ROW GEOMETRY FOR TABULAR IN-CONTEXT LEARNING: WHAT DO THE MODELS USE?

Anonymous authors
Paper under double-blind review

ABSTRACT

Tabular foundation models perform in-context prediction by relating queries to a labeled support set, across tables whose columns differ in number, units, and meaning, so that relations between rows provide a shared structure. We study whether support-to-query prediction can be carried by these relations alone, and which part the models use. Our approach constructs the row Gram matrix of each task from the raw input features standardized with support statistics, and uses it as a row operator that propagates support labels to queries. This formulation connects tabular in-context learning with label propagation, admits several choices of geometry and propagation, and allows changing the row operator while the trained model stays fixed. Across six architectures, the Gram matrix can replace direct access to the raw input features, and changing only the row operator shows that these models rely on first-order propagation along correctly attached support–query relations. We further characterize the Gram matrix as the maximal orthogonal invariant of the normalized rows, show that it remains predictive for classical predictors on 25 OpenML tables, and find closely related internal row structure in frozen TabPFN and TabICL; adding it to TabPFN is neutral on natural tables and helps where the frozen model is weak. These results identify row geometry as a useful object for understanding and designing in-context learning over tabular data.

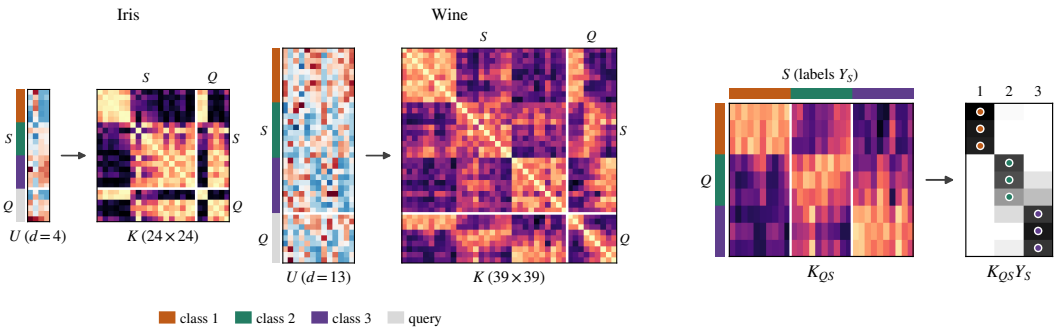
1 INTRODUCTION

Tabular foundation models (TFMs) face a simple difficulty: the same pretrained model is applied to tables whose columns may differ in number, units, order, and meaning (Hollmann et al., 2023; 2025; Qu et al., 2025; Ma et al., 2025; Mueller et al., 2025). Given a new table, the model receives labeled support rows and predicts the labels of the query rows in a single forward pass, which is known as in-context learning (ICL) (Brown et al., 2020; Hollmann et al., 2023; 2025). Following the TFM literature, we refer to one such prediction problem as a *task* (Hollmann et al., 2025). For each query, the model therefore decides which support rows matter, in a feature space it has not seen before.

This raises a basic challenge: the same support-to-query computation is applied to every table, although the raw input features change from one table to another. A representation built on the raw input features is tied to its table, because the meaning of each feature is table-specific. In contrast, every task defines how its rows relate, and this information takes the same form, a matrix of pairwise relations between rows, regardless of how many columns, i.e., features, there are or how they are ordered.

Current TFMs meet this challenge by learning to map every table into a fixed internal representation. TabPFN pads each row to a fixed number of features (Hollmann et al., 2023), TabPFN v2 embeds every cell and alternates attention across features and across rows, with random feature embeddings (Hollmann et al., 2025), and TabICL compresses every row into a fixed-size embedding with attention over each column and then over the features of each row (Qu et al., 2025). In all of these models, the relations between rows emerge inside the attention layers, together with the processing of the raw input features, as we discuss in Appendix H. Recent interpretability studies have taken important steps toward understanding this computation, showing that TFM predictions behave like similarity-weighted combinations of support labels (Han et al., 2025; Miftachov et al., 2026; Koshil et al., 2025; Biloš et al., 2026). Few-shot learning, which addresses the different problem of recognizing new images from a handful of labeled ones, likewise predicts queries from their similarity to labeled

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Gram matrices of two tasks with different N and d.

(b) Label propagation with K_{QS}.

Figure 1: **The row Gram matrix K relates the rows of a task independently of the number of raw input features (a), and its support–query block carries the support labels to the queries (b).** (a) Tasks from the Iris table (Fisher, 1936) ($N = 24$, $d = 4$) and the Wine table (Aeberhard & Forina, 1992) ($N = 39$, $d = 13$). The canonicalized rows U of Equation (2) map to K, whose size $N \times N$ is independent of d. Rows and columns list the support rows S before the query rows Q, separated by white lines, and strips mark the class of each support row; query labels are hidden. (b) On Wine, multiplying K_{QS} by the one-hot support labels Y_S (top strip) gives class scores (columns) whose maximum (dots) recovers the true label (left strip) of all nine queries.

examples (Vinyals et al., 2016; Liu et al., 2019). These works offer valuable insight into how support labels influence the queries. Nonetheless, in all aforementioned methods the relations between rows are formed inside the predictor, so it is not yet known whether a trained model still requires the raw input features once these relations are available, or which of them it uses. We therefore ask:

Can support-to-query prediction in tabular ICL be carried by the relations between rows alone, and if so, which property of those relations is actually used?

Our approach. We identify the separation of row relations from the predictor as a desirable property for answering this question. Intuitively, forming the relations between rows before the predictor allows us to remove the raw input features while keeping these relations. It also allows us to change the relations while the trained model stays fixed, and to provide them to a different predictor. Guided by this property, we build our analysis on the row Gram matrix of the task,

$$K = \bar{U}\bar{U}^\top, \tag{1}$$

where $\bar{U} \in \mathbb{R}^{N \times d}$ collects the N rows of the task, counting both support and query rows, over its d raw input features, after each feature is standardized with support statistics and each row is normalized, as detailed in Section 3.1. The Gram matrix $K \in \mathbb{R}^{N \times N}$ uses no labels and no learned parameters, and its size is set by the number of rows, as illustrated in Figure 1. Therefore, unlike the learned encoders of current TFMs, K describes every table in the same form, for any set of columns. We train in-context models that receive K, as described in Section 3.3, and we then change K while the trained models stay fixed. These interventions, reported in Sections 4.2 and 4.3, show that in-context models of six architectures rely on first-order propagation of support labels along correctly attached support–query relations, as illustrated in Figure 1b, and our formal results, shown in Propositions 1 and 2, explain which relations K keeps and why their correct attachment matters.

Main contributions. The contributions of this work are summarized as follows:

- We show, by changing the row operator of trained models, that models of six architectures rely on first-order propagation along correctly attached support–query relations, and that K can replace direct access to the raw input features, as reported in Sections 4.2 and 4.3.
- We introduce a framework built on K that separates the row relations from the predictor, with formal results behind these interventions, as presented in Section 3.
- We show that K remains predictive with classical predictors on 25 OpenML tables, and that frozen TabPFN and TabICL contain internal row structure closely related to K, as descriptive and conditional evidence reported in Sections 4.4 and 4.5.

2 Related Work

Tabular foundation models and cross-table learning. TabPFN established in-context tabular prediction through synthetic prior fitting (Hollmann et al., 2023; 2025), and TabICL (Qu et al., 2025), TabDPT (Ma et al., 2025), MotherNet (Mueller et al., 2025), and HyperFast (Bonet et al., 2024) extend it to larger tables, real-data pretraining, and generated predictors. To handle a varying number of columns, TabPFN, MotherNet, and TabDPT pad or project each row to a fixed width, while TabPFN v2 and TabICL embed each cell with shared encoders before attention. Cross-table models instead use column names or table-specific featurizers (Wang & Sun, 2022; Zhu et al., 2023), and retrieval (Gorishniy et al., 2024) and set-based models (Zaheer et al., 2017; Lee et al., 2019; Snell et al., 2017) relate samples through a particular encoder or retrieval rule. In all of these models, the row relations are formed inside the predictor, while in this work, we remove the column axis before the predictor, so that the row relations can be removed, changed, or provided to another predictor.

Kernel, nearest-neighbor, and mechanistic views. Kernel perspectives connect ICL to weighted combinations of context labels (Han et al., 2025), SoftKNN-ICL trains in-context models as soft nearest-neighbor classifiers (Koshil et al., 2025), and KernelICL replaces the readout of a TFM with explicit kernels (Miftachov et al., 2026). Mechanistic studies identify similarity-based readouts in pretrained TFMs, trace their predictions across layers, and relate them to exemplar-based prediction (Biloš et al., 2026; Balef et al., 2026a;b; Shaheen et al., 2026). These works characterize the prediction rule, whereas we study the object it operates on, using linear centered kernel alignment (CKA) (Kornblith et al., 2019) to compare it with TFMs.

Support–query relations and label propagation. Graph-based semi-supervised learning propagates labels over affinity graphs (Zhu et al., 2003; Zhou et al., 2003), and few-shot methods predict queries from their relations to labeled examples, through matching (Vinyals et al., 2016), learned relation scores (Sung et al., 2018), graph message passing (Garcia Satorras & Bruna Estrach, 2018; Kim et al., 2019), and label propagation (Liu et al., 2019; Ziko et al., 2020; Ma et al., 2020; Zhu & Koniusz, 2023). Retrieving the nearest support rows also improves TFMs (Thomas et al., 2024; Xu et al., 2025). These works make support–query relations part of the predictor, while in this work, we form them as a label-free, row-indexed state before the predictor and intervene on it while the trained model stays fixed, which applies equally to trained few-shot predictors. Appendix H discusses them further.

3 Row Geometry for Support-to-Query Prediction

Tables differ in the number and meaning of their raw input features, whereas every prediction concerns the support and query rows. We therefore describe each task through the relations between its rows, in three steps. Section 3.1 builds the Gram matrix K of the task, which retains every row relation for any number, offset, order, and, up to a stabilizer, scale of columns. As K records these relations but not the labels, Section 3.2 describes how a row operator carries support labels to queries, and through which entries. Section 3.3 builds a controlled in-context model in which K and the operator change while the trained model stays fixed, so that the experiments test which entries are used.

3.1 From Raw Input Features to Indexed Row Geometry

A task consists of raw input features $X \in \mathbb{R}^{N\times d}$ for N support and query rows and d raw input features, with index sets S and Q that partition the rows. Because d and the meaning and scale of the columns change across tables, we first place each task in a frame defined by its support rows.

Support-conditioned canonicalization. Let $\mu_S, v_S \in \mathbb{R}^d$ collect the support means and population variances of the d raw input features, so that $\mu_{S,j}$ and $v_{S,j}$ refer to feature j. We compute

$$U = \mathrm{clip}_{[-c,c]}\Big(\big(X - \mathbf{1}\mu_S^\top\big)\,\mathrm{diag}(s_S)^{-1}\Big), \qquad s_{S,j} = \sqrt{v_{S,j} + \varepsilon_s}, \tag{2}$$

where $\mathbf{1} \in \mathbb{R}^N$ is the all-ones vector, $s_S \in \mathbb{R}^d$ collects the stabilized standard deviations, $\varepsilon_s = 10^{-6}$ is a stabilizer, and $\mathrm{clip}_{[-c,c]}$ clips each entry to $[-c, c]$, where c is a hyperparameter that limits extreme values, discussed in Appendix B.3. Notably, all statistics come from the support rows, so that query rows and labels do not influence the reference frame. Because Equation (2) removes per-feature shifts and positive rescalings, up to the stabilizer, we call it *support-conditioned canonicalization*.

Indexed row Gram matrix. We then normalize the rows and form their Gram matrix,

$$\bar{u}_i = \frac{u_i}{\|u_i\|_2 + \varepsilon_r}, \qquad K = \bar{U}\bar{U}^\top \in \mathbb{R}^{N\times N}, \tag{3}$$

where u_i and $\bar{u}_i$ denote the i-th rows of U and $\bar{U} \in \mathbb{R}^{N\times d}$, and $\varepsilon_r = 10^{-8}$, so that K_{ij} is the stabilized cosine similarity between rows i and j. We refer to K as the support-canonicalized cosine row geometry of the task, or row geometry for short. As illustrated in Figure 1a, the size of K is determined by N alone, and K is *indexed* by the rows: the entry K_{ij} refers to rows i and j.

Invariance after canonicalization. After canonicalization, the columns of different tables still differ in meaning, so that the information shared across tables lies in the pairwise relations between rows rather than in the orientation of the features. A representation that serves every table is therefore invariant to a common orthogonal transformation of the features of $\bar{U}$ and retains every pairwise relation between its rows. Proposition 1 shows that the Gram matrix K is exactly such a representation, whereas summaries such as the eigenvalues of K or the nearest neighbors of each row retain part of these relations; here, Y and m are the label signal and the support mask of Section 3.2.

Proposition 1 (Maximal orthogonal invariant)**.** *For $A, B \in \mathbb{R}^{N\times d}$ and the orthogonal group $\mathrm{O}(d)$,*

$$AA^\top = BB^\top \quad \Longleftrightarrow \quad B = AO \text{ for some } O \in \mathrm{O}(d).$$

Consequently, every rule $F(A, Y, m)$ with $F(AO, Y, m) = F(A, Y, m)$ for all $O \in \mathrm{O}(d)$ has the form $f(AA^\top, Y, m)$ on realizable Gram matrices.

The reverse direction follows from $OO^\top = I$, and the forward direction follows because A and B have the same left singular vectors and singular values, given by the eigendecomposition of $AA^\top = BB^\top$, so that their compact singular value decompositions differ only in the right singular vectors, as we prove in Appendix A. In other words, K retains every relation between the rows of $\bar{U}$ and removes only the choice of basis. Proposition 1 is a classical result of invariant theory (Weyl, 1946). With $\varepsilon_s > 0$, the invariance of K to per-feature rescaling holds approximately, while its invariance to per-feature shifts, feature permutations, and sign flips remains exact, as we show in Appendices A and G. This invariance concerns the canonicalized rows: because Equation (2) acts on each column separately, an orthogonal transformation of the raw input features can change K, which the rotation suite in Section 4.1 uses to evaluate the models beyond this invariance.

3.2 Support-to-Query Label Propagation

The Gram matrix K of Section 3.1 records how the rows relate to one another, but not which rows carry which labels. We therefore encode the observed labels as a row-indexed signal $Y \in \mathbb{R}^{N\times C}$ over C classes, with one-hot labels on the support rows and zeros elsewhere, and write $m \in \{0, 1\}^N$ for the support mask. For a row operator $G \in \mathbb{R}^{N\times N}$, with the support rows ordered first,

$$G = \begin{bmatrix} G_{SS} & G_{SQ} \\ G_{QS} & G_{QQ} \end{bmatrix}, \qquad Y = \begin{bmatrix} Y_S \\ 0 \end{bmatrix}, \tag{4}$$

where $G_{\mathcal{AB}}$ collects the rows in $\mathcal{A}$ and the columns in $\mathcal{B}$ for $\mathcal{A}, \mathcal{B} \in \{S, Q\}$, and $Y_S \in \mathbb{R}^{|S|\times C}$ contains the support labels. The most direct choice is $G = K$, one step of label propagation, which weights the support labels by their cosine similarity to each query, as illustrated in Figure 1b. More generally, a row operator carries the labels to the queries through specific entries, each referring to one pair of rows. As Proposition 2 shows, support labels reach queries only through the support–query block G_{QS} of Equation (4). Reordering the operator and labels together keeps propagation intact, whereas reordering the operator alone connects the wrong rows.

Proposition 2 (Row correspondence)**.** *The propagated query signal is*

$$(GY)_Q = G_{QS}Y_S.$$

Thus a diagonal G carries no support label to the queries. Moreover, for every permutation matrix P,

$$(PGP^\top)(PY) = P(GY).$$

Propagation is equivariant under joint relabeling of the operator and the label signal, whereas relabeling only the operator is not an invariance.

Both statements follow from block multiplication and $P^\top P = I$, as shown in Appendix A. Because conjugation preserves the spectrum, misaligned relations can change the task at an unchanged spectrum, which Section 4.3 tests.

Separating row geometry from the predictor. A tabular in-context predictor maps (X, Y, m) to the query predictions $\widehat{Y}_Q \in \mathbb{R}^{|Q|\times C}$ without parameter updates. We split it into three parts,

$$X \xrightarrow{\text{Equations (2) and (3)}} K \xrightarrow{\mathcal{P}(\cdot,Y,m)} Z_K \xrightarrow{R_\theta} \widehat{Y}_Q, \qquad X \dashrightarrow R_\theta, \tag{5}$$

where $\mathcal{P}$ propagates the support labels with K into a row-level signal $Z_K \in \mathbb{R}^{N\times D}$, with D the width of the predictor, R_θ is a predictor with parameters θ, and the dashed arrow denotes optional direct access to the raw input features. In this sense, K acts as a relational state of the task that can be changed while R_θ stays fixed. Equation (5) makes three tests possible: removing the dashed arrow tests whether (K, Y, m) suffices for prediction, changing only K in a trained model identifies which of its properties are used, and replacing R_θ tests whether the result depends on the predictor.

3.3 Controlled In-Context Model

Building the propagation $G = K$ of Section 3.2 into the model would already assume the answer to the question of Section 1, namely which property of the row relations the models use. Instead, we train in-context models whose row operator is a learned combination of spectral functions of K and of its entrywise square, and we identify the part of this family that is used after training, by replacing the operator while the trained model stays fixed. For $q \in \{1, 2\}$, the model computes

$$H_q = \sum_f \alpha_f \, g_f(K^{\circ q})\, Y, \qquad Z_K = \frac{1}{2}\sum_{q=1}^{2} \left(H_q W_q + \mathbf{1}\beta_q^\top\right), \tag{6}$$

where $K^{\circ q}$ is the entrywise q-th power of K, $\{g_f\}_f$ is a bank of scalar spectral functions applied to the eigenvalues of their argument, the nonnegative coefficients α_f, which sum to one, come from a permutation-invariant summary of the support rows, $H_q \in \mathbb{R}^{N\times C}$ is the propagated class signal, and $W_q \in \mathbb{R}^{C\times D}$ and $\beta_q \in \mathbb{R}^D$ map it into the hidden state of a backbone of width D, forming what we call the geometry stream. In our runs, the bank contains $g(\lambda) = 1$ and the heat functions $e^{-\lambda}$ and $e^{-\sqrt{\lambda}}$, detailed in Appendix B.3, so that K itself appears only as a replacement in Section 4.3. As the bank includes the identity, which carries nothing across rows, and the backbone also receives feature tokens, the geometry stream allows propagation along K without forcing it, as tested in Section 4.3.

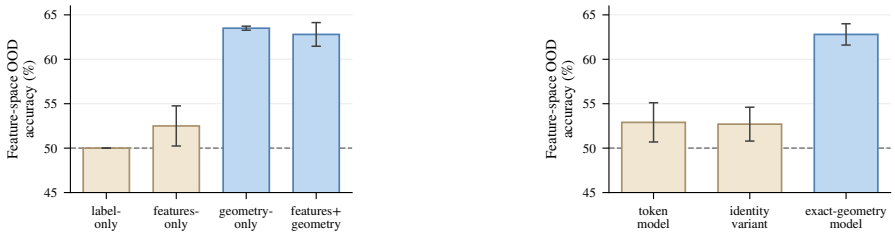
Computational complexity. Forming K costs $\mathcal{O}(N^2 d)$ time, which is quadratic in N as one layer of attention between rows in TabPFN (Hollmann et al., 2025) and TabICL (Qu et al., 2025), while the first-order propagation $KY = \bar{U}(\bar{U}^\top Y)$ costs $\mathcal{O}(NdC)$ time, which is linear in N. The $\mathcal{O}(N^3)$ spectral bank is used by the controlled models of our analysis, and Table 13 and Figure 6 in Appendix E report measured construction times and the accuracy of projected Gram matrices.

4 Experiments

In this section, we conduct an extensive set of experiments to study what tabular in-context models use from row geometry, using the Gram matrix K and the controlled in-context model of Section 3. Because Equation (5) allows us to change one source of information at a time while the rest of the model stays fixed, and Proposition 2 predicts which entries of the row operator matter, our experiments seek to address the following questions:

- **(Q1)** Can the Gram matrix K replace direct access to the raw input features?
- **(Q2)** Which part of the Gram matrix K do the models use?
- **(Q3)** Does the Gram matrix K remain predictive with other predictors and on natural tables?
- **(Q4)** Do pretrained TFMs contain internal row structure similar to the Gram matrix K?

In what follows, we present our main results, and Table 2 lists the appendices with further results.



(a) Input conditions. (b) Matched training budget.

Figure 2: **The Gram matrix K replaces direct access to the raw input features, and the improvement comes from cross-row propagation.** (a) Mean feature-space OOD accuracy (%) of five backbones under the four input conditions. (b) The same metric for six backbones under a matched budget, with a separately trained identity variant that receives the geometry stream without cross-row propagation. Bars show means with the standard deviation across backbones, bars shaded in blue use K, bars shaded in beige are baselines without K, and the dashed line marks chance level.

4.1 Experimental Setup

Tasks. We train on synthetic binary classification tasks from four teacher families (linear, quadratic, axis-aligned tree, and rotated tree), with 8 to 128 raw input features, 64 to 256 support rows, and 128 query rows per task, testing with 192 support rows, as detailed in Appendix B.1. The causal analysis thus covers synthetic binary classification, and Section 4.4 extends it to multiclass natural tables.

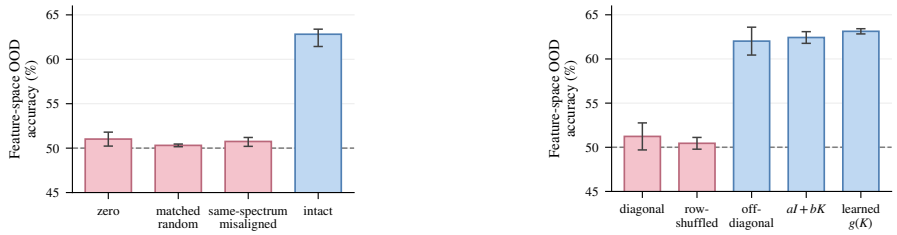
Models. We use six in-context models with different architectures, which we refer to as backbones: retrieval attention, a class-prototype aggregator, a latent-bottleneck perceiver (Jaegle et al., 2021), axial cell attention, non-parametric kernel regression, and a hypernetwork based on MotherNet (Mueller et al., 2025). Each backbone is trained in two variants: a *geometry model*, which receives the Gram matrix K through the geometry stream of Section 3.3 in addition to feature tokens of the raw input features, and a *token model*, which receives only the feature tokens. Five of the backbones can operate on K alone, whereas axial cell attention operates on feature cells and is therefore excluded from the feature-removal comparison. Appendix B.2 describes every backbone and control.

Baselines and controls. We compare the geometry models with the token models and with four categories of baselines and controls, each ruling out an alternative explanation: (1) *input conditions* and a separately trained identity variant, to assess whether the raw input features or the geometry stream explain the improvement; (2) *operator replacements*, which keep the propagation route, its norm, and its spectrum, to evaluate whether correct row attachment is required; (3) *classical predictors*, namely support-adaptive kernel ridge regression (KRR) and k-nearest neighbors (kNN) on K, to test whether K remains predictive without the trained model; and (4) *pretrained comparisons*, which relate the Gram matrix K to frozen TabPFN and TabICL, as detailed in Appendix B.2.

Protocol. Because (Q1) and (Q2) concern changes of the feature space, our main metric is the mean accuracy over three diagnostic suites of feature-space out-of-distribution (OOD) tasks: a feature-count suite, which varies the number of raw input features around a fixed low-dimensional signal, a rotation suite, which rotates the raw input features before canonicalization, changing K, and an irrelevant-feature suite, which adds uninformative features. For (Q3), we report accuracy on single-family tasks and on 25 natural OpenML tables (Vanschoren et al., 2014), and for (Q4), the linear CKA between K and the internal row-interaction maps of frozen TFMs and their AUC when row geometry is added, as described in Sections 4.4 and 4.5. To maintain a fair comparison, we train the token and geometry models with the same curriculum, training budget, and hyperparameter search, and select every model on validation tasks from the training distribution, as elaborated in Appendix B.1. Appendix B.4 details the seeds and intervals, and Table 4 maps experiments to protocols.

4.2 Row Geometry Replaces Direct Feature Access

Setup. Removing the dashed path in Equation (5) while keeping (K, Y, m) tests whether the Gram matrix K and the support labels suffice for prediction. We therefore train the five compatible backbones under four input conditions that differ only in what the model receives: the support labels



(a) Operator replacement at inference.

(b) Reduced operators on the same checkpoint.

Figure 3: **The models rely on correctly attached cross-row propagation, as predicted by Proposition 2.** Feature-space OOD accuracy (%) with the trained model fixed. (a) Zero, matched-random, same-spectrum misaligned, and intact operators over six backbones × five seeds with 95% hierarchical-bootstrap intervals. (b) Reduced operators on one seed-0 checkpoint per backbone, with the standard deviation over backbones. Bars shaded in blue propagate labels along correctly attached relations of K, bars shaded in rose are controls that break it, and the dashed line is chance level.

only (label-only), feature tokens computed from the raw input features (features-only), the Gram matrix K with the support labels (geometry-only), or both (features+geometry).

Results. Removing direct access to the raw input features does not reduce accuracy once the models receive the Gram matrix K and the support labels. As shown in Figure 2a and Table 5, the geometry-only models reach 63.5% mean accuracy on the feature-space OOD tasks, on par with features+geometry (62.8%), whereas features-only reaches 52.5% and label-only remains at chance level (50.0%). We attribute this improvement to cross-row propagation, and not to the geometry stream: under a matched training budget across all six backbones, the separately trained identity variant reaches 52.7%, on par with the tuned token model (52.9%), whereas the exact-geometry model reaches 62.8%, as shown in Figure 2b. In particular, the accuracy advantage over the token model ranges from 7.3% to 13.3% across backbones, with a mean of $+9.9$ in Table 9, and it also appears on IID tasks, where the exact-geometry model reaches 64.9% compared with 53.5% for the token model, as reported in Table 11. Moreover, a projected Gram matrix retains most of this advantage with 60.8%, whereas no source of row geometry helps without propagation, as Table 10 shows. Therefore, the Gram matrix K, itself computed from the raw input features, and the support labels suffice for these models, and direct access to those features is no longer needed.

4.3 Correct Row Correspondence and Cross-Row Propagation

Setup. Proposition 2 makes two predictions for a model that uses the Gram matrix K for support-to-query propagation. First, labels can reach the queries only through the cross-row support–query entries of the operator. Second, the effect depends on these entries remaining attached to the correct support and query rows. We test both predictions after training, keeping each trained model fixed and replacing only its row operator. The first family of replacements, evaluated over six backbones and five seeds, compares the trained (intact) operator with zero, a random matrix with the same Frobenius norm, and a same-spectrum operator attached to misaligned rows. The second family, evaluated on one trained checkpoint per backbone, replaces the learned spectral response by the affine combination $aI + bK$ with scalars $a, b \in \mathbb{R}$, by its off-diagonal part, by a diagonal, or by a row-shuffled operator. These models also receive feature tokens, so that a trained model can predict from the features, from K, or from both. Replacing only the row operator therefore reveals which one the model uses.

Row correspondence. As we show in Figure 3, both predictions of Proposition 2 hold, and the models require correct row correspondence. The intact operator reaches 62.82% accuracy, with a 95% hierarchical-bootstrap interval of $[61.44, 63.39]$, whereas the zero, matched-random, and same-spectrum misaligned operators yield 51.02%, 50.31%, and 50.75%. Each paired difference exceeds 11.8%, and all intervals exclude zero, as reported in Table 6 in Appendix C. On IID tasks, replacing the operator by zero likewise reduces accuracy from 62.65% to 51.06%. The random operator matches the overall magnitude of the trained operator, and the misaligned operator keeps its eigenvalues, yet both bring accuracy to chance level. Therefore, the effect depends on each relation connecting the right rows, rather than on the magnitude or eigenvalues of the operator.

Table 1: **The Gram matrix K remains predictive when the neural model is replaced.** Accuracy (%) ↑ on single-family tasks, ± the standard error, detailed in Appendix B.1. Rows shaded in blue use K, and the row shaded in beige is the baseline without K. The highest accuracy in each column is shown in **bold** and the second highest is underlined; tied values share the mark.

	Teacher family						
Predictor	Linear	Quadratic	Cubic	Tree	Rotated	Nuisance	Average
Token Model (Retrieval)	77.1±0.6	57.4±0.3	57.9±0.3	60.3±0.4	59.9±0.5	58.7±0.5	61.9
KRR on K	**90.1**±0.1	70.3±0.4	**60.2**±0.4	62.5±0.5	62.6±0.4	69.9±0.5	69.3
Geometry Model (MotherNet) (Ours)	88.9±0.3	**72.2**±0.3	58.4±0.4	63.8±0.4	63.8±0.4	**75.1**±0.2	70.4
Geometry Model (Retrieval) (Ours)	88.9±0.3	72.1±0.4	59.2±0.4	**64.0**±0.4	**64.0**±0.4	**75.1**±0.3	**70.6**

Form of the propagation. The second family shows that first-order propagation retains nearly all of the effect. Replacing the learned spectral response (63.13%) by the affine combination $aI + bK$ yields 62.43%, and keeping only its off-diagonal part gives 62.02%. In contrast, the diagonal and row-shuffled operators yield 51.23% and 50.45%, in agreement with the support–query block identity in Proposition 2, as reported in Table 7. Moreover, $aI + bK$ remains within 2.0% of the trained operator on every backbone, as shown in Table 8. Therefore, the models rely on first-order propagation of support labels along correctly attached support–query relations, and the additional spectral functions in the bank of Equation (6) are not needed for this effect. As we show in Appendices C and D, the effect is independent of interactions between queries, and the advantage over the token model stays between 15.9% and 28.6% under every transformation family of Table 12 and Figure 5.

4.4 Row Geometry Across Predictors and Natural Tables

We next provide the Gram matrix K to other predictors, separating the representation from the model.

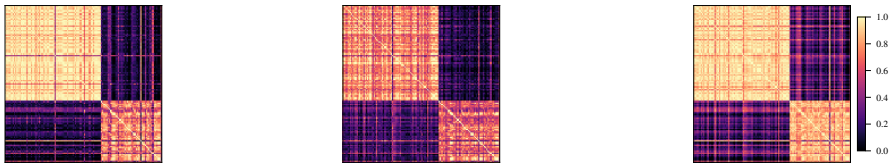
Classical predictors. As we show in Table 1, the Gram matrix K remains predictive after the neural model is removed. KRR on K, a kernel predictor on the canonicalized rows that selects its kernel for each task by leave-one-out validation on the support rows, reaches 69.3% average accuracy, close to the 70.6% and 70.4% of the geometry models with the retrieval and MotherNet backbones. KRR is the most accurate predictor on the linear and cubic families, it stays close to the most accurate neural model on the quadratic, tree, and rotated families, and it outperforms the token model on every family. Because KRR has no trained weights, its accuracy relies only on K and the support labels, so that the information used by the geometry models is already present in K.

Natural tables. As we show in Appendix G, the Gram matrix K also remains predictive on natural data. On the 25 OpenML tables of Table 17, selected by a fixed inclusion rule, KRR and kNN on K reach 80.0% and 78.8% dataset-clustered accuracy, with 95% dataset-bootstrap intervals of $[75.1, 84.8]$ and $[73.6, 83.9]$. A predictor that adapts to each table can therefore use K directly, and these tables measure how much predictive information K retains, rather than compare K with other kernels, tabular models, or TFMs. Moreover, K follows how much information the rows carry rather than how the columns are expressed: per-column affine changes, permutations, and rescaling change K by at most 0.16 and keep KRR accuracy within 0.1%, whereas nuisance features, missingness, and deleting target-relevant features lower it by 2.7% to 6.0%, as reported in Table 18 and Figure 9.

4.5 Row Geometry in Pretrained TFMs

Finally, we compare K descriptively with the row structure inside frozen TabPFN and TabICL.

Internal row structure. As we show in Figure 4 and Appendix F, both pretrained TFMs build internal row structure that closely matches the Gram matrix K, although neither was trained with it. Across the 25 OpenML tables, the linear CKA between their internal row-interaction maps and K is 0.624 for TabPFN and 0.676 for TabICL on average over layers, and 0.903 and 0.831 at the most similar layer, whereas shuffling the rows of K lowers it to 0.035. In a case study on BREAST_W, the similarity is not explained by the class labels alone: after removing the label kernel from both



(a) Gram matrix K. (b) TabPFN (CKA with K: 0.95). (c) TabICL (CKA with K: 0.96).

Figure 4: **Frozen TFMs contain row-interaction structure that closely matches the Gram matrix K.** The input Gram matrix K on BREAST_W and the most similar internal row-interaction maps of frozen TabPFN and TabICL, over the support rows sorted by class; each map is rescaled to $[0, 1]$.

matrices, the partial CKA remains 0.79 for TabPFN and 0.82 for TabICL, against 0.03 for K shuffled within classes. Therefore, pretrained TFMs form row relations closely related to K on their own.

Adding row geometry to frozen TFMs. As a diagnostic intervention, we also add a row-geometry bias, scaled by γ, to the row attention of the frozen models, formed from the Gram matrix K for TabPFN and from TabICL's own representations. As we show in Table 14 and Figure 7, frozen TabPFN tells the correct row geometry apart from a random one: on natural tables, the aligned bias gives a 4.59% higher area under the receiver operating characteristic (ROC) curve (AUC) than a matched-random bias, averaged over the 25 tables, while keeping TabPFN close to its unmodified AUC (78.4% versus 78.8%). For TabICL, whose bias comes from its own representations, the aligned bias lies below the unmodified model. Where frozen TabPFN is weak, K also improves its predictions: on synthetic product structures, where TabPFN v2.5 (Grinsztajn et al., 2025) and v3 (Grinsztajn et al., 2026) reach 51.4% and 54.6% AUC, adding K raises their AUC by 14.7% and 10.9%, and a row Gram bias beats a matched-random one on all 10 control tasks, as reported in Tables 15 and 16. Therefore, the predictions of TabPFN are sensitive to the alignment of the row geometry it is given, and this geometry helps most on the structures that the pretrained model handles least accurately.

4.6 Discussion

Our experiments span six in-context backbones, three suites of feature-space OOD tasks, 25 natural tables, and three frozen TFMs, and they lead to three insights. (i) *The relations between rows carry the information needed for prediction*: in the controlled models, K replaces direct access to the raw input features on every backbone (Section 4.2, Figure 2), and classical predictors on K remain accurate on natural tables (Section 4.4, Table 1, Appendix G), so that support-to-query prediction in tabular ICL can be carried by the relations between rows. (ii) *The models use these relations through first-order propagation along correctly attached rows*: with the trained models fixed, the first-order operator $aI + bK$ recovers nearly all of the effect, which depends on each relation connecting the right rows, as Proposition 2 predicts, rather than on the magnitude or eigenvalues of the operator (Section 4.3, Figure 3). (iii) *Pretrained TFMs form closely related row structure*: frozen TabPFN and TabICL build internal row relations that closely match K (Section 4.5, Figure 4), which on BREAST_W persist after the label kernel is removed (Appendix F), TabPFN distinguishes an aligned K from a random one on average over the 25 tables (Table 14), and adding K improves TabPFN where it is weak (Tables 15 and 16). This descriptive and conditional evidence complements, rather than replaces, the causal evidence of the controlled models. In summary, row geometry emerges as a consistent representation for tabular ICL.

5 Conclusion

In this paper, we asked whether support-to-query prediction in tabular ICL can be carried by the relations between rows alone, and which property of those relations is actually used. To address this question, we separated the row relations from the predictor through the row Gram matrix K. Our theory supports this choice, and our experiments show that, in the controlled settings we study, K can replace direct access to the raw input features, and the models rely on first-order propagation along correctly attached support–query relations, which also gives a practical test of what a tabular in-context model uses once its row relations can be intervened on. By making what tabular in-context learners use directly testable, we envision a positive impact on the design and analysis of TFMs.

Reproducibility Statement

We will release the code, the experiment configurations, the trained checkpoints, the raw and processed results, and the scripts that reproduce every reported table and figure upon acceptance. The appendix specifies statistical units, parameter counts, hyperparameter searches, natural-table inclusion rules, and the geometry biases added to frozen models.

Ethics Statement

This work uses synthetic tasks and public OpenML tables, and it involves no human subjects, personal data, or data collection. We are not aware of harmful applications that are specific to this study, which analyzes what tabular in-context models use and proposes no new deployment.

AI Use Statement

In this work, we used generative AI tools to help with implementing methods under the supervision of the authors, including creating and editing code, to explore related research, which the authors then read themselves, and to polish the authors' own written draft for readability. We did not use them for any other task that requires disclosure: generating synthetic datasets, developing theoretical models or conceptual frameworks, formulating mathematical claims or providing critical ingredients for their proofs, writing proofs, proposing or refining hypotheses, designing or giving feedback on the methodology or experiments, translation, cleaning or reformatting datasets, qualitative or thematic data analysis, and interpreting results. We have reviewed all AI-assisted work: the authors verified and tested all AI-assisted code for correctness, reviewed and approved all AI-assisted text, and checked all suggested literature. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

References

Dimitris Achlioptas. Database-friendly random projections: Johnson–Lindenstrauss with binary coins. *Journal of Computer and System Sciences*, 66(4):671–687, 2003. doi: 10.1016/S0022-0000(03)00025-4.

Stefan Aeberhard and M. Forina. Wine. UCI Machine Learning Repository, https://doi.org/10.24432/C5PC7J, 1992.

Amir Rezaei Balef, Mykhailo Koshil, and Katharina Eggensperger. Is one layer enough? Understanding inference dynamics in tabular foundation models. In *International Conference on Machine Learning (ICML)*, 2026a.

Amir Rezaei Balef, Mykhailo Koshil, Behzad Nourani-Koliji, and Katharina Eggensperger. Do tabular foundation models learn rules or memorize exemplars? In *2nd ICML Workshop on Foundation Models for Structured Data*, 2026b. URL https://openreview.net/forum?id=9nCMtYGxQt.

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

Marin Biloš, James T. Wilson, Anderson Schneider, and Yuriy Nevmyvaka. A mechanistic study of tabular foundation models. *arXiv preprint arXiv:2605.21288*, 2026.

David Bonet, Daniel Mas Montserrat, Xavier Giró-i-Nieto, and Alexander G. Ioannidis. HyperFast: Instant classification for tabular data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 11114–11123, 2024. doi: 10.1609/aaai.v38i10.28988.

E. Bouveresse and D. L. Massart. Improvement of the piecewise direct standardisation procedure for the transfer of NIR spectra for multivariate calibration. *Chemometrics and Intelligent Laboratory Systems*, 32(2):201–213, 1996.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901, 2020.

Zi-Jian Cheng, Ziyi Jia, Zhi Zhou, Yu-Feng Li, and Lan-Zhe Guo. TabFSBench: Tabular benchmark for feature shifts in open environments. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 10025–10089, 2025.

Ronald R. Coifman and Stéphane Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, 2006. doi: 10.1016/j.acha.2006.04.006.

Ronald A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7 (2):179–188, 1936.

Victor Garcia Satorras and Joan Bruna Estrach. Few-shot learning with graph neural networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=BJj6qGbRW.

Yury Gorishniy, Ivan Rubachev, Nikolay Kartashev, Daniil Shlenskii, Akim Kotelnikov, and Artem Babenko. TabR: Tabular deep learning meets nearest neighbors. In *International Conference on Learning Representations*, 2024.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Benjamin Jäger, Dominik Safaric, Simone Alessi, Adrian Hayler, Mihir Manium, Rosen Yu, Felix Jablonski, Shi Bin Hoo, Anurag Garg, Jake Robertson, Magnus Bühler, Vladyslav Moroshan, Lennart Purucker, Clara Cornu, Lilly Charlotte Wehrhahn, Alessandro Bonetto, Bernhard Schölkopf, Sauraj Gambhir, Noah Hollmann, and Frank Hutter. TabPFN-2.5: Advancing the state of the art in tabular foundation models. arXiv preprint arXiv:2511.08667, 2025.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Mihir Manium, Shi Bin Hoo, Magnus Bühler, Anurag Garg, Dominik Safaric, Jake Robertson, Benjamin Jäger, Simone Alessi, Adrian Hayler, Vladyslav Moroshan, Lennart Purucker, Philipp Singer, Alan Arazi, Julien Siems, Jan Hendrik Metzen, Georg Grab, Nick Erickson, Siyuan Guo, Eliott Kalfon, Simon Bing, David Salinas, Clara Cornu, Lilly Charlotte Wehrhahn, Diana Kriuchkova, Kursat Kaya, Lydia Sidhoum, Marie Salmon, Jerry Chen, Madelon Hulsebos, Yann LeCun, Samuel Müller, Bernhard Schölkopf, Sauraj Gambhir, Noah Hollmann, and Frank Hutter. TabPFN-3: Technical report. arXiv preprint arXiv:2605.13986, 2026.

Beimnet Bekele Guta. Localized TabICLv2: Scaling tabular in-context learning through k-NN. In *2nd ICML Workshop on Foundation Models for Structured Data*, 2026. URL https://arxiv.org/abs/2608.16429.

Nathan Halko, Per-Gunnar Martinsson, and Joel A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53 (2):217–288, 2011. doi: 10.1137/090771806.

Chi Han, Ziqi Wang, Han Zhao, and Heng Ji. Understanding emergent in-context learning from a kernel regression perspective. *Transactions on Machine Learning Research*, 2025.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=cp5PvcI6w8_.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637:319–326, 2025. doi: 10.1038/s41586-024-08328-6.

Yixiang Huang, Hongyu Hao, Weichao Ge, Yang Cao, Ming Wu, Chuang Zhang, and Jun Guo. Relation fusion propagation network for transductive few-shot learning. *Pattern Recognition*, 151: 110367, 2024. doi: 10.1016/j.patcog.2024.110367.

Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver: General perception with iterative attention. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 4651–4664, 2021.

Purushottam Kar and Harish Karnick. Random feature maps for dot product kernels. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*, volume 22 of *Proceedings of Machine Learning Research*, pp. 583–591. PMLR, 2012. URL https://proceedings.mlr.press/v22/kar12.html.

Jongmin Kim, Taesup Kim, Sungwoong Kim, and Chang D. Yoo. Edge-labeling graph neural network for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

Vladimir Koltchinskii and Karim Lounici. Concentration inequalities and moment bounds for sample covariance operators. *Bernoulli*, 23(1):110–133, 2017.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3519–3529. PMLR, 2019.

Mykhailo Koshil, Matthias Feurer, and Katharina Eggensperger. In-context learning of soft nearest neighbor classifiers for intelligible tabular machine learning. In *Proceedings of the 4th Table Representation Learning Workshop*, pp. 182–191, Vienna, Austria, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.trl-1.15.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3744–3753. PMLR, 2019.

Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, Eunho Yang, Sung Ju Hwang, and Yi Yang. Learning to propagate labels: Transductive propagation network for few-shot learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=SyVuRiC5K7.

Junwei Ma, Valentin Thomas, Rasa Hosseinzadeh, Alex Labach, Hamidreza Kamkari, Jesse C. Cresswell, Keyvan Golestan, Guangwei Yu, Anthony L. Caterini, and Maksims Volkovs. TabDPT: Scaling tabular foundation models on real data. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-5748. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/fc0e3f908a2116ba529ad0a1530a3675-Abstract-Conference.html.

Yuqing Ma, Shihao Bai, Shan An, Wei Liu, Aishan Liu, Xiantong Zhen, and Xianglong Liu. Transductive relation-propagation network for few-shot learning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pp. 804–810, 2020.

Ratmir Miftachov, Bruno Charron, and Simon Valentin. Inspectable tabular foundation models via in-context kernel learning. In *2nd ICML Workshop on Foundation Models for Structured Data*, 2026.

Andreas Mueller, Carlo Curino, and Raghu Ramakrishnan. MotherNet: Fast training and inference via hyper-network transformers. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=6H4jRWKFc3.

Lennart Purucker, Andrej Tschalzev, Nick Erickson, Gioia Blayer, David Holzmüller, Alan Arazi, Alexander Pfefferle, Mustafa Tajjar, Gaël Varoquaux, and Frank Hutter. Beyond IID: How general are tabular foundation models, really? arXiv preprint arXiv:2606.30410, 2026. URL https://arxiv.org/abs/2606.30410.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICL: A tabular foundation model for in-context learning on large data. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 50817–50847. PMLR, 2025. URL https://proceedings.mlr.press/v267/qu25d.html.

Ivan Rubachev, Akim Kotelnikov, Nikolay Kartashev, and Artem Babenko. On finetuning tabular foundation models. arXiv preprint arXiv:2506.08982, 2025. URL https://arxiv.org/abs/2506.08982.

Nour Shaheen, Junwei Ma, Alex Labach, Frank Hutter, Valentin Thomas, and Anthony L. Caterini. Understanding the surprising generalization properties of tabular foundation models. arXiv preprint arXiv:2608.17957, 2026. URL https://arxiv.org/abs/2608.17957.

Alexander J. Smola and Risi Kondor. Kernels and regularization on graphs. In *Learning Theory and Kernel Machines: 16th Annual Conference on Learning Theory and 7th Kernel Workshop, COLT/Kernel 2003*, volume 2777 of *Lecture Notes in Computer Science*, pp. 144–158. Springer, 2003.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Valentin Thomas, Junwei Ma, Rasa Hosseinzadeh, Keyvan Golestan, Guangwei Yu, Maksims Volkovs, and Anthony Caterini. Retrieval & fine-tuning for in-context tabular models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3442. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/c40daf14d7a6469e65116507c21faeb7-Abstract-Conference.html.

Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. OpenML: Networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2):49–60, 2014.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, volume 29, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/90e1357833654983612fb05e3ec9148c-Abstract.html.

Jonas Wacker, Motonobu Kanagawa, and Maurizio Filippone. Improved random features for dot product kernels. *Journal of Machine Learning Research*, 25(235):1–75, 2024.

Yongdong Wang, David J. Veltkamp, and Bruce R. Kowalski. Multivariate instrument standardization. *Analytical Chemistry*, 63(23):2750–2756, 1991.

Zifeng Wang and Jimeng Sun. TransTab: Learning transferable tabular transformers across tables. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://arxiv.org/abs/2205.09328.

Hermann Weyl. *The Classical Groups: Their Invariants and Representations*. Princeton University Press, second edition, 1946.

David P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1–2):1–157, 2014. doi: 10.1561/0400000060.

Derek Xu, Olcay Cirit, Reza Asadi, Yizhou Sun, and Wei Wang. Mixture of in-context prompters for tabular PFNs. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/3de60dc2ff7ac8866e805bbf457e2291-Abstract-Conference.html.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabás Póczos, Ruslan Salakhutdinov, and Alexander J. Smola. Deep sets. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Dengyong Zhou, Olivier Bousquet, Thomas N. Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems*, volume 16, pp. 321–328, 2003.

Bingzhao Zhu, Xingjian Shi, Nick Erickson, Mu Li, George Karypis, and Mahsa Shoaran. XTab: Cross-table pretraining for tabular transformers. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 43181–43204. PMLR, 2023. URL https://proceedings.mlr.press/v202/zhu23k.html.

Hao Zhu and Piotr Koniusz. Transductive few-shot learning with prototype-based label propagation by iterative graph refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23996–24006, June 2023.

Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pp. 912–919, 2003.

Imtiaz Ziko, Jose Dolz, Eric Granger, and Ismail Ben Ayed. Laplacian regularized few-shot learning. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 11660–11670. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/ziko20a.html.

ORGANIZATION OF THE APPENDIX

Table 2 summarizes the content of each appendix and the part of the main text that it supports. In particular, Appendices C to G report the complete results of the additional experiments, organized by the questions (Q1)–(Q4) of Section 4.

Table 2: **Organization of the appendix.** Each appendix, its content, and the part of the main text that it supports.

Appendix	Content	Supports
THEORY AND PROTOCOLS		
Appendix A	Proofs of Propositions 1, 2 and 3, invariances of support-conditioned canonicalization, and the information kept by the Gram matrix K	Section 3
Appendix B	Tasks, backbones, baselines and controls, training and selection protocol, statistical units, and the natural-table and frozen-TFM protocols	Section 4.1
ADDITIONAL EXPERIMENTS		
Appendix C	Complete operator-replacement results, per-backbone matched-budget and reduced-operator results, IID and per-suite accuracy, and single-query evaluation	(Q1), (Q2)
Appendix D	Transformation families, and correlational analysis of geometry alignment and signal retention across them	(Q2)
Appendix E	First-order propagation without forming K, and approximations of K for large d	Section 3.3
Appendix F	Layer-wise similarity between frozen TFMs and K, complete results of the added row-geometry bias on natural tables, and results on synthetic structures where frozen TabPFN is weak	(Q4)
Appendix G	Admitted OpenML tables, complete transfer results, shift families, canonicalization fidelity, feature deletion, non-overlapping schemas, and instrument transfer	(Q3)
BACKGROUND		
Appendix H	Extended discussion of related work	Section 2

A REPRESENTATION PROPERTIES AND PROOFS

The main text uses two exact properties of the Gram matrix K and one approximation result. We first prove Propositions 1 and 2, then state which transformations the canonicalization removes and show that the operators are permutation equivariant, and finally prove the sketch bound and describe what information the Gram matrix K keeps. These results describe the representation and its operators, while the Gram matrix K is enough for prediction in the controlled settings, as shown by the experiments in Sections 4.2 and 4.3.

Proof of Proposition 1. The reverse implication is immediate: if $B = AO$ with $O \in \mathrm{O}(d)$, then $BB^\top = AOO^\top A^\top = AA^\top$. For the forward implication, let $AA^\top = BB^\top$ have rank r. If $r = 0$, then $A = B = 0$ and $O = I$ suffices. Otherwise, write the eigendecomposition $AA^\top = E\Sigma^2 E^\top$, where $E \in \mathbb{R}^{N\times r}$ has orthonormal columns and $\Sigma \in \mathbb{R}^{r\times r}$ is diagonal with positive entries, and define $V = A^\top E\Sigma^{-1}$ and $W = B^\top E\Sigma^{-1}$ in $\mathbb{R}^{d\times r}$. Then $V^\top V = \Sigma^{-1}E^\top AA^\top E\Sigma^{-1} = I_r$, and likewise $W^\top W = I_r$. Moreover, the column space of A equals the range of $AA^\top$, which is spanned by E, so that $EE^\top A = A$ and hence $E\Sigma V^\top = EE^\top A = A$; in the same way, $E\Sigma W^\top = B$. We extend V and W to orthogonal bases $[V, V_\perp]$ and $[W, W_\perp]$ of $\mathbb{R}^d$, and define $O = VW^\top + V_\perp W_\perp^\top$. Because $W^\top W_\perp = 0$ and $V^\top V_\perp = 0$, we have $OO^\top = VV^\top + V_\perp V_\perp^\top = I$, so O is orthogonal, and $V^\top O = W^\top$, hence $AO = E\Sigma V^\top O = E\Sigma W^\top = B$. For the factorization statement, define $f(\mathcal{K}, Y, m) = F(A, Y, m)$ for any realization A with $AA^\top = \mathcal{K}$. The first part implies that two such realizations differ by a right orthogonal factor, so the assumed invariance of F makes f well defined. □

Invariances of the canonicalization. For the idealized $\varepsilon_s = 0$ standardizer, assume that every raw input feature has nonzero support variance and write $X' = X\,\mathrm{diag}(\kappa) + \mathbf{1}\tau^\top$ with a rescaling vector $\kappa \in \mathbb{R}^d$ whose entries satisfy $\kappa_j \neq 0$ and a shift vector $\tau \in \mathbb{R}^d$. The recomputed support means become $\kappa_j\mu_{S,j} + \tau_j$ and the support standard deviations scale by $|\kappa_j|$, so the standardized features change only by right multiplication with the sign matrix $\Delta = \mathrm{diag}(\mathrm{sign}(\kappa_1), \ldots, \mathrm{sign}(\kappa_d))$. Symmetric clipping commutes with these signs, so that $U' = U\Delta$, and row norms are unchanged, so that $\bar{U}' = \bar{U}\Delta$ and $K' = \bar{U}\Delta\Delta^\top\bar{U}^\top = K$. A feature permutation $X' = XJ$, with J a $d \times d$ permutation matrix, permutes the support means and standard deviations in the same way, so that

$U' = UJ$, because entrywise clipping commutes with J, and again $K' = K$. In the implemented map, the fixed stabilizer $s_{S,j} = \sqrt{v_{S,j} + \varepsilon_s}$ makes arbitrary rescaling approximate rather than exact, while shifts, feature permutations, and sign flips remain exact. Any orthogonal action $U \mapsto UO$ applied after canonicalization preserves row norms, so that $\bar{U} \mapsto \bar{U}O$, and therefore leaves K unchanged.

Proof of Proposition 2. Block multiplication gives

$$GY = \begin{bmatrix} G_{SS}Y_S \\ G_{QS}Y_S \end{bmatrix},$$

because $Y_Q = 0$, so the query signal depends only on G_{QS} and vanishes for diagonal G because G_{QS} is then zero. For a permutation matrix P, $(PGP^\top)(PY) = PG(P^\top P)Y = P(GY)$. Thus joint row relabeling is equivariant. In contrast, conjugating G alone gives $(PGP^\top)Y$, which differs from GY in general, so conjugation with the row-indexed signal and mask held fixed is not an invariance. □

Well-posedness of the operator family. $K = \bar{U}\bar{U}^\top$ is positive semidefinite (PSD). For integer $q \geq 1$, $K^{\circ q}$ is a repeated Hadamard product of PSD matrices and is PSD by the Schur product theorem. Under a consistent row permutation, $(PKP^\top)^{\circ q} = PK^{\circ q}P^\top$. Any scalar spectral function g respects orthogonal similarity, so $g(PK^{\circ q}P^\top) = Pg(K^{\circ q})P^\top$ for every permutation matrix P; multiplying the consistently permuted label signal and applying a row-wise affine map preserves row equivariance. Because the coefficients α_f are produced by a permutation-invariant summary of the task, they are unchanged by the permutation.

Proposition 3 (Concentration of a normalized-row Gaussian sketch)**.** *Fix $\bar{U}$ from Equation (3), draw a projection matrix $\Omega \in \mathbb{R}^{d\times h}$ of width h with independent $\mathcal{N}(0,1)$ entries, form $\widetilde{U} = \bar{U}\Omega/\sqrt{h}$, and set $\widehat{K} = \widetilde{U}\widetilde{U}^\top$. Then $\mathbb{E}[\widehat{K}] = K$. Let*

$$\tilde{r}(K) = \frac{\mathrm{tr}(K)}{\|K\|_2} = \frac{\|\bar{U}\|_F^2}{\|\bar{U}\|_2^2}$$

denote the effective rank of K, where tr is the trace, $\|\cdot\|_2$ the spectral norm, and $\|\cdot\|_F$ the Frobenius norm. If $K \neq 0$, there is a universal constant c_0 such that, for every $t \geq 1$, with probability at least $1 - e^{-t}$,

$$\frac{\|\widehat{K} - K\|_2}{\|K\|_2} \leq c_0 \max\left\{\sqrt{\frac{\tilde{r}(K)}{h}}, \frac{\tilde{r}(K)}{h}, \sqrt{\frac{t}{h}}, \frac{t}{h}\right\}.$$

Proof of Proposition 3. Let $\omega_1, \ldots, \omega_h$ be the columns of Ω and $z_j = \bar{U}\omega_j$. Then $\widehat{K} = h^{-1}\sum_{j=1}^h z_j z_j^\top$, and the z_j are i.i.d. centered Gaussian vectors with covariance K, giving $\mathbb{E}[\widehat{K}] = K$. Let $r = (\mathbb{E}\|z_1\|_2)^2/\|K\|_2$. For i.i.d. centered Gaussian vectors, Koltchinskii & Lounici (2017) show that $\mathbb{E}\|\widehat{K} - K\|_2 \leq C\|K\|_2 \max\{\sqrt{r/h}, r/h\}$, and that for every $t \geq 1$, with probability at least $1 - e^{-t}$, the deviation $\left|\|\widehat{K} - K\|_2 - \mathbb{E}\|\widehat{K} - K\|_2\right|$ is at most $C\|K\|_2 \max\{\sqrt{t/h}, t/h\}$ if $r \leq h$, and at most $C\|K\|_2 \max\{\sqrt{r/h}\sqrt{t/h}, t/h\}$ if $r > h$, for a universal constant C. In the second case, $\sqrt{r/h}\sqrt{t/h} \leq \frac{1}{2}(r/h + t/h) \leq \max\{r/h, t/h\}$. Adding the expectation bound and the deviation bound therefore gives $\|\widehat{K} - K\|_2 \leq 2C\|K\|_2 \max\{\sqrt{r/h}, r/h, \sqrt{t/h}, t/h\}$ in both cases. Finally, $(\mathbb{E}\|z_1\|_2)^2 \leq \mathbb{E}\|z_1\|_2^2 = \mathrm{tr}(K)$ gives $r \leq \tilde{r}(K)$, and the bound is nondecreasing in r, which yields the statement with $c_0 = 2C$. Because $K = \bar{U}\bar{U}^\top$, $\mathrm{tr}(K) = \|\bar{U}\|_F^2$ and $\|K\|_2 = \|\bar{U}\|_2^2$. □

Information kept by the Gram matrix. If an observation transform maps two latent rows with different label distributions to the same observed row, then two queries at these latent rows receive the same row of K, so that swapping the two queries leaves (K, Y, m) unchanged, and any permutation-equivariant predictor that depends only on (K, Y, m) assigns them the same prediction, although their label distributions differ. The feature-deletion and non-overlapping-schema experiments in Appendix G are instances of this case, and paired calibration standards supply cross-domain correspondence in addition to what the unpaired rows contain.

Relation between theory and implementation. Propositions 1 and 2 concern the Gram matrix K and the label signal it acts on, while the experiments show that the Gram matrix K is enough for prediction in the controlled settings. Proposition 3 applies to the normalized-row Gaussian

sketch above; the project-before-normalize variant is a different approximation, which we evaluate empirically in Appendix E.

B Experimental Setup and Reproducibility

The experiments in Section 4 use several protocol families, so this appendix separates the common controlled setup from the natural-table and frozen-model evaluations. Appendices B.1, B.3 and B.4 specify the controlled tasks, the exact operator path, and the statistical units; Appendices B.5 and B.6 then give the natural-table and pretrained protocols. The subsequent sections follow the order of the experiments: Appendix C gives the full operator-replacement results, Appendix D the correlational diagnostic, Appendix E approximation and scale, Appendix F the frozen-TFM measurements, and Appendix G the natural tables and the experiments on what information the Gram matrix K keeps.

B.1 Task Distribution and Training

The controlled models are trained on a mixture of four teacher families: linear (order one), quadratic (order two), axis-aligned tree, and rotated tree. Training tasks have $d \in \{8, 16, 32, 64, 128\}$ raw input features, 64 to 256 labeled support rows, and 128 query rows; a random rotation is applied with probability 0.3, and up to 64 irrelevant features are appended with probability 0.3. The test tasks use the same four families with 192 support rows. The feature-count suite uses $d \in \{384, 512, 768\}$ raw input features around a fixed low-dimensional signal, the rotation suite applies rotations of strength 0, 0.5, and 1, and the irrelevant-feature suite appends 0, 128, or 512 uninformative features, whereas the IID tasks use $d \in \{32, 64, 128\}$. Table 1 additionally evaluates the seed-0 checkpoints of the retrieval and MotherNet models on single-family task sets with $d = 16$, 192 support rows, and 192 query rows, with 80 tasks per family in five repetitions of 16, including cubic and nuisance-feature teachers (16 signal and 112 irrelevant features) that are not part of the training mixture. The standard errors in Table 1 are computed over the five repetitions, and these single-family tasks are easier for feature tokens than the four-family suites of Figures 2 and 3. The token models converge on the training mixture: their final training accuracy is close to their IID validation accuracy, for example 59.7% and 57.1% for retrieval attention, about 53% and 53.8% for kernel regression, and about 50% and 50.2% for the prototype aggregator. Their lower accuracy therefore reflects the difficulty of the four-family mixture for feature tokens, whose signal lies in a low-dimensional subspace of a high and variable number of raw input features. The six backbones are retrieval attention, a class-prototype (Snell et al., 2017) and DeepSet (Zaheer et al., 2017) aggregator, a latent-bottleneck perceiver, axial cell attention as in TabICL (Qu et al., 2025), non-parametric kernel regression, and a hypernetwork based on MotherNet (Mueller et al., 2025). Token and geometry variants use the same context curriculum and model-selection budget within the matched controlled families. The feature-removal comparison does not include axial cell attention, because its representation is defined over feature cells, making removal of the raw feature stream ill-defined.

Training and checkpoint selection use validation tasks that are independent and identically distributed (IID) with the training tasks. The feature-space OOD suites are held out entirely: they are not used for training, for checkpoint selection, or for any hyperparameter choice, so that every OOD accuracy we report measures how a model trained and selected on the training distribution handles a change of the feature space. For a fair comparison, the geometry stream is small, adding only 3,429 trainable parameters to each backbone, at most 1.0% of its size. The main controlled metric is classification accuracy. The matched-budget identity-variant family and the exact-geometry/token comparison use the same learning-rate/weight-decay search and at least five training seeds. The reduced-operator evaluation instead uses one seed-0 checkpoint per backbone, and we report the two protocols separately throughout the paper. Table 3 summarizes the training configuration.

B.2 Backbones, Baselines, and Controls

Backbones. The six backbones cover different ways in which an in-context model can relate queries to labeled support rows: retrieval attention, in which each query attends to the support rows; a class-prototype aggregator, which summarizes the labeled support rows of each class (Snell et al., 2017); a latent-bottleneck perceiver (Jaegle et al., 2021), which routes the rows through a small set of latent vectors; axial cell attention, which attends along both the rows and the feature cells;

Table 3: **Controlled training and model-selection protocol.** The learning-rate and weight-decay grid is selected on IID validation before the final five-seed runs; feature-space OOD tasks are not used for selection.

Group	Setting	Value
MODEL	Hidden width	128
	Layers / heads	4 / 4
	Dropout	0.1
	Precision	bfloat16
TRAINING	Optimizer	AdamW
	Steps / batch size	2,500 / 64
	Warmup	100 steps
	Schedule	cosine, 5% floor
	Gradient clip	1.0
SELECTION	Learning rate	$\{2, 6, 18\} \times 10^{-4}$
	Weight decay	$\{0, 0.05\}$
	Final seeds	0–4
COMPUTE	Accelerator	NVIDIA B200

non-parametric kernel regression, which predicts each query from a kernel-weighted combination of the support labels; and a hypernetwork based on MotherNet (Mueller et al., 2025), which generates the weights of a small classifier from the support set. Each backbone is trained as a token model, which receives feature tokens of the raw input features, and as a geometry model, which additionally receives the Gram matrix K through the geometry stream of Equation (6). The token models with retrieval attention, axial cell attention, and kernel regression already relate queries to labeled support rows through affinities computed from their feature tokens. Therefore, the geometry models differ from them in the affinity along which the support labels propagate, namely the Gram matrix K, and not in the presence of a route from the support rows to the queries.

Input conditions and the identity variant. The input conditions of Figure 2a give a backbone the support labels only (label-only), feature tokens only (features-only), the Gram matrix K and the support labels without feature tokens (geometry-only), or both (features+geometry). The identity variant of Figure 2b is trained separately with the same budget and receives the geometry stream with $g(\lambda) = 1$, which gives the identity operator and therefore no cross-row propagation. It tests whether the geometry stream, rather than the propagation along K, explains the improvement.

Operator replacements and reduced operators. At inference, and with every trained parameter fixed, we replace the learned operator by zero, by a matched-random matrix with the same Frobenius norm, or by a same-spectrum misaligned operator $PGP^\top$ that attaches the same relations to other rows, as reported in Table 6. The reduced operators replace the learned spectral response $g(K)$ by the affine combination $aI + bK$, by its off-diagonal part, by its diagonal, or by a row-shuffled version, as reported in Tables 7 and 8.

Classical predictors. Support-adaptive kernel ridge regression (KRR) on K selects its kernel for each task by leave-one-out validation on the support rows, and kNN on K predicts each query from its most similar support rows. Both use the same Gram matrix K as the geometry models and no trained weights, as reported in Table 1 and Appendix G.

Pretrained comparisons. We compare K with the internal row-interaction maps of frozen TabPFN and TabICL through linear CKA, and we add a row-geometry bias to their row attention, as described in Appendices B.6 and F.

Final-run metadata give token/exact-geometry parameter counts of $832{,}394/835{,}823$ (retrieval), $898{,}442/901{,}871$ (prototype), $1{,}367{,}434/1{,}370{,}863$ (perceiver), $1{,}395{,}338/1{,}398{,}767$ (cell attention), $334{,}731/338{,}160$ (kernel), and $433{,}930/437{,}359$ (MotherNet). Thus the geometry stream adds $3{,}429$ trainable parameters to each corresponding token backbone. The operator replacements at inference keep all of these parameters fixed, and the separately trained identity variant tests whether adding the geometry stream alone explains the improvement.

B.3 Exact Gram Matrix Implementation

For the exact Gram matrix, the runs reported in the paper use support-only means and variances, the stabilized/clipped standardizer in Equation (2), stabilized row normalization, and $K = \bar{U}\bar{U}^\top$. We set the clipping threshold in Equation (2) to $c = 4$, chosen by a grid search over values between 1 and 10. This value leaves every standardized value within four standard deviations of the support mean unchanged and limits the influence of more extreme values on the row norms, and hence on K. The operator signal is the support-label one-hot matrix with zeros on query rows. Degree-q operators use Hadamard powers $K^{\circ q}$, and spectral actions are computed by symmetric eigendecomposition of those PSD matrices followed by scalar functions of clamped nonnegative eigenvalues. The trained runs use orders $q \in \{1, 2\}$ and a bank of three functions of the eigenvalues λ, clamped at zero: the identity function $g(\lambda) = 1$, which yields the identity operator and thus no cross-row propagation, the heat function $g(\lambda) = e^{-\lambda}$, and the fractional heat function $g(\lambda) = e^{-\lambda^{1/2}}$, both at diffusion time one. The functions are not rescaled. The coefficients α_f are the softmax outputs of a two-layer network with 32 hidden units and a GELU activation, applied to a six-dimensional permutation-invariant summary of the support rows that includes $\log |S|$, $\log d$, the entropy of the support labels, and a log stable rank of the rows. The same scalar coefficients are used for both orders and all classes, and the order-specific maps W_q and β_q follow them. No signed-degree matrix or graph Laplacian is used in the trained path.

The experiment configurations record the exact-geometry path, and the training code that produced the reported runs contains this exact builder. Every run stores its checkpoints, its training curve, and its final accuracy on each suite. Unit tests cover the operator path, including a check that the dense spectral operator matches its eigendecomposition form to within 10^{-4} and that a zero scale gives a zero output. The natural-table, internal-CKA, and frozen-model evaluations form K with separate evaluation code, which reproduces each unmodified frozen model bit-exactly at $\gamma = 0$, as described in Appendix B.6.

B.4 Statistical units and model selection

Unless stated otherwise, the controlled experiments use five training seeds per backbone and hierarchical-bootstrap intervals. The input-condition comparison reports five compatible backbone means, each over five training seeds. The operator replacements at inference use six backbones $\times$ five seeds. Its 95% intervals use 50,000 hierarchical-bootstrap replicates, resampling backbones first and seeds within each sampled backbone. The reduced operators use one seed-0 exact-geometry checkpoint per backbone; affine coefficients are chosen on IID validation separately for each backbone. We therefore report the six-backbone mean without seed-level confidence intervals for that panel, and Table 8 reports the per-backbone values.

Natural-table tasks, including the classical-predictor evaluations, use 128 support rows and 128 query rows, and the internal-CKA measurements use 200 support rows and 100 query rows with three seeds. Hyperparameters are selected without labels of the feature-space OOD suites. The P0–P9 analysis resamples transform families before rows. Frozen-model paired intervals first average all recorded shifts/seeds within each table and then bootstrap the 25 tables with 50,000 replicates.

Table 4 summarizes which models, runs, tasks, and uncertainty estimates each experiment uses.

B.5 Natural-table protocol

The manifest contains 25 public OpenML classification tables satisfying a frozen inclusion rule: at least 8 encoded dimensions, at least 300 rows, and between 2 and 20 classes. The frozen natural-table protocol stores two models trained on synthetic data (retrieval and MotherNet) per admitted table. We cluster those records by the underlying OpenML table before bootstrapping.

The shift families are R0 identity, R1 per-column affine, R2 permutation, R3 ill-conditioned rescaling, R4 invertible non-orthogonal mixing, R5 nuisance features, R6 redundant expansion, R7 target-relevant deletion, and R8 missingness/quantization. Natural-table predictor comparisons use accuracy; frozen pretrained comparisons use AUC and are not pooled with them.

Table 4: **Each experiment and its protocol.** Models and runs, tasks, and the reported uncertainty for every figure and table of the experiments. Controlled tasks are the synthetic binary classification tasks of Appendix B.1, and SD and SE denote the standard deviation and the standard error.

Experiment	Where	Models and runs	Tasks	Uncertainty
Input conditions	Figure 2a	Five backbones (no axial cell attention), five seeds each	Controlled, feature-space OOD	SD across backbones
Matched budget	Figure 2b, Tables 9 and 11	Six backbones, five seeds each	Controlled, feature-space OOD and IID	SD across backbones
Operator replacement	Figure 3(a), Table 6	Six backbones, five seeds each	Controlled, feature-space OOD	95% hierarchical bootstrap
Reduced operators	Figure 3(b), Tables 7 and 8	Six backbones, one seed-0 checkpoint each	Controlled, feature-space OOD	SD across backbones
Geometry sources	Table 10	Six backbones, five seeds each	Controlled, feature-space OOD	Means
Classical predictors	Table 1	Seed-0 retrieval token and geometry models, MotherNet geometry model, KRR on K	Single-family tasks, $d = 16$, 192 support and 192 query rows	SE over five repetitions
Correlational diagnostic	Figure 5, Table 12	Five backbones	P0–P9 families, base $d = 16$	Bootstrap over families
Scalable construction	Figure 6	Exact and projected Gram matrices	Controlled, feature-space OOD	Point estimates
Natural tables	Figures 8 and 9, Tables 17 and 18	KRR and kNN on K; retrieval and MotherNet models trained on synthetic tasks	25 OpenML tables, 128 support and 128 query rows	Dataset-clustered bootstrap
Internal row structure	Figure 4, Appendix F	Frozen TabPFN v2 and TabICL, three seeds	25 OpenML tables, 200 support and 100 query rows	Shuffled-row baseline
Added bias, natural tables	Table 14, Figure 7	Frozen TabPFN v2 and TabICL; γ selected on held-out support rows	25 OpenML tables	Paired dataset bootstrap
Added bias, synthetic structures	Tables 15 and 16	Frozen TabPFN v2, v2.5, and v3; γ selected on a held-out validation split	10 control tasks; product and oblique parity, six tasks each, $d = 8$	Tasks improved

B.6 Geometry Biases in Frozen TFMs

For TabPFN v2, we add the bias to the pre-softmax attention logits Λ of every row-attention layer, so that the attention weights become $\mathrm{softmax}(\Lambda + \gamma\Psi)$, where Ψ is the Gram matrix K normalized to unit root mean square, restricted to the support rows as keys and shared across heads. The same bias is used with TabPFN v2.5 and v3 on the synthetic structures in Appendix F. For TabICL, whose ICL blocks attend from each row to the support rows, Ψ is instead the cosine Gram matrix of the model's own query and key row representations inside these blocks, normalized in the same way, so that for TabICL the aligned bias is its own row geometry and not the Gram matrix K. We use the public TabICL classifier checkpoint released in February 2026 (Qu et al., 2025). On natural tables, the matched-random control is a symmetric Gaussian matrix $\frac{1}{2}(Z + Z^\top)$ scaled to the Frobenius norm of the Gram matrix, and the shuffled control permutes the rows and columns of the bias jointly, $\Psi \mapsto P\Psi P^\top$, which keeps its spectrum and misaligns its rows. The scale γ is selected from $\{0.25, 0.5, 1, 2\}$ by AUC on a held-out split of the support rows, namely the last $\max(16, \lfloor |S|/4 \rfloor)$ of them, and the selected scale is then applied with the bias of the full task. Both implementations reproduce the unmodified model bit-exactly at $\gamma = 0$. In each architecture, this bias is the closest

analogue of the geometry stream in Equation (6), and it measures whether adding row geometry after training changes the frozen model's predictions.

C Operator Replacements and Training Controls

This section supports (Q1) and (Q2). It reports the per-backbone input conditions of Section 4.2, the complete results of the operator replacements in Section 4.3, together with the matched-budget training control and the IID and per-suite accuracy of Section 4.2, and the single-query evaluation of Section 4.3.

Input conditions. Removing direct access to the raw input features does not reduce accuracy on any backbone. As reported in Table 5, the geometry-only models reach between 63.1% and 63.7% on all five backbones, on par with features+geometry, whereas the features-only models remain between 50.1% and 56.0%.

Operator replacement at inference. Correct row correspondence is required in every replacement we evaluate. Table 6 reports the multi-seed replacements over six backbones and five seeds with their 95% hierarchical-bootstrap intervals. The intact operator reaches 62.82%, whereas the zero, matched-random, and same-spectrum misaligned operators remain close to chance level, and the intervals of the intact and replaced operators are well separated.

Reduced operators. First-order propagation retains nearly all of the effect on the same trained checkpoints. Table 7 reports the six-backbone means for the reduced operators. The affine combination $aI + bK$ and its off-diagonal part remain within 1.2% of the learned spectral response, whereas the diagonal and row-shuffled operators, which remove the support–query propagation or its row correspondence, remain close to chance level. Table 8 shows that this pattern holds on every backbone: $aI + bK$ remains within 2.0% of the trained operator, the off-diagonal part retains most of the effect, and the diagonal and row-shuffled operators remain between 48.7% and 52.6%.

Matched-budget training control. The improvement over the token model holds on every backbone, and we attribute it to cross-row propagation. Under the identical learning-rate and weight-decay grid and at least five seeds, the exact-geometry model exceeds the tuned token model on each of the six controlled backbones by 7.3% to 13.3%, with means of 62.8% and 52.9%, as reported in Table 9. The separately trained identity variant, which receives the geometry stream without cross-row propagation, matches the token model on every backbone and averages 52.7%, which shows that the geometry stream alone does not explain the improvement. Importantly, propagation is required for every source of row geometry, as Table 10 shows. Without propagation, the exact, principal component analysis (PCA)-based, and projected Gram matrices all remain at 52.7%, whereas with propagation, the projected Gram matrix reaches 60.8% and the exact Gram matrix K reaches 62.8%, in line with Propositions 1 and 2.

IID and per-suite accuracy. The improvement appears on IID tasks and in each of the three OOD suites, as reported in Table 11. It is largest on the rotation suite ($+14.9\%$) and on the irrelevant-feature suite ($+10.6$), and it is also present on the feature-count suite ($+4.3$). Within the matched budget, the token models learn the linear family, reaching 77.1% with the retrieval backbone in Table 1, but their average accuracy on these suites remains close to chance level. Under the evaluation protocol of Table 6, the intact operator reaches 62.65% on IID tasks, and replacing it by zero reduces this accuracy to 51.06%, which matches the collapse on the OOD tasks.

Single-query evaluation. The effect does not depend on the other unlabeled queries. Across six backbones and two seeds, processing all queries jointly gives 62.0%, and rebuilding the task independently for each query gives 62.6%.

Table 5: **The Gram matrix K replaces direct access to the raw input features on every backbone.** Feature-space OOD accuracy (%) ↑ under the four input conditions of Figure 2a, averaged over five seeds per backbone. Columns shaded in blue use the Gram matrix K, and columns shaded in beige are baselines without K. The highest accuracy for each backbone is shown in **bold** and the second highest is underlined.

	Without K		With K (Ours)	
Backbone	Label-Only	Features-Only	Geometry-Only	Features+Geometry
Retrieval	50.0	56.0	**63.5**	63.3
Prototype	50.0	50.1	**63.5**	63.4
Perceiver	50.0	52.8	**63.6**	63.4
MotherNet	50.0	52.7	**63.7**	63.4
Kernel Regression	50.0	51.0	**63.1**	60.4
Mean	50.0	52.5	**63.5**	62.8

Table 6: **Correct row correspondence is required.** Feature-space OOD accuracy (%) ↑ after replacing the trained row operator at inference, over six backbones and five seeds, with 95% hierarchical-bootstrap intervals. The row shaded in blue uses the trained operator on K, and rows shaded in rose are controls that break K.

Row Operator	Accuracy (%) ↑	95% Interval
Zero	51.02	[50.23, 51.80]
Matched Random	50.31	[50.16, 50.46]
Same-Spectrum Misaligned	50.75	[50.20, 51.20]
Intact (Ours)	**62.82**	[61.44, 63.39]

Table 7: **First-order propagation retains nearly all of the effect.** Mean feature-space OOD accuracy (%) ↑ of reduced operators on one seed-0 checkpoint per backbone, over six backbones. The coefficients of $aI + bK$ are selected on IID validation tasks separately for each backbone. Rows shaded in blue propagate labels along the correctly attached relations of K, and rows shaded in rose are controls that break this propagation.

Row Operator	Accuracy (%) ↑
Diagonal Operator	51.23
Row-Shuffled Operator	50.45
Off-Diagonal Part of $aI + bK$	62.02
Affine Combination $aI + bK$	62.43
Learned Spectral Response $g(K)$ (Ours)	**63.13**

Table 8: **The reduced operators behave consistently across backbones.** Feature-space OOD accuracy (%) ↑ of each reduced operator on one seed-0 checkpoint per backbone. Columns shaded in blue propagate labels along the correctly attached relations of K, and columns shaded in rose are controls that break this propagation.

	Without aligned propagation		Aligned first-order		Trained (Ours)
Backbone	Diagonal	Row-Shuffled	Off-Diagonal	$aI + bK$	$g(K)$
Cell Attention	51.2	51.0	62.8	62.8	63.0
Kernel Regression	48.7	49.2	60.0	61.9	62.6
MotherNet	52.1	50.8	63.3	62.8	63.3
Perceiver	52.5	50.5	63.8	62.8	63.2
Prototype	50.2	50.3	60.3	62.9	63.4
Retrieval	52.6	50.9	62.0	61.3	63.3
Mean	51.2	50.5	62.0	62.4	63.1

Table 9: **The exact-geometry model improves over the token model on every backbone.** Feature-space OOD accuracy (%) ↑ under a matched training budget, averaged over five seeds per backbone. The identity variant receives the geometry stream without cross-row propagation. Columns shaded in beige are baselines without cross-row propagation, and the column shaded in blue uses the Gram matrix K.

Backbone	Token Model	Identity Variant	Exact-Geometry Model (Ours)	Improvement
Cell Attention	54.7	54.3	63.1	+8.4
Kernel Regression	51.0	51.1	60.4	+9.4
MotherNet	52.7	52.7	63.4	+10.7
Perceiver	52.8	52.8	63.4	+10.6
Prototype	50.1	50.1	63.4	+13.3
Retrieval	56.0	55.2	63.3	+7.3
Mean	52.9	52.7	**62.8**	+9.9

Table 10: **Propagation is required for every source of row geometry, and the exact Gram matrix K is the most accurate.** Mean feature-space OOD accuracy (%) ↑ over six backbones and five seeds. The PCA-based Gram matrix projects the canonicalized rows to at most 128 principal components, and the projected Gram matrix uses a Gaussian random projection to width $h = 128$; both are cosine-normalized after the projection. Cells shaded in blue propagate labels with a Gram matrix, and cells shaded in beige are variants without propagation or without row geometry.

Geometry Source	Without Propagation	With Propagation
Token Model (No Geometry)	52.9	
PCA-Based Gram Matrix	52.7	–
Projected Gram Matrix	52.7	60.8
Exact Gram Matrix K (Ours)	52.7	**62.8**

Table 11: **The improvement appears on IID tasks and in every OOD suite.** Accuracy (%) ↑ at the final evaluation, averaged over six backbones and five seeds. The OOD mean is the feature-space OOD accuracy used throughout the paper. The row shaded in blue uses the Gram matrix K, and the row shaded in beige is the baseline without K.

		Feature-space OOD suites			
Model	IID	Feature count	Rotation	Irrelevant	OOD mean
Token Model	53.5	51.6	54.3	52.8	52.9
Exact-Geometry Model (Ours)	**64.9**	**55.9**	**69.2**	**63.4**	**62.8**

D GEOMETRY ALIGNMENT AND SIGNAL RETENTION

This section supports (Q2) with a correlational analysis that complements the operator replacements in Section 4.3. Specifically, we ask which of two quantities recorded by the P0–P9 protocol, geometry alignment and signal retention, better accounts for the advantage of the geometry models across transformation families.

Setup. The P0–P9 protocol applies ten transformation families to the raw input features. Families P0 to P4 are invertible transformations of the raw input features: permutation (P0), orthogonal transformation (P1), expansion (P2), affine change (P3), and a general invertible transformation (P4). Families P5 to P9 add features or lose information: irrelevant features (P5), correlated nuisance features (P6), replication (P7), rank-reducing projection (P8), and destruction of the geometry (P9). For each family, we record two quantities. The *signal retention* $\rho \in [0, 1]$ is the fraction of the teacher signal that survives the transformation. For the linear teacher with weight vector $w \in \mathbb{R}^d$ and a linear transformation T of the raw input features, it equals $\rho = \|M_T w\|_2^2 / \|w\|_2^2$, where $M_T = T^+ T$ is the orthogonal projector onto the row space of T and T^+ is its Moore–Penrose pseudoinverse, so that $\rho = 1$ for every invertible transformation. The *geometry alignment* is the cosine similarity between the centered Gram matrices of the original and transformed rows, which equals their linear CKA,

$$\frac{\langle \Pi K \Pi,\ \Pi K' \Pi \rangle_F}{\|\Pi K \Pi\|_F \, \|\Pi K' \Pi\|_F}, \qquad \Pi = I - \tfrac{1}{N} \mathbf{1}\mathbf{1}^\top,$$

where K and K' are the Gram matrices of the canonicalized rows before and after the transformation. The advantage is the accuracy of the geometry model minus that of the token model on the same task, with base dimension $d = 16$ and five backbones: retrieval attention, the prototype aggregator, the perceiver, the MotherNet hypernetwork, and kernel regression. The protocol is fixed before aggregation and uses the transformation family as the clustering unit, and the intervals use $10{,}000$ hierarchical-bootstrap replicates.

Results. As shown in Figure 5, geometry alignment is the more stable correlate of the advantage. For the linear teacher, the Pearson correlation with the advantage is 0.366 for geometry alignment (95% interval $[0.021, 0.548]$) and 0.462 for signal retention (95% interval $[-0.033, 0.588]$). The interval for geometry alignment excludes zero, whereas the interval for signal retention does not, and a sparse-quadratic teacher gives the same qualitative ordering with a stronger coupling to geometry. Furthermore, the advantage of the geometry model over the token model remains large under every family, between 15.9% and 28.6%, as reported in Table 12, and it is smallest under rank-reducing projection and destruction of the geometry, which reduce both signal retention and geometry alignment. Because this analysis is correlational, it complements, rather than replaces, the operator replacements in Section 4.3.

Table 12: **The advantage of the geometry model persists under every transformation family.** Signal retention, geometry alignment, and advantage of the geometry model over the token model (%) ↑ for each P0–P9 family with the linear teacher and base dimension $d = 16$, averaged over five backbones.

	Family	Signal retention	Geometry alignment	Advantage
INVERTIBLE	P0 permutation	1.00	1.00	28.5
	P1 orthogonal	1.00	0.99	28.3
	P2 expansion	1.00	0.99	28.6
	P3 affine	1.00	1.00	27.8
	P4 invertible	1.00	0.84	28.3
ADDITIVE OR LOSSY	P5 irrelevant features	1.00	0.64	27.0
	P6 correlated nuisance	1.00	0.92	28.6
	P7 replication	1.00	0.92	28.6
	P8 rank-reducing projection	0.59	0.70	15.9
	P9 destruction	0.81	0.66	20.3

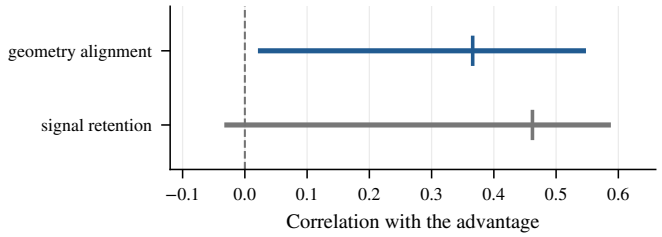


Figure 5: **Geometry alignment is the more stable correlate of the advantage.** Correlation between the advantage of the geometry models and two candidate explanations across transformation families (P0–P9 protocol). Horizontal segments are dependence-aware 95% intervals with the transformation family as the clustering unit, ticks mark the correlation estimates, and the dashed vertical line marks zero.

E SCALABLE CONSTRUCTION OF THE GRAM MATRIX

This section complements the computational complexity analysis in Section 3.3. The experiments in Sections 4.2 and 4.3 use the exact Gram matrix K, so that no approximation affects their results. Here, we describe how the first-order operators can be applied without forming K, and how accurately K can be approximated when d is large.

Cost relative to the backbones. All experiments in Sections 4.2 and 4.3 use the exact Gram matrix K. Forming it costs $\mathcal{O}(N^2 d)$ time and $\mathcal{O}(N^2)$ memory, which is of the same order in N as a single layer of attention between the rows of a task with hidden width D, which costs $\mathcal{O}(N^2 D)$ time. Row attention of this kind is used by TabPFN (Hollmann et al., 2025) and TabICL (Qu et al., 2025), and the backbones in our experiments that compare rows with one another, namely retrieval attention, axial cell attention, non-parametric kernel regression, and the MotherNet hypernetwork, have the same quadratic dependence on N. As we show next, the first-order propagation does not require forming K, and the first-order operators that retain the effect in Section 4.3 preserve the complexity class of every backbone we consider, including the class-prototype aggregator and the latent-bottleneck perceiver, whose cost is linear in N. The full Gram matrix and the eigendecomposition used by the spectral bank in Equation (6), which costs $\mathcal{O}(N^3)$ time per task, are required for the analysis.

First-order propagation without forming K. Because $K = \bar{U}\bar{U}^\top$, the first-order propagation factorizes as $KY = \bar{U}(\bar{U}^\top Y)$, which costs $\mathcal{O}(NdC)$ time and $\mathcal{O}(Nd)$ memory. The same holds for the affine combination, $(aI + bK)Y = aY + b\,\bar{U}(\bar{U}^\top Y)$, and for its off-diagonal part, because the diagonal of K is computed in $\mathcal{O}(Nd)$ time. The query signal of Proposition 2 is obtained in the same way as $K_{QS}Y_S = \bar{U}_Q(\bar{U}_S^\top Y_S)$, where $\bar{U}_S$ and $\bar{U}_Q$ contain the normalized support and query

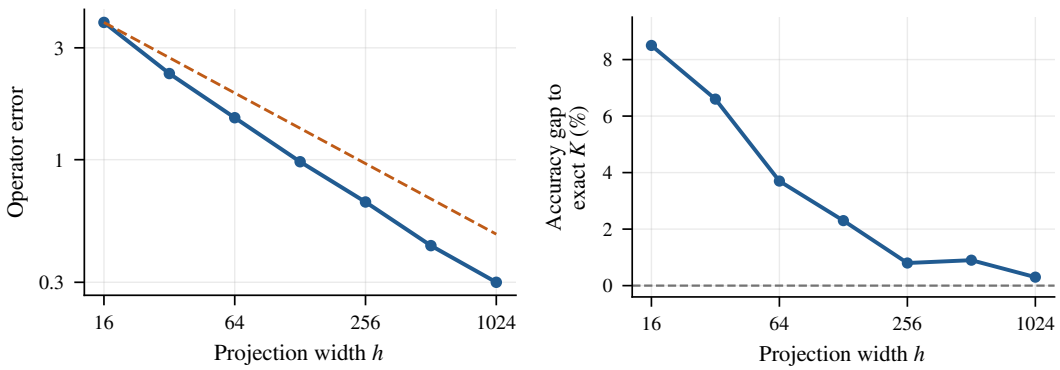
Table 13: **Building the exact Gram matrix K takes milliseconds up to $N = 8{,}192$ rows.** Time (ms) $\downarrow$ and peak GPU memory (GB) $\downarrow$ to build the Gram matrix for N rows and d raw input features on one NVIDIA B200 in float32, as the median over five repetitions after two warm-up runs, excluding host–device transfer. The projected and PCA variants use width $h = 256$. Columns shaded in blue are our constructions of the Gram matrix, and columns shaded in beige are the PCA baseline.

		PCA		Projected		Exact K (Ours)	
N	d	ms	GB	ms	GB	ms	GB
512	128	5.86	0.01	0.07	0.01	0.07	0.01
	1,024	21.44	0.03	0.06	0.01	0.07	0.01
	16,384	27.36	0.23	0.13	0.05	0.35	0.08
2,048	128	3.60	0.03	0.08	0.03	0.08	0.03
	1,024	63.01	0.06	0.08	0.03	0.23	0.04
	16,384	221	0.87	0.24	0.16	2.56	0.30
8,192	128	4.38	0.29	0.31	0.29	0.42	0.29
	1,024	68.98	0.39	0.30	0.31	2.24	0.34
	16,384	5,736	3.80	0.71	0.82	33.85	1.35

rows. Therefore, the first-order operators that retain the effect in Section 4.3 are linear in N, and they preserve the complexity class of every backbone we consider.

Approximating K for large d. Forming the exact Gram matrix costs $\mathcal{O}(N^2 d)$ time. The normalized-row Gaussian sketch in Proposition 3 is unbiased for the Gram matrix K and satisfies a standard effective-rank concentration bound, and its factor $\widetilde{U} \in \mathbb{R}^{N \times h}$ can be used in the factorized propagation above. We also evaluate a project-before-normalize variant, which projects the canonicalized rows to width h and cosine-normalizes only after the projection, at a cost of $\mathcal{O}(Ndh + N^2 h)$ to form its Gram. Because this variant differs from the sketch in Proposition 3, we evaluate it empirically.

Results. A moderate projection width recovers both the operator and the downstream accuracy of the exact Gram matrix K, as shown in Figure 6. We measure the operator error as the relative spectral-norm error $\|\widehat{K} - K\|_2 / \|K\|_2$, where $\widehat{K}$ is the Gram matrix of the projected rows. This error can exceed one when the projection width h is much smaller than the effective rank $\tilde{r}(K)$ of Proposition 3. Across projection widths $h = 16, 32, 64, 128, 256, 512, 1024$, the operator error decreases from 3.85 to 0.30. These operator errors are measured on synthetic Gaussian rows with $N = 384$ and $d = 64$, averaged over 20 draws, for which the exact operator has an effective rank $\tilde{r}(K) \approx 33.6$. The gap in feature-space OOD accuracy to the exact Gram matrix K, which reaches 63.3%, contracts from 8.5% at $h = 16$ to 0.3% at $h = 1024$. For projection width $h = 256$, the measured construction time of the projected variant falls below that of the exact implementation for $d \geq 1024$ at $N = 512$, and for $d \geq 512$ at $N \geq 2048$. Table 13 reports the full measurements: at $N = 8{,}192$ and $d = 1{,}024$, building the exact K takes 2.2 ms and 0.34 GB, whereas the PCA variant takes 69 ms.



(a) Operator error.

(b) Accuracy gap to the exact Gram matrix K.

Figure 6: **A moderate projection width recovers the exact Gram matrix K.** (a) The operator error of the project-before-normalize variant (solid) decreases across the full width sweep, shown with an $h^{-1/2}$ reference (dashed). (b) The gap in feature-space OOD accuracy (%) to the exact Gram matrix K contracts toward zero, which the dashed line marks.

F Row Geometry Inside Frozen TFMs

This section supports (Q4) and complements Section 4.5 with the complete results of the measurements on frozen TabPFN and TabICL: the similarity between their internal row structure and the Gram matrix K, their response to a bias derived from K on natural tables, and the response of TabPFN to the same bias on synthetic structures where the frozen model is weak.

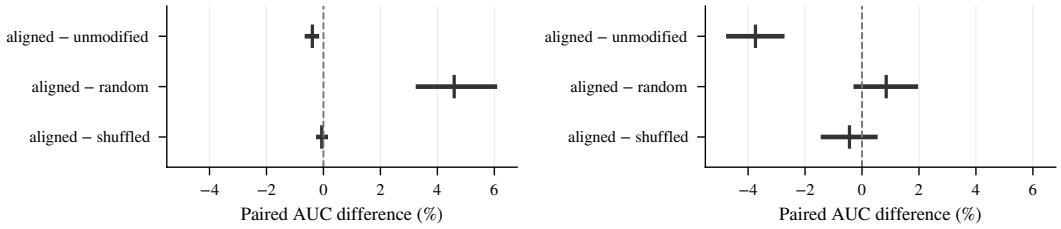
Internal row structure. Both models build internal row structure that closely matches the Gram matrix K, at their most similar layer and on average across layers. At each attention layer, the internal row-interaction map is the cosine similarity between the query vector of one row and the key vector of another, averaged over heads, computed over 100 query rows and 200 support rows; for TabICL, the layers are the 12 blocks of its ICL transformer. We compare each map with K restricted to the same rows and columns through linear CKA on double-centered matrices, $\mathrm{CKA}(A, B) = \langle \Pi_n A \Pi_m, \Pi_n B \Pi_m \rangle_F / (\|\Pi_n A \Pi_m\|_F \|\Pi_n B \Pi_m\|_F)$ for $A, B \in \mathbb{R}^{n \times m}$, with $\Pi_n = I_n - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$, and the maximal CKA is the maximum over layers for each table, averaged over tables and seeds. Across the 25 tables and three seeds, the layer-averaged and maximal linear CKA with the Gram matrix K are 0.624 and 0.903 for TabPFN, and 0.676 and 0.831 for TabICL, compared with 0.035 when the rows of K are shuffled. Figure 4 in the main text shows the most similar internal row-interaction maps on one table.

Class structure. Because the rows of a task share their labels, both K and the internal row-interaction maps of a TFM partly reflect the class structure. We separate the two on BREAST_W, the table of Figure 4, for which the Gram matrix and the most similar internal row-interaction maps of both models are stored over 120 class-sorted support rows. Linear CKA with the label kernel $Y_S Y_S^\top$ is 0.89 for K, 0.87 for TabPFN, and 0.90 for TabICL, so that class structure is a large part of all three matrices. Writing $L = Y_S Y_S^\top$ for the label kernel over the support rows, we remove it from each centered matrix, $A_\perp = \Pi_n A \Pi_n - \frac{\langle \Pi_n A \Pi_n, \Pi_n L \Pi_n \rangle_F}{\|\Pi_n L \Pi_n\|_F^2} \Pi_n L \Pi_n$ with $n = 120$, and define the partial CKA as $\mathrm{CKA}(A_\perp, B_\perp)$. The partial CKA between K and the internal row-interaction map remains 0.79 for TabPFN and 0.82 for TabICL. In contrast, shuffling the rows of K within each class, which keeps the class structure and destroys the row geometry, reduces the partial CKA to 0.03 on average over 200 shuffles (maximum 0.19). Therefore, on this table, the internal row structure of both TFMs matches the row geometry of K beyond what the labels explain. The maps of the other 24 tables were not stored, so this control covers one table.

Adding row geometry to frozen TFMs. TabPFN distinguishes an aligned row geometry from a random one, and the response to an added bias depends on the architecture, as reported in Table 14 and Figure 7. For TabPFN, the aligned bias remains close to the unmodified model and is 4.59 AUC points above the matched-random bias, with an interval that excludes zero and a higher AUC on all 25 tables. For TabICL, whose aligned bias is the row geometry of its own internal representations, the

Table 14: **TabPFN distinguishes an aligned row geometry from a random one.** AUC (%) ↑ of the unmodified frozen models and with three added row-geometry biases, and paired differences with dataset-bootstrap 95% intervals, computed after averaging the recorded shifts and seeds within each table. Columns shaded in beige, rose, and blue report the unmodified model, the controls that break the aligned bias, and the aligned row-geometry bias, formed from K for TabPFN and from its own row representations for TabICL, respectively.

	AUC (%) ↑				Paired difference (95% interval)	
Model	Unmodified	Random	Shuffled	Aligned (Ours)	Aligned − Unmodified	Aligned − Random
TabPFN	78.8	73.8	78.5	78.4	$-0.39\ [-0.66, -0.15]$	$+4.59\ [3.24, 6.10]$
TabICL	82.7	78.1	79.4	79.0	$-3.74\ [-4.77, -2.72]$	$+0.85\ [-0.30, 1.97]$



(a) TabPFN. (b) TabICL.

Figure 7: **The response of frozen TFMs to an added row-geometry bias depends on the architecture.** Horizontal segments are 95% dataset-bootstrap intervals for the paired AUC differences (%) between the aligned bias and the unmodified model, the matched-random bias, and the shuffled bias; ticks mark the paired means, and the dashed vertical line marks zero.

added bias changes the predictions more noticeably, and the aligned bias lies below the unmodified model. The protocol of both interventions is described in Appendix B.6.

Synthetic structures where frozen TabPFN is weak. Adding row geometry improves frozen TabPFN on synthetic tasks, and most on structures where the frozen model is weak. All tasks use $d = 8$ Gaussian raw input features. The control set applies ten variants to a linear sign teacher $y = \mathbb{1}[w^\top x > 0]$, where $\mathbb{1}[\cdot]$ is the indicator function: the identity, an orthogonal rotation, appended nuisance features, invertible non-orthogonal mixing with condition numbers 4, 16, and 64, rank-reducing projections to one half and one quarter of the dimensions, and second- and third-order product teachers. The product structure labels each row by the sign of a product of random projections, $y = \mathbb{1}[\prod_{j=1}^{k} w_j^\top x > 0]$ for k projections, and the oblique-parity structure labels it by the parity of the number of positive projections, $y = \mathbb{1}\big[\,|\{j \le 7 : v_j^\top x > 0\}| \text{ is odd}\big]$, where w_j and v_j are random unit directions. The matched-random bias is the Gram matrix of random Gaussian embeddings, which is independent of the data. Every bias, including the row Gram bias, is normalized to unit root mean square before it is scaled by γ, so that all biases have matched norms, and γ is selected by accuracy on a labeled validation split that is disjoint from both the support rows and the query rows, from $\{0, 0.5, 1, 2, 4\}$ for the control set and from a grid extending to $\gamma = 8$ for the product and oblique-parity structures.

On the control set, a row Gram bias improves AUC on all 10 tasks for each of the three TabPFN versions, and it exceeds the matched-random bias on all 10 tasks, as reported in Table 15. On product and oblique-parity structures, where the frozen model is weaker, adding the exact Gram matrix K yields larger improvements, as reported in Table 16, and other constructions of the Gram matrix, such as PCA-based, Nyström, and randomized-sketch approximations, give improvements of a similar size.

Table 15: **A row Gram bias improves frozen TabPFN on every control task, beyond a matched-random bias.** AUC (%) ↑ of the frozen models and changes in AUC (%) on the 10 synthetic control tasks. Ranges are over three row Gram matrices, computed from the raw input features, from their PCA projection, and from a learned row encoder. The columns shaded in beige, rose, and blue report the frozen model, the change from the matched-random bias, and the improvement from the row Gram bias, respectively, and the last column counts the tasks that the row Gram bias improves.

		Change in AUC from			
TabPFN	Frozen AUC	Matched Random	Row Gram (Ours)	Row Gram − Random	Tasks
v2	94.9	+0.5	+2.8 to +3.3	+2.3 to +2.9	10/10
v2.5	93.8	+2.0	+3.8 to +3.9	+1.8 to +2.0	10/10
v3	95.7	−0.2	+1.8 to +2.0	+2.0 to +2.2	10/10

Table 16: **The improvement from the Gram matrix K is largest where frozen TabPFN is weakest.** AUC (%) ↑ of the frozen model and improvement in AUC (%) from adding the exact Gram matrix K, over six tasks per structure. The oblique-parity structure uses seven projections. The column shaded in beige reports the frozen model, and the column shaded in blue reports the improvement from K.

Structure	TabPFN Version	Frozen AUC	Improvement from K (Ours)	Tasks Improved
Product	v2.5	51.4	+14.7	6/6
	v3	54.6	+10.9	5/6
Oblique parity	v2.5	90.0	+5.6	6/6
	v3	91.8	+3.9	6/6

G Natural Tables and the Information Kept by the Gram Matrix

This section supports (Q3) and complements Section 4.4. We first list the admitted OpenML tables and report the complete transfer results, and we then report the results across shift families and four analyses of what the Gram matrix K preserves, which support the summary in Section 4.4.

Admitted tables. Table 17 lists the 25 public OpenML classification tables that satisfy the fixed inclusion rule of Appendix B.5. They span 522 to 45,312 rows, 8 to 100 encoded features, and 2 to 7 classes.

Transfer to natural tables. The Gram matrix K remains predictive across all 25 tables when it is paired with a predictor that adapts to each task. KRR on the Gram matrix K reaches 80.0% accuracy and kNN on K reaches 78.8%, with 95% dataset-bootstrap intervals of $[75.1, 84.8]$ and $[73.6, 83.9]$, whereas the features+geometry and token models trained on synthetic data reach 51.9% and 39.0%, as shown in Figure 8. KRR is more accurate than the features+geometry model on 25 of the 25 tables, with a mean paired difference of +28.1% (95% dataset-bootstrap interval $[17.3, 39.6]$), and kNN is more accurate on 21 of the 25 tables, with a mean paired difference of +26.9% ($[15.7, 38.9]$).

Shift families. Predictions based on the Gram matrix K follow the information that the rows carry, and not the way in which the raw input features are expressed, as reported in Table 18. Per-column affine changes, permutations, ill-conditioned rescaling, and redundant copies of features re-express the same information, and KRR on K keeps its accuracy between 81.9% and 82.2% under all four, even though redundant copies change K by 0.36. In contrast, nuisance features, missingness, and target-relevant deletion add unrelated features or remove information, change K by 0.53 to 0.76, and lower the accuracy of KRR to between 76.1% and 79.4%. Frozen TabPFN v2 shows the same pattern, which suggests that a pretrained TFM is sensitive to the same kinds of shifts as the predictors on K, as illustrated in Figure 9.

Fidelity of support-conditioned canonicalization. Support-conditioned canonicalization removes per-column affine changes in practice, as the invariances in Appendix A predict. On a construction check with four tables, per-column affine changes alter the stabilized Gram matrix K by only 0.015 in relative norm and leave the accuracy of the geometry model nearly unchanged. In contrast, a rotation of the raw input features before per-column standardization, which lies outside these invariances

Table 17: **The 25 OpenML tables differ in size, feature dimension, and number of classes.** N is the number of rows, d the encoded dimensionality, and C the number of classes.

Dataset	OpenML id	N	d	C	Dataset	OpenML id	N	d	C
authorship	458	841	70	4	jm1	1053	10885	21	2
bank_marketing	1461	45211	51	2	kc1	1067	2109	21	2
breast_w	15	699	9	2	kc2	1063	522	21	2
car	40975	1728	21	4	ozone	1487	2534	72	2
churn	40701	5000	33	2	pc1	1068	1109	21	2
climate	40994	540	18	2	pc3	1050	1563	37	2
cmc	23	1473	24	3	qsar_biodeg	1494	1055	41	2
credit_g	31	1000	61	2	segment	40984	2310	18	7
diabetes	37	768	8	2	spambase	44	4601	57	2
eeg_eye	1471	14980	14	2	steel_plates	40982	1941	27	7
electricity	151	45312	14	2	vehicle	54	846	18	4
hill_valley	1479	1212	100	2	wine_quality_white	40498	4898	11	7
ilpd	1480	583	11	2					

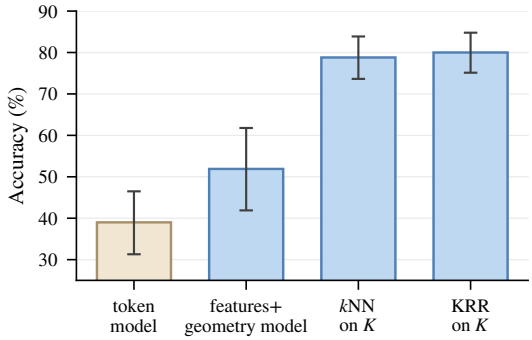


Figure 8: **The Gram matrix K remains predictive across natural tables, whereas the weights of models trained on synthetic tasks transfer to a lesser extent.** Dataset-clustered accuracy (%) on 25 OpenML tables with 95% dataset-bootstrap intervals; the token and features+geometry models are trained on synthetic tasks. Bars shaded in blue use the Gram matrix K, and the bar shaded in beige is the baseline without K.

because it does not commute with per-column standardization, changes the Gram matrix by 0.896 and changes the accuracy accordingly.

Feature deletion. The Gram matrix K tracks the information that remains after target-relevant features are deleted. In this case, K describes the reduced observation, and accuracy follows the information that remains, which is the empirical counterpart of the statement on the information kept by the Gram matrix in Appendix A. We measure this effect with the feature-shift protocol of TabFSBench (Cheng et al., 2025) on four of its tables. The features are ranked by their absolute Pearson correlation with the target, and the most or least relevant features are replaced by their support mean for numeric features or their mode for categorical features, so that the number of raw input features is unchanged. Replacing the most relevant features lowers the accuracy of the geometry model by 11.8% when the replacement is applied to both the support and query rows, and by 23.7 points when it is applied to the query rows only, whereas replacing the least relevant features lowers it by only 0.4 and 4.8 points.

Non-overlapping schemas. When support and query rows observe different sets of columns, restricting the comparison to the shared columns yields the most accurate geometry, because every compared feature then has a common meaning. Filling the union of columns instead introduces correspondences between schemas that the observed data do not support. For the geometry model, restricting the comparison to the shared columns is 12.1%, 19.5%, and 18.4% more accurate than filling the union of columns when the fraction of shared columns is 0.68, 0.49, and 0.30, respectively.

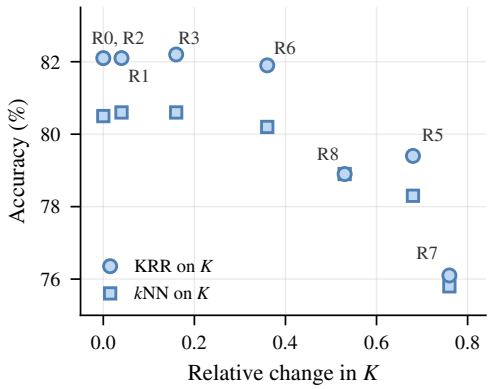


Figure 9: **Predictions based on the Gram matrix K follow the information that the rows carry, and not the way in which the raw input features are expressed.** Accuracy (%) of KRR and kNN on K for each shift family against the relative change in K with respect to R0. Re-expressing the raw input features by per-column affine changes, permutations, rescaling, or redundant copies (R1 to R3 and R6) leaves the accuracy of both predictors unchanged, whereas missingness, nuisance features, and target-relevant deletion (R8, R5, R7) lower it. Markers shaded in blue use the Gram matrix K.

Table 18: **Predictions based on the Gram matrix K follow the information in the rows, and not the way in which the raw input features are expressed.** Relative change in K, measured as $\|K - K_0\|_F / \|K_0\|_F$ with respect to the Gram matrix K_0 of R0, the fraction of the five nearest neighbors of each row that are shared with R0, averaged over rows, and accuracy (%) ↑ of KRR and kNN on K across the shift families R0–R3 and R5–R8 on the 25 OpenML tables. The AUC (%) ↑ of frozen TabPFN v2 is shown for the families evaluated with it. Columns shaded in blue report predictors on K, and the column shaded in beige reports frozen TabPFN v2 without K.

Shift Family	Change in K	Neighbor Overlap	TabPFN v2 AUC	kNN on K	KRR on K
R0 identity	0.00	1.00	83.8	80.5	82.1
R1 per-column affine	0.04	0.95	83.9	80.6	82.1
R2 permutation	0.00	0.98	–	80.5	82.1
R3 ill-conditioned rescaling	0.16	0.83	83.8	80.6	82.2
R5 nuisance features	0.68	0.28	80.9	78.3	79.4
R6 redundant expansion	0.36	0.69	–	80.2	81.9
R7 target-relevant deletion	0.76	0.43	68.5	75.8	76.1
R8 missingness	0.53	0.33	79.7	78.9	78.9

Instrument transfer. Paired calibration data complement the row geometry when rows come from different measurement instruments. On corn and pharmaceutical spectra, the shared-projection bridge matches plain ridge regression when only unpaired observations are available. On corn, the bridge reaches a mean root mean squared error of prediction (RMSEP) of 0.68 without any paired standards, which is comparable to Piecewise Direct Standardization (PDS) (Wang et al., 1991; Bouveresse & Massart, 1996) with four paired standards (0.66), while Direct Standardization with four standards reaches 0.57. With 32 paired standards, PDS reaches 0.31, which quantifies how much paired cross-domain information adds to the unpaired rows.

H Extended Related Work

How TFMs handle changing columns. TFMs process tables whose columns differ in number, type, and meaning, and they do so with learned encoders that are shared across tables. TabPFN encodes each row as one token after zero-padding and rescaling its features to a fixed width, and it averages predictions over ensembles of feature and class rotations and power transforms (Hollmann et al., 2023). MotherNet adapts this architecture to generate the weights of a child network (Mueller et al., 2025), and TabDPT likewise maps each row to a fixed width, projecting wider tables and padding

narrower ones (Ma et al., 2025). TabPFN v2 instead gives every cell its own representation and alternates attention across the features of a row and across the rows of a feature, adding random feature identifiers so that columns with similar distributions remain distinguishable (Hollmann et al., 2025). TabICL embeds each column with a set transformer that captures its distribution, applies attention across the features of each row, and compresses every row into a fixed-size embedding on which it performs ICL (Qu et al., 2025). Cross-table models use column names or table-specific featurizers (Wang & Sun, 2022; Zhu et al., 2023). In all of these models, the representation of a new table is learned, and the relations between rows are formed inside the attention layers together with the processing of the features. The Gram matrix K offers a complementary route: it removes the column axis before any learned component, it has size $N \times N$ for any number of raw input features, and its invariance to the offset and order of the columns holds exactly by construction, and to their scale up to a small stabilizer.

Our work builds on five lines of research: tabular foundation models, cross-table and set-based learning, kernel and nearest-neighbor views of in-context learning, mechanistic studies of TFMs, and graph-based methods that propagate labels over sample affinities. The main difference between these works and ours lies in where the row relations appear. Prior methods use them within a particular predictor, whereas we form them before the predictor, as the Gram matrix K, and study them directly.

Tabular foundation models and in-context prediction. Prior-fitted and in-context tabular models replace task-specific optimization by a forward pass conditioned on labeled support rows. TabPFN established this paradigm for small-data classification through synthetic prior fitting (Hollmann et al., 2023), with later versions extending its accuracy and operating range (Hollmann et al., 2025). TabICL scales the same support-query interface to much larger tables (Qu et al., 2025), while TabDPT studies scaling on real-data pretraining (Ma et al., 2025). MotherNet (Mueller et al., 2025) and HyperFast (Bonet et al., 2024) instead generate or parameterize task-specific predictors from the observed table. In all of these models, the same model infers the input–label relation from the support rows and decides which support rows inform each query. We study the same in-context prediction problem. Differently from these works, we make the row relations available as a separate input that can be removed, changed, or provided to another predictor.

Cross-table transfer, retrieval, and set processing. A separate line of work studies how tabular models handle heterogeneous schemas and data sources. TransTab (Wang & Sun, 2022) and XTab (Zhu et al., 2023) learn representations intended to transfer across tables with different columns, and recent evaluations examine how far current tabular foundation models generalize beyond the distributions on which they were developed (Purucker et al., 2026). Retrieval-based tabular models such as TabR explicitly select training examples that are relevant to a query (Gorishniy et al., 2024), while Deep Sets (Zaheer et al., 2017) and Set Transformers (Lee et al., 2019) provide general permutation-invariant machinery for reasoning over collections. Prototypical networks similarly summarize a labeled support set by class representatives (Snell et al., 2017). These approaches use relations between samples through a particular encoder, retrieval rule, or predictor. In contrast, we compute one Gram matrix K before the predictor, so that feature access, row correspondence, cross-row propagation, and the predictor itself can be changed one at a time.

Kernel and nearest-neighbor views of in-context learning. Kernel perspectives connect in-context prediction to non-parametric regression and weighted combinations of context labels (Han et al., 2025). In tabular learning, SoftKNN-ICL trains an in-context model to behave as a soft nearest-neighbor classifier (Koshil et al., 2025). KernelICL makes the final prediction rule explicit by replacing a TFM readout with Gaussian, dot-product, or k-nearest-neighbor (kNN) kernels, which yields an inspectable weighted average of context labels (Miftachov et al., 2026). The main difference in our work is where the kernel enters. Specifically, the Gram matrix K is computed before, and independently of, the prediction rule. We keep it when the raw input features are removed, change it while the trained model stays fixed, and provide it to other predictors. This identifies which row relations suffice for prediction before any particular kernel or neural readout is chosen.

Mechanistic accounts of tabular foundation models. Recent mechanistic work identifies qualitatively different similarity-based readouts inside pretrained TFMs and confirms them with targeted interventions (Biloš et al., 2026). A large-scale study of layerwise dynamics further shows substantial redundancy across the layers of six TFMs (Balef et al., 2026a). These analyses start from a trained architecture and ask which algorithm it has learned. Our work is complementary and proceeds in the opposite direction: we first define the Gram matrix K from the input alone, identify what the models

require from it by removing the raw input features and replacing the operator while the model is fixed, and only then ask whether pretrained TFMs contain internal row structure similar to the Gram matrix K. We use linear centered kernel alignment (CKA) (Kornblith et al., 2019) only for this last comparison, while our conclusions about what the models use are drawn from replacing the operator in fixed models.

Affinity propagation and Gram operators. Graph-based semi-supervised learning propagates labels over an affinity graph (Zhu et al., 2003; Zhou et al., 2003), while manifold (Belkin et al., 2006) and diffusion methods (Coifman & Lafon, 2006) use neighborhood or diffusion operators to encode sample geometry. Dot-product kernels (Kar & Karnick, 2012; Wacker et al., 2024), graph kernels (Smola & Kondor, 2003), and random-projection results (Achlioptas, 2003; Woodruff, 2014; Halko et al., 2011) provide the mathematical tools behind our Gram construction and its approximations. Previous works use these tools to build predictors, while in this work, we use them to analyze one. Accordingly, the Gram matrix K is the object under study, and the controlled model is flexible by design, so that replacing parts of its operator after training identifies which part is needed.

Support–query relations in few-shot learning and TFMs. Metric-based few-shot learning predicts a query from its similarity to labeled examples, through matching networks (Vinyals et al., 2016) and learned relation scores (Sung et al., 2018). Graph-based methods represent a few-shot task as a graph over support and query examples and pass messages along it (Garcia Satorras & Bruna Estrach, 2018; Kim et al., 2019), and transductive methods propagate support labels over support–query graphs (Liu et al., 2019; Ziko et al., 2020; Ma et al., 2020; Zhu & Koniusz, 2023; Huang et al., 2024). In tabular ICL, retrieving the nearest support rows for each query improves TFMs (Thomas et al., 2024; Xu et al., 2025; Guta, 2026), fine-tuning aligns the query–key similarities of TabPFN with the target (Rubachev et al., 2025), and recent studies relate TFM generalization to exemplar-based retrieval and aggregation (Balef et al., 2026b; Shaheen et al., 2026). Our work shares the view that support–query relations are central, and it differs in where these relations live: we form them as a label-free, row-indexed state before the predictor, so that we can remove the raw input features, replace the relations while the trained model stays fixed, and provide them to other predictors.