Principal Component Analysis

Laurenz Wiskott
Institute for Theoretical Biology
Humboldt-University Berlin
Invalidenstraße 43
D-10115 Berlin, Germany
11 March 2004

1 Intuition

Problem Statement  Experimental data to be analyzed is often represented as a number
of vectors of fixed dimensionality. A single vector could for example be a set of temperature
measurements across Germany. Taking such a vector of measurements at different times
results in a number of vectors that altogether constitute the data. Each vector can also be
interpreted as a point in a high dimensional space. Then the data are simply a cloud of
points in this space.

When analyzing such data one often encounters the problem that the dimensionality of
the data points is too high to be visualized or analyzed with some particular technique. Thus
the problem arises to reduce the dimensionality of the data in some optimal way.

To keep things simple we insist that the dimensionality reduction is done linearly, i.e. we
are looking for a low-dimensional linear subspace of the data space, onto which the data can
be projected. As a criterion for what the optimal subspace might be it seems reasonable to
require that it should be possible to reconstruct the original data points from the reduced
ones as well as possible. Thus if one were to project the data back from the low-dimensional
space into the original high-dimensional space, the reconstructed data points should lie as
close as possible to the original ones, with the mean squared distance between original and
reconstructed data points being the reconstruction error. The question is, how can we find
the linear subspace that minimizes this reconstruction error.

Projection and Reconstruction  The task of principal component analysis (PCA) is
to reduce the dimensionality of some high-dimensional data points by linearly projecting
them onto a lower-dimensional space in such a way that the reconstruction error made
by this projection is minimal. In order to develop an intuition for PCA we first take a
closer look at what it means to project the data points and to reconstruct them. Figure 1
illustrates the process. (a) A few data points are given in a two-dimensional space and are
represented by pairs of numbers \((x_1, x_2)\). (b) In order to reduce the dimensionality down
to one, we have to choose a one-dimensional subspace and project the data points onto

© 2004 Laurenz Wiskott
it. (c) The points can now be represented by just one number, $y$, and we do not care that they originally came from a two-dimensional space. (d) If we want to reconstruct the original two-dimensional positions of the data points as well as possible, we have to embed the one-dimensional space in the original two-dimensional space in exactly the orientation used during the projection. However, we cannot recover the accurate 2D-position; the points remain on the one-dimensional subspace. The reconstruction error is therefore the average distance of the original 2D-positions from the one-dimensional subspace (the length of the projection arrows in (b)). For mathematical convenience one actually takes the average squared distance.

**Reconstruction Error and Variance** The question now is how we can find the direction of the one-dimensional subspace that minimizes the reconstruction error. For that it is interesting to inspect more closely what happens as we rotate the subspace. Figure 2 illustrates the projection onto two different subspaces. Focus just on the one point $\mathbf{x}$ and its projection $\mathbf{x}_\parallel$, $d$ is the distance of $\mathbf{x}$ from the origin, $r$ is the distance of $\mathbf{x}$ from $\mathbf{x}_\parallel$ in the subspace, and $v$ is the distance of $\mathbf{x}_\parallel$ from the origin. $r$ and $v$ depend on the direction of the subspace while $d$ does not. Interestingly, since the triangles between $\mathbf{x}$, $\mathbf{x}_\parallel$, and the origin are right-angled, $r$ and $v$ are related by Pythagoras’ theorem, i.e. $r^2 + v^2 = d^2$. We know that $r^2$ contributes to the reconstruction error, $v^2$ on the other hand contributes to the variance of the projected data within the subspace. Thus we see that the sum over the reconstruction error plus the variance of the projected data is constant and equals the variance of the original data. Therefore, minimizing the reconstruction error is equivalent to maximizing the variance of the projected data.

![Figure 1](image1.png)  
**Figure 1:** Projection of 2D data points onto a 1D subspace and their reconstruction.

![Figure 2](image2.png)  
**Figure 2** Variance of the projected data and reconstruction error as the linear subspace is rotated.
Covariance Matrix  How can we determine the direction of maximal variance? The first we can do is to determine the variances of the individual components. If the data points (or vectors) are written as \( \mathbf{x} = (x_1, x_2)^T \) (\( T \) indicates transpose), then the variances of the first and second component can be written as \( C_{11} := \langle x_1 x_1 \rangle \) and \( C_{22} := \langle x_2 x_2 \rangle \) (angle brackets indicate averaging over all data points). If \( C_{11} \) is large compared to \( C_{22} \), then the direction of maximal variance is close to \( (1, 0)^T \), while if \( C_{11} \) is small, the direction of maximal variance is close to \( (0, 1)^T \). (Notice that variance doesn’t have a polarity, so that one could use the inverse vector \( (-1, 0)^T \) instead of \( (1, 0)^T \) equally well for indicating the direction of maximal variance.)

But what if \( C_{11} \) is of similar value as \( C_{22} \), like in the example of Figure 1? Then the co-variance between the two components, \( C_{12} := \langle x_1 x_2 \rangle \), can give us additional information (notice that \( C_{21} := \langle x_2 x_1 \rangle \) is equal to \( C_{12} \)). A large positive value of \( C_{12} \) indicates a strong correlation between \( x_1 \) and \( x_2 \) and that the data cloud is extended along the \( (1, 1)^T \) direction. A negative value would indicate anti-correlation and an extension along the \( (-1, 1)^T \) direction. A small value of \( C_{12} \) would indicate no correlation and thus little structure of the data, i.e. no prominent direction of maximal variance. The variances and covariances are conveniently arranged in a matrix with components \( C_{ij} \), which is called covariance matrix (assuming zero mean data). Figure 3 shows several data clouds and the corresponding covariance matrices.

![Figure 3: Several data distributions and their covariance matrices.](image)

Covariance Matrix and Higher Order Structure  Notice that the covariance matrix only gives you information about the general extent of the data (the second order moments). It does not give you any information about the structure of the data cloud. Figure 4 shows different data distributions that all have the same covariance matrix. Thus as long as we consider only the covariance matrix, i.e. second order moments, we can always assume a Gaussian data distribution with an ellipsoid shape, because the covariance matrix does not represent any more structure in any case.

PCA by Diagonalizing the Covariance Matrix  Now that we have learned that the covariance matrix in principle contains the information about the direction of maximal variance the question arises how we can get at this information. From Figure 3 (a) and (b) we can see that there are two fundamentally different situations: in (a) the data cloud is aligned with the axes of the coordinate system and the covariance matrix is diagonal; in (b) the data cloud is oblique to the axes and the matrix is not diagonal. In the former case the direction of maximal variance is simply the axis belonging to the largest value on the diagonal of the covariance matrix. In the latter case, we cannot directly say what the direction
of maximal variance might be. Thus, since the case of a diagonal covariance matrix is so much simpler, the strategy we are going to take is to make a non-diagonal covariance matrix diagonal by rotating the coordinate system accordingly. This is illustrated in Figure 5. From linear algebra we know that diagonalizing a matrix can be done by solving the corresponding eigenvalue equation. It will turn out that the eigenvectors of the covariance matrix point into the directions of maximal (and minimal) variance and that the eigenvalues are equal to the variances along these directions. Projecting the data onto the eigenvectors with largest eigenvalues is therefore the optimal linear dimensionality reduction.

2 Formalism

Definition of the PCA-Optimization Problem  The problem of principal component analysis (PCA) can be formally stated as follows.

**Principal Component Analysis (PCA):** Given a set \( \{ \mathbf{x}^\mu : \mu = 1, ..., M \} \) of \( I \)-dimensional data points \( \mathbf{x}^\mu = (x_{1\mu}, x_{2\mu}, ..., x_{I\mu})^T \) with zero mean, \( \langle \mathbf{x}^\mu \rangle = 0 \), find an orthogonal matrix \( \mathbf{U} \) with determinant \( |\mathbf{U}| = +1 \) generating the transformed data points \( \mathbf{x}^\mu := \mathbf{U}^T \mathbf{x}^\mu \) such that for any given dimensionality \( P \) the data projected onto the first \( P \) axes, \( \mathbf{x}_\parallel^\mu := (x_{1\mu}, x_{2\mu}, ..., x_{P\mu}, 0, ..., 0)^T \), has the smallest reconstruction error \( E := \langle \|\mathbf{x}^\mu - \mathbf{x}_\parallel^\mu\|^2 \rangle_\mu \) among all possible projections onto a \( P \)-dimensional subspace. The row vectors of matrix \( \mathbf{U} \) define the new axes and are called the *principal components*.

Some remarks: (i) If one has non-zero-mean data, one typically removes the mean before applying PCA. Even though all the math is valid also for non-zero-mean data, the results

![Figure 4: Different data distributions with identical covariance matrices.](image)

![Figure 5: Diagonalizing the covariance matrix by rotating the coordinate system.](image)
would typically be undesired and nonintuitive.  (ii) Since matrix $U$ is orthogonal and has
determinant value $+1$, it corresponds simply to a rotation of the data $x$. Thus, the ‘shape’
of the data cloud remains the same, just the ‘perspective’ changes. Note also that one can
interpret the multiplication with matrix $U^T$ either as a rotation of the data or as a rotation
of the coordinate system. Either interpretation is valid. (iii) Projecting the data $x'$ onto
the $P$-dimensional linear subspace spanned by the first $P$ axes is simply done by setting all
components higher than $P$ to zero. This can be done, because we still have an orthonormal
coordinate system. If $U$ and therefore the new coordinate system were not orthogonal then
the projection became a mathematically more complex operation. (iv) The reconstruction
error has to be minimal for any $P$. This has the advantage that we do not need to decide
on $P$ before performing PCA. Often $P$ is actually choosen based on information obtained
during PCA and governed by a constraint, such as that the reconstruction error should be
below a certain threshold. (v) From now on we will drop the index $\mu$ and keep in mind that
averages indicated by $\langle \cdot \rangle$ are always taken over all data points indexed with $\mu$.

**Mapping from High- to Low-dimensional Space and New Coordinate System**

Assume some data points $x$ are given in an $I$-dimensional space and a linear subspace is
spanned by $P$ orthonormal vectors

$$v_p := (v_{1p}, v_{2p}, ..., v_{Ip})^T$$

with $v_p^Tv_q = \delta_{pq} := \begin{cases} 1 & \text{if } p = q \\ 0 & \text{otherwise} \end{cases}$

We will typically assume $P < I$ and speak of a high($I$)-dimensional space and a low($P$)-
dimensional (sub)space. However, $P = I$ may be possible as a limiting case as well.

Arranging these vectors in a matrix yields

$$V := (v_1, v_2, ..., v_P)$$

$$= \begin{pmatrix} v_{11} & v_{12} & \cdots & v_{1P} \\ v_{21} & v_{22} & \cdots & v_{2P} \\ \vdots & \vdots & \ddots & \vdots \\ v_{I1} & v_{I2} & \cdots & v_{IP} \end{pmatrix}$$

This matrix can be used to map the data points $x$ into the subspace spanned by the vectors
$v_p$ yielding

$$y := V^Tx.$$  

If $P < I$ then the dimensionality is reduced and some information is lost; if $P = I$ all
information is preserved. In any case the mapped data are now represented in a new coordinate
system the axes of which are given by the vectors $v_p$. With $P = 2$ and $I = 3$, for example,
we have

$$y = \begin{pmatrix} v_1^Tx \\ v_2^Tx \end{pmatrix} = \begin{pmatrix} v_{11} & v_{21} \\ v_{12} & v_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = V^Tx.$$ 

or

$$y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} v_{11} & v_{21} & v_{31} \\ v_{12} & v_{22} & v_{32} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = V^Tx.$$
Note that $y$ is $P$-dimensional while $x$ is $I$-dimensional.

It is important to realize that we have done two things here: firstly, we have moved
the points from the high-dimensional space onto the low-dimensional subspace (the points
that were already in the subspace have not been moved, of course) and secondly, we have
represented the moved points in a new coordinate system that is particularly suitable for
the low-dimensional subspace. Thus, we went from the high-dimensional space and the old
coordinate system to the low-dimensional subspace and a new coordinate system. Note
also that points in the high-dimensional space can generally not be represented in the new
coordinate system.

**Transforming from New to Old Coordinate System** Interestingly, since the vectors
$v_p$ are orthonormal, matrix $V$ can also be used to transform the points back from the
new to the old coordinate system, although, the lost dimensions cannot be recovered, of
course. Thus the mapped points $y$ in the new coordinate system become points $x_\parallel$ in the
old coordinate system and are given by

$$x_\parallel := Vy$$

$$\Longleftrightarrow VV^T x.$$ \tag{6}

$y$ and $x_\parallel$ are equivalent representations, i.e. they contain the same information, just in
different coordinate systems.

**Combined matrix $V^TV$** Before we look at the combined matrix $VV^T$ consider $V^TV$.
The latter is obviously a $P \times P$-matrix and performs a transformation from the new coor-
dinate system to the old coordinate system (6) and back again (5) plus a mapping from the
high-dimensional space onto the low-dimensional space. However, since all points in the new
coordinate system lie within the low-dimensional subspace already, the mapping onto the
low-dimensional space is without any effect. Thus, only the back and forth transformation
between the two coordinate systems remains and that in combination is without any effect
either. This means that $V^TV$ is the identity matrix, which can be verified easily

$$\left( V^TV \right)_{pq} = v_p^T v_q \overset{(2)}{=} \delta_{pq}$$ \tag{8}

$$\Longleftrightarrow V^TV = 1_P.$$ \tag{9}

with $1_P$ indicating the identity matrix of dimensionality $P$. With $P = 2$, for example, we have

$$V^TV = \begin{pmatrix} v_1^T \\ v_2^T \end{pmatrix} \begin{pmatrix} v_1 & v_2 \end{pmatrix} \overset{(2)}{=} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$ 

**Projection from High- to Low-dimensional Space** As we have seen above (7) the
combined matrix $VV^T$ maps the points $x$ onto the low-dimensional subspace but in contrast
to matrix $V^T$ alone the mapped points are represented within the old coordinate system
and not the new one. It turns out that this is a projection operation with the important
property that it does not make a difference whether you apply it once or twice, i.e. $PP = P$.
Let us therefore define the projection matrix

$$P := VV^T.$$ \tag{10}
and verify that

$$PP \overset{(10)}{=} V V^T V V^T$$
$$\overset{(10)}{=} V V^T$$
$$\overset{(10)}{=} P.$$  

A closer look at $P$ shows that

$$P := V V^T$$
$$= (v_1, ..., v_P) \left( \begin{array}{c}
  v_1^T \\
  \vdots \\
  v_P^T 
\end{array} \right)$$
$$= \sum_{p=1}^{P} v_p v_p^T.$$  

$P$ is obviously an $I \times I$-matrix. If $P = I$ then projecting from the old to the new and back to the old coordinate system causes no information loss and $P = 1_I$. The smaller $P$ the more information is lost and the more does $P$ differ from the identity matrix. Consider, for example

$$v_1 := \frac{1}{2}(\sqrt{2}, -1, 1)^T \Rightarrow v_1 v_1^T = \frac{1}{4} \left( \begin{array}{ccc}
  2 & -\sqrt{2} & \sqrt{2} \\
  -\sqrt{2} & 1 & -1 \\
  \sqrt{2} & -1 & 1 
\end{array} \right),$$

$$v_2 := \frac{1}{2}(0, \sqrt{2}, \sqrt{2})^T \Rightarrow v_2 v_2^T = \frac{1}{4} \left( \begin{array}{ccc}
  0 & 0 & 0 \\
  0 & 2 & 2 \\
  0 & 2 & 2 
\end{array} \right), \text{ and}$$

$$v_3 := \frac{1}{2}(-\sqrt{2}, -1, 1)^T \Rightarrow v_3 v_3^T = \frac{1}{4} \left( \begin{array}{ccc}
  2 & \sqrt{2} & -\sqrt{2} \\
  \sqrt{2} & 1 & -1 \\
  -\sqrt{2} & -1 & 1 
\end{array} \right)$$

for which you can easily verify that $P$ (16) successively becomes the identity matrix as you take more of the $v_p v_p^T$-terms.

**Variance** The variance of a multi-dimensional data set is defined as the sum over the variances of its components. Since we assume zero-mean data, we have

$$\text{var}(x) := \sum_{i=1}^{I} \langle x_i^2 \rangle$$
$$= \langle \sum_{i=1}^{I} x_i^2 \rangle$$
$$= \langle x^T x \rangle$$

This also holds for the projected data, of course, $\text{var}(y) = \langle y^T y \rangle$. 

7
Reconstruction Error  The reconstruction error $E$ is defined as the mean square sum over the distances between the original data points $x$ and the projected ones $x_\parallel$. If we define the projection vectors ('Lotvektoren' in German) $x_\perp = x - x_\parallel$ (in contrast to the projected vectors $x_\parallel$) we can write the reconstruction error as the variance of the projection vectors and find

$$E = \langle x_\perp^T x_\perp \rangle$$

$$= \langle (x - x_\parallel)^T (x - x_\parallel) \rangle$$

$$= \langle (x - VV^T x)^T (x - VV^T x) \rangle$$

$$= \langle x^T x - 2x^T VV^T x + x^T V \underbrace{(V^T V)}_{=1_P} V^T x \rangle$$

$$= \langle x^T x \rangle - \langle x^T V \underbrace{(V^T V)}_{=1_P} V^T x \rangle$$

$$= \langle x^T x \rangle - \langle x_\parallel^T x_\parallel \rangle$$

$$= \langle x^T x \rangle - \langle y^T y \rangle.$$  

This means that the reconstruction error equals the difference between the variance of the data minus the variance of the projected data. Thus, this verifies our intuition that minimizing the reconstruction error is equivalent to maximizing the variance of the projected data.

Covariance Matrix  We have already argued heuristically that the covariance matrix $C_x$ with $C_{xij} := \langle x_i x_j \rangle$ plays an important role in performing PCA. It is convenient to write the covariance matrix in vector notation:

$$C_x := \langle xx^T \rangle = \frac{1}{M} \sum_\mu x^\mu x^{\mu T}.$$  

It is an easy exercise to show that this definition is equivalent to the componentwise one given above. Since $(xx^T)^T = xx^T$ (remember $(AB)^T = B^TA^T$), one can also see that $C_x$ is symmetric, i.e. $C_x^T = C_x$.

Eigenvalue Equation of the Covariance Matrix  Since the covariance matrix is symmetric, its eigenvalues are real and a set of orthogonal eigenvectors always exists. In mathematical terms, for a given covariance matrix $C_x$ we can always find a complete set of eigenvalues $\lambda_i$ and corresponding eigenvectors $u_i$ such that

$$C_x u_i = u_i \lambda_i$$ (eigenvalue equation),

$$\lambda_i \geq \lambda_{i+1}$$ (eigenvalues are ordered),

$$u_i^T u_j = \delta_{ij}$$ (eigenvectors are orthonormal).

If we combine the eigenvectors into an orthogonal matrix $U$ and the eigenvalues into a diagonal matrix $\Lambda$,

$$U := (u_1, u_2, ..., u_I),$$

$$\Lambda := \text{diag}(\lambda_1, \lambda_2, ..., \lambda_I),$$
then we can rewrite (30) and (28) as
\[ U^T U = 1_I \quad (\text{matrix } U \text{ is orthogonal}), \]
\[ UU^T = 1_I \quad (\text{since } U^{-1} = U^T \text{ and } U \text{ is quadratic}), \]
\[ C_x U = U \Lambda \quad (\text{eigenvalue equation}), \]
\[ U^T C_x U = U^T U \Lambda \equiv \Lambda. \]

**Total Variance of the Data** \( \mathbf{x} \) Given the eigenvector matrix \( U \) and the eigenvalue matrix \( \Lambda \) it is easy to compute the total variance of the data
\[
\langle x^T x \rangle = \langle \text{tr}(x^T x) \rangle \quad (\text{since } \lambda = \text{tr}(\lambda) \text{ for any real } \lambda) \tag{37}
\]
\[
= \langle \text{tr}(xx^T) \rangle \quad (\text{since } \text{tr}(AB) = \text{tr}(BA) \text{ for any matrices } A, B) \tag{38}
\]
\[
= \text{tr}((xx^T)) \quad (\text{since } \text{tr}(\cdot) \text{ and } \langle \cdot \rangle \text{ commute}) \tag{39}
\]
\[
\overset{(27)}{=} \text{tr}(C_x) \tag{40}
\]
\[
\overset{(34)}{=} \text{tr}(UU^T C_x) \tag{41}
\]
\[
= \text{tr}(U^T C_x U) \tag{42}
\]
\[
\overset{(36)}{=} \text{tr}(\Lambda) \tag{43}
\]
\[
\overset{(32)}{=} \sum_i \lambda_i. \tag{44}
\]

Thus the total variance of the data is simply the sum of the eigenvalues of its covariance matrix.

Notice that on the way of this proof we have shown some very general properties. From line (37) to (40) we have shown that the variance of some multi-dimensional data equals the trace of its covariance matrix. From line (40) to (42) we have shown that the trace remains invariant under any orthogonal transformation of the coordinate system.

**Diagonalizing the Covariance Matrix** We can now use matrix \( U \) to transform the data such that the covariance matrix becomes diagonal. Define \( \mathbf{x}' := U^T \mathbf{x} \) and denote the new covariance matrix by \( C_x' \). We have

\[
\mathbf{x}' = U^T \mathbf{x} \tag{45}
\]

\[
\implies C_x' = \langle x' x'^T \rangle \tag{46}
\]
\[
= \langle (U^T \mathbf{x})(U^T \mathbf{x})^T \rangle \tag{47}
\]
\[
= U^T \langle xx^T \rangle U \tag{48}
\]
\[
= U^T C_x U, \tag{49}
\]
\[
= \Lambda \tag{50}
\]

and find that the transformed data \( \mathbf{x}' \) have a diagonal covariance matrix. Working with \( \mathbf{x}' \) instead of \( \mathbf{x} \) will simplify further analysis without loss of generality.

**Variance of \( y \) for a Diagonalized Covariance Matrix** Now that we have the data represented in a coordinate system in which the covariance matrix is diagonal, we can try
to answer the question, which is the \( P \)-dimensional subspace that minimizes the reconstruction error. Our intuition would predict that it is simply the space spanned by the first \( P \) eigenvectors. To show this analytically, we take an arbitrary set of \( P \) orthonormal vectors \( \mathbf{v}'_p \), and compute the variance of \( \mathbf{y} \).

\[
\mathbf{y} := \mathbf{V}'^T \mathbf{x}' \quad \Rightarrow \quad \langle \mathbf{y}^T \mathbf{y} \rangle = \langle \mathbf{x}'^T \mathbf{V}' \mathbf{V}'^T \mathbf{x}' \rangle
\]

\[
= \langle \text{tr}(\mathbf{x}'^T \mathbf{V}' \mathbf{V}'^T \mathbf{x}') \rangle \quad \text{(since } \lambda = \text{tr}(\lambda) \text{ for any real } \lambda) \quad (51)
\]

\[
= \langle \text{tr}(\mathbf{V}'^T \mathbf{x}' \mathbf{x}'^T \mathbf{V}') \rangle \quad \text{(since } \text{tr}(\mathbf{ABC}) = \text{tr}(\mathbf{BCA}) \text{ if defined)} \quad (52)
\]

\[
= \text{tr}(\mathbf{V}'^T \mathbf{C}_x \mathbf{V}') \quad \text{(since } \text{tr}(\cdot) \text{ and } \langle \cdot \rangle \text{ commute)} \quad (53)
\]

\[
= \text{tr}(\mathbf{V}'^T \mathbf{V}' \Lambda \mathbf{V}') \quad (54)
\]

\[
= \sum_i \lambda_i \sum_p (v'_{ip})^2. \quad \text{(as one can work out on a sheet of paper)} \quad (55)
\]

**Constraints of Matrix \( \mathbf{V}' \)** Note that, since the vectors \( \mathbf{v}'_p \) are orthonormal, \( \mathbf{V}' \) can always be completed to an orthogonal \( I \times I \)-matrix by adding \( I - P \) additional orthonormal vectors. Since we know that an orthogonal matrix has normalized row as well as column vectors, we see that, by taking away the \( I - P \) additional column vectors, we are left with the constraints

\[
\sum_p (v'_{ip})^2 \leq 1 \quad \text{(row vectors of } \mathbf{V}' \text{ have norm less or equal one)}, \quad (56)
\]

\[
\sum_i (v'_{ip})^2 = 1 \quad \text{(column vectors of } \mathbf{V}' \text{ have norm one)}, \quad (57)
\]

\[
\Rightarrow \sum_{ip} (v'_{ip})^2 = P \quad \text{(square sum over all matrix elements equals } P). \quad (58)
\]

Notice that Constraint (58) is a direct consequence of Constraint (59) and does not need to be verified separately in the following considerations.

**Finding the Optimal Subspace** Since the variance (57) of \( \mathbf{y} \) as well as the constraints (58, 59, 60) of Matrix \( \mathbf{V}' \) are linear in \((v'_{ip})^2\), maximization of the variance \( \langle \mathbf{y}^T \mathbf{y} \rangle \) is obviously achieved by putting as much ‘weight’ as possible on the large eigenvalues, which are the first ones. The simplest way of doing that is to set

\[
v'_{ip} := \delta_{ip} := \begin{cases} 1 & \text{if } i = p \\ 0 & \text{otherwise} \end{cases}, \quad (59)
\]

with the Kronecker symbol \( \delta_{ip} \).

Since \( I \geq P \) we can verify the constraints

\[
\sum_p (v'_{ip})^2 \leq 1 \quad \text{(since } \sum_i \delta_{ip}^2 = 1 \text{ if } i \leq P \text{ otherwise)} \quad (60)
\]

\[
\sum_{ip} (v'_{ip})^2 = P \quad \text{(square sum over all matrix elements equals } P). \quad (61)
\]

and see from (62) that there is actually as much ‘weight’ on the first, i.e. large, eigenvalues as Constraint (58) permits.
**Interpretation of the Result** What does it mean to set $v'_{ip} := \delta_{ip}$? It means that $V'$ projects the data $x'$ onto the first $P$ axis, which in fact is a projection onto the first $P$ eigenvectors of the covariance matrix $C_x$. Thus, if we define

$$V := UV' \quad \text{(31, 61)}$$

$$V_\pi = (u_1, u_2, ..., u_P) \quad \text{(65)}$$

we can go back to the original coordinate system and find

$$y \overset{(51)}{=} V'^T x' \quad \text{(66)}$$

$$y \overset{(45)}{=} V'^T U^T x \quad \text{(67)}$$

$$y \overset{(64)}{=} V^T x \quad \text{(68)}$$

which we know has maximal variance. Thus, if we start from the original data $x$ we would set $v_p := u_p$.

For the variance of $y$ we find

$$\langle y^T y \rangle \overset{(57)}{=} \sum_{i=1}^{I} \sum_{p=1}^{P} (v'_{ip})^2 \quad \text{(69)}$$

$$\overset{(61)}{=} \sum_{i=1}^{I} \sum_{p=1}^{P} \delta_{ip}^2 \quad \text{(70)}$$

$$= \sum_{i=1}^{P} \lambda_i \quad \text{(71)}$$

which is the sum over the first $P$ largest eigenvalues of the covariance matrix. Likewise one can determine the reconstruction error as

$$E \overset{(26)}{=} \langle x^T x \rangle - \langle y^T y \rangle \quad \text{(72)}$$

$$\overset{(44, 71)}{=} \sum_{i=1}^{I} \lambda_i - \sum_{i=1}^{P} \lambda_i \quad \text{(73)}$$

$$= \sum_{i=P+1}^{I} \lambda_i \quad \text{(74)}$$

Notice that this is just one optimal set of weights. We have seen above that the projected data can be rotated arbitrarily without changing its variance and therefore without changing its reconstruction error. This is equivalent to a rotation of the projection vectors $v_p$ within the space spanned by the first eigenvectors.

**Whitening or Sphering** Sometimes it is desirable to transform a data set such that it has variance one in all directions. Such a normalization operation is called *whitening* or *sphering*. The latter term is quite intuitive, because a spherical data distribution has the same variance in all directions. Intuitively speaking sphering requires to stretch and compress the data distribution along the axes of the principal components such they have variance one. Technically speaking one first transforms the data into a coordinate system
where the covariance matrix is diagonal, then one performs the stretching along the axes, and then one transforms the data back into the original coordinate system. Principal component analysis obviously gives all the required information. The eigenvectors of the covariance matrix provide the axes of the new coordinate system and the eigenvalues \( \lambda_i \) indicate the variances and therefore how much one has to stretch the data. If the original variance is \( \lambda_i \) then one obviously has to stretch by a factor of \( 1/\sqrt{\lambda_i} \) to get variance one. Thus, spherining is achieved by

\[
\bar{x} := U \text{diag} \left( \frac{1}{\sqrt{\lambda_1}}, \frac{1}{\sqrt{\lambda_2}}, \ldots, \frac{1}{\sqrt{\lambda_I}} \right) U^T x.
\]

(75)

It is easy to verify that the spherered data \( \bar{x} \) has variance one in all directions and a unit covariance matrix.