pureGAM: Learning an Inherently Pure Additive Model

Xingzhi Sun*
xingzhi.sun@yale.edu
Yale University
New Haven, Connecticut, USA

Ziyu Wang*
ziyuwang@pku.edu.cn
Peking University
Beijing, China

Rui Ding†
juding@microsoft.com
Microsoft Research
Beijing, China

Shi Han
shihan@microsoft.com
Microsoft Research
Beijing, China

Dongmei Zhang
dongmeiz@microsoft.com
Microsoft Research
Beijing, China

ABSTRACT

Including pairwise or higher-order interactions among predictors of a Generalized Additive Model (GAM) is gaining increasing attention in the literature. However, existing models face an *identifiability* challenge. In this paper, we propose pureGAM, an inherently pure additive model of both main effects and higher-order interactions. By imposing the *pureness condition* to constrain each component function, pureGAM is proved to be identifiable without compromising accuracy. Furthermore, the pureness condition introduces additional interpretability in terms of simplicity. Practically, pureGAM is a unified model to support both numerical and categorical features with a novel learning procedure to achieve optimal performance. Evaluations show that pureGAM outperforms other GAMs and has very competitive performance even compared with opaque models, and its interpretability remarkably outperforms competitors in terms of pureness. We also share a successful adoption of pureGAM in one real-world application.

CCS CONCEPTS

• **Computing methodologies** → **Supervised learning by regression**.

KEYWORDS

Generalized Additive Model; Pureness; Identifiability; Pure Coding; Interpretable Machine Learning

ACM Reference Format:
Xingzhi Sun, Ziyu Wang, Rui Ding, Shi Han, and Dongmei Zhang. 2022. pureGAM: Learning an Inherently Pure Additive Model. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3534678.3539256

*The work was done when the authors were interns at Microsoft Research Asia.
†Corresponding Author.

KDD '22, August 14–18, 2022, Washington, DC, USA

ACM ISBN 978-1-4503-9385-0/22/08…$15.00
https://doi.org/10.1145/3534678.3539256

Table 1: Intuitive illustration of GAM, GA^{k}M and pureGAM

Model	Formula
GAM	$g(y) = \sum_{i=1}^{p} f_i(x_i)$
GA^{k}M	$g(y) = \sum_{u \in \mathcal{D}_k} f_u(x_u)$
pureGAM	$g(y) = \sum_{u \in \mathcal{D}_k} f_u(x_u)$ s.t. $\forall u \in \mathcal{D}_k, f_u(x_u)$ is pure

(1) $g(y) = g\left(\mathbb{E}[Y|x_1, \cdots, x_p]\right)$.
(2) p: The total number of features.
(3) $\mathcal{D}_k$: The set of all non-empty subsets of $\{1, \cdots, p\}$ with cardinality $\leq k$.
(4) The pureness (hierarchical orthogonality) condition of f_u:
$\int f_u(x_u) h_v(x_v)\, w(x)\, dx = 0, \forall v \subset u, \forall h \in \mathcal{L}^2(\mathbb{R}^v)$.

1 INTRODUCTION

Including pairwise or higher-order interactions among the predictors (or features) of a Generalized Additive Model (GAM) is gaining increasing attention in the literature [5, 18, 21, 33]. For instance, GA^{2}M [18] extends GAM by modeling pairwise interactions, as shown in the 2$^{\text{nd}}$ row of Table 1 by setting $k = 2$. Here g is the link function and each f_u is called a component function or component, which is used to model interactions for $|u| \geq 2$.

However, existing models face an *identifiability* challenge. For additive models, the concept of identifiability boils down to the uniqueness of decomposing a model prediction into the sum of specific component functions. For example, given one model learned by GA^{2}M: $g(y) = f_1(x_1) + f_{12}(x_1, x_2)$, the interaction f_{12} can freely absorb its nested main effect (i.e., the univariate component) f_1 to yield another decomposition with the same prediction: $g(y) = 0 + h_{12}(x_1, x_2)$ where $h_{12}(x_1, x_2) := f_1(x_1) + f_{12}(x_1, x_2)$, thus GA^{2}M is non-identifiable. Identifiability is a vital requirement for interpretability since non-uniqueness permits "contradictory" decompositions for representing the same prediction [16].

To achieve identifiability, a desirable approach is to impose certain constraints among the components. A recent work GAMI-Net [33] extends GA^{2}M by introducing a constraint called "marginal clarity". However, our analysis (Section 3.5.1) shows that GAMI-Net still cannot guarantee unique decomposition in the general setting, and thus is non-identifiable. Consequently, the interpretability of all the exiting models is compromised.

Motivated by this, we propose pureGAM, an inherently *pure* additive model of both main effects and higher-order interactions. The overall modeling of pureGAM is shown in the 3$^{\text{rd}}$ row of Table 1,

where interactions are modeled up to order k ($k < p$), and each component is constrained by the pureness condition (the bottom row in Table 1). Theoretically, pureGAM is proved to be **identifiable**. Furthermore, the pureness condition introduces additional interpretability in terms of **simplicity**. Practically, pureGAM is a **unified** model to support both numerical and categorical features with a novel learning procedure to achieve **optimal** performance. Specifically:

Identifiability. pureGAM adopts the pureness condition to constrain each component function. The pureness condition was originally termed the "hierarchical orthogonality condition" [14] in the domain of functional ANOVA. We prove that under the pureness condition, the optimal solution of pureGAM is unique and does not compromise accuracy (w.r.t. standard $\mathcal{L}^2$ loss). In other words, decomposing a p-variate $\mathcal{L}^2$ function (i.e., $g(E(Y|x))$) into the sum of all k-variate subfunctions is unique (up to a minimum $\mathcal{L}^2$ loss), thus pureGAM is identifiable.

Simplicity. The pureness condition entails that one component f_u must NOT "absorb" any of its nested lower-order component f_v : $\forall v \subset u$. Otherwise, the integral is strictly greater than zero which violates the pureness condition. Thus, we say such a component f_u is "pure" w.r.t. all its nested lower-order components. Considering a nested lower-order component f_v is always simpler to interpret than $f_u : v \subset u$, the pureness condition follows Occam's razor: a lower-order component (i.e., a simpler interpretation) is preferred if it has the same prediction performance. It is known that simplicity is beneficial to interpretability since it increases the likelihood that the user both understands and accepts the model [20].

Unification. To tackle categorical features, most GAMs implementations adopt label encoding or one-hot encoding such as EBM [21], GAMI-Net [33], pyGAM [25], or FXAM [8]. These encodings do not satisfy the pureness condition which further leads to non-identifiability. We propose *pure coding*, a novel algorithm for encoding categorical features and representing categorical component functions to satisfy the pureness condition. As a result, pureGAM can be applied to general datasets including both numerical and categorical features.

Optimality. Regarding pureGAM's training procedure, we exploit a joint learning strategy instead of sequential learning. Sequential learning is adopted in GA^{2}M and GAMI-Net, which learns components sequentially from lower orders to higher orders. Our analysis (Section 3.5.1) shows that sequential learning would lead to a sub-optimal solution but learning all the components jointly (joint learning) ensures optimum. Our efficient training algorithm makes such joint learning affordable. For instance, we propose a novel adaptive kernel method for estimating the probability density $w_u(x_u)$ (i.e., the marginal distribution of w on variables x_u) and for kernel smoothing. The computation for estimating pureness, where w_u is a key part, and estimating component functions f_u can largely be shared. In summary, we make the following contributions[1]:

(1) We propose pureGAM, an inherently pure additive model. To the best of our knowledge, among the GAMs that model interactions, pureGAM is the first to be identifiable, interpretable (in terms of simplicity), unified (to support both categorical and numerical features), and has optimal performance.
(2) We propose a novel training procedure to train pureGAM effectively and efficiently. Evaluations show that pureGAM outperforms other GAMs and has very competitive performance even compared with opaque models, and its interpretability remarkably outperforms competitors in terms of pureness.
(3) We present a successful adoption of pureGAM in one real-world application, which utilizes pureGAM's identifiability and interpretability for data analysis and decision-making.

[1]pureGAM's code is available at https://github.com/microsoft/reliableAI.

2 RELATED WORK

GAMs in general. Generalized Additive Models (GAM) are gaining great attentions in the literature of interpretable machine learning [1, 6, 17, 24], mainly due to its standard for interpretability [3, 30] and its broad adoptions in real-world settings [4, 22, 27]. The original modeling of GAM [13] is shown in 1$^{\text{st}}$ row of Table 1, where we want to predict y with given features $(x_1, \cdots, x_p)$. GAM untangles the overall prediction by summing up contributions from each component.

GAMs with interactions. In addition to GA^{2}M [18] and its fast implementation EBM [21], some more recent work extends GAMs by adopting neural networks instead of traditional nonparametric methods (such as smoothing splines or kernels) to estimate each component function. NODE-GA^{2}M [5] adopts a novel neural network architecture NODE [23] and modifies it to mimic GA^{2}M. GAMI-Net [33] extends GA^{2}M by using MLP (Multi-Layer Perceptron) to estimate each component function, and it further introduces a constraint called "marginal clarity", which attempts to make the model more *identifiable* by imposing orthogonality between any interaction component and its nested main effects. However, our analysis in section 3.5.1 shows that GAMI-Net still cannot guarantee unique decomposition, thus all these existing GAMs have limitations in terms of identifiability.

Pureness condition. The pureness condition adopted in pureGAM is inspired by hierarchical orthogonality condition [14], originally used for weighted functional ANOVA to guarantee the uniqueness of perfectly decomposing a p-variate $\mathcal{L}^2$ function into the sum of all p-variate subfunctions (weighted by data distribution). pureGAM's modeling is different in that k (see 3$^{\text{rd}}$ row in Table 1) is typically set with $k \ll p$ to avoid overfitting. Thus, instead of perfect decomposition, we prove that up to a global minimum $\mathcal{L}^2$ loss, decomposing a p-variate $\mathcal{L}^2$ function into the sum of all k-variate subfunctions is unique (weighted by data distribution).

Another related work is "purification" [16], an efficient algorithm that purifies an existing ML model, such that an interaction component does not contain any of its nested lower-order components. It is a post-hoc method that is applied to an existing ML model. In contrast, pureGAM learns an inherently pure additive model. Learning pure model inherently takes advantage that it can directly access the data distribution w from training data but post-hoc methods need to make assumptions on data distribution. Moreover, "purification" is only applicable to purify tree-based models.

The simplicity entailed by the pureness condition is also a generalization of the reluctance principle [34]: that a main effect (i.e.,

univariate component) should be preferred over a pairwise interaction if both have similar prediction performance.

3 APPROACH

For the sake of simplicity, we focus on the case where the link function $g(y)$ is an identity function, i.e., the regression task. pureGAM follows the standard to support other link functions.

3.1 Modeling

Given a dataset $\mathbb{D}$ consisting of N realizations of random variable Y at design values $(x_1, \cdots, x_p)$ or shortly x, where x is sampled from an underlying probability density $w(x)$, we model pureGAM as:

$$\mathbb{E}(Y|x) = \sum_{u \in \mathcal{D}_k} f_u(x_u) \text{ s.t. } \forall u \in \mathcal{D}_k, f_u(x_u) \text{ is pure.} \tag{1}$$

Throughout this paper, all the functions discussed are in weighted-$\mathcal{L}^2$ space $\mathcal{L}^2_w$, or simply, $\mathcal{L}^2$ space[2], since they are essentially identical. Equation (1) is an additive model, with interactions up to order k, satisfying the pureness condition. $\mathcal{D}_k := \{u | u \subset \{1, \cdots, p\}, 0 < |u| \leq k\}$, is a collection of all non-empty subsets of $\{1, \cdots, p\}$ with cardinality $\leq k$. If not further specified, we omit the bias term ($|u| = 0$) for simplicity.

Definition 1. *We define "$f_u(x_u)$ is pure", if*

$$\forall v \subset u, \forall h_v \in \mathcal{L}^2(\mathbb{R}^v), \int f_u(x_u) h_v(x_v) w(x) dx = 0. \tag{2}$$

In other words, f_u is orthogonal to any subfunction h_v (i.e. $v \subset u$) w.r.t. inner product $\langle f, g \rangle_w := \int f(x) g(x) w(x) dx$.

Operationally, we use an equivalent definition of the pureness condition (or the so-called "integrate-to-zero" constraint [14] which turns out to be more computable.

Definition 2. *We say $f_u(x_u)$ is pure if and only if:*

$$\forall i \in u, \int f_u(x_u) w_u(x_u) dx_i = 0. \tag{3}$$

Here w_u is the marginal distribution of w on variables x_u. The integral is a function on $x_{u \setminus i}$ and it is required to be zero for any value $x_{u \setminus i}$ takes.

Setting these up, we propose pureGAM's constrained optimization as follows:

$$\begin{aligned} &\{f_u(x_u) | u \in \mathcal{D}_k\} \\ &= \text{argmin}_{\{g_u \in \mathcal{L}^2\}_{u \in \mathcal{D}_k}} \int \left(\sum_{u \in \mathcal{D}_k} g_u(x_u) - \mathbb{E}(Y|x) \right)^2 w(x) dx \\ &s.t. \forall u \in \mathcal{D}_k, \forall i \in u, \int f_u(x_u) w_u(x_u) dx_i = 0 \end{aligned} \tag{4}$$

Theorem 3.1. *For arbitrary k, solution of (4) is unique when density w is non-degenerate, or more precisely, w is "grid-closed".*

Theorem 3.1 establishes the identifiability of pureGAM's modeling. The requirement that w is "grid-closed" first appeared at [14], which is a mild assumption. The proof is available in Appendix A.1.

[2] $\mathcal{L}^2_w := \left\{ f : \left(\int f^2(x) w(x) dx \right)^{1/2} < \infty \right\}$, i.e., $\mathcal{L}^2_w$ is the space of functions with finite $\mathcal{L}^2$ norm w.r.t. measure w.

Remark 1. *The pureness condition does not compromise the optimality of objective function in (4) but only ensures uniqueness, i.e., without the pureness condition, the minimum loss remains the same.*

Remark 2. *It is known that the optimal solution of the original GAM (i.e., only modeling main effects) is unique, which can be viewed as a special case of pureGAM: when $k = 1$, the integrate-to-zero constraint reduces to restricting each main effect f_i to have zero mean, i.e., the ordinary mean-centering constraint.*

Next, we illustrate the details of pureGAM's training, which consists of two parts: pure learning on numerical features and pure coding on categorical features. For numerical features, we derive a more computable version of integrate-to-zero constraint; for categorical features, we choose coding beforehand to strictly achieve the pureness condition on categorical features.

3.2 Pure Learning on Numerical Features

Learning on numerical features requires estimating each component function f_u and validating its pureness condition, which essentially needs estimation of density w_u. We exploit kernel methods, which are naturally suitable for smoothing and density estimation.

3.2.1 Adaptive Kernel Method. Given a sample $\{x_{u,1} \sim \tilde{y}_1, \cdots, x_{u,N} \sim \tilde{y}_N\}$, kernel smoothing for representing component $f_u(x_u)$ is defined as follows:

$$f_u(x_u) = \frac{\sum_{s=1}^N K\left(\frac{x_u - x_{u,s}}{h}\right) \tilde{y}_s}{\sum_{s=1}^N K\left(\frac{x_u - x_{u,s}}{h}\right)} = \sum_{s=1}^N W_s(x_u, h) \tilde{y}_s, \tag{5}$$

where K is the kernel and h is the bandwidth. Kernel smoothing can be viewed as a weighted average based on observations, where the weight $W_s(x_u, h) = K\left(\frac{x_u - x_{u,s}}{h}\right) / \sum_{s=1}^N K\left(\frac{x_u - x_{u,s}}{h}\right)$.

In our problem, $\{x_{u,1}, \cdots, x_{u,N}\}$ are available in training data, but $\tilde{\boldsymbol{y}} = \{\tilde{y}_1, \cdots, \tilde{y}_N\}$ and bandwidth h are undetermined, which need to be trained together with other components. Therefore, we set $\{\tilde{y}_1, \cdots, \tilde{y}_N\}$ and h as parameters, which lead to the parametrized form of kernel smoother:

$$f_u(x_u | \boldsymbol{\eta}_u, h_u) = \sum_{s=1}^N W_s(x_u, h_u) \eta_{u,s}, \tag{6}$$

where terms are updated by $\tilde{\boldsymbol{y}} \to \boldsymbol{\eta}_u, h \to h_u$, thus each component is modeled by a specific bandwidth h_u.

3.2.2 Empirical Pureness Condition. To make the integrate-to-zero constraint computable from training data, we adopt the parametrized kernel density estimation method to estimate the integration, setting the bandwidth h_u as a parameter.

Lemma 3.1. *For $i \in u$,*

$$\int f_u(x_u) w_u(x_u) dx_i \sim \frac{1}{N h_{u \setminus i}} \sum_{s=1}^N K_{u \setminus i}\left(\frac{x_{u \setminus i} - x_{u \setminus i, s}}{h_{u \setminus i}}\right) f_u(x_{u,s} | \boldsymbol{\eta}_u, h_u). \tag{7}$$

Here $\sim$ denotes asymptotic approximation. In other words, $\frac{1}{N h_{u \setminus i}} \sum_{s=1}^N K_{u \setminus i}\left(\frac{x_{u \setminus i} - x_{u \setminus i, s}}{h_{u \setminus i}}\right) f_u(x_{u,s} | \boldsymbol{\eta}_u, h_u)$ is an empirical version of $\int f_u(x_u) w_u(x_u) dx_i$.

Lemma 3.1 provides a computable version to validate the pureness condition from training data. See Appendix A.2 for proof.

Definition 3. *(Empirical pureness condition)*

$$\forall u \in \mathcal{D}_k, \forall i \in u,$$
$$\frac{1}{Nh_{u\setminus i}} \sum_{s=1}^{N} K_{u\setminus i}\left(\frac{x_{u\setminus i} - x_{u\setminus i,s}}{h_{u\setminus i}}\right) f_u(x_{u,s}|\boldsymbol{\eta}_u, h_u) = 0 \tag{8}$$

3.3 Pure Coding on Categorical Features

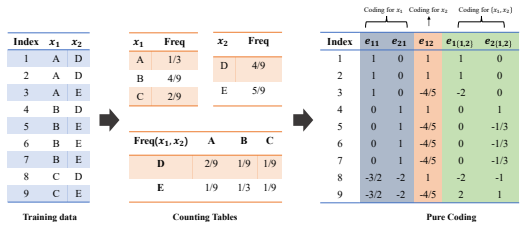


Figure 1: An example of pure coding.

Different from learning on numerical features, we propose an encoding mechanism called pure coding to make the learned components on categorical features strictly satisfy the pureness condition. After pure coding, the learning task on categorical features is reduced to solving a set of linear equations, where accelerating gradient descent could be adopted to improve the training efficiency.

First, we present the discrete version of integrate-to-zero constraint on categorical features as:

$$\forall i \in u, \sum_{x_i \in L_i} f_u(x_u) w_u(x_u) = 0 \tag{9}$$

where L_i is the domain of categorical feature x_i, (i.e., the distinct values that x_i takes). By using empirical frequency $q_u(x_u)$ from training data to approximate $w_u(x_u)$, we have the empirical pureness condition on categorical features as:

Definition 4. *(Discrete empirical pureness condition)*

$$\forall i \in u, \sum_{x_i \in L_i} f_u(x_u) q_u(x_u) = 0 \tag{10}$$

To avoid overwhelming notations, we next illustrate pure coding for main effects and pairwise interactions. Details for $k > 2$ are available in Appendix B.

We first illustrate pure coding for main effects. Supposing x_1 is a categorical feature with domain L_i, we denote $l = |L_i|$. Pure coding transforms each value of L_i to a vector with dimensionality equals to $l-1$ (note that dimensionality for one-hot encoding is l); and the components of the vector are real-values which are calculated based on the frequencies, $q_i(x_i)$ (one-hot encoding only takes 0 or 1), to ensure the pureness condition. The specific coding mechanism is:

Definition 5. *(Pure coding for main effects) Suppose x_i is a categorical feature taking value in $\{1, \cdots, l\}$, denote $q_t := q_{\{i\}}(t), t \in \{1, \cdots, l\}$ as the frequency of $\{x_i = t\}$, and define the pure coding as the $(l-1)$-dimensional vector $\boldsymbol{e}_{\{i\}} = (e_{\{i\},1}, \cdots, e_{\{i\},l-1})^T$, where for $t = 1, 2, \cdots, l-1$,*

$$e_{\{i\},t} := \begin{cases} 1 & x_i = t, \\ -\frac{q_t}{q_l} & x_i = l, \\ 0 & otherwise. \end{cases} \tag{11}$$

In Definition 5, $e_{\{i\},t}$ is indexed by the second subscript t, and the first subscript indicates which component (note: here the component is the variable set, not the vector component) it works on. Figure 1 shows some examples. The training data contains two categorical features x_1 and x_2 with 9 records. To calculate the pure coding for feature x_1, we first calculate the frequencies of its values A, B, and C, which are shown in the top-left of "Frequency Tables". We assign indices 1, 2 and 3 to A, B, and C accordingly. According to Definition 5, the pure coding of x_1 is a $(3-1) = 2$-dimensional vector $(e_{\{1\},1}, e_{\{1\},2})^T$. The vector components are then calculated based on (11). We thus have nine 2-dimensional encoded vectors to represent the component x_1 in training data (the two blue columns in the "Pure Coding" Table). Similarly, we can conduct pure coding for feature x_2 and the encoded vector is shown in the orange column (the dimensionality is 1 since x_2 has only two distinct values).

To illustrate pure coding for interactions, we first define the effective dimensionality:

Definition 6. *(Effective dimensionality)*

$$o_u := \prod_{i \in u} |L_i - 1| \tag{12}$$

o_u is the dimensionality of the encoded (by pure coding) vector for component u.

Definition 7. *(pure coding for pairwise interactions) suppose x_i is a categorical feature taking value in $\{1, \cdots, l\}$, and x_j is a categorical feature taking value in $\{1, \cdots, m\}$, denote $q_{rs} := q_{\{i,j\}}(r, s)$ as the frequency of $\{x_i = r, x_j = s\}$, and define the pure coding as the $(l-1)(m-1)$-dimensional vector $\boldsymbol{e}_{\{i,j\}} = (e_{\{i,j\},rs})^T_{1 \le r \le l-1, 1 \le s \le m-1}$, where for each combination of r and s,*

$$e_{\{i,j\},rs} := \begin{cases} 1 & (x_i, x_j) = (r, s), \\ -\frac{q_{rs}}{q_{rm}} & (x_i, x_j) = (r, m), \\ -\frac{q_{rs}}{q_{ls}} & (x_i, x_j) = (l, s), \\ \frac{q_{rs}}{q_{lm}} & (x_i, x_j) = (l, m), \\ 0, & otherwise, \end{cases} \tag{13}$$

Figure 1 shows an example of pure coding on $\{x_1, x_2\}$. We first calculate the frequencies of (x_1, x_2), which are shown at the bottom of "Frequency Tables". The effective dimensionality is $(3-1)(2-1) = 2$, thus we use a 2-dimensional vector $(e_{\{1,2\},1}, e_{\{1,2\},2})^T$ for encoding (follow a convention that index 1 corresponds to $\{x_1 = A, x_2 = D\}$ and 2 corresponds to $\{x_1 = B, x_2 = D\}$). The vector components are then calculated based on (13), and we have nine 2-dimensional encoded vectors to represent the component $\{x_1, x_2\}$ in training data (the green columns in the "Pure Coding" table).

Similar to Definitions 5 and 7, pure coding for interactions with $k > 2$ can be pre-determined. See Appendix B for details. In summary, for each realization x_u of component u, we obtain the o_u-dimensional vector $\boldsymbol{e}_u := (e_{u,1}, \cdots, e_{u,o_u})^T$ by pure coding. We model each component function of categorical features as

Definition 8. *(Discrete component function)*

$$f_u(x_u|\boldsymbol{\eta}_u) = \boldsymbol{e}_u(x_u)^T \boldsymbol{\eta}_u, \tag{14}$$

where $\boldsymbol{\eta}_u := (\eta_{u,1}, \cdots, \eta_{u,o_u})^T$ is an o_u-dimensional vector of parameters.

Lemma 3.2. *The function (14) satisfies the discrete empirical pureness condition for any parameter $\boldsymbol{\eta}_u \in \mathbb{R}^{o_u}$. Moreover, the space of*

such functions is equal to the entire space of functions on x_u satisfying the discrete empirical pureness condition (10).

Lemma 3.2 guarantees that we can use Definition 8 to model component functions on categorical features and that the pureness condition holds automatically. Take x_1 in Figure 1 as an example component, let's set $\eta_{\{1\}} := (1,0)^T$, thus the component function takes x_1 as input, and the corresponding output is shown in the left column of blue part in "Pure Coding" Table in Figure 1. The discrete empirical pureness condition (10) requires that the values in this column should sum to 0, which is true.

By using pure coding, the learning task on categorical features is reduced to solving linear equations without further constraints, which is much simplified and many accelerating gradient descent algorithms can be exploited to boost training efficiency.

3.4 Training

We have established the theoretical foundation of pureGAM. In this section, we present details on training and a set of optimization techniques to further improve efficiency.

pureGAM's data version of constrained optimization is shown as follows:

$$\operatorname{argmin}_{\boldsymbol{\eta}_u^{\text{num}},\boldsymbol{\eta}_v^{\text{cat}},h_u} \sum_{s=1}^{N}\left(\sum_{u\in\mathcal{D}_k^{\text{num}}} f_u(x_{u,s}|\boldsymbol{\eta}_u^{\text{num}},h_u)+\sum_{v\in\mathcal{D}_k^{\text{cat}}}\boldsymbol{e}_v(x_{v,s})^T\boldsymbol{\eta}_v^{\text{cat}}-y_s\right)^2 \tag{15}$$

$$\text{s.t.}\ \sum_{s=1}^{N} f_u(x_{u,s}|\boldsymbol{\eta}_u^{\text{num}},h_u)K_{u,i,t,s}=0$$
$$\forall u\in\mathcal{D}_k^{\text{num}}\forall i\in u,\forall t\in\{1\cdots,N\} \tag{16}$$

(15) is the empirical squared loss. (16) is the data version of the pureness condition. $x_{u,s}$ denotes the s^{th} instance of x_u in training data. $\mathcal{D}_k^{\text{num}}$ and $\mathcal{D}_k^{\text{cat}}$ denote the components within numerical features and categorical features respectively. To simplify the notation, let $K_{u,i,t,s} := \frac{1}{Nh_{u\setminus i}}K_{u\setminus i}\left(\frac{x_{u\setminus i,t}-x_{u\setminus i,s}}{h_{u\setminus i}}\right)$, so (16) is just the empirical pureness condition (7) applied to all training instances $\{x_t : t\in\{1,\cdots,N\}\}$. Operationally, we use an equivalent constraint $l_{u,i}$:

$$l_{u,i} := \frac{1}{N}\sum_{t=1}^{N}\left(\sum_{s=1}^{N} f_u(x_{u,s}|\eta^{\text{num}},h_u)K_{u,i,t,s}\right)^2 \quad \forall u\in\mathcal{D}_k,\forall i\in u. \tag{17}$$

Adding a regularization term for $l_{u,i}$, we have the final version of loss function for training, where λ denotes the regularization hyperparameter:

$$\mathcal{L}=\sum_{s=1}^{N}\left(\sum_{u\in\mathcal{D}_k^{\text{num}}} f_u(x_{u,s}|\boldsymbol{\eta}_u^{\text{num}},h_u)+\sum_{v\in\mathcal{D}_k^{\text{cat}}}\boldsymbol{e}_v(x_{v,s})^T\boldsymbol{\eta}_v^{\text{cat}}-y_s\right)^2+\lambda\sum_{u\in\mathcal{D}_k^{\text{num}}}\sum_{i\in u} l_{u,i} \tag{18}$$

In addition to the adoption of state-of-the-art training and optimization techniques, below we highlight two key ideas particularly tailored for improving pureGAM's training efficiency.

Parameter sharing. We set each kernel bandwidth h_u as a trainable parameter, which corresponds to a specific component function f_u. Such a strategy leads to better performance than exploiting universal bandwidth h. Furthermore, since the kernel method is naturally suitable for both smoothing (i.e., component function representation) and density estimation (i.e., via kernel density estimation), thus we also use bandwidth h_u for estimating density w_u, as shown in Equations (7) and (17).

Sampling and batching. Parametrizing each component function by an N-dimensional η_u^{num} is expensive especially when N is large. In our problem, the task of kernel smoothing is conducted in a low-dimensional setting since $k \ll p$ (e.g., k is typically set to 2 or 3 in practice). It is known that for the kernel method, sampling is suitable for such a low-dimensional setting to yield almost the same result when N is large [11]. Therefore, we reduce the dimensionality of each η_u^{num} from N to a fixed number S. Moreover, since the training of pureGAM is conducted by SGD (Stochastic Gradient Descent [2]), we set batch size as B, and we let $t\in\{1,\cdots,S\}$ and $s\in\{1,\cdots,B\}$, respectively. As a result, the time complexity of (16) is reduced from $O(N^2)$ to $O(BS)$. Such a strategy is simple but effective, and it is highly efficient according to our experiments.

3.5 Analysis and Discussion

3.5.1 Implications of Pureness. Marginal clarity is an alternative constraint that was introduced in GAMI-Net [33]:

Definition 9. *(Marginal Clarity)*

$$\forall i,j\in\{1,\cdots,p\}\int f_{ij}(x_i,x_j)f_i(x_i)w_{ij}(x_i,x_j)dx_idx_j=0$$

In other words, marginal clarity requires each pairwise interaction f_{ij} to be orthogonal to its nested main effects f_i and f_j. It is easy to see marginal clarity is entailed by the pureness condition but not vice versa: the pureness condition requires f_{ij} to be orthogonal to any univariate functions, but f_i and f_j are not sufficient to represent all the univariate functions.

Here is an example where non-unique decompositions exist and both of them satisfy the marginal clarity constraint. We also show that the decompositions violate the pureness condition. Let probability density ω over (X_1,X_2) be the uniform distribution on the region $\{(x_1,x_2)\in\mathbb{R}^2 : x_1\geq 0, x_2\geq 0, x_1+x_2\leq 1\}$. Let the ground truth function be

$$f(x_1,x_2)=x_1x_2.$$

There exist two representations of $f(x)=f_0+f_1(x_1)+f_2(x_2)+f_{12}(x_1,x_2)$:

$$\begin{cases} f_0=\frac{1}{12}\\ f_1(x_1)=0\\ f_2(x_2)=\frac{1}{10}\left(x_2-\frac{1}{3}\right)\\ f_{12}(x_1,x_2)=x_1x_2-\frac{1}{10}\left(x_2-\frac{1}{3}\right)-\frac{1}{12}\end{cases} \tag{19}$$

$$\begin{cases} f_0=\frac{1}{12}\\ f_1(x_1)=\frac{1}{10}\left(x_1-\frac{1}{3}\right)\\ f_2(x_2)=0\\ f_{12}(x_1,x_2)=x_1x_2-\frac{1}{10}\left(x_2-\frac{1}{3}\right)-\frac{1}{12}\end{cases} \tag{20}$$

Using the fact that

$$\int x_1^a x_2^b\omega(x_1,x_2)dx_1dx_2=\frac{2a!b!}{(a+b+2)!},\forall a,b\in\mathbb{N}, \tag{21}$$

it can be checked that both (19) and (20) satisfy marginal clarity, while neither of them satisfies pureness. Hence the marginal clarity does not imply pureness and allows non-unique decompositions of the same function.

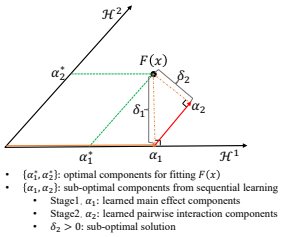


Figure 2: Sequential learning leads to a sub-optimal solution.

3.5.2 Sequential Learning vs. Joint Learning. As illustrated in Section 3.4, we exploit joint learning to train pureGAM. However, sequential learning is adopted more frequently in literature due to its efficiency advantage. Denote $\mathcal{H}^1 := \{f : f = \sum_{u \in \mathcal{D}_1} f_u | f_u \text{ is pure}\}$, $\mathcal{H}^2 := \{f : f = \sum_{u \in \mathcal{D}_2 \setminus \mathcal{D}_1} f_u | f_u \text{ is pure}\}$. Sequential learning first learns the optimal solution in $\mathcal{H}^1$ (standard training of GAM), and then uses the residuals to learn the optimal solution in $\mathcal{H}^2$ (fixing main effects and training for pairwise interactions). We now show that sequential learning would lead to sub-optimal solutions. It is shown that $\mathcal{H}^2$ is not necessarily orthogonal to $\mathcal{H}^1$ w.r.t. inner product $\langle f, g \rangle_w$. For instance, a pure interaction $f_{12} \in \mathcal{H}^2$ could be non-orthogonal with a main effect $f_3 \in \mathcal{H}^1$, i.e., $\langle f_{12}, f_3 \rangle_w \neq 0$, when $w_{123}(x_1, x_2, x_3) \neq w_{12}(x_1, x_2) w_3(x_3)$ (note that the pureness condition only ensures hierarchical orthogonality such that $\langle f_{12}, f_1 \rangle_w = 0$ and $\langle f_{12}, f_2 \rangle_w = 0$). Now we use Figure 2 for an intuitive illustration: in the abstract functional space, according to the projection theorem [19], when sequential learning first fits $F(x)$ on $\mathcal{H}^1$, we come to the components α_1 with minimum residual δ_1 (the head of the orange arrow). Sequential learning then fits the residual δ_1 with interactions on $\mathcal{H}^2$, and we get components α_2 with non-zero residual δ_2 (the head of red arrow). Note that in this case, we have an optimal solution (a perfect fit) with components α_1^* and α_2^* for main effects and interactions respectively. In contrast, joint learning keeps iteration between $\mathcal{H}^1$ and $\mathcal{H}^2$, which eventually converges to the optimal solution. It is worth noting that when input features are independent, sequential learning could find the optimal solution, since now $\mathcal{H}^1 \perp \mathcal{H}^2$, it is easy to verify that $\delta_2 = 0$ thus no further iterations are needed.

Both GA^{2}M and GAMI-Net adopt sequential learning. However, GAMI-Net further conducts a "fine-tuning" stage by jointly re-training all the components, which shows significant performance improvement. As we have discussed, the "fine-tuning" stage essentially adopts the joint learning strategy.

4 EVALUATION

We comprehensively evaluate pureGAM's performance[3] on synthetic datasets generated with various scales, distributions, and ground-truth functions. Furthermore, we evaluate pureGAM's performance on 23 representative real-world datasets.

Comparison Algorithms. We choose 3 representative algorithms for comparison: GAMI-Net [33], EBM [21], and XGBoost [7]. GAMI-Net is the most related competitor. It is the SOTA among GAMs by modeling pairwise interactions and introducing a constraint called "marginal clarity", which attempts to make the model more identifiable. EBM is a highly optimized version of GA^{2}M, which we use as a baseline to represent the GAMs with modeling on interactions. To concisely convey key points from the evaluation, we omit the comparison with NODE-GA^{2}M and other implementation of standard GAMs. NODE-GA^{2}M has a very close performance to EBM but EBM's training efficiency is better [5]. The implementations of the standard GAM, such as pyGAM [25] in python, mgcv [32] in R, show considerably lower performance than the GAMs which model interactions, on both synthetic and real-world datasets. We choose XGBoost as a reference for accuracy. XGBoost is an opaque ML model which is adopted by many winning teams of ML competitions.

Design and Metric. For each dataset, we conduct a 5-fold cross-validation and use average RMSE to measure accuracy. For pureGAM and GAMI-Net, we further measure "how pure the model is" using $\log(\text{pure}) := \frac{\sum_{u \in \mathcal{D}_k^{\text{num}}} \sum_{i \in u} \log(l_{u,i})}{\sum_{u \in \mathcal{D}_k^{\text{num}}} \sum_{i \in u} 1}$, which is the average of the logarithm of $l_{u,i}$ (17) over all the components. According to Lemma 3.1, $l_{u,i}$ is an adequate metric to measure the pureness of a specific component and we choose the logarithm of $l_{u,i}$ since the pureness for GAMI-Net and pureGAM often differ largely. Moreover, for synthetic datasets, we use a more accurate version to calculate $l_{u,i}$, using the ground-truth distribution density instead of kernel density estimated by pureGAM. At last, we also measure the average training time of pureGAM and GAMI-Net (EBM is not included since EBM is trained using a different setting).

4.1 Evaluation on Synthetic Datasets.

Setting. The synthetic datasets are generated following the configurations shown in Table 2. Given these configurations, each dataset is generated using a randomly created ground-truth function with a randomly created density. The ground-truth function is restricted to having interactions up to order $k = 2$. This setting is beneficial to GAMI-Net and EBM since they can only model pairwise interactions. At last, we generate a synthetic dataset such that the ground-truth function is with order $k = 3$ interaction, to show pureGAM's advantage in modeling higher-order interactions. Details of the setting are available in Appendix C.

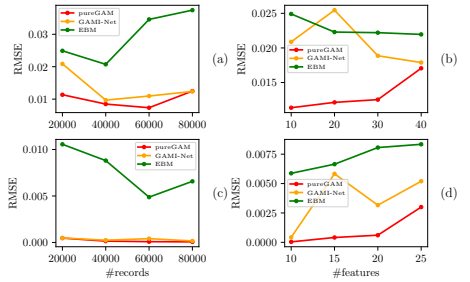
Table 2: Configurations for generating synthetic datasets

Configuration	Value Range
#records	{20000, 40000, 60000, 80000}
#features (Numerical)	{10, 20, 30, 40}
#features (Categorical)	{10, 15, 20, 25}

Performance. The performance (in terms of RMSE) results are shown in Figure 3 with varied data scale or data types. It is shown that pureGAM's and GAMI-Net's performances are consistently better than EBM's, except for one dataset where GAMI-Net has a worse performance. This is mainly because EBM adopts sequential learning which leads to a sub-optimal solution, whereas both pureGAM and GAMI-Net adopt joint learning (GAMI-Net's joint learning is its "fine-tuning" stage). See Section 3.5.2 for detailed illustration. Furthermore, pureGAM's performance is consistently

[3] In the following sections, we use "accuracy" and "performance" interchangeably.

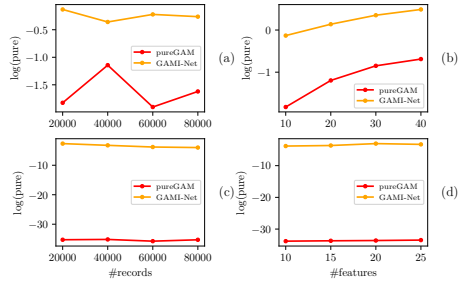
better than GAMI-Net's. This is mainly due to GAMI-Net adopting an additional heuristic called "heredity constraint": not all the pairwise interactions are learned, but only a small subset which is derived from top-n important main effects (EBM also uses this heuristic).



(a)(c): Varying #records with #features=10; (b)(d): Varying #features with #records=20,000;
(a)(b): features are all numerical; (c)(d): features are all categorical.

Figure 3: Performance of pureGAM/GAMI-Net/EBM on synthetic datasets.

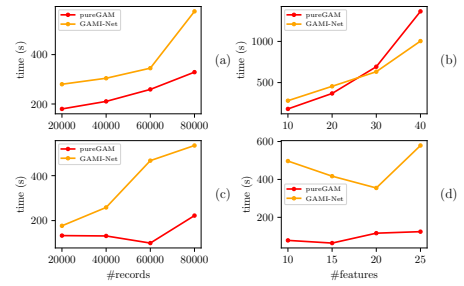
Pureness. Figure 4 shows the pureness results of pureGAM and GAMI-Net on synthetic data. Since pureness is measured by log(pure), the smaller the log(pure) the better the model satisfies the pureness condition. The results show that pureGAM are orders of magnitude better than GAMI-Net in terms of pureness. The contrast between pureGAM and GAMI-Net on numerical datasets shows the limitation of using marginal clarity to approximate the pureness condition. Regarding categorical datasets, the contrast is significantly larger than that on numerical datasets, because for categorical features, pureGAM adopts pure coding to strictly satisfy the pureness condition, but GAMI-Net adopts one-hot encoding and uses a regularization term.



(a)(c): Varying #records with #features=10; (b)(d): Varying #features with #records=20,000;
(a)(b): features are all numerical; (c)(d): features are all categorical.

Figure 4: Pureness results on synthetic data. log(**pure**) **measures pureness in logarithm scale, the smaller the better.**

Training Efficiency. We also compare the training efficiency of pureGAM against GAMI-Net. GAMI-Net has an advantage on training efficiency, at the cost of accuracy, since it benefits from the "heredity" heuristic. The result shows that in most (14/16) cases, pureGAM has a smaller training time mainly due to the optimization techniques (i.e., parameter sharing and sampling) adopted in pureGAM. For example, the training time of pureGAM grows slowly when the number of records increases on numerical datasets. The two cases where GAMI-Net has lower training time reflect the fact that, when the number of features is large, the "heredity" heuristic starts to eliminate more interaction components to maintain a stable computational cost.



(a)(c): Varying #records with #features=10; (b)(d): Varying #features with #records=20,000;
(a)(b): features are all numerical; (c)(d): features are all categorical.

Figure 5: Results of training time on synthetic data.

pureGAM's capability to model higher-order interactions. pureGAM not only supports modeling pairwise interactions but also supports modeling higher-order interactions. Table 3 shows the results of pureGAM under two settings (hyperparameter k set to 2 and 3 respectively) trained on a dataset where the ground-truth function has interactions up to order $k = 3$. As expected, pureGAM ($k = 3$) achieves the best performance which indicates pureGAM's capability to model higher-order interactions.

Table 3: Performance results on synthetic datasets with higher-order interactions

Models	pureGAM($k = 3$)	pureGAM($k = 2$)	GAMI-Net	EBM	XGBoost
RMSE	**0.0128 ± 0.0019**	0.0415 ± 0.0012	0.0417 ± 0.0005	0.0447 ± 0.0018	0.0227 ± 0.0015

4.2 Evaluation on Real Datasets.

Data. We conduct experiments on 23 real-world datasets from the UCI machine learning repository[9], the OpenML platform[29], and the Regression DataSets website[28]. These datasets are collected from diverse domains and are commonly used in literature. We set $k = 2$ for training pureGAM.

Results. Table 5 shows the performance of pureGAM and its competitors[4]. The last two columns show the pureness of pureGAM and GAMI-Net respectively. Regarding accuracy, pureGAM outperforms GAMI-Net on 17/23 datasets and outperforms EBM on 18/23 datasets. Moreover, pureGAM achieves significantly higher accuracy on datasets "air quality" , "electrical grid" , "elevators" and "kinematics" compared with GAMI-Net and EBM. For the datasets where pureGAM does not achieve the highest accuracy, its accuracy is comparable to that of GAMI-Net and EBM. Regarding pureness, it is shown that pureGAM outperforms GAMI-Net on all datasets (23/23).

[4]For better readability, the RMSE values for each dataset have been scaled with a scaling factor of the range of the response values of that dataset, $1/(y_{\max} - y_{\min})$.

5 SURVEY COMPLETION TIME PREDICTION

Post-COVID-19 technologies for education and corporate communication have opened-up great opportunities for online surveys [12]. A survey designer (or designer for short) designs a questionnaire and collects feedback from a set of targeted respondents. The response rate and response quality are primarily important to the success of a survey. Therefore, the Completion Time Prediction (CTP) of a questionnaire becomes an important ML task, since it is often the case that response rate negatively correlates with the completion time, which also indirectly impacts the response quality.

Below we show a real-world adoption of pureGAM (by setting $k = 2$) for CTP task, and how pureGAM's identifiability and interpretability help for 1) identifying top-influential factors and 2) revealing a hidden factor "patience" which is further confirmed highly valuable.

Table 4: Results of performance on CTP dataset

Model	pureGAM	GAMI-Net	XGBoost
RMSE	**0.1614**	0.1645	0.1706

The CTP dataset[5] contains 13080 records with 44 features. Each record represents a specific questionnaire with the response variable to be the median of completion time (since a questionnaire is answered by a set of respondents), and the features are extracted from the content of the questionnaire, such as *#ChoiceQuestions*, *#TextQuestions*, *#RequiredQuestions*, and so on (the meaning of each feature is as intuitive as the feature name). The overall performance[6] of pureGAM is shown in Table 4. pureGAM achieves the highest accuracy compared with GAMI-Net and the opaque model XGBoost. Analysts are satisfied with pureGAM's performance.

Identifying top-influential factors. The learned pureGAM model can naturally be used for presenting component-level importances (a.k.a., global sensitivity analysis [15]). Using pureGAM, the response y can be modeled by $y = \sum_{u\in\mathcal{D}_k} f_u(x_u) + \epsilon$ where f_u's are the component functions learned by pureGAM and $\epsilon \sim N(0, \sigma^2)$. By deduction from the weighted functional ANOVA, we have

$$\begin{aligned}\mathrm{Var}(Y) &= \sum_{u\in\mathcal{D}_k}\left(\mathrm{Var}(f_u) + \mathrm{Cov}\left(f_u, \sum_{v\in\mathcal{D}_k, v\neq u} f_v\right)\right) + \hat{\epsilon}^2 \\ &= \mathrm{Var}(Y)\sum_{u\in\mathcal{D}_k}\mathrm{imp}_u + \hat{\epsilon}^2, \end{aligned} \tag{22}$$

where we denote component importance $\mathrm{imp}_u := \frac{\mathrm{Var}(f_u)+\mathrm{Cov}(f_u, \sum_{v\in\mathcal{D}_k, v\neq u} f_v)}{\mathrm{Var}(Y)}$ to reflect the normalized importance of a specific component u. The closer imp_u to 1 indicates that the component "explains" more fraction of variance (thus it is a higher influential factor) and the closer imp_u to 0 indicates that it is a trivial component with almost no contribution to prediction (thus it is not an influential factor).

Figure 6 shows a snapshot of the component importances composed of 6 out of 44 features. Each block indicates the importance of a specific component, where a diagonal block indicates a main effect and a non-diagonal block indicates a pairwise interaction. Based on this, the top-influential factor analysis can be conducted at two levels: component-level and feature-level (i.e., an influential factor can either be an important component or an important feature). One benefit of pureGAM is that we can directly remove the trivial components and the remaining model is still valid without re-training (because of the pureness condition, which imposes orthogonality constraints among the components, removing one component does not impact the other components).

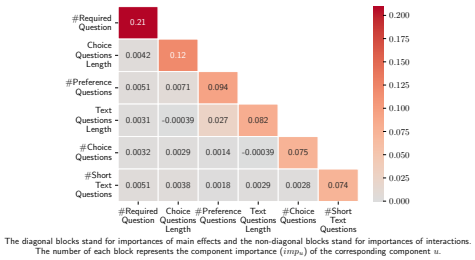


Figure 6: Heatmap of component importances on CTP.

Perhaps for analysts, it is more intuitive to conduct feature-level analysis. It is easy to see that a trivial feature is one with negligible importance on all the components that contain this feature. For example, in Figure 6, we find that feature *LengthSections* is trivial (all the components that contain *LengthSections* have small importances of < 0.01). On the other hand, the top-influential features are more intuitively conveyed by the heatmap. For example, *#RequiredQuestions* is the most important feature.

Following the aforementioned analysis procedure, analysts have eliminated 30+ low-influential features, and the resulting model is much simpler to interpret. Analysts are satisfied with this new model: the importance of top-influential factors (both components and features) are mostly aligned with domain expert knowledge.

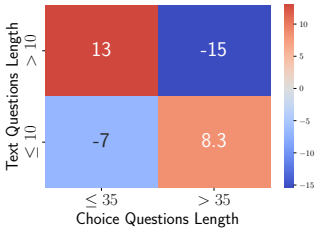


Figure 7: Binarized heatmap of interaction {*#TextQuestions* , *#ChoiceQuestions*}.

Revealing a hidden factor. Analysts further inspect the interaction components that pureGAM has learned. One of the most important interaction is the pair {*#TextQuestions*, *#ChoiceQuestions*}, and a simplified visualization of the learned component function is shown in Figure 7 (the original component function represents a two dimensional smooth surface), where we binarize each feature thus the smooth surface reduces to a heatmap with 2×2 blocks. The interaction function shows that when the numbers of both types of questions are small (i.e., *#TextQuestions* ≤ 10 and *#ChoiceQuestions*

[5]The dataset is extracted following legal compliance, but we omit the details due to confidentiality.

[6]For better readability, the RMSE values have been scaled with a scaling factor of the range of the response values of the dataset, $1/(y_{\max} - y_{\min})$.

$\leq$ 35), a respondent needs 1.7 minutes less to complete (i.e., the contribution from this interaction is −1.7, bottom left block); as the number of either type of questions get larger, a respondent needs an additional 2 (*#ChoiceQuestions* > 35 and *#TextQuestions* $\leq$ 10) or 3.1 (*#ChoiceQuestions* $\leq$ 35 and *#TextQuestions* > 10) minutes to finish respectively. However, a counter-intuitive observation is that when the numbers of both questions are large (the top-right block), the time used to complete a questionnaire is reduced by 3.7 minutes, which is even more than the time-reduction when the numbers of both questions are small.

Considering that pureGAM's model is unique with the best performance, analysts are inclined to trust such an interaction component and they further conclude that it reveals an important hidden factor: a respondent's "patience". As the length of a questionnaire grows larger (e.g., *#ChoiceQuestions* and *#TextQuestions* are two typical question types to reflect the length of a questionnaire), the respondent is becoming impatient, and the consequence is that they do not fill the questionnaire carefully thus the time becomes shorter. Apparently, the impatience of the respondent directly impacts the response quality, which is a serious issue. The insight revealed by pureGAM is further confirmed with domain experts. Therefore, the analysts discuss with engineers and they decide to add a feature to the survey service to "alert" designers: when a designer is creating or editing a questionnaire, and if the length of the questionnaire exceeds a certain threshold (determined based on the pureGAM model), the designer will see a notification that suggests that he should control the length of the survey.

CONCLUSION

We propose pureGAM, an inherently pure additive model of both main effects and higher-order interactions. By imposing the pureness condition to constrain each component function, pureGAM is proved to be identifiable. pureGAM is a unified model to support both numerical and categorical features with a novel learning procedure to achieve optimal performance. Evaluations have shown that pureGAM has very competitive performance even compared with opaque models, and its interpretability remarkably outperforms competitors in terms of pureness.

REFERENCES

[1] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion* 58 (2020), 82–115.

[2] Léon Bottou. 2010. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*. Springer, 177–186.

[3] Andreas Buja, Trevor Hastie, and Robert Tibshirani. 1989. Linear smoothers and additive models. *The Annals of Statistics* (1989), 453–510.

[4] Raffaella Calabrese et al. 2012. Estimating bank loans loss given default by generalized additive models. *UCD Geary Institute Discussion Paper Series, WP2012/24* (2012).

[5] Chun-Hao Chang, Rich Caruana, and Anna Goldenberg. 2021. NODE-GAM: Neural Generalized Additive Model for Interpretable Deep Learning. *arXiv preprint arXiv:2106.01613* (2021).

[6] Chun-Hao Chang, Sarah Tan, Ben Lengerich, Anna Goldenberg, and Rich Caruana. 2021. How interpretable and trustworthy are gams?. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 95–105.

[7] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. Association for Computing Machinery, New York, NY, USA, 785–794. https://doi.org/10.1145/2939672.2939785

[8] Rui Ding, Tianchi Qiao, Yunan Zhu, Zhitao Zou, Shi Han, and Dongmei Zhang. 2021. A Unified and Fast Interpretable Model for Predictive Analytics. *arXiv preprint arXiv:2111.08255* (2021).

[9] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

[10] Włodzimierz Greblicki, Adam Krzyżak, and Mirosław Pawlak. 1984. Distribution-free pointwise consistency of kernel regression estimate. *The annals of Statistics* (1984), 1570–1575.

[11] László Györfi, Michael Kohler, Adam Krzyżak, and Harro Walk. 2002. *A distribution-free theory of nonparametric regression*. Vol. 1. Springer.

[12] Najmul Hasan and Yukun Bao. 2020. Impact of "e-Learning crack-up" perception on psychological distress among college students during COVID-19 pandemic: A mediating role of "fear of academic year loss". *Children and Youth Services Review* 118 (2020), 105355.

[13] TJ Hastie and RJ Tibshirani. 1990. Generalized Additive Models, vol. 43 CRC Press. (1990).

[14] Giles Hooker. 2007. Generalized functional anova diagnostics for high-dimensional functions of dependent variables. *Journal of Computational and Graphical Statistics* 16, 3 (2007), 709–732.

[15] Bertrand Iooss and Paul Lemaître. 2015. A review on global sensitivity analysis methods. In *Uncertainty management in simulation-optimization of complex systems*. Springer, 101–122.

[16] Benjamin Lengerich, Sarah Tan, Chun-Hao Chang, Giles Hooker, and Rich Caruana. 2020. Purifying interaction effects with the functional anova: An efficient algorithm for recovering identifiable additive models. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2402–2412.

[17] Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. 2021. Explainable ai: A review of machine learning interpretability methods. *Entropy* 23, 1 (2021), 18.

[18] Yin Lou, Rich Caruana, Johannes Gehrke, and Giles Hooker. 2013. Accurate intelligible models with pairwise interactions. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. 623–631.

[19] David G Luenberger. 1997. *Optimization by vector space methods*. John Wiley & Sons.

[20] Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence* 267 (2019), 1–38.

[21] Harsha Nori, Samuel Jenkins, Paul Koch, and Rich Caruana. 2019. Interpretml: A unified framework for machine learning interpretability. *arXiv preprint arXiv:1909.09223* (2019).

[22] Amandine Pierrot and Yannig Goude. 2011. Short-term electricity load forecasting with generalized additive models. *Proceedings of ISAP power* 2011 (2011).

[23] Sergei Popov, Stanislav Morozov, and Artem Babenko. 2019. Neural oblivious decision ensembles for deep learning on tabular data. *arXiv preprint arXiv:1909.06312* (2019).

[24] Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence* 1, 5 (2019), 206–215.

[25] Daniel Servén and Charlie Brummitt. 2018. pyGAM: generalized additive models in python. *Zenodo. doi* 10 (2018).

[26] Charles J Stone. 1994. The use of polynomial splines and their tensor products in multivariate function estimation. *The Annals of Statistics* (1994), 118–171.

[27] Nemanja Tomić and Sanja Božić. 2014. A modified geosite assessment model (M-GAM) and its application on the Lazar Canyon area (Serbia). *International journal of environmental research* 8, 4 (2014), 1041–1052.

[28] Luis Torgo. [n.d.]. Regression DataSets. https://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html

[29] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. 2013. OpenML: Networked Science in Machine Learning. *SIGKDD Explorations* 15, 2 (2013), 49–60. https://doi.org/10.1145/2641190.2641198

[30] Zijie J Wang, Alex Kale, Harsha Nori, Peter Stella, Mark Nunnally, Duen Horng Chau, Mihaela Vorvoreanu, Jennifer Wortman Vaughan, and Rich Caruana. 2021. GAM Changer: Editing Generalized Additive Models with Interactive Visualization. *arXiv preprint arXiv:2112.03245* (2021).

[31] Larry Wasserman. 2006. *All of Nonparametric Statistics (Springer Texts in Statistics)*. Springer-Verlag, Berlin, Heidelberg.

[32] S.N Wood. 2017. *Generalized Additive Models: An Introduction with R* (2 ed.). Chapman and Hall/CRC.

[33] Zebin Yang, Aijun Zhang, and Agus Sudjianto. 2021. GAMI-Net: An explainable neural network based on generalized additive models with structured interactions. *Pattern Recognition* 120 (2021), 108192.

[34] Guo Yu, Jacob Bien, and Ryan Tibshirani. 2019. Reluctant interaction modeling. *arXiv preprint arXiv:1907.08414* (2019).

A PROOFS OF THEOREMS AND LEMMAS

A.1 Proof of Theorem 3.1

as defined in the beginning of this paper, let there be p features, and $\mathcal{D}_d := \{u : u \subset \{1, \cdots, p\}, |u| \leq d, u \neq \emptyset\}$. $|u|$ denotes the cardinality of set u.

To prove Theorem 3.1, we need to first prove the following lemmas. Below we assume each function $f_u \in \mathcal{L}^2$, and the inner product is defined by $\langle f, g \rangle_w := \int f(x)g(x)w(x)dx$.

Lemma A.1. *If f_u is pure and g_u is pure, then $\alpha f_u + \beta g_u$ is pure $\forall \alpha, \beta \in \mathbb{R}$.*

The proof is straightforward based on the definition of pureness (2).

Lemma A.2. *$\forall u, v \subset \{1, \cdots, p\}$, $\forall f_u, f_v$, if f_u, f_v are both pure, then $f_u + f_v$ is pure.*

Proof. This follows from the definition: f_u is pure *iff* $\int f_u(x)w(x)dx_i = 0 \forall i \in u$. □

Denote $\mathcal{F}_d := \{f | f = \sum_{u \in \mathcal{D}_d} f_u \text{ s.t. } f_u \text{ is pure}\}$, $\mathcal{G}_d := \{f : f = \sum_{u \in \mathcal{D}_d} f_u\}$.

Definition 10. ***(pure representation)**. A function h is said to have a pure representation f if $f \in \mathcal{F}_d$ and $h = f$.*

Lemma A.3. *$\mathcal{F}_d = \mathcal{G}_d$.*

Proof. $\forall v \subset \{1, \cdots, p\}$, define $\mathcal{G}^v = \{f : f = \sum_{u \subseteq v} f_u\}$, and define $\mathcal{F}^v = \{f \in \mathcal{G}^v : f \text{ is pure}\}$. We first show that $\mathcal{G}^v = \mathcal{F}^v$ by induction over the cardinality of v. $\forall |v| = 1, \mathcal{F}^v = \mathcal{G}^v$ because For each $g(x) \in \mathcal{G}^v$, denote $\bar{g} = \int g(x)w(x)dx$, then we have $g(x) = \bar{g} + (g(x) - \bar{g})$. It is easy to check that $\bar{g}$ is pure and $g(x) - \bar{g}$ is pure, thus any univariate function has pure representation. Suppose $\mathcal{F}^u = \mathcal{G}^u$ for $|u| < d$. For any $|v| = d$, let $\mathcal{E}^v = \{g | g = \sum_{u \subset v} g_u\}$. It is easy to see $\mathcal{E}^v$ is closed and convex. Furthermore, $\forall g \in \mathcal{E}^v, g$ has a pure representation, because of the induction hypothesis and lemma A.2. $\forall f \in \mathcal{G}^v$, By the projection theorem [19], there is a unique element $e \in \mathcal{E}^v$ minimizing $||f - e||$. Denote $\delta = f - e$. We have $\delta \in \mathcal{G}^v$, and δ is orthogonal to $\mathcal{G}^u, \forall u \subset v$. Hence, $\delta \in \mathcal{F}^v$ by definition. Since $e \in \mathcal{E}^v \subset \mathcal{F}^v$ and $\delta \in \mathcal{F}^v$, $f = \delta + e \in \mathcal{F}^v$.

Then we show $\mathcal{F}_d = \mathcal{G}_d$. $\forall f \in \mathcal{G}_d, f = \sum_{u \in \mathcal{D}_d} f_u$. As shown above, $f_u \in \mathcal{F}^u$ for each u. By lemma A.2, there sum has a pure representation. Hence, $f \in \mathcal{F}_d$.

□

We further need to define a "grid-closed" density w, which is a mild assumption for ruling out degenerate densities. Degenerate densities would produce collinearity or concurvity among features, which leads to non-unique optimal solutions.

Definition 11. *A density w is said to be grid closed if its support $\Omega \subseteq \mathbb{R}^p$ satisfies that, for any $x \in \Omega$ and any $u \subset \{1, \cdots, p\}$, there exists at least another point $y \in \Omega, y \neq x$ such that $y_u = x_u$.*

Lemma A.4. *Let w be grid closed. $\{f_u | u \in \mathcal{D}_k, f_u \text{ is pure}\}$ are linearly independent under the inner product $\langle \cdot, \cdot \rangle_w$, if at least one $f_u \neq 0$.*

The proof of lemma A.4 is a reminder of lemma B.2 in [14].

Lemma A.5. *If a function g has pure representation $g = \sum_{u \in \mathcal{D}_k} h_u$ s.t. h_u is pure, with a grid closed density w, then such representation is unique.*

Proof. Otherwise, suppose g has two pure representations: $g = \sum_{u \in \mathcal{D}_k} h_u$ s.t. h_u is pure, $g = \sum_{u \in \mathcal{D}_k} f_u$ s.t. h_u is pure, which implies that $0 = \sum_{u \in \mathcal{D}_k} (h_u - f_u)$ s.t. $(h_u - f_u)$ is pure (lemma A.1), and by considering pure components are linearly independent if w is grid closed (lemma A.4), we get $h_u - f_u = 0, \forall u \in \mathcal{D}_k$, thus the representation is unique. □

Theorem 3.1. *For arbitrary k, solution of (4) is unique when density w is non-degenerate, or more precisely, w is grid closed.*

Proof. Following standard deduction, it is shown that $\mathcal{G}_k := \{f | f = \sum_{u \in \mathcal{D}_k} f_u\}$ is a closed subspace in $\mathcal{L}^2$ (this is a direct consequence of lemma 3.1 in [26]. Denote $F(x) := \mathbb{E}[Y|x]$, according to the projection theorem, there exists a unique minimizing element f in $\mathcal{G}_k$ such that $\langle F - f, F - f \rangle_w < \langle F - h, F - h \rangle_w, \forall h \in \mathcal{G}_k$ which exactly minimizes the objective of (4). We further know that the representation of f into pure components is also unique (lemma A.3 and A.5), we thus have completed the proof. □

A.2 Proof of Lemma 3.1

Lemma 3.1. *For $i \in u$,*

$$\frac{1}{Nh_{u\setminus i}} \sum_{s=1}^{N} K_{u\setminus i}\left(\frac{x_{u\setminus i} - x_{u\setminus i,s}}{h_{u\setminus i}}\right) f_u(x_{u,s}|\boldsymbol{\eta}_u, h_u) \sim \int f_u(x_u)\, w_u(x_u) dx_i. \tag{23}$$

Here $\sim$ denotes asymptotic approximation. In other words, $\frac{1}{Nh_{u\setminus i}} \sum_{s=1}^{N} K_{u\setminus i}\left(\frac{x_{u\setminus i} - x_{u\setminus i,s}}{h_{u\setminus i}}\right) f_u(x_{u,s}|\boldsymbol{\eta}_u, h_u)$ is an empirical version of $\int f_u(x_u) w_u(x_u) dx_i$.

Proof. By Theorem 6.27 of [31], the estimation

$$\frac{1}{Nh_{u\setminus i}} \sum_{s=1}^{N} K_{u\setminus i}\left(\frac{x_{u\setminus i} - x_{u\setminus i,s}}{h_{u\setminus i}}\right) \to w_{u\setminus i}(x_{u\setminus i}) \tag{24}$$

if $h_{u\setminus i} \to 0$ as $n \to \infty$ and $nh \to \infty$, and by Theorem 2 of [10] the estimation

$$\frac{\frac{1}{Nh_{u\setminus i}} \sum_{s=1}^{N} K_{u\setminus i}\left(\frac{x_{u\setminus i} - x_{u\setminus i,s}}{h_{u\setminus i}}\right) f(x_{u,s})}{\frac{1}{Nh_{u\setminus i}} \sum_{s=1}^{N} K_{u\setminus i}\left(\frac{x_{u\setminus i} - x_{u\setminus i,s}}{h_{u\setminus i}}\right)} \to \mathbb{E}(f(X_u)|X_{u\setminus i} = x_{u\setminus i}) = \int f(x_u)\, w_{i|u\setminus i}(x_i|x_{u\setminus i}) dx_i \tag{25}$$

$w_{u\setminus i}$-a.e. under moderate conditions (these conditions are satisfied by common kernels such as the Gaussian kernel). Multiplying (24) and (25), we obtain that

$$\frac{1}{Nh_{u\setminus i}} \sum_{s=1}^{N} K_{u\setminus i}\left(\frac{x_{u\setminus i} - x_{u\setminus i,s}}{h_{u\setminus i}}\right) f(x_{u,s}) \to w_{u\setminus i}(x_{u\setminus i}) \int f(x_u)\, w_{i|u\setminus i}(x_i|x_{u\setminus i}) dx_i = \int f(x_u)\, w_u(x_u) dx_i \tag{26}$$

$w_{u\setminus i}$-a.e. □

Table 5: Results of performance and pureness on real-world datasets

	RMSE				log(pure)	
Model	**pureGAM**	**GAMI-Net**	**EBM**	**XGB (ref)**	**pureGAM**	**GAMI-Net**
air quality	**0.0172 ± 0.0004**	0.0203 ± 0.0020	0.0221 ± 0.0008	0.0152 ± 0.0002	**-1.0953 ± 0.0484**	-1.0491 ± 0.0222
electrical grid	**0.0475 ± 0.0007**	0.0545 ± 0.0012	0.0528 ± 0.0010	0.0419 ± 0.0006	**-2.2100 ± 0.1108**	-1.8016 ± 0.0731
elevators	**0.0336 ± 0.0003**	0.0370 ± 0.0010	0.0378 ± 0.0011	0.0328 ± 0.0004	**-1.3436 ± 0.0500**	-1.0745 ± 0.0646
kinematics	**0.1159 ± 0.0007**	0.1219 ± 0.0016	0.1195 ± 0.0012	0.0817 ± 0.0009	**-2.6466 ± 0.1915**	-1.7422 ± 0.2427
abalone	0.0761 ± 0.0036	**0.0758 ± 0.0041**	0.0787 ± 0.0031	0.0825 ± 0.0023	**-1.2712 ± 0.0755**	-1.1903 ± 0.4487
ailerons	**0.0182 ± 0.0223**	0.0186 ± 0.0228	0.0454 ± 0.0011	0.0443 ± 0.0008	**-0.3983 ± 0.4887**	-0.3159 ± 0.3870
airfoil	0.0593 ± 0.0030	**0.0575 ± 0.0039**	0.0641 ± 0.0029	0.0376 ± 0.0032	**-2.5233 ± 0.0425**	-1.5154 ± 0.1410
airplane	**0.0263 ± 0.0014**	0.0299 ± 0.0027	0.0348 ± 0.0015	0.0274 ± 0.0020	**-1.2363 ± 0.0665**	-0.9708 ± 0.1115
bank	**0.0400 ± 0.0010**	0.0406 ± 0.0012	0.0422 ± 0.0008	0.0404 ± 0.0015	**-2.0890 ± 0.0477**	-1.0506 ± 0.1543
california house	**0.0850 ± 0.0018**	0.0875 ± 0.0042	0.0872 ± 0.0014	0.0757 ± 0.0011	**-1.2847 ± 0.0339**	-0.6651 ± 0.5452
CCCP	**0.0514 ± 0.0014**	0.0520 ± 0.0010	0.0543 ± 0.0008	0.0397 ± 0.0015	**-1.7065 ± 0.1367**	-1.2723 ± 0.1305
delta elevators	**0.0527 ± 0.0010**	0.0527 ± 0.0011	0.0529 ± 0.0011	0.0572 ± 0.0009	**-1.2604 ± 0.1079**	-0.7851 ± 0.2458
disclosure z	**0.1169 ± 0.0100**	0.1196 ± 0.0121	0.1179 ± 0.0097	0.1381 ± 0.0087	**-0.6953 ± 0.1672**	-0.1634 ± 0.3267
era	0.2000 ± 0.0084	0.1987 ± 0.0074	**0.1978 ± 0.0060**	0.1998 ± 0.0057	**-1.0922 ± 0.2008**	-0.0953 ± 0.1907
parkinsons tele	**0.0481 ± 0.0019**	0.0499 ± 0.0027	0.0493 ± 0.0022	0.0486 ± 0.0028	**-1.5337 ± 0.0940**	-1.0180 ± 0.3416
seoul bike	0.1191 ± 0.0030	0.1195 ± 0.0026	**0.1183 ± 0.0021**	0.1185 ± 0.0028	**-1.5933 ± 0.0318**	-1.1291 ± 0.1506
steel industry	**0.0236 ± 0.0040**	0.0263 ± 0.0037	0.0277 ± 0.0033	0.0255 ± 0.0049	**-1.6094 ± 0.1374**	-1.0310 ± 0.1637
skill craft	0.0687 ± 0.0066	**0.0673 ± 0.0068**	0.0678 ± 0.0071	0.0716 ± 0.0075	**-1.2495 ± 0.0729**	-0.5005 ± 0.2740
treasury	0.0118 ± 0.0023	**0.0112 ± 0.0023**	0.0135 ± 0.0015	0.0117 ± 0.0017	**-1.0223 ± 0.0470**	-0.7126 ± 0.1518
weather wizmir	0.0219 ± 0.0004	**0.0189 ± 0.0012**	0.0246 ± 0.0014	0.0205 ± 0.0015	**-1.1576 ± 0.0416**	-0.5930 ± 0.0804
wind	**0.0738 ± 0.0021**	0.0738 ± 0.0016	0.0744 ± 0.0013	0.0737 ± 0.0017	**-1.3443 ± 0.2990**	-1.2849 ± 0.1054
wine red	0.1275 ± 0.0065	0.1303 ± 0.0058	**0.1253 ± 0.0063**	0.1193 ± 0.0059	**-1.0192 ± 0.1350**	-0.4908 ± 0.6060
wine white	0.1161 ± 0.0034	0.1190 ± 0.0034	**0.1122 ± 0.0026**	0.0998 ± 0.0027	**-1.3334 ± 0.1812**	-0.8494 ± 0.3533

B PURE CODING FOR GENERAL ORDER-k INTERACTIONS

Let $(x_1, \cdots, x_k)$ be k categorical features. Suppose x_i takes values in $L_i = \{1, \cdots, l_i\}, i = 1, \cdots, k$. Let $q_{r_1 \cdots r_k} := q(r_1, \cdots, r_k)$ be the frequency of $\{x_1 = r_1, \cdots, x_k = r_k\}$.

Lemma B.1. *Assume $q_{r_1 \cdots r_k} \neq 0, \forall r_i \in \{1, \cdots, l_i\}, i = 1, \cdots, k$. $\forall r_i \in \{1, \cdots, l_i - 1\}, i = 1, \cdots, k$, define $e_{\{1,\cdots,k\} r_1 \cdots r_k}(x_1, \cdots, x_k)$*

$$e_{\{1,\cdots,k\} r_1 \cdots r_k} = \begin{cases} \frac{1}{q_{r_1 \cdots r_k}} & (x_1, \cdots, x_k) = (r_1, \cdots, r_k), \\ -\frac{1}{q_{l_1 r_2 \cdots r_k}} & (x_1, \cdots, x_k) = (l_1, r_2, \cdots, r_k), \\ \vdots & \vdots \\ -\frac{1}{q_{r_1 \cdots r_{k-1} l_k}} & (x_1, \cdots, x_k) = (r_1, \cdots, r_{k-1}, l_k), \\ \frac{1}{q_{l_1 \cdots l_k}} & (x_1, \cdots, x_k) = (l_1, \cdots, l_k), \\ 0, & otherwise. \end{cases} \tag{27}$$

Then $\{e_{\{1,\cdots,k\} r_1 \cdots r_k} : r_i \in \{1, \cdots, l_i - 1\}, i = 1, \cdots, k\}$ forms a basis of the solution to the pureness condition.

Proof. Consider the isomorphism from the space functions on $x_1, \cdots, x_k$ to the space of $l_1 \times \cdots \times l_k$-tensor of values taken on each category combination

$$f \to (f(s_1, \cdots, s_k))_{s_i \in \{1,\cdots,l_i\}},$$

If f satisfies the pureness condition

$$\forall i \in \{1, \cdots, k\}, \sum_{x_i \in L_i} f(x_1, \cdots, x_k) q(x_1, \cdots, x_k) = 0, \tag{28}$$

then there is an isomorphism

$$(f(s_1, \cdots, s_k))_{s_i \in \{1,\cdots,l_i\}} \to (f(s_1, \cdots, s_k))_{s_i \in \{1,\cdots,(l_i-1)\}},$$

because given $(f(s_1, \cdots, s_k))_{s_i \in \{1,\cdots,(l_i-1)\}}$, there exists a unique preimage $(f(s_1, \cdots, s_k))_{s_i \in \{1,\cdots,l_i\}}$, where

$$\begin{aligned} & f(l_1, r_2, \cdots, r_k) \\ &= -\frac{1}{q(l_1, r_2 \cdots, r_k)} \sum_{r_1=1}^{l_1} f((r_1, r_2, \cdots, r_k)) q(r_1, r_2, \cdots, r_k) \\ & \vdots \\ & f(r_1, r_2, \cdots, l_k) \\ &= -\frac{1}{q(r_1, r_2 \cdots, l_k)} \sum_{r_k=1}^{l_1} f((r_1, r_2, \cdots, r_k)) q(r_1, r_2, \cdots, r_k) \end{aligned} \tag{29}$$

is uniquely determined following from the pureness condition.

Thus the space of functions satisfying pureness has dimension $(l_1 - 1) \cdots (l_k - 1)$. On the other hand, $\{e_{\{1,\cdots,k\} r_1 \cdots r_k \{1,\cdots,k\}} : r_i \in \{1, \cdots, l_i\}, i = 1, \cdots, k\}$ satisfy the pureness condition, and are linearly independent. Thus, they form a basis of a $(l_1 - 1) \cdots (l_k - 1)$-dimensional subspace of the pure functions, and this is the entire space because the dimensions coincide. □

Lemma 3.2 below is a direct corollary of Lemma B.1.

Lemma 3.2. *with $\boldsymbol{e}_u(x_u) = (e_{\{1,\cdots,k\} r_1, \cdots, r_k})^T_{r_i \in \{1,\cdots,l_i-1\}, i=1,\cdots,k}$ defined in (27),*
let

$$f_u(x_u | \boldsymbol{\eta}_u) = \boldsymbol{e}_u(x_u)^T \boldsymbol{\eta}_u. \tag{30}$$

This function (30) satisfies the discrete empirical pureness condition for any parameter $\boldsymbol{\eta}_u \in \mathbb{R}^{o_u}$. Moreover, the space of such functions is equal to the entire space of functions on x_u satisfying the discrete empirical pureness condition (10)

C SYNTHETIC DATASETS

Numerical Data. We generate data at different scales from a multivariate normal distribution, with random covariance matrices. The ground truth is a sum of main effects and interactions randomly selected from the following functions
$x, x^2, 2^x, \sin(x), xy, 2^{x+y}, \sin(x + y)$, and each term is multiplied by a random coefficient.

For synthetic data with interaction up to order $k = 3$, terms of functions randomly selected from $xyz, 2^{x+y+z}, \sin(x + y + z)$, in addition to the ones above, are added to the ground truth function.

Categorical Data. We first generate numerical data and then categorize them by cutting at quantiles. Data points are added to make sure there each categorical value combination is populated by at least one point. The ground truth function is the sum of univariate component functions on every feature and bivariate component functions on a randomly selected subset of feature pairs. Each component function takes a randomly selected number for each categorical value (or pair of values).