Data (Attribution) for Alignment

Growing neural networks

Louis Jaburi
EleutherAI

Iliad Intensive, April 2026
[Draft 2026.04.13]

Contents

Introduction

Contents. These notes are structured as follows:

1. "Causality and Counterfactuals" motivates the problem of data attribution through the lens of causal reasoning, exhibiting the shortcomings of naive counterfactual approaches.
2. "Influence Functions" introduces the classical theory of influence functions for statistical models and discusses their translation to modern neural networks.
3. "Bayesian Influence Functions" introduces susceptibilities and Bayesian influence functions, connecting them to the classical theory via a power series expansion.
4. "Unrolling" presents the training-dynamics approach to attribution, deriving the basic formula and connecting it to influence functions in the appropriate limit.
5. "Practical Considerations and Open Problems" discusses distributional versus single-model attribution, entangled latent concepts, and the distinction between similarity metrics and genuine attribution.

Prerequisites. We assume the following background, most of it standard for a machine-learning audience.

- *Linear algebra*: vectors, matrices, inner products, eigenvalues, quadratic forms, Hessians.
- *Multivariable calculus*: gradients, the chain rule, Taylor expansion, the implicit function theorem.
- *Probability*: expectations, conditional probability, Bayes' rule, Gaussian distributions.
- *Deep learning basics*: stochastic gradient descent, empirical risk minimisation, a standard supervised training loop.

Familiarity with the material from the singular learning theory tutorial (degeneracy, the local learning coefficient, posterior broadening) is useful but not required. We flag connections to the SLT day in remarks as they arise.

Fast-track. There is more material here than comfortably fits into a single day. Readers short on time may prefer one of the following routes; each is self-contained.

Route 1. *From classical to modern influence functions.* Skim Section 1 for motivation, then read Section 2 in full, including the derivation exercise.

Route 2. *The Bayesian perspective.* Read Section 2.1 for the influence function formula, then Section 3 in full. The connection exercise (Section 3.2) is the conceptual payoff.

Route 3. *Training-dynamics attribution.* Skim Section 2, then read Section 4 in full. The long exercise in Section 4.3 recovers the influence function as a limit of unrolling.

All three routes depend on the motivational material in Section 1. If reading only one section, read that one.

Acknowledgements. This tutorial was developed for the April 2026 Iliad Intensive and its template was inspired by the SLT day. Exercises and examples adapted from specific sources are credited inline.

1 Causality and Counterfactuals

1.1 Data attribution as causal analysis

At the heart of data attribution, we find a causal question: **Which training examples caused a model to behave the way it does?** If mechanistic interpretability aims for the causal analysis of a forward pass, data attribution aims for the causal analysis of a training run.[1] Our north star is a decomposition of the training data into "causes" of the model's behaviour.

Two asymmetries between the two settings are worth keeping in mind from the start. First, both problems inherit the same conceptual difficulties of causal reasoning in general (explained in the next section) and there is no reason to expect data attribution to escape them. Second, a forward pass is cheap, while a training run is not. Where mechanistic interpretability can validate a claim by running many ablations, data attribution cannot in general afford even one retraining. That being said, there are also advantages: Generally speaking, the training process is more robust and the ablations we run may be more human understandable than high-dimensional modification of activations or weights.

Now, before developing such data attribution methods, we should be aware of the tension between causality and the ways we will validate a hyptohesis. Finding a "true" causal relationship in practice can be much more difficult than it may seem at first glance. In the following section we take the question seriously, and find that the naive counterfactual answer fails in ways that matter.

1.2 Counterfactual attribution and its failures

The classical notion of causation is due to Lewis (1973): A caused B iff, had A not occurred, B would not have occurred. Applied to data attribution, this says that a training example "causes" a behaviour when removing it would have altered the behaviour, also knows as the *leave-one-out (LOO) counterfactual*. We relax this defintion from a binary to a gradual one.

Definition 1.1 (Counterfactual attribution)**.** We attribute an action A to a behaviour B iff

(i) had A occurred, B would have occurred to a higher degree;

(ii) had A not occurred, B would have occurred to a lesser degree (or not at all).

Definition 1.1 is the simplest formalisation of cause. But as Mueller (2024) emphasises, it breaks in at least two cases.

Overdetermination: redundant causes are missed. When two training examples are each *sufficient* to produce a behaviour, removing either one leaves the behaviour intact, and LOO attributes credit to neither.

Example 1.2 (Overdetermination in the bakery)**.** OpenCroissant, a bakery, sources four ingredients: flour, butter, yeast, and sugar. Each can come from a standard or premium supplier. The bakery has observed that using all premium ingredients produces exceptionally fluffy croissants and wishes to identify which upgrades are responsible.

Unbeknownst to the bakery, exceptional fluffiness depends on two independent leavening mechanisms.

[1] In Section 4 we make this analogy explicit.

- *Steam leavening.* Premium butter has higher fat content and better water distribution, producing more steam and hence more lift.
- *Biological leavening.* A premium yeast strain ferments more vigorously, producing more gas and hence more lift.

Either mechanism alone causes exceptional fluffiness: puff pastry rises from steam alone, and bread rises from yeast alone. Downgrading premium butter to standard therefore has negligible effect on fluffiness (the threshold is still met), and likewise for yeast. Counterfactual attribution (Definition 1.1) concludes that neither butter nor yeast was a cause albeit both were.

The same pattern may arise in learning, where once a behavior is learned, i.e. has hit saturation, additional examples will not affect it and appear as causally irrelevant.

Non-transitivity: counterfactuals do not chain. Direct ablation will not capture *how* an action causes a behavior.

Example 1.3 (Non-transitivity of counterfactuals)**.** The following scenario is due to Ned Hall, as reported in Hitchcock (2001).

(A) A hiker is wandering through the mountains when a boulder rolls down the hill.

(B) The hiker sees the boulder and dodges.

(C) The hiker is unharmed.

Counterfactually, (A) caused (B): had the boulder not rolled, the hiker would not have dodged. And (B) caused (C): had the hiker not dodged, she would have been crushed. But (A) did not cause (C): whether or not the boulder rolled, she ends up unharmed.

In the data-attribution setting this corresponds to chains of mediation. A training example may induce an intermediate feature, and that feature may drive a downstream capability, without the datapoint itself passing the counterfactual test against the capability end-to-end. Methods that look only at marginal effects miss the mediation.

1.3 Beyond leave-one-out: Shapley values

A natural response to overdetermination is to stop looking only at the LOO swap and instead ask how a training example contributes across *all* subsets of the training set it could be part of. This is the idea behind *Shapley values* (Shapley, 1953).

Definition 1.4 (Shapley value)**.** Let $v : 2^{[n]} \to \mathbb{R}$ be a set function (a "game"). The Shapley value of element i is

$$\varphi_i(v) \;:=\; \frac{1}{n} \sum_{S \subseteq [n] \setminus \{i\}} \binom{n-1}{|S|}^{-1} \big(v(S \cup \{i\}) - v(S)\big).$$

We can think of $[n]$ as a set of players and $v(S)$ as the value of the coalition. In our previous example, the players are the premium ingredients and the value is the resulting fluffiness. Then $v(S)$ will measure the fluffiness of the croissant when the ingredients in S are premium and the rest are standard.

Shapley values partially address overdetermination. Returning to the baker example: in coalitions $S \subset [n]$ where neither butter nor yeast is premium, upgrading one has a large marginal effect; in coalitions where the other leavener is already premium, the marginal effect is small. Averaging gives each ingredient roughly half the credit.

Remark 1.5 (Shapley still dilutes credit). Shapley values redistribute credit more fairly but do not fully solve overdetermination. Each of butter and yeast receives roughly half the credit for fluffiness, yet each *alone* is sufficient, so one could argue each deserves the full credit. Furthermore, Shapley values over a training set of size n require evaluating $v(S)$ on 2^n coalitions, each demanding a retraining. This is hopeless for realistic models.

1.4 Notation and goals

We close the section by fixing notation that will be used throughout.

- A training set is $\mathcal{D} = \{z_i\}_{i=1}^n$ with $z_i = (x_i, y_i)$.
- A model is a function $f_w : X \to Y$ parameterised by $w \in W \subset \mathbb{R}^{d_p}$.
- The per-example loss is $L_j(w) = L(w, z_i) = \ell(f_w(x_i), y_i)$ for some pointwise loss ℓ.
- We introduce a vector of *data weights* $\beta = (\beta_1, \ldots, \beta_n) \in \mathbb{R}^n$ and define the *weighted training loss*
$$L(w; \beta) \;:=\; \frac{1}{n} \sum_{i=1}^{n} \beta_i \, L(w, z_i),$$
and we write $L(w) := L(w, 1)$ for the unweighted loss.
- The behaviour we want to attribute is a differentiable scalar observable $\phi : W \to \mathbb{R}$. This could be the loss on a held-out point, an average loss on a task, the logit of a specific completion, or any other scalar function of the trained parameters.

In general we want to know "If I change β_i, how will this affect ϕ?" This question is typicall split into two parts:

- *Parameter influence*: "If I change β_i, how will this affect the final parameters?"
- *Influence on ϕ*: "If I change the final parameters, how will this affect ϕ?"

In practice we may think of the influence then as a measure of the composition of these maps

$$B \xrightarrow{\pi} \text{"}W\text{"} \to \mathbb{R} \tag{1}$$
$$\beta \mapsto \pi(\beta) \mapsto \phi(w(\beta)). \tag{2}$$

where we are intentionally vague about the first map and where the different approaches, influence functions, Bayesian influence functions, and unrolling, differ in how they interpret it.

That being said, all attribution methods in these lecture notes can be phrased as estimators of partial derivatives or finite differences of this map. The *influence* of an example i is then computed as first-order approximation:

$$\boxed{\left.\frac{\partial\, \phi(\pi(\beta))}{\partial \beta_i}\right|_{\beta=\mathbf{1}}} \tag{3}$$

2 Influence Functions

In this section we develop *influence functions*: a closed-form, first-order approximation to the leave-one-out (LOO) counterfactual. The classical theory (Hampel, 1974; Cook, 1977) predates machine learning by decades. It was developed in robust statistics, where the question was which observations a parameter estimate is most sensitive to. For simple statistical models the answer is short: a single application of the implicit function theorem yields an inverse-Hessian-times-gradient formula whose factors have a clean operational interpretation.

The formula we arrive at then *can* be translated to the setup of neural networks, but its interpretation becomes unclear: The classical derivation assumes a unique global minimum and an invertible Hessian, neither of which holds for an neural network which is typically **not** trained to full convergence. We will see a partial fix for this issue in Section 2.2.

2.1 The influence function formula

Influence functions originate in robust statistics (Hampel, 1974): given a statistical estimator, how much does it change when a single observation is added or perturbed?

Example 2.1 (Influence on the mean)**.** Let $T(F) = \mathbb{E}_F[z]$ be the mean of a distribution F. If we contaminate F by placing a fraction ϵ of its mass at a point z, the new mean is

$$T\big((1-\epsilon)F + \epsilon\,\delta_z\big) \;=\; (1-\epsilon)\,\mathbb{E}_F[z] \;+\; \epsilon\,z \;=\; T(F) \;+\; \epsilon\,(z - T(F)).$$

The rate of change at $\epsilon = 0$ is $z - T(F)$: the influence of z on the mean is simply how far z is from the current mean. Observations far from the centre of the data move the mean the most.

We now translate this idea into the setting of Section 1.4, whose notation we use throughout. Recall that the weighted training loss $L(w;\beta) = \sum_i \beta_i L(w, z_i)$ defines a family of optimization problems parameterized by the weight vector β. In the case of influence functions, we measure the influence of a data point on the *optimal* parameters

$$w^*(\beta) \;:=\; \operatorname*{arg\,min}_{w \in W} L(w;\beta),$$

and we write $w^* := w^*(\mathbf{1})$ for the trained parameters. We assume throughout this subsection that w^* is the unique global minimum of L and that the Hessian $H := \nabla_w^2 L(w^*)$ is positive definite, so that the implicit function theorem makes $w^*(\beta)$ well-defined and smooth in a neighborhood of $\beta = \mathbf{1}$. In the notation of Equation (1) we set $\pi(\beta) = w^*(\beta)$ and Equation (3) yields then the following formula:

Definition 2.2 (Influence function)**.** The *parameter influence* of training example z_i at the trained parameters w^* is

$$\mathcal{I}_{\text{param}}(z_i) \;:=\; \left.\frac{\partial w^*(\beta)}{\partial \beta_i}\right|_{\beta=\mathbf{1}}.$$

Under the assumptions above, the implicit function theorem gives (Exercise 2.3)

$$\boxed{\mathcal{I}_{\text{param}}(z_i) \;=\; -\,H^{-1}\,\nabla_w L(w^*, z_i).}$$

where H is the Hessian of L at w^*. For any differentiable observable $\phi : W \to \mathbb{R}$, the chain rule then yields the *influence on* ϕ:

$$\mathcal{I}(z_i, \phi) \;:=\; \left.\frac{\partial\,\phi(w^*(\beta))}{\partial \beta_i}\right|_{\beta=\mathbf{1}} \;=\; \nabla_w \phi(w^*)^\top\,\mathcal{I}_{\text{param}}(z_i) \;=\; -\,\nabla_w \phi(w^*)^\top\,H^{-1}\,\nabla_w L(w^*, z_i).$$

Thus we get three components

- $\nabla_w L(w^*, z_i)$ is the direction in parameter space along which z_i pulls the optimizer.
- H^{-1} rescales this direction by local curvature: directions in which the loss is flat (small eigenvalue of H) get amplified, since a small force in a flat direction moves the minimum a lot.
- $\nabla_w \phi(w^*)$ projects the parameter change onto the direction that matters for the observable ϕ.

In this section, we will focus on $\mathcal{I}_{\text{param}}(z_i)$. The influence on any observable ϕ is obtained by computing the dot product $\mathcal{I}_{\text{param}}(z_i) \cdot \nabla_w \phi(w^*)$.

Connection to leave-one-out. Setting $\beta = \mathbf{1} - e_i$ (i.e., removing z_i) and applying the first-order Taylor expansion gives

$$w^*(\mathbf{1} - e_i) \;-\; w^* \;\approx\; -\mathcal{I}_{\text{param}}(z_i) \;=\; H^{-1}\,\nabla_w L(w^*, z_i), \tag{4}$$

i.e. the parameter influence is, to first order, the change in w^* that would result from removing z_i from the training set entirely.

Exercise 2.3 (Derivation of the influence function)**.** Let $w^*(\beta) = \arg\min_{w \in W} L(w; \beta)$ as above and assume the Hessian $H = \nabla_w^2 L(w^*)$ is invertible.

(a) By differentiating the first-order optimality condition $\nabla_w L(w^*(\beta); \beta) = 0$ with respect to β_i, show that

$$\left.\frac{\partial w^*(\beta)}{\partial \beta_i}\right|_{\beta=\mathbf{1}} \;=\; -\,H^{-1}\,\nabla_w L(w^*, z_i).$$

Where in the argument did you use the assumption that w^* is the unique global minimum?

(b) Use the chain rule to obtain the formula for $\mathcal{I}(z_i, \phi)$ in Definition 2.2. Make sure to keep track of the evaluation at $\beta = \mathbf{1}$.

Exercise 2.4 (Linear regression)**.** Consider linear regression with training data $\{(x_i, y_i)\}_{i=1}^n$, $x_i \in \mathbb{R}^p$, and squared loss $L(w, (x_i, y_i)) = \frac{1}{2}(y_i - w^\top x_i)^2$. Write $A := \sum_i x_i x_i^\top$. In this case the minimizer, the influence function, and the exact LOO update all have closed-form solutions. We will see that in general the influence functions can diverge in extreme cases.

(a) Derive the minimizer w^* by setting $\nabla_w L(w) = 0$. Show that $w^* = A^{-1} \sum_i x_i y_i$.

(b) Compute the parameter influence $\mathcal{I}_{\text{param}}(z_j)$ from Definition 2.2.

(c) Define $w^*(\beta_j) := \arg\min_w \sum_{i \neq j} L(w, z_i) + \beta_j L(w, z_j)$, so that $\beta_j = 1$ is the unperturbed model and $\beta_j = 0$ corresponds to fully removing z_j. Using the Sherman–Morrison formula[a], show that

$$w^*(\beta_j) \;-\; w^* \;=\; -\,\frac{(1 - \beta_j)\, A^{-1} x_j\, r_j}{1 - (1 - \beta_j)\, h_j},$$

where $r_j := y_j - x_j^\top w^*$ is the residual at z_j and $h_j := x_j^\top A^{-1} x_j$ is the *leverage* of x_j. Note that this is a rational function of β_j. This is the ground truth effect of perturbing z_j.

(d) Expand the closed form from (c) as a Taylor series in $(1 - \beta_j)$ around the unperturbed point $\beta_j = 1$ by recognizing $1/(1 - (1 - \beta_j)h_j)$ as a geometric series, and obtain

$$w^*(\beta_j) - w^* \;=\; -\,A^{-1} x_j\, r_j \sum_{k=1}^{\infty} (1 - \beta_j)^k\, h_j^{k-1}.$$

Identify the $k = 1$ term with the parameter influence from part (b), and observe that the higher-order terms ($k \geq 2$) are suppressed by powers of the leverage h_j.

(e) Specialize to the LOO endpoint $\beta_j = 0$. Sum the geometric series in h_j to obtain the closed-form deletion update

$$w^*_{(-j)} - w^* \;=\; -\,A^{-1} x_j\, r_j \left(1 + h_j + h_j^2 + h_j^3 + \cdots\right) \;=\; -\,\frac{A^{-1} x_j\, r_j}{1 - h_j}.$$

The IF prediction is the h_j^0 leading term. Discuss the two regimes $h_j \to 0$ and $h_j \to 1$: when does keeping only the linear term suffice, and when do all the higher-order corrections matter?

[a] For an invertible matrix M and vectors u, v: $(M + uv^\top)^{-1} = M^{-1} - \frac{M^{-1}u\,v^\top M^{-1}}{1+v^\top M^{-1}u}$.

Remark 2.5 (What is leverage?). The quantity $h_j = x_j^\top A^{-1} x_j$ that appeared in Exercise 2.4 measures how *unusual* the input x_j is relative to the bulk of the design. To see this, observe that $\widehat{\Sigma}_{xx} := A/n = \frac{1}{n}\sum_i x_i x_i^\top$ is the empirical second-moment matrix of the inputs, so

$$h_j \;=\; \tfrac{1}{n}\, x_j^\top\, \widehat{\Sigma}_{xx}^{-1}\, x_j$$

is $1/n$ times a *Mahalanobis-style* squared norm of x_j, taken in the metric defined by the data itself. Geometrically: directions that the data samples a lot correspond to large eigenvalues of $\widehat{\Sigma}_{xx}$ and contribute little to h_j, while directions that are barely sampled correspond to small eigenvalues of $\widehat{\Sigma}_{xx}$ and so blow up under $\widehat{\Sigma}_{xx}^{-1}$. *Inputs that point in under-sampled directions get large leverage.* High-leverage points are unusual in feature space.

Figure 1 illustrates the effect on a simple linear regression: removing a low-leverage point (A, near the bulk of the data) barely changes the fit, and the IF approximation is accurate. Removing a high-leverage point (B, an outlier on the x-axis) changes the fit dramatically, and the IF underestimates the true LOO effect.

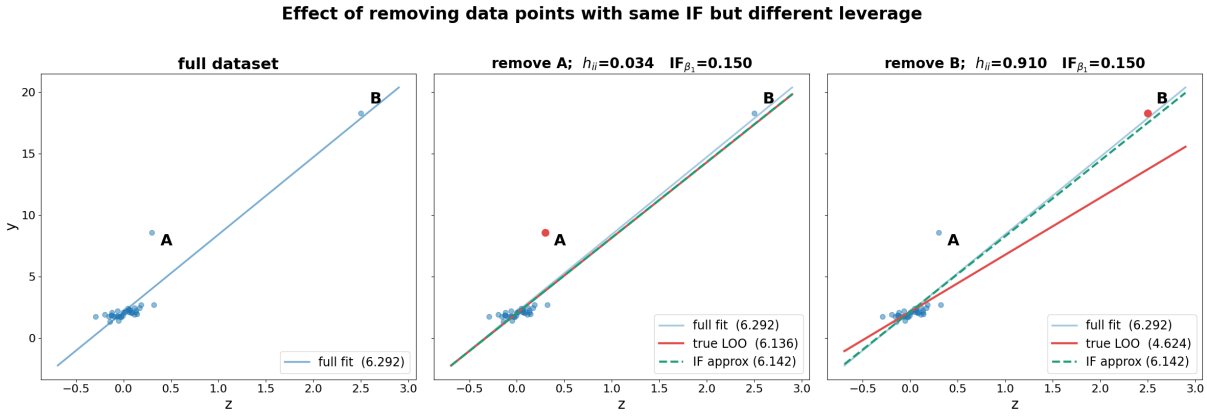

Figure 1: Influence approximation vs. LOO ground truth for a linear regression. *Left:* the full dataset. *Centre:* removing the low-leverage point A ($h_A \approx 0.03$) barely perturbs the fit; the IF prediction closely matches LOO. *Right:* removing the high-leverage point B ($h_B \approx 0.83$) moves the regression line substantially; the IF underestimates the true change. Note that both points have the same parameter influence.

2.2 Translation to modern neural networks

The derivation in Section 2.1 relied on two assumptions:

(i) w^* is the unique global minimum of the training loss;

(ii) the Hessian $H = \nabla_w^2 L(w^*)$ is positive definite.

While neither holds for a modern neural network, the quantities in Definition 2.2 are, in principle, computable. Koh and Liang (2017) popularized the influence function formula as a tool for deep learning, and a substantial body of work has tried to make it useful at scale.

(1) Non-uniqueness and a singular Hessian. For a neural network the global minimum is usually not unique. The set of minimizers is generically a positive-dimensional zero locus, and the Hessian at any point on it is singular: it has a nontrivial null space corresponding to directions of parameter movement that leave the loss unchanged. This is the degeneracy phenomenon observed in the SLT day, and it means we cannot invert H as in Definition 2.2.

The standard remedy is to *dampen* the Hessian: replace H^{-1} with $(H+\lambda I)^{-1}$ for some $\lambda > 0$. This is the same as adding an L_2-regularizer $\frac{\lambda}{2}\|w - w^*\|^2$ to the training objective $L(\beta, w)$; this yields a a new loss function $L_{reg}(w, \beta) = L(w, \beta) + \frac{\lambda}{2} \cdot ||w - w^*||^2$.

(2) Negative curvature. Even after damping away the zero eigenvalues, H at a non-converged checkpoint can have negative eigenvalues, in which case $(H + \lambda I)^{-1}$ may still be ill-conditioned or even point in the wrong direction. The standard fix is to replace H with the *Gauss–Newton Hessian* (GNH)

$$G \;=\; \mathbb{E}_{(x,y)\sim\mathcal{D}}\big[J^\top H_y J\big],$$

where $J = \partial f_w/\partial w$ is the parameter-output Jacobian and H_y is the Hessian of the per-example loss with respect to the network outputs (not the parameters). For the cross-entropy and squared losses H_y is positive semidefinite, hence so is G. The GNH is a local approximation to the true Hessian that ignores second derivatives of the network outputs with respect to the parameters. It is not hard to show that G and H differ by a term that vanishes at a critical point of L, so the GNH is a reasonable proxy for H when w^* is close to optimal.

(3) Non-converged checkpoints. In practice w^* is not a critical point of L. The classical formula in Definition 2.2 is then evaluated at a point where $\nabla L \neq 0$.

The fix proposed by Bae et al. (2022) is to replace the original training loss by a surrogate for which w^* *is* optimal by construction. Concretely, define the *proximal Bregman objective* (PBO) as

$$L_{\mathrm{PBO}}(w;\beta) \;=\; \sum_{i=1}^{n} D_{L_y}\big(f_w(x_i),\, f_{w^*}(x_i)\big) \;+\; \sum_{i=1}^{n}(\beta_i - 1)\, L(w, z_i) \;+\; \frac{\lambda}{2}\,\|w - w^*\|^2, \tag{5}$$

where $D_{L_y}(\hat{y}, \hat{y}^*)$ is the Bregman divergence of the output-space loss L_y around $\hat{y}^*$. The first term penalizes any change in the network's predictions on the training set, with the current model w^* as the reference; the second is the perturbation of interest; the third is the damping. By construction, w^* is the minimizer of $L_{\mathrm{PBO}}(\,\cdot\,;\mathbf{1})$ regardless of whether it is a critical point of the original loss. The minimizer of $L_{\mathrm{PBO}}(\,\cdot\,;\beta)$ for nearby β defines the *proximal Bregman response function* (PBRF).

When one re-derives the influence function formula starting from L_{PBO} instead of L, the result is

$$\mathcal{I}_{\mathrm{PBRF}}(z_i, \phi) \;=\; -\,\nabla_w \phi(w^*)^\top\,(G + \lambda I)^{-1}\,\nabla_w L(w^*, z_i), \tag{6}$$

the same expression that the modern literature uses. This is effectively the counterfactual it approximates. As Bae et al. (2022) put it: *if influence functions are the answer, the question is the PBRF, not LOO retraining.* We refer to their paper for a detailed decomposition of the remaining gap between the PBRF and the original LOO counterfactual.

Remark 2.6 (Connection to the SLT day). A central obstacle to the application of influence functions to modern neural networks is *degeneracy*: the set of minimizers of an overparameterized model is singular and the Hessian has a large null space. In Section 3 we will see a degeneracy-aware alternative: Bayesian influence functions, which replace inverse-Hessian computations with covariances over a localized posterior.

Remark 2.7 (Scaling and EK-FAC). Even with the PBRF reframing, computing $(G+\lambda I)^{-1}$ for a billion-parameter model is not directly feasible. Grosse et al. (2023) scale influence functions to large language models by approximating G with EK-FAC (a Kronecker-factored approximation to layerwise curvature).

Exercise 2.8 (Failure of the first-order approximation in a degenerate model)**.** Consider the toy two-parameter model from the SLT lecture day in which $w = (a, b)$ enters the loss only through the product $u = ab$, with squared loss $L(w, z) = \frac{1}{2}(z - ab)^2$ on training data $\{z_1, \ldots, z_n\} \subset \mathbb{R}$.

(a) Write down the Hessian of the training loss at any minimizer (a^*, b^*) with $a^*b^* = \bar{z} := \frac{1}{n}\sum_i z_i$. Show that it has rank one, and identify the one-dimensional null space geometrically.

(b) Compute the influence function $\mathcal{I}(z_i, \phi)$ for the observable $\phi(w) = ab$, treating H^{-1} as a Moore–Penrose pseudoinverse[a]. How does it compare with the exact LOO change in ϕ?

(c) Repeat (b) using the damped inverse $(H + \lambda I)^{-1}$ for a small $\lambda > 0$. Discuss how the answer depends on λ, on the choice of minimizer (a^*, b^*) along the singular set, and on n. What does this say about the proximity gap?

[a] Concretely you can take the eigedecomposition of H and only invert the non-zero eigenvalues

3 Bayesian Influence Functions

Classical influence functions give a first-order approximation to the change in a *point estimate* w^* when the training data is perturbed. The Bayesian reformulation replaces this point estimate by a *distribution* over parameters: if we perturb the training data, how does our posterior over the parameter space change? This leads naturally to the Bayesian generalization of influence functions introduced in Kreer et al. (2025). In the setting of Equation (1) we set $\pi(\beta)$ not to be a single weight, but a whole distribution over parameters p_β, namely the posterior.

Susceptibilities. The Bayesian influence function is one instance of a more general object studied by Baker et al. (2025). Given a tempered Gibbs posterior $p_\beta \propto e^{-\beta L(w)}\varphi(w)$ at inverse temperature β, any pair of observables $\phi, \psi : W \to \mathbb{R}$ defines a *susceptibility*

$$\chi(\phi \,|\, \psi) \;:=\; \frac{\partial}{\partial h}\, \mathbb{E}_{p_h}[\phi(w)]\Big|_{h=0}, \qquad p_h \;\propto\; e^{-\beta\left(L(w) - h\,\psi(w)\right)}\,\varphi(w). \tag{7}$$

The differentiation argument we will apply in Exercise 3.2 shows $\chi(\phi \,|\, \psi) = \beta\,\mathrm{Cov}_{p_\beta}(\phi, \psi)$ in full generality. Taking $\psi = -L(\cdot, z_i)$ gives the Bayesian influence function developed below. Other choices of (ϕ, ψ) yield other sensitivity notions, see e.g. the refined learning coefficients (Baker et al., 2025, Appendix D.2). We will not need the general framework again; we mention it only to locate the BIF within the wider susceptibility family.

3.1 Bayesian influence functions

We now specialize the susceptibility story to data attribution. Instead of asking "how does $\phi(w^*)$ change when we perturb the training weights?" (Definition 2.2) we ask "how does $\mathbb{E}_{w\sim p}[\phi(w)]$ change?" The answer turns out to be remarkably clean: the derivative becomes a covariance, and the Hessian inversion of the classical formula is replaced by posterior sampling.

The tempered posterior. Given training data $\mathcal{D} = \{z_1, \ldots, z_n\}$ with per-sample losses $L(w, z_i)$, sample weights $\beta = (\beta_1, \ldots, \beta_n)$, and a prior $\varphi(w)$, define the *tempered posterior*

$$p_\beta(w \mid \mathcal{D}) \;\propto\; \exp\Big(-\sum_{i=1}^{n} \beta_i\, L(w, z_i)\Big)\,\varphi(w). \tag{8}$$

When $\beta = \mathbf{1}$ and the loss is a negative log-likelihood, this is the standard Bayesian posterior. When the losses are not log-likelihoods, p_β is a *Gibbs measure*; the mathematics is identical.

Definition 3.1 (Bayesian influence function)**.** The *Bayesian influence function* (BIF) of training example z_i on the observable ϕ is the derivative of the posterior expectation of ϕ with respect to the sample weight β_i:

$$\mathrm{BIF}(z_i, \phi) \;:=\; \frac{\partial}{\partial \beta_i}\, \mathbb{E}_{w\sim p_\beta}[\phi(w)]\bigg|_{\beta=\mathbf{1}}.$$

The definition is the natural Bayesian analogue of Definition 2.2: replace $\phi(w^*(\beta))$ with $\mathbb{E}_{p_\beta}[\phi(w)]$. The following exercise shows that this derivative has a clean closed form. Recall that for two real-valued random variables X and Y defined on the same probability space, the *covariance* is

$$\mathrm{Cov}(X, Y) \;=\; \mathbb{E}[XY] - \mathbb{E}[X]\,\mathbb{E}[Y].$$

When X and Y are functions of the random parameter $w \sim p$, we write $\mathrm{Cov}_{w\sim p}(X(w), Y(w))$ to indicate that the expectation is taken over the posterior p. If ϕ is vector-valued ($\phi : W \to \mathbb{R}^d$), the covariance $\mathrm{Cov}(X, \phi)$ is the vector whose j-th entry is $\mathrm{Cov}(X, \phi_j)$.

Exercise 3.2 (The BIF is a covariance)**.** Let $p_\beta(w \mid \mathcal{D})$ be the tempered posterior of Equation (8) and write $p = p_{\mathbf{1}}$ for the unperturbed posterior. By differentiating $\mathbb{E}_{p_\beta}[\phi(w)] = \int \phi(w)\, p_\beta(w \mid \mathcal{D})\, dw$ with respect to β_i, show that

$$\boxed{\mathrm{BIF}(z_i, \phi) \;=\; -\,\mathrm{Cov}_{w\sim p}\big(L(w, z_i),\ \phi(w)\big).} \tag{9}$$

Hint: Write $p_\beta \propto e^{-\sum_j \beta_j L(w, z_j)}\varphi(w)$ and differentiate the ratio $\int \phi\, p_\beta\, dw \,/\, \int p_\beta\, dw$ using the quotient rule. The key identity is $\partial_{\beta_i} \log Z(\beta) = -\mathbb{E}_{p_\beta}[L(w, z_i)]$.

Remark 3.3 (Statistical physics). The identity $\partial_{\beta_i}\mathbb{E}[\phi] = -\mathrm{Cov}(L_i, \phi)$ is a standard fluctuation–response relation in statistical physics: the response of an observable to a change in an external field equals the covariance of that observable with the conjugate energy.

Localization. Computing expectations over the global posterior $p(w \mid \mathcal{D})$ is generally intractable for neural networks, and we are typically interested in the behavior of a specific trained checkpoint w^*, not a global Bayesian average. Following Kreer et al. (2025), we *localize* the posterior by adding a Gaussian penalty centered at w^*:

$$p_\gamma(w \mid \mathcal{D}, w^*) \;\propto\; \exp\Big(-\sum_{i=1}^{n} L(w, z_i) - \frac{\gamma}{2}\|w - w^*\|^2\Big), \tag{10}$$

where $\gamma > 0$ is a *localization strength* that controls how tightly the posterior concentrates around w^*. The *local BIF* is then

$$\mathrm{BIF}_\gamma(z_i, \phi) \;=\; -\,\mathrm{Cov}_\gamma\big(L(w, z_i),\ \phi(w)\big), \tag{11}$$

where Cov_γ denotes covariance under p_γ. We will see in Exercise 3.5 that the localization parameter γ plays exactly the role of the damping parameter λ in the PBRF formula (6).

Practical computation via SGLD. The local BIF requires estimating a covariance under p_γ. This is done by drawing samples from p_γ using stochastic gradient Langevin dynamics (SGLD), which adds calibrated Gaussian noise to SGD updates:

$$w_{t+1} \;=\; w_t - \frac{\epsilon}{2}\Big(\frac{n}{m}\sum_{k\in B_t} \nabla_w L(w_t, z_k) + \gamma(w_t - w^*)\Big) + \eta_t, \qquad \eta_t \sim \mathcal{N}(0, \epsilon\, I),$$

where B_t is a mini-batch of size m and ϵ is the step size. After a burn-in period, the iterates $\{w_t\}$ are approximately distributed according to p_γ, and the covariance in (11) is estimated by the sample covariance of $(L(w_t, z_i), \phi(w_t))$ across draws. Running multiple independent chains from w^* improves coverage.

Example 3.4 (Bayesian linear regression)**.** Take the linear regression setup of Exercise 2.4 with a Gaussian prior $w \sim \mathcal{N}(0, \tau^2 I)$ added. The posterior is conjugate: $w \mid \mathcal{D} \sim \mathcal{N}(\mu, V)$ with $V = (\tau^{-2} I + \sigma^{-2} A)^{-1}$ and $\mu = \sigma^{-2} V \sum_i x_i y_i$. A direct Gaussian-moment computation applied to (9) with the observable $\phi(w) = w$ gives

$$\mathrm{BIF}(z_j, w) \;=\; -\mathrm{Cov}_p\big(L(w, z_j),\, w\big) \;=\; \frac{V\, x_j\, r_j}{\sigma^2} \;=\; (A + \lambda I)^{-1}\, x_j\, r_j, \qquad \lambda := \sigma^2/\tau^2,$$

where $r_j = y_j - x_j^\top \mu$ is the posterior-mean residual. This is the *damped* influence function with the damping parameter λ equal to the ratio of observation noise to prior variance. In the flat-prior limit $\tau \to \infty$ we have $\lambda \to 0$, $\mu \to w^*$, and we recover the classical IF of Exercise 2.4.

3.2 Exercise: Connecting Bayesian and classical influence functions

As we have seen in the SLT day, tools involving the Hessian are usually a low-order approximation of an expansion. This is also true for influence functions: In the non-degenerate case the classical IF is the leading-order term of the BIF under a Laplace approximation. The following exercise makes this precise by computing a power series expansion of the BIF covariance and identifying the classical IF as the leading-order term. The argument follows Kreer et al. (2025, Appendix A).

Exercise 3.5 (Power series expansion of Bayesian influence functions)**.** Let w^* be a local minimum of the training loss $L(w) = \sum_{i=1}^n L(w, z_i)$ with positive-definite Hessian $H = \nabla_w^2 L(w^*)$. Write $\Delta w = w - w^*$, and abbreviate $g_\phi = \nabla_w \phi(w^*)$, $H_\phi = \nabla_w^2 \phi(w^*)$ for the gradient and Hessian of the observable, and $g_i = \nabla_w L(w^*, z_i)$, $H_i = \nabla_w^2 L(w^*, z_i)$ for the per-sample loss.

(a) **(Laplace approximation.)** Consider the posterior $p(w \mid \mathcal{D}) \propto e^{-L(w)}\, \varphi(w)$. Taylor-expand $L(w)$ to second order around w^*, using the fact that $\nabla L(w^*) = 0$ at a minimum. Argue that for large n the quadratic term in L dominates the prior φ, and conclude that

$$p(w \mid \mathcal{D}) \;\approx\; \mathcal{N}(w^*,\; H^{-1}).$$

Under this approximation, $\Delta w \sim \mathcal{N}(0, H^{-1})$. (This is called the *Laplace approximation*; the Bernstein–von Mises theorem guarantees it is asymptotically exact for non-singular models.)

(b) **(Taylor expansion of the covariance.)** Expand $\phi(w)$ and $L(w, z_i)$ in Taylor series around w^*:

$$\begin{aligned}
\phi(w) &= \phi(w^*) + g_\phi^\top \Delta w + \tfrac{1}{2} \Delta w^\top H_\phi\, \Delta w + \cdots, \\
L(w, z_i) &= L(w^*, z_i) + g_i^\top \Delta w + \tfrac{1}{2} \Delta w^\top H_i\, \Delta w + \cdots.
\end{aligned}$$

Using the fact that constant terms drop out of covariances and that $\mathrm{Cov}(X, Y) = \sum_{k,m \geq 1} \mathrm{Cov}(T_k[\phi],\, T_m[L_i])$ where $T_k[f]$ denotes the k-th order term in the Taylor expansion of f, show that the BIF decomposes as

$$\mathrm{BIF}(z_i, \phi) = -\sum_{\substack{k,m \geq 1 \\ k+m \text{ even}}} \mathrm{Cov}_{\mathcal{N}}\big(T_k[\phi],\; T_m[L_i]\big).$$

Why do terms with $k + m$ odd vanish?

(c) **(Leading order: classical IF.)** Compute the $(k, m) = (1, 1)$ term:

$$\mathrm{Cov}_{\mathcal{N}}\big(g_\phi^\top \Delta w,\ g_i^\top \Delta w\big) \ = \ g_\phi^\top H^{-1} g_i.$$

Conclude that $-g_\phi^\top H^{-1} g_i \ = \ -\nabla_w \phi(w^*)^\top H^{-1} \nabla_w L(w^*, z_i)$ is the classical influence function $\mathcal{I}(z_i, \phi)$ from Definition 2.2. This is the leading-order term of the BIF.

(d) **(Second-order correction.)** Compute the $(k, m) = (2, 2)$ term using Isserlis' theorem (the Gaussian moment identity $\mathbb{E}[\Delta w_a \Delta w_b \Delta w_c \Delta w_d] = \Sigma_{ab}\Sigma_{cd} + \Sigma_{ac}\Sigma_{bd} + \Sigma_{ad}\Sigma_{bc}$ with $\Sigma = H^{-1}$). Show that

$$\mathrm{Cov}_{\mathcal{N}}\Big(\tfrac{1}{2}\Delta w^\top H_\phi\, \Delta w,\ \tfrac{1}{2}\Delta w^\top H_i\, \Delta w\Big) = \tfrac{1}{2}\ \mathrm{tr}\big(H_\phi\, H^{-1}\, H_i\, H^{-1}\big).$$

This correction involves the *Hessians* of the observable and per-sample loss. It captures second-order curvature interactions that the classical IF misses entirely. In the linear regression setting of Example 3.4, why does this correction vanish?

(e) **(Localized version: damped IF.)** Now consider the local BIF of Equation (11) with localization strength γ. The localized posterior is approximately $\mathcal{N}(w^*, (H + \gamma I)^{-1})$. Repeat the leading-order computation of part (c) to show that

$$\mathrm{BIF}_\gamma(z_i, \phi) \ \approx \ -\nabla_w \phi(w^*)^\top\, (H + \gamma I)^{-1}\, \nabla_w L(w^*, z_i).$$

This is precisely the damped influence function of Equation (6), with the localization strength γ playing the role of the damping parameter λ. The local BIF is thus a natural, higher-order generalization of the damped IF: it agrees at leading order and includes all the corrections from parts (b)–(d) with H^{-1} replaced by $(H + \gamma I)^{-1}$.

Remark 3.6. We conclude that the classical (damped) IF is the leading term of the (local) BIF under a Laplace approximation. For non-singular models with enough data, the Laplace approximation converges and the higher-order terms are small. But for singular models (neural networks), the Laplace approximation fails, the posterior is not Gaussian, and the BIF captures geometry that the classical IF cannot see.

3.3 Long Exercise: Influence functions as optimal linear transport

The previous exercise showed that the classical IF is the leading-order term of the BIF under a Laplace approximation. We now take a different perspective, due to Mlodozeniec et al. (2025): instead of asking "what does the BIF reduce to?" we ask "what is the *best* way to shift the parameters to approximate the effect of a data perturbation?" The answer turns out to be: the influence function, and its optimality does not require the Laplace approximation to hold.

Setup. Let $L(w) = \sum_{i=1}^n L(w, z_i)$ denote the total training loss, and abbreviate $L_k(w) := L(w, z_k)$ for the per-sample loss at example k. Define the *Boltzmann distributions* at inverse temperature β:

$$p_0^\beta(w) \propto e^{-\beta\, L(w)}, \qquad\qquad p_\varepsilon^\beta(w) \propto e^{-\beta\left(L(w) - \varepsilon\, L_k(w)\right)},$$

so that p_ε^β is the Boltzmann distribution after downweighting sample k by ε. The low-temperature limit $\beta \to \infty$ concentrates p_0^β on the set of global minima $\mathcal{S}_L = \{w : L(w) = \inf L\}$.

Now suppose we want to approximate the effect of the perturbation without resampling from p_ε^β. The simplest approach is to take each parameter w drawn from p_0^β and *shift* it to $w + \varepsilon\, r(w)$

for some smooth vector field $r : \mathbb{R}^d \to \mathbb{R}^d$. If r is chosen well, the distribution of the shifted parameters should be close to p_ε^β. The question is: what is the optimal r?

To make this precise, we need to know the density of the shifted parameters. If $w \sim p_0^\beta$ and we set $\tilde{w} = w + \varepsilon\, r(w)$, then by the multivariate change-of-variables formula, $\tilde{w}$ has density

$$q_\varepsilon(\tilde{w}) \;=\; \frac{p_0^\beta\big(\tilde{w} - \varepsilon\, r(\tilde{w}) + O(\varepsilon^2)\big)}{\big|\det\big(I + \varepsilon\, \nabla r(w)\big)\big|}\,, \tag{12}$$

where ∇r is the $d \times d$ Jacobian matrix of r and the denominator is the usual volume-change factor.[2] We measure how well q_ε approximates p_ε^β by the KL divergence $D_{\mathrm{KL}}(q_\varepsilon \| p_\varepsilon^\beta)$. Rather than computing this directly, it is cleaner to write it as an expectation over the *original* distribution p_0^β:

$$D_{\mathrm{KL}}(q_\varepsilon \| p_\varepsilon^\beta) \;=\; \mathbb{E}_{w\sim p_0^\beta}\Big[\log p_0^\beta(w) - \log p_\varepsilon^\beta\big(w + \varepsilon r(w)\big) - \log\big|\det\big(I + \varepsilon \nabla r(w)\big)\big|\Big]. \tag{13}$$

This identity follows by substituting (12) into the definition of KL and changing variables back to w. We define the *asymptotic KL cost*

$$\mathcal{F}(r, \varepsilon) \;:=\; \lim_{\beta\to\infty} \frac{1}{\beta}\, D_{\mathrm{KL}}(q_\varepsilon \| p_\varepsilon^\beta). \tag{14}$$

The $1/\beta$ normalization kills the determinant term (which is $O(1)$) and the entropy terms, isolating the energy contribution (which is $O(\beta)$).

Exercise 3.7 (Influence functions as optimal parameter shifts)**.** Let w^* be a non-degenerate minimum of L with positive-definite Hessian $H = \nabla^2 L(w^*)$. In this exercise we work at the unique minimum, so the $\beta \to \infty$ limit simply evaluates everything at w^*.

(a) **(KL in terms of energies.)** Starting from (13), substitute $p_0^\beta(w) = e^{-\beta L(w)}/Z(\beta, 0)$ and $p_\varepsilon^\beta(w) = e^{-\beta(L(w) - \varepsilon L_k(w))}/Z(\beta, \varepsilon)$, and show that

$$\begin{aligned}\frac{1}{\beta}\, D_{\mathrm{KL}} \;&= \mathbb{E}_{p_0^\beta}\big[L(w + \varepsilon r) - \varepsilon\, L_k(w + \varepsilon r) - L(w)\big] \\ &\quad +\; \frac{1}{\beta}\log\frac{Z(\beta, \varepsilon)}{Z(\beta, 0)} \;-\; \frac{1}{\beta}\,\mathbb{E}_{p_0^\beta}\big[\log|\det(I + \varepsilon \nabla r)|\big],\end{aligned}$$

where $Z(\beta, \varepsilon) = \int e^{-\beta(L(w) - \varepsilon L_k(w))}\, dw$. Explain why the last two terms vanish in the $\beta \to \infty$ limit. *Hint:* $\log|\det(I + \varepsilon \nabla r)| = O(1)$, and by Laplace's method $\frac{1}{\beta}\log Z(\beta, \varepsilon) \to -\inf_w[L(w) - \varepsilon L_k(w)]$.

(b) **(Taylor expansion at w^*.)** Since the expectation under p_0^β concentrates at w^* as $\beta \to \infty$, expand $L(w^* + \varepsilon r)$ and $\varepsilon\, L_k(w^* + \varepsilon r)$ using $\nabla L(w^*) = 0$:

$$\begin{aligned}L(w^* + \varepsilon r) &= L(w^*) + \tfrac{\varepsilon^2}{2}\, r^\top H\, r + O(\varepsilon^3), \\ \varepsilon\, L_k(w^* + \varepsilon r) &= \varepsilon\, L_k(w^*) + \varepsilon^2\, \nabla L_k(w^*)^\top r + O(\varepsilon^3).\end{aligned}$$

Similarly, show that $\inf_w[L(w) - \varepsilon L_k(w)] = L(w^*) - \varepsilon\, L_k(w^*) - \frac{\varepsilon^2}{2}\, \nabla L_k^\top H^{-1} \nabla L_k + O(\varepsilon^3)$, by minimizing the Taylor expansion of $L - \varepsilon L_k$ around w^*. Combine everything to obtain

$$\mathcal{F}(r, \varepsilon) \;=\; \frac{\varepsilon^2}{2}\,\big(r - H^{-1}\nabla L_k\big)^\top H\,\big(r - H^{-1}\nabla L_k\big) \;+\; O(\varepsilon^3). \tag{15}$$

Hint: After substitution, the terms that do not depend on r should assemble into $\frac{\varepsilon^2}{2}\nabla L_k^\top H^{-1}\nabla L_k$ and cancel with the partition-function contribution, leaving a perfect square.

[2] For small ε the map $w \mapsto w + \varepsilon r(w)$ is a diffeomorphism, so the inverse is well-defined. The exact formula requires evaluating r at the pre-image; the expression above is correct to the order we need.

(c) **(IFs are optimal.)** The cost (15) is a squared Mahalanobis distance (in the Hessian metric) between the shift r and the influence function $r_{\mathrm{IF}} := H^{-1}\nabla L_k(w^*)$. Conclude that the IF minimizes $\mathcal{F}$ to $O(\varepsilon^2)$ among *all* smooth shift maps $w \mapsto w + \varepsilon r(w)$, and that the minimum cost is $O(\varepsilon^3)$.

State the result in words: *the influence function is the approximately KL-optimal way to shift the parameters of the unperturbed Boltzmann distribution to approximate the perturbed one.*

(d) **(Degenerate case.)** Now suppose w^* lies on a minimum manifold $\mathcal{S}_L$ and H is singular with null space N. Argue (without detailed proof) that:

- The cost (15) generalizes to $\mathcal{F}(r, \varepsilon) = \frac{\varepsilon^2}{2}(r - H^+\nabla L_k)^\top H\,(r - H^+\nabla L_k) + O(\varepsilon^3)$, where H^+ is the Moore–Penrose pseudoinverse.
- The minimizer is $r = H^+\nabla L_k + r_{\mathrm{NS}}$ for any $r_{\mathrm{NS}} \in N$: directions in the null space of H have zero cost, because the Hessian assigns zero curvature to movement along the minimum manifold.
- The IF (with pseudoinverse) is therefore optimal for the directions that matter, but the component of the shift along the flat directions is unconstrained.

Connect this to the SLT perspective: for singular models, the minimum is a variety, and the IF tells you how to move *off* the variety but is silent about movement *along* it.

(e) **(Linear regression check.)** Specialize to the Bayesian linear regression setting of Example 3.4 with prior $w \sim \mathcal{N}(0, \tau^2 I)$. The Boltzmann distribution at inverse temperature $\beta = 1$ is the Bayesian posterior $\mathcal{N}(\mu, V)$, which is already Gaussian, no need to take $\beta \to \infty$. A constant shift $w \mapsto w + \varepsilon r$ sends $\mathcal{N}(\mu, V)$ to $\mathcal{N}(\mu + \varepsilon r,\, V)$. The true perturbed posterior (downweighting z_k by ε) has mean $\mu + \varepsilon\,\mathrm{BIF}(z_k, w) + O(\varepsilon^2)$ and covariance $V + O(\varepsilon)$. Using the KL formula for Gaussians with the same covariance, show that

$$D_{\mathrm{KL}}\big(\mathcal{N}(\mu + \varepsilon r,\, V) \,\big\|\, \mathcal{N}(\mu + \varepsilon b,\, V)\big) \;=\; \frac{\varepsilon^2}{2}\,(r - b)^\top V^{-1}(r - b),$$

which is minimized at $r = b = \mathrm{BIF}(z_k, w) = (A + \lambda I)^{-1} x_k r_k$. Verify that this is consistent with (15): the "Hessian of the energy" (training loss plus prior) is $V^{-1} = \sigma^{-2}A + \tau^{-2}I$, and $V^{-1} \cdot \mathrm{BIF} = \sigma^{-2} x_k r_k = \nabla_w L(w, z_k)\big|_{w=\mu}$.

(f) **(Comparison with Exercise 3.5.)** The Laplace expansion exercise and this exercise both relate the IF to the BIF, but they answer different questions:

- Exercise 3.5 is an *algebraic identity*: BIF = IF + higher-order corrections. It requires the Laplace approximation (hence non-singular H, Gaussian posterior) and tells you what happens when you truncate the BIF.
- This exercise is an *optimality result*: the IF shift minimizes the KL divergence between the shifted and true perturbed distributions. The $\beta \to \infty$ result (parts a–d) needs only mild regularity of $\mathcal{L}$ and works even for singular H (via the pseudoinverse).

Summarize: the Laplace expansion tells you *what* the IF is (the leading Gaussian term of the BIF). The optimality perspective tells you *why* it works (it is the best first-order correction to the parameter distribution). The second viewpoint does not require the posterior to be Gaussian, which is why Mlodozeniec et al. (2025) argue that it provides a better explanation for the empirical success of IFs in deep learning.

Remark 3.8 (Optimal transport). In the language of optimal transport, the map $w \mapsto w + \varepsilon r(w)$ is called a *transport map*, and the distribution of the shifted parameters is its *pushforward*, written $T_{\varepsilon\#}p_0^\beta$. The cost functional $\mathcal{F}$ is the asymptotic KL divergence between the pushforward and the target. Exercise 3.7 then says that the influence function is the *KL-optimal transport map* from the unperturbed to the perturbed Boltzmann distribution, at first order in ε. This is the perspective developed in Mlodozeniec et al. (2025), who prove a more general version (their Theorem 3) that applies to all smooth diffeomorphisms, not just maps of the form $w + \varepsilon r(w)$.

4 Unrolling

The influence functions of Section 2 and the Bayesian influence functions of Section 3 both have something in common: they analyze the *endpoint* of training, not the *dynamic* that lead to it. Under the assumptions of the unique global minimum, this is not a problem: the training trajectory is irrelevant, since all roads lead to the same outcome. But in practice the training path will matter a lot. Ultimately, we are studying generalization and this is ultimately a question about the training dynamics. Similarly, the BIF looks at the posterior expectation shifts. To give a concrete failure mode: a training point that appears in the first mini-batch of the first epoch will have a very different effect compared to the same point appearing towards the end of training.

If we write down the training trajectory as a composition of β-dependent maps:

$$\begin{array}{ccccccccc} \mathbb{R}^p & \xrightarrow{h_0(\cdot\,;\beta)} & \mathbb{R}^p & \xrightarrow{h_1(\cdot\,;\beta)} & \cdots & \xrightarrow{h_{T-1}(\cdot\,;\beta)} & \mathbb{R}^p & \xrightarrow{\phi} & \mathbb{R} \\ w_0 & \longmapsto & w_1 & \longmapsto & \cdots & \longmapsto & w_T & \longmapsto & \phi(w_T) \end{array}$$

where each h_t is one optimizer step (a gradient step, an Adam update, etc.): $w_{t+1} = h_t(w_t; \beta)$, we can observe an analogy to a forward pass through a neural network:

$$\begin{array}{ccccccccc} \mathbb{R}^{d_0} & \xrightarrow{f_1(\cdot\,;w)} & \mathbb{R}^{d_1} & \xrightarrow{f_2(\cdot\,;w)} & \cdots & \xrightarrow{f_L(\cdot\,;w)} & \mathbb{R}^{d_L} & \xrightarrow{\ell} & \mathbb{R} \\ a_0 & \longmapsto & a_1 & \longmapsto & \cdots & \longmapsto & a_L & \longmapsto & \ell(a_L) \end{array}$$

where each f_ℓ is one layer: $a_\ell = f_\ell(a_{\ell-1}; w)$.

	Forward pass	**Training trajectory**
Inputs	Model weights w	Data weights β
Intermediate states	Activations $a_1, \ldots, a_L$	Parameters $w_0, \ldots, w_T$
Output	Loss ℓ(logits)	Observable $\phi(w_T)$
Differentiation method	Backpropagation	Unrolling

Table 1: Data attribution via unrolling is to a training run what backprop is to a forward pass: differentiating a composition of maps to identify which inputs matter most.

This invites the following question: Can we backprop through the *training trajectory*? The gradients with respect to the β_i then indicate which data points would need to be up- or down-weighted to change the final observable $\phi(w_T)$. And in principle, we can! We can differentiate through the training computation, step by step, treating the final parameters as an explicit function of every gradient update along the way. The result is a chain-rule expression that decomposes data attribution into per-step contributions and naturally incorporates the optimizer, learning rate schedule, mini-batch ordering, and training duration. We will derive the unrolling formula similar to the backprop formula for a forward pass, and then discuss two approaches to making it practical: one that materializes Jacobians and one that computes Jacobian-vector products implicitly. In the setup of Equation (1) we take $\pi(\beta) = w_T(\beta)$, the final checkpoint of the training run.

4.1 The training-dynamics approach to attribution

We fix notation for the training process. Given a dataset $\mathcal{D} = \{z_1, \ldots, z_n\}$ with sample weights $\beta = (\beta_1, \ldots, \beta_n)$, the optimizer runs for T steps. At step t, a mini-batch $B_t \subseteq \{1, \ldots, n\}$ of size b is drawn, and the parameters are updated by

$$w_{t+1}(\beta) \;=\; w_t(\beta) \;-\; \frac{\eta_t}{b} \sum_{i \in B_t} \beta_i \, \nabla_w L\big(w_t(\beta),\, z_i\big), \tag{16}$$

where η_t is the learning rate at step t. At $\beta = \mathbf{1}$ this is ordinary SGD on the unweighted loss. The initial parameters w_0 are independent of β. For simplicity we can assume that we are training for one epoch only, so that each example appears in exactly one mini-batch, but the formulas hold more generally.

The counterfactual. We want

$$\left.\frac{\partial\, \phi(w_T(\beta))}{\partial \beta_i}\right|_{\beta=\mathbf{1}},$$

the sensitivity of a final-model observable ϕ to an infinitesimal change in the weight of training example z_i.

4.2 The unrolling formula

Central to computing the data weight gradient is the Jacobian of one SGD step with respect to the current parameters. Define

$$J_t \;:=\; \left.\frac{\partial w_{t+1}}{\partial w_t}\right|_{\beta=\mathbf{1}} \;=\; I \;-\; \frac{\eta_t}{b} \sum_{i \in B_t} \nabla_w^2 L(w_t, z_i) \;=\; I - \eta_t \, \widehat{H}_t, \tag{17}$$

where $\widehat{H}_t := \frac{1}{b} \sum_{i \in B_t} \nabla_w^2 L(w_t, z_i)$ is the mini-batch Hessian at step t. The Jacobian of the map from step t to step t' is the ordered product

$$J_{t:t'} \;:=\; J_{t'-1}\, J_{t'-2} \cdots J_t \;=\; \prod_{s=t}^{t'-1} J_s \qquad (t < t'). \tag{18}$$

Similarly, the direct effect of β_i at step t (when $i \in B_t$) is the per-example gradient:

$$\left.\frac{\partial w_{t+1}}{\partial \beta_i}\right|_{\substack{\beta=\mathbf{1} \\ w_t \text{ fixed}}} \;=\; -\,\frac{\eta_t}{b}\, \nabla_w L(w_t, z_i) \cdot \mathbf{1}[i \in B_t]. \tag{19}$$

The full derivative is obtained by chaining direct effects through remaining steps:

$$\boxed{\left.\frac{\partial w_T(\beta)}{\partial \beta_i}\right|_{\beta=\mathbf{1}} \;=\; -\sum_{t=0}^{T-1} \frac{\eta_t}{b}\; \mathbf{1}[i \in B_t]\; J_{(t+1):T}\; \nabla_w L(w_t, z_i).} \tag{20}$$

Each term in the sum corresponds to one training step in which z_i appeared in the mini-batch: it is the gradient of z_i at the parameters of that step, propagated forward through the Jacobians of all subsequent steps.

The influence on an observable ϕ follows by the chain rule:

$$\left.\frac{\partial\, \phi(w_T(\beta))}{\partial \beta_i}\right|_{\beta=\mathbf{1}} \;=\; -\,\nabla_w \phi(w_T)^\top \sum_{t=0}^{T-1} \frac{\eta_t}{b}\; \mathbf{1}[i \in B_t]\; J_{(t+1):T}\; \nabla_w L(w_t, z_i). \tag{21}$$

Remark 4.1 (Which questions does unrolling answer?). Equation (20) looks like a sum over training steps and, for a fixed realization of the mini-batch sequence, it *is* a deterministic chain-rule computation. But the training run has randomness (initialization, mini-batch ordering, data augmentation), and so does the counterfactual: if we change β_i, the mini-batch sequence might be the same or it might differ. Unrolling as written differentiates through a *specific* training run: it answers the *single-model* attribution question, "how would this particular training trajectory have ended differently?" The *distributional* question, which looks at the expected change over random training, would require averaging (20). This distinction, which we will revisit in Section 5, is the training-dynamics analogue of the single-posterior-mode vs. full-posterior distinction in Section 3.

Exercise 4.2 (Derivation of the unrolling formula)**.** (a) Starting from the SGD update (16), apply the chain rule to write $\frac{\partial w_{t+1}}{\partial \beta_i}\big|_{\beta=\mathbf{1}}$ as a sum of two terms: one from the explicit dependence on β_i (the direct effect) and one from the dependence of w_t on β_i (the indirect effect). Show that the result is the recursion

$$\frac{\partial w_{t+1}}{\partial \beta_i} \;=\; J_t \,\frac{\partial w_t}{\partial \beta_i} \;-\; \frac{\eta_t}{b}\,\mathbf{1}[i \in B_t]\,\nabla_w L(w_t, z_i),$$

with initial condition $\frac{\partial w_0}{\partial \beta_i} = 0$. Compute the Jacobian J_t explicitly in terms of the mini-batch Hessian $\widehat{H}_t$ which leads to (17).

(b) Solve the recursion by "unrolling" it (i.e. by substituting repeatedly and using (18)) to obtain (20).

(c) **(Preconditioned SGD.)** Suppose instead that the optimizer uses a preconditioner $P_t \succ 0$:

$$w_{t+1}(\beta) \;=\; w_t(\beta) \;-\; \frac{\eta_t}{b}\,P_t \sum_{i \in B_t} \beta_i\,\nabla_w L(w_t(\beta), z_i). \tag{22}$$

This includes momentum-free Adam and natural gradient methods as special cases (with appropriate P_t). Show that the step Jacobian becomes $J_t^{(P)} = I - \eta_t\,P_t \widehat{H}_t$ and that the unrolling formula generalizes to

$$\frac{\partial w_T(\beta)}{\partial \beta_i}\bigg|_{\beta=\mathbf{1}} \;=\; -\sum_{t=0}^{T-1} \frac{\eta_t}{b}\,\mathbf{1}[i \in B_t]\; J^{(P)}_{(t+1):T}\; P_t\,\nabla_w L(w_t, z_i), \tag{23}$$

where $J^{(P)}_{(t+1):T} = \prod_{s=t+1}^{T-1}(I - \eta_s P_s \widehat{H}_s)$. The only change is that each gradient is premultiplied by the preconditioner at the step where it appears. What does this say about the effect of using Adam vs. SGD on the attribution of a data point that appears early in training?

Remark 4.3 (TracIn). Pruthi et al. (2020) observed that the Jacobian products $J_{(t+1):T}$ in (20) are both the most expensive and the most unstable part of the computation. Their method, TracIn, simply drops them, setting $J_{(t+1):T} \approx I$:

$$\mathrm{TracIn}(z_i, \phi) \;:=\; \sum_{t \in \mathcal{C}} \eta_t\,\nabla_w \phi(w_t)^\top \nabla_w L(w_t, z_i), \tag{24}$$

where $\mathcal{C}$ is a set of checkpoints (typically one per epoch).

There are two ways to make (20) practical. The first is to compute the Jacobian products $J_{(t+1):T}$ explicitly, or rather an approximation thereof. The second is to avoid materializing Jacobians altogether and instead compute Jacobian-vector products implicitly by reverse-mode automatic differentiation through the training loop. We will discuss both approaches in turn.

4.2.1 Materializing the Jacobian

In this subsection we will derive the SOURCE method (Bae et al., 2024), which introduces a small number of approximations of Jacobians that reduce the full unrolling computation to something resembling an influence-function calculation at each of several checkpoints.

Stationary segments. Divide the T training steps into L segments at checkpoints $0 = T_0 < T_1 < \cdots < T_L = T$, and let $K_\ell = T_\ell - T_{\ell-1}$ be the number of steps in segment ℓ. Within each segment, make the *stationarity approximation*: the expected mini-batch Hessian and per-example gradient are approximately constant,

$$\bar{H}_\ell \;\approx\; \frac{1}{K_\ell} \sum_{t=T_{\ell-1}}^{T_\ell - 1} \widehat{H}_t, \qquad \bar{g}_{i,\ell} \;\approx\; \frac{1}{K_\ell} \sum_{t=T_{\ell-1}}^{T_\ell - 1} \nabla_w L(w_t, z_i). \tag{25}$$

Similarly, write $\bar{\eta}_\ell$ for the average learning rate in segment ℓ.

Segment Jacobian. Under the stationarity approximation, the product of K_ℓ step Jacobians within segment ℓ simplifies to a matrix power:

$$J_{T_{\ell-1}:T_\ell} \;\approx\; (I - \bar{\eta}_\ell\, \bar{H}_\ell)^{K_\ell} \;\approx\; \exp\!\big(-\bar{\eta}_\ell\, K_\ell\, \bar{H}_\ell\big) \;=:\; \bar{S}_\ell. \tag{26}$$

The matrix exponential is well approximated by the matrix power when $\bar{\eta}_\ell\, \bar{H}_\ell$ has small spectral norm. In an eigenbasis of $\bar{H}_\ell$ with eigenvalue σ, the segment Jacobian acts as the scalar filter $F_S(\sigma) = e^{-\bar{\eta}_\ell K_\ell \sigma}$: high-curvature directions (σ large) are exponentially damped, while low-curvature directions (σ small) are preserved.

Segment response. Return to the unrolling formula (20) and restrict the sum to the steps inside segment ℓ. The contribution of datum z_i during this segment is

$$R_{i,\ell} \;:=\; \sum_{t=T_{\ell-1}}^{T_\ell - 1} \frac{\eta_t}{b}\, \mathbf{1}[i \in B_t]\; J_{(t+1):T_\ell}\; \nabla_w L(w_t, z_i), \tag{27}$$

where the Jacobian $J_{(t+1):T_\ell}$ propagates each gradient contribution forward to the segment boundary T_ℓ (propagation from T_ℓ to the end of training is handled by the inter-segment Jacobians $\bar{S}_{\ell'}$ in the full formula below). Now apply the stationarity approximation: replace each $\nabla_w L(w_t, z_i)$ by $\bar{g}_{i,\ell}$, each learning rate by $\bar{\eta}_\ell$, replace the indicator $\mathbf{1}[i \in B_t]$ by its expectation b/n (datum z_i appears in a random batch with probability b/n), and use the segment Jacobian approximation $J_{(t+1):T_\ell} \approx (I - \bar{\eta}_\ell \bar{H}_\ell)^{T_\ell - 1 - t}$. This gives

$$R_{i,\ell} \;\approx\; \frac{\bar{\eta}_\ell}{n} \sum_{k=0}^{K_\ell - 1} (I - \bar{\eta}_\ell\, \bar{H}_\ell)^k\; \bar{g}_{i,\ell}.$$

The sum is a matrix geometric series. Using $\sum_{k=0}^{K-1} M^k = (I - M^K)(I - M)^{-1}$ with $M = I - \bar{\eta}_\ell \bar{H}_\ell$ and passing to the matrix exponential:

$$\bar{r}_{i,\ell} \;:=\; \frac{1}{n}\big(I - e^{-\bar{\eta}_\ell K_\ell \bar{H}_\ell}\big)\, \bar{H}_\ell^{-1}\, \bar{g}_{i,\ell}. \tag{28}$$

To understand this, work in an eigenbasis of $\bar{H}_\ell$. On an eigenvalue σ, the matrix $(I - e^{-\bar{\eta}_\ell K_\ell \bar{H}_\ell})\bar{H}_\ell^{-1}$ acts as the scalar

$$F_r(\sigma) \;=\; \frac{1 - e^{-\bar{\eta}_\ell K_\ell\, \sigma}}{\sigma}. \tag{29}$$

Note the similarity to the classical influence function formula, which applies $1/\sigma$ (the inverse Hessian eigenvalue), and with the damped IF (6), which applies $1/(\sigma + \lambda)$. Thus F_r can be seen as a principled damping constant:

- *Large* σ (directions the optimizer has fully converged along): $e^{-\bar{\eta}_\ell K_\ell \sigma} \approx 0$, so $F_r(\sigma) \approx 1/\sigma$. Agrees with the classical IF.
- *Small* σ (directions the optimizer has barely moved along): $e^{-\bar{\eta}_\ell K_\ell \sigma} \approx 1 - \bar{\eta}_\ell K_\ell \sigma$, so $F_r(\sigma) \approx \bar{\eta}_\ell K_\ell$. A finite constant, not $1/\sigma \to \infty$.
- *Crossover* at $\sigma_* \approx 1/(\bar{\eta}_\ell K_\ell)$: directions with curvature below this threshold are automatically suppressed.

Full formula. Chaining the segment Jacobians and responses across all L segments:

$$\tau_{\text{SOURCE}}(\phi, z_i) \;=\; \nabla_w \phi(w_T)^\top \sum_{\ell=1}^{L} \Big(\prod_{\ell'=\ell+1}^{L} \bar{S}_{\ell'} \Big) \bar{r}_{i,\ell}, \tag{30}$$

where the product is ordered with $\ell' = L$ on the left. With a single segment ($L = 1$), the formula reduces to $\nabla_w \phi^\top \bar{r}_{i,1}$, which is the classical IF with H^{-1} replaced by the filter F_r.

Remark 4.4 (Unrolling provides a principled damping). The crossover $\sigma_* = 1/(\bar{\eta}K)$ in the filter F_r plays exactly the role of the damping parameter λ in the PBRF formula $(H + \lambda I)^{-1}$ from Section 2.2, but it is not a free parameter. It is the reciprocal of $\bar{\eta}K$, the total *training effort* (learning rate $\times$ number of steps) in the segment. Directions that the optimizer has not had time to converge along are automatically down-weighted, because the Jacobian products have not had enough steps to amplify them. The practical consequence: the damping λ that influence-function methods require careful tuning of has a natural value determined by the training process, and unrolling recovers it without any tuning.

4.2.2 Implicit JVPs and the REPLAY algorithm

The second approach to evaluating (20) avoids materializing Jacobians altogether. Instead, it uses *reverse-mode automatic differentiation* through the training loop, computing Jacobian-vector products (JVPs) implicitly. This is the strategy of the MAGIC method (Ilyas and Engstrom, 2025), building on the metagradient framework of Engstrom et al. (2025).

Setup. Model the entire training process as a composition of T differentiable update functions:

$$s_{t+1} = h_t\big(s_t,\, g_t(s_t, \beta)\big), \qquad s_0 = s_{\text{init}}, \tag{31}$$

where s_t is the full optimizer state at step t (parameters, momentum buffers, etc.), $g_t(s_t, \beta) = \sum_{i \in B_t} \beta_i \nabla L(s_t, z_i)$ is the weighted mini-batch gradient, and h_t is the optimizer's update rule (SGD, Adam, etc.). The attribution target is $\phi(s_T)$, which depends on β through the entire chain of updates.

The metagradient decomposition. Applying the chain rule through this composition:

$$\frac{\partial\, \phi(s_T)}{\partial \beta} \;=\; \sum_{t=0}^{T-1} \underbrace{\frac{\partial\, \phi(s_T)}{\partial s_{t+1}}}_{A_{t+1}} \cdot \underbrace{\frac{\partial\, h_t(s_t, g_t(s_t, \beta))}{\partial \beta}}_{B_t}. \tag{32}$$

The vector $A_{t+1} := \partial \phi(s_T)/\partial s_{t+1}$ is the sensitivity of the final observable to the state at step $t+1$; it satisfies the backward recursion

$$A_t \;=\; A_{t+1}\, \frac{\partial\, h_t(s_t, g_t)}{\partial s_t}, \qquad A_T = \nabla_{s_T} \phi(s_T). \tag{33}$$

This is just backpropagation through the training loop. The vector B_t is the direct effect of β at step t: it captures how the gradient at step t depends on the data weights. For vanilla SGD, B_t reduces to $-(\eta_t/b)\,[\nabla_w L(w_t, z_i) \cdot \mathbf{1}[i \in B_t]]_{i=1}^n$.

REPLAY: efficient reverse-mode through training. The backward recursion (33) requires the optimizer state s_t at every step. Naively this means storing all T states, which is prohibitive for long training runs. The REPLAY algorithm (Engstrom et al., 2025) solves this with a *hierarchical checkpointing* scheme:

1. Save k evenly-spaced checkpoints along the training trajectory.
2. When the backward pass needs a state between two checkpoints, *replay* the training forward from the nearest saved checkpoint to regenerate it.
3. Apply this strategy recursively (a k-ary tree of depth $\log_k T$).

The result is exact (to floating-point precision) differentiation through all T training steps, using $O(k \log_k T)$ stored states and $O(T \log_k T)$ total forward-pass work, a logarithmic overhead over the cost of training itself.

Comparison with explicit unrolling. The implicit approach has two main advantages:

- *Exactness.* No stationarity or matrix-exponential approximations are needed; the Jacobian-vector products are computed exactly by autodiff.
- *Generality.* Any differentiable optimizer (Adam, LAMB, etc.) and any training pipeline (multi-stage, curriculum learning) are handled transparently, since h_t is just a function.

The price is computational: REPLAY requires $O(T \log T)$ training-equivalent steps and careful engineering of the checkpointing schedule, whereas the explicit approach requires only a handful of checkpoints and an EK-FAC-style Hessian approximation at each. The choice depends on the scale: for moderate-size models where replaying training is feasible, MAGIC gives exact answers; for large-scale models where even one extra training pass is expensive, the SOURCE approximation with $L = 3$–6 segments may be the only practical option.

4.3 Long Exercise: From unrolling to influence functions

The unrolling formula (20) and the influence function (6) appear to be very different objects: one is a sum over training steps of Jacobian-propagated gradients, the other is a single inverse-Hessian-times-gradient computation at the endpoint. The following exercise shows that, in the right limit, unrolling converges to the classical influence function. The key observation, due to Mlodozeniec et al. (2025), is that the joint process (w_t, r_t), where $r_t := \partial w_t / \partial \varepsilon\big|_{\varepsilon=0}$ is the unrolling response , forms a *Markov chain*, and its limiting behavior can be analyzed with standard stochastic-approximation tools. This yields a convergence result that requires only that SGD reaches a local minimum, *not* that the loss is convex.

Exercise 4.5 (Convergence of unrolling to influence functions)**.** Consider the SGD update with an interpolated loss that downweights example z_k by ε:

$$w_{t+1}(\varepsilon) \;=\; w_t(\varepsilon) \;-\; \frac{\eta_t}{b} \sum_{i \in B_t} \big(1 - \varepsilon\, \mathbf{1}[i = k]\big)\, \nabla_w L(w_t(\varepsilon), z_i), \tag{34}$$

so $\varepsilon = 0$ is the original training run and $\varepsilon = 1/n$ corresponds to removing z_k. Define the *response* $r_t := \partial w_t(\varepsilon)/\partial \varepsilon\big|_{\varepsilon=0}$.

$$\begin{pmatrix} w_{t+1} \\ r_{t+1} \end{pmatrix} \;=\; \underbrace{\begin{pmatrix} I & 0 \\ 0 & J_t \end{pmatrix}}_{\text{propagation}} \begin{pmatrix} w_t \\ r_t \end{pmatrix} \;+\; \underbrace{\begin{pmatrix} -\frac{\eta_t}{b} \sum_{i \in B_t} \nabla_w L(w_t, z_i) \\ \frac{\eta_t}{b}\, \mathbf{1}[k \in B_t]\, \nabla_w L(w_t, z_k) \end{pmatrix}}_{\text{driving terms}} \tag{35}$$

(a) **(The response recursion.)** By differentiating (34) with respect to ε at $\varepsilon = 0$, derive (35), where $J_t = I - \frac{\eta_t}{b} \sum_{i \in B_t} \nabla_w^2 L(w_t, z_i)$ is the step Jacobian from (17). The top row is ordinary SGD; verify that the bottom row is the same recursion as Exercise 4.2(a), but now with stochastic batches. Since the driving terms depend only on (w_t, r_t) and the i.i.d. batch selection B_t, the joint process is a Markov chain. Why is this observation useful?

(b) **(Deterministic warm-up: full-batch GD.)** As a warm-up, consider full-batch gradient descent with constant learning rate η (i.e. $B_t = \{1, \ldots, n\}$ for all t). Assume training has converged: $w_t \approx w^*$ and $\nabla_w^2 L(w_t) \approx H$ for all t in a window of length K. Show that the response over this window satisfies

$$r_T \;\approx\; -\big[I - (I - \eta H)^K\big]\, H^{-1}\, \nabla_w L(w^*, z_k).$$

Hint: Sum the geometric series $\sum_{t=0}^{K-1} (I - \eta H)^t$ using the identity $\sum_{t=0}^{K-1} M^t = (I - M^K)(I - M)^{-1}$.

Take $K \to \infty$ (assuming $\eta < 2/\lambda_{\max}(H)$) and recover the influence function $r_\infty = -H^{-1} \nabla_w L(w^*, z_k) = r_{\mathrm{IF}}$.

(c) **(Stochastic case: convergence to IF.)** Now return to SGD with i.i.d. batches and a decaying learning rate satisfying $\sum_t \eta_t = \infty$, $\sum_t \eta_t^2 < \infty$ (the Robbins–Monro conditions). Assume SGD converges to a local minimum w^* with $H = \nabla_w^2 L(w^*)$ positive semidefinite. The continuous-time ODE that the response tracks is

$$\dot{r}(t) \;=\; -H\, r(t) \;+\; \nabla_w L(w^*, z_k). \tag{36}$$

(You do not need to prove that SGD tracks this ODE, this follows from standard stochastic approximation theory.)

i. Show that the equilibrium of (36) in the column space of H is $r_{\mathrm{IF}} = H^+ \nabla_w L(w^*, z_k)$, where H^+ is the pseudoinverse. This is the influence function, with H^+ in place of H^{-1} because the Hessian may be singular at a local minimum of an overparameterized model.

ii. Show that the component of $r(t)$ in the *null space* of H grows linearly: if P_0 is the projector onto $\ker(H)$, then $P_0\, r(t) = P_0\, r(0) + t\, P_0 \nabla_w L(w^*, z_k)$. Why does this component not converge? Under what condition on $\nabla_w L(w^*, z_k)$ does this runaway term vanish?

The full result (Mlodozeniec et al. (2025), Theorem 2) is: on the set of SGD trajectories that converge to a local minimum, $r_t \to r_{\mathrm{IF}} + r_{\mathrm{NS}}$ almost surely, where $r_{\mathrm{NS}} \in \ker(H)$. The influence function is the limiting response, up to a component in the flat directions of the loss.

4.4 Exercise: Influence depends on training time

The previous sections analyzed unrolling's asymptotics in the converged limit. But the whole point of unrolling is that the training *path* matters. The following exercise makes this concrete in a setting where the influence function admits a closed form at every point during training: a two-layer deep linear network (DLN), the same class of models from the SLT lecture day. The setup and analytic formula follow Lee et al. (2025).

Exercise 4.6 (Stagewise influence in a deep linear network)**.** A two-layer linear network $f_W(x) = W_2W_1x$ with $W_1, W_2 \in \mathbb{R}^{d\times d}$ is trained on $\{(x_i, y_i)\}_{i=1}^n$ with squared loss. Assume whitened inputs ($\frac{1}{n}\sum_i x_i x_i^\top = I$) and, for simplicity, that the input–output cross-covariance $\Sigma_{xy} = \frac{1}{n}\sum_i x_i y_i^\top$ is already diagonal with entries $s_1 > s_2 > \cdots > s_d > 0$.[a]

Dynamics. Under gradient flow with small balanced initialization, the network's modes evolve independently. The effective weight matrix at time t is $W(t) = \mathrm{diag}(\mathcal{G}_1(t), \ldots, \mathcal{G}_d(t))$, where each mode strength follows

$$\mathcal{G}_k(t) \;=\; \frac{s_k\, e^{2s_k t/\tau}}{e^{2s_k t/\tau} - 1 + s_k/\mathcal{G}_k(0)}, \tag{37}$$

with $\mathcal{G}_k(0) \ll s_k$ and time constant τ. Mode k transitions from ≈ 0 to $\approx s_k$ around time $t_k^* \approx \frac{\tau}{2s_k}\log(s_k/\mathcal{G}_k(0))$. Larger singular values saturate first: the network learns dominant structure before fine structure.

Analytic influence. Now perturb the weight of training example $z_p = (x_p, y_p)$ by ε, so the perturbed cross-covariance is $\Sigma_{xy}(\varepsilon) = \Sigma_{xy} + \varepsilon\, y_p x_p^\top$. Since Σ_{xy} is diagonal, the perturbation shifts its eigenvalues and rotates its eigenvectors. The influence of z_p on the weight matrix at time t decomposes as

$$\mathcal{I}_p(t) \;:=\; \left.\frac{\partial W(t,\varepsilon)}{\partial \varepsilon}\right|_{\varepsilon=0} \;=\; A\,\mathcal{G}(t) \;+\; \mathcal{G}'_\varepsilon(t) \;+\; \mathcal{G}(t)\,B, \tag{38}$$

with three terms:

- A and B are skew-symmetric matrices describing how the perturbation *rotates* the left/right singular bases. Their off-diagonal entries are $A_{jk} = B_{kj} = (y_p)_j(x_p)_k/(s_k - s_j)$ for $j \neq k$.
- $\mathcal{G}'_\varepsilon(t) \;=\; \mathrm{diag}\big(g'_t(s_1)\,(y_p)_1(x_p)_1,\;\ldots,\;g'_t(s_d)\,(y_p)_d(x_p)_d\big)$ encodes the change in mode strengths, where
$$g'_t(s_k) \;:=\; \frac{\partial\, \mathcal{G}_k(t)}{\partial s_k} \tag{39}$$
is the sensitivity of the k-th mode strength to a change in the corresponding singular value at time t.

The influence on the loss of a test example $z_q = (x_q, y_q)$, with residual $r_q(t) = y_q - W(t)x_q$, is

$$\mathcal{I}(z_p,\, \ell_q)(t) \;=\; -\,r_q(t)^\top \big(A\,\mathcal{G}(t) + \mathcal{G}'_\varepsilon(t) + \mathcal{G}(t)\,B\big)\, x_q. \tag{40}$$

(a) **(Three sources of influence.)** Interpret the three terms in (38):

- $A\,\mathcal{G}(t)$ and $\mathcal{G}(t)\,B$: the perturbation *rotates* the singular bases, mixing already-learned modes into each other. Why are these terms proportional to the current mode strengths $\mathcal{G}(t)$ rather than to their derivatives?
- $\mathcal{G}'_\varepsilon(t)$: the perturbation *shifts* the singular values, changing how fast each mode is learned. Why does this term involve $g'_t(s_k)$, sensitivity of the dynamics to the singular value, rather than $\mathcal{G}_k(t)$ itself?

(b) **(When does influence peak?)** Using the dynamics (37), argue that $g'_t(s_k)$ is peaked around $t \approx t_k^*$ (the transition time of mode k). Conclude that the $\mathcal{G}'_\varepsilon$ term, the part of influence that acts through the learning speed of each mode, is concentrated in time

around the moment the mode is being learned. Before and after, this contribution is negligible.

Hint: Consider the limits $t \ll t_k^*$ (mode not yet learning, $\mathcal{G}_k \approx \mathcal{G}_k(0)$) and $t \gg t_k^*$ (mode saturated, $\mathcal{G}_k \approx s_k$). In both cases, how sensitive is $\mathcal{G}_k$ to a small change in s_k?

(c) **(Sign flips.)** Consider $d = 2$ with $s_1 \gg s_2$: mode 1 captures a coarse distinction ("animal vs. plant") and mode 2 a fine distinction ("dog vs. cat" within animals). A dog example z_{dog} and a cat example z_{cat} share the same mode-1 coordinate but have opposite mode-2 coordinates: $(x_{\text{dog}})_2 = +(x_{\text{cat}})_2$. Using (40), argue that the influence of z_{dog} on a cat test loss can change sign during training: positive while mode 1 is being learned (shared structure), negative after mode 2 is learned (competing structure). At what time is the sign flip sharpest?

[a] The general case replaces the standard basis with the left/right singular vectors U, V of Σ_{xy} throughout; see Remark 4.7.

Remark 4.7 (General singular basis). When $\Sigma_{xy} = USV^\top$ is not diagonal, the formulas above hold with all vectors expressed in the singular basis: replace x_p by $V^\top x_p$, y_p by $U^\top y_p$, r_q by $U^\top r_q$, and $W(t)$ by $U^\top W(t)V$. The influence on W itself becomes $\mathcal{I}_p(t) = U(A\mathcal{G}(t) + \mathcal{G}'_\varepsilon(t) + \mathcal{G}(t)B)V^\top$, with $A_{jk} = (U^\top y_p)_j (V^\top x_p)_k / (s_k - s_j)$. See Lee et al. (2025) for the full derivation, including the degenerate case $s_j = s_k$.

Remark 4.8 (The developmental view of attribution). Lee et al. (2025) show that the same phenomenology (influence sign flips, non-monotonic trajectories) appears in language models, where e.g. the influence of delimiter tokens spikes when the model learns pairing structure and then decays. The practical consequence is that data attribution is not a one-shot computation but a time-varying signal, and methods like unrolling that track the training trajectory are better suited to capture this than methods that look only at the final checkpoint.

5 Practical Considerations and Open Problems

5.1 Distributional versus single-model attribution

5.2 Entangling latent concepts

5.3 Similarity metrics versus attribution

6 Further Readings

References

Juhan Bae, Nathan Ng, Alston Lo, Marzyeh Ghassemi, and Roger Grosse. If influence functions are the answer, then what is the question? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Juhan Bae, Wu Lin, Jonathan Lorraine, and Roger Grosse. Training data attribution via approximate unrolled differentiation. *Journal of Machine Learning Research*, 2024.

George Baker, Liam Carroll Wang, Jesse Hoogland, and Daniel Murfet. Studying small language models with susceptibilities, 2025.

R. Dennis Cook. Detection of influential observation in linear regression. *Technometrics*, 19(1): 15–18, 1977.

Logan Engstrom, Andrew Ilyas, Benjamin Chen, Axel Feldmann, William Moses, and Aleksander Mądry. Optimizing ML training with metagradient descent, 2025.

Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, Evan Hubinger, Kamile Lukošiūtė, Karina Nguyen, Nicholas Joseph, Sam McCandlish, Jared Kaplan, and Samuel R. Bowman. Studying large language model generalization with influence functions, 2023. URL https://arxiv.org/abs/2308.03296.

Frank R. Hampel. The influence curve and its role in robust estimation. *Journal of the American Statistical Association*, 69(346):383–393, 1974.

Christopher Hitchcock. The intransitivity of causation revealed in equations and graphs. *Journal of Philosophy*, 98(6):273–299, 2001.

Andrew Ilyas and Logan Engstrom. MAGIC: Near-optimal data attribution for deep learning, 2025.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning (ICML)*, 2017.

Philipp Alexander Kreer, Wilson Wu, Maxwell Adam, Zach Furman, and Jesse Hoogland. Bayesian influence functions for hessian-free data attribution, 2025.

Jin Hwa Lee, Matthew Smith, Maxwell Adam, and Jesse Hoogland. Influence dynamics and stagewise data attribution, 2025.

David Lewis. Causation. *Journal of Philosophy*, 70(17):556–567, 1973.

Bruno Mlodozeniec, Isaac Reid, Sam Power, David S. Krueger, Murat A. Erdogdu, Richard E. Turner, and Roger B. Grosse. Distributional training data attribution: What do influence functions sample?, 2025.

Aaron Mueller. Missed causes and ambiguous effects: Counterfactuals pose challenges for interpreting neural networks, 2024. URL https://arxiv.org/abs/2407.04690.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Lloyd S. Shapley. A value for n-person games. In H. W. Kuhn and A. W. Tucker, editors, *Contributions to the Theory of Games, Volume II*, pages 307–317. Princeton University Press, 1953.

A Solutions

A.1 Solutions to exercises from Section 1

Solution to Exercise ?? (Leave-one-out attribution for croissants).

A.2 Solutions to exercises from Section 2

Solution to Exercise 2.3 (Derivation of the influence function). [To be filled. Sketch: differentiate the optimality condition $\nabla_w L(w^*(\beta);\beta) = 0$ with respect to β_i and apply invertibility of H. The unique-minimum assumption is what licenses the implicit function theorem; without it the response function is not well-defined as a single-valued map. Part (b) is a one-line chain rule. Part (c) only changes the normalization.]

Solution to Exercise 2.4 (Linear regression). *(a)* The gradient of the training loss is $\nabla_w L(w) = -\sum_i x_i(y_i - x_i^\top w) = -\sum_i x_i y_i + Aw$. Setting this to zero gives $Aw^* = \sum_i x_i y_i$, hence $w^* = A^{-1}\sum_i x_i y_i$.

(b) The Hessian of the unweighted training loss is $H = \sum_i x_i x_i^\top = A$, and the gradient of the per-example loss at $z_j = (x_j, y_j)$ is $\nabla_w L(w^*, z_j) = -x_j(y_j - x_j^\top w^*) = -x_j\, r_j$. Plugging into Definition 2.2 with $\phi(w) = w$ (so $\nabla_w \phi(w^*) = I$),

$$\mathcal{I}(z_j, w^*) \;=\; -A^{-1}\big(-x_j r_j\big) \;=\; A^{-1} x_j\, r_j,$$

a residual times a leverage-weighted input.

(c) The normal equations for $w^*(\beta_j)$ read

$$\Big(\sum_{i\neq j} x_i x_i^\top + \beta_j\, x_j x_j^\top\Big)\, w \;=\; \sum_{i\neq j} x_i y_i \;+\; \beta_j\, x_j y_j,$$

or equivalently $\big(A - (1-\beta_j)\, x_j x_j^\top\big)\, w = \sum_i x_i y_i - (1-\beta_j)\, x_j y_j$. Sherman–Morrison applied to the rank-one update on the left gives

$$\big(A - (1-\beta_j)\, x_j x_j^\top\big)^{-1} \;=\; A^{-1} \;+\; \frac{(1-\beta_j)\, p_j p_j^\top}{1 - (1-\beta_j)\, h_j},$$

where we write $p_j := A^{-1} x_j$ and $h_j := x_j^\top p_j$. Multiplying through and using the identities $A^{-1}\sum_i x_i y_i = w^*$ and $p_j^\top \sum_i x_i y_i = x_j^\top w^*$,

$$\begin{aligned}
w^*(\beta_j) \;&=\; w^* \;-\; (1-\beta_j)\, p_j\, y_j \;+\; \frac{(1-\beta_j)\, p_j \big(x_j^\top w^* \;-\; (1-\beta_j)\, h_j\, y_j\big)}{1 - (1-\beta_j)\, h_j} \\
&=\; w^* \;+\; p_j \cdot \frac{-(1-\beta_j)\, y_j \big(1 - (1-\beta_j) h_j\big) \;+\; (1-\beta_j)\, x_j^\top w^* \;-\; (1-\beta_j)^2\, h_j\, y_j}{1 - (1-\beta_j)\, h_j} \\
&=\; w^* \;+\; p_j \cdot \frac{(1-\beta_j)\big(x_j^\top w^* - y_j\big)}{1 - (1-\beta_j)\, h_j} \\
&=\; w^* \;-\; \frac{(1-\beta_j)\, p_j\, r_j}{1 - (1-\beta_j)\, h_j},
\end{aligned}$$

the claimed expression. The result is rational in β_j because the matrix being inverted is linear in β_j.

(d) Expanding $1/(1-(1-\beta_j)h_j)$ as a geometric series in $(1-\beta_j)h_j$ (valid for $|(1-\beta_j)h_j| < 1$),

$$w^*(\beta_j) - w^* \;=\; -(1-\beta_j)\, A^{-1} x_j\, r_j \sum_{k=0}^{\infty} (1-\beta_j)^k\, h_j^k \;=\; -A^{-1} x_j\, r_j \sum_{k=1}^{\infty} (1-\beta_j)^k\, h_j^{k-1}.$$

Reading off term by term:

- The $k = 1$ term is $-A^{-1} x_j\, r_j\, (1-\beta_j) = \mathcal{I}(z_j, w^*) \cdot (\beta_j - 1)$, exactly the influence-function prediction — as it must be, since the IF *is* the linear coefficient of $w^*(\beta_j)$ at $\beta_j = 1$.
- The $k = 2$ term, $-A^{-1} x_j\, r_j\, (1-\beta_j)^2\, h_j$, is the second-order Taylor correction. It carries one factor of leverage and is doubly suppressed: small both in $(1-\beta_j)$ and in h_j.
- Each subsequent term picks up another factor of $(1-\beta_j)h_j$. The IF/LOO "error" is precisely this tail, viewed as a series in the perturbation size and the leverage.

(e) At $\beta_j = 0$ the perturbation factor $(1 - \beta_j)$ is unity, and the higher-order terms no longer have a small parameter in front of them — only powers of h_j remain. The series collapses to a pure geometric series in h_j:

$$w^*_{(-j)} - w^* \;=\; -A^{-1}x_j\, r_j \left(1 + h_j + h_j^2 + h_j^3 + \cdots\right) \;=\; -\frac{A^{-1}x_j\, r_j}{1 - h_j}.$$

The IF prediction $-A^{-1}x_j\, r_j$ is the $k = 0$ leading term; everything else is the Taylor remainder. So the often-quoted "factor of $1/(1 - h_j)$" between IF and LOO is not a non-perturbative effect: it is the sum of all the higher-order Taylor terms, which OLS happens to admit in closed form because $w^*(\beta_j)$ is rational. For more general M-estimators the response function is no longer rational and the higher-order terms do not sum to such a clean expression — but they are still there, with the same qualitative behavior.

In the two limiting regimes:

- $h_j \to 0$ (low leverage). Each higher-order term carries a factor of h_j^{k-1}, so they are individually negligible and the IF approximation is essentially exact.
- $h_j \to 1$ (high leverage). The geometric series barely converges and the higher-order terms together carry an arbitrarily large total weight. The IF underestimates the true LOO change without bound.

This is a clean demonstration that even in a strictly convex, fully-converged setting the linearization in Definition 2.2 loses control over precisely the points one most wants to attribute — the ones the model could not fit without them. The lesson generalizes: high-leverage / outlier examples are exactly where IF approximations should be distrusted, and the modern fixes from Section 2.2 (damping, GNH, PBRF) all address related but more severe versions of the same issue.

Solution to Exercise 2.8 (Failure of the first-order approximation in a degenerate model). *(a)* The per-example loss is $\ell_i = \frac{1}{2}(z_i - ab)^2$. Its second derivatives are

$$\frac{\partial^2 \ell_i}{\partial a^2} = b^2, \qquad \frac{\partial^2 \ell_i}{\partial b^2} = a^2, \qquad \frac{\partial^2 \ell_i}{\partial a\, \partial b} = 2ab - z_i.$$

Summing over i and evaluating at any minimizer (a^*, b^*) with $a^* b^* = \bar{z}$,

$$H \;=\; n \begin{pmatrix} (b^*)^2 & a^* b^* \\ a^* b^* & (a^*)^2 \end{pmatrix} \;=\; n\, \mathbf{v}\mathbf{v}^\top, \qquad \mathbf{v} := \begin{pmatrix} b^* \\ a^* \end{pmatrix},$$

where we used $\sum_i (2a^* b^* - z_i) = n(2\bar{z} - \bar{z}) = n a^* b^*$ for the off-diagonal. This is rank one, with image $\operatorname{span}(\mathbf{v})$.

The null space is $\operatorname{span}\big((a^*, -b^*)^\top\big)$. Geometrically: the minimizers form the hyperbola $\{(a, b) : ab = \bar{z}\}$, and the tangent to this hyperbola at (a^*, b^*) is exactly the direction $(a^*, -b^*)$ — the infinitesimal generator of the rescaling symmetry $(a, b) \to (\alpha a,\, b/\alpha)$ at $\alpha = 1$. The null space of H is the tangent to the manifold of minimizers.

(b) We need two gradients at the minimizer. For the observable $\phi(w) = ab$:

$$\nabla_w \phi \;=\; (b^*,\, a^*)^\top \;=\; \mathbf{v}.$$

For the per-example loss at z_i:

$$\nabla_w L(w^*, z_i) \;=\; -(z_i - \bar{z})\, (b^*,\, a^*)^\top \;=\; -(z_i - \bar{z})\, \mathbf{v}.$$

Both point in the $\mathbf{v}$ direction — neither has a component in the null space of H. The Moore–Penrose pseudoinverse of $H = n\,\mathbf{v}\mathbf{v}^\top$ is

$$H^+ \;=\; \frac{1}{n\,\|\mathbf{v}\|^4}\,\mathbf{v}\mathbf{v}^\top, \qquad \|\mathbf{v}\|^2 = (a^*)^2 + (b^*)^2.$$

Plugging into the IF formula,

$$\mathcal{I}(z_i, \phi) \;=\; -\mathbf{v}^\top H^+\big(-(z_i - \bar{z})\,\mathbf{v}\big) \;=\; (z_i - \bar{z})\,\frac{\|\mathbf{v}\|^4}{n\,\|\mathbf{v}\|^4} \;=\; \frac{z_i - \bar{z}}{n}.$$

The $\|\mathbf{v}\|$ factors cancel: the pseudoinverse IF is *independent* of the choice of minimizer (a^*, b^*).

Comparison with exact LOO. Removing z_i changes the sample mean to $\bar{z}_{(-i)} = (n\bar{z} - z_i)/(n-1)$, and the observable at any new minimizer is $\phi = \bar{z}_{(-i)}$. So the exact LOO change is

$$\phi(w^*_{(-i)}) - \phi(w^*) \;=\; \bar{z}_{(-i)} - \bar{z} \;=\; -\,\frac{z_i - \bar{z}}{n-1}.$$

The IF predicts $-\mathcal{I}(z_i, \phi) = -(z_i - \bar{z})/n$. The ratio is $n/(n-1) = 1/(1-h)$ with leverage $h = 1/n$ — every point has equal leverage in this one-effective-parameter model, and we recover the same $1/(1-h)$ correction as in Exercise 2.4.

(c) The eigenvalues of $H = n\,\mathbf{v}\mathbf{v}^\top$ are $n\|\mathbf{v}\|^2$ (eigenvector $\hat{\mathbf{v}} = \mathbf{v}/\|\mathbf{v}\|$) and 0 (eigenvector $\hat{\mathbf{v}}_\perp = (a^*, -b^*)^\top/\|\mathbf{v}\|$). So

$$(H + \lambda I)^{-1} \;=\; \frac{1}{n\|\mathbf{v}\|^2 + \lambda}\,\hat{\mathbf{v}}\hat{\mathbf{v}}^\top \;+\; \frac{1}{\lambda}\,\hat{\mathbf{v}}_\perp\hat{\mathbf{v}}_\perp^\top.$$

Since $\nabla\phi$ and ∇L_i both lie in the $\hat{\mathbf{v}}$ direction, the $1/\lambda$ term does not contribute, and

$$\mathcal{I}_\lambda(z_i, \phi) \;=\; \frac{(z_i - \bar{z})\,\|\mathbf{v}\|^2}{n\|\mathbf{v}\|^2 + \lambda}.$$

Dependence on the minimizer. For $\lambda > 0$ the answer depends on $\|\mathbf{v}\|^2 = (a^*)^2 + (b^*)^2$, which varies along the orbit $ab = \bar{z}$. For instance, the "balanced" minimizer $(\sqrt{\bar{z}}, \sqrt{\bar{z}})$ gives $\|\mathbf{v}\|^2 = 2\bar{z}$, while the "unbalanced" choice $(\bar{z}, 1)$ gives $\|\mathbf{v}\|^2 = \bar{z}^2 + 1$. Different minimizers yield different IFs. This is the proximity gap: the damping term $\frac{\lambda}{2}\|w - w^*\|^2$ breaks the rescaling symmetry and anchors the computation at a particular point on the orbit.

Limit $\lambda \to 0$. $\mathcal{I}_\lambda \to (z_i - \bar{z})/n$ regardless of (a^*, b^*), recovering the pseudoinverse answer from (b). The damping-induced dependence on the minimizer vanishes.

Limit $\lambda \to \infty$. $\mathcal{I}_\lambda \to 0$: heavy damping kills all attribution, because the proximal penalty makes it too costly to move w at all.

The punchline: for this particular observable ($\phi = ab$, which is constant along the orbit), the pseudoinverse gives a clean, minimizer-independent answer and the damping only introduces a spurious dependence. But for an observable that *does* vary along the orbit — say $\phi(w) = a^2$ — $\nabla\phi$ would have a component in the null direction $(a^*, -b^*)$, the $1/\lambda$ term would contribute, and the IF would *diverge* as $\lambda \to 0$. In that case damping is not a nuisance but a necessity, and the choice of minimizer genuinely matters. This is the core tension of the proximity gap: it is a distortion for orbit-invariant queries, but a regularizer for orbit-dependent ones.

A.3 Solutions to exercises from Section 3

Solution to Exercise 3.2 (The BIF is a covariance). Write $Z(\beta) = \int \exp(-\sum_j \beta_j L(w, z_j))\,\varphi(w)\,dw$ for the normalizing constant, so that

$$\mathbb{E}_{p_\beta}[\phi(w)] \;=\; \frac{1}{Z(\beta)}\int \phi(w)\,\exp\Big(-\sum_j \beta_j L(w, z_j)\Big)\,\varphi(w)\,dw.$$

Differentiating with respect to β_i and applying the quotient rule:

$$\frac{\partial}{\partial \beta_i}\,\mathbb{E}_{p_\beta}[\phi] \;=\; \frac{1}{Z}\int \phi(w)\big(-L(w,z_i)\big)\,e^{-\sum_j \beta_j L_j}\,\varphi\,dw \;-\; \frac{1}{Z^2}\cdot\frac{\partial Z}{\partial \beta_i}\int \phi\, e^{-\sum_j \beta_j L_j}\,\varphi\,dw.$$

The first term is $-\mathbb{E}_{p_\beta}[\phi(w)\,L(w,z_i)]$. For the second, $\partial_{\beta_i} Z = -Z\,\mathbb{E}_{p_\beta}[L(w,z_i)]$, so it equals $+\mathbb{E}_{p_\beta}[L(w,z_i)]\,\mathbb{E}_{p_\beta}[\phi(w)]$. Evaluating at $\beta = \mathbf{1}$:

$$\mathrm{BIF}(z_i,\phi) = -\mathbb{E}_p[\phi\cdot L_i] + \mathbb{E}_p[\phi]\,\mathbb{E}_p[L_i] = -\mathrm{Cov}_{w\sim p}(L(w,z_i),\,\phi(w)).$$

Structurally: in the classical IF, the "correlation" between L_i and ϕ is mediated by the inverse Hessian acting on their gradients at a point; in the BIF it is mediated by the full posterior distribution. The inverse Hessian is how a Gaussian posterior would produce a covariance (as we will see in Exercise 3.5), so the BIF strictly generalizes the classical formula.

Solution to Exercise 3.5 (Power series expansion of Bayesian influence functions). *(a)* The log-posterior is $\log p(w \mid \mathcal{D}) = -L(w) + \log\varphi(w) + \text{const}$. Taylor-expanding $L(w)$ around the minimum w^*:

$$L(w) = L(w^*) + \underbrace{\nabla L(w^*)^\top}_{=0}\Delta w + \tfrac{1}{2}\,\Delta w^\top H\,\Delta w + O(\|\Delta w\|^3).$$

The linear term vanishes because w^* is a critical point. Substituting into $\log p$:

$$\log p(w \mid \mathcal{D}) \approx -L(w^*) - \tfrac{1}{2}\,\Delta w^\top H\,\Delta w + \log\varphi(w) + \text{const}.$$

Since $L = \sum_{i=1}^n L_i$, the Hessian $H = \sum_i \nabla^2 L_i$ scales as $O(n)$, while the prior contributes $O(1)$ to the log-density. For large n the quadratic term dominates, giving $p(w \mid \mathcal{D}) \approx \mathcal{N}(w^*,\,H^{-1})$. This is the Laplace approximation. (The Bernstein–von Mises theorem makes this rigorous: under regularity, the posterior converges to this Gaussian in total variation as $n \to \infty$.)

(b) Since $\mathrm{Cov}(X + c, Y) = \mathrm{Cov}(X, Y)$ for any constant c, the constant terms $\phi(w^*)$ and $L(w^*, z_i)$ drop out. The covariance of the two Taylor series is bilinear, so it distributes over the sum of terms:

$$\mathrm{Cov}(\phi, L_i) = \sum_{k\geq 1}\sum_{m\geq 1}\mathrm{Cov}(T_k[\phi],\,T_m[L_i]).$$

Under $\Delta w \sim \mathcal{N}(0, H^{-1})$, $T_k[\phi]$ is a degree-k polynomial in Δw. The covariance $\mathrm{Cov}(T_k, T_m) = \mathbb{E}[T_k T_m] - \mathbb{E}[T_k]\mathbb{E}[T_m]$ involves moments of Δw of degree $k+m$. For a centered Gaussian, odd moments vanish: $\mathbb{E}[\Delta w_{a_1}\cdots\Delta w_{a_\ell}] = 0$ when ℓ is odd. If $k+m$ is odd, then all moments in $\mathbb{E}[T_k T_m]$ and $\mathbb{E}[T_k]\mathbb{E}[T_m]$ involve an odd total degree, so both vanish and $\mathrm{Cov}(T_k, T_m) = 0$.

(c) $T_1[\phi] = g_\phi^\top \Delta w$ and $T_1[L_i] = g_i^\top \Delta w$ are linear in Δw. Their covariance is

$$\mathrm{Cov}(g_\phi^\top\Delta w,\; g_i^\top\Delta w) = g_\phi^\top\,\mathbb{E}[\Delta w\,\Delta w^\top]\,g_i = g_\phi^\top H^{-1} g_i.$$

Negating gives $-g_\phi^\top H^{-1} g_i = -\nabla\phi(w^*)^\top H^{-1}\nabla L(w^*, z_i)$, which is the classical influence function $\mathcal{I}(z_i,\phi)$ from Definition 2.2.

(d) Write $Q_\phi = \frac{1}{2}\Delta w^\top H_\phi\,\Delta w$ and $Q_i = \frac{1}{2}\Delta w^\top H_i\,\Delta w$. We need $\mathrm{Cov}(Q_\phi, Q_i) = \mathbb{E}[Q_\phi Q_i] - \mathbb{E}[Q_\phi]\mathbb{E}[Q_i]$ with $\Delta w \sim \mathcal{N}(0,\Sigma)$, $\Sigma = H^{-1}$.

For the expectation: $\mathbb{E}[Q_\phi] = \frac{1}{2}\,\mathrm{tr}(H_\phi\Sigma)$ and $\mathbb{E}[Q_i] = \frac{1}{2}\,\mathrm{tr}(H_i\Sigma)$.

For the cross-moment: $\mathbb{E}[Q_\phi Q_i] = \frac{1}{4}\sum_{a,b,c,d}(H_\phi)_{ab}(H_i)_{cd}\,\mathbb{E}[\Delta w_a\Delta w_b\Delta w_c\Delta w_d]$. By Isserlis' theorem:

$$\mathbb{E}[\Delta w_a\Delta w_b\Delta w_c\Delta w_d] = \Sigma_{ab}\Sigma_{cd} + \Sigma_{ac}\Sigma_{bd} + \Sigma_{ad}\Sigma_{bc}.$$

The first pairing gives $\frac{1}{4}\,\mathrm{tr}(H_\phi\Sigma)\,\mathrm{tr}(H_i\Sigma) = \mathbb{E}[Q_\phi]\mathbb{E}[Q_i]$, which cancels in the covariance. The other two pairings each give $\frac{1}{4}\,\mathrm{tr}(H_\phi\Sigma H_i\Sigma)$ (by cyclicity of the trace and symmetry of H_ϕ, H_i, and Σ). So

$$\mathrm{Cov}(Q_\phi, Q_i) = 2\cdot\tfrac{1}{4}\,\mathrm{tr}(H_\phi\Sigma H_i\Sigma) = \tfrac{1}{2}\,\mathrm{tr}(H_\phi H^{-1} H_i H^{-1}).$$

In the linear regression setting with $\phi(w) = w_k$ (a coordinate function), $H_\phi = \nabla^2 w_k = 0$: the observable is linear in w, so its Hessian vanishes and the entire $(2,2)$ correction is zero. This is why the BIF equals the (damped) IF exactly for linear regression — all corrections beyond leading order involve H_ϕ or higher derivatives of ϕ, which vanish for a linear observable.

(e) The localized posterior of Equation (10) has effective Hessian $H_{\text{eff}} = H + \gamma I$, so the Laplace approximation gives $\Delta w \sim \mathcal{N}(0, (H+\gamma I)^{-1})$. The leading-order $(1,1)$ computation from part (c) becomes

$$\text{BIF}_\gamma(z_i, \phi) \approx -g_\phi^\top (H+\gamma I)^{-1} g_i = -\nabla\phi(w^*)^\top (H+\gamma I)^{-1} \nabla L(w^*, z_i),$$

which is the damped IF of Equation (6) with γ in the role of λ. The higher-order corrections from parts (b)–(d) carry through with H^{-1} replaced by $(H+\gamma I)^{-1}$ throughout.

(f) Summary: For regular (non-singular) models, the posterior is approximately Gaussian (Bernstein–von Mises), the Taylor expansion converges, and the $(1,1)$ term dominates (it scales as $O(1/n)$ while higher terms scale as $O(1/n^2)$ and beyond). The classical IF is a good approximation because it *is* the leading term. For singular models (neural networks), the posterior is concentrated on a positive-dimensional variety, not at an isolated point. The Laplace approximation fails: the Hessian has a large null space, the posterior is non-Gaussian, and the Taylor series does not converge around w^*. In this regime, the BIF — defined as an exact covariance under the true posterior — captures the full geometry, while the classical IF is at best the leading term of an expansion that does not converge. The BIF is the fundamental object; the IF is the Gaussian shadow it casts.

Solution to Exercise 3.7 (Influence functions as optimal parameter shifts). *(a)* Using $p_0^\beta(w) = e^{-\beta L(w)}/Z(\beta,0)$ and $p_\varepsilon^\beta(w) = e^{-\beta(L(w)-\varepsilon L_k(w))}/Z(\beta,\varepsilon)$, substituting into (13) gives

$$\begin{aligned} D_{\text{KL}}(q_\varepsilon \| p_\varepsilon^\beta) &= \mathbb{E}_{w\sim p_0^\beta}\big[\log p_0^\beta(w) - \log p_\varepsilon^\beta(w+\varepsilon r) - \log|\det(I+\varepsilon\nabla r)|\big] \\ &= \mathbb{E}_{p_0^\beta}\big[-\beta L(w) - \log Z(\beta,0) + \beta(L(w+\varepsilon r) - \varepsilon L_k(w+\varepsilon r)) \\ &\qquad + \log Z(\beta,\varepsilon) - \log|\det(I+\varepsilon\nabla r)|\big]. \end{aligned}$$

Dividing by β and rearranging gives the claimed expression. As $\beta\to\infty$: (i) $\frac{1}{\beta}\log|\det(I+\varepsilon\nabla r)| \to 0$ since the log-determinant is $O(1)$ (bounded for smooth r and small ε); (ii) by Laplace's method, $\frac{1}{\beta}\log Z(\beta,\varepsilon) \to -\inf_w[L(w)-\varepsilon L_k(w)]$, so $\frac{1}{\beta}\log\frac{Z(\beta,\varepsilon)}{Z(\beta,0)} \to \inf L - \inf(L-\varepsilon L_k)$, which is a constant independent of r; (iii) the expectation concentrates on w^*.

(b) At w^*, $\nabla L(w^*) = 0$, so the Taylor expansion of $L(w^*+\varepsilon r)$ starts at the quadratic term: $L(w^*) + \frac{\varepsilon^2}{2} r^\top H r + O(\varepsilon^3)$. For εL_k: $\varepsilon L_k(w^*+\varepsilon r) = \varepsilon L_k(w^*) + \varepsilon^2 \nabla L_k^\top r + O(\varepsilon^3)$. Combined:

$$L(w^*+\varepsilon r) - \varepsilon L_k(w^*+\varepsilon r) - L(w^*) = -\varepsilon L_k(w^*) + \frac{\varepsilon^2}{2} r^\top H r - \varepsilon^2 \nabla L_k^\top r + O(\varepsilon^3).$$

For the infimum: expanding $L(w^*+\delta) - \varepsilon L_k(w^*+\delta)$ to second order in δ and minimizing, the optimal perturbation is $\delta^* = \varepsilon H^{-1}\nabla L_k$, giving

$$\inf_w[L - \varepsilon L_k] = L(w^*) - \varepsilon L_k(w^*) - \frac{\varepsilon^2}{2}\nabla L_k^\top H^{-1}\nabla L_k + O(\varepsilon^3).$$

So $\inf L - \inf(L-\varepsilon L_k) = \varepsilon L_k(w^*) + \frac{\varepsilon^2}{2}\nabla L_k^\top H^{-1}\nabla L_k + O(\varepsilon^3)$.

Adding the two contributions:

$$\begin{aligned} \mathcal{F}(r,\varepsilon) &= \frac{\varepsilon^2}{2} r^\top H r - \varepsilon^2 \nabla L_k^\top r + \frac{\varepsilon^2}{2}\nabla L_k^\top H^{-1}\nabla L_k + O(\varepsilon^3) \\ &= \frac{\varepsilon^2}{2}\big(r^\top H r - 2\nabla L_k^\top r + \nabla L_k^\top H^{-1}\nabla L_k\big) + O(\varepsilon^3). \end{aligned}$$

Completing the square: $r^\top Hr - 2\nabla L_k^\top r + \nabla L_k^\top H^{-1}\nabla L_k = (r - H^{-1}\nabla L_k)^\top H(r - H^{-1}\nabla L_k)$, since expanding the right side gives $r^\top Hr - 2\nabla L_k^\top H^{-1} \cdot Hr + \nabla L_k^\top H^{-1} H H^{-1}\nabla L_k = r^\top Hr - 2\nabla L_k^\top r + \nabla L_k^\top H^{-1}\nabla L_k$.

(c) Since H is positive definite, the squared Mahalanobis form $(r - r_{\text{IF}})^\top H(r - r_{\text{IF}}) \geq 0$ with equality iff $r = r_{\text{IF}} = H^{-1}\nabla L_k$. At the minimum, $\mathcal{F}(r_{\text{IF}}, \varepsilon) = O(\varepsilon^3)$: the IF shifts the distribution with only a *third-order* KL residual. Any other shift $r \neq r_{\text{IF}}$ incurs a strictly positive $O(\varepsilon^2)$ cost proportional to its Mahalanobis distance from the IF. The IF is the unique optimal shift among all smooth vector fields r, not just among constant maps.

(d) When H is singular, the quadratic form $r^\top Hr$ is zero for $r \in N = \text{Null}(H)$: movement in the null space has zero Hessian cost because the loss is flat along the minimum manifold. The pseudoinverse H^+ inverts only the non-degenerate directions, and the completed square becomes $(r - H^+\nabla L_k)^\top H(r - H^+\nabla L_k)$. This vanishes whenever $r - H^+\nabla L_k \in N$, i.e., the minimizer is $r = H^+\nabla L_k + r_{\text{NS}}$ for any $r_{\text{NS}} \in N$.

Geometrically: the IF determines the optimal shift *off* the minimum manifold (perpendicular to $\mathcal{S}_L$), but movement *along* $\mathcal{S}_L$ is invisible to the energy at this order. In the SLT language: the degeneracy of the singular set means that many shifts are equally good at leading order — the IF picks one, but the null-space ambiguity is real.

(e) For linear regression with Gaussian prior, the posterior is $\mathcal{N}(\mu, V)$ with $V^{-1} = \sigma^{-2}A + \tau^{-2}I$. A constant shift $w \mapsto w + \varepsilon r$ sends $\mathcal{N}(\mu, V)$ to $\mathcal{N}(\mu + \varepsilon r, V)$. The true perturbed mean is $\mu + \varepsilon b + O(\varepsilon^2)$ with $b = \text{BIF}(z_k, w) = (A + \lambda I)^{-1} x_k r_k$. For two Gaussians with the same covariance:

$$D_{\text{KL}}\big(\mathcal{N}(\mu + \varepsilon r, V) \| \mathcal{N}(\mu + \varepsilon b, V)\big) = \frac{\varepsilon^2}{2}(r - b)^\top V^{-1}(r - b).$$

This is minimized at $r = b = (A + \lambda I)^{-1} x_k r_k$. To check consistency with (15): the "Hessian of the energy" in the Boltzmann distribution $p \propto e^{-(\text{loss}+\text{prior})}$ is $H_{\text{eff}} = \sigma^{-2}A + \tau^{-2}I = V^{-1}$, and $H_{\text{eff}}^{-1}\nabla_w L(\mu, z_k) = V \cdot \sigma^{-2} x_k r_k = (A + \lambda I)^{-1} x_k r_k = \text{BIF}$. The general formula (15) reduces to $\frac{\varepsilon^2}{2}(r - b)^\top V^{-1}(r - b)$, exactly as computed directly.

(f) The two exercises are complementary:

- The Laplace expansion (Exercise 3.5) tells you *what* the IF is: the leading-order term of the BIF covariance under a Gaussian approximation to the posterior. The higher-order corrections (Hessian products, etc.) are visible but require the Laplace approximation to hold.
- The transport perspective (this exercise) tells you *why* the IF works even when the Laplace approximation fails: it minimizes the KL cost of transporting the parameter distribution, and this optimality holds in the $\beta \to \infty$ limit under mild regularity assumptions — no Gaussianity required. For singular models, the pseudoinverse handles the degenerate directions naturally (part d), and the result still says: the IF is the best you can do with a smooth first-order map.

This provides what Mlodozeniec et al. (2025) call a "distributional" justification for IFs in deep learning. The original justification (implicit function theorem, convexity) does not hold for neural networks. The Laplace/BIF justification (Exercise 3.5) requires Gaussianity of the posterior, which also fails. But the transport optimality result asks only for bounded derivatives and a well-defined minimum — conditions much closer to what actually holds in practice.

A.4 Solutions to exercises from Section 4

Solution to Exercise 4.2 (Derivation of the unrolling formula). *(a)* Differentiating (16) with respect to β_i at $\beta = \mathbf{1}$ via the chain rule:

$$\begin{aligned}\frac{\partial w_{t+1}}{\partial \beta_i} &= \frac{\partial w_t}{\partial \beta_i} - \frac{\eta_t}{b} \sum_{j \in B_t} \Big[\underbrace{\nabla_w^2 L(w_t, z_j) \, \frac{\partial w_t}{\partial \beta_i}}_{\text{indirect}} + \underbrace{\delta_{ij} \, \nabla_w L(w_t, z_j)}_{\text{direct}} \Big] \\ &= \Big(I - \frac{\eta_t}{b} \sum_{j \in B_t} \nabla_w^2 L(w_t, z_j) \Big) \frac{\partial w_t}{\partial \beta_i} - \frac{\eta_t}{b} \, \mathbf{1}[i \in B_t] \, \nabla_w L(w_t, z_i) \\ &= J_t \, \frac{\partial w_t}{\partial \beta_i} - \frac{\eta_t}{b} \, \mathbf{1}[i \in B_t] \, \nabla_w L(w_t, z_i).\end{aligned}$$

The initial condition is $\partial w_0 / \partial \beta_i = 0$ since w_0 does not depend on β.

(b) Substituting the recursion repeatedly:

$$\begin{aligned}\frac{\partial w_T}{\partial \beta_i} &= J_{T-1} \frac{\partial w_{T-1}}{\partial \beta_i} - \frac{\eta_{T-1}}{b} \, \mathbf{1}[i \in B_{T-1}] \, \nabla_w L(w_{T-1}, z_i) \\ &= J_{T-1} \Big(J_{T-2} \frac{\partial w_{T-2}}{\partial \beta_i} - \frac{\eta_{T-2}}{b} \, \mathbf{1}[i \in B_{T-2}] \, \nabla_w L(w_{T-2}, z_i) \Big) - \frac{\eta_{T-1}}{b} \, \mathbf{1}[i \in B_{T-1}] \, \nabla_w L(w_{T-1}, z_i) \\ &= \cdots = - \sum_{t=0}^{T-1} \frac{\eta_t}{b} \, \mathbf{1}[i \in B_t] \, J_{(t+1):T} \, \nabla_w L(w_t, z_i),\end{aligned}$$

where the last step follows from $\partial w_0 / \partial \beta_i = 0$ killing the $J_{0:T}$ term. This is (20).

(c) With the preconditioned update (22), the indirect effect picks up the preconditioner in the Hessian term: $J_t^{(P)} = I - (\eta_t / b) \sum_{j \in B_t} P_t \nabla_w^2 L(w_t, z_j) = I - \eta_t P_t \widehat{H}_t$. The direct effect becomes $-(\eta_t / b) \, \mathbf{1}[i \in B_t] \, P_t \nabla_w L(w_t, z_i)$. Unrolling the recursion as in (b) gives (23). The preconditioner P_t at step t acts as a local rescaling of the gradient: Adam's adaptive scaling amplifies gradients in directions with historically small second moments. A datum appearing early in training (when Adam has not yet accumulated accurate statistics) will have its gradient rescaled differently than the same datum appearing later, when P_t has stabilized. This is a concrete mechanism by which optimizer choice affects per-datum attribution.

Solution to Exercise 4.5 (Convergence of unrolling to influence functions). *(a)* Differentiating (34) with respect to ε at $\varepsilon = 0$: the first row is just the SGD update, independent of ε. For the response, the chain rule gives two contributions: the indirect effect through $w_t(\varepsilon)$ picks up the mini-batch Hessian (giving $J_t \, r_t$ as in Exercise 4.2), and the direct effect from the $(1 - \varepsilon \, \mathbf{1}[i = k])$ factor contributes $+(\eta_t / b) \, \mathbf{1}[k \in B_t] \, \nabla_w L(w_t, z_k)$. This is (35). Given the current state (w_t, r_t) and the batch selection B_t (which is i.i.d. and independent of history), the next state (w_{t+1}, r_{t+1}) depends only on (w_t, r_t) — no earlier states are needed. This is the Markov property.

(b) With full-batch GD, $B_t = \{1, \ldots, n\}$, the response recursion becomes deterministic: $r_{t+1} = (I - \eta H) r_t + \eta \nabla_w L(w^*, z_k)$. With $r_0 = 0$, unrolling gives $r_K = \eta \sum_{t=0}^{K-1} (I - \eta H)^t \, \nabla_w L(w^*, z_k)$. Using $\sum_{t=0}^{K-1} M^t = (I - M^K)(I - M)^{-1}$ with $M = I - \eta H$:

$$r_K = [I - (I - \eta H)^K] \, H^{-1} \, \nabla_w L(w^*, z_k).$$

(Note the sign: the interpolation $1 - \varepsilon \, \mathbf{1}[i = k]$ downweights z_k, so the direct effect has the opposite sign from the β-upweighting in Exercise 4.2.) Since $\|I - \eta H\| < 1$ when $\eta < 2 / \lambda_{\max}(H)$, $(I - \eta H)^K \to 0$ and $r_\infty = H^{-1} \nabla_w L(w^*, z_k) = r_{\mathrm{IF}}$.

(c) i. At equilibrium $\dot{r} = 0$, the ODE (36) gives $H r_\infty = \nabla_w L(w^*, z_k)$. In the column space of H, this has the unique solution $r_{\mathrm{IF}} = H^+ \nabla_w L(w^*, z_k)$. (If H is invertible, $H^+ = H^{-1}$ and this is the classical IF.)

ii. Project the ODE onto $\ker(H)$: $P_0\dot{r}(t) = -P_0 H\, r(t) + P_0\nabla_w L(w^*, z_k) = P_0\nabla_w L(w^*, z_k)$, since $P_0 H = 0$. Integrating: $P_0 r(t) = P_0 r(0) + t\, P_0\nabla_w L(w^*, z_k)$. This grows linearly unless $P_0\nabla_w L(w^*, z_k) = 0$, i.e. unless the per-example gradient has no component in the Hessian null space. The null space of H at a local minimum corresponds to flat directions along the minimum manifold; the response diverges if the perturbation "pushes" along these flat directions, because the optimizer has no restoring force.

Solution to Exercise 4.6 (Stagewise influence in a deep linear network). *(a)* The $A\mathcal{G}(t)$ and $\mathcal{G}(t)B$ terms describe how the perturbation rotates the singular basis. A rotation mixes already-learned modes into each other, so its effect on the weight matrix is proportional to the current mode strengths: if mode k has not yet been learned ($\mathcal{G}_k \approx 0$), rotating its basis direction contributes nothing to $W(t)$. The $\mathcal{G}'_\varepsilon(t)$ term, by contrast, acts through the learning *dynamics*: perturbing s_k changes the speed at which mode k is learned, and this matters precisely when the dynamics are sensitive to s_k — not when $\mathcal{G}_k(t)$ is large or small per se, but when $\mathcal{G}_k(t)$ is *changing rapidly* as a function of s_k. This is why the derivative $g'_t(s_k) = \partial\mathcal{G}_k(t)/\partial s_k$ appears rather than $\mathcal{G}_k(t)$ itself.

(b) From (37), $\mathcal{G}_k(t) = s_k e^{2s_k t/\tau}/(e^{2s_k t/\tau} - 1 + s_k/\mathcal{G}_k(0))$. For $t \ll t_k^*$: $e^{2s_k t/\tau} \approx 1$, so $\mathcal{G}_k \approx \mathcal{G}_k(0)$ regardless of s_k — the mode hasn't started learning yet, and a small change in s_k makes no difference, so $g'_t(s_k) \approx 0$. For $t \gg t_k^*$: $e^{2s_k t/\tau} \gg s_k/\mathcal{G}_k(0)$, so $\mathcal{G}_k \approx s_k$ and $g'_t(s_k) \approx 1$ — but the *residual* in mode k is also ≈ 0, so this term's contribution to the loss influence (40) is negligible. The product $g'_t(s_k) \times$ (residual in mode k) is peaked around $t \approx t_k^*$, when the mode is in transition: the dynamics are maximally sensitive to s_k *and* the residual is still nonzero.

(c) Write $x_{\text{dog}} = (\alpha, +\delta)$ and $x_{\text{cat}} = (\alpha, -\delta)$, where α is the shared mode-1 component and $\pm\delta$ are opposite mode-2 components (since Σ_{xy} is diagonal, the standard basis is the singular basis). During mode-1 learning ($t \approx t_1^*$, mode 2 not yet active): the residual of the cat test example has a large mode-1 component, and z_{dog}'s perturbation increases s_1 (via $(y_{\text{dog}})_1(x_{\text{dog}})_1 > 0$), accelerating mode-1 learning. This reduces the cat test loss — positive influence. After mode-2 learning ($t \approx t_2^*$): the cat's residual is now dominated by mode 2, and z_{dog}'s mode-2 perturbation has the wrong sign for the cat (because $(x_{\text{dog}})_2$ and $(x_{\text{cat}})_2$ have opposite signs). This increases the cat test loss — negative influence. The sign flip is sharpest around t_2^*, when $g'_t(s_2)$ peaks and the mode-2 residual transitions from large to small.