Case study on snapshot ensembles

Apostolos Psaros

October 20, 2020

1 Main idea

Here is the main problem:

- Different local optima typically correspond to well-performing (low test error) and diverse (making different mistakes) predictions.
- By training with GD-based optimizers we obtain 1 set of optimal weights: no uncertainty and no-diversity.
- Can we modify the standard training procedure in order to 1) average different predictions and reduce generalization error and 2) obtain uncertainty estimates?

Families of approaches:

- Scalable MCMC samplers: Similar updates to SGD with added noise. Learning rate schedule and amount of noise are selected so that the samples correspond to the true posterior.
- Use a preconditioner (like we do in AdaGrad and Adam for different reasons) so that the SGD trajectory becomes a true sampler. Interesting but hard theoretical problem.
- Dropout variational inference. Predictions are not very diverse.
- Bayes by backprop: Variational inference with subsampling and reparametrization trick. Similar criticism as dropout.
- Efficient Laplace approximation: Fit a Gaussian to the obtained optimum by using second order information of the loss.
- Wilson's approaches: Use trajectory parameters and average weights instead of predictions.

Snapshot ensemble:

- Use cyclical learning rate (see Fig. 1).
- Anneal learning rate up to a certain value and then restart.
- Perform many cycles and sample (take a snapshot) just before the restart.
- This is called warm restart.
- Simple modification to standard algorithm and for free accuracy increase and uncertainty estimate.
- Papers: Smith (2017), Huang et al. (2017), Loshchilov and Hutter (2016), and Garipov et al. (2018).

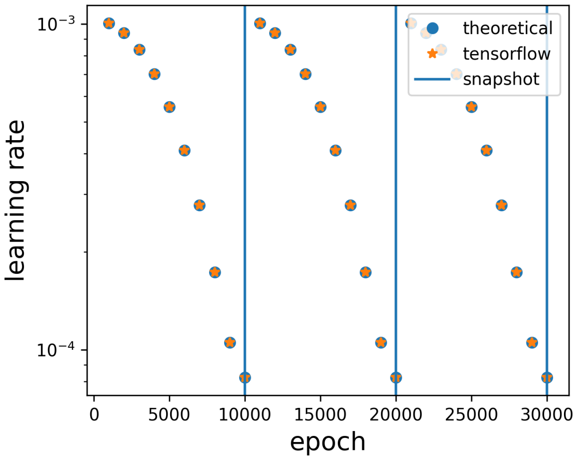


Fig. 1. Learning rate schedule: theoretical (SGDR paper) and as computed by tensorflow (sanity check). Snapshot locations also shown (taken before lr restart).

2 Experiment information

True function:

$$f(x) = \begin{cases} 5 + \sum_{k=1}^{15} sin(kx) & \text{if } -\pi \leq x < 0 \\ cos(10x) & \text{if } 0 \leq x \leq \pi \end{cases} \tag{1}$$

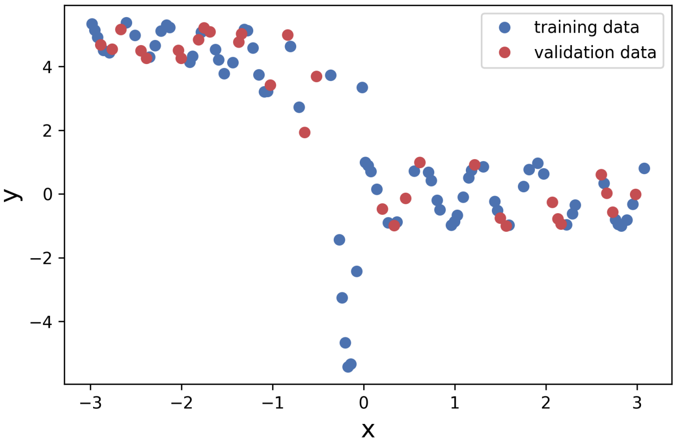


Fig. 2. Training and validation data.

Data:

Training datapoints	70
Validation datapoints	30
Noise std	0
Evaluation datapoints	200

What we want to see:

- Snapshots should give low error predictions and diverse; i.e., with high point-wise standard deviation.
- When ensembled, the final predictions should give a better test error compared to a single model.
- Uncertainty should be higher at the locations of large error. Predictions should agree on over-specified areas and disagree on under-specified ones.

- Snapshot ensembles (1 trajectory) are expected to be less accurate than deep ensembles (many trajectories).
- Snapshots with constant learning rate should be less diverse than snapshots with cosine annealing and restarts.
- But snapshots with cosine annealing are expected to be less diverse than deep ensembles.

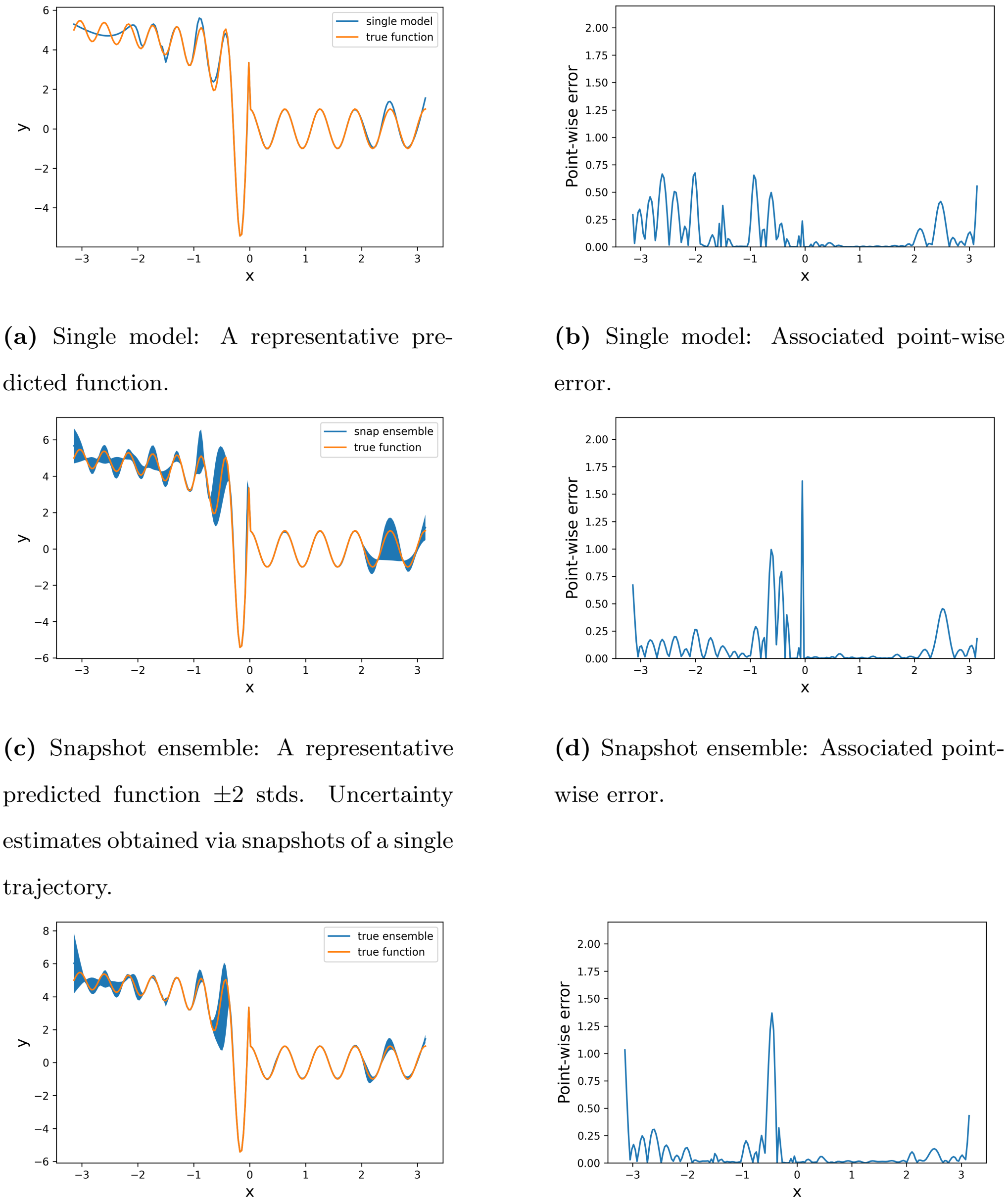
Architecture and hyperparameters:

	Standard lr	Cosine annealing
Depth	5*	
Width	49*	
Activation function	TanH	
Max budget (epochs)	60,000	
Optimizer	Adam	
Initializer	Xavier	
Repetitions	6	
Number of cycles (if not varying)	NA	6
Snapshot step (if not varying)	NA	$10,000$
Number of snapshots (if not varying)	NA	6
Constant lr	4e-4*	NA
Cosine annealing min lr	NA	8e-5**
Cosine annealing max lr	NA	1e-3**

*: obtained via random search

**: obtained via random search with fixed architecture

3 Figures



(a) Single model: A representative predicted function.

(b) Single model: Associated point-wise error.

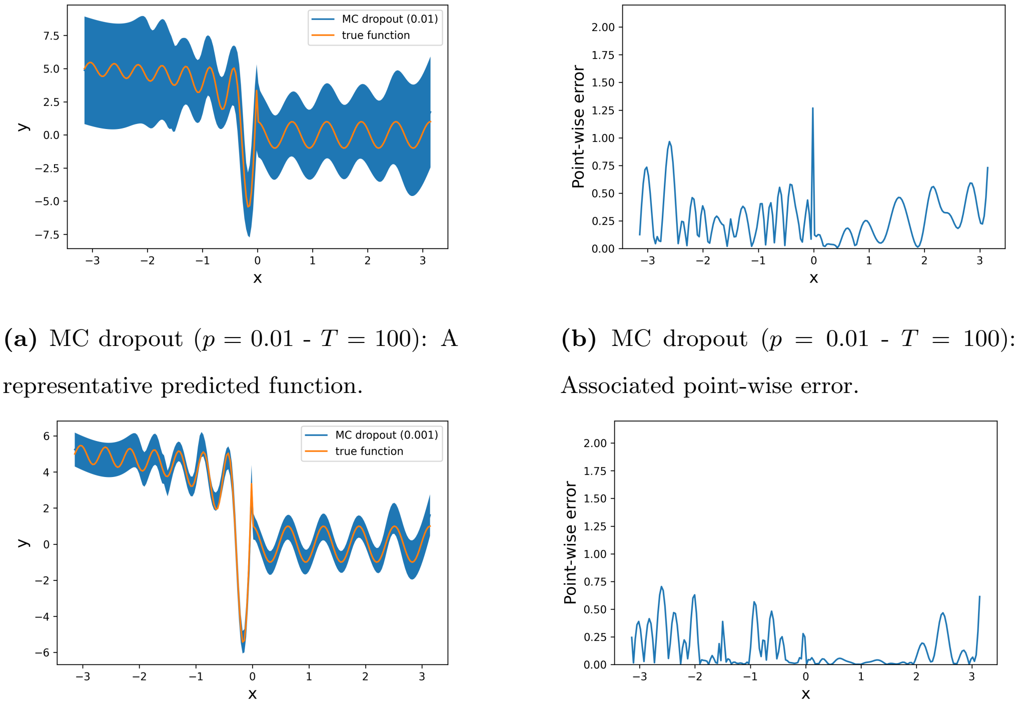
(c) Snapshot ensemble: A representative predicted function ±2 stds. Uncertainty estimates obtained via snapshots of a single trajectory.

(d) Snapshot ensemble: Associated point-wise error.

(e) True ensemble: A representative predicted function ±2 stds. Uncertainty estimates obtained via different trajectories.

(f) True ensemble: Associated point-wise error.

Fig. 3. Representative predicted functions, uncertainty estimates and point-wise errors for a single model, a snapshot ensemble and a true ensemble.



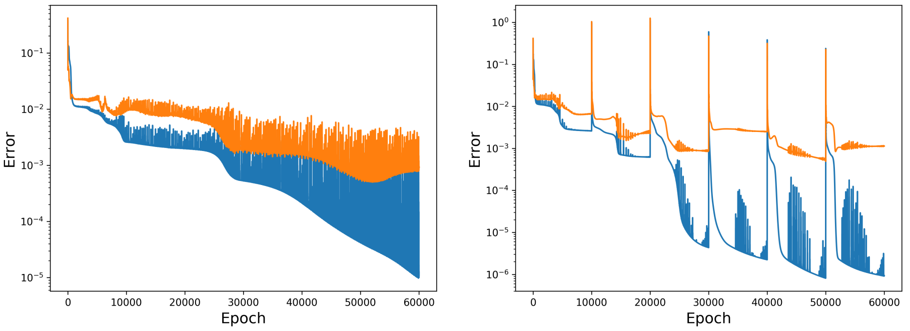
(a) MC dropout ($p = 0.01$ - $T = 100$): A representative predicted function.

(b) MC dropout ($p = 0.01$ - $T = 100$): Associated point-wise error.

(c) MC dropout ($p = 0.001$ - $T = 100$): A representative predicted function.

(d) MC dropout ($p = 0.001$ - $T = 100$): Associated point-wise error.

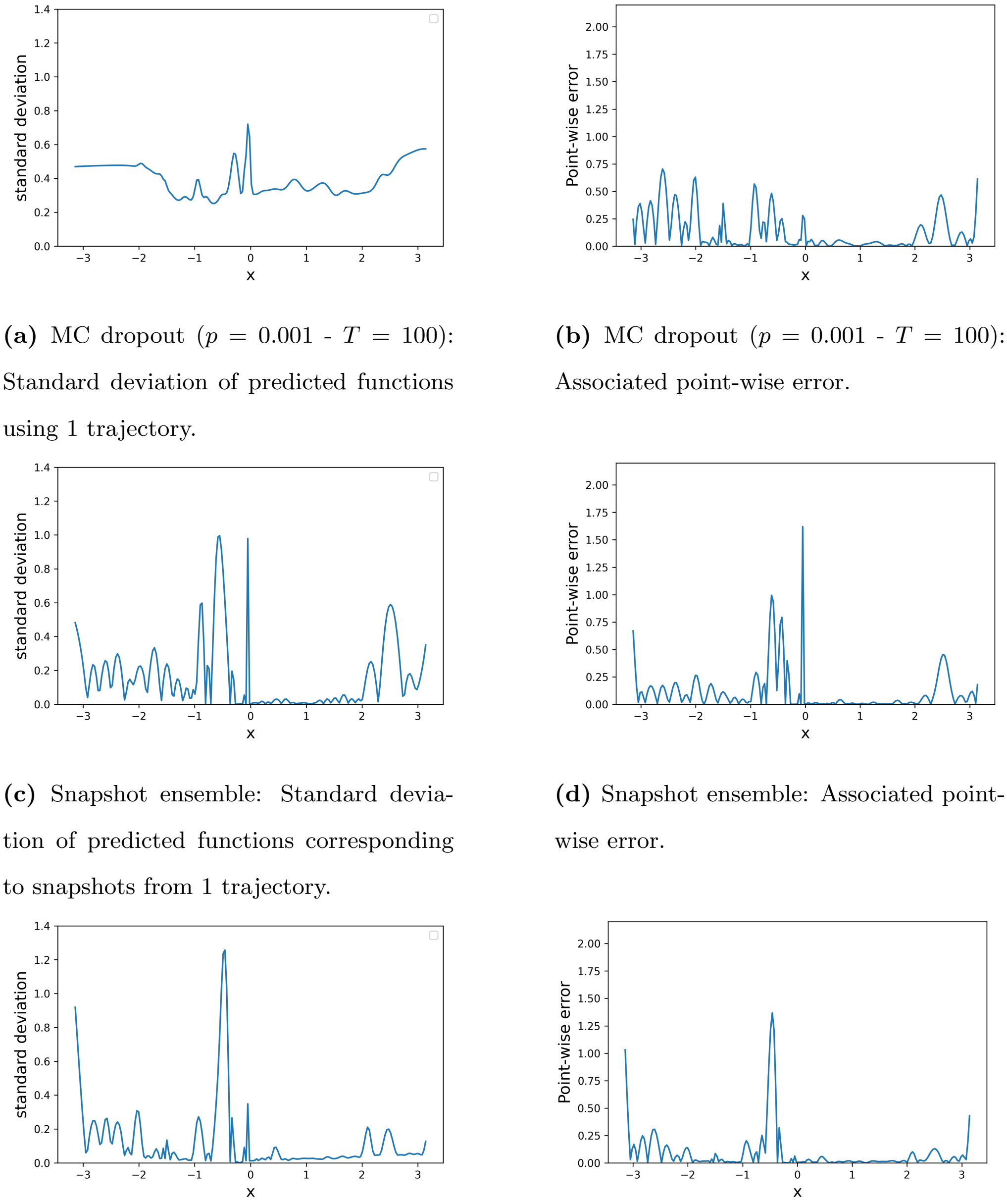
Fig. 4. Representative predicted functions, uncertainty estimates and point-wise errors for MC dropout.



(a) Standard learning rate.

(b) Cosine annealing.

Fig. 5. Representative training/validation loss trajectories. Overfit begins after 60,000 epochs for this specific architecture.



(a) MC dropout ($p = 0.001$ - $T = 100$): Standard deviation of predicted functions using 1 trajectory.

(b) MC dropout ($p = 0.001$ - $T = 100$): Associated point-wise error.

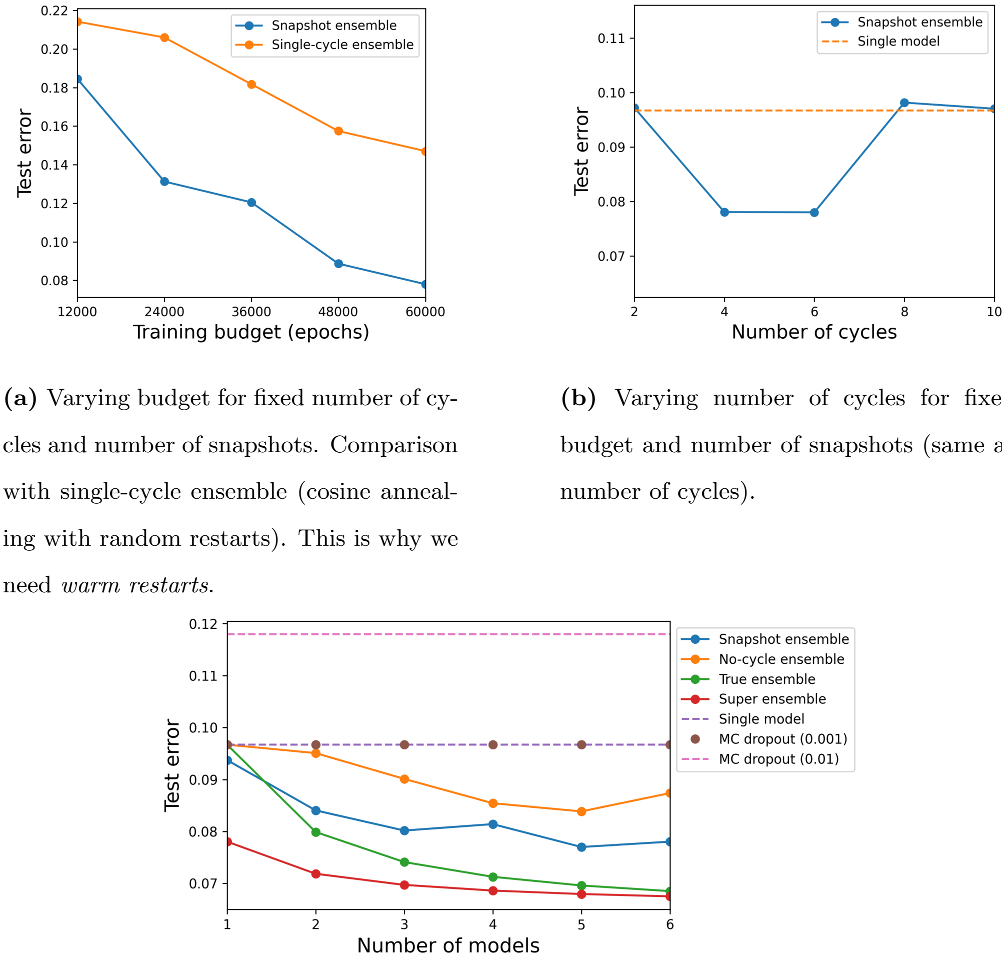
(c) Snapshot ensemble: Standard deviation of predicted functions corresponding to snapshots from 1 trajectory.

(d) Snapshot ensemble: Associated point-wise error.

(e) True ensemble: Standard deviation of predicted functions corresponding to final parameters from different trajectories.

(f) True ensemble: Associated point-wise error.

Fig. 6. Representative uncertainty estimates and point-wise errors for a snapshot ensemble and a true ensemble.

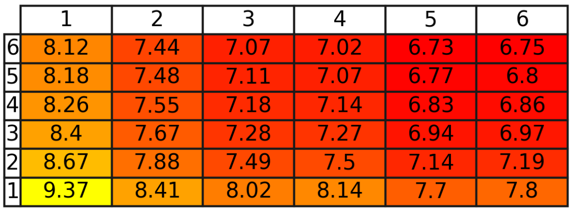


(a) Varying budget for fixed number of cycles and number of snapshots. Comparison with single-cycle ensemble (cosine annealing with random restarts). This is why we need *warm restarts*.

(b) Varying number of cycles for fixed budget and number of snapshots (same as number of cycles).

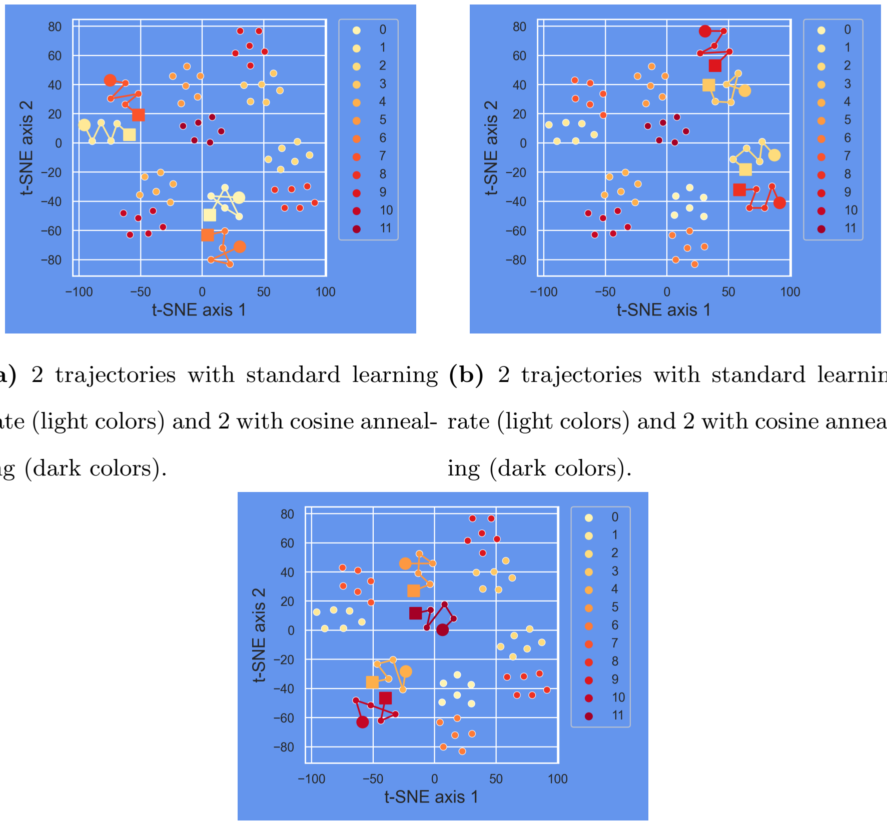
(c) Snapshot ensemble (cosine annealing and varying number of snapshots), no-cycle ensemble (standard lr and varying number of snapshots), true ensemble (varying number of trajectories), and super ensemble (varying number of trajectories and fixed number of snapshots). Single model and MC dropout ($p = 0.001$ and 0.01) are included as well. Budget and number of cycles is fixed for all cases.

Fig. 7. Relative ℓ_2 test error. Comparisons between single models, snapshot ensembles, no-cycle ensembles (standard lr), single-cycle ensembles (cosine annealing with no warm restarts), and true ensembles.



	1	2	3	4	5	6
6	8.12	7.44	7.07	7.02	6.73	6.75
5	8.18	7.48	7.11	7.07	6.77	6.8
4	8.26	7.55	7.18	7.14	6.83	6.86
3	8.4	7.67	7.28	7.27	6.94	6.97
2	8.67	7.88	7.49	7.5	7.14	7.19
1	9.37	8.41	8.02	8.14	7.7	7.8

Fig. 8. Super ensembles: Relative % ℓ_2 test error for varying number of trajectories (rows) and number of snapshots in a trajectory (columns).

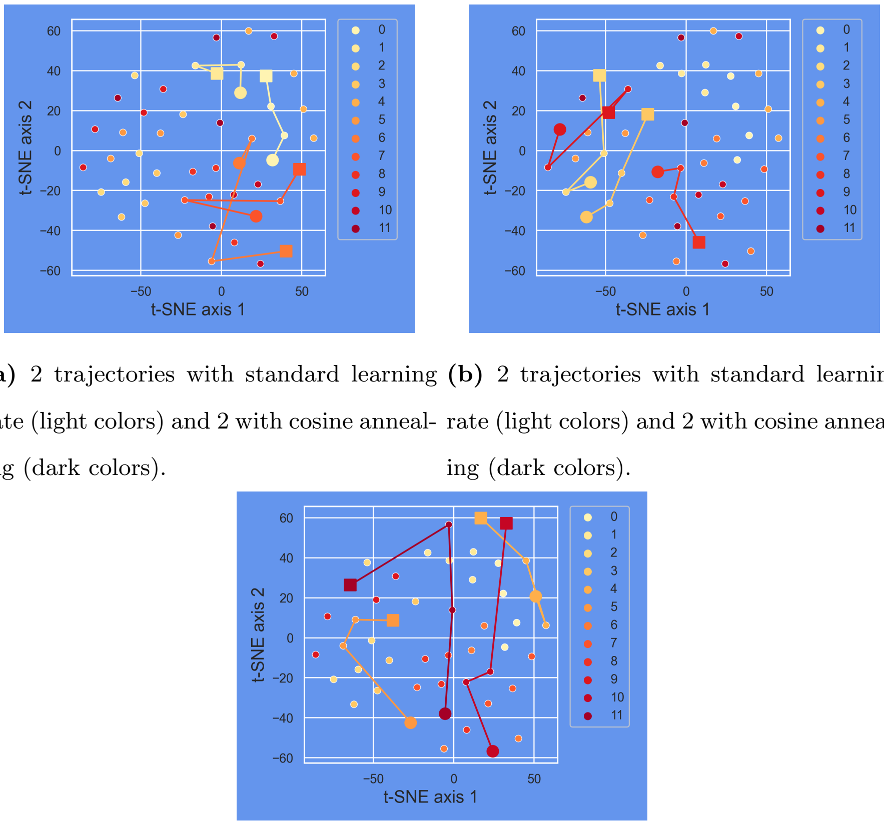


(a) 2 trajectories with standard learning rate (light colors) and 2 with cosine annealing (dark colors).

(b) 2 trajectories with standard learning rate (light colors) and 2 with cosine annealing (dark colors).

(c) 2 trajectories with standard learning rate (light colors) and 2 with cosine annealing (dark colors).

Fig. 9. t-SNE of parameters from different trajectories for both standard learning rate and cosine annealing. Large circles represent final parameters, small circles earlier snapshots and squares the earliest considered snapshots (not necessarily the initializations). Lighter colors represent standard learning rate and darker colors cosine annealing.

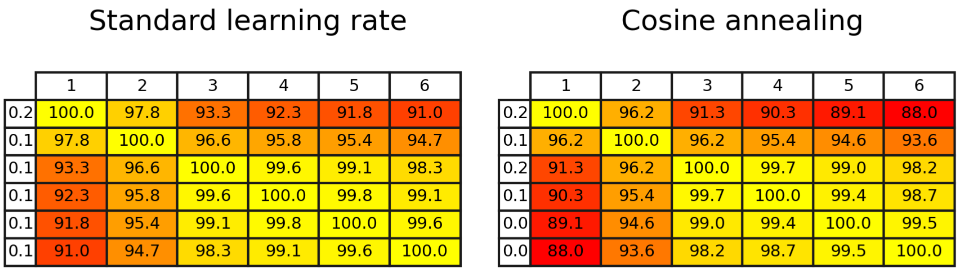


(a) 2 trajectories with standard learning rate (light colors) and 2 with cosine annealing (dark colors).

(b) 2 trajectories with standard learning rate (light colors) and 2 with cosine annealing (dark colors).

(c) 2 trajectories with standard learning rate (light colors) and 2 with cosine annealing (dark colors).

Fig. 10. t-SNE of predicted functions from different trajectories for both standard learning rate and cosine annealing. Large circles represent predictions corresponding to final parameters, small circles to earlier snapshots and squares to the earliest considered snapshots (not necessarily the initializations). Lighter colors represent standard learning rate and darker colors cosine annealing.



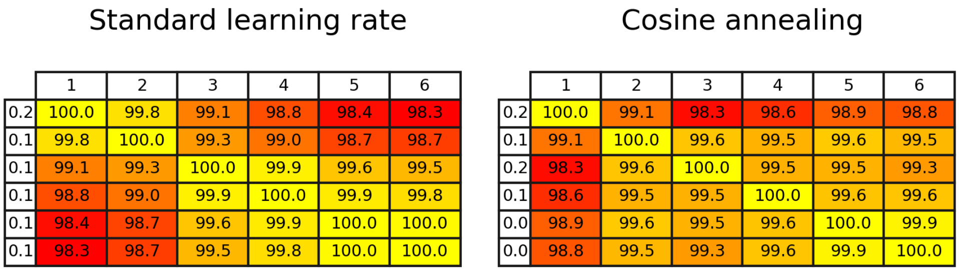
Standard learning rate

	1	2	3	4	5	6
0.2	100.0	97.8	93.3	92.3	91.8	91.0
0.1	97.8	100.0	96.6	95.8	95.4	94.7
0.1	93.3	96.6	100.0	99.6	99.1	98.3
0.1	92.3	95.8	99.6	100.0	99.8	99.1
0.1	91.8	95.4	99.1	99.8	100.0	99.6
0.1	91.0	94.7	98.3	99.1	99.6	100.0

Cosine annealing

	1	2	3	4	5	6
0.2	100.0	96.2	91.3	90.3	89.1	88.0
0.1	96.2	100.0	96.2	95.4	94.6	93.6
0.2	91.3	96.2	100.0	99.7	99.0	98.2
0.1	90.3	95.4	99.7	100.0	99.4	98.7
0.0	89.1	94.6	99.0	99.4	100.0	99.5
0.0	88.0	93.6	98.2	98.7	99.5	100.0

(a) Cosine similarity (%) of parameters corresponding to different snapshots. Far-left column shows the associated relative ℓ_2 test error.



Standard learning rate

	1	2	3	4	5	6
0.2	100.0	99.8	99.1	98.8	98.4	98.3
0.1	99.8	100.0	99.3	99.0	98.7	98.7
0.1	99.1	99.3	100.0	99.9	99.6	99.5
0.1	98.8	99.0	99.9	100.0	99.9	99.8
0.1	98.4	98.7	99.6	99.9	100.0	100.0
0.1	98.3	98.7	99.5	99.8	100.0	100.0

Cosine annealing

	1	2	3	4	5	6
0.2	100.0	99.1	98.3	98.6	98.9	98.8
0.1	99.1	100.0	99.6	99.5	99.6	99.5
0.2	98.3	99.6	100.0	99.5	99.5	99.3
0.1	98.6	99.5	99.5	100.0	99.6	99.6
0.0	98.9	99.6	99.5	99.6	100.0	99.9
0.0	98.8	99.5	99.3	99.6	99.9	100.0

(b) Cosine similarity (%) of predicted functions corresponding to different snapshots. Far-left column shows the associated relative ℓ_2 test error.

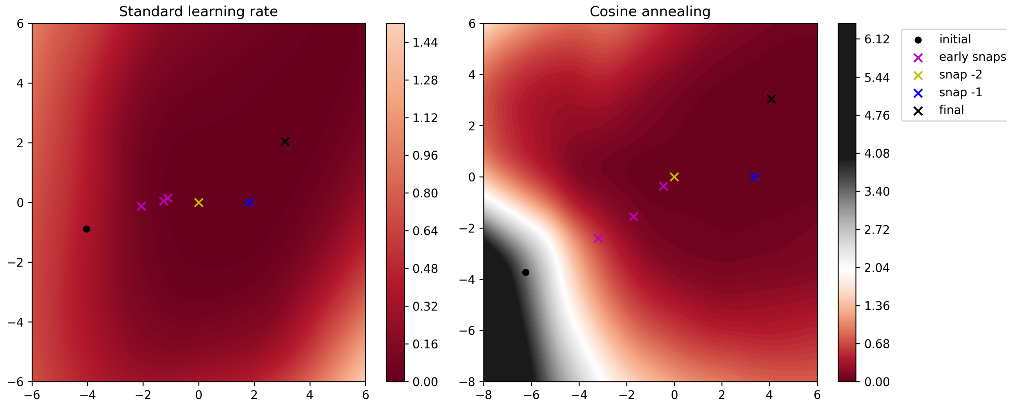


Fig. 12. Training loss on the plane defined by the last 3 snapshots of one trajectory. Earlier snapshots and initial parameters are projected on the plane.

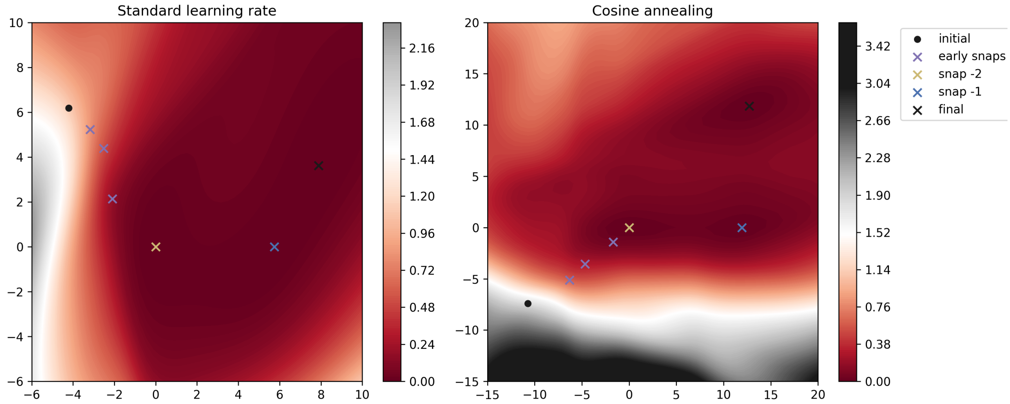


Fig. 13. If we increase the restart learning rate from 1e-3 to 5e-3: Training loss on the plane defined by the last 3 snapshots of one trajectory. Earlier snapshots and initial parameters are projected on the plane.

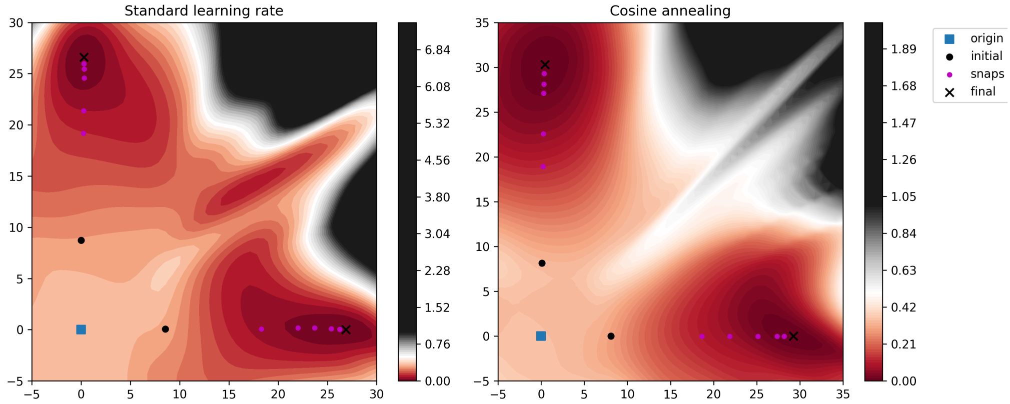


Fig. 14. Training loss on the plane defined by the origin (0) and the final parameters from 2 trajectories with different initializations. Earlier snapshots and initial parameters are projected on the plane.

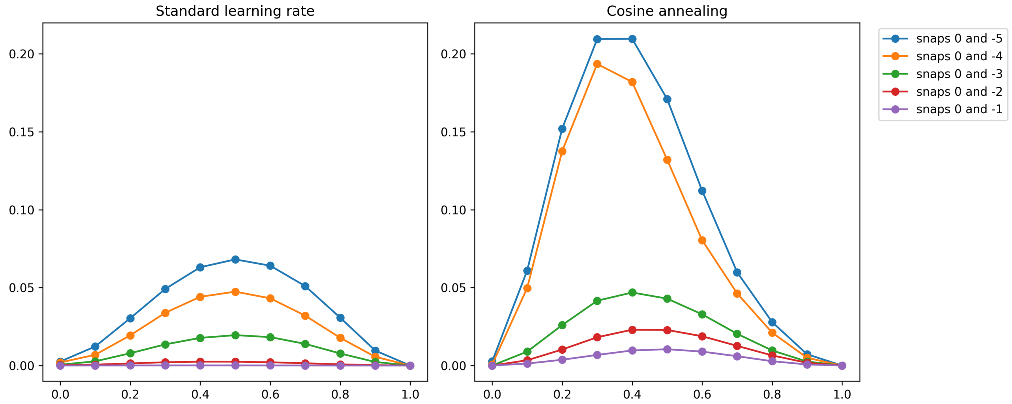


Fig. 15. Training loss on the line connecting the final parameters from a trajectory and earlier snapshots of that trajectory.

4 What's next for function approximation

Make sure the results hold for:

- "Easier" functions that are well approximated by a single model.
- Different number of datapoints.
- Added noise to the data.

Also try to evaluate the quality of the obtained uncertainty.

5 Towards ePINNs?

Incorporate to PINNs:

- As it is for better accuracy and for-free uncertainty.
- Try to devise a method for active learning while training: resample at high-uncertainty areas.

What I think would be publishable:

- Have 3 techniques: 1 for-free (like snapshot ensembles), 1 with intermediate cost (like scalable MCMC) and a more expensive one (like replica MCMC).

- Compare and discuss applicability for PINNs for big data and large dimensions.

References

Garipov, T., Izmailov, P., Podoprikhin, D., Vetrov, D. P., and Wilson, A. G. (2018). "Loss Surfaces, Mode Connectivity, and Fast Ensembling of Dnns". *Advances in Neural Information Processing Systems*, pp. 8789–8798.

Huang, G. et al. (2017). "Snapshot Ensembles: Train 1, Get M for Free". *arXiv:1704.00109 [cs]*. arXiv: 1704.00109 [cs].

Loshchilov, I. and Hutter, F. (2016). "Sgdr: Stochastic Gradient Descent with Warm Restarts". *arXiv preprint arXiv:1608.03983*. arXiv: 1608.03983.

Smith, L. N. (2017). "Cyclical Learning Rates for Training Neural Networks". *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, pp. 464–472.