R1: For a nonsymmetric matrix ... We will clarify from the start that our method is designed for symmetric matrices.

R1: PI is unstable if two eigenvalues are close. So what is the advantage of PI vs the analytic solution? The instability of PI is due to inaccurate initialization of the eigenvectors. In our case, the forward pass provides accurate values using SVD and PI then becomes stable during the backward pass. Another advantage over the analytic solution what we demonstrate is that **PI yields an upper bound on the gradients' magnitude, which guarantees they will not explode.** As shown in Sec. 2.2, PI is the geometric expansion of the analytic solution: With $\lambda_i \geq \lambda_j$, the term $1/(\lambda_i - \lambda_j)$, which appears in the analytic solution, is approximated by $1/\lambda_i + 1/\lambda_i \left(\lambda_j/\lambda_i\right) + 1/\lambda_i \left(\lambda_j/\lambda_i\right)^2 + \cdots + 1/\lambda_i \left(\lambda_j/\lambda_i\right)^{K-1} \leq K/\lambda_i$ in PI, where K represents the power iteration number. When $\lambda_i = \lambda_j$, the term $1/(\lambda_i - \lambda_j)$ in the analytic solution explodes, while the PI gradients are naturally upper bounded by K/λ_i.

R1: A hidden regularizer, ϵ, is introduced. What if you set ϵ to 0 or 10^{-12}. ϵ is commonly used to stabilize the computation (*e.g.,* [2] [3] & [r1]). As shown in Eq. 13, ϵ controls the gradients' upper bound and appears in the denominator. Thus, too small an epsilon, *e.g.,* 10^{-12}, will yield a large upper bound and increase the risk of gradient explosion. Following standard practice, e.g., [3], ϵ is set to 10^{-4} for *all* the methods, including SVD and PI, but our method always achieves 100% success rate while the others do not.

R1: Your method uses some tricks. The comparison seems unfair, since you can also truncate SVD. For a fair comparison, we recomputed results using the same tricks to truncate the eigenvalues for our SVD baseline. For PI, truncation was already used in our submission to mitigate the round-off errors caused by the deflation process. The results on CIFAR10 are given in the table below. Note that, for matrix dimensions $d > 16$, SVD and PI still fail in all cases. By contrast, we achieve 100% success rate even when the dimension is as large as 128.

Methods	Evaluation Metrics	$d = 4$	$d = 8$	$d = 16$	$d = 32$	$d = 64$	$d = 128$
SVD	*Min Error & Suc. Rate*	$4.50\%, 60\%$	$4.75\%, 33.3\%$	$4.65\%, 40\%$	$-, 0\%$	$-, 0\%$	$-, 0\%$
PI	*Min Error & Suc. Rate*	$4.44\%, \mathbf{100}\%$	$6.28\%, 6.7\%$	$-, 0\%$	$-, 0\%$	$-, 0\%$	$-, 0\%$
Ours	*Min Error & Suc. Rate*	$4.59\%, \mathbf{100}\%$	$4.43\%, \mathbf{100}\%$	$\mathbf{4.40}\%, \mathbf{100}\%$	$4.46\%, \mathbf{100}\%$	$4.44\%, \mathbf{100}\%$	$4.75\%, \mathbf{100}\%$

R1: Results on CIFAR100 are far away from SOTA performance. It would be nice to compare with [r1]. Our paper focuses on solving the stability issues of ED, not designing better normalization layers (*i.e.*, PCA & ZCA). Stability is measured as the success rate of the methods, which, for our purpose, is more important than accuracy. ZCA and PCA constitute two applications of our method to demonstrate stability. We therefore just used simple backbones (*i.e.*, ResNet18/50), which translates to accuracies inferior to the SOTA. By contrast, [r1] focuses on designing a better normalization layer using an iterative normalization method. We nonetheless acknowledge the relevance of this paper, which we will cite in the final version. Note that [r1] was not published at the time of NeurIPS submission.

R1: I doubt ED is widely used as there are numerical issues. Justify why ED is important for deep learning. Indeed, ED has many numerical stability issues, and this is exactly what our paper addresses. Nevertheless, as discussed in the introduction from Line 14 to 18, ED has been used for image/point matching [6,7,8], second-oder pooling [4], and pose estimation [9]. It has not been well integrated into deep networks because of the numerical instability, and, as stated by R3, one can expect that our paper will bring insight to this problem and be the basis for many new ideas.

R2: Will the failure cases become more or less common when matrices are large enough? As shown in the table above, the baselines' failures become more common as d increases, whereas our method succeeds 100% of the time for all dimensions. This remains true when increasing d to 128 (twice as many as in the submission), by putting the ZCA layer on top of a 128-channel conv. layer. The underlying reasons are that, thanks to our use of SVD in the forward pass, we have more accurate eigen value/vector estimates than the PI baseline, and that, as shown in Eq. 13, the gradients of our method are always bounded regardless of matrix size while those of the SVD baseline may easily explode.

R2: The convergence behavior looks similar whether using the existing methods and the proposed one. The convergence curves shown in Fig. 2 are based solely on the successful cases for the baselines and ignore the failure cases (see success rates in Table 2). Including these numerous failures would render the baseline curves entirely meaningless.

R2: In Tables 3 and 4, the prediction error is not monotone with the matrix size. For ZCA in Table 3, with ResNet18, all values are virtually the same, and with ResNet50, the trend shows that larger d values, which only our method can handle, give better results. For PCA in Table 4 (a,b), when too much information is preserved, some noise is kept and the accuracy drops. Conversely, when too much information is removed, some useful signal is discarded and the accuracy also drops. The right number of dimensions to preserve is dataset dependent and can be determined by cross validation.

R3: Do we have to do blocks of d? Dividing the features into blocks of dimension d is only useful to compare our approach with the baselines, which only succeed for small matrix dimensions. Given our stabilized ED method, the blocks become unnecessary, and the largest dimension d in each experiment corresponds to not using blocks.

R3: PCA denoising doesn't look better than batch normalization (BN). While PCA denoising indeed has marginal improvement over BN, it is not our main focus. Similarly to ZCA, PCA denoising only is another application to demonstrate the stability of our method. The baselines to truly look at in Table 5 are PCA(PI) and PCA(SVD), which often break down in the training phase. We will emphasize this in the final version.

R3: Is there a way of minimizing the dependency on ZCA? We will minimize the emphasis on ZCA whitening and PCA denoising in the abstract and introduction.