Loss Landscape Geometry, Optimization Dynamics, and Probing Methods

Inderjeet

MA24M011

Abstract

The loss surfaces of modern neural networks are extremely high dimensional and notoriously non-convex, yet they are remarkably easy to optimize in practice. This raises a natural question: *what properties of the landscape make such stable training possible?*

A large part of the answer lies in the geometry of the loss function. Curvature, flatness, sharpness, and the way different minima are connected all influence how gradients behave, how fast optimizers converge, and why certain solutions generalize better.

In this assignment, we bring together these geometric ideas and study:

- key quantities such as the largest Hessian eigenvalue, Hessian trace, gradient norms, flatness measures, and mode connectivity,
- how these geometric features shape optimization dynamics,
- how they relate to generalization (including perturbation and PAC–Bayes perspectives),
- how architectural choices influence landscape geometry,
- and practical, scalable methods for probing geometry in real networks.

Contents

1 Introduction

Despite their massive size and highly non-convex loss surfaces, deep neural networks can often be optimized reliably using simple first-order methods. Empirically, training tends to converge smoothly and finds solutions that generalize well—even when the model has far more parameters than training samples.

Understanding this phenomenon requires looking beyond the loss value itself and examining the **geometry of the landscape**. Certain geometric patterns, such as broad flat valleys or connected basins, make training easier. In contrast, sharp regions with steep curvature can cause instability or poor generalization.

This report focuses on several geometric quantities that provide insight into the shape of the loss surface and the dynamics of optimization. We discuss their definitions, what they reveal about learning, and how to estimate them efficiently for modern networks.

2 Geometry of the Loss Landscape

We begin by introducing the main concepts used to characterize the local and global geometry of the loss function.

2.1 Largest Hessian Eigenvalue $\lambda_{\max}$

The Hessian,

$$H(\theta) = \nabla_\theta^2 L(\theta),$$

captures local curvature. Its largest eigenvalue,

$$\lambda_{\max}(H) = \max_{\|v\|=1} v^\top H v,$$

corresponds to the steepest curvature direction.
Geometric interpretation:

- A large $\lambda_{\max}$ indicates a sharp region where the loss rises quickly in at least one direction.
- A small $\lambda_{\max}$ suggests the landscape is smooth even in the steepest direction.

Optimization relevance: Gradient descent is stable only when

$$\eta \lambda_{\max} < 2.$$

Thus, sharp regions force the use of small learning rates, slowing training and causing zig-zag behavior in narrow valleys.
Generalization relevance: Solutions with low $\lambda_{\max}$ (flat minima) tend to be more stable under small perturbations and usually generalize better, whereas sharp minima often overfit.
Architecture effect: Modern architectural ideas—such as residual connections, BatchNorm, and increased width—naturally reduce $\lambda_{\max}$ and make the landscape smoother.

2.2 Hessian Trace $\mathrm{tr}(H)$

The trace of the Hessian,

$$\mathrm{tr}(H) = \sum_{i=1}^{d} \lambda_i,$$

summarizes the overall magnitude of curvature across all directions.
Interpretation: If we add small isotropic noise $\delta \sim \mathcal{N}(0, \sigma^2 I)$ to the parameters, then:

$$\mathbb{E}[L(\theta + \delta)] \approx L(\theta) + \frac{1}{2}\sigma^2 \,\mathrm{tr}(H).$$

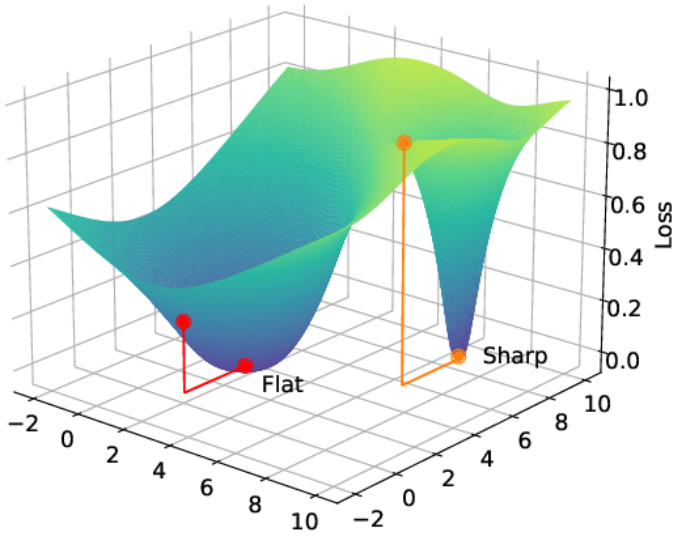

Figure 1: Illustration of sharp vs. flat minima

A large trace therefore amplifies noise, while a small trace corresponds to broad flat regions where the loss barely changes.

Architecture effect: Overparameterized and wide networks often have many eigenvalues close to zero, reducing the effective curvature and making the landscape much flatter overall.

2.3 Gradient Norm $\|\nabla L(\theta)\|$

The gradient norm,

$$\|\nabla L(\theta)\| = \sqrt{\sum_i \left(\frac{\partial L}{\partial \theta_i}\right)^2},$$

indicates how steep the surface is locally.

Large norms early in training help the optimizer make progress quickly, while small norms suggest slowing dynamics or proximity to critical points. Architectures like ResNets maintain healthier gradient flow, avoiding vanishing gradients in very deep models.

2.4 Flatness and Sharpness: Local Picture

A minimum θ^* is considered flat if:

$$L(\theta^* + \delta) \approx L(\theta^*) \quad \text{for many small } \delta.$$

Flat regions tolerate parameter noise, while sharp minima react strongly to even tiny perturbations. This idea is closely tied to robustness and generalization.

2.5 Mode Connectivity

An intriguing discovery in deep learning research is that independently trained solutions often lie in the same broad basin. Given two such solutions θ_1 and θ_2, linear interpolation:

$$\theta(t) = (1-t)\theta_1 + t\theta_2,$$

reveals whether they are separated by a barrier or connected by a smooth low-loss path.

Connected minima typically correspond to flatter regions and often generalize better.

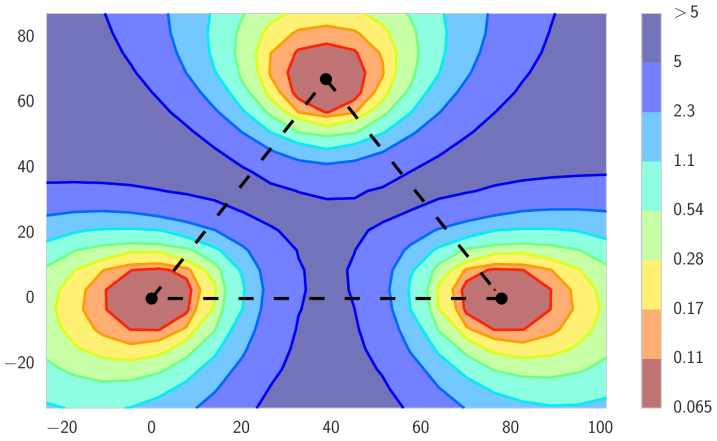

Figure 2: Mode connectivity between minima

3 Optimization Dynamics Explained by Curvature

The behavior of gradient-based optimizers is deeply influenced by curvature. Near a local minimum,

$$L(\theta) \approx L(\theta^*) + \frac{1}{2}(\theta - \theta^*)^\top H(\theta - \theta^*),$$

and gradient descent updates approximately follow:

$$e_{t+1}^{(i)} = (1 - \eta\lambda_i)\, e_t^{(i)}.$$

This expression immediately shows how curvature determines stability and convergence speed.

3.1 Stability and Condition Number

The learning rate must satisfy:

$$0 < \eta < \frac{2}{\lambda_{\max}}.$$

The condition number,

$$\kappa = \frac{\lambda_{\max}}{\lambda_{\min}},$$

governs how well-behaved the landscape is. A high κ (poor conditioning) causes zig-zag trajectories and slow convergence, whereas low κ leads to smooth, efficient optimization.

Architectures like ResNets reduce $\lambda_{\max}$ and effectively increase the smaller eigenvalues, greatly improving conditioning.

3.2 Why SGD Prefers Flat Minima

SGD updates include noise:

$$\theta_{t+1} = \theta_t - \eta(\nabla L(\theta_t) + \xi_t).$$

In sharp minima, even small noise kicks the optimizer out of the region. In flat minima, noise has little effect, allowing SGD to linger and eventually settle there. This gives SGD an implicit preference for flatter solutions—one reason why overparameterized models often generalize well.

3.3 Momentum and Ill-Conditioned Valleys

Momentum smooths oscillations in sharp directions and accelerates progress along flatter directions. By averaging gradients across iterations, it reduces zig-zag behavior and effectively reshapes the geometry of the landscape experienced by the optimizer.

3.4 Adaptive Optimizers and Curvature

Adaptive methods like Adam rescale each coordinate based on recent gradient magnitudes. This adaptively shrinks updates in sharp directions (large gradients) and enlarges updates in flatter ones—providing a form of curvature-aware preconditioning.

3.5 Learning-Rate Warmup

Very deep or wide networks, especially transformers, often exhibit unstable curvature early in training. Warmup gradually increases the learning rate to prevent the optimizer from stepping too aggressively before curvature stabilizes.

4 Flatness, Generalization, and Width

4.1 PAC–Bayes / Perturbation View

PAC–Bayes bounds frame generalization as a combination of training error and a complexity term. Flat minima correspond to parameter regions that can be described more compactly, resulting in lower complexity and tighter generalization bounds. Sharp minima, being highly sensitive, require far more precision to specify and thus generalize worse.

4.2 Role of Width

Wider networks tend to:

- reduce $\lambda_{\max}$,
- push many eigenvalues close to zero,
- lower the effective trace,
- expand flat regions in the landscape.

This explains why very wide networks often train faster and generalize surprisingly well, even deep in the overparameterized regime.

4.3 Architecture $\Rightarrow$ Geometry $\Rightarrow$ Generalization

Architectural choices influence geometry, which in turn influences generalization. Residual connections, BatchNorm, and width all smooth the loss landscape, making it easier for optimizers to find robust, flat solutions.

5 Efficient Landscape Probing Methods

Directly computing the Hessian is impractical ($O(n^2)$) for large networks, but several practical techniques allow us to estimate curvature and flatness efficiently.

These include power iteration for $\lambda_{\max}$, Hutchinson's estimator for the trace, sharpness measures based on perturbations, and mode connectivity analysis.

5.1 Estimating $\lambda_{\max}$: Power Iteration

Power iteration repeatedly applies *Hessian–vector products (HVPs)*—computed efficiently using the Pearlmutter trick, which evaluates Hv without ever forming the Hessian explicitly—to approximate the largest eigenvalue $\lambda_{\max}$. This requires only a forward and backward pass (roughly the cost of computing a gradient), making it a computationally cheap and scalable method for estimating sharpness in modern deep networks.

5.2 Estimating the Hessian Trace: Hutchinson Estimator

Randomized trace estimation uses noise vectors to approximate

$$\mathrm{tr}(H) \approx \frac{1}{K} \sum_{k=1}^{K} v_k^\top (H v_k).$$

This method scales cleanly to large models and provides reliable curvature estimates.

5.3 Gradient Norm Tracking

Since gradients are computed during training anyway, tracking their norms is nearly free and gives insight into whether optimization is progressing or stagnating.

5.4 Sharpness via ϵ-Perturbations

Perturbation-based sharpness measures directly evaluate how sensitive a solution is to small parameter noise—an intuitive and widely used proxy for flatness.

5.5 Weight-Noise Robustness

Adding Gaussian noise to weights and checking how performance deteriorates offers a straightforward measure of valley width and robustness.

5.6 Mode Connectivity

Evaluating loss along interpolation paths between independently trained minima reveals whether they lie in the same basin or in separated regions of the loss surface.

6 Implementation Code

All the probing methods described above can be implemented with standard autodiff tools. You can refer the link: Github

7 Takeaway and Synthesis

Across the geometric measures and optimization analyses presented in this report, a coherent picture of modern deep learning emerges.

SGD and generalizable solutions. Although neural network loss landscapes are highly non-convex, the implicit noise in stochastic gradient descent naturally biases the optimizer toward wide, flat valleys. Such regions are robust to parameter perturbations and exhibit low PAC–Bayes complexity, helping explain why SGD reliably finds solutions that generalize well in practice.

Architectural influence on landscape topology. Design choices such as residual connections, Batch Normalization, and increased width do more than improve empirical performance—they reshape the underlying loss landscape. These architectures tend to reduce extreme curvature, improve conditioning, and create large, connected basins that are easier for optimizers to explore.

Geometry as a predictor of trainability and generalization. Curvature quantities such as the largest eigenvalue $\lambda_{\max}$, the Hessian trace, and the overall eigenvalue spectrum correlate strongly with both optimization behavior and generalization. Flatness, low curvature, and mode connectivity consistently signal solutions that are robust and easier to reach, while sharp or poorly conditioned regions often lead to optimization instability or overfitting.

Predicting optimization difficulty from landscape structure. Several geometric indicators reliably forecast how challenging training will be. Large $\lambda_{\max}$ values restrict the allowable learning rate, high condition numbers produce zig–zagging dynamics, and large trace values indicate sensitivity to noise. Conversely, well-conditioned spectra and smooth curvature dynamics predict stable, efficient optimization.

Taken together, these insights show that geometry, optimization, and architecture are deeply intertwined. Understanding the structure of the loss landscape—both locally through curvature and globally through connectivity—offers powerful intuition for why deep networks train as effectively as they do, and why certain solutions ultimately generalize better than others.

8 References

- *The Elements of Statistical Learning* — Trevor Hastie, Robert Tibshirani, and Jerome Friedman.
- *Understanding Deep Learning* — Simon J.D. Prince.
- *Mathematics of Neural Networks* — Bart M. N. Smets.
- *Neural Network Theory* — Philipp Christian Petersen, University of Vienna.