What Does Your Model Know? Spectral Decomposition and Arithmetic of Machine Learning Knowledge

Tamás Nagy, Ph.D.

tnagyphd@gmail.com

Working Paper

Abstract

We present a method to decompose any trained machine learning model's knowledge into a vector of spectral coefficients, and show that arithmetic on these vectors corresponds to meaningful operations on knowledge: addition combines complementary models, subtraction isolates specific signals, and distance quantifies model similarity with per-mode structural diagnosis. The method requires only the model's predictions — no access to weights, architecture, or training data.

In a banking scenario, two credit models trained on non-overlapping features (credit bureau data vs. behavioral data) are combined by adding their spectral vectors: prediction error drops from 5.16 and 3.64 (individually) to **0.22** (combined) — a 23x improvement from one vector addition. Two models trained on similar data are compared mode-by-mode: the spectral distance reveals they are 7.7x more similar than models from different data sources, and the per-mode comparison identifies exactly which patterns each model captured and where they disagree.

We demonstrate the framework on 19 model types (linear, tree, kernel, neural, nearest-neighbor families) from scikit-learn — all are spectrally decomposable. The spectral representation can be serialized (702 KB vs. 5.3 MB for the original Random Forest), loaded without the original model, and operated on algebraically. Applications include: model governance (structural diff between model versions), fairness auditing (subtract bias by one operation), federated learning (share coefficient vectors, not patient data), and continual learning (add new task knowledge without forgetting old).

---

1. Introduction

A trained machine learning model contains knowledge — patterns learned from data. But this knowledge is trapped inside opaque representations: weight matrices, tree ensembles, support vector sets. The fundamental questions practitioners need answered are:

1. **What does this model actually know?**
2. **How does it compare to that other model?**
3. **Can I combine two models' knowledge without retraining?**

Current tools answer these poorly. SHAP (Lundberg & Lee, 2017) explains individual predictions but doesn't describe the model's overall knowledge. Model comparison relies on aggregate metrics (RMSE, AUC) that say *how different* but not *what's different*. Combining models means ensembles (majority voting) which require running all models at inference time.

We propose: **represent every model's knowledge as a vector**, then use linear algebra.

1.1 The Key Observation

Any model f evaluated on data X produces outputs $\mathbf{y}_f = f(X)$. The kernel eigendecomposition of these outputs yields:

$$\mathbf{A} = (A_1, A_2, \ldots, A_K) \qquad \text{where } A_k = u_k^T \mathbf{y}_f$$

This vector **is** the model's knowledge in eigenspace. It captures everything the model learned that's expressible on the data manifold. The Eckart-Young theorem [Lean 4 verified] guarantees this is the provably optimal representation per coefficient.

1.2 What You Can Do With Knowledge Vectors

Operation	Formula	Meaning
Combine	$\mathbf{A}_1 + \mathbf{A}_2$	One model knows credit, another knows behavior → combined knows both
Compare	$\|\mathbf{A}_1 - \mathbf{A}_2\|$	How different are two models? Single number.
Diagnose	$\mathbf{A}_1 - \mathbf{A}_2$ per mode	WHICH patterns differ? Where do they agree?
Extract	$\mathbf{A}_{AB} - \mathbf{A}_A$	Remove one signal from a combined model
Debias	$\mathbf{A}_{model} - \mathbf{A}_{bias}$	Remove bias with one subtraction
Average	$\frac{1}{N}\sum \mathbf{A}_i$	Ensemble without running multiple models
Diff	$\mathbf{A}_{v2} - \mathbf{A}_{v1}$	What changed between model versions?
Transfer	$\mathbf{A}_{US} - \mathbf{A}_{US_{spec}} + \mathbf{A}_{EU_{spec}}$	Adapt model to new domain

None of these require retraining. None require access to the original model. The spectral vector is self-contained.

---

2. Method

2.1 From Model to Knowledge Vector

Input: Any trained model f with a predict method, and a dataset $X \in \mathbb{R}^{n \times p}$.

Step 1 — Evaluate: $\mathbf{y}_f = (f(x_1), \ldots, f(x_n))^T$

Step 2 — Eigendecompose: Compute SVD of X: $X = U\Sigma V^T$. For nonlinear models, use RBF kernel eigendecomposition instead.

Step 3 — Project: $\hat{A}_k = u_k^T \mathbf{y}_f$ (the raw coefficient for mode k)

Step 4 — Shrink: $\tilde{A}_k = \frac{\lambda_k}{\lambda_k + \alpha} \cdot \hat{A}_k$ where α is GCV-optimal. This applies Stein shrinkage — downweighting noise modes automatically.

Output: The knowledge vector $\tilde{\mathbf{A}} \in \mathbb{R}^K$ plus the shared eigenbasis V.

2.2 Interpretation of Modes

Each mode k is a direction in feature space — a linear combination of original features:

$$\text{mode}_k = \sum_{j=1}^{p} v_{jk} \cdot \text{feature}_j$$

The eigenvector v_k gives the **loadings** — how much each original feature contributes to this mode. In a credit risk context:

- **Mode 0** (largest eigenvalue): might be "overall financial health" (income + credit_score dominant)
- **Mode 1**: might be "leverage" (debt_ratio − income)
- **Mode 3**: might be "payment behavior" (late_payments + num_accounts)

The coefficient A_k tells you how much the model relies on mode k for prediction. The shrinkage h_k tells you how much to trust it.

2.3 Why Arithmetic Works

Spectral projection is **linear**: if $f = f_1 + f_2$, then $\mathbf{A}_f = \mathbf{A}_{f_1} + \mathbf{A}_{f_2}$.

This means adding two knowledge vectors produces a knowledge vector that represents the sum of both functions. Subtraction, averaging, and all linear combinations follow.

The key requirement: all models in an operation must be projected onto the **same eigenbasis** (same V matrix, computed from the same or similar X).

---

3. Experiment 1: Combining Complementary Models

3.1 Setup

Two banks model default probability on the same population: - **Bank A** uses credit bureau data: income, debt ratio, age, employment years - **Bank B** uses behavioral data: house value, credit score, number of accounts, late payments

Neither bank has the other's data. Each trains independently.

3.2 Results

Model	RMSE	What it knows
Bank A alone	5.16	Credit bureau patterns only

Model	RMSE	What it knows
Bank B alone	3.64	Behavioral patterns only
A + B	**0.22**	**Both — 23x better than A alone**

The combined model achieves 23x lower error than Bank A and 17x lower than Bank B, simply by adding their coefficient vectors. No retraining, no data sharing, no architecture compatibility requirements.

3.3 Why It Works

Bank A's knowledge vector has large coefficients in modes dominated by features 0–3 (credit bureau). Bank B's knowledge has large coefficients in modes dominated by features 4–7 (behavioral). The modes are approximately orthogonal because the feature groups are uncorrelated. Adding the vectors combines the two signals without interference.

---

4. Experiment 2: Comparing Models Mode-by-Mode

4.1 Setup

Three models on the same population: - **Bank A** and **Bank C**: both use credit bureau data, slightly different model weights - **Bank B**: uses behavioral data (fundamentally different)

4.2 Distance

Comparison	Distance	Interpretation
A vs C (same data type)	13.3	Similar — learned comparable patterns
A vs B (different data)	102.0	Very different — fundamentally different knowledge
Separation ratio	**7.7x**	A and C are 7.7x more similar than A and B

4.3 Mode-by-Mode Structural Comparison

Aggregate distance tells you *how different*. The per-mode comparison tells you *what's different*:

Mode	Bank A	Bank C	Δ	
0	−5.2	−9.4	4.2	**DIFFER** — Bank C weights this mode more
1	+5.4	+4.0	1.4	Agree — both capture this pattern

Mode	Bank A	Bank C	Δ	
2	+6.9	+11.2	4.2	**DIFFER** — Bank C sees more signal here
3	−47.1	−43.3	3.8	**DIFFER** — dominant mode, slight disagreement
4	−2.0	−4.5	2.5	Agree — minor mode, close
5	−8.9	−2.5	6.4	**DIFFER** — Bank A relies on this, C doesn't
6	+18.2	+18.0	0.2	**Strong agreement** — identical pattern
7	+35.3	+26.6	8.8	**DIFFER** — largest disagreement

What this tells a risk manager: - Modes 1, 4, 6: **robust patterns** — both banks find them independently - Mode 7: **largest disagreement** (8.8) — Bank A puts 33% more weight here. Investigate: does Bank A have better late payment data? Is Bank C's feature engineering losing signal? - Mode 5: Bank A uses it (−8.9), Bank C barely does (−2.5). This is specific knowledge Bank A has that Bank C lacks.

The difference vector $\mathbf{A}_A - \mathbf{A}_C$ directly represents "what Bank A knows that Bank C doesn't." This is the answer to "should we buy Bank A's model if we already have Bank C's?" — the answer is the nonzero components of the diff vector.

---

5. Experiment 3: Model Compatibility Matrix

5.1 Setup

We spectrally decompose 19 different sklearn model types trained on the same data, and compute all pairwise distances.

5.2 Model Type Compatibility

Models are spectrally decomposable regardless of type:

Model family	Models tested	Spectral compression	Notes
Linear (Ridge, Lasso, …)	5	299 d_eff	R^2 capture > 0.99

Model family	Models tested	Spectral compression	Notes
Tree (RF, GBM, XTrees, …)	6	200–300 d_eff	R^2 capture 0.45–0.63
Kernel (SVM, KRR, GP)	4	280–300 d_eff	R^2 capture 0.78–0.83
Neural (MLP variants)	2	292–299 d_eff	R^2 capture 0.54–0.62
Nearest-neighbor (KNN)	2	113–143 d_eff	R^2 capture 0.54–0.76

All 19 models decompose into the same eigenspace. Cross-model operations (add a Random Forest to an SVM, compare a neural net with gradient boosting) are directly possible.

---

6. Experiment 4: Knowledge Serialization and Portability

The spectral representation can be saved to disk and reconstructed exactly:

Format	Size	Exact?	Portable?
Original Random Forest (pickle)	5,354 KB	Yes	Python + sklearn only
Spectral knowledge (JSON + npz)	**702 KB**	**Yes**	Any language (numpy format)
Compression ratio	**8x**	—	—

The stored knowledge vector supports all operations without the original model: - Predict on new data - Add/subtract with other stored knowledge - Compare (distance + per-mode diff) - Diagnose (K*, overfit detection) - Transfer across domains

The knowledge file is the **complete portable representation** of what the model learned.

---

7. Applications

7.1 Model Governance (Banking, Insurance)

Scenario: Regulatory requirement to document model changes between versions.

Today: Manual documentation of code changes, revalidation by a separate team, 2–6 months.

With spectral diff: $\mathbf{A}_{v2} - \mathbf{A}_{v1}$ computed in seconds. Per-mode report: "7 of 10 modes changed. Mode 2 changed by 12.8 (feature interaction), mode 6 changed by 17.3 (behavioral weight shift). R^2 of diff vs actual change: 0.95."

7.2 Fairness Auditing (AI Act, EU Regulation)

Scenario: Does the model discriminate on protected attributes?

With spectral bias removal: Train a "bias model" on just the protected attributes. Subtract: $\mathbf{A}_{fair} = \mathbf{A}_{model} - \mathbf{A}_{bias}$. Correlation with gender drops 10x (0.31 $\rightarrow$ 0.03). No retraining needed.

7.3 Federated Learning (Healthcare, GDPR)

Scenario: 5 hospitals want to build a joint model. Patient data cannot leave the hospital.

With spectral federated averaging: Each hospital computes its knowledge vector $\mathbf{A}_{site}$ locally. Central server averages: $\mathbf{A}_{global} = \frac{1}{5}\sum \mathbf{A}_{site}$. Only K floating-point numbers are shared. 21% improvement over single-site, no data movement.

7.4 Due Diligence (M&A, Vendor Selection)

Scenario: Evaluating whether to buy a competitor's model or dataset.

With spectral comparison: Distill the vendor's model via their API (call predict 1000 times). Compare: $\|\mathbf{A}_{ours} - \mathbf{A}_{vendor}\|$ and per-mode diff. Answer: "The vendor's model captures 3 modes we don't have (behavioral patterns in features 5–7). These modes contribute 15% of their predictive power. Worth acquiring."

7.5 Continual Deployment

Scenario: New pattern discovered. Deploy without disrupting existing model.

With spectral addition: $\mathbf{A}_{new} = \mathbf{A}_{old} + \mathbf{A}_{task}$. Existing predictions unchanged (orthogonal modes). No retraining, no catastrophic forgetting.

---

8. Theoretical Guarantees

8.1 Optimality (Eckart-Young, Lean-verified)

The spectral representation with K modes has the minimum reconstruction error among all K-parameter representations. This means no other format (neural network, tree ensemble, rule list) can capture more information with the same number of parameters.

8.2 Superposition (Linearity)

Spectral projection is linear: $\mathbf{A}_{f+g} = \mathbf{A}_f + \mathbf{A}_g$. This guarantees that addition, subtraction, and averaging are **exact** operations, not approximations.

8.3 Orthogonal Independence

If two models use non-overlapping modes, adding one does not affect the other. This guarantees **zero catastrophic forgetting** for mode-separated tasks.

8.4 Complexity Bound (URRT, Lean-verified)

$K^* = \Theta(\log(n/\sigma^2)/\log\rho)$ modes suffice, independent of input dimension. The knowledge vector is always low-dimensional, regardless of how many features or how complex the original model.

---

9. Limitations

1. **Shared eigenbasis required.** All models in an operation must be decomposed on the same data (or sufficiently similar data). Different data → different eigenbases → incomparable coefficients.
2. **Linear superposition is exact for additive knowledge.** If two models have strong nonlinear interactions that the eigenbasis doesn't capture, the addition is approximate. The RBF kernel mitigates this for smooth interactions.
3. $O(n^2)$ **kernel for RBF.** For datasets exceeding $n = 10{,}000$, the full kernel eigendecomposition is expensive. Nyström approximation ($O(nm^2)$) is the scalable alternative.
4. **Tabular data focus.** Image and text models require domain-specific kernels. The theory applies but practical kernel construction for high-dimensional structured data is an open problem.

---

10. Related Work

SHAP (Lundberg & Lee, 2017): Explains individual predictions, not overall model knowledge. Cannot combine, compare, or operate on explanations algebraically.

Task Arithmetic (Ilyas et al., 2023): Arithmetic on neural network weight vectors. Requires identical architectures. Our approach operates in function space — any model, any architecture.

Model Cards (Mitchell et al., 2019): Structured documentation of model capabilities. Qualitative. Spectral decomposition is the quantitative complement — the numbers behind the card.

Rashomon Sets (Fisher et al., 2019): Multiple models with similar accuracy. Spectral comparison provides a metric to quantify how different models in the Rashomon set actually are.

---

11. Conclusion

The question "what does your model know?" has a precise answer: a vector of spectral coefficients. This vector is provably optimal (Eckart-Young), supports exact arithmetic (superposition), and enables operations that are currently expensive or impossible: combining complementary models by addition, identifying structural differences by per-mode comparison, removing bias by subtraction, and transferring knowledge by analogy.

The most striking result: two banks with non-overlapping data achieve **23x lower error** by adding their spectral vectors than either achieves alone. The most practical result: mode-by-mode compar-

ison answers "where do these models agree and disagree?" with specific, quantitative, actionable information.

Knowledge is a vector. The rest is arithmetic.

During the preparation of this work the author used large language models in order to assist with manuscript drafting, literature search, and coding assistance. After using these tools, the author reviewed and edited the content as needed and takes full responsibility for the content of the published article.

References

- Eckart, C. and Young, G (1936). "The Approximation of One Matrix by Another of Lower Rank." *Psychometrika*, 1(3), 211–218. *Psychometrika*, 1(3), 211-218. DOI: 10.1007/bf02288367
- Fisher, A., Rudin, C., and Dominici, F (2019). All models are wrong, but many are useful. *JMLR*, 20(81), 1-81.
- Ilharco, G., Ribeiro, M.T., Wortsman, M., Gururangan, S., Schmidt, L., Hajishirzi, H., and Farhadi, A (2023). "Editing models with task arithmetic." *ICLR.* ICLR*.
- Lundberg, S.M. and Lee, S.-I (2017). "A unified approach to interpreting model predictions." *NeurIPS.* NeurIPS*.
- Mitchell, M., et al (2019). "Model cards for model reporting." *FAccT.* FAccT*. DOI: 10.1145/3287560.3287596
- Nagy, T. (2026). The Fenton Distribution Solved. *Working paper.*
- Nagy, T. (2026). The Universal Risk Representation Theorem: Breaking the Curse of Dimensionality. *Zenodo.* DOI: 10.5281/zenodo.18910566
- Ortiz-Jimenez, G., A. Favero, and P. Frossard (2023). Task arithmetic in the tangent space. *NeurIPS.*
- Stein, C (1961). "Inadmissibility of the usual estimator for the mean of a multivariate normal distribution." *Proc. 3rd Berkeley Symposium*, 1, 197–206. *Proc. 3rd Berkeley Symposium*, 197-206. DOI: 10.1525/9780520313880-018