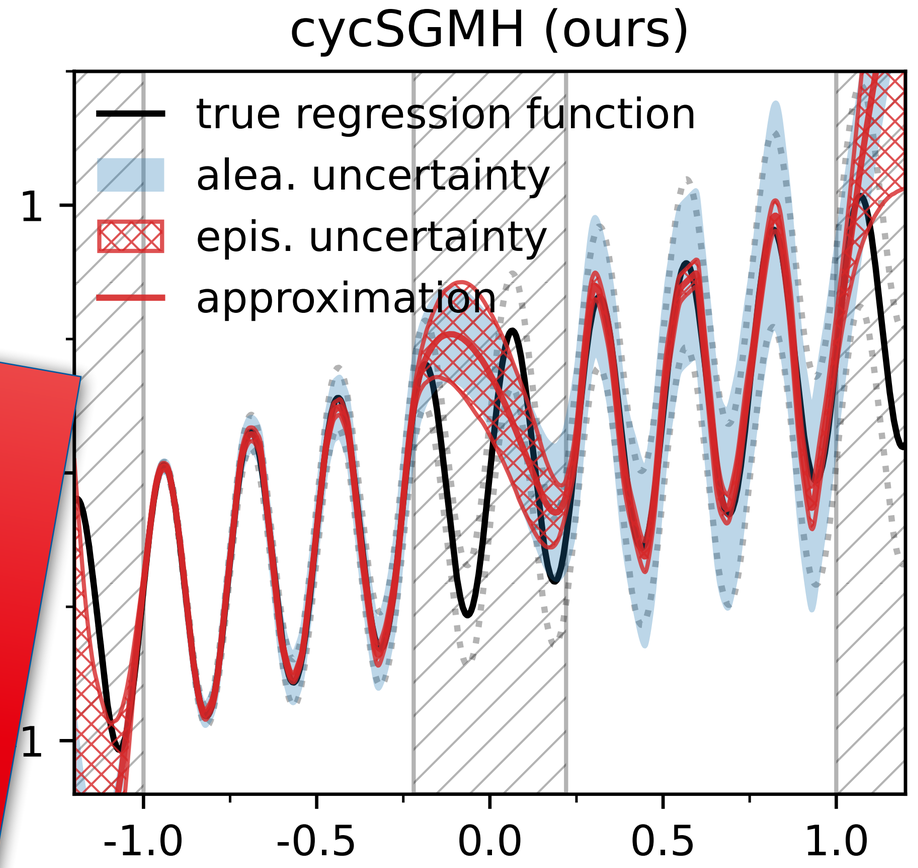
Available on trained networks

Good epistemic uncertainties

+ Good out-of-distri
+ Fast and stable uncertainty quantifica

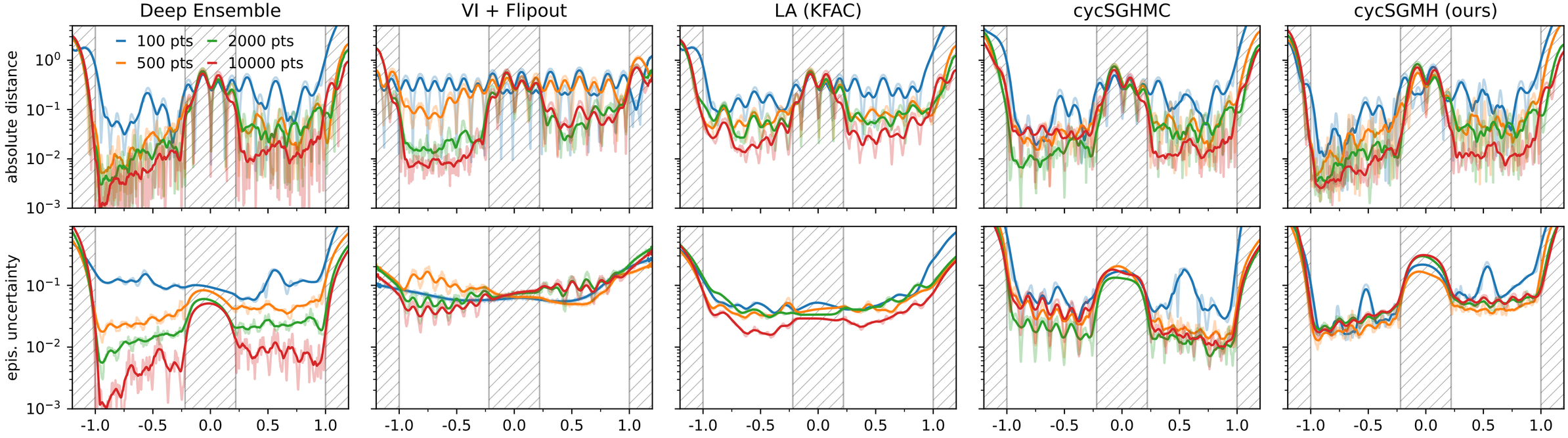
Adaptations:

- Full-loss, other optimizers, etc.

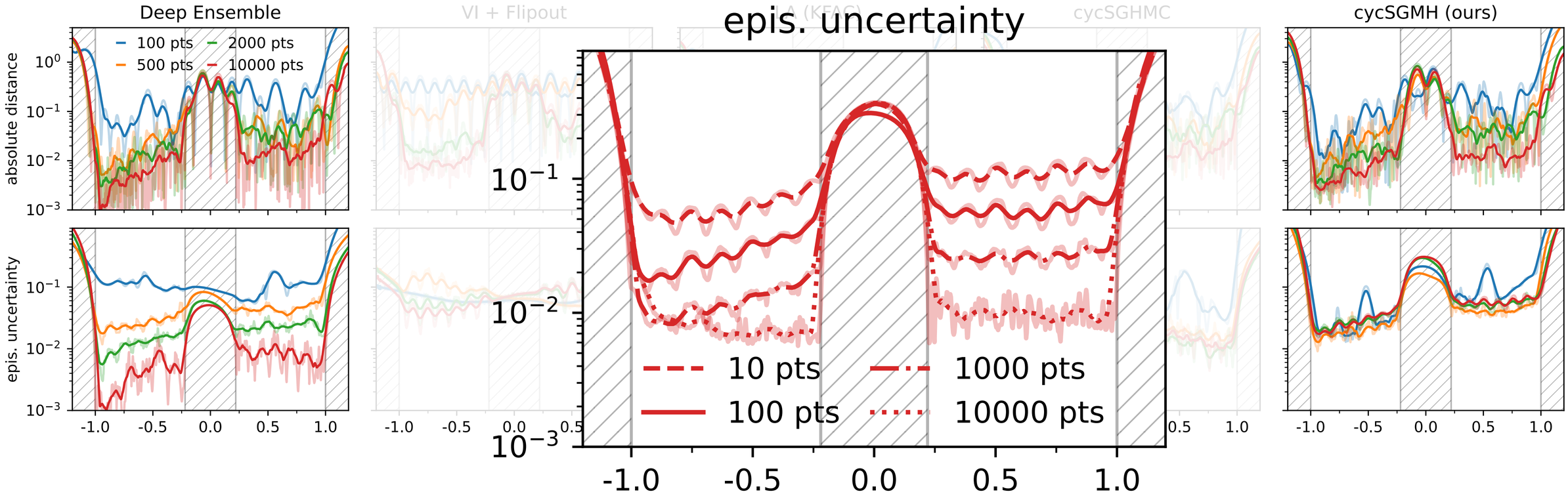


Time per sample (Nvidia Tesla P100):
16.7 ± 1.1 s

Scaling behavior with the training set size



Scaling behavior with the training set size



$\Longrightarrow$ competitive credible sets can be regained by adapting the batchsize

Conclusion

	Deep Ensembles	VI	LA	MCMC
Active Learning	**Yes**	**Yes**	**Yes**	No
Available for trained Network	No, but hyperparameters can be reused	No	**Yes**	**Yes**
Posterior Approximation	Set of maxima	Uncorrelated Gaussian	Correlated Gaussian	**Exact**
Reliable out-of-Distribution (OOD) detection	For large datasets	No	No	**Yes**
Speed - training - posterior sampling	Very slow	Slow Fast	**Fast** **Fast**	Slow
Cost scaling with dataset size	Linear	Linear	Linear	**Constant**

Conclusion

DASHH

	Deep Ensembles	VI	LA	MCMC	csgMH
Active Learning	**Yes**	**Yes**	**Yes**	No	No
Available for trained Network	No, but hyperparameters can be reused	No	**Yes**	**Yes**	**Yes**
Posterior Approximation	Set of maxima	Uncorrelated Gaussian	Correlated Gaussian	**Exact**	**Exact**
Reliable out-of-Distribution (OOD) detection	For large datasets	No	No	**Yes**	**Yes**
Speed - training - posterior sampling	Very slow	Slow Fast	**Fast** **Fast**	Slow	Tolerable
Cost scaling with dataset size	Linear	Linear	Linear	**Constant**	Linear (csgMH with optimal alea. unc.)

Bonus: Further methods for uncertainty estimation

- Dropout MC (1506.02142)
- Contour SGLD (2202.09867)
- Last-layer Implementations (1802.09127)
- Techniques based on out-of distribution detection, often for labeled data

Bonus: Code

List of Code recommondations, used for this study:

- ***Variational Inference:*** BLITZ, bayesian-torch for Flipout
- ***Laplace Approximation:*** Laplace
- ***MCMC methods:*** cycSGHMC, cycSGMH (ours, TBA), Interacting ContourSGLD

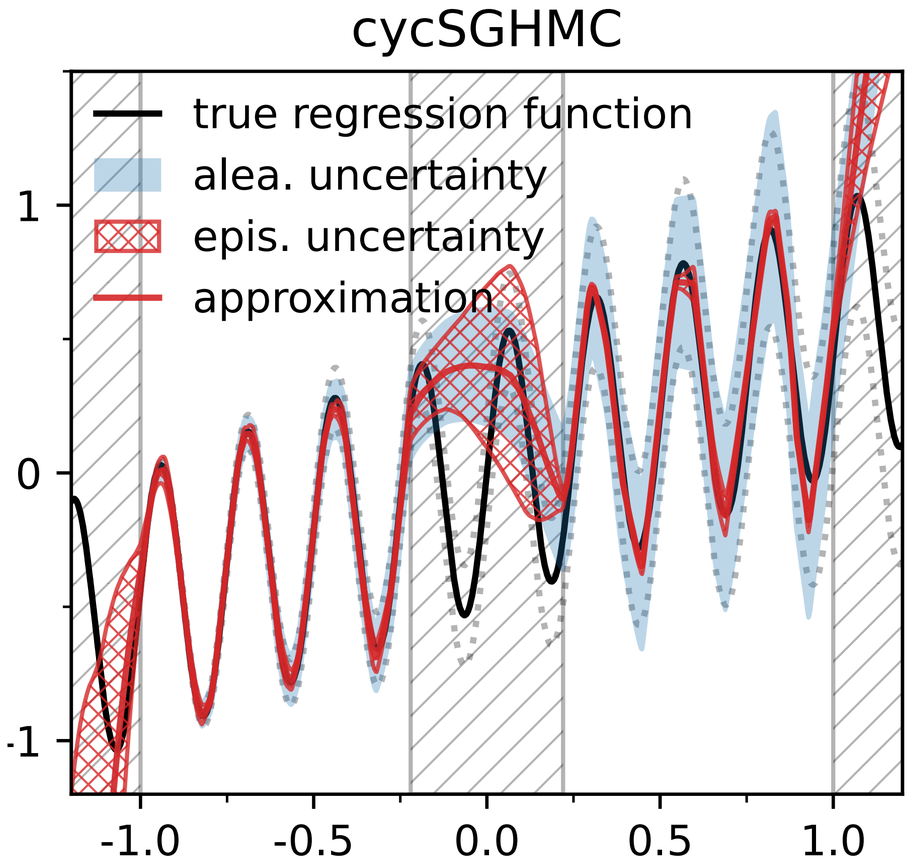
Bonus: Cyclic sgHMC

Description (1902.03932):

- Stochastic Gradient Hamiltonian Monte Carlo (sgHMC) $\theta_k = \theta_{k-1} + v_{k-1}$ with
  $v_k = v_{k-1} - \alpha_k \nabla U\left(\theta_k\right) - \eta v_{k-1} + \sqrt{2}(\eta - \hat{\gamma})\alpha_k \epsilon_k$
- `LD with momentum term´

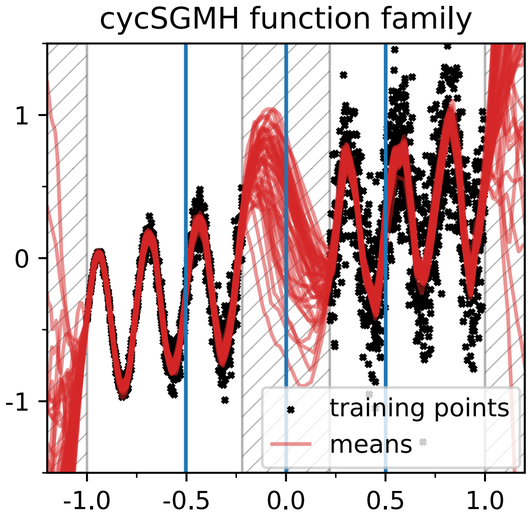
Pros & Cons:

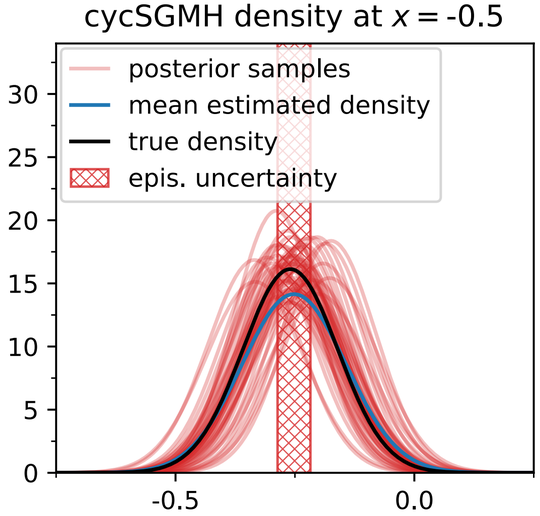
- Better convergence than sgLD

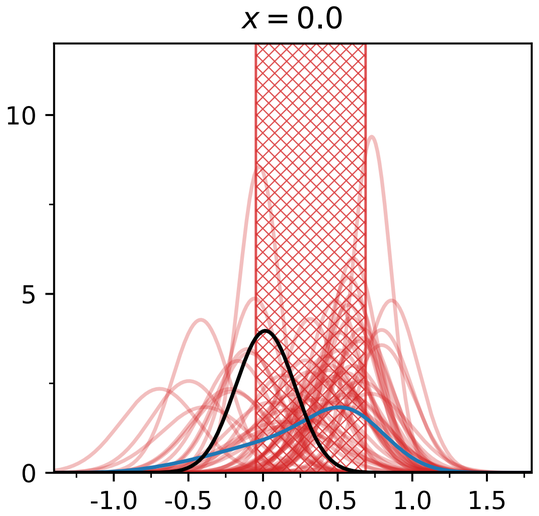


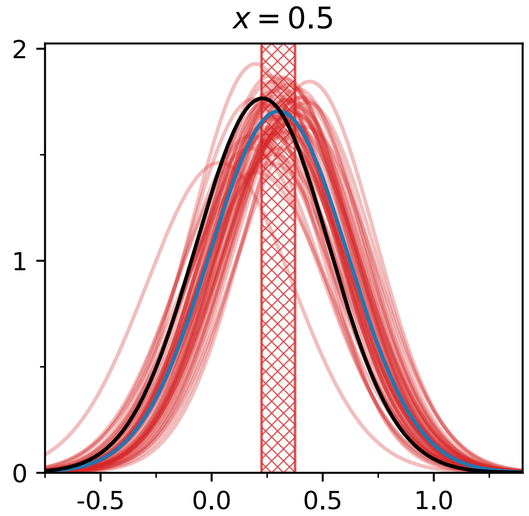
Time per sample (Nvidia Tesla P100):
127 ± 6 s

Bonus: Function clouds









- Sampling from the cycSGMH chain returns one Gaussian fit
- Multiple samples can estimate multimodal likelihood densities or such with tails

Bonus: Full vs stochastic acceptance probability