What is decomposition? Give a definition of matrix decomposition and provide 3 examples of its use in AI and Data Science. Then describe its use and importance.

1 Introduction

Matrices and linear algebra are fundamental concepts within AI and data science. Matrices provide a well-structured and succinct way to represent information, systems of linear equations and linear functions (Deisenroth et al., 2020). For example, matrices are often used to represent structured, relational data sets as a series of columns and rows, although aren't limited to this form of data. The rows may represent individuals within a class and the columns represent the grades across several subjects. Techniques from linear algebra can then be applied to the matrices to extract informational content that can be more easily processed by humans to aid decision making. Matrix decompositions are one such technique.

Put simply, decompositions work 'to factorise the matrix into the product of several matrices' (Yang, 2021). Decomposing a matrix is analogous to factorising numbers - it's a transformation into its 'canonical form' (Weisstein, 2023). Many common decompositions provide a low-dimension approximation to high-dimensional data, which is easier to visualise and comprehend (Brunton and Kutz, 2022).

The remainder of this assignment will first introduce some popular methods of matrix decompositions in algebraic form, then three common applications of matrix decompositions in AI and data science will be discussed.

2 Common Decompositions

2.1 Eigendecomposition

Eigendecomposition factorises a square matrix into the product of its eigenvalues and eigenvector expressions (Yang, 2021). Geometrically, we can think of eigenvectors as the direction of the skew of a matrix transformation in each dimension - they show the flow of information. More precisely, eigenvectors are vectors that do not rotate and are only scaled by a matrix.

This relationship is defined by the eigenvalue equation:

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$$

where $\mathbf{A}$ is a square matrix of real values and vector $\mathbf{x}$ is the eigenvector belonging to eigenvalue λ (Abdi, 2007).

By combining the set of eigenvectors to be denoted by $\mathbf{X}$, where the i^{th} column is eigenvector $\mathbf{x}_i$ and storing the eigenvalues in a diagonal matrix Λ, where the i^{th} diagonal element corresponds to the eigenvectors $\mathbf{x_i}$, this gives:

$$\mathbf{A}\mathbf{X} = \Lambda\mathbf{X}$$

and multiplying both sides by the inverse of $\mathbf{X}$ gives

$$\mathbf{A} = \mathbf{X}\Lambda\mathbf{X}^{-1}$$

To reiterate, the eigenvectors $\mathbf{X}$ represent the direction of skew from a matrix transformation. The magnitude of the skew is given by the corresponding eigenvalue within Λ. And the relationship between the eigenvectors is given by the inverse eigenvector, $\mathbf{X^{-1}}$.

In addition to the applications we explore below, another useful feature of this decomposition is when the decomposition is applied many times ($\mathbf{A}$ is multiplied) the $\mathbf{X}$ and $\mathbf{X^{-1}}$ simplify into the identity matrix by definition, significantly simplifying the calculation and saving computational effort.

2.2 Single Value Decomposition

Single Value Decomposition (SVD) is a generalisation of eigendecomposition and is well defined for all matrices. It 'provides a hierarchy of low-rank approximations' to a matrix (Brunton and Kutz, 2022).

SVD is commonly denoted by:

$$M = U\Sigma V^T$$

where M is a $m \times n$ matrix, U is a $m \times m$ unitary matrix of left singular value vectors, Σ is a $m \times n$ diagonal matrix containing the singular values relating to the corresponding singular value vectors and V^T is a transposed $n \times n$ unitary matrix of right singular value vectors (Brunton and Kutz, 2022).

A unitary matrix is square, invertible and has orthonormal columns.[1] The singular, diagonal values within Σ are in descending order and strictly greater than zero. That is

$$0 \geq \sigma_{11} \geq \sigma_{22} \geq ... \geq \sigma_{nn}$$

In geometric terms, both U and V^T are rotational matrices, Σ represents the stretching of space in the directions provided by the V matrix. The smaller the σ_{ii}, the less information provided. Ignoring the dimensions with small singular values provides a low-rank representation.

There are many other popular forms of matrix decomposition not shown here. These include UV decomposition, which is commonly used in recommender systems and we will cover in Section 3.3.

3 Applications and Importance

3.1 Feature extraction in Data Exploration

Often, machine learning works with high-dimensional data which is difficult for humans to identify correlations within and to visualise (Deisenroth et al., 2020). Because of this, algorithms that project high-dimensional data onto lower dimensions while maintaining the 'most statistically descriptive factors' are ubiquitous within data science and AI (Brunton and Kutz, 2022).

Principal component analysis (PCA) is one of the most common forms of dimensionality reduction, first proposed in Pearson (1901). PCA is mostly employed during the data exploration phase of an analysis as it helps the researcher to understand the most important features within a dataset and, equally, helps to identify

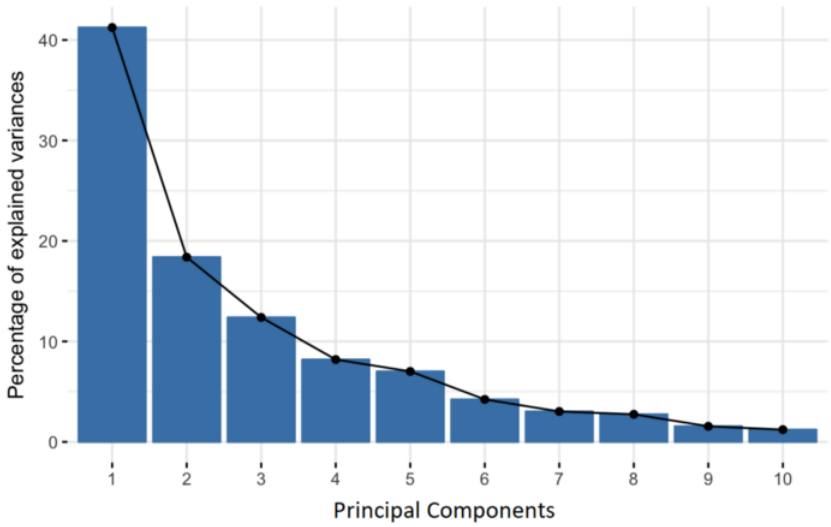
[1] Orthonormal vectors are orthogonal with unit length.

variables that are redundant or highly correlated (collinear). The more correlated are features the greater the reduction potential in PCA, all else equal (Brunton & Kutz, 2022).

PCA applies the eigendecomposition to a covariance matrix with standardised features. To standardise the features, the feature mean is subtracted from each figure and then divided by the feature standard deviation. This ensures all of the features have the same unit. Without standardising, features with smaller units (meaning larger values) and larger variances will over influence the resulting covariance matrix. The covariance matrix has each variables' variance in the diagonal and covariance-pairs in the off diagonal. As covariances are commutative, this produces a symmetrical square matrix of size $p \times p$, where p is the number of features in the data set.

The result of the eigendecomposition identifies a coordinate system based on orthogonal 'principal components', which are a linear combination of the dataset's features that capture the most variance. Generally, the number of principal components is equal to the number of features in the data. Because of the ordering of the diagonal eigenvalue matrix, the principal components are also ranked in terms of their informational value. The variance or informational value of a PC can be calculated by dividing a given eigenvalue by the sum of all eigenvalues in the eigenvalue matrix. The variance contribution can then be plotted in a scree plot such as in Figure 1. This then helps the researcher determine how many PCs they are willing to drop, depending on the amount of informational content of each. Often, the majority of information is stored in the first two PCs. Running machine learning algorithms on a much smaller number of PCs then significantly reduces the computation and time required to calculate accurate predictions.

Figure 1: Example scree plot from PCA



While PCA is a useful tool, one drawback is that it relies 1st and 2nd order dependencies (variances and covariances) and therefore does not capture potential higher-order dependencies (Shlens, 2003).[2] Also, given the PCs are linear combinations of multiple features, it can also make interpretation of the results difficult.

3.2 Image Compression

Image compression is one of the most ubiquitous applications of matrix decomposition in modern life. As society increasingly interacts virtually and the quality of cameras increase, the amount of visual data also increases. Often, the limiting factor in the amount of visual data that can be communicated is broadband bandwidth or memory space. To help overcome these limitations, image compression algorithms use SVD to identify the aspects of a matrix storing image data with most informational content and disregards sections with less informational content. This provides a final image of much smaller size but with little loss of image quality. While this can significantly reduce the size of an image, it does result in information being lost to some

[2] i.e. where $\mathbb{E}[x_i x_j x_k] \neq 0$.

extent and, in some cases, over-compressed images can appear blurred (Yang, 2021). In general, the more compression the worse the quality of the output.

To give a simple example, a greyscale image of dimensions $M \times N$ can be represented by a 2-dimensional matrix $\mathbf{X} \in \mathbb{R}^{M \times N}$ with each pixel a number in the matrix whose values represent the intensity of the pixel. In this example the data lie in an $M \times N$ dimensional space. Decomposing this matrix using SVD, then selecting the first n (where $n < M \times N$) dimensions, remembering that the singular values are ranked, we can reduce the size of the matrix while preserving the most information.

3.3 Recommender Systems

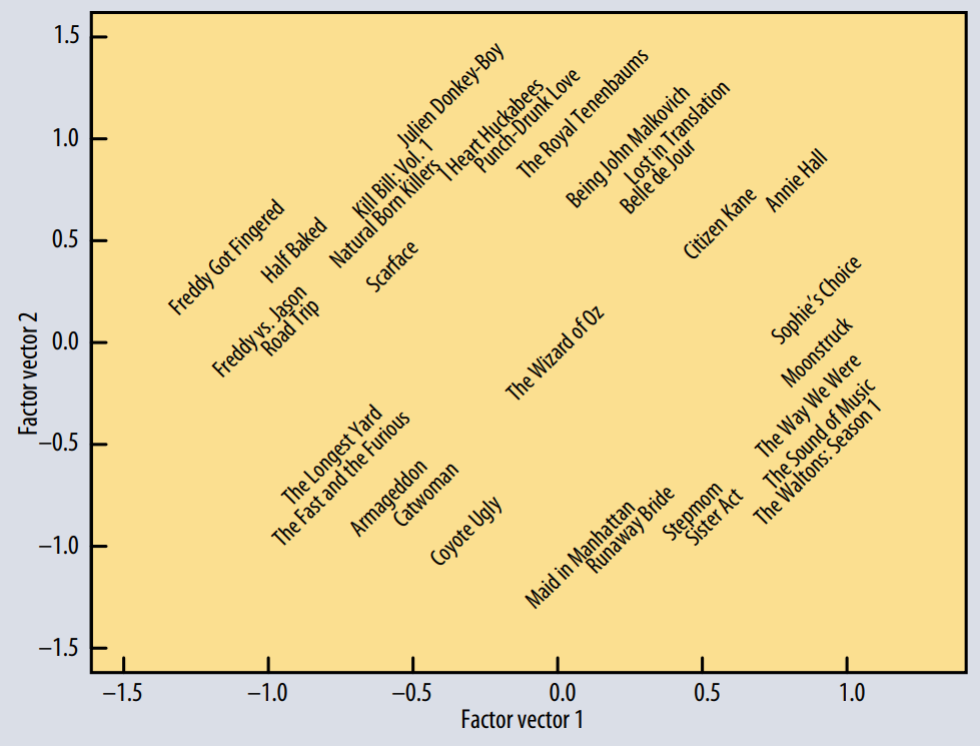
The final application of matrix decompositions we will explore are recommender systems. The objective is to take ratings provided by an individual for an item and draw recommenders for that individual of other items based on the ratings of a larger population of individuals. recommender systems are used throughout modern technology, with common examples seen in platforms such as Amazon and Netflix. The structure of the data is different to classification with user-item-rating tuples rather than feature-label pairs. Intuitively, the matrix has users in the rows and items in the columns with the values being the individual's rating for the item.

One important feature of such data is its sparseness - that is, many of the potential data points will be missing. The average Netflix user will have rated on a small proportion of the total films leading to a large proportion of missing data. In these circumstances, matrix decompositions perform well due to their ability to efficiently strip out these low informational data.

More recent forms of recommender systems are significantly more complex than those just described. They will also use implicit feedback such as time on screen or number of times viewed, along with social and contextual information (Mehta & Rana, 2017).

One further advantage of matrix decompositions is the ability to then plot the data along the first two factor vectors (similar to PCs). Figure 1 shows the position of films on the first two principal components using the Netflix Prize Competition. There are a range of different factors we can see influencing the positions and clusters, from gender of the lead character, to more abstract concepts such as 'quirkiness' (Koren, Bell & Volinsky, 2009).

Figure 2: Netflix Prize Competition recommender system factor plot (Koren, Bell & Volinsky, 2009)



4 Conclusion

This essay has explored matrix decompositions, first in theory and then with three common applications within AI and data science. While several important applications for matrix decompositions have been explored, there are many more not discussed. One overarching implication of matrix decompositions not explored in detail is their ability to significantly reduce computational efficiency and memory storage. However, throughout the essay attention is also drawn to some of the drawbacks of using these techniques. Most frequently, this is a loss of some degree of information, although the aim is to minimise total information loss. Overall, as big data becomes increasingly common across industries, matrix decomposition techniques are also likely to grow in popularity in order to distill complex information quickly and efficiently.

5 References

Abdi, H. (2007) The Eigen-Decomposition: Eigenvalues and Eigenvectors.

Brunton, S. L. and Kutz, J. N. (2022) Data-driven science and engineering: Machine learning, dynamical systems, and control. Cambridge University Press.

Chen (2020) Recommender System: Singular Value Decomposition (SVD) & Truncated SVD. Towards Data Science. https://towardsdatascience.com/recommender-system-singular-value-decomposition-svd-truncated-svd-97096338f361

Deisenroth, M. P., Faisal, A. A., and Ong, C. S. (2020) Mathematics for Machine Learning. Cambridge: Cambridge University Press.

Koren, Y., Bell, R., and Volinsky, C. (2009) Matrix factorization techniques for recommender systems" IEEE Computing, 42(8), 30-37.

Mehta, R., and Rana, K. (2017) A review on matrix factorization techniques in recommender systems. In 2017 2nd International Conference on Communication Systems, Computing and IT Applications (CSCITA), Mumbai, India, pp. 269-274. doi: 10.1109/CSCITA.2017.8066567.

Pearson, K. (1901) On Lines and Planes of Closest Fit to Systems of Points in Space. Philosophical Magazine, 2(11), 559–572.

Shlens, J. (2003) A TUTORIAL ON PRINCIPAL COMPONENT ANALYSIS Derivation, Discussion and Singular Value Decomposition.

Weisstein, E. W. (2023) Matrix Decomposition. From MathWorld-A Wolfram Web Resource.

Yang, B. (2021) Application of Matrix Decomposition in Machine Learning. IEEE International Conference on Computer Science, Electronic Information Engineering and Intelligent Control Technology (CEI), Fuzhou, China, pp. 133-137.