COMP3204/COMP6223 - Computer Vision

3

Covariance and Principal Components

Summary

Understanding the shape of data in a feature space is important to effectively using it. In addition, by understanding the distribution of really highly dimensional data, it is possible to determine the most important modes of variation of that data, and thus represent the data in a space with many fewer dimensions.

Key points

Variance and covariance

- Mathematicians talk about variance and covariance in terms of **random variables** and **expected values**.
  - For our purpose, a random variable can be thought of as the set of values from a single dimension of some or all the data in a feature space.
  - The expected value of such a variable is just its mean value.
- **Variance** ($\sigma^2(x)$) of a set of n data points, $x = [x_1, x_2, \dots, x_n]$, is the average squared difference from the mean (μ):

$$\sigma^2(x) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2$$

  - Variance measures how "spread-out" the data is from the mean
- **Covariance** measures how two variables (x and y) change together:

$$\sigma(x, y) = \frac{1}{n}\sum_{i=1}^{n}(x - \mu_x)(y - \mu_y)$$

  - Variance is the covariance when the two variables are the same! $\sigma(x, x) = \sigma^2(x)$
  - A covariance of 0 means that the variables are **uncorrelated**
    - Covariance is in fact related to correlation:

$$\rho(x, y) = \frac{\sigma(x, y)}{\sqrt{\sigma^2(x)}\sqrt{\sigma^2(y)}}$$

  - Also note that $\sigma(x, y) = \sigma(y, x)$
- The covariance matrix, $\mathbf{\Sigma}$, encodes how all possible pairs of dimensions in a n-dimensional dataset (i.e. points in a feature space), X, vary together:

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma(X_1, X_1) & \sigma(X_1, X_2) & \dots & \sigma(X_1, X_n) \\ \sigma(X_2, X_1) & \sigma(X_2, X_2) & \dots & \sigma(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots \\ \sigma(X_n, X_1) & \sigma(X_n, X_2) & \dots & \sigma(X_n, X_n) \end{bmatrix}$$

  where X_i refers to the i-th element of all the vectors in the feature space.
  - The covariance matrix is a **symmetric matrix**
- **Mean centring** a set of vectors is the process of subtracting the mean (computed from all [or a significant sample] of the vectors) from each vector.
- If you have a set of n mean-centred vectors, you can form them into a matrix, $\mathbf{Z}$, where each **row** corresponds to one of your vectors. The covariance matrix is then directly proportional to the transpose of $\mathbf{Z}$ multiplied by $\mathbf{Z}$:

$$\mathbf{\Sigma} \propto \mathbf{Z}^{\mathrm{T}}\mathbf{Z}$$

Written by Jonathon Hare. Send any comments/bugs/typos to jsh2@ecs.soton.ac.uk.

Principle axes of variation

- A basis is a set of linearly independent vectors that forms a "coordinate system".
  - As the vectors are linearly independent, they are **orthogonal.**
  - For a given dimensionality, there are an infinite number of possible basis.
- In the two-dimensional case, the covariance matrix (or indeed any other 2x2 symmetric matrix) can be seen to define an ellipse with major and minor axes (the actual reason for this is related to a mathematical concept called "Quadratic forms", which is even applicable in higher dimensions – see Enrico's slides from last year if you want the proof).
  - The major axis is along the dimension of which the underlying data is most spread.
  - The minor axis is **perpendicular** to the major axis.
- With more dimensions a similar pattern emerges:
  - The (first) principle axis is along the dimension of which the underlying data is most spread.
  - The second principle axis is in the direction in which the data is most spread orthogonal to the principal axis.
  - The third principle axis is in the direction in which the data is most spread orthogonal to the principal axis and the second principal axis.
  - And so on…
- The set of principal axes is a **basis.**

The Eigendecomposition of the covariance matrix

- An **eigenvector** of a square matrix **A** is a non-zero vector v that, when the matrix is multiplied by v, yields a constant multiple of v commonly denoted as λ:

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

  - λ is called the **eigenvalue** of **A** corresponding to the vector v.
- If **A** is NxN, then there are at most N unique eigenvalue-vector pairs.
- If **A** is symmetric, then the set of all eigenvectors of **A** is a basis and the eigenvectors are **orthogonal**.
- If the matrix **A** is a covariance matrix, then it turns out that **the eigenvectors are the principal components**!
  - The vector with the largest eigenvalue is the principal axis, the vector with the second larges eigenvalue is the second principal axis, and so on.
  - **Eigenvalues turn out to be proportional to the variance along an axis!**
- Formally, the **Eigendecomposition** factorises a diagonalisable square matrix **A** such that:

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

  where **Q** is the square (N×N) matrix whose i^{th} column is the eigenvector q_i of **A** and $\mathbf{\Lambda}$ is the diagonal matrix whose diagonal elements are the corresponding eigenvalues (*i.e.*, $\mathbf{\Lambda}_{ii} = \lambda_i$).
  - **The Eigendecomposition is thus a way of finding the principal axes**
  - If **A** is a real symmetric matrix (such as a covariance matrix) then **Q** is an orthogonal matrix and $\mathbf{Q}^{-1} = \mathbf{Q}^{\mathrm{T}}$
- The Eigendecomposition can be solved analytically for very small matrices (i.e. $N \leq 4$). For larger matrices it is solved using iterative numerical methods.
  - All numerical algebra software/libraries will have an Eigendecomposition function; many will allow you to efficiently find the largest-k eigenvalue-vector pairs rather than computing them all (which can be very expensive if N is large).
  - It is common practise to re-arrange the columns of **Q** and corresponding eigenvalues in $\mathbf{\Lambda}$, such that the eigenvalues decrease (i.e. $\lambda_i > \lambda_{i+1}$).

Dimensionality reduction with Principle Component Analysis

- A linear transform (**W**) maps vectors z_i (rows of **Z**) from one space to another:

$$\mathbf{T} = \mathbf{Z}\mathbf{W}$$

  where **T** is the transformed space (vectors t_i from the rows of **T** correspond to the original vectors z_i in the transformed space).
  - **T** can have fewer dimensions than **Z**.

- PCA is mathematically defined as an **orthogonal linear transformation** (meaning it rotates and scales) that transforms the data to a new coordinate system such that the greatest variance by any projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on.
    - PCA thus projects data in an original space to a new space defined by the basis of principal axes. The transform matrix is just the eigenvector matrix **Q:**
    $$\mathbf{W} = \mathbf{Q}$$
    - Because the new (principle) axes are sorted by variance, we can choose to ignore any axes with small variance, thus providing a way of **reducing the dimensionality** of the data.
        - Keeping only the first L principal components (i.e. columns of $\mathbf{Q}$, assuming the eigenvectors are sorted by decreasing eigenvalue) gives a truncated transformation:
        $$\mathbf{T}_L = \mathbf{Z}\mathbf{Q}_L$$
        where the matrix $\mathbf{T}_L$ now has n rows but only L columns.
    - Given a low-dimensional vector formed from PCA, it is possible to reconstruct the original vector: $t_L = \mathrm{z}\mathbf{Q}_L \Rightarrow z = t_L\mathbf{Q}_L^{-1} = t_L\mathbf{Q}_L^{\mathrm{T}}$
        - Then add the mean vector to get back into the original space before mean centring.
    - Summary of the steps for PCA:
        1. Mean-centre the data vectors
        2. Form the vectors into a matrix $\mathbf{Z}$, such that each row corresponds to a vector
        3. Perform the Eigendecomposition of the matrix $\mathbf{Z}^{\mathrm{T}}\mathbf{Z}$, to recover the eigenvector matrix $\mathbf{Q}$ and diagonal eigenvalue matrix $\mathbf{\Lambda}$:
        $$\mathbf{Z}^{\mathbf{T}}\mathbf{Z} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{-\mathbf{1}}$$
        4. Sort the columns of $\mathbf{Q}$ and corresponding diagonal values of $\mathbf{\Lambda}$ so that the eigenvalues are decreasing.
        5. Select the L largest eigenvectors of $\mathbf{Q}$ (the first L columns) to create the transform matrix $\mathbf{Q}_L$.
        6. Project the original vectors into a lower dimensional space, $\mathbf{T}_\mathrm{L}$:
        $$\mathbf{T}_L = \mathbf{Z}\mathbf{Q}_L$$

Eigenfaces

- Eigenfaces was an early approach to face recognition. It worked by applying PCA to features created by flattening the raw grey-level pixel values of an image into a vector, allowing images to be represented in far fewer dimensions (typical images used are 100x200 pixels, corresponding to 20000 dimensions; Eigenfaces are typically ~100 -200 dimensions, and can work with even fewer).
    - All the images need to be the same size, and aligned (i.e. the eyes need to be in the same place in each image)
    - Eigenfaces can be seen as a generative model: given a low-dimensional vector, it is possible to "generate" an estimate of what the higher dimensional image vector should look like (see the bullet on reconstruction above).
    - The original paper on Eigenfaces, used the low-dimensional vectors with a k-nearest-neighbour classifier to perform recognition.
- Overall approach:
    - From the training images:
        - The images are flattened into vectors
        - The mean vector is computed and stored
        - The vectors are mean centred.
        - PCA is applied to the vectors to project them into a lower dimensional space. The transform matrix (eigenvector matrix) is stored.
        - The low dimensional vectors are used as the training data for a classifier
            - e.g. KNN with a distance threshold
    - For each face image that is to be recognized:
        - The image is flattened into a vector, and the mean vector is subtracted

- The vector is projected by the PCA basis (transform matrix) into the lower dimensional space.
- The lower dimensional vector is given to the classifier, which generates a class label.

Further reading

- Mark's book covers PCA in the appendices
- Wikipedia has good coverage of all the key ideas:
  - http://en.wikipedia.org/wiki/Variance
  - http://en.wikipedia.org/wiki/Covariance
  - http://en.wikipedia.org/wiki/Covariance_matrix
  - http://en.wikipedia.org/wiki/Eigenvalue,_eigenvector_and_eigenspace
  - http://en.wikipedia.org/wiki/Eigendecomposition_of_a_matrix
  - http://en.wikipedia.org/wiki/Eigenface

Practical exercises

- OpenIMAJ tutorial chapter 13 covers Eigenfaces and PCA.

Written by Jonathon Hare. Send any comments/bugs/typos to jsh2@ecs.soton.ac.uk.