PCA-Enhanced Autoencoders for Nonlinear Dimensionality Reduction in Low Data Regimes

Muhammad Al-Digeil[†,*], Yuri Grinberg[†], Daniele Melati[◇],
Jens H. Schmid[‡], Pavel Cheben[‡], Siegfried Janz[‡],
Dan-Xia Xu[‡]

[†] Digital Technologies Research Centre, National Research Council of Canada, Canada
[‡] Advanced Electronics and Photonics Research Centre, National Research Council of Canada, Canada
[◇] Université Paris-Saclay, CNRS, France

Abstract

Many scientific domains, such as nanophotonic design, gene expression, and materials design, are limited by high costs of acquiring data. This data is often intrinsically low-dimensional, nonlinear, and benefits from dimensionality reduction. Autoencoders (AE) provide nonlinear dimensionality reduction but are typically ineffective for low data regimes. Principal Component Analysis (PCA) is data-efficient but limited to linear dimensionality reduction. We propose a technique that harnesses the benefits of both methods by using PCA to initialize an AE. The proposed approach outperforms both PCA and standard AEs in low-data regimes and is comparable to the best of either of the two in other scenarios.

Keywords: Dimensionality Reduction, Autoencoders, Principal Component Analysis (PCA), Limited Datasets

1. **Introduction**

While the proliferation of large datasets has been a driving force behind the progress and success of machine learning methods, many application domains remain constrained by relatively small low-dimensional datasets. For instance, collecting useful data can be computationally expensive in domains, like photonics design [1], that rely on simulation-based-optimizations interacting with physics equation solvers. Gene expression data is often obtained from experimental processes with very limited data due to costs and ethical considerations [2]. Materials design problems can involve running computationally intensive models as well as expensive experiments, all resulting in small useful datasets [3–5]. Yet, these datasets can often benefit from dimensionality reduction techniques to analyze the low dimensional space for future experiment design or to assist in some other downstream tasks.

Principal Component Analysis (PCA) [6] is a classical dimensionality reduction approach with provably optimal performance for linearly dependent data [7]. It is often the first, if not the only, method used for datasets where the number of features is smaller than the number of data points ($n < m$). However, its efficiency clearly suffers when data exhibits curvature as PCA cannot distinguish between nonlinear structure and lack of structure in data. In many domains where dimensionality reduction is desired, the data is inherently nonlinear and therefore nonlinear methods are in principle expected to yield superior results.

Autoencoders [8] are commonly used for nonlinear dimensionality reduction as they offer a convenient way of controlling the reduction by setting network's bottleneck size. However, their typical data requirements are significantly larger than PCA when trained with randomly initialized weights. Yet, it is well known that the behaviour of PCA with p components can be modeled using an autoencoder with a bottleneck of the same number of neurons and linear activation functions [9]. In this paper, we explore the use of PCA to

[*] muhammad.al-digeil@nrc-cnrc.gc.ca

initialize the weights of an autoencoder in a numerically stable way. Specifically, we propose PCA-Robust autoencoders (PCA-Robust) which, given a PCA dimensionality reduction to p dimensions, are initialized with PCA-derived weights with linear activation functions and a bottleneck of size p thus replicating PCA's behavior. The activation function is then allowed to change smoothly to a nonlinear one as part of the normal training process, enabling the network to capture nonlinear behavior to the extent that the data supports it.

Another work has previously explored a different PCA-based initialization of autoencoders [10]. However, the proposed initialization is a heuristic, and the main objective of that work is to speed up a slow training process on large datasets rather than improve performance on scarce data. Linear variational autoencoders to address data scarcity problems in collaborative filtering was previously proposed [11]. However, this model cannot capture nonlinear relationship between parameters. Allerbo and Jörnsten [12] present techniques for inducing sparsity in dimensionality reduction using autoencoders. However, the sparsity regularization is meant to primarily address interpretability issues rather than improve projection error performance. Also, sparsity regularization introduces extra parameters that require tuning and is therefore less suitable to problems with small datasets.

Various other nonlinear dimensionality reduction methods exist such as t-SNE [13], Kernel PCA [14], Locally Linear Embedding [15], Isomap [16], etc. (see [17]). In this study we focus on autoencoders for several reasons. First, they provide a seamless way of reconstructing data from the reduced to the original space, a necessary requirement of its application to nanophotonic design. Second, autoencoders offer a built-in way to map out-of-sample points on the lower dimension. This paves the way for developing adaptive data acquisition schemes that minimize the number of samples required to achieve a desired level of accuracy. Finally, Neural Network implementations are widely available, making their use easily accessible to communities outside the research domain.

We present the methodology as well as the algorithmic details of the robust PCA-based initialization. To study the impact of the proposed approach we first analyze its performance on carefully designed synthetic data where we can control the degree of nonlinearity. We then follow up with a variety of datasets coming from the nanophotonic component design domain as well as gene expression data and a benchmark breast cancer dataset. Our results demonstrate that there exists a synergy between linear and nonlinear methods in that the proposed approach improves upon both baselines in most low data regimes while being comparable to the best performing method in all other situations.

2. PCA-Enhanced Autoencoders

A central piece of our method is the use of the Parametric ReLU (PReLU, Figure 2) activation function for the autoencoder [18]. While the original motivation for this activation function was to improve the training process in large networks, we adopt it for different reasons. PReLU allows for the smooth transition from linear to nonlinear function approximation [19]. When we manually set the slope for the negative values of ReLU to 1 the network represents a linear transformation. In fact, emulating PCA is the best this network can do since PCA gives the optimal linear dimensionality reduction under the L_2 loss function. The amount by which the slope for the negative values is adjusted becomes another parameter to be estimated from data [18]. PReLU, as it is discussed here, is distinct from Leaky ReLU. With Leaky ReLU the α slope (Figure 2) is configured manually and in PReLU α is treated as an independent learnable parameter per node during training and is adjusted by the learning algorithm.

We assume that the architecture of the AE takes the vase-shaped form shown in Figure 1, where first the data is expanded in its dimensionality and then reduced back to the original dimension n before being reduced to the bottleneck dimension q. Although not necessary, it makes the robust weights initialization procedure easier to implement and present, while still allowing a significant degree of flexibility in the choice of the architecture.

For the n-dimensional signal, let $\mathbf{X}$ be the original data matrix with n columns (features) and m rows (samples). We assume that $\mathbf{X}$ is already centered and possibly scaled, depending on the application needs. Let $\hat{\mathbf{X}}$ be the result of representing $\mathbf{X}$ using just the first q linear principal components. That is,

$$\hat{\mathbf{X}} = \mathbf{U}^{m\times q} \cdot \mathbf{S}^{q\times q} \cdot {\mathbf{V}^{n\times q}}^{\top},$$

being a rank$-q$ singular value decomposition of matrix $\mathbf{X}$.

The functionality of a linear Autoencoder as in Figure 1 with the bottleneck dimension q can be described by the following equation:

$$\begin{gathered}\mathbf{x}^{\top} \cdot \mathbf{W}_{e_1}^{n\times\cdot} \cdots \mathbf{W}_{e_i}^{\cdot\times n} \cdot \mathbf{W}_{enc}^{n\times q} \cdot \mathbf{W}_{dec}^{q\times n} \cdot \mathbf{W}_{d_1}^{n\times\cdot} \\ \cdots \mathbf{W}_{d_i}^{\cdot\times n} = \hat{\mathbf{x}}^{\top}\end{gathered}$$

where each $\mathbf{W}^{r\times s} \in \mathbb{R}^{r\times s}$ represents the weights of a layer of the AE having r inputs and s nodes, and subscript identifies the layer itself (e_i, d_i - encoder and decoder layers respectively, enc, dec the bottleneck layers of the encoder and decoder respectively). We seek to generate the weights such that the following equality holds:

$$\begin{gathered}\mathbf{X} \cdot \mathbf{W}_{e_1}^{n\times\cdot} \cdots \mathbf{W}_{e_i}^{\cdot\times n} \cdot \mathbf{W}_{enc}^{n\times q} \cdot \mathbf{W}_{dec}^{q\times n} \cdot \mathbf{W}_{d_1}^{n\times\cdot} \\ \cdots \mathbf{W}_{d_i}^{\cdot\times n} = \hat{\mathbf{X}}, \end{gathered} \tag{2.1}$$

implying that the AE acts as PCA.

The above equation implies that there is no unique solution to the initialization of weights due to both larger number of degrees of freedom (parameters) as well as scaling. This creates different possibilities for the initialization. We consider two options, both allowing a significant degree of randomness. The simplified version initializes all the layers randomly except for the bottleneck (any random distribution is acceptable as long as it generates full rank matrices with probability 1), then computes the bottleneck layers (last encoder layer and first decoder layer) to match PCA. Let $\mathbf{W}_{enc-} = \prod_j \mathbf{W}_{e_j}$ and $\mathbf{W}_{dec+} = \prod_j \mathbf{W}_{d_j}$ be the weights corresponding to the product of all the encoder layers' weights except for the bottleneck, and the product of all the decoder layers' weights except for the bottleneck respectively. While $\mathbf{W}_{enc-}$ and $\mathbf{W}_{dec+}$ are products of random matrices, the bottleneck layer weights are calculated as follows:

$$\mathbf{W}_{enc} = \mathbf{W}_{enc-}^{-1} \cdot \mathbf{V};\ \mathbf{W}_{dec} = \mathbf{V}^{\top} \cdot \mathbf{W}_{dec+}^{-1}. \tag{2.2}$$

It is easy to see that Eq. (2.1) holds as a result. We call this procedure PCA-Naive.

The second, more involved initialization option that we use in our implementation ensures that all the intermediate AE layers are non-expansive and non-contractive operators with respect to the original input, i.e. they maintain the magnitude of the input vector, with the exception of the bottleneck layers only. It is implemented by sampling random orthogonal matrices in a correct order and solving linear systems to stitch those matrices together. This ensures the numerical stability of the neural network, making the generalization less sensitive to the initialization process. The algorithm is outlined in Alg. 1 .

3. Experiments

The design of the experiments is intended to replicate a low-data regime with the downstream tasks requiring the choice of a single dimensionality reduction model that is later explored. The exploration of the latent space can be accomplished by sampling using reinjections [20]. The total available data in our experiments is split into 80% for training, 10% for validation, and 10% for model selection, where the validation set is used to identify when

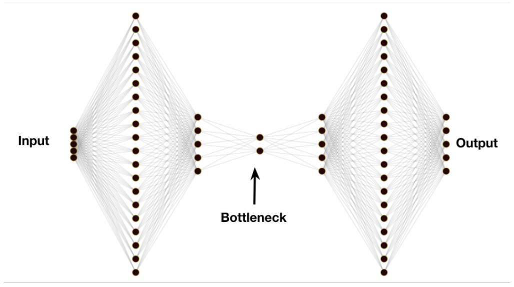


Figure 1. AutoEncoder architecture used in experiments

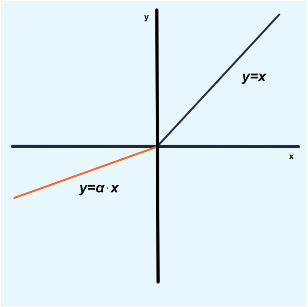


Figure 2. PReLU, $\alpha = 1$ initially and varies independently for each node in training

Algorithm 1 Stable PCA-based weights initialization

Parameters:

$\mathbf{X} \in \mathbb{R}^{m \times n}$ - centered (optionally scaled) data matrix

k - bottleneck dimension

$arch$ - list of dimensions for encoder/decoder layers

Output: List of weights

```
## Robust weight initialization
function RWI($\mathbf{A} \in \mathbb{R}^{m \times n}, arch$ : List)
    if size of $arch \leq 2$ then
        return $\mathbf{A}$
    Remove last element from $arch$
    $\mathbf{B}$ = Call RO(first and last elements of $arch$)
    $\mathbf{W} = \mathbf{B}^{\dagger} \cdot \mathbf{A}$
    return List(Call RWI($\mathbf{B}$,$arch$),$\mathbf{W}$)

## Random orthonormal matrix
function RO($m, n$): return random orthonormal matrix $\in \mathbb{R}^{m \times n}$

$\mathbf{A}$ =RO($n, n$)
Enc =RWI($\mathbf{A}, arch$)                                ▷ encoder weights init
$\mathbf{W}_{enc-} = \prod \mathbf{Enc}$
Dec =RWI($\mathbf{A}, arch$)                                ▷ decoder weights init
$\mathbf{W}_{dec+} = \prod \mathbf{Dec}$
$\mathbf{P}$ = PCA($\mathbf{X}, k$)
return List($\mathbf{Enc}, \mathbf{W}_{enc-}^{-1}\mathbf{P}, \mathbf{P}^{-1}\mathbf{W}_{dec+}^{-1}, \mathbf{Dec}$)
```

to stop the training process. The model selection set is used to choose the best performing AE model among several randomly initialized and trained AE models. The same model selection is also applied to the AE models that reproduce PCA since this process allows significant degree of randomness as well. All final models are evaluated on a much larger testing set for analysis purposes.

Training is performed using the Adam optimizer with the slope of PReLU activation function for all the layers initialized to 1. Along with the PCA, the compared AE models are the PCA-Robust initialization (Alg. 1), PCA-Naive initialization and Random initialization. Although it was tracked during all the experiments, PCA-Naive is not depicted in the charts because it performed poorly—consistently worse and significantly less stable than the other methods. The Euclidean distance between the input and output vectors is used as the loss function, consistent with an implicit objective of PCA.

We also considered the possibility that simple orthogonalization of the initial random weights may have accounted for some gains. However, in our experiments with synthetic data we observed no statistically significant difference between random initialization and orthogonalized random initialization. Hence this possibility was not pursued further.

All autoencoder architectures we experiment with have 7 layers including the input and output layers. The choice of this architecture is somewhat arbitrary and was not optimized for any of the problems.

For our experiments we used the ADAM optimizer with a custom learning rate schedule. We found a learning rate of 10^{-4} with an exponential decay of 0.99 per epoch yielded the best results. We ran each training run for a maximum of 1000 epochs. The experiments were run on a node with dual Intel Xeon Gold 6130 CPUs clocked at 2.1GHZ. The node's available memory is 192GB. The experiments, submitted as jobs to the node, ran on CPU and consumed no more than 32GB of memory per run.

Finally, both synthetic and real experiments are repeated with different total sample sizes (20, 30, 40, 50, 80 and 100), representing varying sizes of the low data regimes.

3.1. **Power Function**

In our first experiment the data is generated from a function that takes two parameters, satisfying:

$$x^n + y^n = z \tag{1}$$

As shown in Figure 3 , adjusting n allows for the manipulation of the degree of curvature of the surface while maintaining the same range of a function for a fixed domain $x, y \in [0, 1]$. As n approaches 1 the surface approximates a linear surface more closely.

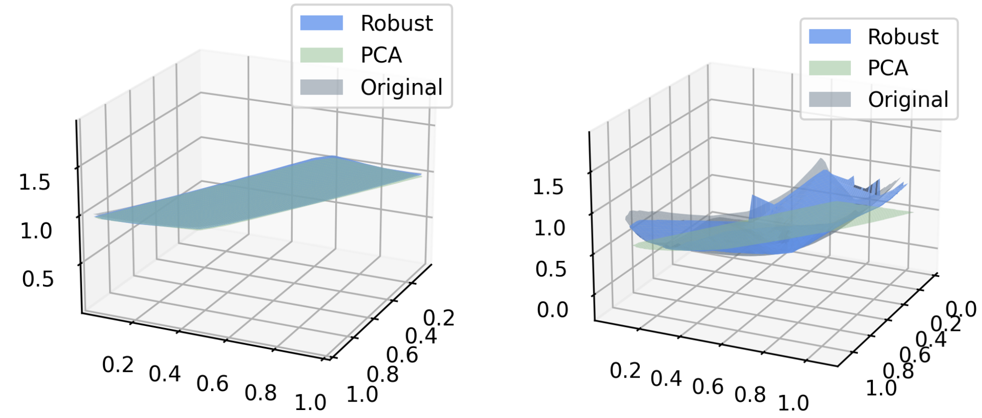
In this experiment we study the effect of curvature and sample size on the respective performances of PCA, a Randomly-initialized AE and two AE's: one initialized with our stable algorithm (PCA-Robust), and one initialized naively without consideration for the stability issues discussed in the previous section (PCA-Naive). We vary the sample sizes (using 20, 30, 40, 50, 80 and 100) and different values for n in equation (1) (n= 1.1, nearly linear , Figure. 3a-left, and, n=4, clearly curved, Figure. 3b-right). Altogether, these values cover a range of curvature and data availability of the latent 2D subspace.

1000 points are randomly generated for each value of n and from these 250 are designated the test set. The rest are used to draw training samples from. For each data sample, the AEs were each initialized accordingly. All three autoencoders (PCA-Robust, PCA-Naive and Random) undergo the same training process on the given sample. Finally, the experiment is repeated 200 times (resulting in different data samples) and the statistics are presented.

For the synthetic experiments, the layer sizes in sequence from the input layer to the output layer are 3-20-3-2-3-20-3, based on 3-dimensional data.

Figure 4 shows the average errors of different models evaluated on the test data (size=250) as a function of the training data size. The results show that PCA-Robust performs significantly better than PCA for all cases. The result for Random for $n = 1.1$ are not shown as they are significantly worse.

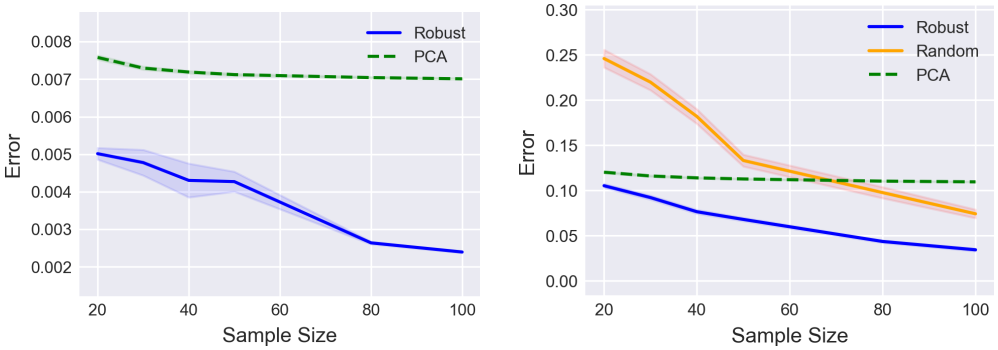
Moreover, when curvature is increased, similar performance patterns hold and all AE-based methods begin to do better than PCA for larger data sizes as expected. Most importantly, PCA-Robust AE consistently outperforms other models in most data regimes and at least as well as any other model otherwise.



(a) $n = 1.1$

(b) $n = 4$

Figure 3. 2D surface satisfying $x^n + y^n = z$ (gray) along with PCA projections (green) and PCA-Robust (blue) for 40 training samples.



(a) $n = 1.1$

(b) $n = 4$

Figure 4. Comparison of dimensionality reduction techniques on a synthetic dataset generated with different curvatures

3.2. **Nanophotonic Component Design**

Here we present several real-world problems that motivated this work in the first place, and illustrate the practical utility of dimensionality reduction with limited data. In nanophotonic component design, the geometry on a chip is manipulated on a nanometer scale to generate functions for applications ranging from optical communication to biological sensing [21]. Designing such devices requires solving Maxwell's partial differential equations to obtain the electromagnetic field distribution—a computationally expensive process. Such design problems are typically posed as optimization problems where parameters represent physical quantities such as material properties and/or geometry. The introduction of an adjoint based optimization method in nanophotonic design [22, 23] allowed an efficient gradient-based optimization that easily scales up to a large number of parameters. Acquiring a single optimized design typically takes hours to days of computation depending on the problem and computational resources. However, in practice a single optimized design is often not the best one to be fabricated. The preferred course of action is to present a collection of optimized designs among which the designer chooses one or a handful to manufacture, based on a variety of considerations, such as other figures of merit or fabrication reliability, that are not captured in the objective function.

In [1, 24, 25], PCA was used to characterize a subspace of optimized designs from a small collection of such designs. Exploring this lower dimensional subspace became computationally feasible by simple sampling. While the linear approach was useful, the data also exhibited some curvature [1]. Using a method that can capture the lower dimensional subspace more accurately, including its curvature, implies that exploring this subspace will identify a collection of better performing designs to consider for manufacturing. Here we consider three design problems - two grating couplers and a power splitter, as described below - where we demonstrate that the proposed autoencoders can capture some of the curvature of the design space and generalize well despite being trained on limited data.

3.2.1. **Grating Coupler 1**

A vertical grating coupler is a device that diffracts light injected in-plane vertically upwards, or couples light signal in the photonic chip to and from optical fibers or free space. It is a necessary device for connecting photonic chips to the surrounding environment. A schematic of the device that is considered here is shown in Figure 5 (Top).

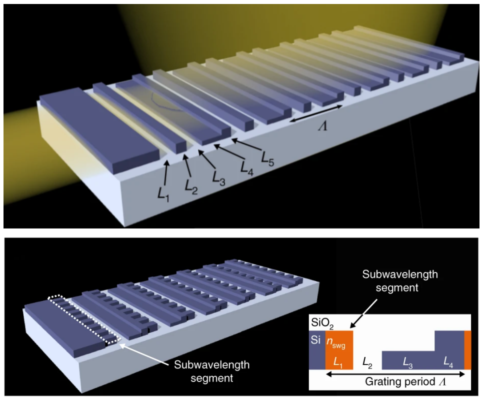
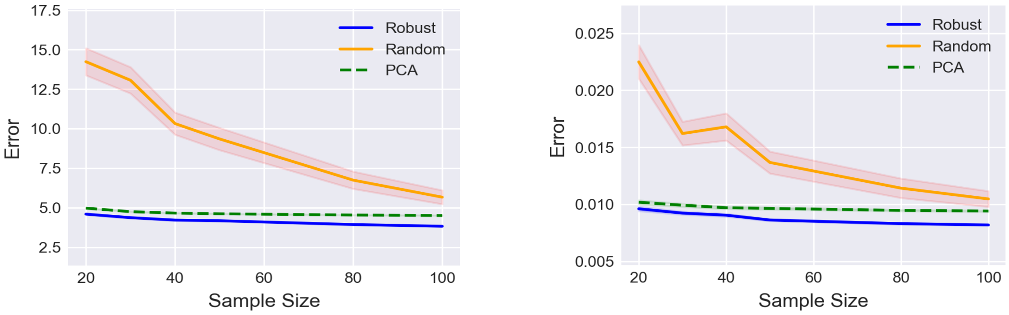


Figure 5. Schematic representations of grating couplers' structures: Top - Grating Coupler 1; Bottom - Grating coupler 2. (reproduced with permission [1]).

The complete dataset of optimized structures for this vertical grating coupler consists of 540 designs, also referred to as a set of good designs. These were obtained from computationally intensive simulation-based optimizations and selected from a set consisting of more than 30,000 candidate designs based on an optical performance criterion (a waveguide-fiber coupling efficiency of at least 74%). The designs are characterized by five segment values (L_1 to L_5) representing different widths of silicon blocks, which are the parameters to the design problem (see Figure 5 (Top)). In [1], it was shown that two principal components were enough to capture most of the optimized design subspace within this five dimensional design problem. Only a small set of such designs were required to identify the subspace. The layer sizes for this experiment were 5-20-5-2-5-20-5 (see Figure 1) for the 5-dimensional data. Note that the size, n, of the bottleneck for the experiments is compared to PCA with n principal components as in [1].

Similar to the setup in [1], we choose 2 as a reduced dimension and simulate scenarios where only a small number of these optimized designs are available, ranging from 20 to 100. These are randomly sampled from the available good designs set. To corroborate our entire experimental setup that can be sensitive to data partitioning due to the low data regime, we set aside half of the available designs designs (270) as the test set. The experiments are further repeated 50 times creating different data splits to measure accurate statistics of the results. The error bars represent the uncertainty of an average estimate.

Figure 6 (Left) shows the performance of different methods evaluated on the test set. We notice that our proposed Robust model consistently outperforms randomly-initialized



(a) Grating Coupler 1

(b) Grating Coupler 2

Figure 6. Comparison of techniques on Grating Coupler 1 (left)and 2 (right) datasets. Robust depicts the average projection error of PCA-Robust AE, Random is Randomly initialized AE, and PCA depicts PCA's performance

autoencoders as well as PCA across all dataset sizes. As expected, the performance of randomly-initialized autoencoders slowly catches up and is expected to match the performance of Robust model when sufficient data is provided.

Note that with only 20 samples the PCA-Robust model has lower error than the standard Autoencoder with 100 samples in Figure 6, while PCA requires around double the amount of samples to catch up to the same error. This offers a significant reduction in the amount of data required to collect to reliably identify the lower dimensional subspace. Similar conclusions can be made in the next device (Grating Coupler 2) we consider.

3.2.2. **Grating Coupler 2**

A similar experiment was conducted on data obtained from another vertical grating coupler design, as depicted in Figure 5 (Bottom). Compared to the previous example, here a section based on a subwavelength metamaterial is introduced in the device structure, as described in [1, 24]. While the number of variables defining the structure is also five - $(L_1, \cdots, L_4, n_{swg})$, the variable n_{swg} represents a different physical quantity - the effective material index of the subwavelength metamaterial section - which has a different interpretation as well as a different order of magnitude [21]. The other four variables, as previously, represent silicon segment widths.

The complete dataset consists of 1502 optimized designs (coupling efficiency $> 74\%$) of which half were used as the test set. Since the variables in this design problem are of different magnitudes, the values were scaled for all dimensionality reduction methods. The structure of the experiment is otherwise identical to the previous grating coupler design setup.

The performance results are shown in Figure 6 (Right). Trends very similar to those of the Grating Coupler 1 experiment are observed.

3.2.3. **Power Splitter**

Another nanoponotonic component ubiquitous in integrated circuits is a power splitter. Its role is to split the incoming wave carrying the signal into two or more ports propagating the same signal but with reduced intensity. We consider a parameterized design of a 1x2 power splitter as in [25] targeting equal splitting ratio on the output ports . The splitter is defined by a single silicon shape, the parameters directly controlling its boundary. Examples of such optimized splitters are shown in Figure 7. Ten parameters define the boundary at equally spaced location along the X axis, while the boundary points in between them are interpolated and smoothed.

The Y-Splitter data is 10-dimensional and we compare different dimensionality reductions, specifically, 4 and 5. This resulted in layer sizes of 10-20-10-4-10-20-10 and 10-20-10-5-10-20-10 respectively.

The complete dataset consists of 645 designs that achieve at least 97% coupling efficiency, defined as the amount of power coupled into the output ports (since the device is symmetric the power is guaranteed to be split equally). The optimized structures are obtained from hundreds of gradient based optimizations with random initial conditions. Figure 8 represents the performance results on this dataset for two different choices of the reduced dimension. Similar to previous experiments, PCA-Robust method outperforms its competitors except for PCA at the smallest dataset regime, where their performance is comparable.

An interesting outcome revealed by these two experiments is the observation that with 100 samples, the average projection error for PCA reduced to 5 dimensions (108 ± 0.7) is marginally better than that of PCA-Robust reduced to 4 (114 ± 1.4). First, it suggests that PCA needs more dimensions to capture approximately the same amount of information as a nonlinear method – a sign of an increasingly curved subspace. Second, exploring a lower-dimensional subspace will often require significantly less samples. It is particularly important in applications such as this one, where sampling is costly. Similarly, randomly-initialized autoencoders require more data to catch up with PCA (approximately 50 more samples when reduced to 4 dimensions and 80 more samples when reduced to five dimensions).

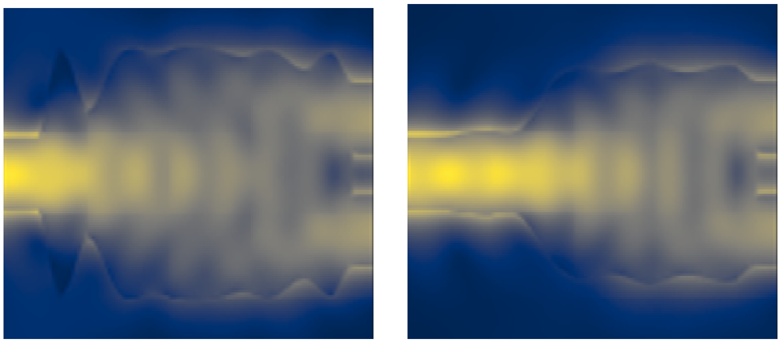
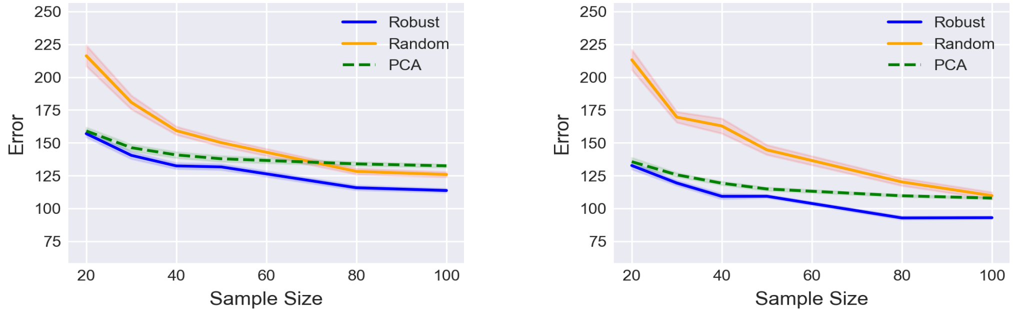


Figure 7. Examples of two optimized 1x2 power splitters. The light travels from the left port (waveguide) and being split equally into two output ports (waveguides) on the right. Both power splitters are defined by their (symmetric) boundaries at the top and bottom representing the silicon shape. Yellow represents the electric field intensity profile.

3.3. **Gene Expression**

Another application that often offers only limited data where PCA is used is gene expression. We consider a subset of gene expression data associated with varying conditions of fungal stress in cotton [26]. The data contains 662 samples. Each sample has six features, each is real-valued and indicates a condition associated with fungal stress in cotton. PCA is applied to this data to enable the visualization of the relationships between samples and thus a reduction that preserves more information, that is, lower average projection error, can provide a more accurate representation of these relationships.

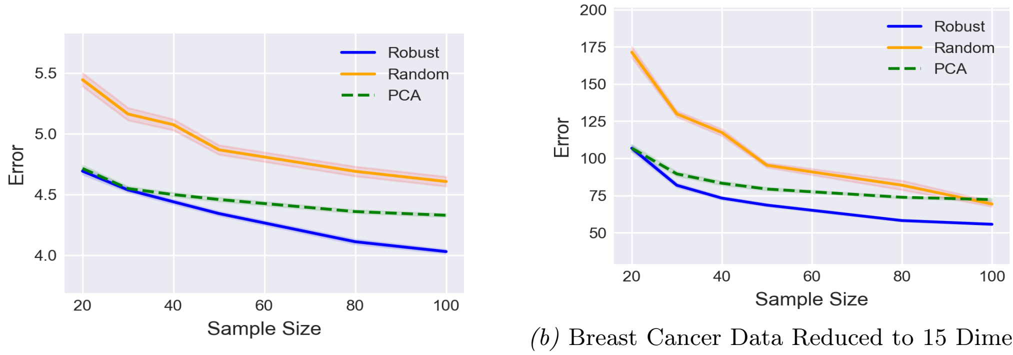
In our experiments we reduce the samples to two dimensions. The autoencoder layers were 6-20-6-2-6-20-6. Figure 9a shows the comparison of the three methods. We can observe that PCA-Robust autoencoder maintains the same performance as PCA for a number of



(a) Y-splitter Data Reduced to 4 Dimensions

(b) Y-splitter Data Reduced to 5 Dimensions

Figure 8. Y-splitter data reduced to different dimensions.



(a) Cotton Data

(b) Breast Cancer Data Reduced to 15 Dimensions

Figure 9. Performance comparison on the Cotton and Breast Cancer datasets.

data sizes, suggesting that the amount of data is not yet sufficient to pick up any nonlinearity that generalizes well. For larger data sizes, there is sufficient data to improve upon the linear method and PCA-Robust approach leverages that information. Also, randomly-initialized autoencoders never catch up to the PCA-Robust model and rarely to PCA in all the above plots, implying a potentially significant increase in data requirement in this practical setting.

3.4. **Breast Cancer Wisconsin Dataset**

The Breast Cancer Wisconsin Dataset [27], included in sci-kit learn package[28], consists of 569 total samples labeled “benign” and “malignant”. These labels are irrevelant for our purposes since we’re interested in comparing the average projection error in the dimensionality reduction of PCA and PCA-Robust. Each sample contains 30 real-valued features describing charecteristics of the cell nuclei depicted in an image of a fine-needle aspirate of a breast mass. Since the feature size is 30 we expanded the width of our vase as compared to our earlier experiments. The new width had 100 nodes and a bottleneck of 15 nodes (to obtain 15 reduced dimensions), resulting in 30-100-30-15-30-100-30 size architecture.

Figure 9b shows that, the PCA-Robust autoencoder outperforms other methods on all except for the smallest data set size, where its performance is comparable to that of PCA.

4. **Conclusion and Future Work**

We demonstrated that autoencoders, with proper initialization, offer a viable solution for dimensionality reduction even in a limited data regime for problems where number of data points is larger than the number of features. Accordingly, a choice does not have to be made between linear and nonlinear model fitting on small datasets. Instead, stable PCA initialization provides the best of both worlds, allowing the AE training to proceed smoothly and introducing nonlinearity only to the extent allowed by the data. These results are encouraging in domains where, due to data scarcity, only PCA has been used for dimensionality reduction. Furthermore, the explicit encoding-decoding nature of autoencoders allow sampling the low-dimensional space for applications such as nanophotonic component design. This is achievable for non-generative autoencoders using reinjection [20]. Exploring the high dimensional data regime ($m < n$), where PCA is no longer the optimal linear method [29] deserves a separate treatment which we defer to future work.

The practical impact of the improvement of the Robust model is illustrated in the experiments, which show a significant reduction in data requirements compared to randomly initialized AEs. In applications where data is difficult to obtain, this reduces the burden of expensive data generation.

While the idea of initializing the autoencoders with PCA is not new, we demonstrate that, unlike its naive implementation, a numerically stable initialization is critical to the final model's training and performance. Our experimental results suggest it is significantly more favourable for the typical gradient-based training method(s) of AE.

Finally, initialization of the network weights in the proposed fashion might offer a degree of stability and robustness similar to the proposals that address adversarial example issues in neural networks and enforce general smoothness [30, 31]. We leave this as an avenue for exploration in future work.

References

[1] D. Melati, Y. Grinberg, M. K. Dezfouli, S. Janz, P. Cheben, J. H. Schmid, A. Sánchez-Postigo, and D.-X. Xu. "Mapping the global design space of nanophotonic components using machine learning pattern recognition". In: *Nature communications* 10.1 (2019), pp. 1–9.

[2] M. F. Festing and D. G. Altman. "Guidelines for the design and statistical analysis of experiments using laboratory animals". In: *ILAR journal* 43.4 (2002), pp. 244–258.

[3] D. Xue, P. V. Balachandran, J. Hogden, J. Theiler, D. Xue, and T. Lookman. "Accelerated search for materials with targeted properties by adaptive design". In: *Nature communications* 7.1 (2016), pp. 1–9.

[4] Y. Zhang and C. Ling. "A strategy to apply machine learning to small datasets in materials science". In: *Npj Computational Materials* 4.1 (2018), pp. 1–8.

[5] Z. Rao et al. *Machine learning-enabled high-entropy alloy discovery*. 2022. DOI: 10.48550/ARXIV.2202.13753. URL: https://arxiv.org/abs/2202.13753.

[6] K. Pearson. "LIII. On lines and planes of closest fit to systems of points in space". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2.11 (1901), pp. 559–572.

[7] M. E. Tipping and C. M. Bishop. "Probabilistic principal component analysis". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 61.3 (1999), pp. 611–622.

[8] M. A. Kramer. "Nonlinear principal component analysis using autoassociative neural networks". In: *AIChE journal* 37.2 (1991), pp. 233–243.

[9] P. Baldi and K. Hornik. "Neural networks and principal component analysis: Learning from examples without local minima". In: *Neural networks* 2.1 (1989), pp. 53–58.

[10] M. Seuret, M. Alberti, M. Liwicki, and R. Ingold. "PCA-Initialized Deep Neural Networks Applied to Document Image Analysis". In: *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*. Vol. 01. 2017, pp. 877–882. DOI: 10.1109/ICDAR.2017.148.

[11] Z. Pan, W. Liu, Z. Meng, and J. Yin. "Linear Variational Autoencoder for Top-N Recommendation". In: *2022 7th International Conference on Big Data Analytics (ICBDA)*. 2022, pp. 296–303. DOI: 10.1109/ICBDA55095.2022.9760352.

[12] O. Allerbo and R. Jörnsten. “Non-linear, Sparse Dimensionality Reduction via Path Lasso Penalized Autoencoders.” In: *J. Mach. Learn. Res.* 22 (2021), pp. 283–1.

[13] G. E. Hinton and S. Roweis. “Stochastic neighbor embedding”. In: *Advances in neural information processing systems* 15 (2002).

[14] B. Schölkopf, A. Smola, and K.-R. Müller. “Kernel principal component analysis”. In: *International conference on artificial neural networks*. Springer. 1997, pp. 583–588.

[15] S. T. Roweis and L. K. Saul. “Nonlinear dimensionality reduction by locally linear embedding”. In: *science* 290.5500 (2000), pp. 2323–2326.

[16] J. B. Tenenbaum, V. d. Silva, and J. C. Langford. “A global geometric framework for nonlinear dimensionality reduction”. In: *science* 290.5500 (2000), pp. 2319–2323.

[17] A. N. Gorban, B. Kégl, D. C. Wunsch, A. Y. Zinovyev, et al. *Principal manifolds for data visualization and dimension reduction*. Vol. 58. Springer, 2008.

[18] K. He, X. Zhang, S. Ren, and J. Sun. “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1026–1034.

[19] A. L. Maas, A. Y. Hannun, A. Y. Ng, et al. “Rectifier nonlinearities improve neural network acoustic models”. In: *Proc. icml*. Vol. 30. 2013, p. 3.

[20] M. Solinas, C. Galiez, R. Cohendet, S. Rousset, M. Reyboz, and M. Mermillod. “Generalization of iterative sampling in autoencoders”. In: *2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE. 2020, pp. 877–882.

[21] P. Cheben, R. Halir, J. H. Schmid, H. A. Atwater, and D. R. Smith. “Subwavelength integrated photonics”. In: *Nature* 560.7720 (2018), pp. 565–572.

[22] C. M. Lalau-Keraly, S. Bhargava, O. D. Miller, and E. Yablonovitch. “Adjoint shape optimization applied to electromagnetic design”. In: *Optics express* 21.18 (2013), pp. 21693–21701.

[23] J. S. Jensen and O. Sigmund. “Topology optimization for nano-photonics”. In: *Laser & Photonics Reviews* 5.2 (2011), pp. 308–321.

[24] M. K. Dezfouli, Y. Grinberg, D. Melati, P. Cheben, J. H. Schmid, A. Sánchez-Postigo, A. Ortega-Moñux, G. Wangüemert-Pérez, R. Cheriton, S. Janz, et al. “Perfectly vertical surface grating couplers using subwavelength engineering for increased feature sizes”. In: *Optics Letters* 45.13 (2020), pp. 3701–3704.

[25] D. Melati, Y. Grinberg, M. K. Dezfouli, J. H. Schmid, P. Cheben, S. Janz, R. Cheriton, A. Sánchez-Postigo, J. Pond, J. Niegemann, et al. “Design of multi-parameter photonic devices using machine learning pattern recognition”. In: *Integrated Photonics Platforms: Fundamental Research, Manufacturing and Applications*. Vol. 11364. International Society for Optics and Photonics. 2020, p. 1136408.

[26] R. Bedre, K. Rajasekaran, V. R. Mangu, L. E. Sanchez Timm, D. Bhatnagar, and N. Baisakh. “Genome-wide transcriptome analysis of cotton (Gossypium hirsutum L.) identifies candidate gene signatures in response to aflatoxin producing fungus Aspergillus flavus”. In: *PLoS One* 10.9 (2015), e0138025.

[27] D. Dua and C. Graff. *UCI Machine Learning Repository*. 2017. URL: http://archive.ics.uci.edu/ml.

[28] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[29] K. Yata and M. Aoshima. “Effective PCA for high-dimension, low-sample-size data with noise reduction via geometric representations”. In: *Journal of multivariate analysis* 105.1 (2012), pp. 193–215.

[30] M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier. “Parseval networks: Improving robustness to adversarial examples”. In: *International Conference on Machine Learning*. PMLR. 2017, pp. 854–863.

[31] C. Anil, J. Lucas, and R. Grosse. “Sorting out Lipschitz function approximation”. In: *International Conference on Machine Learning*. PMLR. 2019, pp. 291–301.