Beyond PCA

Limits of PCA

PCA is just one dimensionality reduction technique. Essentially it is doing a linear transform into a space that has certain properties with respect to the data, so unlike the Fourier transform it is customised around the data. It is customising its basis to the data. That is a good thing.

Hence PCA used a lot, but it isn't always the right approach.

1. PCA is "optimal" only in one sense, and that is closely linked with the idea that the data is jointly normal. The idea is based on the assumption that the mean and covariance of the data characterise the distribution, which intrinsically links to the data being normal. That can cause problems, for instance when there are outliers in the data that don't fit normality they can skew the directions unduly.
2. Some data (*e.g.,* heights) is intrinsically non-negative, and hence mean removal results in non-physical negative values.
3. PCA is a linear transform. Not all data lives on a linear space.
4. PCA dimension reduction is aimed a preserving or enhancing a particular property, but that doesn't necessarily solve all embedding problems – sometimes you might aim to preserve distances between points in the new space, for instance.
5. PCA's direction vectors tend to include bits over everything. That isn't ideal because they aren't *explanitory*. It would be more appealing if the basis vectors themselves had some meaning. For instance, Fourier basis vectors have a meaning in terms of frequency, and sparse basis vectors are groupings of variables.
6. PCA starts from a set of measurements in $\mathbb{R}^n$. That precludes non-numerical data (*e.g.,* categorical data), as well as data for which there is no simple mapping to a finite dimensional space (*e.g.,* strings).

Alternatives

There are many other approaches. In essence, all modelling is about finding a "sparse" or simple model for data. Classification is about taking ugly, complex messes of data and converting them into a finite list of classes. But there is a large class of methods that are *called* dimension reduction:

- Independent Components Analysis (ICA), which is aimed at separating components of the signal (the direction vectors shouldn't overlap). It's used to separate out multiple signals that have been mixed together (*e.g.,* speech in a noisy room).
    - https://www.cs.helsinki.fi/u/ahyvarin/papers/NN00new.pdf
- Multidimensional Scaling (MDS), which starts from a distance matrix between the data points, so it can handle almost any type of data as long as you can define a distance metric (it doesn't even need to be a true metric).
    - https://blog.paperspace.com/dimension-reduction-with-multi-dimension-scaling/
- VAE autoencoders: a neural network approach to dimension reduction. They come with a decoder that is useful (most dimension reduction algorithms are effectively one way).
    - https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73
- word2vec: a neural network approach specialised for embedding words based on inferred distances derived from a text.
    - https://pytorch.org/tutorials/beginner/nlp/word_embeddings_tutorial.html
    - https://rguigoures.github.io/word2vec_pytorch/

    - https://notrocketscience.blog/word2vec-with-pytorch-implementing-original-paper/
- A standard wavelet transform uses a fixed basis like the Fourier transform, but an overcomplete wavelet transform proposes a large dictionary of potential basis functions from which we can choose a subset. In general, such approaches might be called *basis pursuit* where we search a large set of potential basis functions for a sparse representation.
    - https://web.stanford.edu/group/SOL/papers/BasisPursuit-SIGEST.pdf
    - https://elad.cs.technion.ac.il/wp-content/uploads/2018/02/Sparse_and_BP.pdf

And there are many others, *e.g.,* see

- https://machinelearningmastery.com/dimensionality-reduction-for-machine-learning/
- https://www.analyticsvidhya.com/blog/2018/08/dimensionality-reduction-techniques-python/

Underlying each of these is often an optimisation problem. There are two main parts that determine what you are doing:

1. The objective of the optimisation, which defines what the method is trying to achieve, and
2. Often there is an additional *regularisation* term that prevents overfitting (and has other positive affects like inducing a variance-bias tradeoff).

We'll be talking much more about that in Lewis's section of the course.