Introduction to Sharpness Aware Minimization

Csongor Horváth

November 29, 2024

Abstract

A literature overview on the topics of Sharpness Aware Minimization for Deep Learning.

1 Sharpness measures and generalization

Better generalization has been related to local geometry of the loss landscape in several previous works [Hochreiter and Schmidhuber 1997, Keskar et al. 2017]. Some motivation is the difference between the loss function $L_{\mathcal{D}}$ and $L_{\mathbf{S}}$ would indicate that if the difference is reasonably small, which we want to assume, as we expect our data represents the distribution. Other motivations can arise from minimal description length or the Bayesian view.
Based on the work of Keskar et al, the study of this area got popular again at google they showed that multiple flatness measure have steady positive correlation with other parameters not just generalization [Jiang et al. 2019], such as total loss, weight decay, optimizer, learning rate, batch size, depth, etc.

In [Dinh et al. 2017] it was noted that due to the possible re-parametrization of NNs. One such example is scaling 2 FC layer with ReLu activation up and down by α. They state that such re-parametrization doesn't change the nature of the function, but can influence most traditional sharpness measure. This created a search for re-parametrization invariant sharpness measure and modification of the training algorithms such that they will be re-parametrization invariant.

2 Sharpness aware minimization - original paper

In this section we follow the work of [Foret et al. 2021] and motivate and introduce SAM in the same way.

Let denote the n samples with $\mathbf{S}$ and the underlying distribution with $\mathcal{D}$ and the loss $L_{\mathcal{D}}(w) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[l(w,x,y)]$ and $L_{\mathbf{S}} = \sum_{i=1}^{n} l(w,x_i,y_i)$.

The goal of training a model is to find w with low $L_{\mathcal{D}}$ loss, but it is usually achieved through minimizing $L_{\mathbf{S}}$. It would be nice to relate the two.

Theorem 2.1. *Assume $L_{\mathcal{D}}(w) \leq \mathbb{E}_{\epsilon_i\sim\mathcal{N}(0,\rho)}[L_{\mathcal{D}}(w+\epsilon)]$, so that adding Gaussian noise to the parameters not decrease the error. Then:*

$$\begin{aligned} L_{\mathcal{D}}(w) &\leq \mathbb{E}_{\epsilon_i\sim\mathcal{N}(0,\rho)}[L_{\mathcal{D}}(w+\epsilon)] \\ &\leq \mathbb{E}_{\epsilon_i\sim\mathcal{N}(0,\rho)}[L_{\mathbf{S}}(w+\epsilon)] + \frac{\frac{1}{4}(1+\frac{\|w\|_2^2}{k\rho^2}) + \frac{1}{4} + \log\frac{n}{\delta} + 2\log(6n+3k)}{n-1} \\ &\leq (1-1/\sqrt{n})max_{\|\epsilon\|_2\leq\rho}L_S(w+\epsilon) + h(\frac{\|w\|_2^2}{\rho^2}) \end{aligned}$$

So the important conclusion is that:

$$L_{\mathcal{D}}(w) \leq max_{\|\epsilon\|_2 \leq \rho} L_S(w+\epsilon) + h(\frac{\|w\|_2^2}{\rho^2})$$

here $h : \mathbb{R}_+ \mapsto \mathbb{R}_+$ *strictly increasing.*

This expression gives us an upper bound for the true loss under some realistic assumption, though the gap term can be large in the given formulation, but decreasing if $n \to \infty$. Also as only the $\|w\|$ is what typically changing during a training process, we can add weight decay to account for this term. If we minimize the r.h.s. than we are bounding the true loss we are interested in.

Definition 2.2. *Sharpness-Aware-Minimization*:

$$min_w \max_{\|\epsilon\|_p \leq \rho} L_{\mathbf{S}}(w+\epsilon) \; + \lambda \|w\|_2^2$$

where $L_{\mathbf{S}}^{SAM}(w) = \max_{\|\epsilon\|_p \leq \rho} L_{\mathbf{S}}(w+\epsilon)$

2.1 Approximating the SAM term

In order to use first order optimizer methods (e.g. SGD, Adam, etc) we need to approximate the gradient of the SAM loss in a parameter point w. This can be done by approximating the solution to the inner optimization problem first, from first order approximation we get

$$\epsilon^*(w) = argmax_{\|\epsilon\| \leq \rho} L_{\mathbf{S}}(w+\epsilon) \approx argmax\, \epsilon^T \nabla_w L_{\mathbf{S}}(w) \approx \rho \frac{sign(\nabla_w L_{\mathbf{S}}(w)) |\nabla_w L\_bS(w)|^{q-1}}{(\|\nabla_w L_{\mathbf{S}}(w)\|_q^q)^{\frac{1}{p}}}$$

$$\nabla L_{\mathbf{S}}^{SAM} \approx \nabla_w L_{\mathbf{S}}(w)|_{w+\hat{\epsilon}(w)}$$

Input: Training set $\mathcal{S} \triangleq \cup_{i=1}^{n} \{(x_i, y_i)\}$, Loss function $l : \mathcal{W} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+$, Batch size b, Step size $\eta > 0$, Neighborhood size $\rho > 0$.
Output: Model trained with SAM
Initialize weights w_0, $t = 0$;
while *not converged* **do**
  Sample batch $\mathcal{B} = \{(x_1, y_1), ...(x_b, y_b)\}$;
  Compute gradient $\nabla_w L_{\mathcal{B}}(w)$ of the batch's training loss;
  Compute $\hat{\epsilon}(w)$ per equation 2;
  Compute gradient approximation for the SAM objective (equation 3): $g = \nabla_w L_{\mathcal{B}}(w)|_{w+\hat{\epsilon}(w)}$;
  Update weights: $w_{t+1} = w_t - \eta g$;
  $t = t + 1$;
end
return w_t

Algorithm 1: SAM algorithm

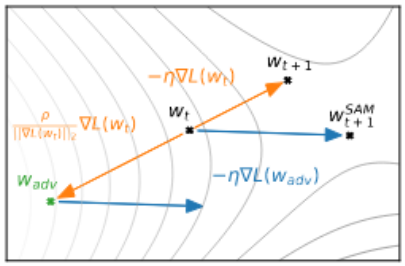


Figure 2: Schematic of the SAM parameter update.

Figure 1: Algorithm and visualization from the original paper [Foret et al. 2021]

2.2 Claims of the original paper

The below results are coming from the paper [Foret et al. 2021].

It is worth noting that the introduced SAM model has one extra hyper parameter. It has been showed that in general $\rho = 0.05$ is a reasonable choice in practice. Note that for some

application this might not hold. Additionally the computational cost of SAM is around twice as for SGD due to the inner calculation of $\hat{\epsilon}$, which require the calculation of one extra gradient and the update of the weights.

The main results showed that using SAM the authors achieved smaller loss levels for multiple settings:

		CIFAR-10		CIFAR-100	
Model	Augmentation	SAM	SGD	SAM	SGD
WRN-28-10 (200 epochs)	Basic	**2.7**$_{\pm 0.1}$	3.5$_{\pm 0.1}$	**16.5**$_{\pm 0.2}$	18.8$_{\pm 0.2}$
WRN-28-10 (200 epochs)	Cutout	**2.3**$_{\pm 0.1}$	2.6$_{\pm 0.1}$	**14.9**$_{\pm 0.2}$	16.9$_{\pm 0.1}$
WRN-28-10 (200 epochs)	AA	**2.1**$_{\pm <0.1}$	2.3$_{\pm 0.1}$	**13.6**$_{\pm 0.2}$	15.8$_{\pm 0.2}$
WRN-28-10 (1800 epochs)	Basic	**2.4**$_{\pm 0.1}$	3.5$_{\pm 0.1}$	**16.3**$_{\pm 0.2}$	19.1$_{\pm 0.1}$
WRN-28-10 (1800 epochs)	Cutout	**2.1**$_{\pm 0.1}$	2.7$_{\pm 0.1}$	**14.0**$_{\pm 0.1}$	17.4$_{\pm 0.1}$
WRN-28-10 (1800 epochs)	AA	**1.6**$_{\pm 0.1}$	2.2$_{\pm <0.1}$	**12.8**$_{\pm 0.2}$	16.1$_{\pm 0.2}$
Shake-Shake (26 2x96d)	Basic	**2.3**$_{\pm <0.1}$	2.7$_{\pm 0.1}$	**15.1**$_{\pm 0.1}$	17.0$_{\pm 0.1}$
Shake-Shake (26 2x96d)	Cutout	**2.0**$_{\pm <0.1}$	2.3$_{\pm 0.1}$	**14.2**$_{\pm 0.2}$	15.7$_{\pm 0.2}$
Shake-Shake (26 2x96d)	AA	**1.6**$_{\pm <0.1}$	1.9$_{\pm 0.1}$	**12.8**$_{\pm 0.1}$	14.1$_{\pm 0.2}$
PyramidNet	Basic	**2.7**$_{\pm 0.1}$	4.0$_{\pm 0.1}$	**14.6**$_{\pm 0.4}$	19.7$_{\pm 0.3}$
PyramidNet	Cutout	**1.9**$_{\pm 0.1}$	2.5$_{\pm 0.1}$	**12.6**$_{\pm 0.2}$	16.4$_{\pm 0.1}$
PyramidNet	AA	**1.6**$_{\pm 0.1}$	1.9$_{\pm 0.1}$	**11.6**$_{\pm 0.1}$	14.6$_{\pm 0.1}$
PyramidNet+ShakeDrop	Basic	**2.1**$_{\pm 0.1}$	2.5$_{\pm 0.1}$	**13.3**$_{\pm 0.2}$	14.5$_{\pm 0.1}$
PyramidNet+ShakeDrop	Cutout	**1.6**$_{\pm <0.1}$	1.9$_{\pm 0.1}$	**11.3**$_{\pm 0.1}$	11.8$_{\pm 0.2}$
PyramidNet+ShakeDrop	AA	**1.4**$_{\pm <0.1}$	1.6$_{\pm <0.1}$	**10.3**$_{\pm 0.1}$	10.6$_{\pm 0.1}$

Table 1: Results for SAM on state-of-the-art models on CIFAR-{10, 100} (WRN = WideResNet; AA = AutoAugment; SGD is the standard non-SAM procedure used to train these models).

Model	Epoch	SAM		Standard Training (No SAM)	
		Top-1	Top-5	Top-1	Top-5
ResNet-50	100	**22.5**$_{\pm 0.1}$	6.28$_{\pm 0.08}$	22.9$_{\pm 0.1}$	6.62$_{\pm 0.11}$
	200	**21.4**$_{\pm 0.1}$	5.82$_{\pm 0.03}$	22.3$_{\pm 0.1}$	6.37$_{\pm 0.04}$
	400	**20.9**$_{\pm 0.1}$	5.51$_{\pm 0.03}$	22.3$_{\pm 0.1}$	6.40$_{\pm 0.06}$
ResNet-101	100	**20.2**$_{\pm 0.1}$	5.12$_{\pm 0.03}$	21.2$_{\pm 0.1}$	5.66$_{\pm 0.05}$
	200	**19.4**$_{\pm 0.1}$	4.76$_{\pm 0.03}$	20.9$_{\pm 0.1}$	5.66$_{\pm 0.04}$
	400	**19.0**$_{\pm <0.01}$	4.65$_{\pm 0.05}$	22.3$_{\pm 0.1}$	6.41$_{\pm 0.06}$
ResNet-152	100	**19.2**$_{\pm <0.01}$	4.69$_{\pm 0.04}$	20.4$_{\pm <0.0}$	5.39$_{\pm 0.06}$
	200	**18.5**$_{\pm 0.1}$	4.37$_{\pm 0.03}$	20.3$_{\pm 0.2}$	5.39$_{\pm 0.07}$
	400	**18.4**$_{\pm <0.01}$	4.35$_{\pm 0.04}$	20.9$_{\pm <0.0}$	5.84$_{\pm 0.07}$

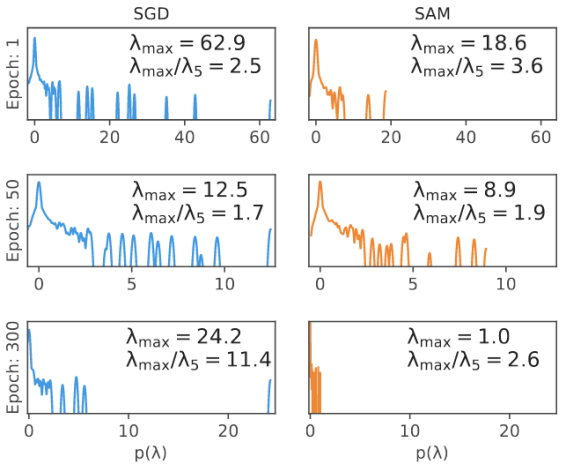
Table 2: Test error rates for ResNets trained on ImageNet, with and without SAM.

As the claim is flatness can lead to more robust results, it has also been showed that SAM helps with artificial noise resilience

And the final interesting result in the original paper was about the eigenvalues of the achieved parameters. Namely, they observed the spectra of the Hessian to motivate that the resulting parameters are indeed more flat.

Method	Noise rate (%)			
	20	40	60	80
Sanchez et al. (2019)	94.0	92.8	90.3	74.1
Zhang & Sabuncu (2018)	89.7	87.6	82.7	67.9
Lee et al. (2019)	87.1	81.8	75.4	-
Chen et al. (2019)	89.7	-	-	52.3
Huang et al. (2019)	92.6	90.3	43.4	-
MentorNet (2017)	92.0	91.2	74.2	60.0
Mixup (2017)	94.0	91.5	86.8	76.9
MentorMix (2019)	**95.6**	**94.2**	91.3	**81.0**
SGD	84.8	68.8	48.2	26.2
Mixup	93.0	90.0	83.8	70.2
Bootstrap + Mixup	93.3	92.0	87.6	72.0
SAM	95.1	93.4	90.5	77.9
Bootstrap + SAM	95.4	**94.2**	**91.8**	79.9

Table 4: Test accuracy on the clean test set for models trained on CIFAR-10 with noisy labels. Lower block is our implementation, upper block gives scores from the literature, per Jiang et al. (2019).

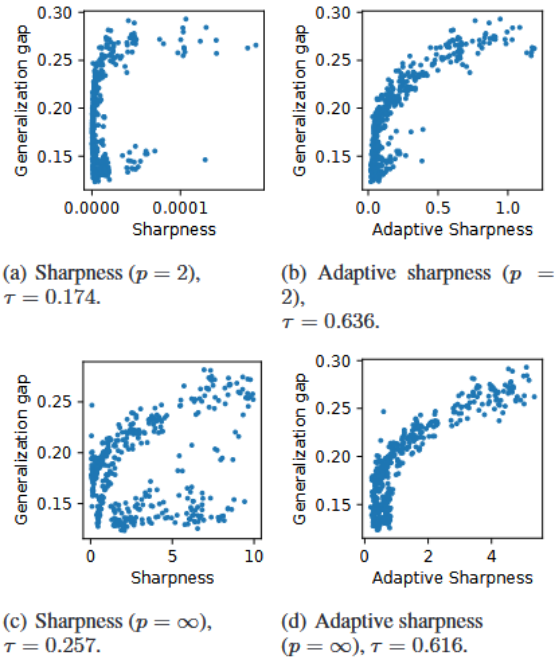


(a) The spectrum of the Hessian with and without SAM

3 Further works

3.1 The re-scaling invariance problem

It has been studied that rescaling can influence the choice of $\hat{\epsilon}$, therefore [Kwon et al. (2021)] introduced a re-parametrization invariant version of the $\hat{\epsilon}$ update, where they normalize by the weight norm, which as a measure leads to better correlations with generalization and other variables in the training process.



(a) Sharpness ($p = 2$), $\tau = 0.174$.

(b) Adaptive sharpness ($p = 2$), $\tau = 0.636$.

(c) Sharpness ($p = \infty$), $\tau = 0.257$.

(d) Adaptive sharpness ($p = \infty$), $\tau = 0.616$.

(a) adaptive sharpness and generalization

Table 2. Maximum test accuracies for SGD, SAM and ASAM on CIFAR-10 dataset.

Model	SGD	SAM	ASAM
DenseNet-121	$91.00_{\pm 0.13}$	$92.00_{\pm 0.17}$	$\mathbf{93.33}_{\pm 0.04}$
ResNet-20	$93.18_{\pm 0.21}$	$93.56_{\pm 0.15}$	$\mathbf{93.82}_{\pm 0.17}$
ResNet-56	$94.58_{\pm 0.20}$	$95.18_{\pm 0.15}$	$\mathbf{95.42}_{\pm 0.16}$
VGG19-BN*	$93.87_{\pm 0.09}$	94.60	$\mathbf{95.07}_{\pm 0.05}$
ResNeXt29-32x4d	$95.84_{\pm 0.24}$	$96.34_{\pm 0.30}$	$\mathbf{96.80}_{\pm 0.06}$
WRN-28-2	$95.13_{\pm 0.16}$	$95.74_{\pm 0.08}$	$\mathbf{95.94}_{\pm 0.05}$
WRN-28-10	$96.34_{\pm 0.12}$	$96.98_{\pm 0.04}$	$\mathbf{97.28}_{\pm 0.07}$
PyramidNet-272†	$98.44_{\pm 0.08}$	$98.55_{\pm 0.05}$	$\mathbf{98.68}_{\pm 0.08}$

(b) ASAM results on CIFAR 10

3.2 Computational and efficiency improvements

The main problem known to SAM is it's computational overhead. It is around twice as expensive to iterate through the same amount of data as SGD / Adam.

It made it a rational research objective for many to try and improve it's computational speed. The simplest idea, which appeared in many work is to reduce the time spent on $\hat{\epsilon}$ computation. The most simplistically by only performing the SAM step instead of the SGD step at certain times. The used gradient can be decided by a random variable [Zhao, Zhang, and Hu 2023] which can perform similarly or even better then SAM with up to 50% of computational loss regained compared to SGD. It has also been studied if we only compute every nth step as SAM, which was inefficient, but when in the steps between the gradient was updated based on the difference between the SAM and simple gradient at the time of nth step SAM calculations the method performed similarly as SAM, but the overhead in time were reduced by around 80% [Liu et al. (2022)]. There were also efforts to choose $\hat{\epsilon}$ from a random distribution as in [[Ujváry et al. 2022]] it was shown that this can be done by changing only the perturbation type of the method, though it slightly reduced accuracy, but it is fast computationally. It has been studied to only compute part of $\hat{\epsilon}$ to make the training faster, it has been done by random mask ESAM [Du, Yan, et al. (2022)] which outperforms SAM in precision and speed with other modification. Other masks have been studied (e.g. Fisher information mask related to the Hessian and Fisher information) Sparse SAM [Mi et al. 2022] these methods performs worse then SAM though results in slightly better speedup (60%).

3.3 Current state of the art for SAM like methods

There is multiple motivation which led to similar algorithms trying to improve SAM. Here I provide motivation for the basics on which these works follow.

The earliest result GSAM [Zhuang et al. 2022] is based on the realization that SAM objective

$$f_p = max_{\|\epsilon\| \leq \rho} f(w + \epsilon)$$

might not be sufficient as this only describes flatness well if $f(w) = 0$.

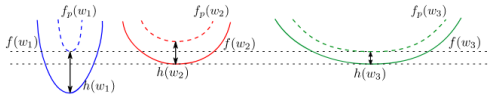


Figure 1: Consider original loss f (solid line), perturbed loss $f_p \triangleq \max_{\|\delta\| \leq \rho} f(w+\delta)$ (dashed line), and surrogate gap $h(w) \triangleq f_p(w) - f(w)$. Intuitively, f_p is approximately a max-pooled version of f with a pooling kernel of width 2ρ, and SAM minimizes f_p. From left to right are the local minima centered at w_1, w_2, w_3, and the valleys become flatter. Since $f_p(w_1) = f_p(w_3) < f_p(w_2)$, SAM prefers w_1 and w_3 to w_2. *However, a low f_p could appear in both sharp (w_1) and flat (w_3) minima, so f_p might disagree with sharpness.* On the contrary, a smaller surrogate gap h indicates a flatter loss surface (Lemma 3.3). From w_1 to w_3, the loss surface is flatter, and h is smaller.

Figure 5: Caption

So they introduce minimizing this with low $h(w) = max_{\|\epsilon\| \leq \rho} f(w+\epsilon) - f(w)$

Goal to minimize this without giving up the minimization of f_p, thus we decompose the gradient:

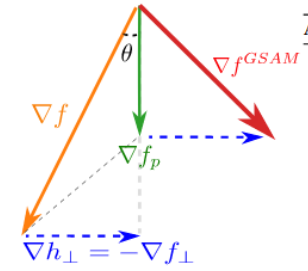


Figure 2: ∇f is decomposed into parallel and vertical ($\nabla f_\perp$) components by projection onto ∇f_p. $\nabla f^{GSAM} = \nabla f_p - \alpha \nabla f_\perp$

Algorithm 1 GSAM Algorithm

For $t = 1$ to T

0) ρ_t schedule: $\rho_t = \rho_{min} + \frac{(\rho_{max} - \rho_{min})(lr - lr_{min})}{lr_{max} - lr_{min}}$

1a) $\Delta w_t = \rho_t \frac{\nabla f^{(t)}}{\|\nabla f^{(t)}\| + \epsilon}$

1b) $w_t^{adv} = w_t + \Delta w_t$

2) Get $\nabla f_p^{(t)}$ by back-propagation at w_t^{adv}.

3) $\nabla f^{(t)} = \nabla f_\parallel^{(t)} + \nabla f_\perp^{(t)}$ Decompose $\nabla f^{(t)}$ into components that are parallel and orthogonal to $\nabla f_p^{(t)}$.

4) Update weights:

Vanilla $w_{t+1} = w_t - \eta_t \nabla f^{(t)}$

SAM $w_{t+1} = w_t - \eta_t \nabla f_p^{(t)}$

GSAM $w_{t+1} = w_t - \eta_t (\nabla f_p^{(t)} - \alpha \nabla f_\perp^{(t)})$

Figure 6: Caption

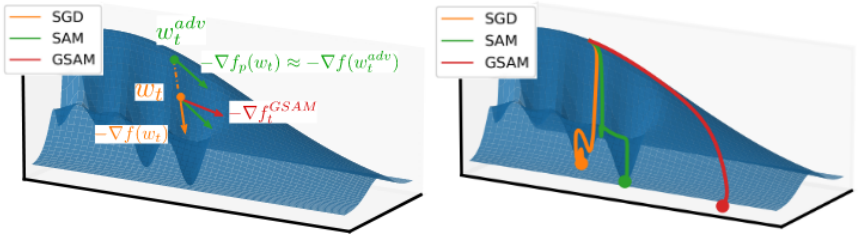


Figure 3: Consider the loss surface with a few sharp local minima. **Left**: Overview of the procedures of SGD, SAM and GSAM. SGD takes a descent step at w_t using $\nabla f(w_t)$ (orange), which points to a sharp local minima. SAM first performs gradient ascent in the direction of $\nabla f(w_t)$ to reach w_t^{adv} with a higher loss, followed by descent with gradient $\nabla f(w_t^{adv})$ (green) at the perturbed weight. Based on $\nabla f(w_t)$ and $\nabla f(w_t^{adv})$, GSAM updates in a new direction (red) that points to a flatter region. **Right**: Trajectories by different methods. SGD and SAM fall into different sharp local minima, while GSAM reaches a flat region. A video is in the supplement for better visualization.

Figure 7: Caption

Similar idea and concussions were found in [T. Li et al. 2024] through an empirical analysis of SAM. There the idea of GSAM was combined with the efficient computational scheme derived in VaSSO [B. Li and Giannakis 2023] where the choice of $\hat{\epsilon}$ is looked at as an adversarial step and it is argued that due to the stochastic nature of the process $\hat{\epsilon}$ is not estimating the right adversarial direction as one adversarial for a given batch might be friendly to others or most. The choice of $\hat{\epsilon} = \rho \frac{d_t}{\|d_t\|}, \quad d_t = (1-\alpha) d_{t-1} + \alpha g_t(w_t)$ is showed to hold similar convergence and better sharpness representation close to optimum as well as empirical performance.

3.4 What about computational performance

The above methods notably trying to find and address accuracy issues within SAM, nevertheless they have the same computational complexity. The only result I know claiming to compete with previous results in the speed of SGD is SAF[Du, Zhou, et al. 2022]. The idea here is to estimate flatness from historically computed parameters. A memory efficient version was also created.

$$\hat{\epsilon} = L_{\mathcal{B}}^{tra}(f_\tau, \mathbb{Y}^{(}e - E)) = \frac{\lambda}{B} \sum_{x_i \in B,\ y_i \in \mathbb{Y}^{(}e-E)} KL(\frac{1}{\tau}y, \frac{1}{\tau}f_\tau(x_i))$$

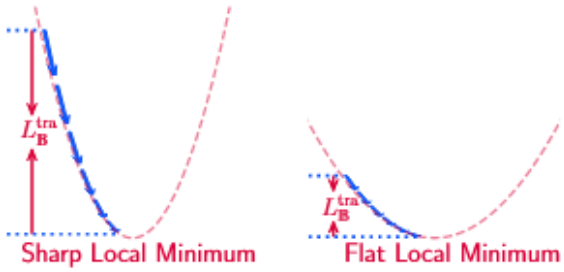


Figure 8: Caption

3.5 data less improvement and realistics use cases

There is multiple claim made in literature to the point that SAM algorithms ade more significant improvement to the result when training is done with less data such claims can be found in [Bahri, Mobahi, and Tay 2022] and [Chen, Hsieh, and Gong 2022]. In these papers SAM was used for ViT and LLMs for improving generalization.

In [Qu et al. 2022] SAM was used to Federated learning and when combined the vanilla SAM in local machines with momentum method and using FedAVG for aggregation state of the art performance similar to other methods has been found.

4 My work on SAM

The basis of my work on SAM like methods originates from the paper [Ujváry et al. 2022]. In this the authors related the SAM method to variational inference and similar method based on the similarity, that in all cases the loss we try to optimize looks like a perturbation with a covariance matrix plus a penalty term.

$$L_{MFVI}(\mu, \Sigma) = \mathbb{E}_{\tau \sim \mathcal{N}(\mu,\Sigma)} L(\tau) + \frac{1}{N} KL[\mathcal{N}(\mu, \Sigma || \mathcal{N}(0, \sigma_0^2 I)]$$

$$L_{SAM}(\mu, \Sigma) = max_{\epsilon^T \Sigma^{-1} \epsilon \leq p} [L(\mu + \epsilon) - L(\mu)] + L(\mu) + \alpha \|\mu\|_2^2$$

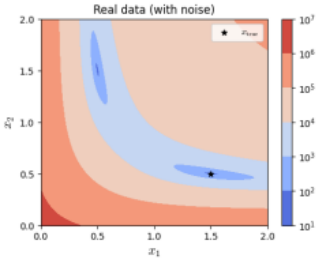
4.1 My results

From this the simple question I tried to investigate what choice of diagonal matrix can make sense, and weather if we can choose it to related to the problem. What would it mean?

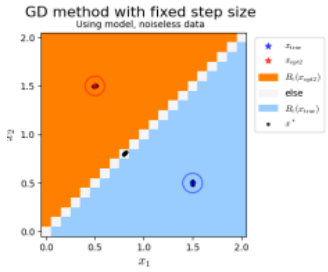
In general we could say that we are searching for minima, where we want different tolerance on different parameters. It is easy to construct such toy problem in 2D where it is interpretable nicely. My results:

Name	perturbation	covariance	$\Sigma =$	Penalty
SAM (Foret et al., 2020)	worst-case	fixed	$\frac{\rho^2}{p} I$	$\mathrm{KL} = L_2$
Random SAM (MFVI μ only)	Gaussian	fixed	$\frac{\rho^2}{p} I$	KL
MFVI (Hinton and van Camp, 1993)	Gaussian	learned	$\mathrm{diag}(\sigma_i)$	KL
Variational SAM	worst-case	learned	$\mathrm{diag}(\sigma_i)$	KL
Adaptive SAM (Kwon et al., 2021)	worst-case	μ-adaptive	$\mathrm{diag}(\lvert\frac{1}{\mu_i}\rvert)$	L_2
Fisher SAM (Kim et al., 2022)	worst-case	μ-adaptive	$\mathrm{diag}(F(\mu))$	L_2
Evolution Strategy (ES) (Rechenberg, 1978; Beyer and Schwefel, 2002)	Gaussian	fixed	$\frac{\rho^2}{p} I$	none
CMA-ES (Hansen and Ostermeier, 2001), VO (Staines and Barber, 2012), NES (Wierstra et al., 2008)	Gaussian	learned	$\mathrm{diag}(\sigma_i)$	none

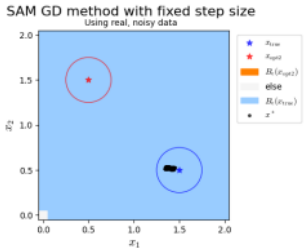
Table 1: An overview of methods mentioned in this work. All methods can be related to MFVI or SAM by changing the perturbation type, shape of Σ, or penalty terms.



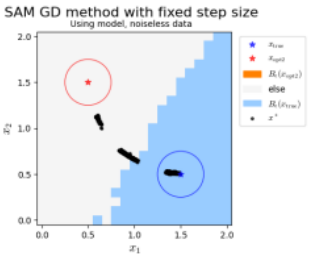
(a) Loss landscape



(b) Convergence of SGD

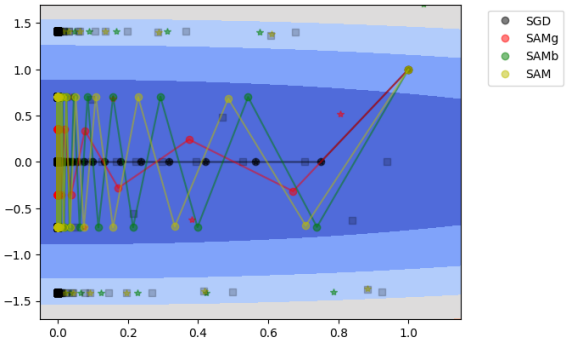


(c) Noisy convergence plot

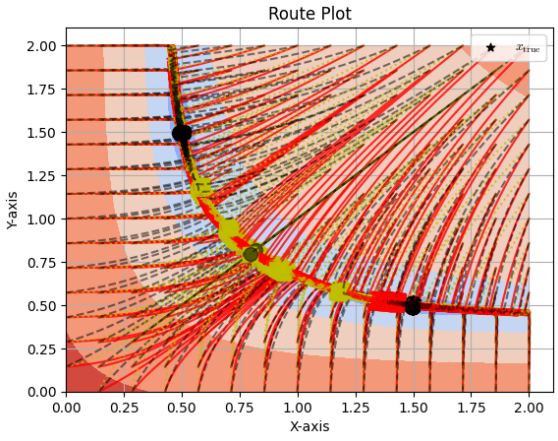


(d) Noiseless convergence plot

During my thesis I tried to test it out for other settings, but I showed that it can only have limited advantage in 1D problems. It can't help with the solutions of ill-conditioned linear systems and that it is not useful for fair ML.



(a) Convergence locally on $ax_1^2 + bx_2^2$



(b) Convergence routes

During my PhD I worked on the idea if it can be used to improve the training or fine tuning

of ViT, but my results so far indicate that there is no practical improvement achieved compared with the traditional SAM method. On reason is that even though we might have some motivation to prefer imposing sharpness in some parameter, but due to rescaling it imposes no practical improvement as there is no downside to have the same sharpness on all parameter.

Questions

- Can we found a DL or Inverse problem setting, where we have sound motivation for trying this method?
- What is the exact mechanism which leads to possible better performance on low data? (How low data is feasible?)
- Best to my knowledge using adaptive ρ shrpness parameter havn't been investigated so far.

References

Bahri, Dara, Hossein Mobahi, and Yi Tay (Mar. 2022). *Sharpness-Aware Minimization Improves Language Model Generalization*. DOI: 10.48550/arXiv.2110.08529. arXiv: 2110.08529. (Visited on 11/26/2024).

Chen, Xiangning, Cho-Jui Hsieh, and Boqing Gong (Mar. 2022). *When Vision Transformers Outperform ResNets without Pre-training or Strong Data Augmentations*. DOI: 10.48550/arXiv.2106.01548. arXiv: 2106.01548. (Visited on 11/27/2024).

Dinh, Laurent et al. (May 2017). *Sharp Minima Can Generalize For Deep Nets*. DOI: 10.48550/arXiv.1703.04933. arXiv: 1703.04933. (Visited on 11/27/2024).

Du, Jiawei, Hanshu Yan, et al. (May 28, 2022). *Efficient Sharpness-aware Minimization for Improved Training of Neural Networks*. DOI: 10.48550/arXiv.2110.03141. arXiv: 2110.03141. URL: http://arxiv.org/abs/2110.03141 (visited on 11/26/2024). Pre-published.

Du, Jiawei, Daquan Zhou, et al. (Dec. 2022). "Sharpness-Aware Training for Free". In: *Advances in Neural Information Processing Systems* 35, pp. 23439–23451. (Visited on 11/26/2024).

Foret, Pierre et al. (Apr. 2021). "Sharpness-Aware Minimization for Efficiently Improving Generalization". In: arXiv:2010.01412. arXiv:2010.01412 [cs, stat]. DOI: 10.48550/arXiv.2010.01412. URL: http://arxiv.org/abs/2010.01412.

Hochreiter, Sepp and Jürgen Schmidhuber (Jan. 1, 1997). "Flat Minima". In: *Neural Computation* 9.1, pp. 1–42. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.1.1. URL: https://doi.org/10.1162/neco.1997.9.1.1 (visited on 11/27/2024).

Jiang, Yiding et al. (2019). *Fantastic Generalization Measures and Where to Find Them*. arXiv: 1912.02178 [cs.LG]. URL: https://arxiv.org/abs/1912.02178.

Keskar, Nitish Shirish et al. (2017). "On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima". In: *International Conference on Learning Representations*. (Visited on 09/27/2024).

Kwon, Jungmin et al. (June 2021). *ASAM: Adaptive Sharpness-Aware Minimization for Scale-Invariant Learning of Deep Neural Networks*. DOI: 10.48550/arXiv.2102.11600. arXiv: 2102.11600. (Visited on 11/26/2024).

Li, Bingcong and Georgios Giannakis (Dec. 2023). "Enhancing Sharpness-Aware Optimization Through Variance Suppression". In: *Advances in Neural Information Processing Systems* 36, pp. 70861–70879. (Visited on 11/26/2024).

Li, Tao et al. (Mar. 2024). *Friendly Sharpness-Aware Minimization*. DOI: 10.48550/arXiv.2403.12350. arXiv: 2403.12350. (Visited on 11/26/2024).

Liu, Yong et al. (2022). "Towards Efficient and Scalable Sharpness-Aware Minimization". In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12360–12370. URL: https://openaccess.thecvf.com/content/CVPR2022/html/Liu_Towards_Efficient_and_Scalable_Sharpness-Aware_Minimization_CVPR_2022_paper.html (visited on 11/26/2024).

Mi, Peng et al. (Dec. 2022). "Make Sharpness-Aware Minimization Stronger: A Sparsified Perturbation Approach". In: *Advances in Neural Information Processing Systems* 35, pp. 30950–30962. (Visited on 11/26/2024).

Qu, Zhe et al. (June 2022). "Generalized Federated Learning via Sharpness Aware Minimization". In: *Proceedings of the 39th International Conference on Machine Learning*. PMLR, pp. 18250–18280. (Visited on 11/26/2024).

Ujváry, Szilvia et al. (Oct. 2022). "Rethinking Sharpness-Aware Minimization as Variational Inference". In: *NeurIPS*. arXiv. arXiv: 2210.10452 [cs, stat]. (Visited on 09/27/2024).

Zhao, Yang, Hao Zhang, and Xiuyuan Hu (Apr. 2023). *Randomized Sharpness-Aware Training for Boosting Computational Efficiency in Deep Learning*. DOI: 10.48550/arXiv.2203.09962. arXiv: 2203.09962. (Visited on 11/26/2024).

Zhuang, Juntang et al. (Mar. 2022). *Surrogate Gap Minimization Improves Sharpness-Aware Training*. DOI: 10.48550/arXiv.2203.08065. arXiv: 2203.08065. (Visited on 11/27/2024).