

Symbolic Regression Using Synergistic Basis Functions

Hamide Sarvary Karizaky

Amirkabir University of Technology

Mohammad Mehdi Ebadzadeh

ebadzadeh@aut.ac.ir

Amirkabir University of Technology

Research Article

Keywords: Symbolic Regression, Basis Functions, Synergy-based Measure, Interpretability

Posted Date: October 22nd, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-11025063/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Symbolic Regression Using Synergistic Basis Functions

Hamide Sarvary Karizaky¹ and Mohammad Mehdi Ebadzadeh^{1*}

^{1*} Department of Computer Engineering and Information Technology,
Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran.

*Corresponding author(s). E-mail(s): ebadzadeh@aut.ac.ir;
Contributing authors: h.sarvari@aut.ac.ir;

Abstract

Symbolic regression is a powerful approach for discovering mathematical relationships from data, but conventional approaches often rely on computationally expensive evolutionary search and may produce unnecessarily complex representations. To address these limitations, this paper introduces Synergistic Basis Functions Regression (SBFR), a non-evolutionary approach to symbolic regression. SBFR replaces evolutionary search with synergy-guided basis-function generation and selection. It generates candidate basis functions in tree form and dynamically maintains a subset of the most synergistic functions. The selected basis functions are iteratively updated using a synergy-based measure that evaluates their contribution in combination with the current feature set, after which a regression model is constructed using the selected functions. The effectiveness of SBFR is evaluated on nine benchmark regression datasets and compared with corresponding baseline regression models. The results show that SBFR provides competitive predictive performance while maintaining compact feature representations. Across the benchmark datasets, SBFR achieves average relative reductions of 40.14% in root mean squared error (RMSE) and 41.53% in mean absolute error (MAE), together with an average relative improvement of 21.56% in the coefficient of determination (R^2), compared with the corresponding baseline regressors. Furthermore, the SBFR variant employing Random Forest as the downstream regression model typically selects only 4–12 basis functions across the evaluated datasets, resulting in compact feature representations while avoiding the need for evolutionary search.

Keywords: Symbolic Regression, Basis Functions, Synergy-based Measure, Interpretability

1 Introduction

Regression is a fundamental task in data mining and machine learning that aims to learn the relationship between input variables and a continuous target variable. Traditional approaches such as linear regression provide simple and interpretable models but rely on predefined functional forms, which may be inadequate for nonlinear relationships and complex variable interactions. Nonlinear models such as Artificial Neural Networks (ANNs) offer greater modeling flexibility [Hornik \(1991\)](#), but their learned representations are generally difficult to interpret.

Symbolic regression (SR) has emerged as an alternative approach that combines predictive modeling with explicit mathematical representation. Unlike conventional regression methods that typically assume a predefined model structure, SR searches over a space of mathematical expressions to identify a functional form that describes the relationship between input variables and the target. The resulting expressions are explicit mathematical models, allowing the discovered relationships to be inspected and interpreted directly. This property makes SR particularly attractive in applications where understanding the underlying relationships is as important as predictive accuracy.

Despite its interpretability advantages, SR is challenging because of its large and complex search space. The number of possible symbolic expressions grows rapidly with the number of input variables, mathematical operators, and allowable structures, leading to substantial computational costs. An effective SR method must therefore efficiently identify useful expressions while avoiding redundant or unnecessarily complex representations.

GP has been one of the most widely used approaches for SR. By representing mathematical expressions as tree structures and evolving them through genetic operators, GP can explore diverse symbolic models without requiring a predefined functional form. Various extensions have been proposed to improve search efficiency, including statistical enhancements [Haeri et al. \(2017\)](#) and semantic-aware operators [Uy et al. \(2011\)](#); [Moraglio et al. \(2012\)](#); [Pawlak et al. \(2014\)](#); [La Cava and Moore \(2019\)](#). Nevertheless, evolutionary SR methods require repeated evaluation of large populations of candidate expressions, which can result in high computational costs and slow convergence, particularly for complex search spaces.

To address some of the limitations of evolutionary search, several non-evolutionary SR methods have been developed. FFX [McConaghy \(2011\)](#), for example, employs pathwise regularized learning within a generalized linear model framework to identify compact sets of basis functions. Other approaches include elite basis selection [Chen et al. \(2017a\)](#), Interaction-Transformation [de França \(2018\)](#), grammar-based variational autoencoders [Kusner et al. \(2017\)](#), and compressed sensing [Ouyang et al. \(2018\)](#). However, existing non-evolutionary approaches still face challenges related to basis-function selection and the exploitation of interactions among candidate functions. For example, the computational cost of FFX may increase with the size of the candidate basis-function set, potentially making basis-function selection increasingly expensive.

An important but relatively underexplored aspect of symbolic model construction is the interaction among candidate basis functions. The predictive contribution of a basis function cannot always be assessed independently of the functions already

included in a model. A function that provides limited improvement in isolation may become highly useful when combined with complementary functions, whereas individually useful functions may become redundant when selected together. Therefore, selecting basis functions solely according to their individual predictive contribution may overlook valuable combinations and result in redundant representations. This observation motivates the development of a SR approach that explicitly evaluates the synergistic contribution of candidate functions in the context of the functions already selected.

Building on our previous work on synergistic feature construction for classification [Karizaky and Ebadzadeh \(2026\)](#), this study extends the concept to SR without evolutionary search. The proposed Synergistic Basis Functions Regression (SBFR) reformulates synergy for regression and dynamically constructs and selects symbolic basis functions according to their contribution to regression performance.

SBFR evaluates each newly generated candidate in the context of the current feature vector and either adds it to the existing set or replaces an existing basis function when it provides a greater synergistic improvement. This dynamic selection process promotes complementary basis functions, reduces redundancy, and maintains a compact symbolic representation.

Experimental results on nine benchmark regression datasets demonstrate that SBFR can substantially improve predictive performance for several downstream regression models while maintaining compact symbolic representations. Compared with the corresponding baseline regressors, SBFR achieves average improvements of 40.14% in RMSE, 41.53% in MAE, and 21.56% in R^2 .

The main contributions of this work are summarized as follows:

- The synergistic feature construction concept developed in our previous work for classification [Karizaky and Ebadzadeh \(2026\)](#) is extended to SR, resulting in the non-evolutionary Synergistic Basis Functions Regression (SBFR) framework.
- A dynamic selection strategy is introduced to retain complementary basis functions, reduce redundancy, and maintain a compact symbolic representation without evolutionary search.
- SBFR is comprehensively evaluated on nine benchmark regression datasets against state-of-the-art SR methods in terms of predictive performance and model compactness.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 presents SBFR, while Section 4 describes the experimental setup and results. Section 5 concludes the paper and discusses future work. Tables 1 and 2 provide a reference for the abbreviations and symbols used throughout the paper.

Table 1: List of Abbreviations.

Abbr.	Description
ANN	Artificial Neural Network
BGP	Behavioral GP
ConVAE	Convolutional Variational Autoencoder

Abbr.	Description
DAE	Denosing Autoencoder
DSR	Deep Symbolic Regression
DT	Decision Tree
EBR	Elite Bases Regression
EC	Evolutionary Computation
EQL	Equation Learner
EvoF	Evolutionary Forest
FEAT	Feature Engineering and Automatic Transformation
faiGP	Functional Algebraic GP
FFX	Fast Function Extraction
GEP	Gene Expression Programming
GP	Genetic Programming
GPU	graphics processing unit
GSGP	Geometric Semantic GP
KNN	K-Nearest Neighbors
LSTM	Long Short-Term Memory
MGGP	Multigene GP
MINLP	Mixed-Integer Nonlinear Programming
ML	Machine Learning
MLP	Multi-Layer Perceptron
MRGP	Multiple Regression GP
MSE	Mean Squared Error
M*GP	Multi-dimensional GP
NEAT	NeuroEvolution of Augmenting Topologies
RF	Random Forest
RHH	Ramped Half-and-Half
RMSE	Root Mean Squared Error
SBFR	Synergistic Basis Functions Regression
SGP	Semantic GP
SR	Symbolic Regression
SVR	Support Vector Regression

2 Related Work

SR has attracted considerable research attention because of its ability to discover explicit mathematical models while providing predictive capability and interpretability. Numerous approaches have been developed to improve search efficiency, scalability, and model compactness. These methods can broadly be categorized into evolutionary (GP-based), non-evolutionary, and hybrid approaches. The following sections review representative methods in each category, with emphasis on their search strategies, computational efficiency, and ability to construct compact symbolic models.

Table 2: List of Symbols.

Symbol	Description
n, p	Number of samples and input variables in the dataset.
X, Y	Input matrix $X \in \mathbb{R}^{n \times p}$ and corresponding continuous target values $Y \in \mathbb{R}^n$ for a regression task.
x_i, y_i	Input vector and corresponding target value of the i -th sample.
T	Training dataset $\{(x_i, y_i)\}_{i=1}^n$.
$\phi(x)$	Mapping from the original input space $\mathbb{R}^p$ to a space of m constructed symbolic basis functions.
f	Candidate symbolic basis function represented as a tree.
FV	Current set of selected symbolic basis functions.
m	Number of currently selected symbolic basis functions.
$F(\phi(X), Y)$	Cross-validated RMSE of a regression model trained on the transformed features $\phi(X)$.
$Synergy(f, FV)$	Synergy of a candidate basis function f with respect to the currently selected basis-function set FV .
$FV_{\text{tmp}, i}$	Temporary basis-function set obtained by replacing the i -th function in FV .
FV_{Add}	Basis-function set obtained after adding a new candidate function.
S_{Add}	Synergy value obtained when adding a candidate basis function.
SV_{Rep}	Vector of synergy values obtained for replacement operations.
$improve_{\text{threshold}}$	Minimum required improvement in the regression objective for accepting a candidate basis function.
FS, TS	Function set and terminal set used to construct symbolic expressions.
D	Maximum depth of a symbolic expression tree.
b	Upper bound on the branching factor of a symbolic expression tree.
k	Number of cross-validation folds.
I	Number of SBFR candidate-generation iterations.
M	Final number of retained basis functions in SBFR.
$C_{\text{model}}(n, m, k)$	Computational cost of training and evaluating the downstream regression model over k cross-validation folds.
<i>Symbols used for baseline methods</i>	
G	Number of generations in evolutionary baseline methods.
P	Population size in GP-based baseline methods.
t	Average number of nodes per symbolic tree.
T_{M3GP}	Number of trees per individual in M3GP.
Q	Number of fidelity levels in Evolutionary Forest.
B_{FFX}	Number of basis functions generated by FFX.
L_{gen}	Average computational cost of generating one FFX basis function.
λ_{path}	Number of regularization parameters used in the pathwise regularization of FFX.

2.1 Evolutionary (GP-based) Approaches for SR

GP is one of the most widely adopted evolutionary paradigms for SR. Recent extensions have focused on improving search efficiency, scalability, robustness, and interpretability through enhanced representations, semantic guidance, feature discovery, and deterministic learning.

2.1.1 Search Enhancement and Semantic Guidance

The representation of symbolic expressions and the design of genetic operators strongly influence the efficiency of GP-based SR. Conventional syntactic representations may lead to redundant exploration and non-local changes in program behavior, motivating the incorporation of semantic information into evolutionary search. Semantic GP (SGP) uses program behavior to guide initialization, selection, and genetic variation. Geometric semantic GP (GSGP) [Moraglio et al. \(2012\)](#) modifies genetic operations based on program semantics, while semantic mutation and crossover [Uy et al. \(2011\)](#); [Pawlak et al. \(2014\)](#) guide variation using behavioral information. Lexicase selection [Ding et al. \(2022\)](#) promotes behavioral diversity by selecting individuals based on performance across individual cases, while adaptive semantic operators [La Cava and Moore \(2019\)](#); [Yang et al. \(2024\)](#) dynamically adjust the search process based on semantic information. Multi-tree representations have also been investigated to improve expression construction and search flexibility [Al-Helali et al. \(2021\)](#). Coefficient optimization has been incorporated into GP to improve the numerical parameters of evolved expressions [Virgolin and Bosman \(2022\)](#). Other strategies address computational cost and expression growth through surrogate fitness evaluation [Kattan and Ong \(2015\)](#); [Kattan et al. \(2016\)](#) and bloat-control mechanisms [Trujillo et al. \(2016\)](#); [Stanley and Miikkulainen \(2002\)](#); [Sareni and Krahenbuhl \(1998\)](#). Alternative tree-, linear-, graph-, and grammar-based representations have also been explored to modify the search space and improve expression construction [Bi et al. \(2020\)](#); [La Cava and Moore \(2020\)](#); [Huang et al. \(2023, 2022\)](#); [Manazir and Raza \(2019\)](#); [Criado et al. \(2024\)](#).

2.1.2 Feature Discovery and Deterministic Learning

Another line of research uses evolutionary search primarily to discover informative features or subexpressions, while delegating their combination and parameter estimation to deterministic learning models. Feature importance, permutation analysis, and dimensionality reduction have been used to guide GP toward informative variables [Al-Helali et al. \(2024\)](#); [Chen et al. \(2016, 2017b\)](#); [Zhong et al. \(2021\)](#).

In deterministic-enhanced GP, evolved features are combined using conventional learning algorithms, particularly linear regression. Population-wide methods jointly optimize evolved features [De Melo \(2014\)](#); [Arnaldo et al. \(2015, 2014\)](#); [De Melo and Banzhaf \(2018\)](#); [La Cava and Moore \(2017a,b\)](#), whereas MRGP and BGP extract useful building blocks from program traces [Arnaldo et al. \(2014\)](#); [Krawiec and O'Reilly \(2014\)](#); [Krawiec et al. \(2016\)](#). Multi-tree and multi-gene approaches, including MGGP, M3GP, M4GP, and FEAT, similarly combine multiple evolved components using

machine learning models [Searson et al. \(2010\)](#); [Muñoz et al. \(2019\)](#); [La Cava et al. \(2018\)](#).

Recent studies have incorporated graphics processing unit (GPU) acceleration., multi-objective optimization, adaptive tree structures, and external feature selection to further improve efficiency and interpretability [Cárdenas Florido et al. \(2024\)](#); [Batista et al. \(2024\)](#); [Niu et al. \(2025\)](#); [Gharoun et al. \(2024\)](#). Nevertheless, these methods still depend on evolutionary search and therefore incur the computational cost of population evaluation and repeated fitness estimation. Moreover, comparatively limited attention has been given to the incremental and cooperative contribution of a candidate basis function when combined with an already constructed basis set.

2.2 Non-evolutionary Methods for SR

Despite extensive efforts to improve GP, its evolutionary nature requires repeated evaluation of populations of candidate expressions, which can make symbolic search computationally demanding. These limitations have motivated the development of non-evolutionary approaches that avoid population-based evolutionary search and instead employ alternative strategies for symbolic model construction, including deterministic optimization, structured enumeration, probabilistic inference, and neural learning.

2.2.1 Deterministic Approaches

Deterministic SR methods replace stochastic evolutionary search with systematic enumeration, optimization, or structured basis-function construction [Dong and Zhong \(2025\)](#). Grammar-guided methods construct candidate expressions within predefined structural constraints [Kammerer et al. \(2020\)](#), while exhaustive approaches explore constrained expression spaces using mechanisms such as Pareto prioritization and algebraic equivalence checking [Kronberger et al. \(2019\)](#); [Austel et al. \(2020\)](#). Other approaches employ semantic hashing to reduce redundant exploration [Kim et al. \(2023a\)](#), or dimensional and structural constraints to control the search space [Neumann et al. \(2020\)](#); [Engle and Sahinidis \(2022\)](#). Mathematical programming formulations, particularly mixed-integer nonlinear programming (MINLP), provide an optimization-based alternative that can incorporate structural and domain-specific constraints, although scalability remains challenging for large search spaces [Gunawan et al. \(2017\)](#).

Basis-function approaches instead generate candidate symbolic functions and select a subset for regression. Fast Function Extraction (FFX) [McConaghy \(2011\)](#) combines symbolic basis generation with pathwise regularized learning, while Elite Bases Regression (EBR) [Chen et al. \(2017a\)](#) iteratively maintains a small set of candidate functions according to their contribution to the regression model. Other extensions incorporate nonlinear least-squares optimization or hybrid evolutionary search [Kammerer et al. \(2022\)](#); [Icke and Bongard \(2013\)](#), and Bayesian approaches perform posterior inference over symbolic structures and parameters [Jin et al. \(2019\)](#).

Although deterministic methods avoid stochastic evolutionary search, exhaustive and optimization-based approaches may still face combinatorial or computational

challenges. Basis-function methods offer greater scalability, but their effectiveness depends strongly on candidate generation and selection. In particular, existing selection strategies generally emphasize individual predictive contribution, correlation, or regularization, with less emphasis on the synergistic contribution of a candidate function in combination with an already selected set.

2.2.2 Deep Learning-Based Methods for SR

Deep learning has emerged as another non-evolutionary paradigm for SR, including deep reinforcement learning, transformer-based models, equation learners, and generative approaches [Petersen et al. \(2021\)](#); [Kim et al. \(2023b\)](#). Deep Symbolic Regression (DSR), for example, formulates expression construction as a sequential decision process and learns a policy for generating symbolic expressions [Petersen et al. \(2020\)](#). Subsequent studies have incorporated free parameters, physical constraints, grammar guidance, and more efficient policy learning [Landajuela et al. \(2022\)](#); [Tenachi et al. \(2023\)](#); [Guo et al. \(2024\)](#); [Crochepierre et al. \(2022\)](#).

Transformer and generative approaches learn mappings between numerical observations and symbolic expressions, often using pre-training on synthetic equations to accelerate equation discovery [Biggio et al. \(2021\)](#); [Vastl et al. \(2022\)](#); [Holt et al. \(2022\)](#); [Kamienny et al. \(2023\)](#). Equation Learners (EQL) use symbolic operators as network components combined with sparse learning to obtain interpretable expressions [Martius and Lampert \(2016\)](#); [Sahoo et al. \(2018\)](#); [Costa et al. \(2020\)](#); [Dong et al. \(2024\)](#); [Chen \(2020\)](#). Despite their ability to avoid conventional evolutionary search, these approaches generally require substantial training data and computational resources, and their performance can depend on the training distribution and architecture.

2.3 Hybrid Methods

Hybrid approaches combine evolutionary search with complementary non-evolutionary learning techniques to exploit their respective strengths. In SR, these methods typically use learning models to guide the evolutionary search, modify representation or variation mechanisms, or construct and evaluate symbolic features more efficiently.

One direction uses neural networks to guide the evolutionary search. Mundhenk et al. [Mundhenk et al. \(2021\)](#) proposed a neural-guided symbolic regression framework in which a neural model generates promising initial populations for GP, progressively improving the quality of starting points for evolutionary search. Similarly, Zhong et al. [Zhong et al. \(2018\)](#) introduced a deep learning-assisted framework in which an encoder predicts symbol positions and a decoder translates the resulting sequences into Gene Expression Programming (GEP) expressions for subsequent fitness evaluation. These approaches demonstrate that learned models can provide useful search guidance while retaining the flexible representation of evolutionary SR.

Another direction integrates neural learning mechanisms with evolutionary or symbolic representations. Dong et al. [Dong et al. \(2024\)](#) proposed the evolving Equation Learner (eEQL), which incorporates Automatically Defined Functions (ADFs) into an equation-learning architecture, allowing the network structure to evolve during the

learning process. Razavi et al. [Razavi and Gamazon \(2022\)](#) proposed functional algebraic GP (faiGP), which combines GP with a convolutional variational autoencoder (ConVAE) and a Bayesian classifier. The ConVAE provides dimensionality reduction, while the Bayesian component learns prior distributions for operators and operands, thereby guiding the generation of GP expressions and reducing code bloat. Wittenberg et al. [Wittenberg et al. \(2020\)](#) further replaced conventional GP recombination and mutation operators with a probabilistic model based on denoising autoencoder long short-term memory networks (DAE-LSTMs).

A closely related family of hybrid methods combines evolutionary feature construction with deterministic machine learning models. Rather than requiring GP to directly optimize all parameters of the final predictive model, these methods use GP to discover nonlinear transformations or basis functions and delegate their combination or parameter estimation to a deterministic learner. This strategy has been explored in population-wide, multigene, and multi-tree GP approaches [De Melo \(2014\)](#); [Arnaldo et al. \(2015, 2014\)](#); [Muñoz et al. \(2019\)](#); [La Cava et al. \(2018\)](#). More recent methods, including M5GP and M6GP, further combine GP-based feature construction with linear regression or multi-objective optimization [Cárdenas Florido et al. \(2024\)](#); [Batista et al. \(2024\)](#). Although these approaches can improve the efficiency and interpretability of GP-based models, their effectiveness still depends on the evolutionary search used to generate candidate features.

The proposed Synergistic Basis Functions Regression (SBFR) is related to this family of feature-construction approaches, as it also separates nonlinear basis-function construction from predictive model learning. However, unlike conventional GP-based methods, SBFR does not evolve basis functions as individuals within a population. Instead, candidate basis functions are iteratively generated and evaluated according to their contribution to the predictive model and their synergy with the basis functions already selected. This provides a direct mechanism for refining the basis-function representation based on cooperative effects among candidate functions. Furthermore, SBFR is model-agnostic and can employ different learning algorithms, including linear regression, support vector regression (SVR), and k-nearest neighbors (KNN), for evaluating the constructed basis representations.

SBFR is also closely related to Fast Function Extraction (FFX), as both construct predictive models using nonlinear basis functions. However, their search and selection mechanisms differ fundamentally. FFX generates basis functions through deterministic enumeration and uses pathwise regularized learning within a generalized linear model to select and combine them [McConaghy \(2011\)](#). In contrast, SBFR iteratively generates and evaluates candidate functions according to their predictive improvement and synergistic contribution, dynamically refining the selected basis set rather than selecting from a predetermined collection. Moreover, SBFR is model-agnostic and can employ different downstream learning algorithms.

Overall, existing SR methods have substantially improved search efficiency, scalability, and interpretability, but important trade-offs remain between evolutionary flexibility, computational cost, and compact symbolic representation. In particular, comparatively limited attention has been given to basis-function construction

strategies that explicitly consider both the individual predictive contribution of a candidate and its cooperative effect when combined with an existing basis set. SBFR addresses this gap by iteratively constructing and refining a compact basis-function representation based on predictive contribution and synergy.

3 Synergistic Basis Functions Regression (SBFR)

Synergistic Basis Functions Regression (SBFR) is a non-evolutionary regression framework that constructs predictive models using symbolic basis functions generated through a genetic-programming-inspired tree construction strategy. Candidate basis functions are generated using the *Ramped Half-and-Half* (RHH) procedure, which combines the *full* and *grow* methods to produce expressions with diverse structures and depths.

Unlike conventional approaches that separate basis-function generation and selection, SBFR integrates both processes into an iterative candidate generation and selection procedure. At each step, a newly generated basis function is evaluated in the context of the currently selected feature vector based on two complementary criteria: its improvement when added to the feature vector and its synergistic effect when replacing an existing basis function. A candidate is retained only if it satisfies the predefined improvement criterion.

Let the set of basis functions selected at a given stage be represented by the feature vector

$$FV_m = \{f_1, f_2, \dots, f_m\},$$

where m denotes the number of selected basis functions. The value of m is dynamically determined by the SBFR selection process based on the synergy and improvement criteria, subject to a predefined maximum of 30 basis functions. Thus, the final number of selected basis functions is not fixed in advance and may vary across datasets and regression models. Applying these functions to the input data X produces the transformed feature matrix

$$Z = FV_m(X) = [f_1(X), f_2(X), \dots, f_m(X)].$$

The overall objective of SBFR is to construct a compact basis-function representation that provides low prediction error for a downstream regression model:

$$FV^* = \arg \min_{FV} \text{RMSE}(f(FV(X)), Y). \quad (1)$$

where f denotes the downstream regression model, X is the input data, and Y is the target vector. Rather than exhaustively searching the symbolic expression space, SBFR approximates this objective by iteratively generating and evaluating candidate basis functions using RHH.

The selection process employs two complementary criteria: *Synergy by Adding* (S_{Add}) and *Synergy by Replacing* (S_{Rep}). S_{Add} measures the improvement obtained by adding a candidate to the current feature vector, whereas S_{Rep} evaluates the improvement obtained by replacing an existing basis function with the candidate. Accordingly,

SBFR can either expand the feature vector or revise its composition when a more effective candidate is identified.

SBFR is model-agnostic: the constructed basis functions can be evaluated using different downstream regression models, including linear regression, support vector regression (SVR), and k -nearest neighbors (KNN). This separation allows the same symbolic construction mechanism to be evaluated independently of the choice of predictive model.

Figure 1 illustrates the overall SBFR workflow, including RHH-based candidate generation, synergy evaluation, candidate acceptance or rejection, iterative refinement of the selected basis functions, and final model training.

3.1 SBFR Algorithm

Algorithm 1 summarizes the main procedure of SBFR. The algorithm starts with an empty feature vector and iteratively constructs a set of synergistic symbolic basis functions. At each iteration, a candidate basis function is generated using the RHH strategy and evaluated with respect to the current feature vector using the two proposed synergy measures: Synergy by Adding and Synergy by Replacing. If the current feature vector contains m selected basis functions, the candidate requires one evaluation for Synergy by Adding and up to m evaluations for Synergy by Replacing, corresponding to the possible replacement of each existing basis function.

Based on the computed synergy values, the feature vector is updated according to the predefined improvement threshold. The candidate is added when its Synergy by Adding is positive and no replacement provides a greater improvement. Otherwise, an existing basis function is replaced when the best Synergy by Replacing is positive and exceeds Synergy by Adding. If neither operation satisfies the acceptance criterion, the candidate is discarded and the current feature vector remains unchanged.

The search continues until a predefined stopping condition is satisfied, such as reaching the computational budget, observing no improvement, or achieving the target prediction error ($RMSE = 0$). The final output is a compact set of synergistic basis functions for the downstream regression model.

3.2 Initialization

SBFR starts with an empty feature vector (FV) containing no basis functions. During the iterative search, candidate basis functions are generated and evaluated with respect to the current feature vector. A candidate is incorporated into FV only when it satisfies the predefined improvement criterion.

3.3 Candidate Generation and Evaluation

Candidate basis functions are generated using the RHH strategy, which combines the *full* and *grow* methods to produce symbolic expressions with diverse structures and depths from the predefined function and terminal sets. Each candidate represents a potential transformation of the input variables. The detailed RHH procedure is presented in Algorithm 3 in Appendix A.

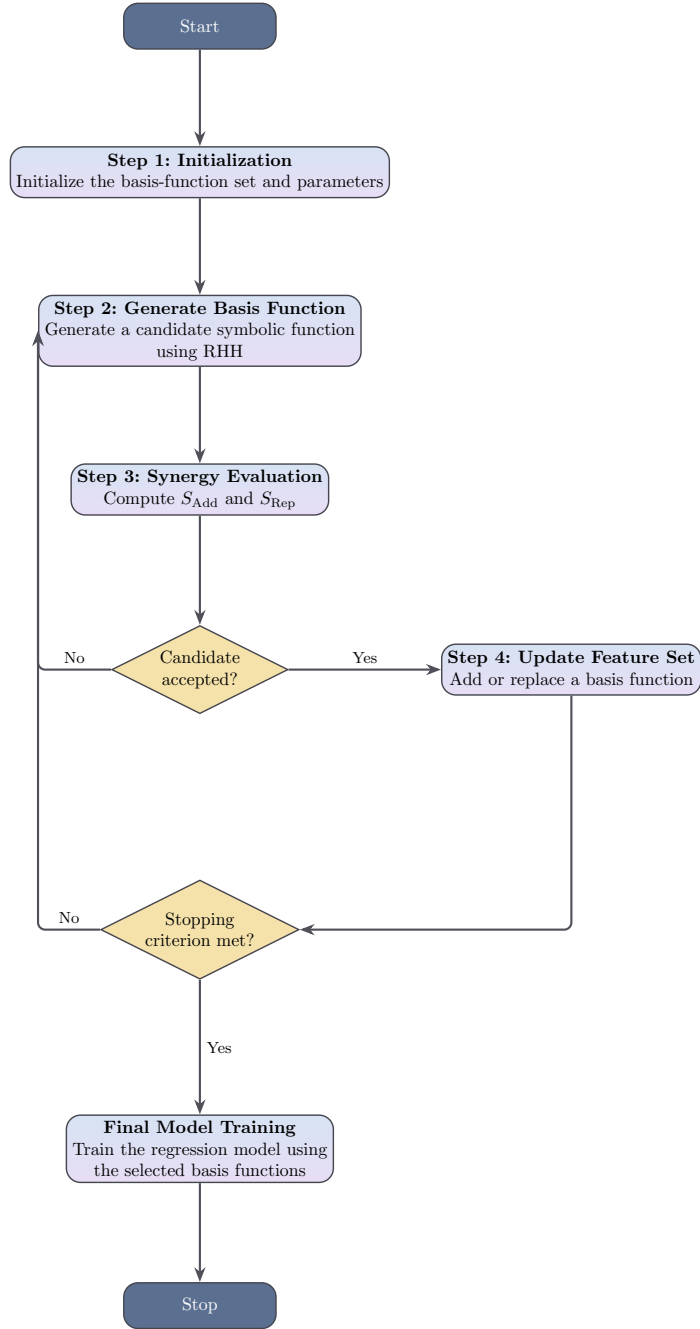


Fig. 1: Flowchart of the SBFR algorithm.

Each candidate is evaluated using two complementary criteria: *Synergy by Adding* (S_{Add}) and *Synergy by Replacing* (S_{Rep}). S_{Add} measures the improvement obtained

Algorithm 1 Synergistic Basis Function Regression (SBFR)

Require: Training dataset X , target values Y , function set FS , terminal set TS , depth range $[d_{\min}, d_{\max}]$, evaluation function F , maximum number of synergy evaluations Max_{attempts} , maximum number of basis functions Max_{basis} , improvement threshold $improve_{\text{threshold}}$, convergence limit $Max_{\text{no_improve}}$

Ensure: Final feature vector FV

```
1:  $FV \leftarrow \emptyset$ 
2:  $attempts \leftarrow 0$ 
3:  $no\_improve \leftarrow 0$ 
4:  $RMSE \leftarrow \infty$ 
5: while  $attempts < Max_{\text{attempts}}$  and  $no\_improve < Max_{\text{no\_improve}}$  and  $RMSE > 0$  do
6:    $f \leftarrow \text{RHH}(FS, TS, [d_{\min}, d_{\max}])$ 
7:    $S_{\text{Add}} \leftarrow \text{SynergyByAdding}(f, FV, X, Y, F)$ 
8:    $attempts \leftarrow attempts + 1$ 
9:   if  $attempts < Max_{\text{attempts}}$  then
10:     $S_{\text{Rep}} \leftarrow \text{SynergyByReplacing}(f, FV, X, Y, F)$ 
11:     $attempts \leftarrow attempts + |FV|$ 
12:   end if
13:    $(FV, Best_{\text{synergy}}) \leftarrow \text{UpdatingBasisFunctionVector}(f, FV, S_{\text{Rep}}, S_{\text{Add}}, improve_{\text{threshold}}, Max_{\text{basis}})$ 
14:   if  $Best_{\text{synergy}} > 0$  then
15:      $no\_improve \leftarrow 0$ 
16:   else
17:      $no\_improve \leftarrow no\_improve + 1$ 
18:   end if
19:    $RMSE \leftarrow F(FV, X, Y)$ 
20: end while
21: return  $FV$ 
```

by adding the candidate to the current feature vector, whereas S_{Rep} measures the improvement obtained by replacing each existing basis function with the candidate. Thus, for a feature vector containing m basis functions, one additional feature-vector evaluation is required for S_{Add} and up to m evaluations for S_{Rep} .

The predictive performance of each resulting feature representation is measured using the cross-validated RMSE of the downstream regression model. Based on the resulting synergy values, the candidate is either added, used to replace an existing basis function, or discarded when the improvement criterion is not satisfied. The detailed evaluation procedure is provided in Algorithm 4 in Appendix A.

3.4 Synergy-Based Evaluation and Selection

SBFR evaluates each candidate basis function according to its contribution to the predictive performance of the current feature representation. This contribution is quantified using two complementary measures: *Synergy by Adding* (S_{Add}) and *Synergy*

by *Replacing* (S_{Rep}). Both measures quantify the relative change in the cross-validated RMSE resulting from incorporating the candidate into the current feature vector (FV).

- **Synergy by Adding** (S_{Add}) measures the relative reduction in RMSE obtained by adding the candidate to the current FV .
- **Synergy by Replacing** (S_{Rep}) measures the relative reduction in RMSE obtained by replacing each existing basis function with the candidate.

For both measures, a positive synergy indicates a reduction in prediction error. SBFR uses a predefined improvement threshold, $improve_{\text{threshold}}$, to prevent acceptance based on marginal improvements. A candidate modification is therefore considered beneficial only when its synergy exceeds this threshold. The resulting selection process either expands the feature vector by adding the candidate or improves its composition by replacing an existing basis function.

3.4.1 Synergy by Adding (S_{Add})

Synergy by Adding evaluates the improvement obtained by appending a candidate basis function f to the current feature vector FV . The synergy score is defined as the relative reduction in RMSE:

$$S_{\text{Add}}(f, FV) = \frac{RMSE(FV) - RMSE(FV \cup f)}{RMSE(FV)} \times 100. \quad (2)$$

A candidate is considered for addition when

$$S_{\text{Add}}(f, FV) > improve_{\text{threshold}}. \quad (3)$$

However, addition is selected only when no valid replacement provides a greater improvement. Thus, SBFR compares the adding and replacement alternatives before updating the feature vector. The detailed procedure for computing the cross-validated RMSE and S_{Add} is given in Algorithm 5 in Appendix A.

3.4.2 Synergy by Replacing (S_{Rep})

Synergy by Replacing evaluates whether a candidate basis function f can improve the current representation by replacing an existing basis function. If FV contains m basis functions, up to m replacement scenarios are evaluated. For the i -th replacement, the synergy score is defined as

$$S_{\text{Rep}}[i] = - \left(\frac{RMSE(FV_{\text{temp},i}) - RMSE(FV)}{RMSE(FV)} \times 100 \right), \quad (4)$$

where $FV_{\text{temp},i}$ denotes the temporary feature vector obtained by replacing the i -th basis function in FV with the candidate f . The negative sign ensures that a reduction in RMSE corresponds to a positive synergy value.

The best replacement is determined as

$$S_{\text{Rep}}^{\max} = \max_i S_{\text{Rep}}[i]. \quad (5)$$

A replacement is selected when

$$S_{\text{Rep}}^{\text{max}} > \text{improve}_{\text{threshold}} \quad \text{and} \quad S_{\text{Rep}}^{\text{max}} > S_{\text{Add}}. \quad (6)$$

Otherwise, if S_{Add} exceeds the improvement threshold, the candidate is added to FV ; if neither condition is satisfied, the candidate is discarded. When the maximum number of basis functions is reached, only replacement is permitted. The detailed procedure for computing the cross-validated RMSE and S_{Rep} is given in Algorithm 6 in Appendix A.

Figure 2 illustrates the construction of the temporary feature vectors used to evaluate the adding and replacing alternatives for a candidate basis function.

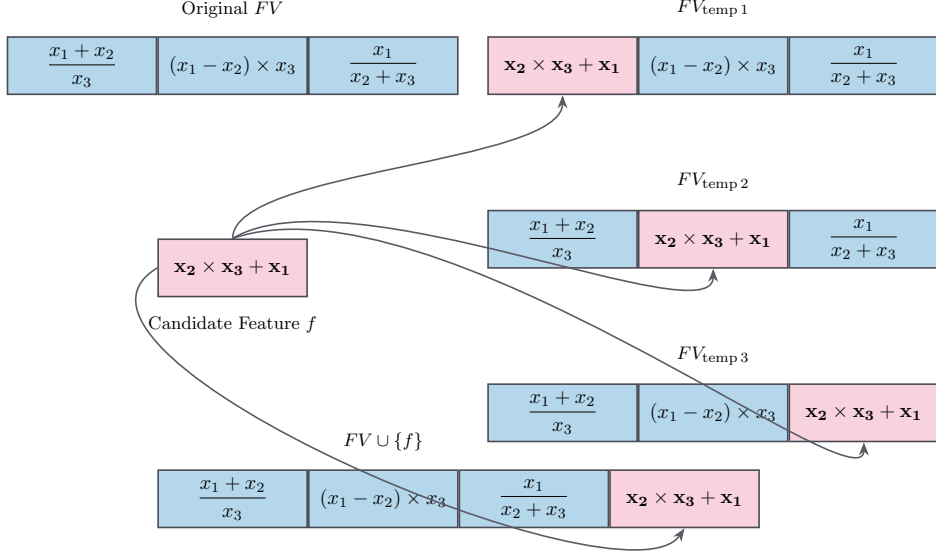


Fig. 2: Construction of temporary feature vectors for evaluating synergy by replacing and synergy by adding. The candidate feature is highlighted in pink, while the original basis functions are shown in blue.

3.5 Update Feature List

Algorithm 2 determines whether a candidate basis function f is added to or replaces a function in the current feature vector (FV). To enforce the predefined improvement threshold, $\text{improve}_{\text{threshold}}$ is subtracted from the addition and replacement synergy scores. Consequently, only candidates whose original synergy exceeds the threshold can obtain a positive adjusted score.

The algorithm compares the maximum adjusted replacement synergy, $\max_{\text{rep}}$, with the adjusted addition synergy. If $\max_{\text{rep}} > 0$ and $\max_{\text{rep}} > S_{\text{Add}}$, the corresponding

basis function is replaced by f . Otherwise, if $S_{\text{Add}} > 0$ and $|FV| < \text{Max}_{\text{basis}}$, f is appended to FV . If neither condition is satisfied, the candidate is discarded. Thus, the number of selected basis functions remains bounded by $\text{Max}_{\text{basis}}$.

Algorithm 2 Updating Basis Function Vector

Require: Candidate function f , feature vector FV , synergy by adding S_{Add} , synergy by replacing vector S_{Rep} , improvement threshold $\text{improve}_{\text{threshold}}$, maximum number of basis functions $\text{Max}_{\text{basis}}$

Ensure: Updated feature vector FV and best adjusted synergy

```

1: for  $i = 1$  to  $|S_{\text{Rep}}|$  do
2:    $S_{\text{Rep}}[i] \leftarrow S_{\text{Rep}}[i] - \text{improve}_{\text{threshold}}$ 
3: end for
4:  $S_{\text{Add}} \leftarrow S_{\text{Add}} - \text{improve}_{\text{threshold}}$ 
5:  $\text{max}_{\text{rep}} \leftarrow \max(S_{\text{Rep}})$ 
6: if  $\text{max}_{\text{rep}} > 0$  and  $\text{max}_{\text{rep}} > S_{\text{Add}}$  then
7:    $j^* \leftarrow \arg \max_j S_{\text{Rep}}[j]$ 
8:    $FV[j^*] \leftarrow f$ 
9:    $\text{best}_{\text{synergy}} \leftarrow \text{max}_{\text{rep}}$ 
10: else if  $S_{\text{Add}} > 0$  and  $|FV| < \text{Max}_{\text{basis}}$  then
11:    $FV \leftarrow FV \cup \{f\}$ 
12:    $\text{best}_{\text{synergy}} \leftarrow S_{\text{Add}}$ 
13: else
14:   Discard  $f$ 
15:    $\text{best}_{\text{synergy}} \leftarrow 0$ 
16: end if
17: return  $FV, \text{best}_{\text{synergy}}$ 

```

3.6 Stopping Condition

The iterative search terminates when one of the following conditions is satisfied:

- **Maximum attempts:** the maximum number of synergy evaluations, $\text{Max}_{\text{attempts}}$, is reached.
- **Convergence:** no sufficient improvement is observed over $\text{Max}_{\text{no_improve}}$ consecutive candidate evaluations.
- **Target performance:** the regression RMSE reaches zero.

After termination, the final feature vector FV is used to construct the feature matrix for training the final regression model.

3.7 Train Regression Model

During the search, temporary regression models are trained to evaluate the cross-validated RMSE of candidate feature representations. These models are used only for

candidate evaluation. After termination, the selected basis functions in FV are applied to the input data X to construct the final feature matrix:

$$Z = FV(X). \quad (7)$$

A final regression model is then trained using Z and the target vector Y . The resulting model, together with the selected basis functions in FV , constitutes the final output of SBFR.

4 Experimental Analysis

This section evaluates SBFR on nine benchmark regression datasets and compares it with 13 baseline methods in terms of predictive performance and model complexity. The experimental setup, parameter settings, and evaluation criteria are first described, followed by the experimental results and an illustrative SBFR solution.

4.1 Baseline Methods

To evaluate the effectiveness of the proposed SBFR framework, thirteen baseline methods are considered. The comparison includes conventional machine learning regression methods, namely Decision Tree (DT) [Breiman et al. \(1984\)](#), K-Nearest Neighbors (KNN) [Altman \(1992\)](#), Support Vector Regression (SVR) [Drucker et al. \(1997\)](#), Multi-Layer Perceptron (MLP) [Bishop \(1995\)](#), Random Forest (RF) [Breiman \(2001\)](#), Ridge Regression [Hoerl and Kennard \(1970\)](#), Gradient Boosted Decision Trees (GBDT) [Friedman \(2001\)](#), and XGBoost [Chen and Guestrin \(2016\)](#).

In addition, five SR approaches are included as competitors: gplearn(GPL), Geometric Semantic GP (GSGP), M3GP, Evolutionary Forest(EvoF), and Fast Function Extraction (FFX). Detailed descriptions of all baseline methods are provided in [Appendix B](#).

4.2 Datasets

The proposed SBFR framework is evaluated on nine SR benchmarks, including both synthetic and real-world datasets. The synthetic benchmarks comprise *bm1*, *bm2*, *keijzer5*, and *X05Y1Z15*, which are widely used in SR and genetic programming research [Keijzer \(2003\)](#); [Pennachin et al. \(2010\)](#); [Astarabadi and Ebadzadeh \(2019\)](#). The real-world datasets include *Concrete Compressive Strength* from the UCI Machine Learning Repository [Lichman \(2013\)](#), *Pollen* from the StatLib repository [StatLib \(2025\)](#), *RatPol2D*, a benchmark commonly used in SR studies [Amir Haeri et al. \(2015\)](#); [Amini et al. \(2020\)](#); [Zojaji and Ebadzadeh \(2018\)](#), the high-dimensional *Toxicity (LD50)* dataset from cheminformatics research [Haeri et al. \(2017\)](#); [Castelli et al. \(2011\)](#), and the *Real Estate Valuation* dataset for property price prediction.

These datasets provide diverse characteristics in terms of sample size, input dimensionality, and functional complexity. A summary is presented in [Table 3](#), while detailed dataset descriptions and underlying functions are provided in [Appendix C](#).

4.3 Tools, Parameters, and Settings

The SBFR framework was implemented in Python using Scikit-Learn (version 1.1.2) for the downstream regression models Pedregosa et al. (2011). The source code and implementation details are publicly available on GitHub¹. The parameter settings of SBFR are summarized in Table 3, while the default Scikit-Learn parameters were used unless otherwise specified.

Since SBFR constructs and selects basis functions rather than directly defining a regression model, the resulting feature representation was evaluated using six downstream regressors: Ridge Regression, Support Vector Regression (SVR), K-Nearest Neighbors (KNN), Decision Tree (DT), Random Forest (RF), and Multi-Layer Perceptron (MLP). The same selected basis functions were provided to each regressor to assess the quality of the constructed representation independently of the downstream learning algorithm.

To ensure consistency and computational efficiency, the number of trees in RF was set to 5, the maximum number of iterations for Ridge, SVR, and MLP to 1000, and the number of neighbors in KNN to $K = 3$. The maximum depth of DT was set to 6. SVR used a linear kernel with $C = 1$. These settings were fixed across all datasets.

Table 3: Parameter settings for the SBFR method.

Parameter	Value / Description
Maximum number of basis functions (Max_{basis})	30
Candidate generation	Ramped Half-and-Half (RHH)
Convergence criterion	No sufficient improvement over 100 consecutive candidate evaluations
Maximum number of synergy evaluations	50,000 (safeguard limit)
Function set	(+, −, ×, ÷) with protected division
Terminal set	Input variables and ephemeral random constants sampled from [0, 1]
Improvement threshold ($improve_{\text{threshold}}$)	1%
Number of independent trials	30

4.4 Results and Discussion

Performance is evaluated using R^2 , MAE, and RMSE. R^2 measures the explained variance, whereas MAE and RMSE quantify prediction error, with RMSE placing greater emphasis on larger errors.

Statistical significance is assessed using the non-parametric Friedman test at $\alpha = 0.05$ Chapman (2012); Demšar (2006). When the null hypothesis is rejected,

¹<https://github.com/hsarv/SBFR>

post-hoc comparisons are performed to determine which methods are statistically comparable to the best-performing approach. Highlighted entries in the result tables indicate statistically comparable methods.

All comparative results are reported as mean $\pm$ standard deviation over 30 independent runs. For presentation clarity, standard deviations are scaled by 10^{-2} . The following subsections present the detailed results for RMSE, MAE, and R^2 , together with the corresponding statistical analyses, focusing on predictive performance, stability, and consistency across the nine benchmark datasets.

4.4.1 Performance of SBFR-Based Methods versus Baseline Regressors

This section evaluates the predictive performance of the proposed SBFR-based configurations by comparing each configuration with its corresponding baseline regression model across the nine benchmark datasets.

Test-set RMSE Evaluation.

Table 4 reports the test-set RMSE (Mean $\pm$ StD) over 30 independent runs. Lower RMSE indicates better predictive performance. To assess the effect of SBFR-based feature construction, each baseline regressor is compared with its corresponding SBFR-enhanced configuration. The baseline models are trained using the original input features, whereas the SBFR-based configurations use the symbolic basis functions constructed by SBFR. The six paired configurations are **SBFR-DT**, **SBFR-KNN**, **SBFR-MLP**, **SBFR-RF**, **SBFR-Ridge**, and **SBFR-SVR**.

Figure 3 visualizes these paired comparisons across the nine datasets, highlighting the changes in test-set RMSE resulting from SBFR-based feature construction.

Additionally, Figure 4 illustrates the distribution of test-set RMSE values over the 30 independent runs for each evaluated method. The box plots complement the mean and standard deviation reported in Table 4 by showing the median, interquartile range, and overall variability of the results. Several methods exhibit substantial variability across runs and datasets, particularly some SR and neural-network-based approaches, indicating less consistent predictive performance. In contrast, narrower distributions indicate greater stability across independent runs. Overall, the boxplots provide a complementary assessment of the robustness and consistency of the evaluated methods beyond their average predictive performance.

Regression Method Rankings

For each dataset, the evaluated regression methods were ranked according to their test-set RMSE, with lower RMSE values receiving better ranks. Ties were assigned the corresponding average rank. The average rank of each method was then calculated across the nine benchmark datasets, providing a concise measure of overall predictive performance.

Figure 5 presents the average ranks based on test-set RMSE. Lower average ranks indicate better overall performance across the benchmark datasets. Among the evaluated methods, **EvoF** achieves the lowest average rank of 2.89, followed by **SBFR-RF** with an average rank of 3.78. The baseline RF and XGB methods obtain average ranks of

Table 4: Mean and standard deviation of test-set RMSE (Mean $\pm$ StD) over 30 independent runs. Standard deviation values are reported after scaling by 10^{-2} . Highlighted entries indicate methods whose performance is not significantly worse than the best-performing method for the corresponding dataset, based on the Friedman test ($\alpha = 0.05$) followed by post-hoc analysis.

Methods	Concrete	Pollen	RatPol2D	Real Estate	Toxicity	X05Y1Z15	bm1	bm4	keijzer5
DT	97.9 $\pm$ 724.8	767.6 $\pm$ 1908.7	57.2 $\pm$ 834.4	47.2 $\pm$ 318.0	66.8 $\pm$ 364.3	38.4 $\pm$ 788.0	0.8 $\pm$ 6.8	6.3 $\pm$ 45.3	31.5 $\pm$ 273.8
EvoF	69.0 $\pm$ 432.6	546.0 $\pm$ 973.1	16.9 $\pm$ 414.4	36.2 $\pm$ 291.5	50.4 $\pm$ 272.4	23.5 $\pm$ 715.4	0.2 $\pm$ 4.2	1.2 $\pm$ 47.8	12.2 $\pm$ 198.9
GBDT	69.4 $\pm$ 405.9	543.6 $\pm$ 651.3	52.1 $\pm$ 472.1	36.7 $\pm$ 214.5	54.0 $\pm$ 283.2	30.4 $\pm$ 319.4	0.8 $\pm$ 6.8	8.6 $\pm$ 45.7	56.4 $\pm$ 619.7
GPL	122.2 $\pm$ 489.3	1027.7 $\pm$ 31538.3	237.1 $\pm$ 5686.0	47.9 $\pm$ 488.2	53.2 $\pm$ 416.5	88.2 $\pm$ 1605.5	0.0 $\pm$ 0.3	55.8 $\pm$ 956.2	196.3 $\pm$ 4051.5
KNN	134.7 $\pm$ 615.7	621.2 $\pm$ 1152.2	34.1 $\pm$ 513.5	45.6 $\pm$ 333.8	52.5 $\pm$ 259.2	60.4 $\pm$ 540.2	0.7 $\pm$ 12.5	6.8 $\pm$ 24.0	57.9 $\pm$ 457.2
MLP	95.5 $\pm$ 379.0	604.7 $\pm$ 866.3	370.7 $\pm$ 897.7	42.6 $\pm$ 289.1	568.5 $\pm$ 49004.2	140.7 $\pm$ 748.0	0.0 $\pm$ 0.3	56.0 $\pm$ 118.3	555.8 $\pm$ 696.7
RF	74.9 $\pm$ 543.2	558.3 $\pm$ 888.4	37.8 $\pm$ 743.6	35.3 $\pm$ 228.0	49.8 $\pm$ 184.9	29.8 $\pm$ 495.7	0.6 $\pm$ 10.5	3.5 $\pm$ 29.7	23.7 $\pm$ 205.2
Ridge	152.5 $\pm$ 457.7	542.5 $\pm$ 701.6	350.5 $\pm$ 416.5	46.7 $\pm$ 297.3	212.3 $\pm$ 10822.4	96.6 $\pm$ 395.7	0.0 $\pm$ 0.0	118.6 $\pm$ 120.5	574.9 $\pm$ 668.2
SVR	237.7 $\pm$ 510.6	1027.3 $\pm$ 1199.3	311.6 $\pm$ 450.1	58.4 $\pm$ 205.2	48.0 $\pm$ 151.9	274.6 $\pm$ 805.7	13.7 $\pm$ 191.2	71.1 $\pm$ 170.7	547.9 $\pm$ 734.0
XGB	68.3 $\pm$ 496.3	601.3 $\pm$ 791.5	37.0 $\pm$ 746.0	38.2 $\pm$ 205.5	53.4 $\pm$ 295.4	25.5 $\pm$ 686.0	1.1 $\pm$ 14.9	4.7 $\pm$ 34.7	15.8 $\pm$ 166.3
FFX	89.7 $\pm$ 631.2	551.5 $\pm$ 2548.2	62.7 $\pm$ 363.5	46.5 $\pm$ 1155.7	121.8 $\pm$ 2900.5	53.3 $\pm$ 756.3	1.2 $\pm$ 9.5	8.3 $\pm$ 39.9	317.2 $\pm$ 13442.8
GSGP	189.7 $\pm$ 963.2	2051.8 $\pm$ 3977.6	532.2 $\pm$ 2878.6	56.6 $\pm$ 433.9	51.4 $\pm$ 489.1	392.1 $\pm$ 5358.5	0.0 $\pm$ 0.0	689.7 $\pm$ 188352.8	1100.9 $\pm$ 2506.4
M3GP	110.0 $\pm$ 880.5	558.2 $\pm$ 1538.2	144.6 $\pm$ 1249.2	49.6 $\pm$ 340.6	63.0 $\pm$ 331.3	27.5 $\pm$ 1274.6	1.1 $\pm$ 7.7	12.5 $\pm$ 767.4	10.3 $\pm$ 37.9
SBFR-DT	103.6 $\pm$ 833.3	616.5 $\pm$ 1764.1	66.6 $\pm$ 1022.5	49.5 $\pm$ 399.6	65.5 $\pm$ 445.1	52.0 $\pm$ 696.5	1.1 $\pm$ 8.1	9.5 $\pm$ 81.0	24.6 $\pm$ 883.8
SBFR-KNN	90.7 $\pm$ 630.5	578.6 $\pm$ 1549.8	32.8 $\pm$ 517.3	40.9 $\pm$ 301.5	53.9 $\pm$ 269.6	34.2 $\pm$ 850.7	0.7 $\pm$ 11.1	5.9 $\pm$ 25.7	14.7 $\pm$ 865.6
SBFR-MLP	87.9 $\pm$ 1450.2	557.7 $\pm$ 0.0	78.8 $\pm$ 611.8	51.9 $\pm$ 1234.0	100.1 $\pm$ 6735.0	36.6 $\pm$ 1078.8	0.0 $\pm$ 0.3	4.0 $\pm$ 0.0	61.0 $\pm$ 1127.0
SBFR-RF	74.7 $\pm$ 521.8	554.9 $\pm$ 1271.4	23.8 $\pm$ 520.1	36.8 $\pm$ 286.9	52.0 $\pm$ 370.6	24.1 $\pm$ 592.2	0.5 $\pm$ 13.1	0.7 $\pm$ 15.8	7.3 $\pm$ 238.0
SBFR-Ridge	92.7 $\pm$ 1256.3	535.7 $\pm$ 1654.4	221.7 $\pm$ 17366.0	56.3 $\pm$ 1392.7	183.1 $\pm$ 12543.4	50.6 $\pm$ 1808.0	0.0 $\pm$ 0.0	43.7 $\pm$ 3613.7	127.5 $\pm$ 5389.4
SBFR-SVR	185.2 $\pm$ 1484.1	876.9 $\pm$ 5574.4	269.2 $\pm$ 551.4	49.8 $\pm$ 374.5	47.7 $\pm$ 134.7	198.9 $\pm$ 1061.6	11.4 $\pm$ 131.9	37.4 $\pm$ 87.6	396.8 $\pm$ 1675.2

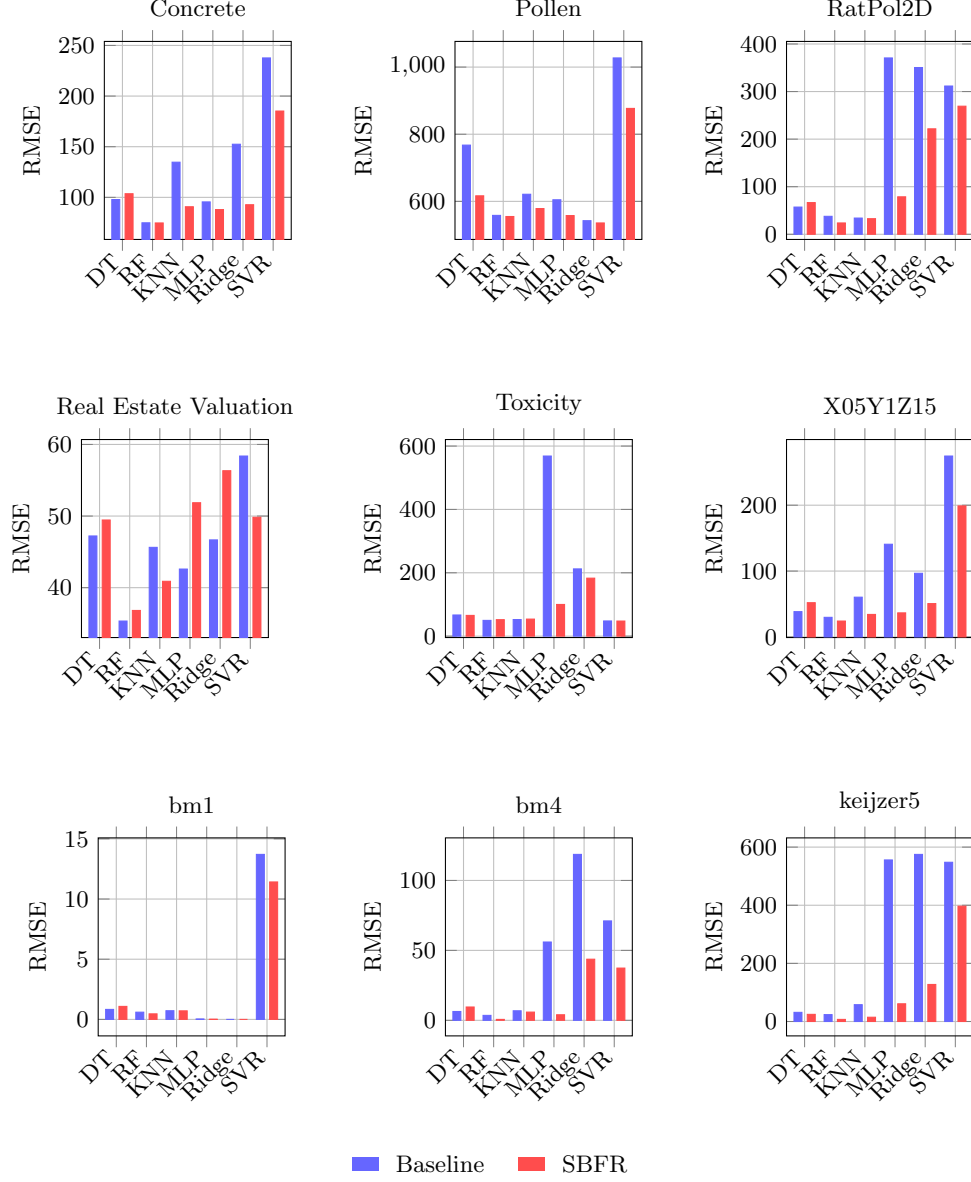


Fig. 3: Comparison of the test-set RMSE of the baseline regressors and their SBFR-enhanced versions across the benchmark datasets. Lower RMSE values indicate better performance.

5.22 and 6.56, respectively. Notably, SBFR-RF achieves a substantially better average rank than the standalone RF model, indicating that the basis functions constructed by SBFR can provide a more effective representation for RF on the evaluated datasets.

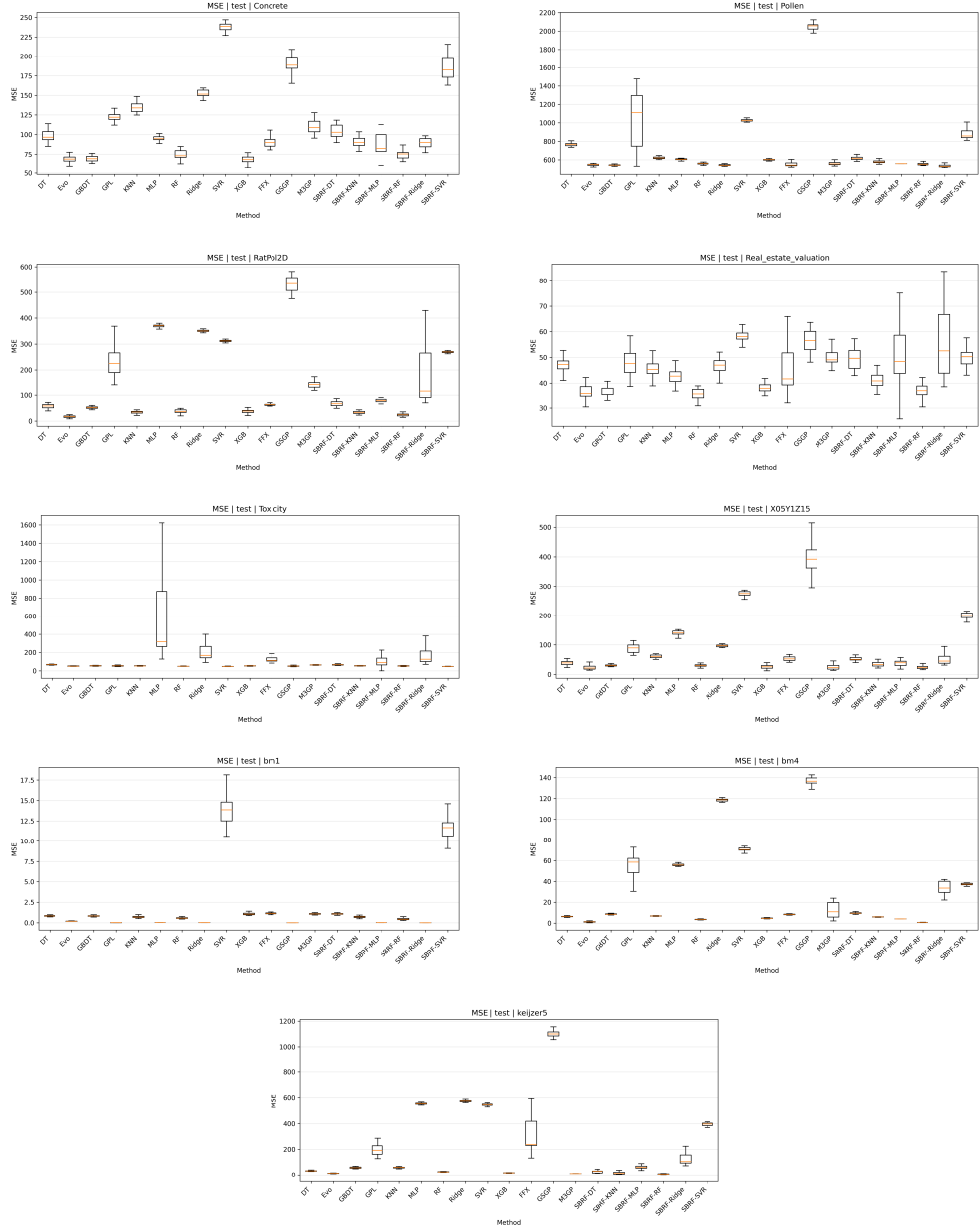


Fig. 4: Boxplots of test-set RMSE values obtained by evaluated regression methods across the benchmark regression datasets.

The effect of SBFR varies across the downstream regression algorithms. SBFR-KNN achieves an average rank of 7.11 compared with 10.00 for KNN, while SBFR-MLP

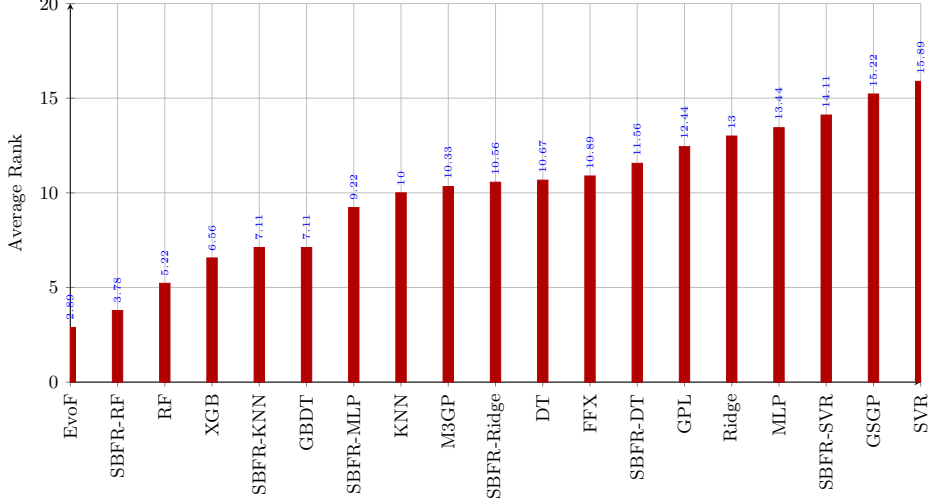


Fig. 5: Average rank of the evaluated methods based on test-set RMSE across the nine benchmark datasets. Lower average rank indicates better predictive performance.

improves the average rank from 13.44 to 9.22. Similarly, **SBFR-Ridge** improves the rank from 13.00 to 10.56, and **SBFR-SVR** improves it from 15.89 to 14.11. In contrast, **SBFR-DT** obtains an average rank of 11.56, which is slightly worse than the rank of the standalone DT model (10.67). Overall, SBFR improves the average rank of five out of the six considered downstream regression models, with the largest improvement observed for MLP and KNN.

These results indicate that the effectiveness of SBFR-based feature construction depends on the downstream regression model. The particularly strong ranking obtained by **SBFR-RF** suggests that the synergistic basis functions provide a useful representation for ensemble-based regression on these benchmarks, whereas the comparatively weaker result for **SBFR-DT** shows that the benefits of the constructed representation are not uniform across all learning algorithms.

Statistical Significance of Test Performance

To assess the statistical significance of the differences in test-set RMSE, post-hoc pairwise comparisons were performed using **SBFR-RF** as the control method. Holm and Finner corrections were applied to account for multiple comparisons. Table 5 reports the unadjusted and adjusted P -values. Values below the significance level of $\alpha = 0.05$ are highlighted in red.

The results indicate that the test-set RMSE of **SBFR-RF** differs significantly from that of most of the evaluated methods. After correction for multiple comparisons, statistically significant differences ($p < 0.05$) are observed for DT, GBDT, GPL, KNN, MLP, RF, Ridge, SVR, XGB, FFX, GSGP, M3GP, and all the other evaluated SBFR-based configurations. In contrast, the comparison between **SBFR-RF** and EvoF does not reach statistical significance, with an adjusted p -value of 0.12 under both the Holm

and Finner procedures. Thus, at the $\alpha = 0.05$ significance level, there is insufficient evidence to conclude that the test-set RMSE distributions of EvoF and SBFR-RF differ significantly.

The ranking results are consistent with this statistical analysis. Although EvoF achieves the best average rank of 2.89, SBFR-RF obtains the second-best average rank of 3.78. Moreover, SBFR-RF ranks better than the standalone RF model, which has an average rank of 5.22. These results indicate that SBFR-RF provides a strong and competitive overall performance across the evaluated datasets, while its test-set RMSE is not statistically distinguishable from that of the best-ranked EvoF method.

Overall, these findings demonstrate that SBFR-RF provides highly competitive predictive performance across the nine benchmark datasets. Although EvoF achieves the best average rank, its test-set RMSE is not statistically different from that of SBFR-RF. Furthermore, the comparison with the standalone RF model is statistically significant after multiple-comparison correction, with $p_{\text{Holm}} = 0.03$ and $p_{\text{Finner}} = 0.01$. This result provides statistical evidence that incorporating SBFR-based feature construction can improve the test-set RMSE performance of the Random Forest model. More broadly, the significant differences observed for SBFR-RF against several baseline and SBFR-based configurations support the effectiveness of the proposed feature-construction framework when combined with downstream regression models.

Table 5: Post-hoc comparison of SBFR-RF with the evaluated regression methods based on test-set RMSE using Holm and Finner corrections. P-values below $\alpha = 0.05$ are highlighted in red.

Algorithm	Unadjusted P	pHolm	pFinner
DT	0.00	0.00	0.00
EvoF	0.12	0.12	0.12
GBDT	0.00	0.00	0.00
GPL	0.00	0.00	0.00
KNN	0.00	0.00	0.00
MLP	0.00	0.00	0.00
RF	0.01	0.03	0.01
Ridge	0.00	0.00	0.00
SVR	0.00	0.00	0.00
XGB	0.00	0.00	0.00
FFX	0.00	0.00	0.00
GSGP	0.00	0.00	0.00
M3GP	0.00	0.00	0.00
SBFR-DT	0.00	0.00	0.00
SBFR-KNN	0.00	0.00	0.00
SBFR-MLP	0.00	0.00	0.00
SBFR-Ridge	0.00	0.00	0.00
SBFR-SVR	0.00	0.00	0.00

Performance on Additional Metrics

The performance of the SBFR-based configurations is further evaluated using the coefficient of determination (R^2) and mean absolute error (MAE), in addition to the RMSE

results discussed above. These complementary metrics provide a broader assessment of predictive performance. Higher R^2 values indicate better agreement between the predicted and observed target values, whereas lower MAE and RMSE values indicate smaller prediction errors.

The results reported in Tables 6 and 7 show that SBFR-based configurations can achieve competitive performance across multiple benchmark datasets. However, the magnitude and consistency of the observed differences vary across datasets and downstream regression models. This variation indicates that the predictive utility of the constructed symbolic representations depends on their interaction with the characteristics of the dataset and the underlying regression algorithm.

Figure 6 summarizes the overall performance of the evaluated methods using average ranks for test-set MAE and R^2 . For each metric, the methods are ranked on each benchmark dataset, and the resulting ranks are averaged across the nine datasets. For MAE, lower values correspond to better performance and therefore lower ranks. For R^2 , higher values correspond to better performance and are assigned lower ranks. Consequently, lower average ranks indicate better overall performance.

The average-rank results show that SBFR-RF achieves the second-best average rank for both MAE and R^2 , with ranks of 4.11 and 3.78, respectively. In both cases, it is preceded only by EvoF, which achieves average ranks of 2.44 for MAE and 2.89 for R^2 . Moreover, SBFR-RF ranks better than the standalone RF model for both metrics, with RF obtaining average ranks of 4.78 for MAE and 5.00 for R^2 . These results indicate that the SBFR-enhanced Random Forest configuration maintains a consistently strong position when predictive performance is assessed using metrics beyond RMSE.

The other SBFR-based configurations also exhibit competitive rankings. In particular, SBFR-KNN and SBFR-MLP obtain relatively favorable average ranks for both MAE and R^2 , although their performance varies across individual datasets. The dataset-level results further show that the best-performing configuration is not identical across all datasets, highlighting the interaction between symbolic feature construction and the downstream regression model.

Overall, the results across MAE and R^2 are broadly consistent with the RMSE analysis. The strong average ranks achieved by SBFR-RF across all three metrics indicate that the proposed SBFR-generated representations can provide a competitive basis for improving downstream regression performance. At the same time, the variation among regression models and datasets suggests that the benefits of symbolic feature construction are dependent on the characteristics of both the data and the downstream learner.

Performance Improvement Across Regression Metrics

Table 8 reports the relative performance change obtained by the SBFR-based regression configurations compared with their corresponding baseline regression models across the nine benchmark datasets. The analysis considers three complementary evaluation metrics: the coefficient of determination (R^2), root mean squared error (RMSE), and mean absolute error (MAE).

For each dataset and evaluation metric, the difference between an SBFR-based configuration and its corresponding baseline is normalized by the maximum absolute

Table 6: Mean and standard deviation of test-set MAE (Mean $\pm$ StD) over 30 independent runs. Highlighted entries indicate methods whose performance is not significantly worse than the best-performing method for the corresponding dataset, based on the Friedman test ($\alpha = 0.05$) followed by post-hoc analysis.

Methods	Concrete	Pollen	RatPol2D	Real Estate	Toxicity	X05Y1Z15	bm1	bm4	keijzer5
DT	63.0 $\pm$ 441.6	577.3 $\pm$ 1516.5	26.8 $\pm$ 245.6	33.9 $\pm$ 214.8	53.6 $\pm$ 367.0	15.1 $\pm$ 178.0	0.6 $\pm$ 4.9	3.5 $\pm$ 17.1	17.5 $\pm$ 119.3
EvoF	45.2 $\pm$ 301.4	416.2 $\pm$ 681.8	7.4 $\pm$ 130.4	26.2 $\pm$ 223.4	42.4 $\pm$ 260.3	8.2 $\pm$ 100.6	0.1 $\pm$ 3.2	0.5 $\pm$ 26.4	6.3 $\pm$ 75.3
GBDT	51.6 $\pm$ 287.8	424.6 $\pm$ 554.4	31.8 $\pm$ 296.5	26.3 $\pm$ 180.4	44.8 $\pm$ 282.4	16.9 $\pm$ 76.1	0.6 $\pm$ 4.8	5.6 $\pm$ 28.3	42.4 $\pm$ 460.1
GPL	96.7 $\pm$ 320.3	803.7 $\pm$ 23786.2	165.0 $\pm$ 2552.2	35.5 $\pm$ 330.9	42.6 $\pm$ 185.4	52.0 $\pm$ 797.8	0.0 $\pm$ 0.3	44.2 $\pm$ 786.7	147.5 $\pm$ 3053.2
KNN	104.5 $\pm$ 419.5	485.1 $\pm$ 792.7	18.1 $\pm$ 176.0	34.6 $\pm$ 221.0	44.5 $\pm$ 226.1	32.3 $\pm$ 197.9	0.5 $\pm$ 7.7	5.0 $\pm$ 16.1	37.9 $\pm$ 139.3
MLP	73.3 $\pm$ 272.4	482.5 $\pm$ 757.5	317.3 $\pm$ 910.9	31.6 $\pm$ 181.9	186.5 $\pm$ 12447.1	100.2 $\pm$ 425.1	0.0 $\pm$ 0.3	45.0 $\pm$ 90.1	481.0 $\pm$ 722.1
RF	52.8 $\pm$ 312.4	431.1 $\pm$ 607.2	16.4 $\pm$ 208.9	25.9 $\pm$ 176.5	42.3 $\pm$ 217.2	13.0 $\pm$ 123.8	0.5 $\pm$ 6.1	2.1 $\pm$ 11.2	15.3 $\pm$ 103.7
Ridge	124.8 $\pm$ 399.9	432.0 $\pm$ 678.6	307.1 $\pm$ 505.4	37.6 $\pm$ 215.3	104.9 $\pm$ 2204.8	71.4 $\pm$ 288.1	0.0 $\pm$ 0.0	102.8 $\pm$ 120.4	500.2 $\pm$ 752.1
SVR	204.4 $\pm$ 517.6	889.9 $\pm$ 1310.1	265.4 $\pm$ 485.1	48.1 $\pm$ 207.1	41.8 $\pm$ 181.6	232.0 $\pm$ 903.9	8.2 $\pm$ 125.8	58.4 $\pm$ 126.9	475.6 $\pm$ 762.6
XGB	46.0 $\pm$ 283.1	464.8 $\pm$ 722.8	14.7 $\pm$ 186.4	27.6 $\pm$ 162.2	44.3 $\pm$ 281.9	9.6 $\pm$ 111.5	0.8 $\pm$ 9.8	2.6 $\pm$ 14.1	10.0 $\pm$ 87.6
FFX	68.6 $\pm$ 310.2	421.0 $\pm$ 929.6	42.9 $\pm$ 326.7	30.8 $\pm$ 379.9	84.5 $\pm$ 1358.9	32.8 $\pm$ 222.0	0.9 $\pm$ 8.7	6.0 $\pm$ 33.2	260.0 $\pm$ 11577.7
GSGP	162.1 $\pm$ 797.2	1753.4 $\pm$ 4294.9	414.0 $\pm$ 3254.4	45.6 $\pm$ 386.1	42.6 $\pm$ 362.6	287.9 $\pm$ 4007.8	0.0 $\pm$ 0.0	148.6 $\pm$ 9350.2	923.0 $\pm$ 3031.1
M3GP	80.8 $\pm$ 646.7	422.0 $\pm$ 997.2	102.9 $\pm$ 849.2	36.4 $\pm$ 358.0	51.3 $\pm$ 328.6	17.2 $\pm$ 617.9	1.0 $\pm$ 6.3	9.0 $\pm$ 594.8	8.8 $\pm$ 32.1
SBFR-DT	74.8 $\pm$ 581.4	471.0 $\pm$ 1608.0	41.4 $\pm$ 417.5	35.4 $\pm$ 304.5	53.0 $\pm$ 432.2	27.4 $\pm$ 213.5	1.0 $\pm$ 6.7	6.9 $\pm$ 52.5	18.8 $\pm$ 643.0
SBFR-KNN	66.9 $\pm$ 500.5	439.2 $\pm$ 919.1	18.3 $\pm$ 242.5	30.0 $\pm$ 236.9	44.8 $\pm$ 280.4	17.9 $\pm$ 430.8	0.5 $\pm$ 7.2	4.3 $\pm$ 14.0	10.5 $\pm$ 589.3
SBFR-MLP	64.1 $\pm$ 790.4	419.4 $\pm$ 0.0	46.2 $\pm$ 815.0	35.1 $\pm$ 361.9	67.0 $\pm$ 1963.5	20.5 $\pm$ 103.1	0.0 $\pm$ 0.2	2.8 $\pm$ 0.0	41.9 $\pm$ 808.6
SBFR-RF	52.0 $\pm$ 322.2	425.6 $\pm$ 976.7	12.5 $\pm$ 135.6	26.9 $\pm$ 188.1	43.1 $\pm$ 339.9	10.5 $\pm$ 144.5	0.3 $\pm$ 7.3	0.3 $\pm$ 3.1	5.0 $\pm$ 136.1
SBFR-Ridge	66.1 $\pm$ 372.0	412.7 $\pm$ 1163.3	74.6 $\pm$ 2215.3	36.7 $\pm$ 624.0	93.5 $\pm$ 3430.2	26.2 $\pm$ 347.9	0.0 $\pm$ 0.0	23.4 $\pm$ 526.1	68.9 $\pm$ 1211.6
SBFR-SVR	158.4 $\pm$ 1347.2	772.4 $\pm$ 6684.3	221.6 $\pm$ 621.0	39.9 $\pm$ 302.4	41.5 $\pm$ 178.1	155.7 $\pm$ 1029.6	7.5 $\pm$ 106.3	29.4 $\pm$ 109.0	324.4 $\pm$ 1115.1

Table 7: Mean and standard deviation of test-set R^2 (Mean $\pm$ StD) over 30 independent runs. Highlighted entries indicate methods whose performance is not significantly worse than the best-performing method for the corresponding dataset, based on the Friedman test ($\alpha = 0.05$) followed by post-hoc analysis.

Methods	Concrete	Pollen	RatPol2D	Real Estate	Toxicity	X05Y1Z15	bm1	bm4	keijzer5
DT	0.8 ± 2.6	0.5 ± 2.9	1.0 ± 0.8	0.6 ± 7.0	-1.0 ± 21.6	1.0 ± 0.5	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
EvoF	0.9 ± 1.3	0.7 ± 1.2	1.0 ± 0.1	0.8 ± 3.9	-0.2 ± 7.9	1.0 ± 0.2	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
GBDT	0.9 ± 1.2	0.8 ± 1.0	1.0 ± 0.4	0.8 ± 3.6	-0.3 ± 9.0	1.0 ± 0.2	1.0 ± 0.0	1.0 ± 0.1	1.0 ± 0.2
GPL	0.7 ± 2.8	0.0 ± 53.4	0.5 ± 24.3	0.6 ± 8.7	-0.3 ± 19.4	0.9 ± 2.8	1.0 ± 0.0	0.8 ± 7.3	0.9 ± 5.0
KNN	0.7 ± 2.6	0.7 ± 1.6	1.0 ± 0.3	0.6 ± 5.9	-0.2 ± 8.7	1.0 ± 0.5	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.2
MLP	0.8 ± 1.8	0.7 ± 1.2	-0.1 ± 6.8	0.7 ± 4.1	-189.8 ± 29351.2	0.8 ± 1.6	1.0 ± 0.0	0.8 ± 1.1	0.1 ± 1.5
RF	0.9 ± 1.7	0.7 ± 1.1	1.0 ± 0.5	0.8 ± 3.5	-0.1 ± 5.6	1.0 ± 0.2	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
Ridge	0.6 ± 3.1	0.7 ± 0.8	-0.0 ± 1.3	0.6 ± 4.9	-21.4 ± 2123.3	0.9 ± 0.7	1.0 ± 0.0	-0.0 ± 0.2	0.0 ± 1.2
SVR	0.0 ± 0.4	0.1 ± 0.3	0.2 ± 0.8	0.4 ± 3.4	-0.0 ± 3.6	0.4 ± 1.2	0.8 ± 3.4	0.6 ± 1.6	0.1 ± 0.2
XGB	0.9 ± 1.4	0.7 ± 1.3	1.0 ± 0.5	0.7 ± 3.7	-0.3 ± 11.3	1.0 ± 0.3	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
FFX	0.9 ± 2.3	0.7 ± 2.4	1.0 ± 0.4	0.6 ± 23.5	-6.2 ± 372.9	1.0 ± 0.6	1.0 ± 0.0	1.0 ± 0.1	0.6 ± 28.6
GSGP	0.4 ± 7.0	-2.6 ± 13.9	-1.3 ± 24.4	0.4 ± 9.7	-0.2 ± 15.6	-0.3 ± 34.2	1.0 ± 0.0	-264.4 ± 92263.8	-2.7 ± 19.1
M3GP	0.8 ± 3.6	0.7 ± 1.7	0.8 ± 3.4	0.5 ± 11.6	-0.8 ± 20.4	1.0 ± 0.6	1.0 ± 0.0	1.0 ± 1.5	1.0 ± 0.0
SBFR-DT	0.8 ± 3.4	0.7 ± 2.2	1.0 ± 1.2	0.5 ± 8.7	-1.0 ± 30.1	1.0 ± 0.7	1.0 ± 0.0	1.0 ± 0.1	1.0 ± 0.1
SBFR-KNN	0.9 ± 2.1	0.7 ± 1.5	1.0 ± 0.3	0.7 ± 4.7	-0.3 ± 12.0	1.0 ± 0.5	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.1
SBFR-MLP	0.9 ± 5.5	0.7 ± 0.0	0.9 ± 0.7	0.5 ± 26.0	-1.6 ± 215.0	1.0 ± 0.9	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.4
SBFR-RF	0.9 ± 1.3	0.7 ± 1.3	1.0 ± 0.2	0.7 ± 3.7	-0.2 ± 12.3	1.0 ± 0.3	1.0 ± 0.0	1.0 ± 0.0	1.0 ± 0.0
SBFR-Ridge	0.8 ± 4.5	0.8 ± 1.6	0.3 ± 101.0	0.4 ± 31.2	-21.2 ± 3227.9	1.0 ± 1.8	1.0 ± 0.0	0.8 ± 51.6	0.9 ± 5.0
SBFR-SVR	0.4 ± 9.2	0.3 ± 10.7	0.4 ± 1.3	0.5 ± 6.9	-0.0 ± 2.8	0.7 ± 2.6	0.9 ± 1.9	0.9 ± 0.8	0.5 ± 2.3

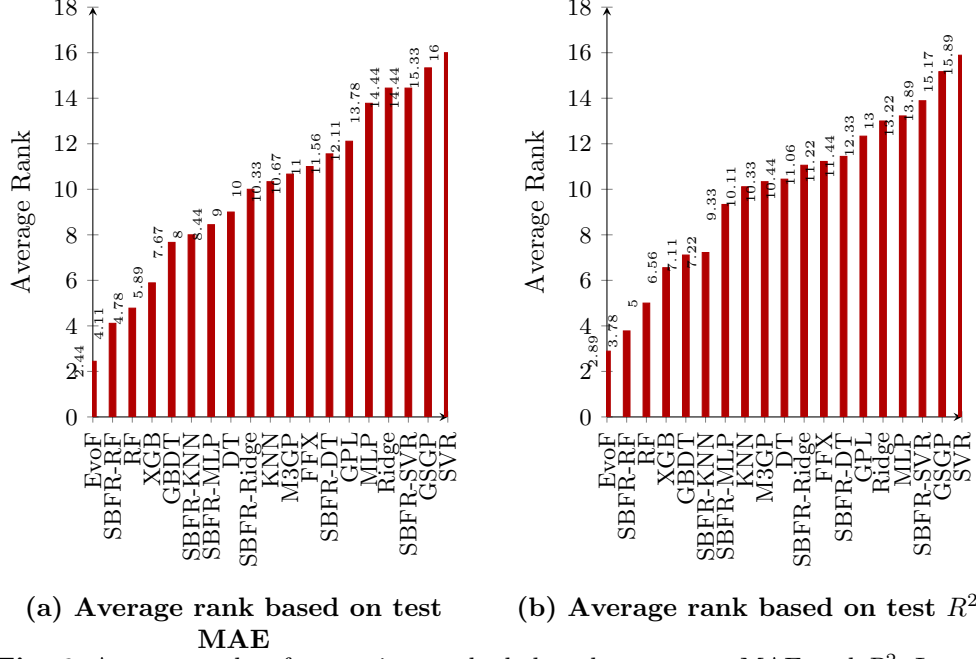


Fig. 6: Average ranks of regression methods based on test-set MAE and R^2 . Lower average ranks indicate better overall performance.

value of the two compared scores. This normalization provides a scale-independent measure of the relative change between the two configurations. For the error-based metrics (RMSE and MAE), positive values indicate a reduction in prediction error, whereas for R^2 , positive values indicate an increase in the coefficient of determination. Accordingly, positive values represent an improvement obtained by the SBFR-based configuration, while negative values indicate that the corresponding baseline performs better for the given dataset and metric.

The results demonstrate that the magnitude and direction of the observed changes vary across datasets and regression models. Nevertheless, the aggregated results are predominantly positive across the three evaluation metrics. This variation indicates that the impact of SBFR-based feature construction depends on the interaction between the generated symbolic representations, the underlying regression algorithm, and the characteristics of the dataset.

As shown in Table 8, the largest relative improvements are observed for the *RatPol2D*, *bm4*, and *keijzer5* datasets, particularly for the error-based metrics. For *bm4*, the reported relative improvements reach 90.33% for RMSE and 91.66% for MAE. On *RatPol2D*, the corresponding improvements are 63.25% and 62.92%, respectively, while the relative improvement in R^2 reaches 37.42%. Similarly, *keijzer5* exhibits improvements of 79.75% in RMSE and 79.23% in MAE.

Averaged across the nine benchmark datasets, the relative changes are 21.56% for R^2 , 40.14% for RMSE, and 41.53% for MAE. These values indicate that, on average,

the SBFR-based configurations are associated with higher R^2 and lower prediction errors than their corresponding baseline configurations. The larger average changes observed for RMSE and MAE further indicate that the effect of SBFR-based feature construction is particularly pronounced for the error-based metrics.

The relative change is positive for all dataset–metric combinations except for RMSE on *bm1*, where a small decrease of 2.39% is observed. Thus, the aggregated results show a generally favorable effect of SBFR-based feature construction across the three evaluation metrics, while also demonstrating that the magnitude of the effect varies across datasets.

Overall, these results provide quantitative evidence that SBFR-based feature construction can improve the predictive performance of the evaluated regression models across multiple metrics and benchmark datasets. The consistently positive average changes, particularly for RMSE and MAE, together with the substantial improvements observed on several datasets, support the utility of the generated symbolic representations as an effective feature-construction mechanism for downstream regression models.

Table 8: Percentage improvement achieved by SBFR-based regression configurations over the corresponding baseline regression models across the benchmark datasets. Positive values indicate improved performance, whereas negative values indicate degradation.

Dataset	R^2 (%)	RMSE (%)	MAE (%)
Concrete	19.03	26.33	30.00
Pollen	24.01	14.54	15.84
RatPol2D	37.42	63.25	62.92
Real Estate Valuation	21.74	18.91	19.35
Toxicity	29.18	21.73	18.21
X05Y1Z15	14.31	48.83	54.74
bm1	1.54	-2.39	1.77
bm4	17.70	90.33	91.66
keijzer5	29.07	79.75	79.23
Average	21.56	40.14	41.53

4.4.2 Dimension of Feature Space

The dimensionality of the resulting feature representation is an important indicator of the compactness and interpretability of SR models. A lower-dimensional representation can reduce computational and inference costs and may facilitate the interpretation of the resulting model. Table 9 reports the median and interquartile range (Q1, Q3) of the representation dimensionality obtained by the evaluated methods across the nine benchmark regression datasets. The comparison includes Evolutionary approaches (EvoF, FFX, GSGP, and M3GP) and the proposed SBFR variants (SBFR-DT, SBFR-KNN, SBFR-MLP, SBFR-RF, SBFR-Ridge, and SBFR-SVR).

Among the proposed SBFR variants, SBFR-SVR generally produces the most compact representations. It achieves the lowest median dimensionality among the SBFR variants on Concrete (8 [6,12]), Pollen (4 [2,5]), RatPol2D (2 [1,8]), Real estate (5 [3,7]), Toxicity (4 [2,5]), and keijzer5 (3 [1,6]). These results suggest that the choice of the internal regression evaluator can substantially influence the compactness of the representation produced by SBFR. In particular, the SVR-based evaluator frequently identifies a small number of basis functions that provide sufficient predictive utility.

Other SBFR variants also produce compact representations on several datasets. SBFR-RF achieves a median of 10 features on X05Y1Z15 and 7 features on bm4, while SBFR-KNN also obtains a median of 7 features on bm4. SBFR-MLP maintains relatively compact representations on Concrete (9 [9,11]) and Real estate (6 [6,8]). These results further indicate that the dimensionality of the resulting representation depends on both the regression evaluator used within SBFR and the characteristics of the dataset.

In contrast, EvoF and GSGP generally produce substantially larger feature representations. EvoF produces median dimensionalities ranging from 115 to 228 features across the benchmark datasets, whereas GSGP exceeds 600 features on eight of the nine datasets. On bm1, however, GSGP produces a median of only 1 feature. Such large representations can increase computational and model complexity and may make the resulting models more difficult to interpret. FFX exhibits considerably more variable behavior, ranging from a highly compact representation on bm1 (1 [0,2]) and keijzer5 (5 [0,26]) to a much larger representation on Concrete (268 [256,277]).

M3GP provides relatively compact representations compared with EvoF and GSGP, with median dimensionalities ranging from 10 [8,12] on bm4 to 22 [22,24] on Pollen. The SBFR variants generally maintain smaller representations than EvoF and GSGP, although SBFR-Ridge tends to produce larger representations than the other SBFR variants, with median dimensionalities ranging from 14 to 63 features across the benchmark datasets.

Overall, the results demonstrate that SBFR can construct compact feature representations while allowing their dimensionality to adapt to the choice of the internal regression evaluator and the characteristics of the dataset. SBFR-SVR exhibits the strongest tendency toward compact representations, achieving the lowest median dimensionality among the SBFR variants on six of the nine benchmark datasets. Combined with the predictive performance reported in the preceding analyses, these results indicate that SBFR can obtain informative symbolic representations without necessarily requiring a large number of constructed features.

Ranking Analysis

To provide a comprehensive comparative assessment of feature-space compactness, Figure 7 presents the average ranks of the evaluated feature-construction methods based on the dimensionality of the final feature representations reported in Table 9.

In this analysis, feature-space dimensionality is defined consistently across all methods as the number of transformed features or basis functions constituting the final feature representation supplied to the downstream predictive model. For SBFR, this

Table 9: Median [Q1, Q3] of the number of basis functions generated by the evaluated methods across the benchmark regression datasets.

Method	Concrete	Pollen	RatPol2D	Real estate	Toxicity	X05Y1Z15	bm1	bm4	keijzer5
EvoF	177 [169, 185]	209 [204, 213]	158 [152, 167]	208 [187, 227]	228 [207, 235]	206 [187, 222]	196 [190, 207]	145 [134, 151]	115 [96, 137]
FFX	268 [256, 277]	103 [94, 108]	81 [74, 85]	153 [122, 176]	217 [107, 251]	79 [65, 95]	1 [0, 2]	65 [59, 73]	5 [0, 26]
GSGP	638 [635, 643]	626 [618, 629]	633 [626, 637]	615 [609, 621]	624 [615, 631]	626 [621, 632]	1 [1, 1]	625 [618, 631]	627 [619, 632]
M3GP	19 [17, 22]	22 [22, 24]	18 [16, 19]	13 [12, 15]	15 [14, 16]	15 [14, 15]	1 [1, 1]	10 [8, 12]	12 [9, 13]
SBFR-DT	19 [17, 20]	17 [17, 17]	16 [15, 18]	14 [14, 15]	15 [14, 17]	16 [14, 17]	1 [1, 1]	9 [8, 10]	14 [12, 17]
SBFR-KNN	17 [16, 19]	10 [8, 17]	10 [8, 12]	9 [7, 10]	11 [10, 12]	15 [14, 16]	1 [1, 2]	7 [5, 10]	13 [12, 16]
SBFR-MLP	9 [9, 11]	14 [13, 14]	14 [12, 17]	6 [6, 8]	8 [7, 9]	15 [13, 16]	3 [2, 4]	11 [10, 13]	18 [16, 20]
SBFR-RF	12 [11, 14]	11 [10, 12]	9 [8, 10]	8 [8, 9]	7 [6, 8]	10 [9, 11]	4 [3, 5]	7 [6, 8]	10 [9, 12]
SBFR-Ridge	49 [47, 51]	14 [12, 16]	37 [33, 41]	36 [34, 39]	55 [53, 57]	38 [36, 40]	63 [55, 72]	28 [22, 32]	40 [38, 41]
SBFR-SVR	8 [6, 12]	4 [2, 5]	2 [1, 8]	5 [3, 7]	4 [2, 5]	11 [9, 15]	20 [13, 29]	11 [4, 20]	3 [1, 6]

corresponds to the number of retained symbolic basis functions, whereas for the baseline feature-construction methods, it corresponds to the dimensionality of their final constructed feature representation.

For each dataset, the methods are ranked according to their feature-space dimensionality, with lower ranks assigned to methods producing more compact representations. The average rank is then calculated across the nine benchmark regression datasets.

- **Top Performers:** SBFR-SVR achieves the best average rank (2.44), followed closely by SBFR-RF (2.56). These results indicate that both variants generally produce highly compact feature representations across the benchmark regression datasets.
- **Second Tier:** SBFR-KNN (3.78) and SBFR-MLP (4.11) follow with competitive average ranks, while M3GP (4.61) and SBFR-DT (4.94) also obtain relatively favorable positions in terms of feature-space compactness.
- **Lower-Ranked Methods:** SBFR-Ridge (7.11), FFX (7.28), EvoF (9.00), and GSGP (9.17) obtain higher average ranks, indicating that they generally produce larger feature representations across the evaluated datasets.

Overall, the ranking analysis demonstrates that several SBFR variants achieve favorable positions in terms of feature-space compactness. In particular, SBFR-SVR and SBFR-RF obtain the two best average ranks, indicating that these configurations generally produce more compact feature representations than the other evaluated methods. This result is consistent with the design objective of SBFR to construct informative symbolic representations without unnecessarily expanding the feature space.

The results also demonstrate that the dimensionality of the representation depends on the regression evaluator used within SBFR. The different average ranks obtained by the SBFR variants indicate that the choice of evaluator influences the number of basis functions retained by the framework. In particular, SBFR-SVR exhibits the strongest tendency toward compact representations, whereas SBFR-Ridge generally produces larger representations.

Compared with the evolutionary feature-construction methods, most SBFR variants achieve more favorable average ranks. Although individual exceptions are

observed on some datasets, SBFR generally produces more compact feature representations than EvoF and GSGP, while M3GP remains competitive in terms of dimensionality. These results highlight the ability of SBFR to adapt the size of the constructed representation to both the dataset and the selected regression evaluator.

Taken together with the predictive-performance results presented above, the findings indicate that SBFR can achieve a favorable trade-off between feature-space compactness and predictive performance. In particular, SBFR-RF combines a highly favorable compactness rank with competitive predictive performance across the evaluated metrics, while SBFR-SVR achieves the best average rank among the evaluated SBFR configurations in terms of feature-space compactness.

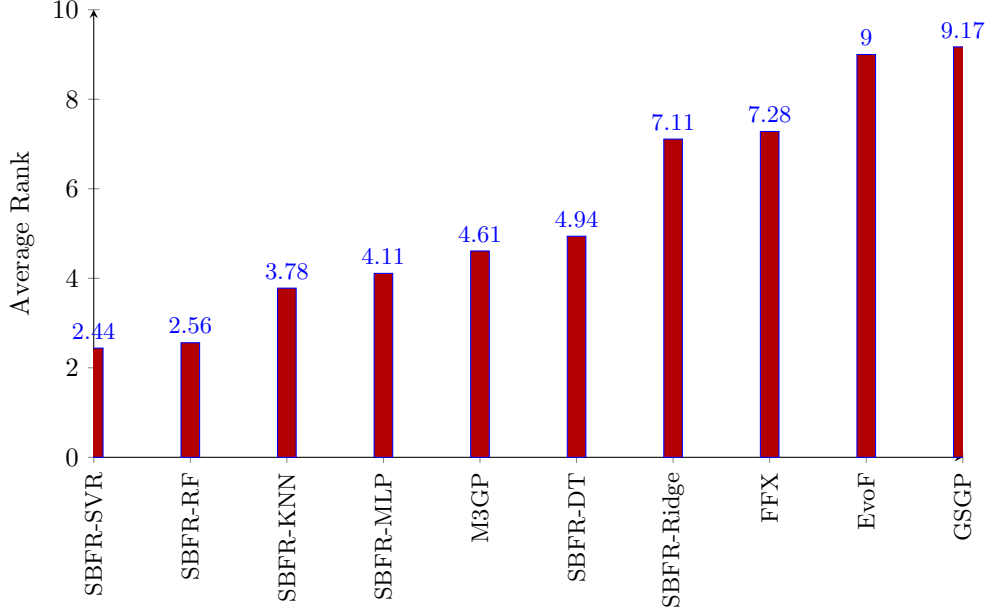


Fig. 7: Average ranks of the evaluated methods based on the dimensionality of their final feature representations. Lower ranks indicate more compact feature representations.

Statistical Significance

A Friedman test was conducted to assess whether statistically significant differences exist among the evaluated methods in terms of feature-space dimensionality, measured as the number of retained basis functions or constructed features in the final feature representation. The test yielded a Friedman statistic of 54.81 with a p -value below 0.001, indicating statistically significant differences among the compared methods. Thus, the null hypothesis that the methods exhibit equivalent performance in terms of feature-space dimensionality is rejected. This result indicates that the observed

differences in representation compactness are unlikely to be attributable to random variation alone and supports the application of post-hoc pairwise comparisons to identify the specific differences between methods.

The post-hoc analysis shows that SBFR-SVR and SBFR-RF achieve statistically significantly lower feature-space dimensionality than EvoF, GSGP, and FFX. These findings are consistent with the average-rank analysis, in which SBFR-SVR and SBFR-RF obtain the two best average ranks (2.44 and 2.56, respectively). Together, the Friedman and post-hoc analyses provide statistical evidence that the SVR- and RF-based configurations of SBFR produce more compact feature representations than several competing feature-construction approaches. In particular, the statistical results support the observed advantage of these two SBFR configurations in terms of feature-space compactness, while the differences among the remaining methods should be interpreted according to their corresponding post-hoc comparisons.

Discussion: Balancing Predictive Performance and Dimensionality

The results indicate a trade-off between predictive performance and feature-space dimensionality. SBFR-SVR achieves the best average rank in dimensionality (2.44), producing highly compact representations, but its predictive performance is less competitive than SBFR-RF. In contrast, SBFR-RF provides a favorable balance, achieving the second-best average rank for test-set RMSE (3.78) while retaining only 4–12 basis functions across the nine datasets.

SBFR-RF also achieves a better average RMSE rank than standalone RF (5.22), while its RMSE is not statistically different from EvoF, which obtains the best overall RMSE rank (2.89). At the same time, SBFR-RF produces substantially more compact representations than EvoF. Other SBFR variants provide different trade-offs, with SBFR-Ridge generally retaining more basis functions, whereas EvoF and particularly GSGP tend to produce larger feature representations without consistently achieving superior predictive performance.

Overall, SBFR-RF demonstrates the most favorable balance between predictive performance and feature-space compactness among the evaluated SBFR configurations, while SBFR-SVR provides the strongest dimensionality reduction.

4.4.3 Example Model Analysis

Figure 8 illustrates an example of the symbolic basis-function representation constructed by SBFR-RF for the RatPol2D dataset. The example demonstrates how SBFR combines the original input variables through elementary arithmetic operations to construct symbolic basis functions that are subsequently supplied to the downstream Random Forest regressor.

To obtain an illustrative high-performing solution, SBFR-RF was executed independently for 30 runs. The resulting representations were evaluated based on their validation-set RMSE, and a solution with high predictive performance was selected for illustration. This procedure provides an illustrative example of the symbolic representations produced by SBFR across independent runs without implying that the selected representation is unique or globally optimal.

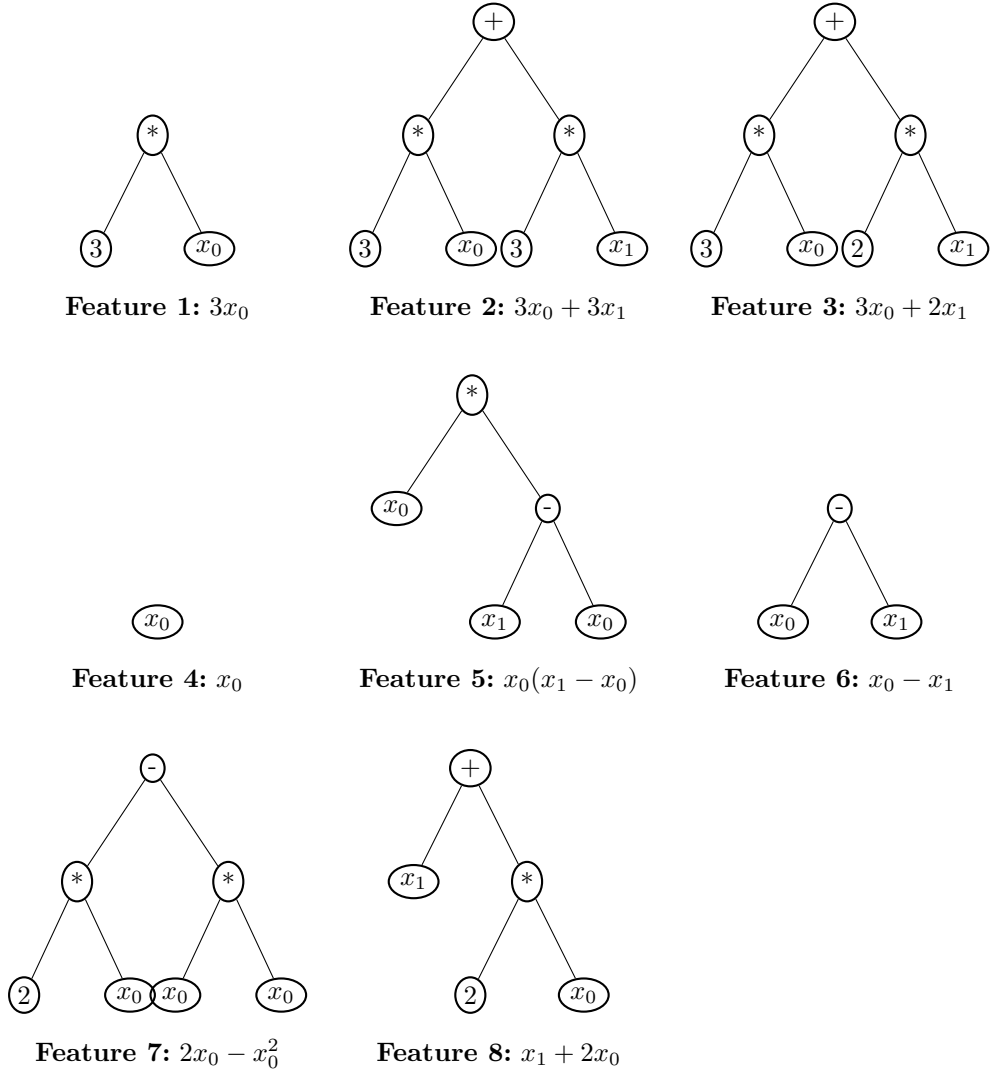


Fig. 8: Expression trees of the eight symbolic features constructed by SBFR-RF for the RatPol2D dataset. The terminal nodes x_0 and x_1 represent the two original input variables.

In the selected example, SBFR constructs eight symbolic basis functions, which are subsequently supplied to the downstream Random Forest regressor. The resulting representation remains compact while incorporating both linear and nonlinear transformations of the original input variables. Features 5 and 7, given by $x_0(x_1 - x_0)$ and $2x_0 - x_0^2$, respectively, introduce nonlinear interactions and transformations of the

original variables. In contrast, Features 1–4, 6, and 8 correspond to linear or affine transformations, including $3x_0$, $3x_0 + 3x_1$, $3x_0 + 2x_1$, x_0 , $x_0 - x_1$, and $x_1 + 2x_0$.

The constructed basis functions are explicitly interpretable because each function is represented as a mathematical expression involving the original variables and the elementary operators used by SBFR. Consequently, the transformations introduced during feature construction can be inspected directly, providing a transparent representation of the transformed feature space. This property is particularly relevant to symbolic feature construction, where the objective is not only to improve predictive performance but also to obtain explicit and interpretable transformations of the original features.

The terminal nodes x_0 and x_1 represent the two original input variables of the RatPol2D benchmark. Overall, this example illustrates how SBFR-RF can construct a compact set of explicit symbolic basis functions that combine simple linear transformations with nonlinear interactions. These constructed features can subsequently be exploited by a conventional regression model to capture potentially complex relationships in the input data while preserving an explicit mathematical representation of the constructed feature space.

4.4.4 Time Complexity Analysis

The computational cost of SBFR is mainly determined by candidate generation, synergy evaluation, and cross-validated regression evaluation.

Basis Function Generation.

Using RHH, the worst-case cost of generating one candidate basis function with maximum depth D and branching factor b is

$$T_{\text{gen}} = O(b^D). \quad (8)$$

For I candidate-generation iterations, the total cost is therefore

$$T_{\text{gen, total}} = O(Ib^D). \quad (9)$$

Synergy Evaluation.

If m basis functions are currently selected, Synergy by Adding requires one evaluation, while Synergy by Replacing requires up to m additional evaluations. Hence, at most $m + 1$ evaluations are required per candidate:

$$T_{\text{syn}} = O((m + 1)T_{\text{eval}}). \quad (10)$$

For N training samples and k -fold cross-validation,

$$T_{\text{eval}} = O(Nm + C_{\text{model}}(N, m, k)), \quad (11)$$

where C_{model} denotes the cost of training and evaluating the downstream regression model.

Overall Complexity.

Let M be the maximum number of retained basis functions. The overall worst-case complexity of SBFR is

$$T_{\text{SBFR}} = O(Ib^D + I(M+1)[NM + C_{\text{model}}(N, M, k)]). \quad (12)$$

Thus, the computational cost mainly depends on the number of candidate evaluations, expression depth, maximum representation size, and the complexity of the downstream regression model. In the experiments, the number of retained basis functions is generally small, with median dimensionalities below 20 for most SBFR variants (Table 9).

Time Complexity of Baseline SR Methods.

For comparison, the main computational costs of the baseline symbolic regression methods are summarized below.

Geometric Semantic GP (GSGP): GSGP evolves a population of P individuals over G generations. Let L denote the average size of the symbolic representations. Evaluating an individual over N samples requires approximately $O(NL)$ operations. Therefore, the main evaluation cost is

$$T_{\text{GSGP}} = O(GPNL). \quad (13)$$

This represents the main cost of individual evaluation and semantic construction, excluding additional costs associated with genetic operators.

M3GP: M3GP evolves T symbolic trees per individual. With an average tree size L , evaluating the trees requires approximately $O(NTL)$ operations. Fitting the resulting T -dimensional linear model requires approximately $O(NT^2 + T^3)$. Thus, for P individuals and G generations,

$$T_{\text{M3GP}} = O(GP(NTL + NT^2 + T^3)). \quad (14)$$

Evolutionary Forest (EvoF): EvoF combines GP with ensemble learning. Let Q denote the number of fidelity levels and L the average representation size. The main symbolic evaluation cost is

$$T_{\text{EvoF}} = O(GPQNL). \quad (15)$$

This term represents the main symbolic evaluation and feature-construction cost, excluding additional ensemble and model-selection operations.

Fast Function Extraction (FFX): FFX generates a library of B candidate basis functions. If L_{gen} is the average generation cost per basis function, library construction requires $O(BL_{\text{gen}})$, while evaluating the library over N samples requires $O(BN)$. Regularized regression over Λ regularization parameters requires approximately $O(BN\Lambda)$. Hence,

$$T_{\text{FFX}} = O(BL_{\text{gen}} + BN + BN\Lambda). \quad (16)$$

For $\Lambda > 1$, this can be simplified to $O(B(L_{\text{gen}} + N\Lambda))$.

Table 10 summarizes the dominant computational complexity terms of the compared methods.

Table 10: Dominant computational complexity terms of the compared methods

Method	Asymptotic Time Complexity
GSGP	$O(GPNL)$
M3GP	$O(GP(TNL + NT^2 + T^3))$
EvoF	$O(GPQNL)$
FFX	$O(B(L_{\text{gen}} + N\Lambda))$
SBFR	$O(Ib^D + I(M + 1)[NM + C_{\text{model}}(N, M, k)])$

5 Conclusion

This paper introduced Synergistic Basis Functions Regression (SBFR), a non-evolutionary framework for SR that constructs compact and interpretable symbolic representations through synergy-driven basis function selection. Instead of evolving complete symbolic models, SBFR iteratively generates candidate basis functions using the Ramped Half-and- Half (RHH) strategy and evaluates their contribution to the current representation. The proposed Synergy by Adding (S_{Add}) and Synergy by Replacing (S_{Rep}) criteria guide the selection and replacement of basis functions, allowing the representation to be progressively refined while controlling its dimensionality.

The experimental evaluation on nine benchmark regression datasets using 30 independent runs showed that SBFR-based configurations achieved competitive and, in several cases, improved predictive performance compared with the evaluated conventional regression and evolutionary feature-construction methods. Averaged across the nine datasets, the relative improvements were 21.56% for R^2 , 40.14% for RMSE, and 41.53% for MAE. The results also showed that predictive performance depends on the interaction between the constructed symbolic representation and the downstream regression model, indicating that different evaluators can lead to different performance-compactness trade-offs.

The dimensionality analysis demonstrated that SBFR can construct compact representations. SBFR-SVR achieved the best average rank in feature-space dimensionality (2.44), whereas SBFR-RF provided the most favorable overall balance between predictive performance and compactness. SBFR-RF achieved the second-best average rank in test-set RMSE (3.78), compared with 5.22 for standalone RF, while retaining relatively few basis functions. Its RMSE was also not statistically different from that of EvoF, which achieved the best average rank in RMSE (2.89). Furthermore, SBFR-SVR and SBFR-RF showed statistically significant dimensionality advantages over several competing methods, including EvoF, GSGP, and FFX.

The constructed basis functions also provide an explicit mathematical representation of the transformed feature space. The RatPol2D example illustrated how SBFR-RF can construct linear, interaction-based, and nonlinear transformations of the original variables while retaining a compact set of interpretable symbolic features.

Overall, the results support SBFR as a promising non-evolutionary approach to SR that jointly considers predictive performance, representation compactness, and interpretability. Among the evaluated configurations, SBFR-RF demonstrated the most favorable balance between predictive performance and compactness, while SBFR-SVR showed the strongest tendency toward dimensionality reduction. These findings suggest that synergy-driven basis-function selection provides a viable alternative to population-based feature-construction strategies by explicitly retaining symbolic transformations according to their contribution to the current representation.

Limitations and Future Work.

Despite the promising results, several limitations remain. First, SBFR relies on cross-validated predictive performance to evaluate candidate basis functions. Although this provides a principled selection criterion, it can increase computational cost due to the repeated evaluation of candidate representations. This cost may become more substantial for large or high-dimensional datasets. Future work could investigate surrogate-assisted evaluation, caching of previously evaluated expressions, and computationally efficient early-stopping strategies.

Second, the current implementation uses a relatively simple function set consisting of addition, subtraction, multiplication, and protected division. Extending the function set with exponential, logarithmic, and trigonometric operators could increase the expressive capability of SBFR. Such extensions would require appropriate mechanisms for numerical stability, domain constraints, and expression-complexity control.

Third, the improvement threshold, $improve_{threshold} = 1$, is fixed throughout the search. Although the sensitivity analysis indicates that this value provides a suitable balance between retaining useful basis functions and avoiding marginal improvements, future work could investigate adaptive thresholding based on search progress, prediction error, or dataset characteristics.

Fourth, although SBFR supports different downstream regression models, only a limited set of evaluators was considered in this study. The differences observed among SBFR-DT, SBFR-KNN, SBFR-MLP, SBFR-RF, SBFR-Ridge, and SBFR-SVR suggest that the choice of evaluator can affect both predictive performance and representation dimensionality. Future studies could therefore consider additional learners, such as gradient-boosting models, Gaussian processes, and other nonlinear or ensemble regressors.

Finally, SBFR was evaluated on nine benchmark regression datasets. Broader evaluation on larger real-world and application-specific datasets is needed to further assess its scalability and generality. Future work could also extend the synergy-based basis-function construction principle to other tasks, including time-series forecasting, classification, and scientific equation discovery.

Appendix A Detailed Description of the SBFR Framework

This appendix presents the main algorithmic components of the SBFR framework, including symbolic basis-function generation, synergy-based evaluation and selection,

and cross-validated estimation of feature utility. These algorithms complement the high-level description provided in the main text.

A.1 Generate Basis Functions

Candidate basis functions are generated using the *Ramped Half-and-Half* (RHH) strategy, which combines the *full* and *grow* methods to generate symbolic expressions with diverse structures and depths. The *full* method generates trees whose branches reach the selected depth, whereas the *grow* method allows branches of different depths.

At each iteration, SBFR randomly selects a tree depth from the predefined range $[d_{\min}, d_{\max}]$ and randomly chooses either the *full* or *grow* method to generate one candidate basis function. The generated expression is subsequently evaluated according to its synergy with the current feature vector. This process is repeated iteratively, with one candidate generated per iteration. The RHH procedure is summarized in Algorithm 3.

Algorithm 3 Ramped Half-and-Half (RHH) Generation Procedure

Require: Function set FS , terminal set TS , depth range $[d_{\min}, d_{\max}]$

Ensure: Generated candidate function f

```

1:  $d \leftarrow \text{RandomDepth}(d_{\min}, d_{\max})$ 
2:  $method \leftarrow \text{RandomChoice}(\{Full, Grow\})$ 
3: if  $method = Full$  then
4:    $f \leftarrow \text{Full}(FS, TS, d)$ 
5: else
6:    $f \leftarrow \text{Grow}(FS, TS, d)$ 
7: end if
8: return  $f$ 

```

By varying both tree depth and generation method, RHH introduces structural diversity into the candidate basis functions. These symbolic expressions are then passed to the synergy-based evaluation and selection procedure.

A.2 Evaluation of Basis Function Vectors

The predictive utility of a basis-function vector (FV) is evaluated using k -fold cross-validation. First, the selected basis functions are applied to the input data to construct the transformed feature matrix. In each fold, the selected regression model is trained on the training partition and evaluated on the held-out validation partition. The root mean squared error (RMSE) on the validation partition of fold i is defined as

$$RMSE_i = \sqrt{\frac{1}{|D_i|} \sum_{j \in D_i} (y_j - \hat{y}_j)^2}, \quad (\text{A1})$$

where D_i denotes the validation set of fold i , y_j is the observed target value, and $\hat{y}_j$ is the corresponding prediction produced by the regression model. The cross-validated RMSE is then computed as the mean of the RMSE values obtained across the k folds:

$$CV\text{-}RMSE = \frac{1}{k} \sum_{i=1}^k RMSE_i. \quad (\text{A2})$$

The resulting $CV\text{-}RMSE$ is used as the predictive performance measure of the basis-function vector. The procedure is summarized in Algorithm 4.

Algorithm 4 Cross-validated Evaluation of a Basis Function Vector

Require: Original inputs X , target values Y , basis function vector FV , number of folds k , regression model M

Ensure: Cross-validated RMSE

```

1:  $Z \leftarrow \text{ApplyBasisFunctions}(FV, X)$ 
2:  $RMSE \leftarrow 0$ 
3: for  $i = 1$  to  $k$  do
4:   Split  $(Z, Y)$  into training and validation folds
5:    $model \leftarrow \text{Train}(M, Z_{train}, Y_{train})$ 
6:    $error_i \leftarrow \text{ComputeRMSE}(model, Z_{val}, Y_{val})$ 
7:    $RMSE \leftarrow RMSE + error_i$ 
8: end for
9:  $RMSE \leftarrow \frac{RMSE}{k}$ 
10: return  $RMSE$ 

```

This cross-validated RMSE provides an estimate of the predictive performance of the feature vector across different data partitions. Together with the synergy criteria, it is used to compare candidate basis functions and identify those that improve the current representation.

A.3 Synergy-Based Evaluation and Selection

SBFR evaluates each candidate basis function according to its contribution to the current feature vector (FV). Two complementary criteria are used: Synergy by Adding (S_{Add}), which evaluates appending a candidate to FV , and Synergy by Replacing (S_{Rep}), which evaluates replacing each existing basis function with the candidate. The resulting cross-validated RMSE values are used to quantify the relative improvement in predictive performance.

A.3.1 Synergy by Adding

Synergy by Adding measures the relative improvement obtained by appending a candidate basis function to the current feature vector. The synergy score is defined in Equation 2.

A positive S_{Add} indicates that the candidate reduces the prediction error. The candidate is considered for addition when its synergy exceeds $improve_{\text{threshold}}$ and is preferable to the best replacement alternative.

Example:

Algorithm 5 Synergy by Adding (S_{Add})

Require: Original inputs X , target values Y , basis function vector FV , candidate function f , evaluation function F

Ensure: Synergy score S_{Add}

- 1: $base_error \leftarrow F(FV, X, Y)$
 - 2: $FV_{\text{new}} \leftarrow FV \cup \{f\}$
 - 3: $new_error \leftarrow F(FV_{\text{new}}, X, Y)$
 - 4: $S_{\text{Add}} \leftarrow \frac{base_error - new_error}{base_error} \times 100$
 - 5: **return** S_{Add}
-

Assume $FV = f_1, f_2, f_3$ with $RMSE(FV) = 10.5$. If adding a candidate $f = 2x_2$ reduces the RMSE to 8.2, then

$$S_{\text{Add}}(f, FV) = \frac{10.5 - 8.2}{10.5} \times 100 \approx 21.9\%.$$

Thus, the candidate provides a positive improvement and can be considered for addition according to the selection criterion.

A.3.2 Synergy by Replacing

Synergy by Replacing evaluates whether a candidate provides a better alternative to an existing basis function. For an FV containing m basis functions, the candidate is temporarily substituted for each basis function, producing a replacement synergy vector SV_{Rep} .

Algorithm 6 Synergy by Replacing (S_{Rep})

Require: Original inputs X , target values Y , basis function vector FV , candidate function f , evaluation function F

Ensure: Synergy vector S_{Rep}

- 1: $m \leftarrow |FV|$
 - 2: $S_{\text{Rep}} \leftarrow$ zero vector of size m
 - 3: $base_error \leftarrow F(FV, X, Y)$
 - 4: **for** $i = 1$ to m **do**
 - 5: $FV_{\text{temp}} \leftarrow FV$
 - 6: $FV_{\text{temp}}[i] \leftarrow f$
 - 7: $temp_error \leftarrow F(FV_{\text{temp}}, X, Y)$
 - 8: $S_{\text{Rep}}[i] \leftarrow \frac{base_error - temp_error}{base_error} \times 100$
 - 9: **end for**
 - 10: **return** S_{Rep}
-

The candidate is selected for replacement when the highest value in SV_{Rep} exceeds $improve_{\text{threshold}}$ and is preferred to the addition alternative.

Example:

Let $FV = f_1, f_2, f_3$ with $RMSE(FV) = 10.5$ and candidate $f = 2x_2$. If replacing f_1 , f_2 , and f_3 produces RMSE values of 9.0, 11.0, and 9.5, respectively, the corresponding replacement synergy scores are approximately 14.3%, -4.8%, and 9.5%.

The highest score corresponds to replacing f_1 . If it satisfies the selection criterion, the updated feature vector becomes f, f_2, f_3 .

Appendix B Baseline Methods Details

This appendix describes the baseline methods used for comparison with the proposed SBFR framework. The methods are grouped into conventional regression models and SR approaches.

B.1 Conventional Regressors

The conventional regressors are applied directly to the original feature space to provide a reference for evaluating the effect of symbolic basis-function construction in SBFR.

- **Decision Tree Regressor:** A non-parametric model that recursively partitions the input space to minimize prediction error [Breiman et al. \(1984\)](#).
- **K-Nearest Neighbors (KNN):** A lazy learning method that predicts target values based on the nearest training instances [Altman \(1992\)](#).
- **Support Vector Regressor (SVR):** A kernel-based regression model that fits a function within an ϵ -insensitive margin and can model nonlinear relationships [Drucker et al. \(1997\)](#).
- **Multi-Layer Perceptron (MLP):** A feedforward neural network trained using backpropagation for modeling nonlinear relationships [Bishop \(1995\)](#).
- **Random Forest (RF):** An ensemble of decision trees trained on bootstrap samples for robust nonlinear prediction [Breiman \(2001\)](#).
- **Ridge Regression (Ridge):** A regularized linear regression method using an ℓ_2 penalty on the regression coefficients:

$$\min_{\beta} \left(\sum_{i=1}^n (y_i - \mathbf{x}_i^\top \beta)^2 + \lambda \|\beta\|_2^2 \right), \quad (\text{B3})$$

where λ controls the regularization strength.

- **Gradient Boosted Decision Trees (GBDT):** An additive ensemble method that sequentially builds decision trees to reduce prediction error [Friedman \(2001\)](#).
- **XGBoost:** An efficient and regularized implementation of gradient-boosted decision trees designed for computational efficiency and scalability [Chen and Guestrin \(2016\)](#).

B.2 SR Methods

The SR baselines include both genetic-programming-based and non-evolutionary approaches.

- **gplearn** ²: A Python implementation of standard GP for symbolic regression. It evolves symbolic expressions using conventional GP operators, including selection, crossover, and mutation.
- **Geometric Semantic GP (GSGP)** ³: A GP approach that performs variation in the semantic space of program outputs rather than directly modifying program syntax.
- **M3GP (Multidimensional Multiclass GP)** ⁴: A GP method that evolves multiple symbolic trees to construct multidimensional representations. The implementation used in this study supports parallel processing and integration with `scikit-learn`.
- **Evolutionary Forest (EvoF)** ⁵: A SR approach that combines evolutionary symbolic feature construction with ensemble learning, incorporating mechanisms such as guided mutation, multi-fidelity evaluation, and ensemble selection.
- **Fast Function Extraction (FFX)** ⁶: A deterministic SR method that constructs a library of candidate basis functions and uses regularized regression to obtain a sparse predictive model.

Appendix C Datasets

This study evaluates SBFR on nine regression and SR benchmark problems: *X05Y1Z15*, *bm1*, *bm2*, *Concrete*, *RatPol2D*, *Pollen*, *keijzer5*, *Toxicity*, and *Real Estate Valuation*. The datasets were selected to provide diversity in functional complexity, input dimensionality, sample size, and data origin, covering both synthetic benchmark functions and real-world regression problems. Their main characteristics are summarized in Table C1.

- **bm1 (Univariate Summation Function)**: The bm1 benchmark is defined as

$$f(x) = \sum_{i=1}^x \frac{1}{i}.$$

It represents a univariate summation problem with cumulative and asymptotic behavior. The training domain is $x \in 1, \dots, 50$, while the test domain extends to $x \in 1, \dots, 120$, allowing evaluation on values beyond the training range. The benchmark has been used in previous SR studies [Astarabadi and Ebadzadeh \(2019\)](#); [Keijzer \(2003\)](#); [Pennachin et al. \(2010\)](#).

- **bm2 (Rational Bivariate Function)**: The bm2 benchmark is defined as

$$f(x_1, x_2) = \frac{1}{1 + \frac{1}{x_1^4}} + \frac{1}{1 + \frac{1}{x_2^4}}.$$

The training inputs are defined on the grid $x_1, x_2 \in [-5 : 0.4 : 5]$, while testing uses the grid $[-6 : 0.24 : 6] \setminus 0$. This benchmark evaluates approximation of nonlinear rational relationships over a broader test domain [Astarabadi and Ebadzadeh \(2019\)](#).

²<https://github.com/trevorstevens/gplearn>

³<https://github.com/jespb/Python-GSGP>

⁴<https://github.com/jespb/Python-M3GP>

⁵<https://github.com/hengzhe-zhang/EvolutionaryForest>

⁶<https://github.com/ffx-org/ffx>

- **keijzer5:** The keijzer5 benchmark is a three-variable SR problem defined by

$$f(x_0, x_1, x_2) = \frac{30x_0x_2}{(x_0 - 10)x_1^2}.$$

The input variables are independently sampled according to

$$x_0 \sim \mathcal{U}[-1, 1], \quad x_1 \sim \mathcal{U}[1, 2], \quad x_2 \sim \mathcal{U}[-1, 1],$$

with 2,000 generated instances. The problem combines multiplicative and rational relationships and is widely used as a SR benchmark [Astarabadi and Ebadzadeh \(2019\)](#); [Pennachin et al. \(2010\)](#); [Keijzer \(2003\)](#); [Qi et al. \(2023\)](#).

- **X05Y1Z15:** The X05Y1Z15 benchmark is defined as

$$f(x_1, x_2, x_3) = (1 + x_1^{0.5} + x_2^{-1} + x_3^{-1.5})^2.$$

The three input variables are sampled uniformly from $[1, 6]$, resulting in 1,216 instances. The function combines fractional and inverse powers with nonlinear interactions among the input variables [Astarabadi and Ebadzadeh \(2019\)](#).

- **Concrete Compressive Strength:** The Concrete Compressive Strength dataset is a real-world regression dataset from the UCI Machine Learning Repository [Yeh \(1998\)](#). It contains 1,030 instances and eight quantitative input variables describing concrete composition and age. The target variable is compressive strength, and the dataset contains no missing values. It has been widely used in regression research [Hien and Hoai \(2012\)](#); [Hien et al. \(2011\)](#); [Amir Haeri et al. \(2015\)](#); [Amini et al. \(2020\)](#); [Azad and Ryan \(2011\)](#); [Žegklitz and Posík \(2020\)](#); [La Cava and Moore \(2020\)](#); [Zojaji and Ebadzadeh \(2018\)](#); [Ruberto et al. \(2020\)](#).
- **Pollen:** The Pollen dataset originates from the StatLib Datasets Archive and was generated by David Coleman. It contains 3,848 observations and five variables [StatLib \(2025\)](#). Four variables describe geometric and physical characteristics of pollen grains (ridge, nub, crack, and weight), while density is the target variable. The dataset has been used in regression and symbolic modeling studies [Amir Haeri et al. \(2015\)](#); [Amini et al. \(2020\)](#); [Nguyen et al. \(2016\)](#); [Zojaji and Ebadzadeh \(2018\)](#).
- **RatPol2D:** RatPol2D is a two-dimensional synthetic SR benchmark defined by

$$f(x_1, x_2) = \frac{(x_1 - 3)^4 + (x_2 - 3)^3 - (x_2 - 3)}{(x_2 - 2)^4 + 10}.$$

The training and validation sets each contain 50 samples drawn from $[0.05, 6.05]$. The test set is generated on the grid $[-0.25 : 0.2 : 6.35]$. The benchmark combines polynomial and rational components with nonlinear interactions and has been used in previous symbolic regression studies [Amir Haeri et al. \(2015\)](#); [Amini et al. \(2020\)](#); [Zojaji and Ebadzadeh \(2018\)](#); [Hien and Hoai \(2012\)](#).

- **Toxicity (LD50):** The Toxicity dataset is a high-dimensional chemical regression problem containing 1,216 instances and 626 molecular descriptor variables. The target variable is the LD50 value. It provides a relatively high-dimensional setting for evaluating symbolic feature construction and selection [Haeri et al. \(2017\)](#); [Amir Haeri et al. \(2015\)](#); [Amini et al. \(2020\)](#); [Zojaji and Ebadzadeh \(2018\)](#); [Castelli et al. \(2011\)](#).

- **Real Estate Valuation:** The Real Estate Valuation dataset is a real-world regression dataset from the UCI Machine Learning Repository [Yeh \(2018\)](#). It contains 414 instances and six input variables representing transaction date, house age, distance to the nearest MRT station, number of nearby convenience stores, latitude, and longitude. The target variable is house price per unit area. The original UCI documentation reports a random split of approximately two-thirds of the observations for training and one-third for testing.

The nine datasets cover a broad range of regression characteristics. The synthetic benchmarks (**bm1**, **bm2**, **keijzer5**, **X05Y1Z15**, and **RatPol2D**) provide explicitly defined mathematical functions and controlled input domains, covering summation, rational, polynomial, multiplicative, fractional-power, and nonlinear interaction structures. The real-world datasets (**Concrete**, **Pollen**, **Toxicity**, and **Real Estate Valuation**) introduce measured-data problems with different sample sizes and input characteristics. Their dimensionality ranges from four to 626 input variables, and their sample sizes range from 414 to 3,848 observations. Several of these datasets have also been used in SR and GP research [La Cava et al. \(2019\)](#); [Silva et al. \(2016\)](#); [Ruberto et al. \(2021\)](#); [Ma and Gao \(2020b,a\)](#).

Table C1: Dataset specifications used in the experimental evaluation.

Dataset	Type / Source	Function / Description	Input Specification	Total Instances
bm1	Synthetic	$f(x) = \sum_{i=1}^2 \frac{1}{i}$	Train: $x \in \{1, \dots, 50\}$; Test: $x \in \{1, \dots, 120\}$ 120	
bm2	Synthetic	$\frac{1}{1 + \frac{1}{x_1^2}} + \frac{1}{1 + \frac{1}{x_2^2}}$	Train: $[-5 : 0.4 : 5]^2$; Test: $[-6 : 0.24 : 6]^2 \setminus \{0\}$	3176
keijzer5	Synthetic	$\frac{30x_0x_2}{(x_0 - 10)x_1^2}$	$x_0 \sim \mathcal{U}[-1, 1]$; $x_1 \sim \mathcal{U}[1, 2]$; $x_2 \sim \mathcal{U}[-1, 1]$	2000
X05Y1Z15	Synthetic	$(1 + x_1^{0.5} + x_2^{-1} + x_3^{-1.5})^2$	$x_1, x_2, x_3 \in [1, 6]$	1216
Concrete	UCI Yeh (1998)	Concrete compressive strength	8 input variables, 1 target	1030
Pollen	Synthetic; StatLib StatLib (2025)	Pollen density regression	4 input variables, 1 target	3848
RatPol2D	Synthetic	$\frac{(x_1 - 3)^4 + (x_2 - 3)^5 - (x_2 - 3)}{(x_2 - 2)^4 + 10}$	Train: 50; Validation: 50; Test: 1156 [†]	1256 [†]
Toxicity	Real-world	Chemical toxicity (LD50)	626 input variables, 1 target	1216
Real Estate Valuation	UCI Yeh (2018)	Real estate price prediction	6 input variables, 1 target	414

[†] The grid $[-0.25 : 0.2 : 6.35]$ contains 1,156 points. If this grid was used exactly as specified, the total number of RatPol2D instances is therefore 1,256 (50 training + 50 validation + 1,156 test). The final values should be verified against the actual experimental files.

4 Declarations

4.1 Funding

The authors did not receive support from any organization for the submitted work.

4.2 Competing interests

The authors have no relevant financial or non-financial interests to disclose.

4.3 Data availability

The datasets used in this study are publicly available from their respective sources.

4.4 Code availability

To facilitate reproducibility and further research, the complete source code and experimental configurations used in this study are publicly available at <https://github.com/hsarv/SBFR>.

4.5 Authors' contributions

The first author contributed to the implementation of the proposed method, experimental evaluation, data analysis, and preparation of the original manuscript. The second author contributed to the conceptualization and methodology of the study, supervised the research, and provided critical revisions and scientific guidance throughout the study. Both authors reviewed and approved the final manuscript.

References

- Al-Helali B, Chen Q, Xue B, et al (2021) Multitree genetic programming with new operators for transfer learning in symbolic regression with incomplete data. *IEEE Transactions on Evolutionary Computation* 25(6):1049–1063
- Al-Helali B, Chen Q, Xue B, et al (2024) Genetic programming for feature selection based on feature removal impact in high-dimensional symbolic regression. *IEEE Transactions on Emerging Topics in Computational Intelligence* 8(3):2269–2282
- Altman NS (1992) An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician* 46(3):175–185
- Amini S, Abdollahi M, Haeri M (2020) Rule-centred genetic programming (rcgp): An imperialist competitive approach. *Applied Intelligence* 50(8):2589–2609
- Amir Haeri M, Ebadzadeh MM, Folino G (2015) Improving gp generalization: A variance-based layered learning approach. *Genetic Programming and Evolvable Machines* 16:27–55
- Arnaldo I, Krawiec K, O'Reilly UM (2014) Multiple regression genetic programming. In: *Proceedings of the 2014 annual conference on genetic and evolutionary computation*, pp 879–886
- Arnaldo I, O'Reilly UM, Veeramachaneni K (2015) Building predictive models via feature synthesis. In: *Proceedings of the 2015 annual conference on genetic and evolutionary computation*, pp 983–990
- Astarabadi SSM, Ebadzadeh MM (2019) Genetic programming performance prediction and its application for symbolic regression problems. *Information Sciences* 502:418–433
- Austel V, Cornelio C, Dash S, et al (2020) Symbolic regression using mixed-integer nonlinear optimization. *arXiv preprint arXiv:200606813* URL <https://arxiv.org/>

- Azad R, Ryan C (2011) Variance based selection to improve test set performance in genetic programming. In: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation (GECCO), ACM, pp 1315–1322
- Batista JE, Rodrigues NM, Vanneschi L, et al (2024) M6gp: Multiobjective feature engineering. In: 2024 IEEE Congress on Evolutionary Computation (CEC), IEEE, pp 1–8
- Bi Y, Xue B, Zhang M (2020) Evolving deep forest with automatic feature extraction for image classification using genetic programming. In: International Conference on Parallel Problem Solving from Nature, Springer, pp 3–18
- Biggio L, Bendinelli T, Neitz A, et al (2021) Neural symbolic regression that scales. In: International Conference on Machine Learning, PMLR, pp 936–945
- Bishop CM (1995) Neural networks for pattern recognition. Oxford university press
- Breiman L (2001) Random forests. Machine learning 45(1):5–32
- Breiman L, Friedman J, Stone CJ, et al (1984) Classification and Regression Trees. Wadsworth International Group
- Cárdenas Florido L, Trujillo L, Hernandez DE, et al (2024) M5gp: Parallel multi-dimensional genetic programming with multidimensional populations for symbolic regression. Mathematical and Computational Applications 29(2):25
- Castelli M, Manzoni L, Silva S, et al (2011) A quantitative study of learning and generalization in genetic programming. In: Proceedings of the 14th European Conference on Genetic Programming (EuroGP), Springer, pp 25–36
- Chapman JL (2012) Handbook of Parametric and Nonparametric Statistical Procedures, 5th edn. Chapman and Hall/CRC, Boca Raton, FL
- Chen C, Luo C, Jiang Z (2017a) Elite bases regression: A real-time algorithm for symbolic regression. In: 2017 13th International conference on natural computation, fuzzy systems and knowledge discovery (ICNC-FSKD), IEEE, pp 529–535
- Chen G (2020) Learning symbolic expressions via gumbel-max equation learner networks. arXiv preprint arXiv:201206921
- Chen Q, Xue B, Niu B, et al (2016) Improving generalisation of genetic programming for high-dimensional symbolic regression with feature selection. Proceedings of the Genetic and Evolutionary Computation Conference p 3793–3800
- Chen Q, Zhang M, Xue B (2017b) Feature selection to improve generalization of genetic programming for high-dimensional symbolic regression. IEEE Transactions

on Evolutionary Computation 21(5):792–806

Chen T, Guestrin C (2016) Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp 785–794

Costa A, Dangovski R, Dugan O, et al (2020) Fast neural models for symbolic regression at scale. arXiv preprint arXiv:200710784

Criado PR, Rolanía DB, de la Hoz D, et al (2024) Estimation of distribution algorithm for grammar-guided genetic programming. *Evolutionary Computation* 32(4):339–370

Crochepierre L, Boudjeloud-Assala L, Barbesant V (2022) A reinforcement learning approach to domain-knowledge inclusion using grammar guided symbolic regression. arXiv preprint arXiv:220204367

De Melo VV (2014) Kaizen programming. In: Proceedings of the 2014 annual conference on genetic and evolutionary computation, pp 895–902

De Melo VV, Banzhaf W (2018) Automatic feature engineering for regression models with machine learning: An evolutionary computation and statistics hybrid. *Information Sciences* 430:287–313

Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research* 7(Jan):1–30

Ding L, Boldi R, Helmuth T, et al (2022) Lexicase selection at scale. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp 2054–2062

Dong J, Zhong J (2025) Recent advances in symbolic regression. *ACM Computing Surveys* 57(11):1–37

Dong J, Zhong J, Liu WL, et al (2024) Evolving equation learner for symbolic regression. *IEEE Transactions on Evolutionary Computation*

Drucker H, Burges CJ, Kaufman L, et al (1997) Support vector regression machines. In: Advances in neural information processing systems, pp 155–161

Engle MR, Sahinidis NV (2022) Deterministic symbolic regression with derivative information: General methodology and application to equations of state. *AIChE Journal* 68(6):e17457

de França FO (2018) A greedy search tree heuristic for symbolic regression. *Information Sciences* 442:18–32

Friedman JH (2001) Greedy function approximation: a gradient boosting machine. *Annals of statistics* pp 1189–1232

- Gharoun H, Khorshidi MS, Yazdanjue N, et al (2024) Shadow gene guidance: A novel approach for elevating genetic programming classifications and boosting predictive confidence. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, p 2095–2098
- Gunawan O, Horesh L, Nannicini G, et al (2017) Symbolic regression embedding dimensionality analysis. U.S. Patent Application 20190188256A1, Application No. 15/843,118, filed December 15, 2017
- Guo J, Zhang R, Peng S, et al (2024) Efficient symbolic policy learning with differentiable symbolic expression. In: Proceedings of the Advances in Neural Information Processing Systems
- Haeri MA, Ebadzadeh MM, Folino G (2017) Statistical genetic programming for symbolic regression. *Applied Soft Computing* 60:447–469
- Hien NT, Hoai NX (2012) Learning in stages: A layered learning approach for genetic programming. In: 2012 IEEE RIVF International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future, RIVF 2012
- Hien NT, Hoai NX, McKay B (2011) A study on genetic programming with layered learning and incremental sampling. In: 2011 IEEE Congress of Evolutionary Computation (CEC), IEEE, pp 1179–1185
- Hoerl AE, Kennard RW (1970) Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics* 12(1):55–67
- Holt S, Qian Z, van der Schaar M (2022) Deep generative symbolic regression. In: Proceedings of the International Conference on Learning Representations, pp 1–42, <https://doi.org/10.48550/arXiv.2401.00282>
- Hornik K (1991) Approximation capabilities of multilayer feedforward networks. *Neural networks* 4(2):251–257
- Huang Z, Mei Y, Zhong J (2022) Semantic linear genetic programming for symbolic regression. *IEEE Transactions on Cybernetics* 54(2):1321–1334
- Huang Z, Mei Y, Zhang F, et al (2023) Multitask linear genetic programming with shared individuals and its application to dynamic job shop scheduling. *IEEE Transactions on Evolutionary Computation* 28(6):1546–1560
- Icke I, Bongard JC (2013) Improving genetic programming based symbolic regression using deterministic machine learning. In: 2013 IEEE Congress on Evolutionary Computation, IEEE, pp 1763–1770
- Jin Y, Fu W, Kang J, et al (2019) Bayesian symbolic regression. *arXiv preprint arXiv:191008892*

- Kamienny PA, Lample G, Lamprier S, et al (2023) Deep generative symbolic regression with monte-carlo-tree-search. In: Proceedings of the International Conference on Machine Learning, pp 15655–15668
- Kammerer L, Kronberger G, Burlacu B, et al (2020) Symbolic regression by exhaustive search: Reducing the search space using syntactical constraints and efficient semantic structure deduplication. In: Genetic Programming Theory and Practice XVII. Springer, pp 79–99
- Kammerer L, Kronberger G, Kommenda M (2022) Symbolic regression with fast function extraction and nonlinear least squares optimization. In: Proceedings of the International Conference on Computer Aided Systems Theory, pp 139–146
- Karizaky HS, Ebadzadeh MM (2026) Dynamic cooperative genetic programming for synergistic feature construction in classification problems. *Applied Soft Computing* p 114854
- Kattan A, Ong YS (2015) Surrogate genetic programming: A semantic aware evolutionary search. *Information Sciences* 296:345–359
- Kattan A, Agapitos A, Ong YS, et al (2016) Gp made faster with semantic surrogate modelling. *Information Sciences* 355:169–185
- Keijzer M (2003) Improving symbolic regression with interval arithmetic and linear scaling. In: European Conference on Genetic Programming, Springer, pp 70–82
- Kim J, Leyffer S, Balaprakash P (2023a) Learning symbolic expressions: Mixed-integer formulations, cuts, and heuristics. *INFORMS Journal on Computing* 35(6):1383–1403
- Kim S, Lu P, Mukherjee S, et al (2023b) Integration of neural network-based symbolic regression in deep learning for scientific discovery. *IEEE Transactions on Neural Networks and Learning Systems*
- Krawiec K, O'Reilly UM (2014) Behavioral programming: a broader and more detailed take on semantic gp. In: Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation, pp 935–942
- Krawiec K, Swan J, O'Reilly UM (2016) Behavioral program synthesis: Insights and prospects. *Genetic programming theory and practice XIII* pp 169–183
- Kronberger G, Kammerer L, Burlacu B, et al (2019) Cluster analysis of a symbolic regression search space. In: Genetic Programming Theory and Practice XVI. Springer, pp 85–102
- Kusner MJ, Paige B, Hernández-Lobato JM (2017) Grammar variational autoencoder. In: International conference on machine learning, PMLR, pp 1945–1954

- La Cava W, Moore J (2017a) A general feature engineering wrapper for machine learning using-lexicase survival. In: European Conference on Genetic Programming, Springer, pp 80–95
- La Cava W, Moore JH (2017b) Ensemble representation learning: an analysis of fitness and survival for wrapper-based genetic programming methods. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp 961–968
- La Cava W, Moore JH (2019) Semantic variation operators for multidimensional genetic programming. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp 1056–1064
- La Cava W, Moore JH (2020) Learning feature spaces for regression with genetic programming. *Genetic Programming and Evolvable Machines* 21(3):433–467
- La Cava W, Singh TR, Taggart J, et al (2018) Learning concise representations for regression by evolving networks of trees. arXiv preprint arXiv:180700981
- La Cava W, Silva S, Danai K, et al (2019) Multidimensional genetic programming for multiclass classification. *Swarm and evolutionary computation* 44:260–272
- Landajuela M, Lee CS, Yang J, et al (2022) A unified framework for deep symbolic regression. In: Proceedings of the Advances in Neural Information Processing Systems, pp 33985–33998
- Lichman M (2013) Uci machine learning repository, university of california, irvine, school of information and computer sciences. <https://archive.ics.uci.edu/ml/index.php>
- Ma J, Gao X (2020a) Designing genetic programming classifiers with feature selection and feature construction. *Applied Soft Computing* 97:106826
- Ma J, Gao X (2020b) A filter-based feature construction and feature selection approach for classification using genetic programming. *Knowledge-Based Systems* 196:105806
- Manazir A, Raza K (2019) Recent developments in cartesian genetic programming and its variants. *ACM computing surveys (CSUR)* 51(6):1–29
- Martius G, Lampert CH (2016) Extrapolation and learning equations. arXiv preprint arXiv:161002995
- McConaghy T (2011) Ffx: Fast, scalable, deterministic symbolic regression technology. *Genetic Programming Theory and Practice IX* pp 235–260
- Moraglio A, Krawiec K, Johnson CG (2012) Geometric semantic genetic programming. In: Parallel Problem Solving from Nature-PPSN XII: 12th International Conference, Taormina, Italy, September 1-5, 2012, Proceedings, Part I 12, Springer, pp 21–31

- Mundhenk TN, Landajuela M, Glatt R, et al (2021) Symbolic regression via neural-guided genetic programming population seeding. arXiv preprint arXiv:211100053
- Muñoz L, Trujillo L, Silva S, et al (2019) Evolving multidimensional transformations for symbolic regression with m3gp. *Memetic Computing* 11:111–126
- Neumann P, Cao L, Russo D, et al (2020) A new formulation for symbolic regression to identify physico-chemical laws from experimental data. *Chemical Engineering Journal* 387:123412
- Nguyen QU, Pham TA, Nguyen XH, et al (2016) Subtree semantic geometric crossover for genetic programming. *Genetic Programming and Evolvable Machines* 17:25–53
- Niu J, Gao X, Bai M, et al (2025) Adaptive multi-tree genetic programming for multiple feature construction. In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, p 651–654
- Ouyang R, Curtarolo S, Ahmetcik E, et al (2018) Sisso: A compressed-sensing method for identifying the best low-dimensional descriptor in an immensity of offered candidates. *Physical Review Materials* 2(8):083802
- Pawlak TP, Wieloch B, Krawiec K (2014) Semantic backpropagation for designing search operators in genetic programming. *IEEE Transactions on Evolutionary Computation* 19(3):326–340
- Pedregosa F, Varoquaux G, Gramfort A, et al (2011) Scikit-learn: Machine learning in python. *the Journal of machine Learning research* 12:2825–2830
- Pennachin C, et al (2010) Improved time series prediction and symbolic regression with affine arithmetic. *Genetic Programming and Evolvable Machines* 11(3-4):295–312. Evaluates on Keijzer benchmark functions, including Keijzer5
- Petersen B, Landajuela M, Mundhenk T, et al (2021) Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. arXiv preprint arXiv:210810751
- Petersen BK, Landajuela M, Mundhenk TN, et al (2020) Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. arXiv preprint arXiv:191204871
- Qi X, et al (2023) Relieving coefficient learning in genetic programming for symbolic regression. In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, uses Keijzer5 and Korn8 synthetic benchmarks
- Razavi S, Gamazon ER (2022) Neural-network-directed genetic programmer for discovery of governing equations. arXiv preprint arXiv:220308808

- Ruberto S, Terragni V, Moore JH (2020) Sgp-dt: Semantic genetic programming based on dynamic targets. In: European Conference on Genetic Programming (Part of EvoStar), Springer, pp 167–183
- Ruberto S, Terragni V, Moore JH (2021) Towards effective gp multi-class classification based on dynamic targets. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp 812–821
- Sahoo S, Lampert C, Martius G (2018) Learning equations for extrapolation and control. In: Proceedings of the International Conference on Machine Learning, pp 4442–4450
- Sareni B, Krahenbuhl L (1998) Fitness sharing and niching methods revisited. *IEEE transactions on Evolutionary Computation* 2(3):97–106
- Searson DP, Leahy DE, Willis MJ (2010) Gptips: an open source genetic programming toolbox for multigene symbolic regression. In: Proceedings of the International multiconference of engineers and computer scientists, Citeseer, pp 77–80
- Silva S, Munoz L, Trujillo L, et al (2016) Multiclass classification through multidimensional clustering. In: Genetic Programming Theory and Practice XIII. Springer, p 219–239
- Stanley KO, Miikkulainen R (2002) Evolving neural networks through augmenting topologies. *Evolutionary computation* 10(2):99–127
- StatLib (2025) Statlib dataset repository. <http://lib.stat.cmu.edu/>, accessed: July 2025
- Tenachi W, Ibata R, Diakogiannis FI (2023) Deep symbolic regression for physics guided by units constraints: Toward the automated discovery of physical laws. *The Astrophysical Journal* 959(2):99
- Trujillo L, Muñoz L, Galván-López E, et al (2016) neat genetic programming: Controlling bloat naturally. *Information Sciences* 333:21–43
- Uy NQ, Hoai NX, O’Neill M, et al (2011) Semantically-based crossover in genetic programming: application to real-valued symbolic regression. *Genetic Programming and Evolvable Machines* 12:91–119
- Vastl M, Kulhánek J, Kubalík J, et al (2022) Symformer: End-to-end symbolic regression using transformer-based architecture. *arXiv preprint arXiv:220515764*
- Virgolin M, Bosman PA (2022) Coefficient mutation in the gene-pool optimal mixing evolutionary algorithm for symbolic regression. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp 2289–2297

- Wittenberg D, Rothlauf F, Schweim D (2020) Dae-gp: Denoising autoencoder lstm networks as probabilistic models in estimation of distribution genetic programming. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp 1037–1045
- Yang Y, Chen G, Ma H, et al (2024) Dual-tree genetic programming with adaptive mutation for dynamic workflow scheduling in cloud computing. *IEEE Transactions on Evolutionary Computation*
- Yeh IC (1998) Concrete compressive strength. <https://doi.org/10.24432/C5PK67>
- Yeh IC (2018) Real estate valuation. <https://doi.org/10.24432/C5J30W>
- Zhong J, Lin Y, Lu C, et al (2018) A deep learning assisted gene expression programming framework for symbolic regression problems. In: Proceedings of the Neural Information Processing: 25th International Conference, pp 530–541
- Zhong L, Zhong J, Lu C (2021) A comparative analysis of dimensionality reduction methods for genetic programming to solve high-dimensional symbolic regression problems. In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, p 476–483
- Zojaji Z, Ebadzadeh MM (2018) Semantic schema modeling for genetic programming using clustering of building blocks. *Applied Intelligence* 48:1442–1460
- Žegklitz J, Posík P (2020) Benchmarking state-of-the-art symbolic regression algorithms. *Genetic Programming and Evolvable Machines* 22:5–33