Randomer Forests

Tyler M. Tomita*
Department of Biomedical Engineering
Johns Hopkins University
Baltimore, MD
ttomita@jhu.edu

Mauro Maggioni
Department of Mathematics
Duke University
Durham, NC
mauro@math.duke.edu

Joshua T. Vogelstein
Department of Biomedical Engineering
Johns Hopkins University
Baltimore, MD
jovo@jhu.edu

Abstract

jovo says: please make this submittable asap. this means include bib, put figures in the right place, anonymize, etc.

1 Introduction

Data science is becoming increasingly important as our ability of a society to collect and process data continues to increase. Supervised learning—the act of using predictors to make a prediction about some target data—is of special interest to many applications, ranging from science to government to industry. Classification, a special case of supervised learning, in which the target data is categorical, is one of the most fundamental learning problems that has been the subject of much study. A simple pubmed search for the term "classifier" reveals nearly 9,000 publications, and a similar arxiv search reports that the number of hits is $>$1000. Of all the classifiers, random forests (RFs) [**?**], is generally considered to be best, with good reason. Several recent benchmark papers assess the performance of many different classifiers on many different datasets [**?**, **?**], and both concluded the same thing: random forest are the best classifier.

The reasons for the popularity and utility of random forests are many. Below, we list some of the primary reasons, in order of approximate importance (as assessed subjectively and qualitatively by us):

- **empirical performance**: doing well in a wide variety of contexts
- **scale invariant**: this means that different predictor variables can have totally different units, millimeters and kilometers and milligrams and kilograms, for example, and RF does not care.
- **robust to outliers**: certain data points could be outliers, either because some or all of the values of their feature vector are far from the others
- **computational efficiency**: it is reasonably efficient, moreso than an exhaustive search to find the globally optimal tree, which is NP-hard [**?**]
- **storage efficiency**: when training on lots of data, storage of the classifier can become problematic
- **interpretability**: it is reasonably interpretable, as each variable can be assigned an importance score (such as the "gini" score)

*Use footnote for providing further information about author (webpage, alternative address)—*not* for acknowledging funding agencies.

Although the benefits of RF are many, as with everything, there is room to improve. The main drawback of using RFs, in our opinion, is its sensitivity to rotations. Because when people refer to RFs they often mean Breiman's Forest-IC, which uses axis-parallel [**?**], or othogonal trees [**?**], Breiman also characterized Forest-RC, which used linear combinations of coordinates rather than indivudal coordinates, to split along. Others have studied several different variants "oblique" random forests, including efforts to learn good projections [**?**, **?**], or using principal components analysis to find the directions of maximal variance [**?**, **?**], or directly learn good discriminant directions [**?**]. While each of these approaches deal with rotational invariance, they all also lose at least one of the above benefits of RF, namely scale invariance. The scale invariance property of RF is one of its most important and special properties, distinguishing it from most other classifiers, and making it appropriate in a wider variety of contexts. Moreover, all of the above approaches become outlier sensitive, and lose computational and storage efficiency. Therefore, in the literature, there remains a gap to build a classifier that has excellent empirical performance, while maintaining scale invariance, as well as robustness and computational and storage efficiency.

To bridge this gap, we first state a generalized forest building scheme, linear threshold trees, which includes all of the above algorithms as special cases. This enables us to formally state our objective, and provides a lens that enables us to propose a few novel special cases, which we refer to as "Randomer Forests" (or RerFs for short), for reasons that will become clear in the sequel. We both theoretically and empirically demonstrate that our methods have the same time and space complexity as random forests, while matching or exceeding their performance even on axis aligned data. Moreover, we demonstrate that a variant of RerF is approximately is both affine invariant and robust to outliers, two properties that are relatively easy to obtain independently, though difficult to obtain jointly (though see recent work on robust PCA, for example [**?**]). Finally, we demonstrate on a suite of benchmark datasets, that RerF outperforms RF in terms of both accuracy and interpretability. We conclude that RerF should be the new reference classifier.

2 Linear Threshold Tree Ensembles

Input:

- **Data**: $\mathcal{D}_n = (X_i, y_i) \in (\mathbb{R}^p \times \mathcal{Y})$ for $i \in [n]$
- **Number of trees**: L
- **Data sampling strategy**: a rule for how to obtain $\mathcal{S}_l$, the subset of $[n]$ per tree l
- **Subspace**: $\mathcal{A}^{p \times d}$
- **Distribution on $k \times d$ matrices**: $f_A(\mathcal{D}_n)$, where $k = |\mathcal{S}_l|$
- **Stoppping criteria**: e.g., a maximal tree depth or minimum number of elements per tree

Output: trees, predictions, out of bag errors, etc.

```
Preprocess
for l ∈ [L] do
    Sample S_l
    Construct X^(l) = {X_i}_{S_l}
    for Each leaf node in tree do
        Sample A_{i,j} ~ f_A(D_n)
        Let X̃ = A_{i,j}^T X^(l) ∈ R^{d×|S̃_l|}
        variable selection rule
    end for
end for
```

---

3 Randomer Forests

A random forest is an ensemble of randomized decision trees, where each decision tree consists of internal (split) and terminal (leaf) nodes. At each internal node, a random subset of the ambient (coordinate) dimensions is sampled and subsequent optimization of a split function selects the single best dimension and split point contained in this subset. Therefore, each tree within a random forest can be viewed as a hierarchical series of axis-aligned decision boundaries. Here, we introduce a new method for constructing randomized decision trees, where instead of random subsampling

the dimensions, we randomly project to a lower dimensional subspace. By randomly projecting, decision boundaries are not restricted to alignment with the coordinate axes. Specifically, let $X_i \in \mathbb{R}^{n \times p}$ be a set of n points in p dimensional Euclidean space associated with the ith split node of a decision tree, and let $R_i \in \mathbb{R}^{p \times k}$ be a randomly chosen projection matrix at the ith node. Then for every split node, we compute the projection $X_{proj} = X_i R_i$ and select the optimal split dimension and split point in this randomly chosen k-dimensional subspace.

There exists various choices for the random projection matrix R. If one cares about preserving pairwise distances between points, then R must be chosen carefully. A variety of methods have been established recently for generating random projection matrices that preserve distances. The most common one is to sample each element r_{ij} from a standard normal (Indyk and Motwani, 1998; Dasgupta and Gupta, 2003). Another method generates sparse matrices by setting each element equal to +1 or -1 with probability $1/2\sqrt(p)$ and 0 with probability $1 - 1/\sqrt(p)$ (Li 2006). In this work, we are not concerned with preserving pairwise distances. Rather, we are simply concerned with constructing features at each split node of a tree using linear combinations of the ambient dimensions. We construct random matrices by first, sampling the indices of the k nonzero elements uniformly at random. Next, each nonzero element is randomly assigned -1 or +1 with equal probability. Columns containing all zeros are removed. This results in a matrix with many columns having one nonzero element and a few columns having more than one nonzero element.

jovo says: we need proper pseudocode table, make it sufficient general such that all the variants under consideration are special cases.

In addition to random projections, for classification problems involving c classes we can compute a set of $c-1$ differences in class-conditional means $\{\boldsymbol{\delta_{ij}} = \boldsymbol{\mu_j} - \boldsymbol{\mu_i} : \forall i \in [1, ..., c-1], j \in [2, ..., c], j > i\}$ and sample each of them as a projection with probability k/p. Such supervised projections work especially well when all classes are normally distributed with diagonal covariance matrices, and are often useful even when these conditions are not met. We denote RerFs utilizing these projections as RerF(δ)

When taking linear combinations of ambient dimensions, it is important that the dimensions be scaled similarly. Therefore we also formulate a variant of RerF in which input data is passed to marginal ranks prior to training. Doing so not only scales the dimensions appropriately, but also can improve performance when outliers are present. This robust variant of randomer forest will be denoted as RerF(r).

4 Experimental Evaluation

4.1 Simulated Data

We constructed three synthetic datasets (Trunk, parity, and multimodal) to compare classification performance (Fig 1*jovo says: use fig labels to refer to figs*) and training time (Fig 2) of RerF, RerF(δ), and RerF(δ+r) with that of random forest. Trunk is a well-known binary classification (cite Trunk) in which each class is distributed as a p-dimensional multivariate gaussian with identity covariance matrices. The means of the two classes are $\mu_1 = (1, \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{3}}, ..., \frac{1}{\sqrt{p}})$ and $\mu_2 = (-1, -\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{3}}, ..., -\frac{1}{\sqrt{p}})$. Parity is a binary classification problem in which each of the two classes is distributed as a mixture of 2^{p-1} multivariate gaussians with covariance matrices $\boldsymbol{\Sigma} = \frac{1}{32}\boldsymbol{I}$. The means of class one are the subset of $[0,1]^p$ for which the number of zeros in a mean is even. The means of class two are the subset of $[0,1]^p$ for which the number of zeros in a mean is odd. Multimodal is a four-class classification problem. Each of the four classes is distributed as an equal mixture of two multivariate gaussians. The means are randomly sampled from a p-dimensional multivariate standard gaussian and the covariance matrices are sampled from an inverse Wishart distribution with $10p$ degrees of freedom and a mean covariance matrix equal to the identity matrix. The bottom panels of Figure 1 depict scatter plots of the data for $p = 2$. The top panels depict average misclassification rate over ten trials estimated using the out of bag error. For the trunk and multimodal simulations, 100 points were sampled and p varied from two to 1000 dimensions. For the parity simulation, 100 points were sampled and p varied from two to ten. The appropriate number of trees trained for RF and all variants of RerF for trunk, parity, and multimodal were empirically determined to be 1500, 1000, and 500 respectively (Fig A1). Bayes error is also included

for reference. In binary classification problems in which each class consists of a single multivariate gaussian with equal covariance matrices, such as the Trunk simulation, Bayes error can be computed analytically (Bickel and Levina 2004). In the Parity and Multimodal simulations, Bayes error was estimated by averaging the misclassification rate of the Bayes optimal classifier on 1000 points over ten trials.

Panel A of Figure 1 shows that RerF(δ) and RerF(δ+r) outperform RF across all numbers of dimensions. This can be attributed to projection onto the difference in means. All variants outperform RF up to approximately 250 dimensions. Above this, misclassification rate of RerF is comparable to RF. In Panel B, we observe that all variants of RerF outperform RF for all numbers of dimensions. This can be understood by observing that for p dimensions, all splits in RF up to a tree depth of $p-1$ will result in daughter nodes having chance posterior probabilities of being in either class. Therefore, any splits up to this depth that are not aligned with the coordinate axes can only be better. Multimodal???

In Figure 2, we observe a slight increase training time of all RerF variants compared to RF. Despite this increase, training time for all classifiers scales similarly with the number of dimensions. The apparent increase in training time for the RerF variants is largely due to sampling of the random projection matrices.

4.2 Effects of Transformations and Outliers on Classifier Performance

RF does especially well in classification problems in which the optimal decision boundaries are aligned or nearly aligned with the coordinate axes. Rotations in such situations can lead to a decrease in classification performance. Naturally, we wanted to examine the effects of various transformations on classification performance of RerF. These effects were examined using the Trunk and parity simulations as previously described. The transformations we applied were random rotations, scaling, and general affine transformations. Uniformly random rotation matrices were generated by first performing SVD on a p-dimensional matrix in which each element is sampled from a multivariate standard normal distribution. The rotation matrix was taken as the right singular vectors of this SVD. If the determinant of this matrix was equal to -1, the first two columns were permuted to render the determinant equal to $+1$. Random scaling was performed by applying to each dimension a scaling factor sampled from a uniform distribution on the interval [0,10]. Affine transformations were performed by applying a combination of rotations and scalings as just described. Additionally, we examined the effects of introducing outliers. Outliers were introduced to Trunk and parity simulations by sampling points from the distributions as previously described but instead using covariance matrices scaled by a factor of four. Empirically, an addition of 20 points from these outlier models to the original 100 points was found to produce a noticeable but not overwhelming effect on classifier performance. The classifiers evaluated were RF, RF(s), RF(s+δ). Additionally, Fisherfaces was evaluated as a reference. The misclassification rate for Fisherfaces was estimated using leave-one-out cross validation.

The top panels of Figure 4 illustrate the effects of the transformations and outliers on the Trunk simulation. The bottom panels show these effects on the parity simulation. In the Trunk simulation, rotation results in noticeable degradation in classification performance of RF when the number of dimensions is greater than approximately 100. On the other hand, both RerF(δ) and RerF(δ+r) are unaffected by rotations. RerF(δ) exhibits an increase in misclassification rate when scaling is applied. This is expected because ambient dimensions with a relatively large scale will dominate the variance of new dimensions constructed from a linear combination of the ambient ones. For this same reason, Fisherfaces is also affected by scaling. Since RerF(δ+r) maps all dimensions to the same scale, it is invariant to scaling and is also the only classifier to exhibit invariance to affine transformations. We also observe that all classifiers are only slightly affected by the addition of outliers to the Trunk simulation. As shown in panel E, rotations in the Parity simulation actually improve classification performance of RF. As mentioned previously, all splits in RF up to a tree depth of $p-1$ in the untransformed parity simulation result in chance probabilities of being in either class in the daughter nodes. Therefore, rotating can only improve performance. As in the Trunk simulation, performance of RerF is hurt when scaling is applied in the parity simulation.

4.3 Real Data

In addition to the simulations, RF, RerF, RerF(δ), and RerF(δ+r) were evaluated on 121 datasets as described in []. Classifiers were trained on the entire training sets provided. For each data set, misclassification rates were again estimated by out of bag error, and training time was measured as wall clock time. Misclassification rates and training times were averaged over all 121 datasets (Fig 4).

5 Conclusion

We have proposed a novel method for constructing ensemble classifiers. Like random forests, our method constructs an ensemble of randomized decision trees. However, by randomly projecting the data at each split node, partitions are not restricted to alignment with the coordinate axes. We have constructed datasets demonstrating settings in which RerF outperforms RF. Furthermore, one of the variants, RerF(δ), exhibits robustness to affine transformations, a property lacking in RF.

Much work is still to be done with our proposed method. In this work, we only examined one method of constructing sparse random projection matrices. It is possible that other choices of construction will lead to improved performance in certain settings. Additionally, it will be useful to evaluate the sensitivity of our method to the use of different split criteria, pruning, and other parameters of the decision tree. It will also be of interest to establish consistency theorems for our method. While we only restricted our attention to classification thus far, our method can be generalized to other types of learning problems, such as regression, density estimation, etc.

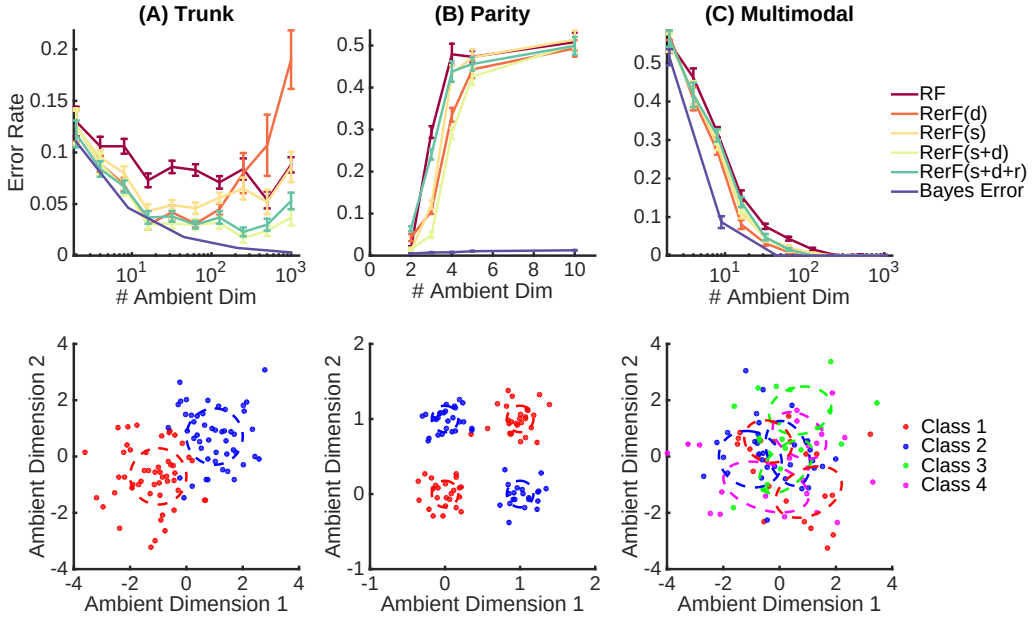

Figure 1: Classification performance comparing Random Forest (RF) to several variants of Randomer Forest (RerF), and Bayes optimal performance, on three distinct simulation settings: (A) Trunk, (B) Parity, and (C) Multimodal (see Methods for details). For all settings, the top panel depicts misclassification rate vs. the number of ambient (coordinate) dimensions, and the bottom panel shows a 2D scatter plot of the first 2 coordinates (dashed circles denote the standard deviation level set). Note that in all settings, for all number of dimensions, RerF outperforms RF, even Trunk and Parity, which were designed specifically for RF because the discriminant boundary naturally lies along the coordinate basis.

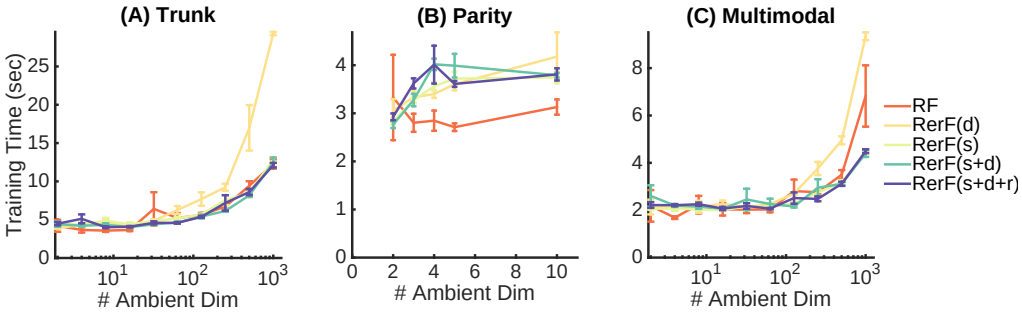

Figure 2: Classifier training time comparing RF to several variants of RerF, same setting as the top row of Figure 1. The only difference is that the y-axis here labels training time (in seconds). Although RerF requires slightly more time than RF (largely due to random sampling of projection matrices), they scale similarly.

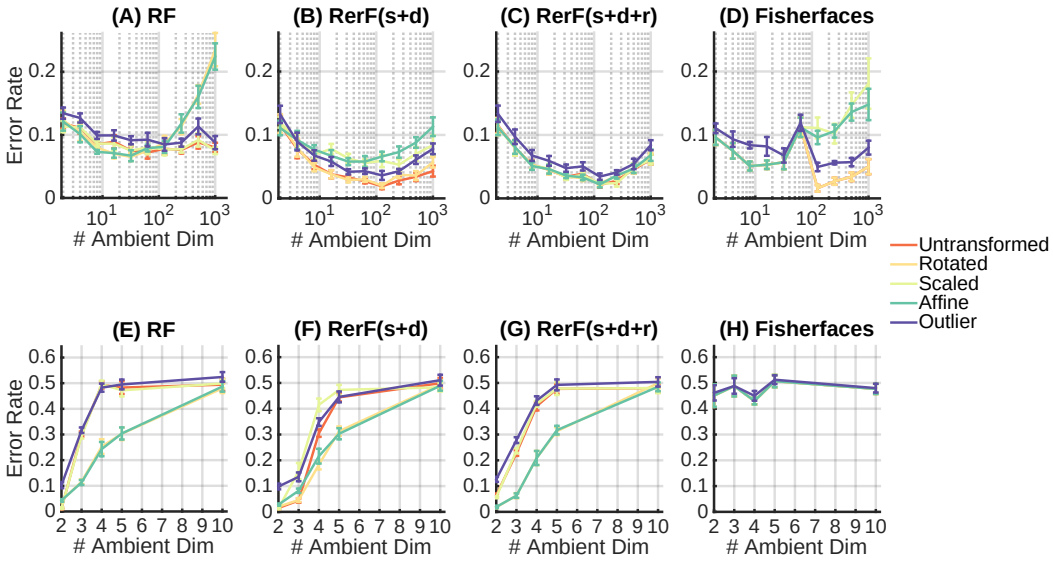

Figure 3: The effect of various transformations applied to the Trunk (panels A-D) and Parity (panels E-H) simulations (see Methods for details) on classification performance of (A,E) RF, (B,F) RerF (s+d), (C,G) RerF (s+d+r), and (D,H) Fisherfaces. Specifically, we consider, rotations, scalings, and affine transformations, as well as introducing outliers. Classification performance of RF is compromised by rotations and therefore affine transformations as well. RerF(s+d) is invariant to rotation, but not scaling and therefore not affine transformation. RerF(s+d+r) is invariant to to affine transformations. Like RerF (s+d), Fisherfaces is invariant to rotation but not scaling. Note that all variants are reasonably robust to outliers.

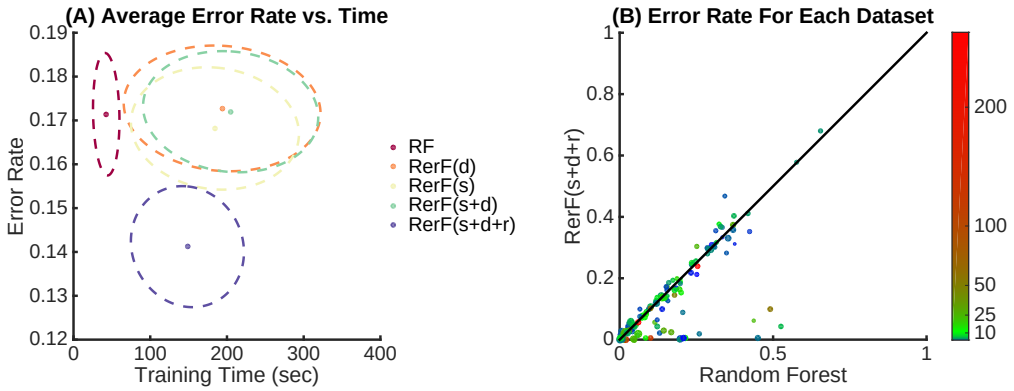

Figure 4: (A) Classification error and training time of Random Forest and Randomer Forests variants for all 121 datasets from the XXX benchmark comparison paper [**?**]. (A) Average (dots) and 0.1 standard deviation level sets (dashed lines) for each method. (B) Classification error of Randomer Forest (sparse+delta+robust) vs. that of Random Forest for each of the 121 datasets. The black line indicates equal classification error of the two algorithms. Color indicates dimensionality of the datasets and size of points indicates number of samples. Note that RerF almost always does as well, and often significantly better.

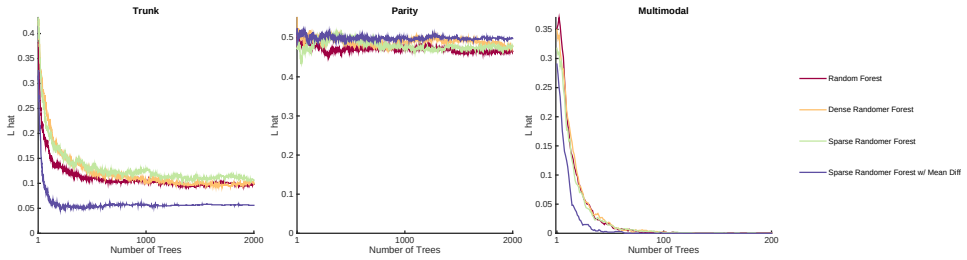

Figure A1: The number of trees until a stable misclassification rate is achieved for RF and the RerF variants in the Trunk, parity, and multimodal simulations. Simulation settings are the same as in Fig1. The number of ambient dimensions for panels A, B, and C were chosen to be 1000, 10, and 1000 respectively.

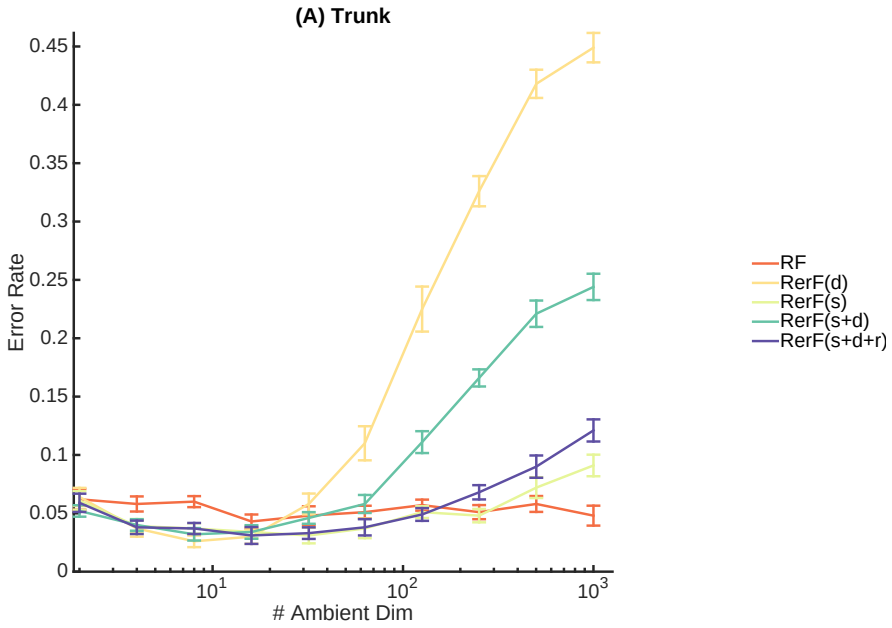

Figure A2: Classification performance comparing Random Forest (RF) to several variants of Randomer Forest (RerF) on the modified Trunk simulation setting. Here, the ith diagonal of the covariance matrices of both classes is equal to the inverse of the difference in means of the ith dimension between the two classes. For small values of p, all variants of RerF perform marginally better than RF. For large values of p, all variants of RerF perform worse than RF.