APPENDIX
I. RELATIONSHIPS TO NEAREST NEIGHBOR AND NEAREST SUBSPACE

One may notice that the use of *all* the training samples of *all* classes to represent the test sample goes against the conventional classification methods popular in face recognition literature and existing systems. These methods typically suggest using residuals computed from "one sample at a time" or "one class at a time" to classify the test sample. The representative methods include:

1) The *nearest neighbor* (NN) classifier: Assign the test sample $\boldsymbol{y}$ to class i if the smallest distance from $\boldsymbol{y}$ to the nearest training sample of class i
$$r_i(\boldsymbol{y}) = \min_{j=1,\ldots,n_i} \|\boldsymbol{y} - \boldsymbol{v}_{i,j}\|_2 \tag{25}$$
is the smallest among all classes.[25]
2) The *nearest subspace* (NS) classifier (e.g., [32]): Assign the test sample $\boldsymbol{y}$ to class i if the distance from $\boldsymbol{y}$ to the subspace spanned by all samples $A_i = [\boldsymbol{v}_{i,1}, \ldots, \boldsymbol{v}_{i,n_i}]$ of class i:
$$r_i(\boldsymbol{y}) = \min_{\alpha_i \in \mathbb{R}^{n_i}} \|\boldsymbol{y} - A_i\alpha_i\|_2 \tag{26}$$
is the smallest among all classes.

Clearly, NN seeks the best representation in terms of just a single training sample,[26] while NS seeks the best representation in terms of all the training samples of each class. The *nearest feature line* (NFL) algorithm [20] strikes a balance between these two by considering the distance of $\boldsymbol{y}$ to the line spanned by any pair of training samples. As NN and NS represent the two extreme cases, we will compare our method with them and see how enforcing sparsity can strike a better balance than methods like NFL.

A. Relationship to Nearest Neighbor

Let us first assume that a test sample $\boldsymbol{y}$ can be well-represented in terms of one training sample, say $\boldsymbol{v}_i$ (one of the columns of A):
$$\boldsymbol{y} = \boldsymbol{v}_i + \boldsymbol{z}_i \tag{27}$$
where $\|\boldsymbol{z}_i\|_2 \leq \varepsilon$ for some small $\varepsilon > 0$. As discussed in Section II-B.2, the recovered sparse solution $\hat{\boldsymbol{x}}$ to (10) satisfies
$$\|\hat{\boldsymbol{x}} - \boldsymbol{x}_0\|_2 \leq \zeta\varepsilon$$
where $\boldsymbol{x}_0 \in \mathbb{R}^n$ is the vector whose i-th entry is 1 and others are all zero, and ζ is a constant that depends on A. Thus, in this case, the ℓ^1-minimization based classifier will give the same identification for the test sample as NN.

On the other hand, in face recognition, test images may have large variability due to different lighting conditions or facial expressions, and the training sets generally do not densely cover the space of all possible face images (as we see in the experimental section, this is the case with the AR database). In this case, it is unlikely that any single training image will be very close to the test image, and nearest-neighbor classification may perform poorly.

Example 3: Figure 16 left shows the ℓ^2-distances between the downsampled face image from subject 1 in Example 1 and each of the training images. Although the smallest distance is correctly associated with subject 1, the variation of the distances for other subjects is quite large. As we will see in Section IV, this inevitably leads to inferior recognition performance when using NN (only 71.6% in this case, comparing to 92.1% of Algorithm 1).[27]

B. Relationship to Nearest Subspace

Let us now assume that the test sample $\boldsymbol{y}$ can be represented uniquely as a linear combination of the training samples A_i of class i:
$$\boldsymbol{y} = A_i\alpha_i + \boldsymbol{z}_i \tag{28}$$
where $\|\boldsymbol{z}_i\|_2 \leq \varepsilon$ for some small $\varepsilon > 0$. Then again according to equation (11), the recovered sparse solution $\hat{\boldsymbol{x}}$ to (10) satisfies
$$\|\hat{\boldsymbol{x}} - \boldsymbol{x}_0\|_2 \leq \zeta\varepsilon$$
where $\boldsymbol{x}_0 \in \mathbb{R}^n$ is a vector of the form $[0, \ldots, 0, \alpha_i^T, 0, \ldots, 0]^T$. That is,
$$\delta_i(\hat{\boldsymbol{x}}) \approx \boldsymbol{x}_0 \;\text{ and }\; \|\delta_j(\hat{\boldsymbol{x}})\| < \zeta\varepsilon \quad \text{for all } j \neq i. \tag{29}$$
We have
$$\|\boldsymbol{y} - A\delta_i(\hat{\boldsymbol{x}})\|_2 \approx \|\boldsymbol{z}_i\|_2 \leq \varepsilon, \quad \text{and} \tag{30}$$
$$\|\boldsymbol{y} - A\delta_j(\hat{\boldsymbol{x}})\|_2 \approx \|\boldsymbol{y}\|_2 \gg \varepsilon \quad \text{for all } j \neq i. \tag{31}$$
Thus, in this case, the ℓ^1-minimization based classifier will give the same identification for the test sample as NS. Notice that for $j \neq i$, $\delta_j(\hat{\boldsymbol{x}})$ is rather different from α_j computed from $\min_{\alpha_j} \|\boldsymbol{y} - A_j\alpha_j\|_2$. The norm of $\delta_j(\hat{\boldsymbol{x}})$ is bounded by the approximation error (29) when $\boldsymbol{y}$ is represented just within class j, whereas the norm of α_j can be very large as face images of different subjects are highly correlated. Further notice that each of the α_j is *an optimal representation* (in the 2-norm) of $\boldsymbol{y}$ in terms of some (different) subset of the training data, whereas *only one* of the $\{\delta_j(\hat{\boldsymbol{x}})\}_{j=1}^k$ computed via ℓ^1-minimization is optimal in this sense; the rest have very small norm. In this sense, ℓ^1-minimization is *more discriminative* than NS, as is the set of associated residuals $\{\|\boldsymbol{y} - A\delta_j(\hat{\boldsymbol{x}})\|_2\}_{j=1}^k$.

Example 4: Figure 16 right shows the residuals of the downsampled features of the test image in Example 1 w.r.t. the subspaces spanned by the 38 subjects. Although the minimum residual is correctly associated with subject 1, the difference from the residuals of the other 37 subjects is not as dramatic as that obtained from Algorithm 1. Compared to the ratio 1:8.6 between the two smallest residuals in Figure 3, the ratio between the two smallest residuals in Figure 16 right is only 1:3. In other words, the solution from Algorithm 1 is more discriminative than that from NS. As we will see Section IV, for the 12×10 downsampled images, the recognition rate of NS is lower than that of Algorithm 1 (91.1% versus 92.1%).

Be aware that the subspace for each subject is only an approximation to the true distribution of the face images. In reality, due to expression variations, specularity, or alignment error, the actual distribution of face images could be nonlinear or multi-modal. Using only the distance to the entire subspace ignores information about the distribution of the samples within the subspace, which could be more important for classification. Even if the test sample is generated from a simple statistical model: $\boldsymbol{y} = A_i\alpha_i + \boldsymbol{z}_i$ with α_i and $\boldsymbol{z}_i$ independent Gaussians, any sufficient statistic (for the

[25] Another popular distance metric for the residual is the ℓ^1-norm distance $\|\cdot\|_1$. This is not to be confused with the ℓ^1-minimization in this paper.

[26] Alternatively, a similar classifier K-NN considers K nearest neighbors.

[27] Other commonly used distance metrics in NN such as ℓ^1-distance give results similar to Figure 16 left.

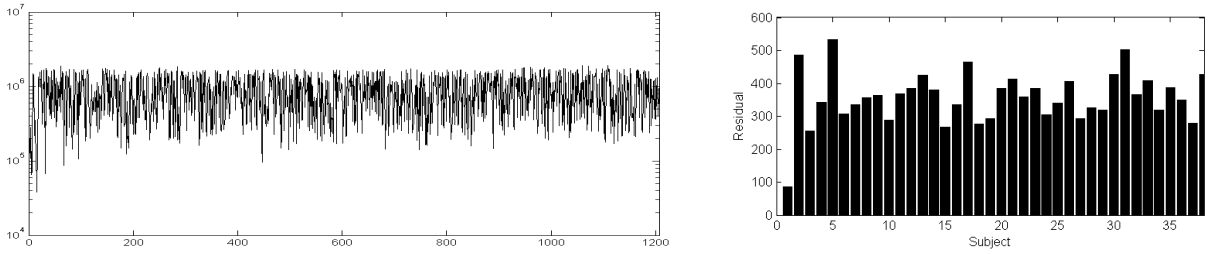

Fig. 16. Left: The ℓ^2-distances (logarithmic scale) between the test image and the training images in Example 1 (as used by nearest neighbor). Right: The residuals of the test image in Example 1 w.r.t. the 38 face subspaces (as used by nearest subspace).

optimal classifier) depends on both $\|\alpha_i\|_2$ and $\|\boldsymbol{z}_i\|_2$, not just the residual $\|\boldsymbol{z}_i\|_2$. While the ℓ^1-minimization based classifier is also suboptimal under this model, it does implicitly use the information in α_i as it penalizes α_i that has a large norm – the ℓ^1-minimization based classifier favors small $\|\boldsymbol{z}_i\|_2$ as well as small $\|\alpha_i\|_1$ in representing the test sample with the training data.

Furthermore, using all the training samples in each class may over-fit the test sample In the case when the solution α_i to

$$\boldsymbol{y} = A_i\alpha_i + \boldsymbol{z}_i \quad \text{subject to} \quad \|\boldsymbol{z}_i\|_2 < \varepsilon$$

is *not unique*, the ℓ^1-minimization (6) will find the sparsest $\alpha_{i0} \in \mathbb{R}^{n_i}$ instead of the least ℓ^2-norm solution $\alpha_{i2} = (A_i^T A_i)^\dagger \boldsymbol{y} \in \mathbb{R}^{n_i}$. That is, the ℓ^1-minimization will use the smallest number of samples necessary in each class to represent the test sample, subject to a small error. To see why the sparse solution α_{i0} respects better the actual distribution of the training samples (inside the subspace spanned by all samples), consider the two situations illustrated in Figure 17.

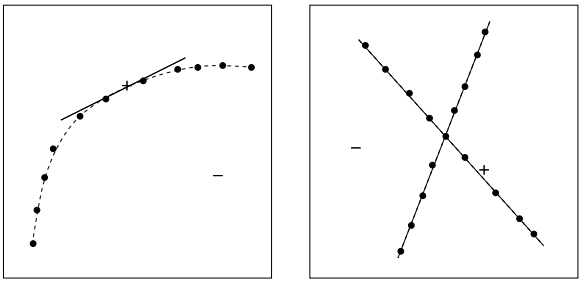

Fig. 17. A sparse solution within the subspace spanned by all training samples of one class. Left: the samples exhibit a nonlinear distribution within the subspace. Right: the samples lie on two lower-dimensional subspaces within the subspace spanned by all the samples.

In the figure on the left, the training samples have a nonlinear distribution within the subspace, say due to pose variation. For the given positive test sample "+," only two training samples are needed to represent it well linearly. For the other negative test sample "-," although it is inside the subspace spanned by all the samples, it deviates significantly from the sample distribution. In the figure on the right, the training samples of one class are distributed on two lower-dimensional subspaces. This could represent the situation in face recognition when the training images contain both varying illuminations and expressions. Again, for a positive test sample "+," typically a small subset of the training samples are needed to represent it well. But if we use the span of all the samples, that could easily over-fit negative samples that do not belong to the same class. For example, as we have shown in Figure 3, although subject 1 has 32 training samples, the test image is well represented using less than 5 large coefficients. In other words, ℓ^1-minimization is very efficient in harnessing sparse structures even within the sample distribution of each class.

From our discussions above, we see that the ℓ^1-minimization based classifier works under a wider range of conditions than NN and NS combined. It strikes a good balance between NN and NS: To avoid under-fitting, it uses multiple (instead of the nearest one) training samples in each class to linearly extrapolate the test sample, but it uses only the smallest necessary number of them to avoid over-fitting. For each test sample, the number of samples needed is automatically determined by the ℓ^1-minimization, because in terms of finding the sparse solution $\boldsymbol{x}_0$, the ℓ^1-minimization is equivalent to the ℓ^0-minimization. As a result, the classifier can better exploit the actual (possibly multi-modal and nonlinear) distributions of the training samples of each class and is therefore likely to be more discriminative among multiple classes. These advantages of Algorithm 1 are corroborated by experimental results presented in Section IV as well as the additional experimental results given below.

C. Experimental Comparison

In this subsection, we provide more detailed numerical results, for easy comparison of Algorithm 1 with NN and NS, in terms of both recognition and validation.

a) Comparison of Recognition Performance: The tables below contain the numerical values plotted in the graphs in Sections IV-A.1 and IV-A.2. Table I gives the performance of our sparse representation based classification (SRC) algorithm on the Extended Yale B database, across different feature transformations and feature dimensions. Here, "E-Random" refers to a variant of the algorithm that uses an *ensemble* of multiple random projections to compute averaged residuals r_i (here, 5 different random projections are used). Aggregating multiple random projections improves the stability of the algorithm, leading to better classification performance. Table II gives the corresponding results for NN and NS. Similarly, using the same experimental setup in Section IV-A.2, Table III gives the result for Algorithm 1, and Table IV for NN and NS.

b) Comparison of Validation Performance: In Section IV-G, we have demonstrated the ability of the robust version of Algorithm 1 to reject invalid test images, in the presence of occlusion. Here, we present further experimental results comparing the algorithm's outlier rejection capability to that of nearest neighbor and nearest subspace, this time without occlusion, working with features rather than the raw image itself. Conventionally, the two major indices used to measure the accuracy of outlier rejection are

TABLE I

RECOGNITION RATES OF SRC ON THE EXTENDED YALE B DATABASE.

Dimension (d)	30	56	120	504
Eigen [%]	86.5	91.63	93.95	96.77
Laplacian [%]	87.49	91.72	93.95	96.52
Random [%]	82.6	91.47	95.53	98.09
Downsample [%]	74.57	86.16	92.13	97.1
Fisher [%]	86.91	N/A	N/A	N/A
E-Random [%]	**90.72**	**94.12**	**96.35**	**98.26**

TABLE II

RECOGNITION RATES OF NEAREST NEIGHBOR (LEFT) AND NEAREST SUBSPACE (RIGHT) ON THE EXTENDED YALE B DATABASE.

Dimension (d)	30	56	120	504
Eigen [%]	74.32	81.36	85.50	88.40
Laplacian [%]	77.13	**83.51**	**87.24**	**90.72**
Random [%]	70.34	75.56	78.79	79.04
Downsample [%]	51.69	62.55	71.58	77.96
Fisher [%]	**87.57**	N/A	N/A	N/A

Dimension (d)	30	56	120	504
Eigen [%]	**89.89**	**91.13**	92.54	93.21
Laplacian [%]	88.98	90.39	91.88	93.37
Random [%]	87.32	91.47	**93.87**	**94.12**
Downsample [%]	80.78	88.15	91.13	93.37
Fisher [%]	81.94	N/A	N/A	N/A

TABLE III

RECOGNITION RATES OF SRC ON THE AR DATABASE.

Dimension (d)	30	54	130	540
Eigen [%]	71.14	80	85.71	91.99
Laplacian [%]	73.71	84.69	90.99	94.28
Random [%]	57.8	75.54	87.55	94.7
Downsample [%]	46.78	67	84.55	93.85
Fisher [%]	**86.98**	**92.27**	N/A	N/A
E-Random [%]	78.54	85.84	**91.23**	**94.99**

TABLE IV

RECOGNITION RATES OF NEAREST NEIGHBOR (LEFT) AND NEAREST SUBSPACE (RIGHT) ON THE AR DATABASE.

Dimension (d)	30	54	130	540
Eigen [%]	68.10	74.82	79.26	80.54
Laplacian [%]	73.10	77.11	**83.83**	**89.70**
Random [%]	56.65	63.66	71.39	74.96
Downsample [%]	51.65	60.94	69.24	73.68
Fisher [%]	**83.40**	**86.84**	N/A	N/A

Dimension (d)	30	54	130	540
Eigen [%]	64.09	77.11	81.97	85.12
Laplacian [%]	65.95	77.54	**84.26**	**90.27**
Random [%]	59.23	68.24	79.97	83.26
Downsample [%]	56.22	67.67	76.97	82.12
Fisher [%]	**80.26**	**85.84**	N/A	N/A

the *false acceptance rate* (FAR) and the *verification rate* (VR). False acceptance rate calculates the percentage of test samples that are accepted and wrongly classified. Verification rate is one minus the percentage of valid test samples that are wrongfully rejected. A good recognition system should achieve high verification rates even at very low false acceptance rates. Therefore, the accuracy and reliability of a recognition system are typically evaluated by the FAR-VR curve (sometimes it is loosely identified as the *receiver operating characteristic* (ROC) curve).

In this experiment, we only use the more challenging AR dataset – more subjects and more variability in the testing data make outlier rejection a more relevant issue. The experiments are run under two different settings. The first setting is the same as in subsection IV-A.2: 700 training images for all 100 subjects and another 700 images for testing. So in this case, there is no real outliers. The role of validation is simply to reject test images that are difficult to classify. In the second setting, we remove the training samples of every third of the subjects and add them into the test set. That leaves us 469 training images for 67 subjects and 700 + 231 = 931 testing images for all 100 subjects. So about half of the test images are true outliers.[28] We compare three algorithms: Algorithm 1, NN, and NS. To be fair, all three algorithms use exactly the same features, 504-dimensional eigenfaces.[29]

Figure 18 shows the FAR-VR curves obtained under the two settings. Notice that Algorithm 1 significantly outperforms NS and NN, as expected. Compared to the performance in Section IV-G, we observe there that the validation performance of Algorithm 1 improves much further with the full image whereas the other methods do not – their performance saturates when the feature dimension is beyond a few hundred.

[28]More precisely, 462 out of the 931 test images belong to subjects not in the training set.

[29]Notice that according to Table III, among all 504-D features, eigenfaces are in fact the worst for our algorithm. We use it anyway as this gives a baseline performance for our algorithm.

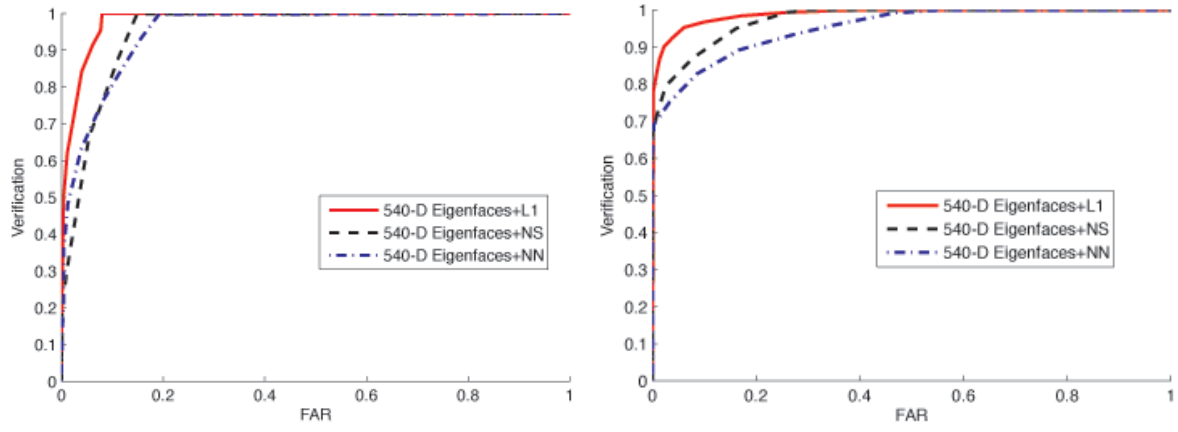

Fig. 18. The FAR-VR curves (solid, red) for SRC using Eigenfaces, compared with the curves of NS and NN using Eigenfaces. Left: 700 images for all 100 subjects in the training, no real outliers in the 700 test images. Right: 469 images for 67 subjects in the training, about half of the 931 test images are true outliers.