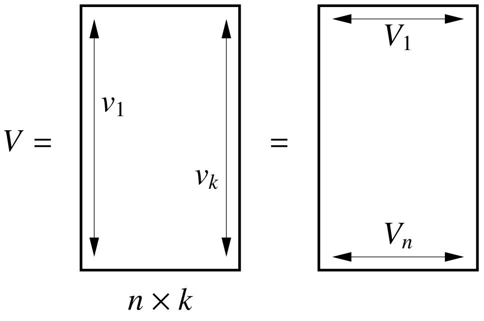
23 Multiple Eigenvectors; Random Projection; Applications

Clustering w/Multiple Eigenvectors

[When we use the Fiedler vector for spectral graph clustering, it tells us how to divide a graph into two graphs. If we want more than two clusters, we can use divisive clustering: we repeatedly cut the subgraphs into smaller subgraphs by computing their Fiedler vectors. However, there are several other methods to subdivide a graph into k clusters in one shot that use multiple eigenvectors rather than just the Fiedler vector v_2. These methods sometimes give better results. They use k eigenvectors in a natural way to cluster a graph into k subgraphs.]

For k clusters, compute first k eigenvectors $v_1 = \mathbf{1}, v_2, \ldots, v_k$ of generalized eigensystem $Lv = \lambda Mv$.
Scale them so that $v_i^\top M v_i = 1$. E.g., $v_1 = \frac{1}{\sqrt{\sum M_{ii}}}\mathbf{1}$. Now $V^\top M V = I$. [The eigenvectors are M-orthogonal.]

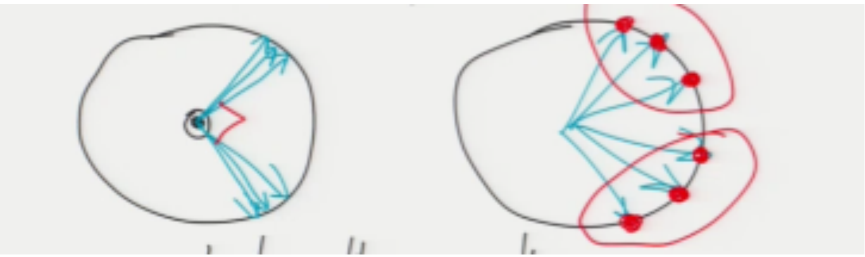


[V's columns are the eigenvectors with the k smallest eigenvalues.]
[Yes, we do include the all-1's vector v_1 as one of the columns of V.]

[Draw this by hand. eigenvectors.pdf]

Row V_i is spectral vector [my name] for vertex i. [The rows are vectors in a k-dimensional space I'll call the "spectral space." When we were using just one eigenvector, it made sense to cluster vertices together if their components were close together. When we use more than one eigenvector, it turns out that it makes sense to cluster vertices together if their spectral vectors point in similar directions.]

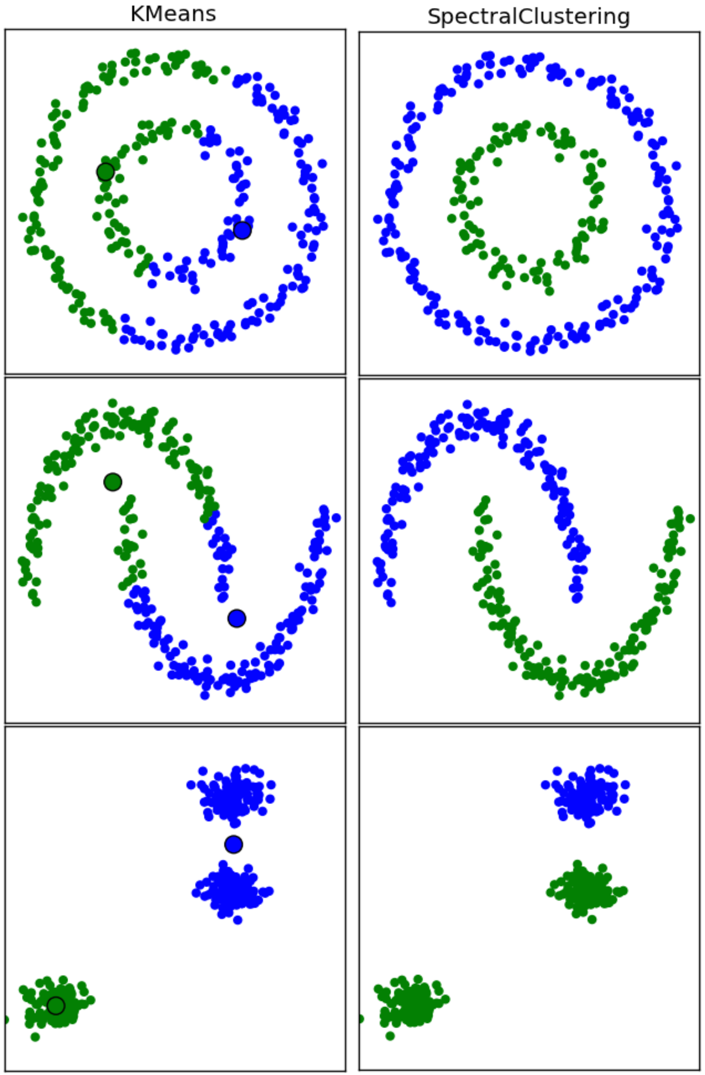
Normalize each row V_i to unit length.
[Now you can think of the spectral vectors as points on a unit sphere centered at the origin.]



[Draw this by hand vectorclusters.png] [A 2D example showing two clusters on a circle. If the graph has k components, the points in each cluster will have identical spectral vectors that are exactly orthogonal to all the other components' spectral vectors (left). If we modify the graph by connecting these components with small-weight edges, we get vectors more like those at right—not exactly orthogonal, but still tending toward distinct clusters.]

k-means cluster these vectors.

[Because all the spectral vectors lie on the sphere, k-means clustering will cluster together vectors that are separated by small angles.]



compkmeans.png, compspectral.png [Comparison of point sets clustered by *k*-means—just *k*-means by itself, that is—vs. a spectral method. To create a graph for the spectral method, we use an exponentially decaying function to assign weights to pairs of points, like we used for image segmentation but without the brightnesses.]

Invented by [our own] Prof. Michael Jordan, Andrew Ng [when he was still a student at Berkeley], Yair Weiss.

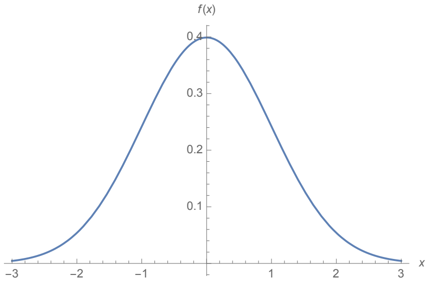
[This wasn't the first algorithm to use multiple eigenvectors for spectral clustering, but it has become one of the most popular.]

THE GEOMETRY OF HIGH-DIMENSIONAL SPACES

[High-dimensional geometry sometimes acts in ways that are completely counterintuitive, defying our intuitions from low-dimensional geometry.]

Consider a random point $p \sim \mathcal{N}(0, I) \in \mathbb{R}^d$.
What is the distribution of its length?

[Looking at the one-dimensional normal distribution, you would expect it to be very common that the length is close to zero, a bit less common that the length is close to 1 or −1, and not rare for the length to be close to 2 or −2. But in high dimensions, that intuition is completely wrong.]



normal.pdf [A one-dimensional normal distribution.]

[If the dimension is very high, the vast majority of the random points are at approximately the same distance from the mean. So they lie in a thin shell. Why? To answer that, let's study the square of the distance. By Pythagoras' Theorem, the squared distance from p to the mean is]

$$\|p\|^2 = p_1^2 + p_2^2 + \ldots + p_d^2$$

[Each component p_i is sampled independently from a univariate normal distribution with mean zero and variance one. The square of a component, p_i^2, is said to come from a chi-squared distribution.]

$$p_i \sim \mathcal{N}(0, 1), \qquad p_i^2 \sim \chi^2(1), \qquad E[p_i^2] = 1, \qquad \mathrm{Var}(p_i^2) = 2$$

[Recall that when you add d independent, identically distributed random numbers, you scale their mean and variance by d, and the standard deviation is the square root of the variance.]

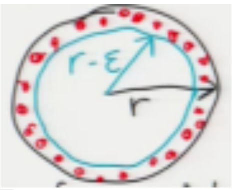
$$\begin{aligned} E[\|p\|^2] &= d\,E[p_1^2] = d \\ \mathrm{Var}(\|p\|^2) &= d\,\mathrm{Var}(p_1^2) = 2d \\ \mathrm{SD}(\|p\|^2) &= \sqrt{2d} \end{aligned}$$

For large d, $\|p\|$ is concentrated in a thin shell around radius $\sqrt{d}$ with a thickness proportional to $\sqrt[4]{2d}$.
[The mean value of $\|p\|$ isn't exactly $\sqrt{d}$, but it is close, because the mean of $\|p\|^2$ is d and the standard deviation is much, much smaller. Likewise, the standard deviation of $\|p\|$ isn't exactly $\sqrt[4]{2d}$, but it's close.]

[So if d is about a million, imagine a million-dimensional egg whose radius is 1,000, and the thickness of the shell is about 67, which is about 10 times the standard deviation. The vast majority of random points are in the eggshell. Not inside the egg; actually in the shell itself. It is very strange that random vectors sampled from a high-dimensional normal distribution almost all have almost the same length.]

[There is a statistical principle hiding here. Suppose you want to estimate the mean of a distribution—in this case, the chi-squared distribution. The standard way to do that is to sample very many numbers from the distribution and take their mean. The more numbers you sample, the more accurate your estimate is—that is, the smaller the standard deviation of your sample mean is. When we sample a vector from a million-dimensional normal distribution and compute its length, that's exactly what we're doing!]

What about a uniform distribution? Consider concentric spheres of radii r & $r - \epsilon$.



[Draw this by hand concentric.png] [Concentric balls. In high dimensions, almost every point chosen uniformly at random in the outer ball lies outside the inner ball.]

Volume of outer ball $\propto r^d$
Volume of inner ball $\propto (r - \epsilon)^d$
Ratio of inner ball volume to outer =

$$\frac{(r-\epsilon)^d}{r^d} = \left(1 - \frac{\epsilon}{r}\right)^d \approx \exp\left(-\frac{\epsilon d}{r}\right) \qquad \text{which is small for large } d.$$

E.g., if $\frac{\epsilon}{r} = 0.1$ & $d = 100$, inner ball has $0.9^{100} = 0.0027\%$ of volume.

Random points from uniform distribution in ball: nearly all are in outer shell.
” ” ” Gaussian ” : nearly all are in some thin shell.

Lessons:
- In high dimensions, sometimes the nearest neighbor and 1,000th-nearest neighbor don't differ much!
- k-means clustering and nearest neighbor classifiers are less effective for large d.

Angles between Random Vectors

What is the angle θ between a random $p \sim \mathcal{N}(0, I) \in \mathbb{R}^d$ and an arbitrary $q \in \mathbb{R}^d$?

Without loss of generality, set $q = [1 \quad 0 \quad 0 \ldots 0]^\top$.
[The value of q doesn't matter, because the direction that p points in is uniformly distributed over all possible directions. By a formula we learned early this semester, the angle between p and q is θ, where …]

$$\begin{aligned} \cos\theta &= \frac{p \cdot q}{\|p\|\,\|q\|} = \frac{p_1}{\|p\|} \\ E[\cos\theta] &= \approx \frac{1}{\sqrt{d}} \end{aligned}$$

If d is large, $\cos\theta$ is almost always very close to zero; θ is almost always very close to 90°!

[In high-dimensional spaces, two random vectors are almost always very close to orthogonal. To put it another way, an arbitrary vector is almost orthogonal to the vast majority of all the other vectors!]

[A former CS 189/289A head TA, Marc Khoury, has a nice short essay entitled “Counterintuitive Properties of High Dimensional Space”, which you can read at
https://marckhoury.github.io/blog/counterintuitive-properties-of-high-dimensional-space]

RANDOM PROJECTION

An alternative to PCA as preprocess for clustering, classification, regression.
Approximately preserves distances between points!

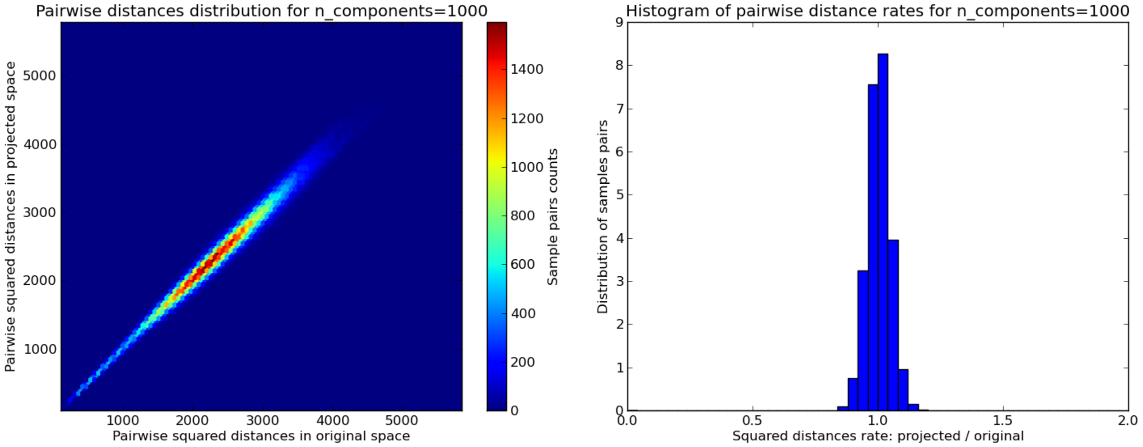
[We project onto a random subspace instead of the PCA subspace, but sometimes preserves distances better than PCA. It works best when you project a very high-dimensional space to a medium-dimensional space. Because it roughly preserves the distances, algorithms like k-means clustering and nearest neighbor classifiers will give similar results to what they would give in high dimensions, but they run much faster.]

Pick a small ϵ, a small δ, and a random subspace $S \subset \mathbb{R}^d$ of dimension k, where $k = \left\lceil \dfrac{2 \ln(1/\delta)}{\epsilon^2/2 - \epsilon^3/3} \right\rceil$.

For any pt q, let $\hat{q}$ be orthogonal projection of q onto S, multiplied by $\sqrt{\frac{d}{k}}$.
[The multiplication by $\sqrt{d/k}$ helps preserve the distances between points after you project.]

Johnson–Lindenstrauss Lemma (modified):
For any two pts $q, w \in \mathbb{R}^d$, $(1 - \epsilon) \|q - w\|^2 \leq \|\hat{q} - \hat{w}\|^2 \leq (1 + \epsilon) \|q - w\|^2$ with probability $\geq 1 - 2\delta$.
Typical values: $\epsilon \in [0.02, 0.5]$, $\delta \in [1/n^3, 0.05]$. [You choose ϵ and δ according to your needs.]

[With these ranges, the squared distance between two points after projecting might change by 2% to 50%. In practice, you can experiment with k to find the best speed-accuracy tradeoff. If you want all inter-sample-point distances to be accurate, you should set δ smaller than $1/n^2$, so you need a subspace of dimension $\Theta(\log n)$. Reducing δ doesn't cost much, but reducing ϵ costs more. You can bring 1,000,000 sample points down to a 10,000-dimensional space with at most a 6% error in the distances.]
[What is remarkable about this result is that the dimension d of the input points doesn't matter!]

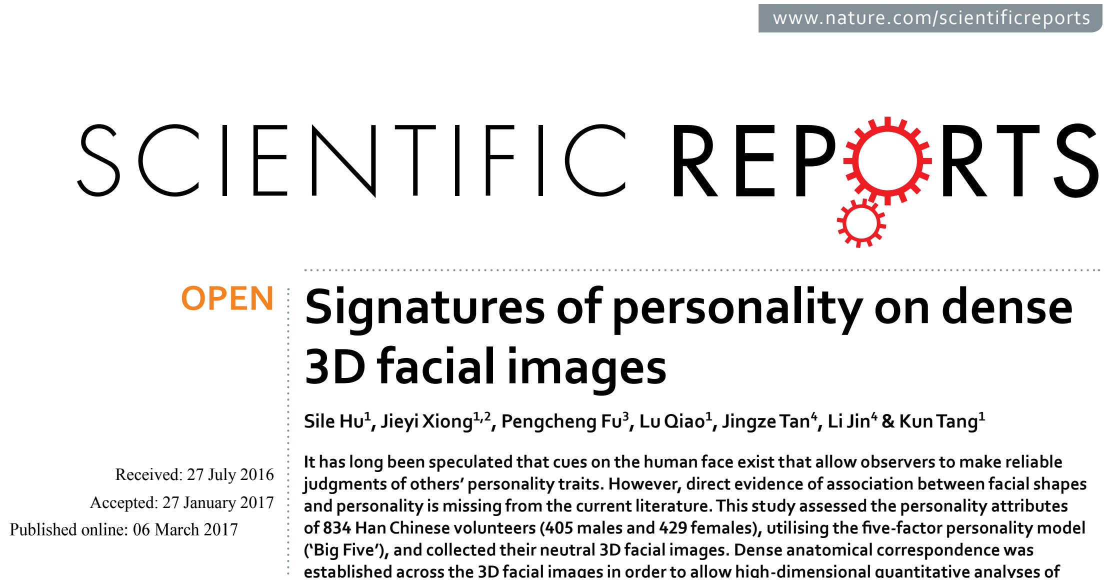


100000to1000.pdf [Comparison of inter-point distances before and after projecting points in 100,000-dimensional space down to 1,000 dimensions.]

[Why does this work? A random projection of $q - w$ is like taking a random vector and selecting k components. The mean of the squares of those k components approximates the mean for the whole population.]

[How do you get a uniformly distributed random projection direction? You can choose each component from a univariate Gaussian distribution, then normalize the vector to unit length. How do you get a random subspace? You can choose k random directions, then use Gram–Schmidt orthogonalization to make them mutually orthonormal. Interestingly, Indyk and Motwani show that if you skip the expensive normalization and Gram–Schmidt steps, random projection still works almost as well, because random vectors in a high-dimensional space are nearly equal in length and nearly orthogonal to each other with high probability.]

PREDICTING PERSONALITY FROM FACES



www.nature.com/scientificreports

SCIENTIFIC REPORTS

OPEN

Signatures of personality on dense 3D facial images

Sile Hu[1], Jieyi Xiong[1,2], Pengcheng Fu[3], Lu Qiao[1], Jingze Tan[4], Li Jin[4] & Kun Tang[1]

Received: 27 July 2016
Accepted: 27 January 2017
Published online: 06 March 2017

It has long been speculated that cues on the human face exist that allow observers to make reliable judgments of others' personality traits. However, direct evidence of association between facial shapes and personality is missing from the current literature. This study assessed the personality attributes of 834 Han Chinese volunteers (405 males and 429 females), utilising the five-factor personality model ('Big Five'), and collected their neutral 3D facial images. Dense anatomical correspondence was established across the 3D facial images in order to allow high-dimensional quantitative analyses of

hu.pdf

Hu et. al (2017).

Big Five (BF) model of personality:

- E: extraversion
- A: agreeableness
- C: conscientiousness
- N: neuroticism
- O: openness

[Researchers have found that these five personality factors are approximately orthogonal to each other. They are highly heritable and highly stable during adulthood.]

Can we predict these traits from 3D faces?

[Studies have shown that people looking at photographs of static faces with neutral expressions can identify the traits better than chance, especially for conscientiousness, extraversion, and agreeableness. This experiment asks whether machine learning can do the same with 3D reconstructions of faces. The subjects were 834 Han Chinese volunteers in Shanghai, China. We don't know whether any of these results might generalize to people who are not Han Chinese.]

[The faces were scanned in high-resolution 3D and a non-rigid face registration system was used to fit a grid of 32,251 vertices to each face in a manner that maps each vertex to an appropriate landmark on the face. (They call this "anatomical homology.") So the design matrix X was 834 × 100,053, representing 834 subjects with 32,251 3D features each.]

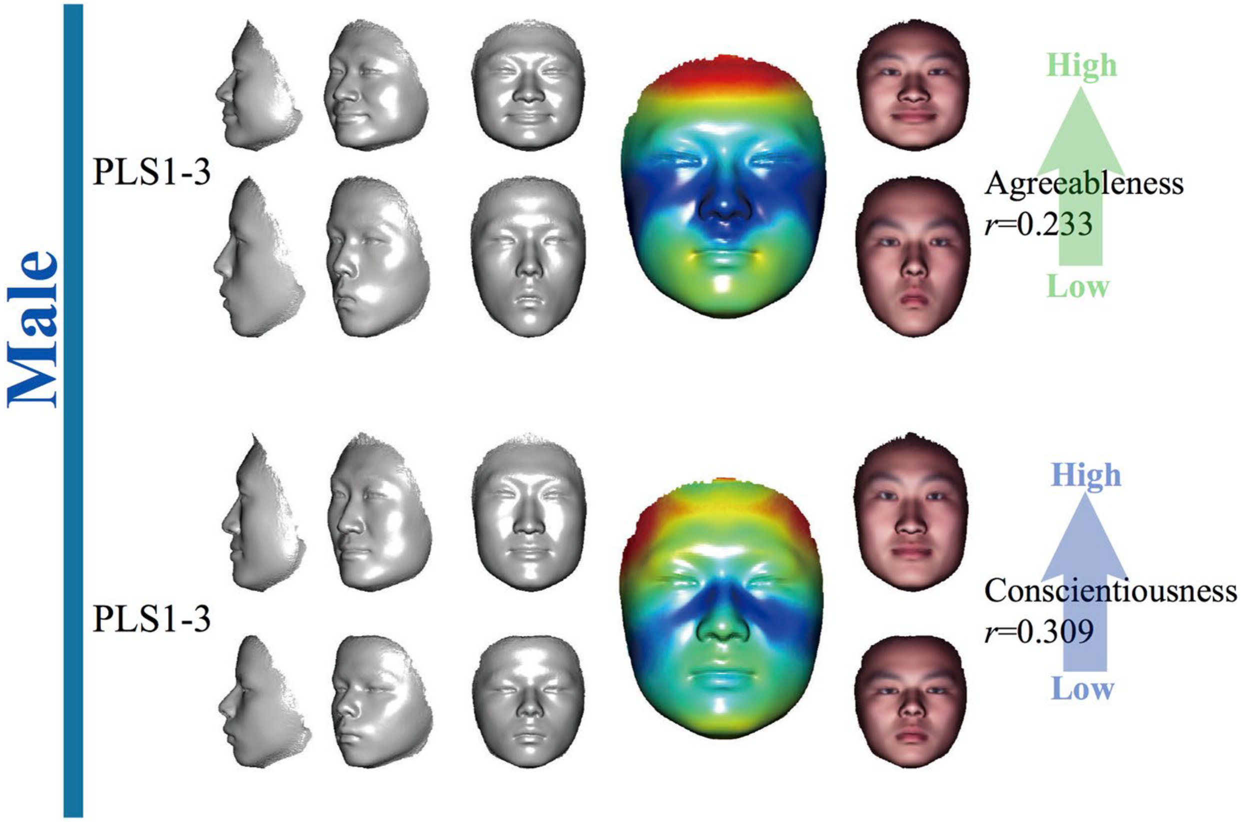
[Subject personalities were evaluated with a self-questionnaire, namely our own Berkeley Personality Lab's Big Five Inventory, translated into Chinese. The authors treated men and women separately.]

Uses partial least squares (PLS) to find associations between personality & faces.

[Everything from here to the end is spoken, not written.]

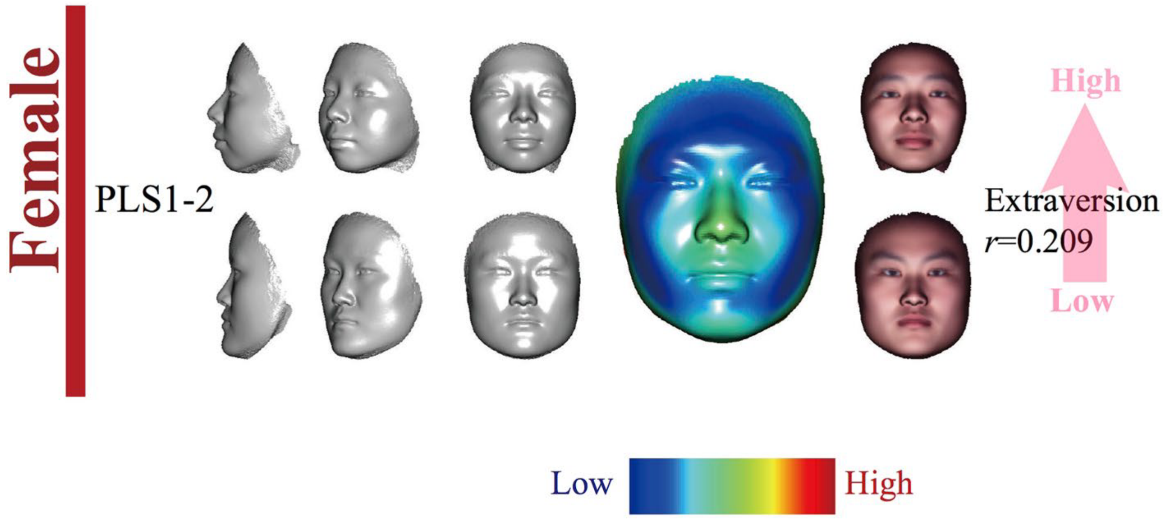
Partial least squares (PLS) is like a supervised version of PCA. It takes in two matrices X and Y with the same number of rows. In our example, X is the face data and Y is the personality data for the 834 subjects. Like PCA, PLS finds a set of vectors in face space that we think of as the most important components. But whereas PCA looks for the directions of maximum variation in X, PLS looks for the directions in X that maximize the correlation with the personality traits in matrix Y.

The researchers found the top 20 or so PLS components and used cross-validation to decide which components have predictive power for each personality trait. They found that the top two components for extraversion in women were predictive, but no components for the other four traits in women were predictive. Men are easier to analyze: they found two or three components were predictive for each of extraversion, agreeableness, conscientiousness, and neuroticism in men. However, the correlations were statistically significant only for agreeableness and conscientiousness.



male.pdf [The relationship between male faces, agreeableness, and conscientiousness. The large, colored faces are the mean faces. Colors indicate the values in the most predictive PLS component vector.]

More agreeable men correlate with much wider mouths that look a bit smiley even when neutral; stronger, forward jaws; wider noses; and shorter faces, especially shorter in the forehead, compared to less agreeable men. More conscientious men tend to have higher, wider eyebrows; wider, opened eyes; a withdrawn upper lip with more mouth tension; and taller faces with more pronounced brow ridges (the bone protuberance above the eyes). The authors note that men with low A and C scores look both more relaxed and more indifferent.



female.pdf [The relationship between female faces and extraversion. The large, colored face is the mean face. Colors illustrate the most predictive PLS component vector.]

More extraverted women correlate with rounder faces, especially in profile, with a more protruding nose and lips but a recessed chin, whereas the introverts have more flat, square-shaped faces. To my eyes, the extraverts also have more expressive mouths.

It's interesting is that physiognomy, the art of judging character from facial shape, used to be considered a pseudoscience, but it's been making a comeback in recent years with the help of machine learning. One reason it fell into disrepute is because, historically, it was sometimes applied across races in fallacious and insulting ways. But if you want to train classifiers that guess people's personalities with some accuracy, you probably need a different classifier for each race. This is a classifier trained exclusively for one race, Han Chinese, which is probably part of why it works as well as it does. If you tried to train one classifier to work on many different races, I suspect its performance would be much worse.

Another thing that's notable is that the authors were able to find statistically significant correlations for some personality traits, the majority of traits defeated them. So while physiognomy has some predictive power, it's only weakly predictive. It's an open question whether machine learning will ever be able to predict personality from visual information substantially better than this or not. Adding a time dimension and incorporating people's movements and dynamic facial expressions seems like a promising way to improve personality predictions.

Tools like this raise some ethical issues. The one that concerns me the most is that, if tools like this are emerging now, many governments probably already had similar tools ten years ago, and have probably been using them to profile us.

One student asked whether these methods might be used by employers to screen prospective employees. I think that tools like this are inferior to simply giving an interviewee a personality test. Such tests are legal in the USA, so long as their questions are not found to violate an employee's right to privacy and the results are not used to discriminate against legally protected groups. The most troubling part of using physiognomy to screen employees would not be that personality testing is unlawful. (It isn't, and quite a few companies do it.) It would be that physiognomy isn't nearly accurate enough. An employer who uses a poorly designed or unvalidated personality test to make personnel decisions might run a higher risk that a court might rule that the test could have a discriminatory effect, violating Title VII of the Civil Rights Act of 1964. Also, they probably won't make good decisions. But perhaps in the future, better measurements, better statistical procedures, and better algorithms might overcome these problems.