Compute-Optimal Scaling Laws

From spectra to learning curves

Princeton 2026 Summer School in Machine Learning

Module 1 left us a model for activation covariance

At a fixed layer, let X denote the random activation vector and

$$\Sigma = \mathbb{E}[X \otimes X].$$

A useful empirical model is a long spectral tail,

$$\lambda_j(\Sigma) \asymp j^{-\alpha}.$$

How does **power-law covariance** affect optimization?

Module 1 left us a model for activation covariance

At a fixed layer, let X denote the random activation vector and

$$\Sigma = \mathbb{E}[X \otimes X].$$

A useful empirical model is a long spectral tail,

$$\lambda_j(\Sigma) \asymp j^{-\alpha}.$$

How does **power-law covariance** affect optimization?

Module 1 left us a model for activation covariance

At a fixed layer, let X denote the random activation vector and

$$\Sigma = \mathbb{E}[X \otimes X].$$

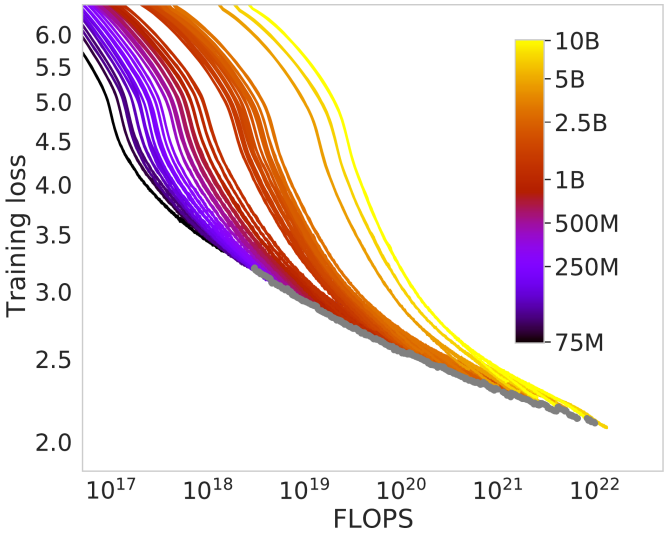
A useful empirical model is a long spectral tail,

$$\lambda_j(\Sigma) \asymp j^{-\alpha}.$$

How does **power-law covariance** affect optimization?

An organizing problem: compute-optimal training

Hoffmann et al. recover the frontier from many training runs



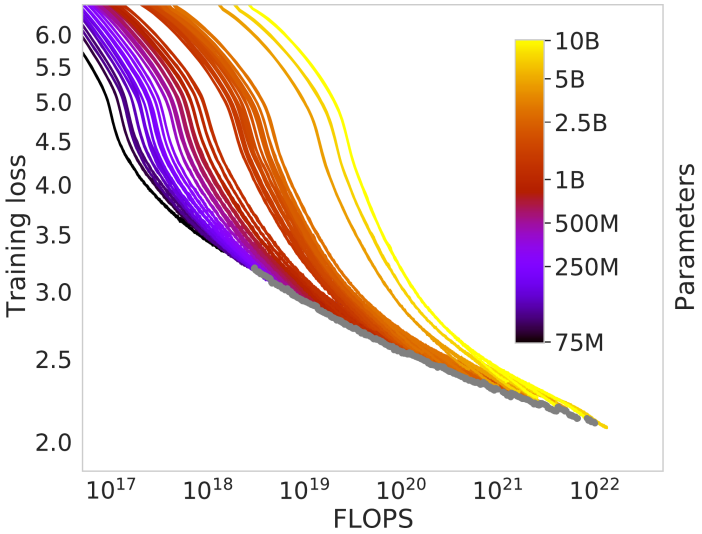
What varies

- Models from 70M to 10B parameters
- Four training horizons for each size
- Loss recorded throughout every run

What survives

At each FLOP budget, retain the smallest observed loss. The gray curve is this lower envelope.

Hoffmann et al. recover the frontier from many training runs



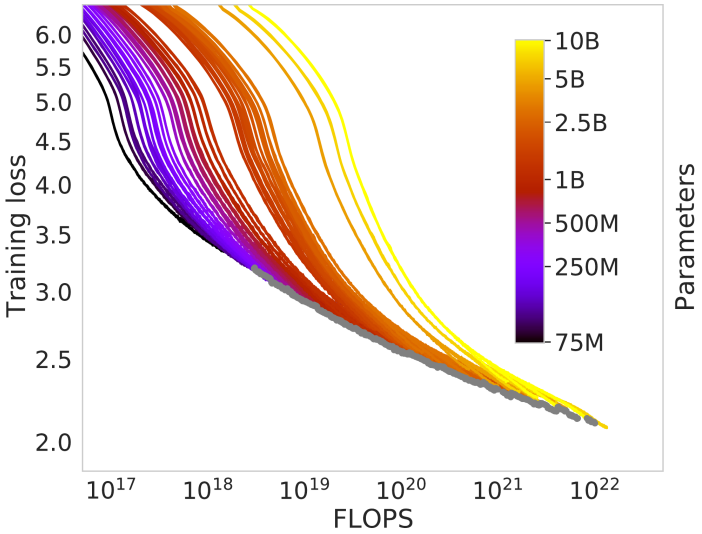
What varies

- Models from 70M to 10B parameters
- Four training horizons for each size
- Loss recorded throughout every run

What survives

At each FLOP budget, retain the smallest observed loss. The gray curve is this lower envelope.

Hoffmann et al. recover the frontier from many training runs



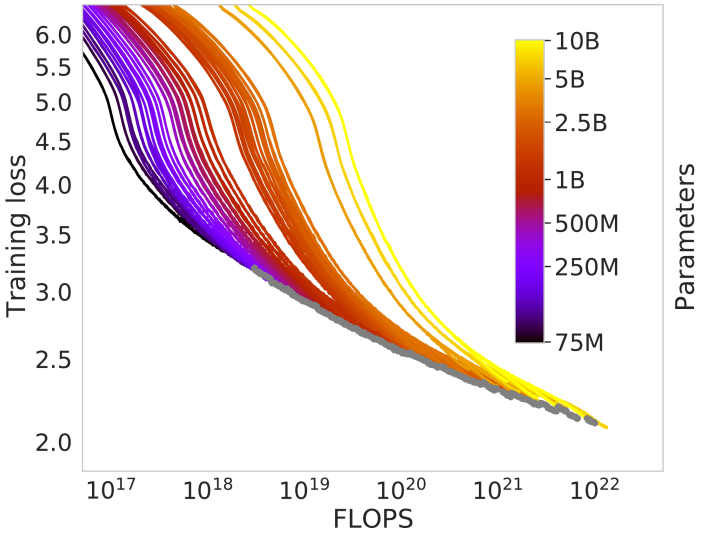
What varies

- Models from 70M to 10B parameters
- Four training horizons for each size
- Loss recorded throughout every run

What survives

At each FLOP budget, retain the smallest observed loss. The gray curve is this lower envelope.

Hoffmann et al. recover the frontier from many training runs



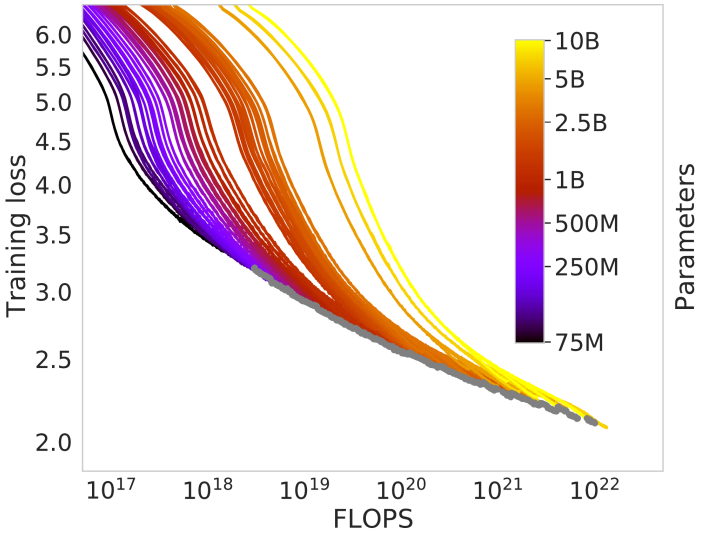
What varies

- Models from 70M to 10B parameters
- Four training horizons for each size
- Loss recorded throughout every run

What survives

At each FLOP budget, retain the smallest observed loss. The gray curve is this lower envelope.

Hoffmann et al. recover the frontier from many training runs



What varies

- Models from 70M to 10B parameters
- Four training horizons for each size
- Loss recorded throughout every run

What survives

At each FLOP budget, retain the smallest observed loss. The gray curve is this lower envelope.

The lower envelope defines the compute-optimal frontier

Our regression model. A model of size d trained on T samples has

$$\boxed{\mathsf{C} \asymp dT.}$$

At fixed C, the lower envelope selects

$$d_\star(\mathsf{C}) \in \arg\min_d \mathcal{L}\left(d, \frac{\mathsf{C}}{d}\right), \qquad T_\star(\mathsf{C}) \asymp \frac{\mathsf{C}}{d_\star(\mathsf{C})}.$$

Transformer comparison

$$\mathsf{C}_{\text{train}} \approx 6ND,$$

N: parameters
D: training tokens

An IsoFLOP experiment estimates the frontier $(d_\star(\mathsf{C}), \mathcal{L}_\star(\mathsf{C}))$ directly from training runs.

The allocation law is determined by whichever loss terms are still visible at the optimum.

The lower envelope defines the compute-optimal frontier

Our regression model. A model of size d trained on T samples has

$$\boxed{\mathsf{C} \asymp dT.}$$

At fixed C, the lower envelope selects

$$d_\star(\mathsf{C}) \in \arg\min_d \mathcal{L}\left(d, \frac{\mathsf{C}}{d}\right), \qquad T_\star(\mathsf{C}) \asymp \frac{\mathsf{C}}{d_\star(\mathsf{C})}.$$

Transformer comparison

$$\mathsf{C}_{\text{train}} \approx 6ND,$$

N: parameters

D: training tokens

An IsoFLOP experiment estimates the frontier $(d_\star(\mathsf{C}), \mathcal{L}_\star(\mathsf{C}))$ directly from training runs.

The allocation law is determined by whichever loss terms are still visible at the optimum.

The lower envelope defines the compute-optimal frontier

Our regression model. A model of size d trained on T samples has

$$\boxed{\mathsf{C} \asymp dT.}$$

At fixed C, the lower envelope selects

$$d_\star(\mathsf{C}) \in \arg\min_d \mathcal{L}\left(d, \frac{\mathsf{C}}{d}\right), \qquad T_\star(\mathsf{C}) \asymp \frac{\mathsf{C}}{d_\star(\mathsf{C})}.$$

Transformer comparison

$$\mathsf{C}_{\text{train}} \approx 6ND,$$

N: parameters

D: training tokens

An IsoFLOP experiment estimates the frontier $(d_\star(\mathsf{C}), \mathcal{L}_\star(\mathsf{C}))$ directly from training runs.

The allocation law is determined by whichever loss terms are still visible at the optimum.

The lower envelope defines the compute-optimal frontier

Our regression model. A model of size d trained on T samples has

$$\boxed{\mathsf{C} \asymp dT.}$$

At fixed C, the lower envelope selects

$$d_\star(\mathsf{C}) \in \arg\min_d \mathcal{L}\left(d, \frac{\mathsf{C}}{d}\right), \qquad T_\star(\mathsf{C}) \asymp \frac{\mathsf{C}}{d_\star(\mathsf{C})}.$$

Transformer comparison

$$\mathsf{C}_{\text{train}} \approx 6ND,$$

N: parameters

D: training tokens

An IsoFLOP experiment estimates the frontier $(d_\star(\mathsf{C}), \mathcal{L}_\star(\mathsf{C}))$ directly from training runs.

The allocation law is determined by whichever loss terms are still visible at the optimum.

The lower envelope defines the compute-optimal frontier

Our regression model. A model of size d trained on T samples has

$$\boxed{\mathsf{C} \asymp dT.}$$

At fixed C, the lower envelope selects

$$d_\star(\mathsf{C}) \in \arg\min_d \mathcal{L}\left(d, \frac{\mathsf{C}}{d}\right), \qquad T_\star(\mathsf{C}) \asymp \frac{\mathsf{C}}{d_\star(\mathsf{C})}.$$

Transformer comparison

$$\mathsf{C}_{\text{train}} \approx 6ND,$$

N: parameters

D: training tokens

An IsoFLOP experiment estimates the frontier $(d_\star(\mathsf{C}), \mathcal{L}_\star(\mathsf{C}))$ directly from training runs.

The allocation law is determined by whichever loss terms are still visible at the optimum.

Chinchilla's square-root allocation became a benchmark

Its influential compute-optimal prediction was that parameters and training tokens should grow at the same rate:

$$d_\star(C) \asymp C^{1/2}, \qquad T_\star(C) \asymp C^{1/2}.$$

The fitted frontier also gave the widely remembered training-length recommendation

$$T_\star \approx 20 d_\star,$$

roughly twenty training tokens per parameter.

Hoffmann et al. (2022).

Chinchilla's square-root allocation became a benchmark

Its influential compute-optimal prediction was that parameters and training tokens should grow at the same rate:

$$d_\star(\mathrm{C}) \asymp \mathrm{C}^{1/2}, \qquad T_\star(\mathrm{C}) \asymp \mathrm{C}^{1/2}.$$

The fitted frontier also gave the widely remembered training-length recommendation

$$T_\star \approx 20 d_\star,$$

roughly twenty training tokens per parameter.

Hoffmann et al. (2022).

Can a simple covariance model reproduce this frontier?

A solvable regression model lets us analyze the frontier

Let $Z_j \overset{\text{iid}}{\sim} \mathcal{N}(0,1)$ and define

$$X_j = j^{-\alpha/2} Z_j, \qquad \Sigma = \mathbb{E}[X \otimes X] = \text{diag}(j^{-\alpha}).$$

The supervised target is

$$Y = \langle X, b \rangle, \qquad b_j = j^{-\beta}.$$

Data exponent α How quickly variance disappears into the tail.

Target exponent β How strongly the target uses those tail directions.

A solvable regression model lets us analyze the frontier

Let $Z_j \overset{\text{iid}}{\sim} \mathcal{N}(0,1)$ and define

$$X_j = j^{-\alpha/2} Z_j, \qquad \Sigma = \mathbb{E}[X \otimes X] = \text{diag}(j^{-\alpha}).$$

The supervised target is

$$Y = \langle X, b \rangle, \qquad b_j = j^{-\beta}.$$

Data exponent α How quickly variance disappears into the tail.

Target exponent β How strongly the target uses those tail directions.

A solvable regression model lets us analyze the frontier

Let $Z_j \overset{\text{iid}}{\sim} \mathcal{N}(0,1)$ and define

$$X_j = j^{-\alpha/2} Z_j, \qquad \Sigma = \mathbb{E}[X \otimes X] = \text{diag}(j^{-\alpha}).$$

The supervised target is

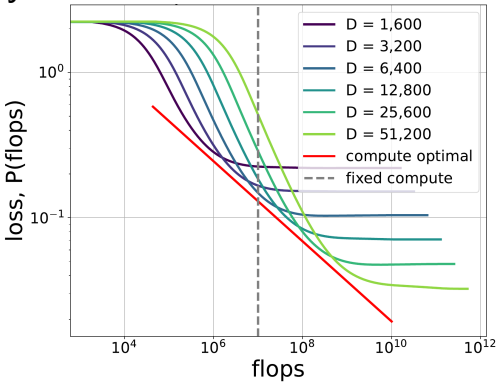
$$Y = \langle X, b \rangle, \qquad b_j = j^{-\beta}.$$

Data exponent α How quickly variance disappears into the tail.

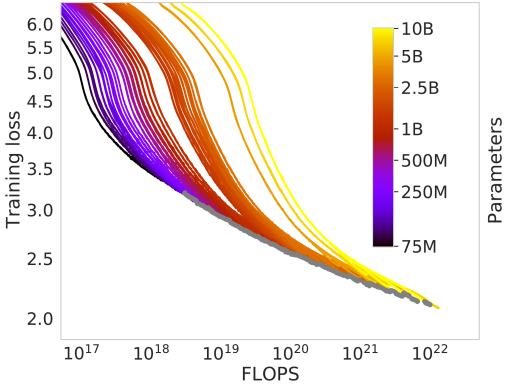
Target exponent β How strongly the target uses those tail directions.

The PLRF model mirrors Chinchilla's lower envelope

Synthetic PLRF



Chinchilla training runs

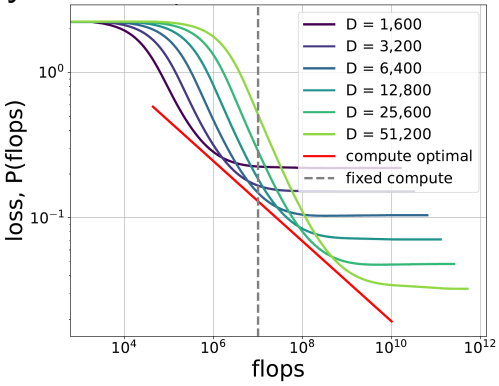


The gray envelope on the right is the empirical analogue of the red compute-optimal frontier on the left.

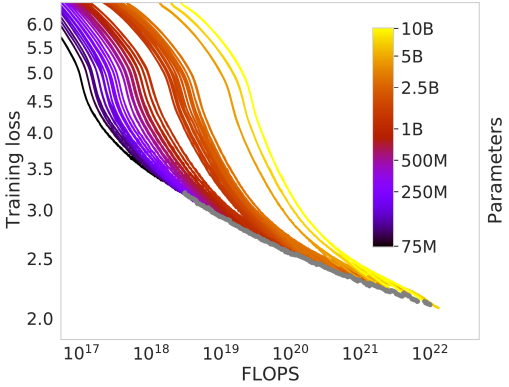
Left: course PLRF example, $\alpha = 0.8$, $\beta = 0.4$. Right: Hoffmann et al. (2022), Figure 2.

The PLRF model mirrors Chinchilla's lower envelope

Synthetic PLRF



Chinchilla training runs



The gray envelope on the right is the empirical analogue of the red compute-optimal frontier on the left.

Left: course PLRF example, $\alpha = 0.8,\ \beta = 0.4$. Right: Hoffmann et al. (2022), Figure 2.

Random features turn ambient structure into model capacity

Use a Gaussian feature map

$$W \in \mathbb{R}^{v \times d}, \qquad W_{jk} \overset{\text{iid}}{\sim} \mathcal{N}(0, 1/d), \qquad \phi(X) = W^\top X.$$

The predictor $\widehat{Y}_\theta = \langle W^\top X, \theta \rangle$ has population loss

$$\mathcal{L}(\theta) = \mathbb{E}[(\widehat{Y}_\theta - Y)^2] = \left\| \Sigma^{1/2}(W\theta - b) \right\|_2^2.$$

What W does: it induces variable model capacity by restricting the learnable target to $\mathrm{col}(W)$. *And more...*

Random features turn ambient structure into model capacity

Use a Gaussian feature map

$$W \in \mathbb{R}^{v \times d}, \qquad W_{jk} \overset{\text{iid}}{\sim} \mathcal{N}(0, 1/d), \qquad \phi(X) = W^\top X.$$

The predictor $\widehat{Y}_\theta = \langle W^\top X, \theta \rangle$ has population loss

$$\mathcal{L}(\theta) = \mathbb{E}[(\widehat{Y}_\theta - Y)^2] = \left\| \Sigma^{1/2}(W\theta - b) \right\|_2^2.$$

What W does: it induces variable model capacity by restricting the learnable target to $\mathrm{col}(W)$. *And more...*

Random features turn ambient structure into model capacity

Use a Gaussian feature map

$$W \in \mathbb{R}^{v \times d}, \qquad W_{jk} \overset{\text{iid}}{\sim} \mathcal{N}(0, 1/d), \qquad \phi(X) = W^\top X.$$

The predictor $\widehat{Y}_\theta = \langle W^\top X, \theta \rangle$ has population loss

$$\mathcal{L}(\theta) = \mathbb{E}[(\widehat{Y}_\theta - Y)^2] = \left\| \Sigma^{1/2}(W\theta - b) \right\|_2^2.$$

What W **does:** it induces variable model capacity by restricting the learnable target to $\mathrm{col}(W)$. *And more...*

We train with fresh-mini-batch SGD

Condition on one draw of W. At update r, draw a fresh mini-batch

$$\{(X_{r,i}, Y_{r,i})\}_{i=1}^{B} \overset{\text{iid}}{\sim} (X, Y).$$

Define the batch loss and take one SGD step:

$$\widehat{\mathcal{L}}_r(\theta) = \frac{1}{B}\sum_{i=1}^{B}(\langle W^\top X_{r,i}, \theta\rangle - Y_{r,i})^2, \qquad \theta_{r+1} = \theta_r - \gamma\nabla\widehat{\mathcal{L}}_r(\theta_r).$$

One pass: batches are never reused. After r updates we have seen $T = Br$ training examples, so at fixed B, tokens and steps are equivalent clocks.

We train with fresh-mini-batch SGD

Condition on one draw of W. At update r, draw a fresh mini-batch

$$\{(X_{r,i}, Y_{r,i})\}_{i=1}^{B} \overset{\text{iid}}{\sim} (X, Y).$$

Define the batch loss and take one SGD step:

$$\widehat{\mathcal{L}}_r(\theta) = \frac{1}{B}\sum_{i=1}^{B}(\langle W^\top X_{r,i}, \theta\rangle - Y_{r,i})^2, \qquad \theta_{r+1} = \theta_r - \gamma \nabla \widehat{\mathcal{L}}_r(\theta_r).$$

One pass: batches are never reused. After r updates we have seen $T = Br$ training examples, so at fixed B, tokens and steps are equivalent clocks.

We train with fresh-mini-batch SGD

Condition on one draw of W. At update r, draw a fresh mini-batch

$$\{(X_{r,i}, Y_{r,i})\}_{i=1}^{B} \overset{\text{iid}}{\sim} (X, Y).$$

Define the batch loss and take one SGD step:

$$\widehat{\mathcal{L}}_r(\theta) = \frac{1}{B}\sum_{i=1}^{B}(\langle W^\top X_{r,i}, \theta\rangle - Y_{r,i})^2, \qquad \theta_{r+1} = \theta_r - \gamma\nabla\widehat{\mathcal{L}}_r(\theta_r).$$

One pass: batches are never reused. After r updates we have seen $T = Br$ training examples, so at fixed B, tokens and steps are equivalent clocks.

The Hessian is a deformed sample covariance

Conditioning on W, the feature-space Hessian is

$$\widehat{K} = W^\top \Sigma W.$$

Its nonzero eigenvalues equal those of

$$H = \Sigma^{1/2} W W^\top \Sigma^{1/2}, \qquad \mathrm{spec}_+(\widehat{K}) = \mathrm{spec}_+(H).$$

Learning curves use $(I - \gamma \widehat{K})^r$ and $(\widehat{K} - zI)^{-1}$.

The optimizer sees a random sample-covariance resolvent, not Σ directly.

The Hessian is a deformed sample covariance

Conditioning on W, the feature-space Hessian is

$$\widehat{K} = W^\top \Sigma W.$$

Its nonzero eigenvalues equal those of

$$H = \Sigma^{1/2} W W^\top \Sigma^{1/2}, \qquad \mathrm{spec}_+(\widehat{K}) = \mathrm{spec}_+(H).$$

Learning curves use $(I - \gamma \widehat{K})^r$ and $(\widehat{K} - zI)^{-1}$.

The optimizer sees a random sample-covariance resolvent, not Σ directly.

The Hessian is a deformed sample covariance

Conditioning on W, the feature-space Hessian is

$$\widehat{K} = W^\top \Sigma W.$$

Its nonzero eigenvalues equal those of

$$H = \Sigma^{1/2} W W^\top \Sigma^{1/2}, \qquad \mathrm{spec}_+(\widehat{K}) = \mathrm{spec}_+(H).$$

Learning curves use $(I - \gamma \widehat{K})^r$ and $(\widehat{K} - zI)^{-1}$.

The optimizer sees a random sample-covariance resolvent, not Σ directly.

The Hessian is a deformed sample covariance

Conditioning on W, the feature-space Hessian is

$$\widehat{K} = W^\top \Sigma W.$$

Its nonzero eigenvalues equal those of

$$H = \Sigma^{1/2} W W^\top \Sigma^{1/2}, \qquad \mathrm{spec}_+(\widehat{K}) = \mathrm{spec}_+(H).$$

Learning curves use $(I - \gamma \widehat{K})^r$ and $(\widehat{K} - zI)^{-1}$.

The optimizer sees a random sample-covariance resolvent, not Σ directly.

Random matrix lens

Marchenko–Pastur contracts toward the identity

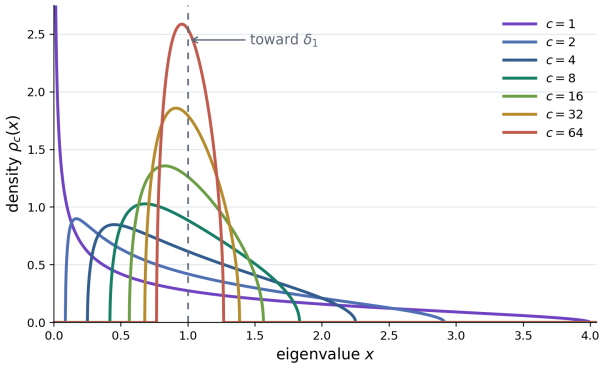
Let $G \in \mathbb{R}^{v \times d}$ have iid standard Gaussian entries and set

$$S = \frac{1}{v} G^\top G, \qquad c = \frac{v}{d} \geq 1, \qquad \mathbb{E}[S] = I$$

Marchenko–Pastur. The limiting spectrum is supported on

$$\left[(1 - c^{-1/2})^2, (1 + c^{-1/2})^2\right].$$

As $c \to \infty$, the law contracts to δ_1.



For PLRF, we need one small generalization: **deformed Marchenko–Pastur**.

Marchenko–Pastur contracts toward the identity

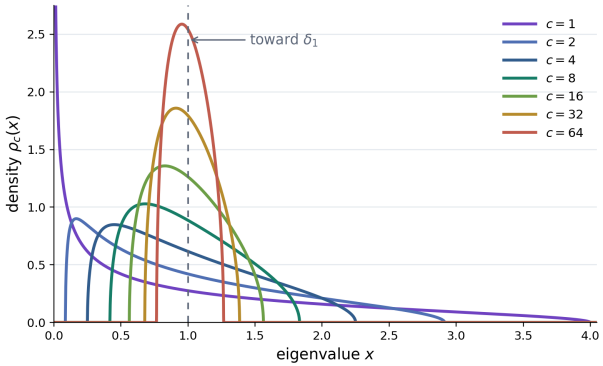
Let $G \in \mathbb{R}^{v \times d}$ have iid standard Gaussian entries and set

$$S = \frac{1}{v} G^\top G, \qquad c = \frac{v}{d} \geq 1, \qquad \mathbb{E}[S] = I$$

Marchenko–Pastur. The limiting spectrum is supported on

$$\left[(1 - c^{-1/2})^2, (1 + c^{-1/2})^2\right].$$

As $c \to \infty$, the law contracts to δ_1.



For PLRF, we need one small generalization: **deformed Marchenko–Pastur**.

Marchenko–Pastur contracts toward the identity

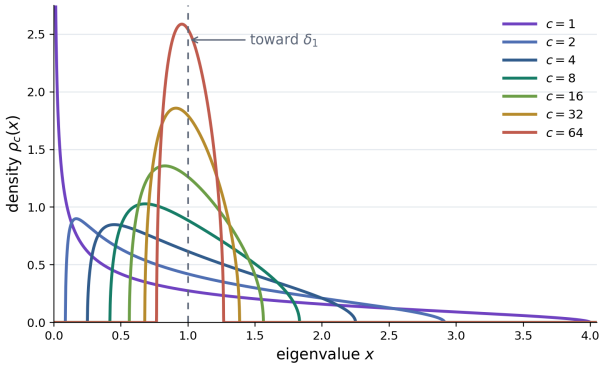
Let $G \in \mathbb{R}^{v \times d}$ have iid standard Gaussian entries and set

$$S = \frac{1}{v} G^\top G, \qquad c = \frac{v}{d} \geq 1, \qquad \mathbb{E}[S] = I$$

Marchenko–Pastur. The limiting spectrum is supported on

$$\left[(1 - c^{-1/2})^2, (1 + c^{-1/2})^2\right].$$

As $c \to \infty$, the law contracts to δ_1.



For PLRF, we need one small generalization: **deformed Marchenko–Pastur**.

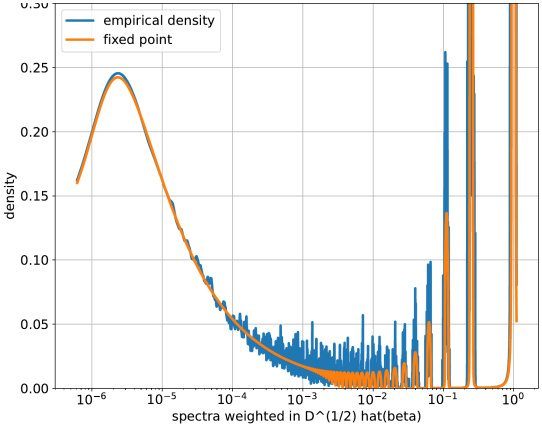
We need the deformed Marchenko–Pastur law

Restore a nontrivial population covariance:

$$\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_v), \qquad W = d^{-1/2} G,$$

$$H = \Lambda^{1/2} W W^\top \Lambda^{1/2}.$$

Deformed MP (informal). The spectrum of H converges to a deterministic law depending on Λ through one scalar self-consistent equation.



Power-law example: $\alpha = 2,\ d = 1000,\ v = 10000.$

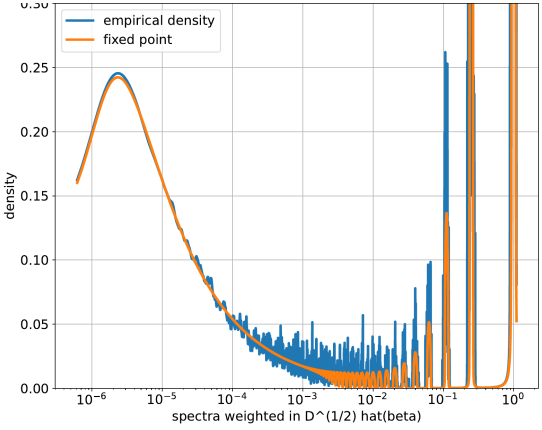
We need the deformed Marchenko–Pastur law

Restore a nontrivial population covariance:

$$\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_v), \qquad W = d^{-1/2} G,$$

$$H = \Lambda^{1/2} W W^\top \Lambda^{1/2}.$$

Deformed MP (informal). The spectrum of H converges to a deterministic law depending on Λ through one scalar self-consistent equation.



Power-law example: $\alpha = 2,\ d = 1000,\ v = 10000.$

The resolvent turns spectral questions into analysis

Write

$$\Lambda = \mathrm{diag}(\lambda_1, \dots, \lambda_v), \quad W = d^{-1/2} G, \quad H = \Lambda^{1/2} W W^\top \Lambda^{1/2}, \quad R(z) = (H - zI)^{-1}.$$

For $\mathrm{Im}\, z > 0$,

$$\frac{1}{v} \mathrm{Tr}\, R(z) = \frac{1}{v} \sum_{\ell=1}^{v} \frac{1}{x_\ell - z}.$$

Contour integrals of $R(z)$ recover spectral densities, projectors, and the training propagators entering the loss.

The resolvent turns spectral questions into analysis

Write

$$\Lambda = \mathrm{diag}(\lambda_1, \dots, \lambda_v), \quad W = d^{-1/2} G, \quad H = \Lambda^{1/2} W W^\top \Lambda^{1/2}, \quad R(z) = (H - zI)^{-1}.$$

For $\mathrm{Im}\, z > 0$,

$$\frac{1}{v} \mathrm{Tr}\, R(z) = \frac{1}{v} \sum_{\ell=1}^{v} \frac{1}{x_\ell - z}.$$

Contour integrals of $R(z)$ recover spectral densities, projectors, and the training propagators entering the loss.

The resolvent turns spectral questions into analysis

Write

$$\Lambda = \operatorname{diag}(\lambda_1, \dots, \lambda_v), \quad W = d^{-1/2} G, \quad H = \Lambda^{1/2} W W^\top \Lambda^{1/2}, \quad R(z) = (H - zI)^{-1}.$$

For $\operatorname{Im} z > 0$,

$$\frac{1}{v} \operatorname{Tr} R(z) = \frac{1}{v} \sum_{\ell=1}^{v} \frac{1}{x_\ell - z}.$$

Contour integrals of $R(z)$ recover spectral densities, projectors, and the training propagators entering the loss.

The matrix derivative is the only calculus we need

Start from

$$RH = I + zR, \qquad RH = (R\Lambda^{1/2}W)(W^\top\Lambda^{1/2}).$$

For $W \mapsto W + \varepsilon\Delta$,

$$\partial_\Delta H = \Lambda^{1/2}(\Delta W^\top + W\Delta^\top)\Lambda^{1/2}, \qquad \boxed{\partial_\Delta R = -R(\partial_\Delta H)R.}$$

The factorization exposes one Gaussian matrix; the directional derivative tracks how the resolvent changes when that matrix moves.

The matrix derivative is the only calculus we need

Start from

$$RH = I + zR, \qquad RH = (R\Lambda^{1/2}W)(W^\top\Lambda^{1/2}).$$

For $W \mapsto W + \varepsilon\Delta$,

$$\partial_\Delta H = \Lambda^{1/2}(\Delta W^\top + W\Delta^\top)\Lambda^{1/2}, \qquad \boxed{\partial_\Delta R = -R(\partial_\Delta H)R.}$$

The factorization exposes one Gaussian matrix; the directional derivative tracks how the resolvent changes when that matrix moves.

The matrix derivative is the only calculus we need

Start from

$$RH = I + zR, \qquad RH = (R\Lambda^{1/2}W)(W^\top\Lambda^{1/2}).$$

For $W \mapsto W + \varepsilon\Delta$,

$$\partial_\Delta H = \Lambda^{1/2}(\Delta W^\top + W\Delta^\top)\Lambda^{1/2}, \qquad \boxed{\partial_\Delta R = -R(\partial_\Delta H)R.}$$

The factorization exposes one Gaussian matrix; the directional derivative tracks how the resolvent changes when that matrix moves.

Gaussian integration by parts can be matrix-valued

Let $\widetilde{W}$ be an independent copy of W. For a smooth compatible matrix function F,

$$\boxed{\mathbb{E}[F(W)W^\top] = \mathbb{E}[(\partial_{\widetilde{W}} F(W))\widetilde{W}^\top].}$$

Take $F(W) = R(W)\Lambda^{1/2}W$, so the exposed product is $F(W)W^\top\Lambda^{1/2} = RH$. Using $\partial R = -R(\partial H)R$, the identity becomes

$$\boxed{\begin{aligned}\mathbb{E}[RH] = \mathbb{E}\Big[&R\Lambda^{1/2}\widetilde{W}\widetilde{W}^\top\Lambda^{1/2} \\ &- R\Lambda^{1/2}(\widetilde{W}W^\top + W\widetilde{W}^\top)\Lambda^{1/2}R\Lambda^{1/2}W\widetilde{W}^\top\Lambda^{1/2}\Big].\end{aligned}}$$

Gaussian integration by parts can be matrix-valued

Let $\widetilde{W}$ be an independent copy of W. For a smooth compatible matrix function F,

$$\boxed{\mathbb{E}[F(W)W^\top] = \mathbb{E}[(\partial_{\widetilde{W}} F(W))\widetilde{W}^\top].}$$

Take $F(W) = R(W)\Lambda^{1/2}W$, so the exposed product is $F(W)W^\top\Lambda^{1/2} = RH$. Using $\partial R = -R(\partial H)R$, the identity becomes

$$\boxed{\begin{aligned}\mathbb{E}[RH] = \mathbb{E}\Big[&R\Lambda^{1/2}\widetilde{W}\widetilde{W}^\top\Lambda^{1/2} \\ &- R\Lambda^{1/2}(\widetilde{W}W^\top + W\widetilde{W}^\top)\Lambda^{1/2}R\Lambda^{1/2}W\widetilde{W}^\top\Lambda^{1/2}\Big].\end{aligned}}$$

Gaussian integration by parts can be matrix-valued

Let $\widetilde{W}$ be an independent copy of W. For a smooth compatible matrix function F,

$$\boxed{\mathbb{E}[F(W)W^\top] = \mathbb{E}[(\partial_{\widetilde{W}} F(W))\widetilde{W}^\top].}$$

Take $F(W) = R(W)\Lambda^{1/2}W$, so the exposed product is $F(W)W^\top\Lambda^{1/2} = RH$. Using $\partial R = -R(\partial H)R$, the identity becomes

$$\boxed{\begin{aligned}\mathbb{E}[RH] = \mathbb{E}\Big[&R\Lambda^{1/2}\widetilde{W}\widetilde{W}^\top\Lambda^{1/2} \\ &- R\Lambda^{1/2}(\widetilde{W}W^\top + W\widetilde{W}^\top)\Lambda^{1/2}R\Lambda^{1/2}W\widetilde{W}^\top\Lambda^{1/2}\Big].\end{aligned}}$$

Two Wick contractions have different geometry

Condition on W. The purple factors are paired under $\mathbb{E}_{\widetilde{W}}$:

$$\mathbb{E}[RH] = \mathbb{E}\Big[R\Lambda^{1/2}\widetilde{W}\widetilde{W}^\top\Lambda^{1/2} - R\Lambda^{1/2}(\widetilde{W}W^\top + W\widetilde{W}^\top)\Lambda^{1/2}R\Lambda^{1/2}W\widetilde{W}^\top\Lambda^{1/2}\Big].$$

Coherent contraction

$$\mathbb{E}_{\widetilde{W}}[\widetilde{W}A\widetilde{W}^\top] = \frac{1}{d}\operatorname{Tr}(A)I.$$

Incoherent contraction

$$\mathbb{E}_{\widetilde{W}}[\widetilde{W}^\top BC\widetilde{W}^\top] = \frac{1}{d}C^\top B^\top.$$

The first pairing closes a trace; the second leaves a matrix correction.

Two Wick contractions have different geometry

Condition on W. The purple factors are paired under $\mathbb{E}_{\widetilde{W}}$:

$$\mathbb{E}[RH] = \mathbb{E}\Big[R\Lambda^{1/2}\widetilde{W}\widetilde{W}^\top\Lambda^{1/2} - R\Lambda^{1/2}(\widetilde{W}W^\top + W\widetilde{W}^\top)\Lambda^{1/2}R\Lambda^{1/2}W\widetilde{W}^\top\Lambda^{1/2}\Big].$$

Coherent contraction

$$\mathbb{E}_{\widetilde{W}}[\widetilde{W}A\widetilde{W}^\top] = \frac{1}{d}\operatorname{Tr}(A)I.$$

Incoherent contraction

$$\mathbb{E}_{\widetilde{W}}[\widetilde{W}^\top BC\widetilde{W}^\top] = \frac{1}{d}C^\top B^\top.$$

The first pairing closes a trace; the second leaves a matrix correction.

Two Wick contractions have different geometry

Condition on W. The purple factors are paired under $\mathbb{E}_{\widetilde{W}}$:

$$\mathbb{E}[RH] = \mathbb{E}\Big[R\Lambda^{1/2}\widetilde{W}\widetilde{W}^\top\Lambda^{1/2} - R\Lambda^{1/2}(\widetilde{W}W^\top + W\widetilde{W}^\top)\Lambda^{1/2}R\Lambda^{1/2}W\widetilde{W}^\top\Lambda^{1/2}\Big].$$

Coherent contraction

$$\mathbb{E}_{\widetilde{W}}[\widetilde{W}A\widetilde{W}^\top] = \frac{1}{d}\operatorname{Tr}(A)I.$$

Incoherent contraction

$$\mathbb{E}_{\widetilde{W}}[\widetilde{W}^\top BC\widetilde{W}^\top] = \frac{1}{d}C^\top B^\top.$$

The first pairing closes a trace; the second leaves a matrix correction.

Two Wick contractions have different geometry

Condition on W. The purple factors are paired under $\mathbb{E}_{\widetilde{W}}$:

$$\mathbb{E}[RH] = \mathbb{E}\Big[R\Lambda^{1/2}\widetilde{W}\widetilde{W}^\top\Lambda^{1/2} - R\Lambda^{1/2}(\widetilde{W}W^\top + W\widetilde{W}^\top)\Lambda^{1/2}R\Lambda^{1/2}W\widetilde{W}^\top\Lambda^{1/2}\Big].$$

Coherent contraction

$$\mathbb{E}_{\widetilde{W}}[\widetilde{W}A\widetilde{W}^\top] = \frac{1}{d}\operatorname{Tr}(A)I.$$

Incoherent contraction

$$\mathbb{E}_{\widetilde{W}}[\widetilde{W}^\top BC\widetilde{W}^\top] = \frac{1}{d}C^\top B^\top.$$

The first pairing closes a trace; the second leaves a matrix correction.

The coherent term creates the self-consistent scalar

The two contractions give the exact master equation

$$\boxed{\mathbb{E}[RH] = \mathbb{E}\left[q_d(z)R\Lambda - \frac{1}{d}RHR\Lambda\right], \qquad q_d(z) = 1 - \frac{1}{d}\operatorname{Tr}(RH).}$$

Using $RH = I + zR$, its jth diagonal entry is

$$\mathbb{E}[(\lambda_j q_d - z)R_{jj}] = 1 + \frac{\lambda_j}{d}\mathbb{E}[(RHR)_{jj}].$$

The coherent contraction produces q_d; the incoherent contraction is the explicit d^{-1} correction.

The coherent term creates the self-consistent scalar

The two contractions give the exact master equation

$$\boxed{\mathbb{E}[RH] = \mathbb{E}\left[q_d(z)R\Lambda - \frac{1}{d}RHR\Lambda\right], \qquad q_d(z) = 1 - \frac{1}{d}\operatorname{Tr}(RH).}$$

Using $RH = I + zR$, its jth diagonal entry is

$$\mathbb{E}[(\lambda_j q_d - z)R_{jj}] = 1 + \frac{\lambda_j}{d}\mathbb{E}[(RHR)_{jj}].$$

The coherent contraction produces q_d; the incoherent contraction is the explicit d^{-1} correction.

The coherent term creates the self-consistent scalar

The two contractions give the exact master equation

$$\boxed{\mathbb{E}[RH] = \mathbb{E}\left[q_d(z)R\Lambda - \frac{1}{d}RHR\Lambda\right], \qquad q_d(z) = 1 - \frac{1}{d}\operatorname{Tr}(RH).}$$

Using $RH = I + zR$, its jth diagonal entry is

$$\mathbb{E}[(\lambda_j q_d - z)R_{jj}] = 1 + \frac{\lambda_j}{d}\mathbb{E}[(RHR)_{jj}].$$

The coherent contraction produces q_d; the incoherent contraction is the explicit d^{-1} correction.

Trace concentration closes the random equation

For fixed $\operatorname{Im} z > 0$,

$$\|R(z)\| \le \frac{1}{\operatorname{Im} z}, \qquad \operatorname{Var}\left(\frac{1}{d}\operatorname{Tr}(\Lambda R(z))\right) \longrightarrow 0.$$

Drop the explicit d^{-1} term and factor the concentrated trace:

$$\mathcal{R}_{jj}(z) = \frac{1}{\lambda_j m(z) - z}, \qquad m(z) = \left[1 + \frac{1}{d}\sum_{j=1}^{v} \frac{\lambda_j}{\lambda_j m(z) - z}\right]^{-1}.$$

A large random matrix has been replaced by one nonlinear scalar equation.

Trace concentration closes the random equation

For fixed $\operatorname{Im} z > 0$,

$$\|R(z)\| \le \frac{1}{\operatorname{Im} z}, \qquad \operatorname{Var}\left(\frac{1}{d}\operatorname{Tr}(\Lambda R(z))\right) \longrightarrow 0.$$

Drop the explicit d^{-1} term and factor the concentrated trace:

$$\boxed{\mathcal{R}_{jj}(z) = \frac{1}{\lambda_j m(z) - z}, \qquad m(z) = \left[1 + \frac{1}{d}\sum_{j=1}^{v} \frac{\lambda_j}{\lambda_j m(z) - z}\right]^{-1}.}$$

A large random matrix has been replaced by one nonlinear scalar equation.

Trace concentration closes the random equation

For fixed $\operatorname{Im} z > 0$,

$$\|R(z)\| \le \frac{1}{\operatorname{Im} z}, \qquad \operatorname{Var}\left(\frac{1}{d}\operatorname{Tr}(\Lambda R(z))\right) \longrightarrow 0.$$

Drop the explicit d^{-1} term and factor the concentrated trace:

$$\boxed{\mathcal{R}_{jj}(z) = \frac{1}{\lambda_j m(z) - z}, \qquad m(z) = \left[1 + \frac{1}{d}\sum_{j=1}^{v} \frac{\lambda_j}{\lambda_j m(z) - z}\right]^{-1}.}$$

A large random matrix has been replaced by one nonlinear scalar equation.

Least squares is a zero-resolvent limit

$$\mathscr{F}_0(d) := \min_{\theta} \left\| \Lambda^{1/2}(W\theta - b) \right\|_2^2 \asymp d^{-\alpha+\max\{0,1-2\beta\}}.$$

★ Great exercise!

The glue is the zero-resolvent limit. Let $y = \Lambda^{1/2} b$. Then

$$\begin{aligned} \mathscr{F}_0 &= \lim_{\eta \downarrow 0} \eta\, y^\top R(-\eta) y \\ &\simeq \lim_{\eta \downarrow 0} \sum_{j=1}^{v} \frac{\eta \lambda_j b_j^2}{\lambda_j m(-\eta) + \eta}. \end{aligned}$$

Use the self-consistent equation for $m(-\eta)$ and analyze the limit $\eta \downarrow 0$.

Least squares is a zero-resolvent limit

$$\mathscr{F}_0(d) := \min_{\theta} \left\| \Lambda^{1/2}(W\theta - b) \right\|_2^2 \asymp d^{-\alpha+\max\{0,1-2\beta\}}.$$

★ Great exercise!

The glue is the zero-resolvent limit. Let $y = \Lambda^{1/2} b$. Then

$$\begin{aligned} \mathscr{F}_0 &= \lim_{\eta \downarrow 0} \eta\, y^\top R(-\eta) y \\ &\simeq \lim_{\eta \downarrow 0} \sum_{j=1}^{v} \frac{\eta \lambda_j b_j^2}{\lambda_j m(-\eta) + \eta}. \end{aligned}$$

Use the self-consistent equation for $m(-\eta)$ and analyze the limit $\eta \downarrow 0$.

Least squares is a zero-resolvent limit

$$\mathscr{F}_0(d) := \min_{\theta} \left\| \Lambda^{1/2}(W\theta - b) \right\|_2^2 \asymp d^{-\alpha+\max\{0,1-2\beta\}}.$$

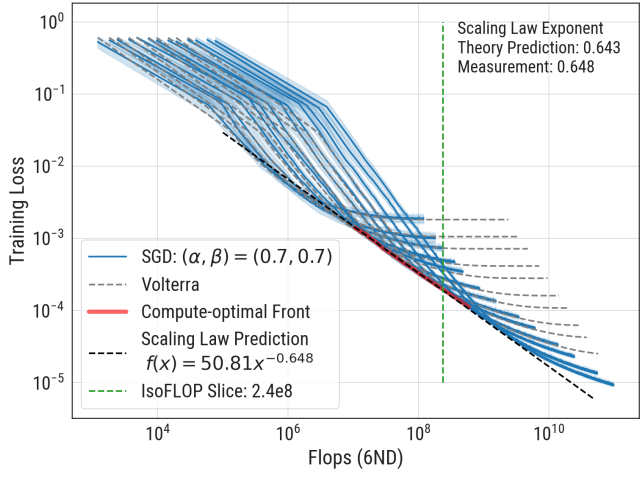
★ Great exercise!

The glue is the zero-resolvent limit. Let $y = \Lambda^{1/2} b$. Then

$$\begin{aligned} \mathscr{F}_0 &= \lim_{\eta \downarrow 0} \eta\, y^\top R(-\eta) y \\ &\simeq \lim_{\eta \downarrow 0} \sum_{j=1}^{v} \frac{\eta \lambda_j b_j^2}{\lambda_j m(-\eta) + \eta}. \end{aligned}$$

Use the self-consistent equation for $m(-\eta)$ and analyze the limit $\eta \downarrow 0$.

Dynamics: a scalar Volterra equation



SGD risk satisfies a scalar Volterra equation

Condition on W. Let $\theta^\star_W$ minimize $\mathcal{L}$, set $\delta_r = \theta_r - \theta^\star_W$, and let $\widehat{K} u_j = \lambda_j u_j$. The recursion closes on the **population risk**:

$$\boxed{P_r := \mathbb{E}_{\mathrm{SGD}}[\mathcal{L}(\theta_r) \mid W] = \mathscr{F}_0 + \mathbb{E}_{\mathrm{SGD}}[\delta_r^\top \widehat{K} \delta_r \mid W], \qquad \widehat{K} = W^\top \Lambda W.}$$

For Gaussian mini-batches, write $q_{0,j} = \mathbb{E}_{\mathrm{SGD}}[\langle \delta_0, u_j \rangle^2 \mid W]$ and $a_j = (1-\gamma\lambda_j)^2 + (\gamma^2/B)\lambda_j^2$. Then

$$\boxed{\begin{aligned} P_r &= F_r + \sum_{s=0}^{r-1} K_{r-1-s} P_s, \\ F_r &= \mathscr{F}_0 + \sum_j \lambda_j a_j^r q_{0,j}, \qquad K_\ell = \frac{\gamma^2}{B} \sum_j \lambda_j^2 a_j^\ell. \end{aligned}}$$

SGD risk satisfies a scalar Volterra equation

Condition on W. Let $\theta^\star_W$ minimize $\mathcal{L}$, set $\delta_r = \theta_r - \theta^\star_W$, and let $\widehat{K} u_j = \lambda_j u_j$. The recursion closes on the **population risk**:

$$\boxed{P_r := \mathbb{E}_{\mathrm{SGD}}[\mathcal{L}(\theta_r) \mid W] = \mathscr{F}_0 + \mathbb{E}_{\mathrm{SGD}}[\delta_r^\top \widehat{K} \delta_r \mid W], \qquad \widehat{K} = W^\top \Lambda W.}$$

For Gaussian mini-batches, write $q_{0,j} = \mathbb{E}_{\mathrm{SGD}}[\langle \delta_0, u_j \rangle^2 \mid W]$ and $a_j = (1 - \gamma\lambda_j)^2 + (\gamma^2/B)\lambda_j^2$. Then

$$\boxed{\begin{aligned} P_r &= F_r + \sum_{s=0}^{r-1} K_{r-1-s} P_s, \\ F_r &= \mathscr{F}_0 + \sum_j \lambda_j a_j^r q_{0,j}, \qquad K_\ell = \frac{\gamma^2}{B} \sum_j \lambda_j^2 a_j^\ell. \end{aligned}}$$

Forcing and kernel separate descent from volatility

Forcing F_r

- mean-gradient relaxation;
- target overlap with each spectral mode;
- finite-feature approximation error.

Kernel K_r

- variance injected by a mini-batch;
- propagated through later iterations;
- fed back in proportion to the current loss.

Random matrix theory makes the random F and K deterministic. The Volterra equation then propagates those spectral inputs through time.

Forcing and kernel separate descent from volatility

Forcing F_r

- mean-gradient relaxation;
- target overlap with each spectral mode;
- finite-feature approximation error.

Kernel K_r

- variance injected by a mini-batch;
- propagated through later iterations;
- fed back in proportion to the current loss.

Random matrix theory makes the random F and K deterministic. The Volterra equation then propagates those spectral inputs through time.

Forcing and kernel separate descent from volatility

Forcing F_r

- mean-gradient relaxation;
- target overlap with each spectral mode;
- finite-feature approximation error.

Kernel K_r

- variance injected by a mini-batch;
- propagated through later iterations;
- fed back in proportion to the current loss.

Random matrix theory makes the random F and K deterministic. The Volterra equation then propagates those spectral inputs through time.

Forcing and kernel separate descent from volatility

Forcing F_r

- mean-gradient relaxation;
- target overlap with each spectral mode;
- finite-feature approximation error.

Kernel K_r

- variance injected by a mini-batch;
- propagated through later iterations;
- fed back in proportion to the current loss.

Random matrix theory makes the random F and K deterministic. The Volterra equation then propagates those spectral inputs through time.

Forcing and kernel separate descent from volatility

Forcing F_r

- mean-gradient relaxation;
- target overlap with each spectral mode;
- finite-feature approximation error.

Kernel K_r

- variance injected by a mini-batch;
- propagated through later iterations;
- fed back in proportion to the current loss.

Random matrix theory makes the random F and K deterministic. The Volterra equation then propagates those spectral inputs through time.

Forcing and kernel separate descent from volatility

Forcing F_r

- mean-gradient relaxation;
- target overlap with each spectral mode;
- finite-feature approximation error.

Kernel K_r

- variance injected by a mini-batch;
- propagated through later iterations;
- fed back in proportion to the current loss.

Random matrix theory makes the random F and K deterministic. The Volterra equation then propagates those spectral inputs through time.

Forcing and kernel separate descent from volatility

Forcing F_r

- mean-gradient relaxation;
- target overlap with each spectral mode;
- finite-feature approximation error.

Kernel K_r

- variance injected by a mini-batch;
- propagated through later iterations;
- fed back in proportion to the current loss.

Random matrix theory makes the random F and K deterministic. The Volterra equation then propagates those spectral inputs through time.

Forcing and kernel separate descent from volatility

Forcing F_r

- mean-gradient relaxation;
- target overlap with each spectral mode;
- finite-feature approximation error.

Kernel K_r

- variance injected by a mini-batch;
- propagated through later iterations;
- fed back in proportion to the current loss.

Random matrix theory makes the random F and K deterministic. The Volterra equation then propagates those spectral inputs through time.

Forcing and kernel separate descent from volatility

Forcing F_r

- mean-gradient relaxation;
- target overlap with each spectral mode;
- finite-feature approximation error.

Kernel K_r

- variance injected by a mini-batch;
- propagated through later iterations;
- fed back in proportion to the current loss.

Random matrix theory makes the random F and K deterministic. The Volterra equation then propagates those spectral inputs through time.

Stability is collective, not only a top-eigenvalue condition

Mean-square stability

$$\|K\|_1 = \frac{\gamma}{B} \sum_i \frac{\lambda_i}{2 - \gamma(1 + B^{-1})\lambda_i} < 1.$$

Collective scale

$$r_{\mathrm{eff}}(H) = \frac{\mathrm{Tr}(H)}{\lambda_1(H)},$$

$$B_{\mathrm{crit}} \asymp r_{\mathrm{eff}}(H).$$

Small batch: $B < B_{\mathrm{crit}}$

$$\|K\|_1 \asymp \frac{\gamma}{B}\,\mathrm{Tr}(H), \qquad \gamma_{\max} \asymp \frac{B}{\mathrm{Tr}(H)}.$$

Trace-controlled: the maximum stable step size grows linearly with B.

Large batch: $B > B_{\mathrm{crit}}$

$$\gamma\lambda_1(H) \lesssim 1, \qquad \gamma_{\max} \asymp \frac{1}{\lambda_1(H)}.$$

Top-mode-controlled: the batch-size gain has saturated.

Stability is collective, not only a top-eigenvalue condition

Mean-square stability

$$\|K\|_1 = \frac{\gamma}{B} \sum_i \frac{\lambda_i}{2 - \gamma(1 + B^{-1})\lambda_i} < 1.$$

Collective scale

$$r_{\mathrm{eff}}(H) = \frac{\mathrm{Tr}(H)}{\lambda_1(H)},$$

$$B_{\mathrm{crit}} \asymp r_{\mathrm{eff}}(H).$$

Small batch: $B < B_{\mathrm{crit}}$

$$\|K\|_1 \asymp \frac{\gamma}{B}\,\mathrm{Tr}(H), \qquad \gamma_{\max} \asymp \frac{B}{\mathrm{Tr}(H)}.$$

Trace-controlled: the maximum stable step size grows linearly with B.

Large batch: $B > B_{\mathrm{crit}}$

$$\gamma\lambda_1(H) \lesssim 1, \qquad \gamma_{\max} \asymp \frac{1}{\lambda_1(H)}.$$

Top-mode-controlled: the batch-size gain has saturated.

Stability is collective, not only a top-eigenvalue condition

Mean-square stability

$$\|K\|_1 = \frac{\gamma}{B} \sum_i \frac{\lambda_i}{2 - \gamma(1 + B^{-1})\lambda_i} < 1.$$

Collective scale

$$r_{\text{eff}}(H) = \frac{\mathrm{Tr}(H)}{\lambda_1(H)},$$

$$B_{\text{crit}} \asymp r_{\text{eff}}(H).$$

Small batch: $B < B_{\text{crit}}$

$$\|K\|_1 \asymp \frac{\gamma}{B}\,\mathrm{Tr}(H), \qquad \gamma_{\max} \asymp \frac{B}{\mathrm{Tr}(H)}.$$

Trace-controlled: the maximum stable step size grows linearly with B.

Large batch: $B > B_{\text{crit}}$

$$\gamma\lambda_1(H) \lesssim 1, \qquad \gamma_{\max} \asymp \frac{1}{\lambda_1(H)}.$$

Top-mode-controlled: the batch-size gain has saturated.

Stability is collective, not only a top-eigenvalue condition

Mean-square stability

$$\|K\|_1 = \frac{\gamma}{B} \sum_i \frac{\lambda_i}{2 - \gamma(1 + B^{-1})\lambda_i} < 1.$$

Collective scale

$$r_{\mathrm{eff}}(H) = \frac{\mathrm{Tr}(H)}{\lambda_1(H)},$$

$$B_{\mathrm{crit}} \asymp r_{\mathrm{eff}}(H).$$

Small batch: $B < B_{\mathrm{crit}}$

$$\|K\|_1 \asymp \frac{\gamma}{B}\,\mathrm{Tr}(H), \qquad \gamma_{\max} \asymp \frac{B}{\mathrm{Tr}(H)}.$$

Trace-controlled: the maximum stable step size grows linearly with B.

Large batch: $B > B_{\mathrm{crit}}$

$$\gamma\lambda_1(H) \lesssim 1, \qquad \gamma_{\max} \asymp \frac{1}{\lambda_1(H)}.$$

Top-mode-controlled: the batch-size gain has saturated.

A power-law memory kernel has one dominant summand

Lemma (Kesten tail reduction)

Suppose $K_r \geq 0$, $\|K\|_1 < 1$, and, for example, $K_r \asymp r^{-s}$ with $s > 1$. Then one large summand dominates a long convolution:

$$\frac{(K * K)_r}{K_r} \longrightarrow 2\|K\|_1.$$

Thus $P = F + K * F + K^{*2} * F + \cdots$ reduces in the PLRF scaling regime to

$$\mathscr{P}_r \asymp \mathscr{F}_r + \frac{1}{\gamma B}\mathscr{K}_r.$$

A power-law memory kernel has one dominant summand

Lemma (Kesten tail reduction)

Suppose $K_r \geq 0$, $\|K\|_1 < 1$, and, for example, $K_r \asymp r^{-s}$ with $s > 1$. Then one large summand dominates a long convolution:

$$\frac{(K * K)_r}{K_r} \longrightarrow 2\|K\|_1.$$

Thus $P = F + K * F + K^{*2} * F + \cdots$ reduces in the PLRF scaling regime to

$$\boxed{\mathscr{P}_r \asymp \mathscr{F}_r + \frac{1}{\gamma B}\mathscr{K}_r.}$$

Four terms become phases

The relevant spectrum is weighted by the residual

Let $y = \Lambda^{1/2} b$. The residual-weighted resolvent is

$$y^\top R(z) y \xrightarrow{\text{RMT}} \underbrace{\sum_{j=1}^{v} \frac{\lambda_j b_j^2}{\lambda_j m(z) - z}}_{=:\, s_{\mathscr{F}}(z)}.$$

Stieltjes inversion.

$$\frac{d\mu_{\mathscr{F}}}{dx}(x) = \frac{1}{\pi} \lim_{\varepsilon \downarrow 0} \operatorname{Im} s_{\mathscr{F}}(x + i\varepsilon).$$

The biased measure drives descent.

$$\mathscr{F}(r) = \int e^{-2\gamma r x} \, \mu_{\mathscr{F}}(dx).$$

The relevant spectrum is weighted by the residual

Let $y = \Lambda^{1/2} b$. The residual-weighted resolvent is

$$y^\top R(z) y \xrightarrow{\text{RMT}} \underbrace{\sum_{j=1}^{v} \frac{\lambda_j b_j^2}{\lambda_j m(z) - z}}_{=:\, s_{\mathscr{F}}(z)}.$$

Stieltjes inversion.

$$\frac{d\mu_{\mathscr{F}}}{dx}(x) = \frac{1}{\pi} \lim_{\varepsilon \downarrow 0} \operatorname{Im} s_{\mathscr{F}}(x + i\varepsilon).$$

The biased measure drives descent.

$$\mathscr{F}(r) = \int e^{-2\gamma r x} \, \mu_{\mathscr{F}}(dx).$$

The relevant spectrum is weighted by the residual

Let $y = \Lambda^{1/2} b$. The residual-weighted resolvent is

$$y^\top R(z) y \xrightarrow{\text{RMT}} \underbrace{\sum_{j=1}^{v} \frac{\lambda_j b_j^2}{\lambda_j m(z) - z}}_{=:\, s_{\mathscr{F}}(z)}.$$

Stieltjes inversion.

$$\frac{d\mu_{\mathscr{F}}}{dx}(x) = \frac{1}{\pi} \lim_{\varepsilon \downarrow 0} \operatorname{Im} s_{\mathscr{F}}(x + i\varepsilon).$$

The biased measure drives descent.

$$\mathscr{F}(r) = \int e^{-2\gamma r x} \, \mu_{\mathscr{F}}(dx).$$

The weighted spectrum splits into peaks and feature bulk

For $\alpha > 1$, $\beta > 1/2$, and mesoscopic $d^{-\alpha} \ll c \ll 1$,

$$\mu_{\mathscr{F}}([c, 2c]) \asymp \underbrace{c^{1+(2\beta-1)/\alpha}}_{\text{population peaks}} + \underbrace{\frac{1}{d} c^{1-1/\alpha}}_{\text{feature bulk}}.$$

Equivalently, in a sufficiently weak topology that averages over the microscopic peaks,

$$d\mu_{\mathscr{F}_{\mathrm{pp}}}(x) \asymp x^{(2\beta-1)/\alpha} dx, \qquad d\mu_{\mathscr{F}_{\mathrm{ac}}}(x) \asymp d^{-1} x^{-1/\alpha} dx.$$

The weighted spectrum splits into peaks and feature bulk

For $\alpha > 1$, $\beta > 1/2$, and mesoscopic $d^{-\alpha} \ll c \ll 1$,

$$\mu_{\mathscr{F}}([c, 2c]) \asymp \underbrace{c^{1+(2\beta-1)/\alpha}}_{\text{population peaks}} + \underbrace{\frac{1}{d} c^{1-1/\alpha}}_{\text{feature bulk}}.$$

Equivalently, in a sufficiently weak topology that averages over the microscopic peaks,

$$d\mu_{\mathscr{F}_{\mathrm{pp}}}(x) \asymp x^{(2\beta-1)/\alpha} dx, \qquad d\mu_{\mathscr{F}_{\mathrm{ac}}}(x) \asymp d^{-1} x^{-1/\alpha} dx.$$

Integrating the two spectral pieces gives two biases

Apply the training filter $e^{-2\gamma r x}$:

$$\mathscr{F}_{\mathrm{pp}}(r) \asymp \int_0^\infty e^{-2\gamma r x} x^{(2\beta-1)/\alpha}\, dx \asymp r^{-(1+(2\beta-1)/\alpha)},$$
$$\mathscr{F}_{\mathrm{ac}}(r,d) \asymp \frac{1}{d}\int_0^\infty e^{-2\gamma r x} x^{-1/\alpha}\, dx \asymp d^{-1} r^{-1+1/\alpha}.$$

Population bias is ordinary modewise relaxation. Embedding bias comes from the target representation being delocalized across the empirical covariance spectrum.

Integrating the two spectral pieces gives two biases

Apply the training filter $e^{-2\gamma r x}$:

$$\mathscr{F}_{\mathrm{pp}}(r) \asymp \int_0^\infty e^{-2\gamma r x} x^{(2\beta-1)/\alpha}\, dx \asymp r^{-(1+(2\beta-1)/\alpha)},$$
$$\mathscr{F}_{\mathrm{ac}}(r,d) \asymp \frac{1}{d}\int_0^\infty e^{-2\gamma r x} x^{-1/\alpha}\, dx \asymp d^{-1} r^{-1+1/\alpha}.$$

Population bias is ordinary modewise relaxation. Embedding bias comes from the target representation being delocalized across the empirical covariance spectrum.

Four mechanisms determine the loss curve

$$\mathscr{P}(r,d) \asymp \underbrace{\mathscr{F}_{\mathrm{pp}}(r)}_{\text{population bias}} + \underbrace{\mathscr{F}_{\mathrm{ac}}(r,d)}_{\text{embedding bias}} + \underbrace{\mathscr{F}_0(d)}_{\text{model capacity}} + \underbrace{\frac{1}{\gamma B}\mathscr{K}_{\mathrm{pp}}(r)}_{\text{stochastic variance}} .$$

term	scaling for $\alpha > 1$	structural meaning
$\mathscr{F}_{\mathrm{pp}}$	$r^{-[1+(2\beta-1)/\alpha]}$	population modes relax
$\mathscr{F}_{\mathrm{ac}}$	$d^{-1}r^{-1+1/\alpha}$	target representation is delocalized across covariance eigenvalues
$\mathscr{F}_0$	$d^{-\alpha+\max(0,1-2\beta)}$	the feature span misses target mass
$\mathscr{K}_{\mathrm{pp}}$	$(\gamma B)^{-1}r^{-2+1/\alpha}$	mini-batches continually inject variance

Four mechanisms determine the loss curve

$$\mathscr{P}(r,d) \asymp \underbrace{\mathscr{F}_{\mathrm{pp}}(r)}_{\text{population bias}} + \underbrace{\mathscr{F}_{\mathrm{ac}}(r,d)}_{\text{embedding bias}} + \underbrace{\mathscr{F}_{0}(d)}_{\text{model capacity}} + \underbrace{\frac{1}{\gamma B}\mathscr{K}_{\mathrm{pp}}(r)}_{\text{stochastic variance}} .$$

term	scaling for $\alpha > 1$	structural meaning
$\mathscr{F}_{\mathrm{pp}}$	$r^{-[1+(2\beta-1)/\alpha]}$	population modes relax
$\mathscr{F}_{\mathrm{ac}}$	$d^{-1}r^{-1+1/\alpha}$	target representation is delocalized across covariance eigenvalues
$\mathscr{F}_{0}$	$d^{-\alpha+\max(0,1-2\beta)}$	the feature span misses target mass
$\mathscr{K}_{\mathrm{pp}}$	$(\gamma B)^{-1}r^{-2+1/\alpha}$	mini-batches continually inject variance

Four mechanisms determine the loss curve

$$\mathscr{P}(r,d) \asymp \underbrace{\mathscr{F}_{\mathrm{pp}}(r)}_{\text{population bias}} + \underbrace{\mathscr{F}_{\mathrm{ac}}(r,d)}_{\text{embedding bias}} + \underbrace{\mathscr{F}_{0}(d)}_{\text{model capacity}} + \underbrace{\frac{1}{\gamma B}\mathscr{K}_{\mathrm{pp}}(r)}_{\text{stochastic variance}} .$$

term	scaling for $\alpha > 1$	structural meaning
$\mathscr{F}_{\mathrm{pp}}$	$r^{-[1+(2\beta-1)/\alpha]}$	population modes relax
$\mathscr{F}_{\mathrm{ac}}$	$d^{-1}r^{-1+1/\alpha}$	target representation is delocalized across covariance eigenvalues
$\mathscr{F}_{0}$	$d^{-\alpha+\max(0,1-2\beta)}$	the feature span misses target mass
$\mathscr{K}_{\mathrm{pp}}$	$(\gamma B)^{-1}r^{-2+1/\alpha}$	mini-batches continually inject variance

Four mechanisms determine the loss curve

$$\mathscr{P}(r,d) \asymp \underbrace{\mathscr{F}_{\mathrm{pp}}(r)}_{\text{population bias}} + \underbrace{\mathscr{F}_{\mathrm{ac}}(r,d)}_{\text{embedding bias}} + \underbrace{\mathscr{F}_{0}(d)}_{\text{model capacity}} + \underbrace{\frac{1}{\gamma B}\mathscr{K}_{\mathrm{pp}}(r)}_{\text{stochastic variance}} .$$

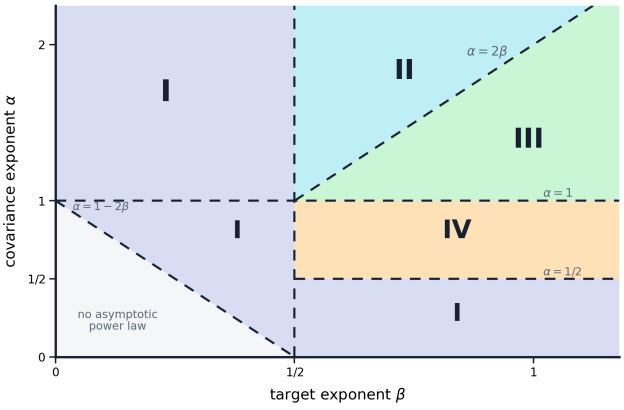
term	scaling for $\alpha > 1$	structural meaning
$\mathscr{F}_{\mathrm{pp}}$	$r^{-[1+(2\beta-1)/\alpha]}$	population modes relax
$\mathscr{F}_{\mathrm{ac}}$	$d^{-1}r^{-1+1/\alpha}$	target representation is delocalized across covariance eigenvalues
$\mathscr{F}_{0}$	$d^{-\alpha+\max(0,1-2\beta)}$	the feature span misses target mass
$\mathscr{K}_{\mathrm{pp}}$	$(\gamma B)^{-1}r^{-2+1/\alpha}$	mini-batches continually inject variance

Four mechanisms determine the loss curve

$$\mathscr{P}(r,d) \asymp \underbrace{\mathscr{F}_{\mathrm{pp}}(r)}_{\text{population bias}} + \underbrace{\mathscr{F}_{\mathrm{ac}}(r,d)}_{\text{embedding bias}} + \underbrace{\mathscr{F}_{0}(d)}_{\text{model capacity}} + \underbrace{\frac{1}{\gamma B}\mathscr{K}_{\mathrm{pp}}(r)}_{\text{stochastic variance}} .$$

term	scaling for $\alpha > 1$	structural meaning
$\mathscr{F}_{\mathrm{pp}}$	$r^{-[1+(2\beta-1)/\alpha]}$	population modes relax
$\mathscr{F}_{\mathrm{ac}}$	$d^{-1}r^{-1+1/\alpha}$	target representation is delocalized across covariance eigenvalues
$\mathscr{F}_{0}$	$d^{-\alpha+\max(0,1-2\beta)}$	the feature span misses target mass
$\mathscr{K}_{\mathrm{pp}}$	$(\gamma B)^{-1}r^{-2+1/\alpha}$	mini-batches continually inject variance

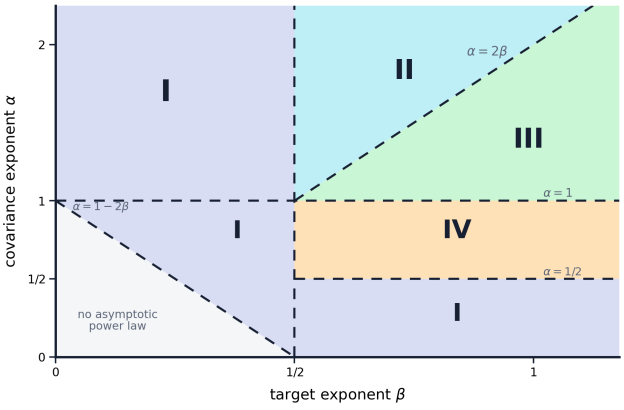
Four active terms organize four phases



Active terms

I	$\mathscr{F}_{\mathrm{pp}} + \mathscr{F}_0$
II	$\mathscr{F}_{\mathrm{pp}} + \mathscr{F}_{\mathrm{ac}} + \mathscr{F}_0$
III	$\mathscr{K}_{\mathrm{pp}} + \mathscr{F}_{\mathrm{ac}} + \mathscr{F}_0$
IV	$\mathscr{F}_{\mathrm{pp}} + \mathscr{K}_{\mathrm{pp}} + \mathscr{F}_0$

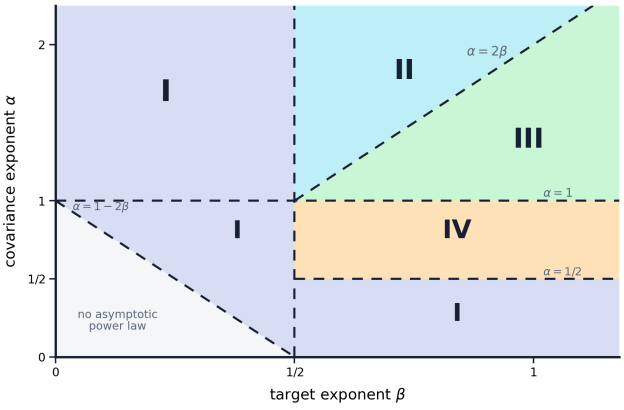
Four active terms organize four phases



Active terms

I	$\mathscr{F}_{\mathrm{pp}} + \mathscr{F}_0$
II	$\mathscr{F}_{\mathrm{pp}} + \mathscr{F}_{\mathrm{ac}} + \mathscr{F}_0$
III	$\mathscr{K}_{\mathrm{pp}} + \mathscr{F}_{\mathrm{ac}} + \mathscr{F}_0$
IV	$\mathscr{F}_{\mathrm{pp}} + \mathscr{K}_{\mathrm{pp}} + \mathscr{F}_0$

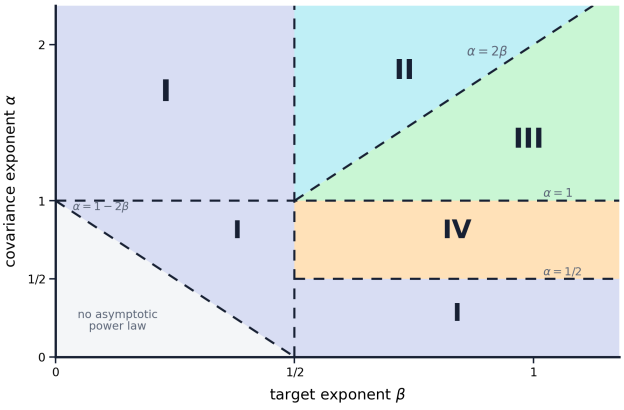
Four active terms organize four phases



Active terms

I	$\mathscr{F}_{\mathrm{pp}} + \mathscr{F}_0$
II	$\mathscr{F}_{\mathrm{pp}} + \mathscr{F}_{\mathrm{ac}} + \mathscr{F}_0$
III	$\mathscr{K}_{\mathrm{pp}} + \mathscr{F}_{\mathrm{ac}} + \mathscr{F}_0$
IV	$\mathscr{F}_{\mathrm{pp}} + \mathscr{K}_{\mathrm{pp}} + \mathscr{F}_0$

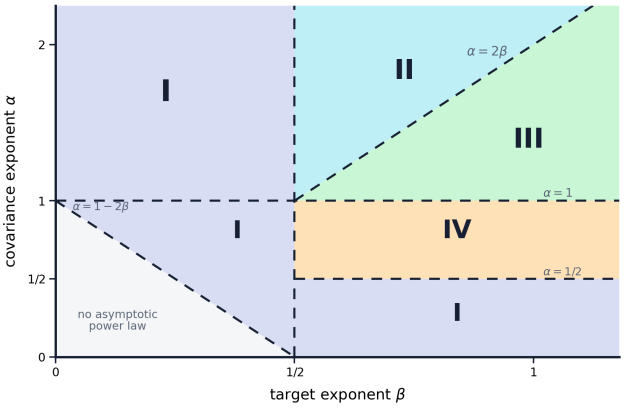
Four active terms organize four phases



Active terms

I	$\mathscr{F}_{\mathrm{pp}} + \mathscr{F}_0$
II	$\mathscr{F}_{\mathrm{pp}} + \mathscr{F}_{\mathrm{ac}} + \mathscr{F}_0$
III	$\mathscr{K}_{\mathrm{pp}} + \mathscr{F}_{\mathrm{ac}} + \mathscr{F}_0$
IV	$\mathscr{F}_{\mathrm{pp}} + \mathscr{K}_{\mathrm{pp}} + \mathscr{F}_0$

Four active terms organize four phases



Active terms

I	$\mathscr{F}_{\mathrm{pp}} + \mathscr{F}_0$
II	$\mathscr{F}_{\mathrm{pp}} + \mathscr{F}_{\mathrm{ac}} + \mathscr{F}_0$
III	$\mathscr{K}_{\mathrm{pp}} + \mathscr{F}_{\mathrm{ac}} + \mathscr{F}_0$
IV	$\mathscr{F}_{\mathrm{pp}} + \mathscr{K}_{\mathrm{pp}} + \mathscr{F}_0$

A two-term balance sets the compute exponent

$$\mathscr{P}(r,d) \asymp r^{-p} + d^{-q}, \qquad C \asymp dr.$$

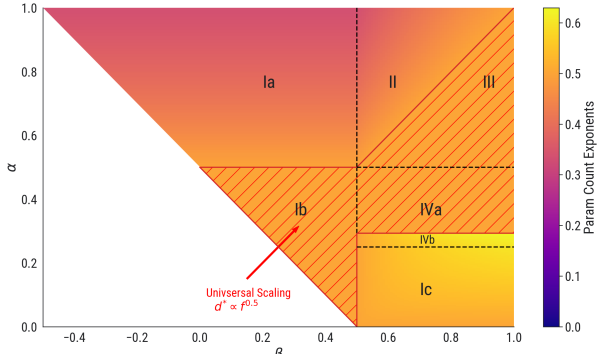
At fixed compute,

$$C^{-p} d^{p} \asymp d^{-q}.$$

Therefore

$$d_\star(C) \asymp C^{p/(p+q)},$$
$$\mathscr{P}_\star(C) \asymp C^{-pq/(p+q)}.$$

The calculation is simple only after identifying the correct active terms.



Red hatching: $d_\star \asymp C^{1/2}$. Source convention: $\alpha_{\text{paper}} = \alpha_{\text{course}}/2$. Paquette et al. (2024), Appendix J, Fig. 8(a).

A two-term balance sets the compute exponent

$$\mathscr{P}(r,d) \asymp r^{-p} + d^{-q}, \qquad \mathrm{C} \asymp dr.$$

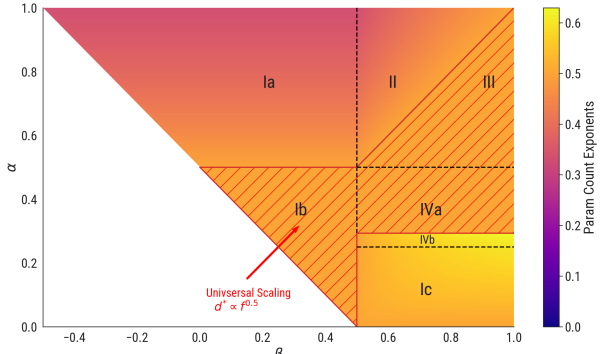
At fixed compute,

$$\mathrm{C}^{-p} d^{p} \asymp d^{-q}.$$

Therefore

$$d_\star(\mathrm{C}) \asymp \mathrm{C}^{p/(p+q)},$$
$$\mathscr{P}_\star(\mathrm{C}) \asymp \mathrm{C}^{-pq/(p+q)}.$$

The calculation is simple only after identifying the correct active terms.



Red hatching: $d_\star \asymp \mathrm{C}^{1/2}$. Source convention: $\alpha_{\mathrm{paper}} = \alpha_{\mathrm{course}}/2$. Paquette et al. (2024), Appendix J, Fig. 8(a).

A two-term balance sets the compute exponent

$$\mathscr{P}(r, d) \asymp r^{-p} + d^{-q}, \qquad \mathrm{C} \asymp dr.$$

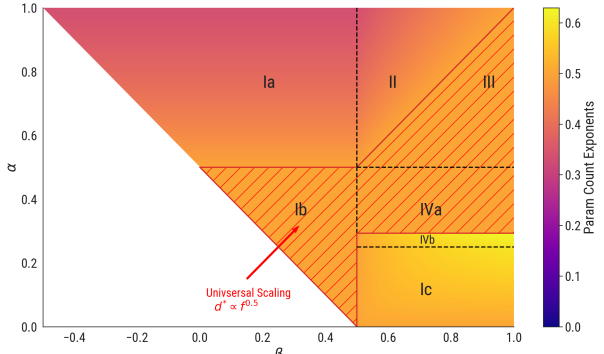
At fixed compute,

$$\mathrm{C}^{-p} d^{p} \asymp d^{-q}.$$

Therefore

$$\boxed{\begin{aligned} d_\star(\mathrm{C}) &\asymp \mathrm{C}^{p/(p+q)}, \\ \mathscr{P}_\star(\mathrm{C}) &\asymp \mathrm{C}^{-pq/(p+q)}. \end{aligned}}$$

The calculation is simple only after identifying the correct active terms.



Red hatching: $d_\star \asymp \mathrm{C}^{1/2}$. Source convention: $\alpha_{\text{paper}} = \alpha_{\text{course}}/2$. Paquette et al. (2024), Appendix J, Fig. 8(a).

A two-term balance sets the compute exponent

$$\mathscr{P}(r, d) \asymp r^{-p} + d^{-q}, \qquad \mathrm{C} \asymp dr.$$

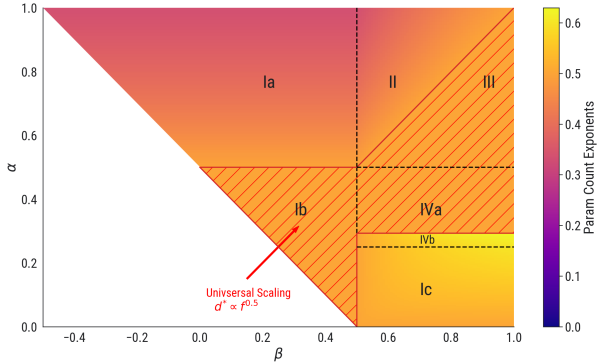
At fixed compute,

$$\mathrm{C}^{-p} d^{p} \asymp d^{-q}.$$

Therefore

$$\boxed{\begin{aligned} d_\star(\mathrm{C}) &\asymp \mathrm{C}^{p/(p+q)}, \\ \mathscr{P}_\star(\mathrm{C}) &\asymp \mathrm{C}^{-pq/(p+q)}. \end{aligned}}$$

The calculation is simple only after identifying the correct active terms.



Red hatching: $d_\star \asymp \mathrm{C}^{1/2}$. Source convention: $\alpha_{\text{paper}} = \alpha_{\text{course}}/2$. Paquette et al. (2024), Appendix J, Fig. 8(a).

A two-term balance sets the compute exponent

$$\mathscr{P}(r, d) \asymp r^{-p} + d^{-q}, \qquad \mathrm{C} \asymp dr.$$

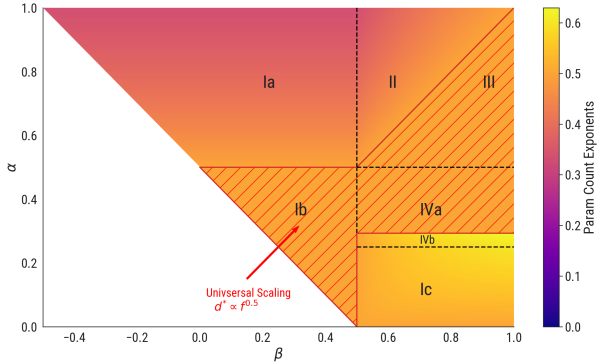
At fixed compute,

$$\mathrm{C}^{-p} d^{p} \asymp d^{-q}.$$

Therefore

$$\boxed{\begin{aligned} d_\star(\mathrm{C}) &\asymp \mathrm{C}^{p/(p+q)}, \\ \mathscr{P}_\star(\mathrm{C}) &\asymp \mathrm{C}^{-pq/(p+q)}. \end{aligned}}$$

The calculation is simple only after identifying the correct active terms.



Red hatching: $d_\star \asymp \mathrm{C}^{1/2}$. Source convention: $\alpha_{\mathrm{paper}} = \alpha_{\mathrm{course}}/2$. Paquette et al. (2024), Appendix J, Fig. 8(a).

Open questions

Beyond least squares: GLMs and nonlinear outputs. Which spectral mechanisms survive a nonlinear observation model? Phase retrieval with power-law data already has a three-stage nonlinear learning curve.
Braun, Loureiro, Minh & Imaizumi (ICLR 2026 oral).

Optimizer dependence. Which terms and phase boundaries change under momentum, signSGD, adaptive methods, or matrix-valued updates?
Next module: momentum and practical reproductions; see also Kim–Song–Yun.

Feature learning with a target hierarchy. Can learned directions themselves have strengths $a_j \asymp j^{-\beta}$, so that β becomes an endogenous feature-learning exponent?
Related models: Ben Arous–Erdogdu–Vural–Wu; Ren–Nichani–Wu–Lee; Sous–Winer.

Open questions

Beyond least squares: GLMs and nonlinear outputs. Which spectral mechanisms survive a nonlinear observation model? Phase retrieval with power-law data already has a three-stage nonlinear learning curve.
Braun, Loureiro, Minh & Imaizumi (ICLR 2026 oral).

Optimizer dependence. Which terms and phase boundaries change under momentum, signSGD, adaptive methods, or matrix-valued updates?
Next module: momentum and practical reproductions; see also Kim–Song–Yun.

Feature learning with a target hierarchy. Can learned directions themselves have strengths $a_j \asymp j^{-\beta}$, so that β becomes an endogenous feature-learning exponent?
Related models: Ben Arous–Erdogdu–Vural–Wu; Ren–Nichani–Wu–Lee; Sous–Winer.

Open questions

Beyond least squares: GLMs and nonlinear outputs. Which spectral mechanisms survive a nonlinear observation model? Phase retrieval with power-law data already has a three-stage nonlinear learning curve.
Braun, Loureiro, Minh & Imaizumi (ICLR 2026 oral).

Optimizer dependence. Which terms and phase boundaries change under momentum, signSGD, adaptive methods, or matrix-valued updates?
Next module: momentum and practical reproductions; see also Kim–Song–Yun.

Feature learning with a target hierarchy. Can learned directions themselves have strengths $a_j \asymp j^{-\beta}$, so that β becomes an endogenous feature-learning exponent?
Related models: Ben Arous–Erdogdu–Vural–Wu; Ren–Nichani–Wu–Lee; Sous–Winer.

Selected references

- **Paquette et al.** *4+3 Phases of Compute-Optimal Neural Scaling Laws*. NeurIPS 2024.
- **Hoffmann et al.** *Training Compute-Optimal Large Language Models*. NeurIPS 2022.
- **Kim, Song & Yun.** *Scaling Laws of signSGD in Linear Regression*. ICLR 2026.
- **Lin et al.** *Scaling Laws in Linear Regression: Compute, Parameters, and Data*. NeurIPS 2024.
- **Maloney, Roberts & Sully.** *A Solvable Model of Neural Scaling Laws*. 2022.
- **Braun et al.** *Fast Escape, Slow Convergence: Learning Dynamics of Phase Retrieval under Power-Law Data*. ICLR 2026 oral.
- **Ben Arous et al.** *Learning Quadratic Neural Networks in High Dimensions*. NeurIPS 2025.
- **Ren et al.** *Emergence and Scaling Laws in SGD Learning of Shallow Neural Networks*. NeurIPS 2025.
- **Sous & Winer.** *Asymmetric Scaling Laws from Sparse Features*. 2026.