Unsupervised learning. Applications in biomechanics

Nerea Izcue Domínguez

End of Degree Project in Mathematics
University of Zaragoza

Project supervisors:
Miguel Lafuente Blasco and Gerardo Sanz Sáiz

June 27, 2022

Preface

This end-of-degree project will revolve around Unsupervised Learning methods. Unsupervised Learning is a set of statistical tools that identify hidden patterns, or clusters, in unlabeled data. Conversely, Supervised Learning is a set of statistical tools that use labeled data to classify data or predict outcomes accurately. Among these methods can be found regression and classification. In Supervised Learning methods, there are a set of r features $X_1, X_2, ..., X_r$ and a response Y measured on n observations. The purpose is to predict the response using the r features. Unlike Supervised Learning methods, Unsupervised ones do not have an associated response variable, Y, and for that reason, prediction is not one of the goals of these methods. These techniques aim at exploring interesting patterns in the measurements on $X_1, X_2, ..., X_r$ and then draw conclusions from the raw data.

Supervised Learning is a well-known area as opposed to Unsupervised Learning, which is a relatively new area in which there is not much information available. Furthermore, Unsupervised Learning is often much more challenging than Supervised Learning. The reason is simple, in Unsupervised Learning there is no way to know if the work done is valid because the answer is simply unknown.

Nowadays, Unsupervised Learning is gaining much more importance in different fields such as finances, medicine, security, online shopping or fraud detection. In clinical cancer studies, these methods are used to study cancer gene expression data (tissues) and predict cancer at early stages. In social media, this tool can be used by content creators to obtain relevant information about the type of audience they have. With the information gathered, a grouping of the followers that share similar features could be done in order to customize the content for each targeted group. Here, the use of Clustering methods, an Unsupervised Learning technique, would detect connections and pack each follower into the correct group they belong to. In online shopping, a company aims to know what type of item they should be showing their potential customers for them to be attracted and eventually buy the product. These companies can find associations between different products and customers, which will reveal what to show in a particular advertisement to each particular customer. The use of this method could help increase their sales and revenue very favorably, leading to a customized customer approach. This is the key to getting customer satisfaction as well as retention.

Unsupervised Learning models have three main usages —clustering, association, and dimensionality reduction. This work will only put focus on two techniques —Principal Component Analysis and Clustering— whose goal is to implement some of these ideas.

On the one hand, Principal Component Analysis, also referred to as PCA, is a statistical tool often used for data visualization or data pre-processing. It allows for reducing the dimensionality, which consists on accurately describing the values of r variables with a small group of $s < r$ variables. The goal is to find a low-dimensional representation of the data that captures as much of the information as possible. PCA reduces dimensionality in exchange for a minimal loss of information. This does not mean that this method only keeps s variables, it means that the s dimensions found by Principal Component Analysis are a linear combination of the r features.

On the other hand, Cluster analysis, also called Data Segmentation, is a broad class of techniques used for finding subgroups in a data set. It attempts to partition observations into "clusters" that those within each cluster are more closely related to one another than objects assigned to different clusters. An object can be described by a set of measurements, or by its relation to other objects. Clustering is popular in many fields, which implies the existence of a big number of Clustering methods such as K-means, K-medoids and Hierarchical Clustering.

Clustering can be approached from two directions. Observations can be clustered based on the features to identify subgroups among them, or features can be clustered based on the observations to discover subgroups among the features. From now on, it will be discussed clustering observations on the basis of the features.

Summarizing both Unsupervised Learning techniques, Principal Component Analysis and Clustering try to simplify the data set given but employ different mechanisms to do so. Principal Component Analysis reduces dimensionality but maintains a good fraction of the variance that describes the data in the most precise way, while Clustering looks to find distinct groups where the observations within each group are quite similar, whereas observations in different groups are quite different from each other.

As previously mentioned, Unsupervised Learning is common practice in Informatics but Mathematics, especially in Statistics. Unsupervised Learning turns out to be very helpful in multiple ways and can be applied to solve real-world problems. The reason arises from the perspective that Unsupervised Learning mimics the human approach of learning, i.e, humans deduce patterns from their background and gradually learn more about them over time. Unsupervised Learning will manage to discover the largest number of notions and underlying patterns that matter to explain the world.

Resumen

El objetivo de este trabajo de Fin de Grado es la inmersión en un área relativamente novedosa pero que a su vez está en continuo desarrollo ya que sus aplicaciones son tan extensas y variadas que se pueden encontrar en multitud de ámbitos. Esta área conocida por distintos nombres como Machine Learning (en español, Aprendizaje de Máquinas), Aprendizaje Automático o Aprendizaje Automatizado, surge como una rama de la Inteligencia Artificial y está cada vez más arraigado en más áreas de las Matemáticas, destacando la Estadística e Informática. A su vez, este tema ha sido elegido no solo por su gran utilidad actualmente, sino además pensando en los estudios de postgrado, ya que en la carrera no se estudian las técnicas que van a ser presentadas a continuación.

Dentro del Machine Learning se distinguen principalmente tres tipos de aprendizaje automatizado: Aprendizaje Supervisado, Aprendizaje No Supervisado y Aprendizaje por Refuerzo. Este documento estará centrado en este segundo tipo —Aprendizaje No supervisado. Este tipo de aprendizaje no cuenta con las ventajas que presenta el Aprendizaje Supervisado, ya que los algoritmos de los mismos tratan un conjunto de datos sin ninguna referencia de resultados conocidos con los que poder comparar. Sin embargo, este tipo de aprendizaje es muy útil para descubrir la estructura subyacente de los datos ya que, aunque sea más impredecible comparado con los otros métodos de aprendizaje, permiten realizar tareas de procesamiento con un nivel mayor de complejidad.

A su vez, debido al gran desarrollo y auge que está teniendo esta área, dentro del Aprendizaje No Supervisado, se distinguen distintos tipos, de los cuales dos de ellos van a ser tratados en este documento —Análisis de Componentes Principales y Clustering. Se van a introducir ambos conceptos explicando la parte teórica y las aplicaciones de los mismos, así como los distintos métodos y algoritmos principales de cada uno. Para concluir, se presentará un conjunto de datos reales y completamente anónimo relativos a atletas de fondo y medio fondo, a los cuales se les ha medido distintas variables durante un estudio biomecánico, y sobre el mismo, se aplicarán las técnicas explicadas en los capítulos previos a este último.

Como se ha comentado en el párrafo anterior, el primer capítulo estará dedicado al Análisis de Componentes Principales o Principal Component Analysis (PCA) en inglés. Es una de las técnicas del Aprendizaje No Supervisado que principalmente se emplea como parte del análisis exploratorio de los datos. Generalmente, esta técnica se aplica sobre un conjunto de observaciones que tienen unas variables asociadas de las cuales solo interesa extraer información de ellas y no construir modelos de predicción mediante variables respuesta y explicativa. El objetivo es crear nuevas variables construidas mediante combinación lineal de las anteriores que estén incorreladas entre sí, a diferencia de lo que podía pasar con las variables originales.

El segundo capítulo estará centrado en la técnica de Clustering o Clusterización, la cual es una de las técnicas más utilizadas en el Machine Learning. El Clustering consiste en la detección automática de grupos o clusters homogéneos de distintas observaciones que poseen ciertas variables o características. El Clustering, también conocido como Segmentación, establece relaciones entre observaciones que tienen mismos patrones analizando grandes cantidades de datos. Se presentarán tres de los métodos de agrupamiento más utilizados en este área —K-medias (en inglés, K-means), K-medoids y Agrupamiento Jerárquico (en inglés, Hierarchical Clustering)— y se explicará en detalle cada algoritmo correspondiente a los mismos.

Por último, en el tercer capítulo, se presentará la aplicación de las dos técnicas presentadas en el documento sobre el conjunto de datos previamente mencionado. Resulta realmente interesante la aplicación de estos métodos, ya que, dentro de estos datos de carrera, como se verá más adelante, no existe ninguna

variable predictora y por lo tanto no tiene sentido el estudio de un modelo de regresión de variable explicativa y variable respuesta. Por lo tanto, con el fin de encontrar distintos perfiles de zancadas, se van a aplicar estas dos técnicas. En primer lugar, el Análisis de Componentes Principales para reducir la dimensionalidad, debido a que sería interesante y muy util pasar de un conjunto de más de diez variables a un conjunto con un menor número de variables que estén incorreladas y que expliquen la máxima información posible de los datos originales. En segundo lugar, el Clustering será aplicado para obtener una división por grupos de los distintos atletas y determinar qué representa cada grupo y que características tiene. Además, aquí se unen estas dos técnicas, ya que, para representar los datos en una gráfica, los ejes serán las dos primeras Componentes Principales que son las componentes que más información recogen.

Como comentario final, al final del documento se adjuntarán varios anexos con la información obtenida al aplicar distintas funciones sobre los datos biomecánicos con el software R. Estos apéndices proporcionan información complementaria para sustentar y apoyar lo enunciado en la memoria, ya sean gráficas que ayudan a visualizar métodos que quizás son difíciles de entender sin una imagen visual o gráficas y técnicas que apoyan los resultados obtenidos pero son redundantes, ya que solo ayudan a afirmar con mayor rotundidad las conclusiones que se presentan a lo largo del documento.

Contents

Glossary

Biomechanics Discipline that can be understood as the study of the human body in motion. Recognized authors describe it in different ways. Hay describes biomechanics as "the science that examines forces acting upon and within a biological structure and effects produced by such forces" [9], whereas Hatze states "biomechanics is the study of the structure and function of biological systems by means of the methods of mechanics" [8].

Pace Function of both stride rate and stride length calculated by the equation: $Speed = Stride\ Rate \times Stride\ Length$. It is measured in m/s.

Stride Length Distance between two successive placements of the same foot, consisting of two step lengths. It is measured in *meters*.

Step length Distance between two ground contacts, from forefoot to forefoot (e.g. left to right or vice-versa) plus the distance the treadmill belt moved during the flight time (i.e. distance during FT, which is: $FT \times Horizontal\ speed$). It is measured in *meters*.

Step Rate Number of steps a runner takes per minute. It is measured in *steps/min*.

Contact time Time from when the foot contacts the ground to when the toes lift off the ground. It is measured in *seconds*.

Flight Time Time from the toes lifting off to the initial ground contact of the consecutive footfall. It is measured in *seconds*.

Step Time Flight time plus ground contact time.

Flight Ratio Ratio of a runner's "flight" time in the air (non ground contact phase) and overall step time.

Contact Ratio Percentage of the stride spent in the air.

Step Angle Angle of the parable tangent, which was derived from the step length and the height obtained with flight time. These parameters allowed to tie step length and flight time. It is measured in *degrees*.

Ground Reaction Force Force exerted by the ground on a body in contact with it.

Vertical stiffness Ratio of the maximal force to the vertical displacement of the centre of mass as it reached its lowest point (i.e., the middle of the stance phase). It describes the global compression of a runner, that is, vertical movement of their center of mass expressed in relation to the concurrent change in vertical ground reaction force, typically assessed in the sagittal plane (90) and commonly calculated as the quotient of maximum ground reaction force and center of mass displacement (DGRF/Ddisplacement). It is measured in kNm^{-1}.

Leg stiffness Refers to how the various elements of the leg spring (i.e., muscles, tendons, and ligaments) behave under compression in the early phase of stance. It describes the ratio between the ground reaction force and the deformation in leg length. It is measured in kNm^{-1}.

Foot Strike Type Represents the three differents foot strike patterns: heel, midfoot, and forefoot striker. RunScribe gives a numeric value for each foot strike type, with 0-6 as heel strike, 6-10 as Midfoot Strike, and 10-16 being Forefoot Strike.

Stance Excursion Change in the angle of the foot measured from the ground to a line between the ankle and toes. It is measured from foot strike to maximum pronation (FS>MP) and from maximum pronation to toe off (FS>TO). It is measured in *degrees*.

Pronation Excursion Total range of angular movement as the foot rolls inward between foot strike and the point of maximum pronation. It is estimated by measuring the rearfoot angle during ground contact while running and/or walking. It is measured in *degrees*.

Impact Gs Vertical component of Peak Gs. It correlates with the ground impact force experienced at foot strike. It is measured in *Gs*.

Braking Gs Horizontal component of Peak Gs. It correlates with the braking forces experienced at foot strike. It is measured in *Gs*.

Shock RunScribe defined composite metric, which combines Impact Gs and Braking Gs into a single metric, representing the total amount of "Shock" incurred per footstep. It is measured in *degrees*.

Acronyms

PCA Principal Component Analysis

PCs Principal Components

PVE Proportion of Variance Explained

PAM Partitioning Around Medoids

CT Contact Time

FT Flight Time

FR Flight Ratio

SL Step Length

SR Step Rate

SA Step Angle

Kvert Vertical Stiffness

Kleg Leg Stiffness

SE Stance Excursion

PE Pronation Excursion

BGs BrakingGs

IGs ImpactGs

COM Center Of Mass

GRF Ground Reaction Force

FS Foot Strike Type

Chapter 1

Principal Component Analysis

1.1 Introduction and Approaching

When a very large data set is introduced, a reasonable first step would be trying to reduce its size under the condition of losing the least amount of information possible to accurately explain the original data set with the new smaller data set. The problem of finding a subset of $m < p$ variables from a data set with n observations can be addressed from two different perspectives.

- The first, and perhaps the most obvious, is to choose a small subset of the original variables.
- The second strategy, and maybe less intuitive, is to build $m < p$ different, new variables from the original ones, but constructed from them.

This second approach solves the problem of dimensionality reduction. It is possible to represent the given information with a smaller number of variables constructed as a linear combination of the originals, with a minimal amount of information loss, achieving a greater reduction in dimensionality. This problem can be addressed from three different perspectives —Descriptive, statistical and geometrical. These approaches are essentially the same and solve with similar manners the problem stated before [27].

Principal Component Analysis is the simplest of these variable-building techniques, due to its restriction to linear functions of the original variables. Furthermore, PCA allows the transformation of the original variables, which are usually correlated, into new uncorrelated variables, which makes the data interpretation easier.

Principal Component Analysis was originally defined in a statistical context by Pearson (1901) [26]. However, Hotelling (1933) [10] introduced a more usual definition in terms of successive maximization of variance 30 years later. Thus, it was not until the boom of technology that introduced sophisticated systems in the late 1940s that the method was implemented on realistically sized problems [15].

Nowadays, the computation of PCA is done by computer but to interpret the output, it is desirable to have some knowledge about how the computations are done. These concepts are based on notions from linear algebra, and the concept of eigenvalues and eigenvectors in particular.

Some books use the expression “Principal Component Analysis”, but others add an “s” to Component. Both forms are in widespread use and mean exactly the same thing.

From here to the end of the document, it will be assumed that there are n observations measurements on a set of p-variables, $X_1, X_2, ..., X_p$, arranged in a $n \times p$ dimension matrix, $\mathbf{X}$, where the columns contain the variables and the rows include the observations. Since the main interesting feature is variance, it will be assumed that the columns’ means of the matrix $\mathbf{X}$ are zero, that is, each of the variables of the matrix $\mathbf{X}$ have a mean of zero. Therefore, the covariance matrix is given by $\mathbf{S} = 1/n\, \mathbf{X}^{\mathbf{T}}\mathbf{X}$.

1.2 Principal Components

Principal Components are new uncorrelated variables that summarize a given set of correlated variables with a smaller number of representative variables explaining as much information as possible. They are constructed as a linear combination of the original variables and they are a low-dimensional representation of the data that describe the most amount of variability possible. The idea of constructing Principal Components comes from the fact that all dimensions are not equally relevant. This means that if there are n observations in a p-dimensional space, not all of them are going to be evenly interesting. Therefore, the Principal Components are the smallest set that has been chosen as the most interesting, bearing in mind that by interesting we mean the variations that the observations undergo along each dimension.

1.2.1 First Principal Component

Definition 1. The **First Principal Component** is the linear combination of a set of features $X_1, X_2, ..., X_p$

$$\mathbf{Z_1} = \mathbf{X a_1} = a_{11}X_1 + a_{21}X_2 + ... + a_{p1}X_p$$

that has the largest variance. The components of the First Principal Component, $z_{11}, z_{21}, ...z_{n1}$, are called **scores** of the First Principal Component vector. The components of the $\mathbf{a_1}$ vector, $a_{11}, a_{21}, ..., a_{p1}$ are referred to as the **loadings** of the First Principal Component and together are named as the First Principal Component loading vector, $\mathbf{a_1} = (a_{11}\ a_{21}\ ...\ a_{p1})^T$.

Note 1. Contribution is a scaled version of the squared correlation between variables and component axes. It assesses the quality of the representation of the variables of each PC. The larger the value of the contribution, the more the variable contributes to the component. The contribution of a variable (var) to a given PC is defined as:

$$\frac{\cos(\text{variable, axis})^2}{\text{total } \cos^2 \text{ of the component}} \times 100,$$

where the total $\cos^2$ of the component is equivalent to the eigenvalue associated to that component.

Since the column means of $\mathbf{X}$ are zero, that is, $\frac{1}{n}\sum_{i=1}^{n} x_{ij} = 0,\ \forall j$, $\mathbf{Z_1}$ will have mean of zero as well, that is, $\frac{1}{n}\sum_{i=1} z_{i1} = 0$. The variance will be:

$$Var(\mathbf{Z_1}) = \frac{1}{n}\mathbf{Z_1^T Z_1} = \frac{1}{n}\mathbf{a_1^T X^T X a_1} = \mathbf{a_1^T S a_1}.$$

The purpose is to maximize the variance, but if there is no restriction on the loading vector, the variance could be maximized without limit just by increasing the modulus of the vector $\mathbf{a_1}$. Hence, if there is no constraint, setting the loading to be arbitrarily large in absolute value would imply an arbitrarily large variance. Therefore, it will be imposed that the sum of squares of the loading equals one, that is, $\mathbf{a_1^T a_1} = 1$. Introducing this restriction by the Lagrange multiplier, the goal is maximizing the expression

$$\phi = \mathbf{a_1^T S a_1} - \lambda(\mathbf{a_1^T a_1} - 1).$$

The solution is obtained equalizing the derivative of ϕ with respect to $\mathbf{a_1}$ with zero,

$$\frac{\partial \phi}{\partial \mathbf{a_1}} = 2\mathbf{S a_1} - 2\lambda \mathbf{a_1} = 0,$$

which is

$$\mathbf{S a_1} = \lambda \mathbf{a_1}.$$

This means that $\mathbf{a_1}$ is an eigenvector of the matrix $\mathbf{S}$ and λ is its respective eigenvalue. Multiplying on the left by $\mathbf{a_1^T}$ this equation, is obtained that the variance of $\mathbf{Z_1}$ is

$$Var(\mathbf{Z_1}) = \mathbf{a_1^T S a_1} = \lambda \mathbf{a_1^T a_1} = \lambda.$$

Consequently, since the purpose is maximizing the variance, λ would be the largest eigenvalue of the matrix $\mathbf{S}$ and its respective eigenvector will be the First Principal Component loading vector.

As previously stated in the introduction, one of the perspective the problem can be addressed from is geometrical. The First Principal Component loading vector defines a direction in the feature space along with the data varies the most. Projecting the n data points in this direction, the projected values are the Principal Component Scores [13].

1.2.2 Second Principal Component

After the First Principal Component $\mathbf{Z_1}$ has been defined, the Second Principal Component $\mathbf{Z_2}$ can be computed.

Definition 2. The **Second Principal Component** is the linear combination of a set of features $X_1, X_2, ..., X_p$

$$\mathbf{Z_2} = \mathbf{X a_2} = a_{12}X_1 + a_{22}X_2 + ... + a_{p2}X_p$$

that has the largest variance among all the linear combinations uncorrelated with $\mathbf{Z_1}$. The components of the Second Principal Component, $z_{12}, z_{22}, ...z_{n2}$, are called **scores** of the Second Principal Component vector. The components of the $\mathbf{a_2}$ vector, $a_{12}, a_{22}, ..., a_{p2}$ are referred to as the **loadings** of the Second Principal Component and together are named as the Second Principal Component loading vector, $\mathbf{a_2} = (a_{12}\ a_{22}\ ...\ a_{p2})^T$.

The best way to compute the Second Principal Component is by finding the best plane of projection of the $\mathbf{X}$ variables. The goal is to maximize the sum of the variances of $\mathbf{Z_1}$ and $\mathbf{Z_2}$, where the plane of projection is spanned by the vectors $\mathbf{a_1}$ and $\mathbf{a_2}$ [27]. Therefore, the function is:

$$\psi = \mathbf{a_1^T S a_1} + \mathbf{a_2^T S a_2} - \lambda_1(\mathbf{a_1^T a_1} - 1) - \lambda_2(\mathbf{a_2^T a_2} - 1)$$

with the same restriction imposed on the calculus of the First Principal Component, the sum of squares of the loadings equals one, that is, $\mathbf{a_i^T a_i} = 1$ for $i = 1, 2$. In the same way as before, equalizing the derivative of ψ with respect to $\mathbf{a_i}$ for $i = 1, 2$ with zero:

$$\begin{cases} \dfrac{\partial \psi}{\partial \mathbf{a_1}} = 2\mathbf{S a_1} - 2\lambda_1 \mathbf{a_1} = 0, \\ \dfrac{\partial \psi}{\partial \mathbf{a_2}} = 2\mathbf{S a_2} - 2\lambda_2 \mathbf{a_2} = 0. \end{cases}$$

The solution of these equations is:

$$\begin{cases} \mathbf{S a_1} = \lambda_1 \mathbf{a_1}, \\ \mathbf{S a_2} = \lambda_2 \mathbf{a_2}. \end{cases}$$

where $\mathbf{a_1}$ and $\mathbf{a_2}$ must be eigenvectors of the matrix of correlations, $\mathbf{S}$. Replacing these values with norm one in the first function, the maximum value of the objective function is

$$\psi = \lambda_1 + \lambda_2.$$

Therefore, λ_1 and λ_2 must be the largest eigenvalues of the covariance matrix, $\mathbf{S}$, and $\mathbf{a_1}$ and $\mathbf{a_2}$ its respective eigenvectors.

Note 2. Covariance between $\mathbf{Z_1}$ and $\mathbf{Z_2}$ is zero due to $\mathbf{a_1}$ and $\mathbf{a_2}$ being orthogonal, that is $\mathbf{a_1^T S a_2} = 0$, since $\mathbf{a_1^T a_2} = 0$.

Note 3. The same result is obtained if instead of maximizing the sum of the variances, which is the trace of the matrix of covariances, the generalized variance, which is the determinant of the matrix of covariances, is maximized [27].

1.2.3 Generalization

The spread of Principal Components of the p components is defined by the largest p eigenvalues of the matrix of covariances, $\mathbf{S}$. Therefore, the rank of the matrix $\mathbf{X}$, thus the rank of the matrix $\mathbf{S}$, is p, existing as many Principal Components as variables obtained calculating the eigenvalues of the matrix of covariances, $\mathbf{S}$.The method to obtain the p eigenvalues is by:

$$|\mathbf{S} - \lambda \mathbf{I}| = 0$$

and the eigenvectors by:

$$(\mathbf{S} - \lambda_i \mathbf{I})\mathbf{a_i} = \mathbf{0} \quad i = 1, ..., p.$$

The eigenvalues λ_i are positive and real, since the matrix $\mathbf{S}$ is positive-definite and symmetric. A consequence of $\mathbf{S}$ being symmetric is that if λ_i and λ_j are two eigenvalues such that $\lambda_i \neq \lambda_j$, their respective eigenvectors are orthogonal. For $\mathbf{S}$ being symmetric and the properties of eigenvalues:

$$\begin{cases} \mathbf{a_i^T S a_j} = (\mathbf{a_i^T S a_j})^T = \mathbf{a_j^T S a_i} = \mathbf{a_i^T} \lambda_\mathbf{j} \mathbf{a_j}, \\ \mathbf{a_i^T S a_j} = \mathbf{a_j^T} \lambda_\mathbf{i} \mathbf{a_i}. \end{cases}$$

Now, since $\lambda_i \neq \lambda_j$:

$$\mathbf{a_j^T a_i} = \mathbf{a_i^T a_j} = 0.$$

Therefore the eigenvectors are orthogonal. Calling $\mathbf{Z} = (\mathbf{Z_1}\ \mathbf{Z_2}\ ...\ \mathbf{Z_p})$ to the matrix whose columns are the p Principal Components, the relationship between the original matrix $\mathbf{X}$ and the matrix of principal components $\mathbf{Z}$ is given by:

$$\mathbf{Z} = \mathbf{XA}$$

where $\mathbf{A} = (\mathbf{a_1}\ \mathbf{a_2}\ ...\ \mathbf{a_p})$ and $\mathbf{A^T A} = \mathbf{I}$. Hence, the Principal Components can be computed with an orthogonal transformation $\mathbf{A}$ over the matrix $\mathbf{X}$, obtaining new uncorrelated variables $\mathbf{Z}$.

Note 4. The interpretation of this operation can be seen as choosing new coordinated axes that coincide with the natural axes of the data [27].

1.3 Properties

Some properties of the PCs will be listed below:

- **Preservation of the Initial Variance**: The sum of the variance of the PCs is equal to the sum of the variance of the original variables. Indeed, since $Var(Z_i) = \lambda_i$ and the sum of the eigenvalues is the trace of the matrix of covariances:

  $$Var(Z_1) + ... + Var(Z_p) = \lambda_1 + ... + \lambda_p = tr(S) = Var(X_1) + ... + Var(X_p).$$

  Although the sum of the variances is the same in the PCs and in the original variables, the distribution is completely different.

  Furthermore, the generalized variance of the PCs is equal to the generalized variance of the original variables. Since the determinant of the original covariance matrix is the product of eigenvalues and the PCs covariance matrix is diagonal with the eigenvalues in it, it is easy to check the proposition:

  $$|\mathbf{S_x}| = \lambda_1 ... \lambda_p = \prod_{i=1}^{p} Var(Z_i) = |\mathbf{S_z}|.$$

- **Proportion of Variance Explained (PVE)**: The PVE is the variance explained by the m-th Principal Component divided by the total variance of the data set. The PVE is a positive quantity. Since the variance of a PC is its associated eigenvalue, corresponding to a real nonzero eigenvector, and the total variance is the sum of all the eigenvalues of the matrix, the PVE explained by the component m is:

  $$PVE_m = \frac{\lambda_m}{\sum_i^p \lambda_i}.$$

Note 5. The PVE can be interpreted similarly as the R^2 in the regression framework of the approximation for $\mathbf{X}$ given by the first m Principal Components [13].

- For $r < p$ variables of the original data set $\mathbf{X}$, the optimal lineal prediction is achieved by the first r Principal Components.
- The covariances between each Principal Component and the data set $\mathbf{X}$ are given by the product of the eigenvector's coordinates, corresponding to a real nonzero eigenvalue:

$$Cov(\mathbf{Z_i}; \mathbf{X_1}, ..., \mathbf{X_p}) = \lambda_i \mathbf{a_i} = (\lambda_i a_{i1}, ..., \lambda_i a_{ip})$$

  where $\mathbf{a_i}$ is the loading of that i-th Principal Component.

  Indeed, remember that $\mathbf{Z} = \mathbf{XA}$ and $\mathbf{a_i S} = \lambda_i \mathbf{a_i}$. Therefore, the covariance matrix between $\mathbf{Z}$ and $\mathbf{X}$ is:

$$Cov(\mathbf{Z}, \mathbf{X}) = \frac{1}{n}\mathbf{Z^T X} = \frac{1}{n}\mathbf{A^T X^T X} = \mathbf{A^T S} = \mathbf{DA^T}$$

  where $\mathbf{A} = (\mathbf{a_1}, \mathbf{a_2}, ..., \mathbf{a_p})$ is the loading matrix whose columns are the eigenvectors of $\mathbf{S}$ and $\mathbf{D} = diag(\lambda_1, \lambda_2, ..., \lambda_p)$ is the diagonal matrix whose diagonal elements are the eigenvalues.
- The correlation between a PC and $\mathbf{X_i}$ is given by:

$$Cor(\mathbf{Z_i}, \mathbf{X_j}) = \frac{Cov(\mathbf{Z_i}, \mathbf{X_j})}{\sqrt{Var(Z_i)Var(X_j)}} = \frac{\lambda_i a_{ij}}{\sqrt{\lambda_i s_j^2}} = \frac{\sqrt{\lambda_i} a_{ij}}{s_j^2}$$

  where s_j is the typical deviation of the variable $\mathbf{X_i}$.
- Standardizing the Principal Components, which can be achieved by dividing each Principal Component by its variance, leads to the same result as applying a multivariate standardization to the initial data.

  On the one hand, PCA transforms the starting data into new uncorrelated variables with different variance, which can be seen geometrically as rotating the ellipse defined by the original points so that they can run into the natural axes. On the other, if the multivariate standardization is applied to the initial data, it transforms into uncorrelated variables with unit variance, which can be seen geometrically as finding the natural axes and making the points run into them and then standardizing those points [27].

1.4 Interpretation

1.4.1 Scaling the Variables

In the introduction, it was recommended that the variables should have a mean of zero in most cases. Unlike many other statistical techniques, scaling the variables has a substantive effect on the model obtained. The result depends on the scales with which the variables were measured, and implies that the covariance matrix, $\mathbf{S}$, will change if the units of measurement on one or more of the variables change. Standardization is useful because most changes of scale are linear transformations of the data, which share the same set of standardized data values. Therefore, variables should be scaled to have a standard deviation of one before performing PCA since depending on an arbitrary choice of scaling results in unwished features for some properties of PCA.

However, there are situations where it might not be recommended to scale the variables to have a standard deviation of one. For example, in case all the variables are measured in the same unit or if all the variables undergo a common change of scale, because it implies that the new covariance matrix is the old matrix multiplied by a scalar. Therefore, in essence, the matrix will have the same eigenvectors and the same proportion of total variance explained by each Principal Component [16].

1.4.2 Uniqueness of the Principal Components

Each Principal Component loading vector and score vector is unique, up to a sign flip. This is because Principal Component loadings determine a direction in the space, and therefore changing the sign does not modify the direction. In the same way, the variance of $\mathbf{Z}$ is the same as the variance of $-\mathbf{Z}$ [13].

1.4.3 Atypical Values

Before computing any PCA, a data cleaning must be performed. One of the elements that corrupt the data to a greater extent is atypical values. If there are atypical values, the covariance matrix could be distorted. PCs could be used to detect atypical multivariate values because an outlier will take a PC and will appear as an outlier in that component. However, this can only be employed with atypical isolated values, and there is no guarantee that it can detect groups of atypical values [27].

1.4.4 Choosing the Number of Components

One of the most challenging questions to answer about PCA is how many Principal Components are needed. This question arises from the need to use the smallest number of Principal Components required to understand and explain the data. It is not easy to choose the adequate number of Principal Components. On the one hand, if fewer PCs are selected, the data will not be clearly explained due to the incomplete model representation. On the other, if more PCs than necessary are selected, the model will be overloaded with too much unnecessary information. Although there are many different methods to compute the number of Principal Components, the decision is very subjective, because, in the end, there is no single or clear answer to this question.

There are diverse methods to select the optimal number of PCs [35]: Elbow method, Akaike Information Criterion (AIC), Minimum Description Length (MDL), Imbedded Error Function (IEF), Cumulative Percent Variance (CPV), Scree Test on Residual Percent Variance (RPV), Average Eigenvalue (AE), Parallel Analysis (PA), Autocorrelation (AC), Cross Validation based on the PRESS and R ratio, Variance of the Reconstruction Error (VRE), etc. Nonetheless, these are the most common ones:

- The **Elbow method** is based on the proportion of variance explained or cumulative proportion of variance explained to select the number of PCs. The method consists on examining a scree plot that devises the explained variance against the number of Principal Components and looking for a point, sometimes called "*elbow*". To find the number of PCs a breakpoint or where the plot stabilizes should be located. This method is pretty easy to compute, the only issue might be that in some cases it is difficult to find an elbow if the curve decreases smoothly.

- The **Average Eigenvalue Approach (AE)** or Eigenvalue-One Rule is a criterion that accepts all eigenvalues with values above the average eigenvalue and rejects those below the average. For correlation-based PCA the average eigenvalue is $\frac{1}{p}trace(\mathbf{S})$, which is 1. Therefore, with this rule, the eigenvalues above 1 are selected.

- The **Cumulative Percent Variance (CPV)** is a measure of the percent variance captured by the first m PCs:
$$CPV(m) = 100\frac{\sum_{i=1}^{m}\lambda_i}{\sum_{j=1}^{p}\lambda_j}\%.$$
There is no rule of thumb when it comes to choosing an optimal percentage. The standard practice changes depending on the area this rule is applied. For example, in exact sciences a percentage below 90% wouldn't be acceptable whereas in biomechanics, a percentage of 70% would be, in many cases, satisfactory. Moreover, The CPV method to select the number of PCs is often ambiguous because the CPV is monotonically increasing with the number of PCs.

As stated before, even though there are many methods to select PCs, there is no widely-accepted way to decide the exact numbers of PCs [13]. A graphical example representing the application of these methods can be seen in the appendix A.

Chapter 2

Clustering

2.1 Introduction and Approaching

Clustering is present in almost every aspect of daily life. It can be found in multiple fields in which its application is very useful. For example, in marketing, this method could be employed to find groups of customers with similar profiles that are prone to purchasing similar items. In streaming services, it identifies viewers who have similar behaviors based on the number of movies or TV shows watched, the genre of the movie, the score it gets from the viewers and critics, age of the user, geographic zone, etc. It offers them a personalized interface that they enjoy, making them more engaged with the streaming service. Regarding health insurance, it helps identify groups of customers according to the number and average age of household members, number of doctor visits per year, number of illnesses or chronic condition per household, etc.

Clustering or Data Segmentation is a data mining technique for grouping or segmenting objects in subgroups or "clusters". The goal of this method is partitioning the data into different groups where the observations within each cluster are more closely related to each other, while objects assigned to different clusters differ from each other.

Therefore, it is essential to define the notion of *degree of similarity* or *dissimilarity between objects*. It depends on the meaning of similarity or difference for an object to be clustered in one group or another. This consideration should be made based on the type of the data set because the measure chosen could vary depending on this.

Clustering was firstly suggested by Driver and Kroeber in 1932 in the field of anthropology [4], but was quickly introduced in psychology by Joseph Zubin in 1938 [36] and Robert Tryon in 1939 [34]. However, it was not until 1943 that Cattell made these methods noted in his trait theory classification in personality psychology [3]. In 1967, the Jenks optimization method was proposed by George Frederick Jenks [14], which is a clustering method that arranges each value into the best class possible. Nowadays, Clustering is a common statistical technique for data analysis used in many areas, and therefore, many clustering methods have been developed. Since there are many algorithms, in this work only two of the best-known clustering approaches are going to be presented —Partitioning and Hierarchical Clustering.

From here until the end, it will be assumed that Clustering will be made over observations on the basis of the features to identify subgroups among them. Even it can be done either way, if someone wants to do so, it is as simple as transposing the data matrix.

Moreover, to make variables comparable, the variables will be standardized, that is, scaled to have a standard deviation of one and a mean of zero. Standardization is key to comparing variables which are measured on different scales, and it should be done before matrix computation [7], [13], [17].

2.2 Clustering Dissimilarity Measure

As stated in the introduction, classifying objects into one cluster or another might depend on what dissimilarity measures have been used. Hence, the choice of dissimilarity measure is very important

because it has a great effect on the Clustering results.

Consequently, the first step before Clustering is to classify observations depending on the distance or dissimilarity between them. The result of this computation is known as a **dissimilarity** or **distance matrix**. The size of this matrix is $N \times N$, where N is the number of objects, and the elements of the matrix d_{ij} record the proximity between the i-th and j-th objects. This matrix of dissimilarities with non-negative entries and zero diagonal elements $d_{ii} = 0$, $i = 1, 2, ..., N$, are provided as input to the Clustering algorithm. There are several distance measures that define the dissimilarity between two objects whose influence on the shape of the cluster is significant. Let x and y be two vectors of length n and $\mathbf{S}$ the covariance matrix. The classical methods for distance measures are:

- **Euclidean distance**:
$$d_{euc}(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} = \|x - y\|_2.$$
- **Manhattan distance**:
$$d_{man}(x,y) = \sum_{i=1}^{n} |(x_i - y_i)|.$$
- **Mahalanobis distance**:
$$d_{mah}(x,y) = \sum_{i=1}^{n} \sqrt{(x_i - y_i)^T \mathbf{S}^{-1}(x_i - y_i)}.$$

The most used distance when all the variables are continuous is the Euclidean distance. It is not advisable to use the Mahalanobis distance, since the only covariance matrix available is that of the whole sample, which may show very different correlations from those between variables within clusters.

Other measures exist, since for non-quantitative attributes squared distance may not be appropriate. Correlation-based distance considers two objects to be similar depending on their features. In these types of measures, the shape of the observation is more significant than the magnitude. In some fields, such as gene expression, these distances are used commonly. The most frequent one, which measures the degree of a linear relationship between two profiles, is [17]:

- **Pearson correlation distance**:
$$d_{cor}(x,y) = 1 - \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}.$$

Hence, depending on the type of the data set and the specific question considered, it should be decided what dissimilarity measure to choose. Euclidean distance groups observations with high values of features and the same happens, with observations with low values. On the contrary, if it is more important the overall profile or shape instead of magnitude, based correlation measured might be more interesting [7].

2.3 Partitioning Clustering

Partitioning Clustering is a cluster analysis method that groups observations based on the characteristics and similarity of the data. One of the main characteristics of these methods is that the number of clusters must be specified before computing them. Two of the main Partitioning Clustering methods are:

- **K-means Clustering**: It groups n observations into K clusters, which are represented by means of the data points belonging to the cluster, in which the criterion on belonging to one cluster or another depends on the distance to the cluster with the nearest mean. It is very sensitive to outliers.
- **K-medoids Clustering**: Quite similar to K-means with the difference that the cluster is not represented by the mean of the data, but an actual data point of the cluster is chosen as *center*. It is less sensitive to outliers.

2.3.1 K-means

K-means Clustering is one of the most popular Unsupervised Learning algorithms for grouping a given data into K non-overlapping clusters. K-means groups observations with similar features in the same cluster, whereas observations very dissimilar from each other are grouped in different clusters. To perform this method, the number of clusters must be specified before computing, the variables have to be quantitative and the dissimilarity measure distance must be Euclidean distance [7].

Let $\mathscr{C} = \{C_1, C_2, ..., C_K\}$ a partition of the set $\{1, 2, ..., n\}$. That is, $C_1, C_2, ..., C_K$ are the sets containing the indices of the observations in each cluster, and so the partition satisfies two different properties:

- Each observation belongs to at least one of the clusters, that is, $C_1 \cup C_2 \cup ... \cup C_K = \{1, ..., n\}$.
- Each observation only belongs to one cluster, they cannot belong to more than one cluster, that is, $C_i \cap C_j = \varnothing$, $\forall i \neq j$.

To sum up, each observation belongs to one and only one cluster. This can be seen as a many-to-one mapping $C_k = C(i)$, so if the i-th observation belongs to the k-th cluster, then $i \in C_k$.

The concept of within-cluster variation is crucial since the algorithm is based on minimizing this amount. Therefore, the **within-cluster variation** can be defined as:

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,j \in C_k} \sum_{m=1}^{p} (x_{im} - x_{jm})^2$$

with $|C_k|$ the number of observations of the cluster k-th. It measures the amount by which the observations within a cluster differ from each other.

Hence, minimizing the within-cluster variation is minimizing the sum of all the pairwise squared Euclidean distances between the observations in the k-th cluster.

This leads into the optimization problem that defines K-mean Clustering, which, as stated before, consists on minimizing the total within-cluster variation, that is:

$$\underset{C_1,...,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} W(C_k) \right\},$$

which is

$$\underset{C_1,...,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,j \in C_k} \sum_{m=1}^{p} (x_{im} - x_{jm})^2 \right\}. \tag{2.1}$$

The goal is to find an algorithm that solves this optimization problem. However, finding a solution to this problem is really difficult but a rather simple algorithm that provides a local optimum, which is often a good solution, is shown below:

Algorithm 1 K-Means algorithm

1. Assign randomly to each of the observations a number from 1 to K. This is going to serve as the initial cluster assignment for the observations.
2. For each of the K clusters, compute the vector of the p feature means for the observations in the k-th cluster, known as the k-th **centroid**.
3. Assign each observation to the cluster whose centroid, μ_i, $i \in 1, 2..., K$, is closest (based on the Euclidean distance). It can be written as:

$$C(i) = \underset{1 \leq k \leq K}{\text{argmin}} \|x_i - \mu_k\|_2^2.$$

4. Repeat steps 2. and 3. until the cluster assignments stop changing.

Steps 2. and 3. reduce the value of the criterion at each step, therefore the algorithm is guaranteed to converge. This can be easily seen thanks to this identity:

$$\frac{1}{|C_k|}\sum_{i,j\in C_k}\sum_{m=1}^{p}(x_{im}-x_{jm})^2 = 2\sum_{i\in C_k}\sum_{m=1}^{p}(x_{im}-\bar{x}_{km})^2 \tag{2.2}$$

where $\bar{x}_{km} = \frac{1}{|C_k|}\sum_{i\in C_k} x_{im}$ is the mean for feature m in cluster C_k. Therefore,

- In step 2, the cluster means for each feature minimize the sum of squared deviations.
- In step 3, reallocating the observations can only improve (2.2).

Since the value of (2.1) never increase, when the cluster assignments stop changing, it means a local optimum has been reached. However, it is recommendable to run multiple times the algorithm changing the initial values because the results will depend on the initial random assignments. This is due to the algorithm finds a local optimum and not a global optimum [13].

2.3.2 K-medoids

On some occasions, K-means is not appropriated because the obtained results are not precise due to some outliers that have a high influence on the method. This is because the Euclidean distance is being used, which gives the highest influence to the largest distances. Hence, other methods, such as **K-medoids** might be more suitable for partitioning a data set into K clusters since it is robust against outliers that produce a very large distance. However, although K-medoid is less sensitive to outliers and noise compared to K-means, it also requires specifying the number of clusters before computing.

As opposed to K-means that uses *centroids*, in K-medoids, each cluster is represented by one of the data point in the cluster, known as **medoid**. The medoid is the data point within a cluster whose average dissimilarity between it and all the other points of the cluster is minimal. It can be seen as the most centralized point of the cluster [17].

K-medoids is far more computationally intensive than K-means. However, the algorithm is not complex and can be computed as follows [7]:

Algorithm 2 K-Medoids algorithm

1. For a given cluster assignment $\mathscr{C} = \{C_1, C_2, ..., C_K\}$, the medoid should be found. That is, the observation in the cluster that minimizes the total distance to other points in the cluster:

$$i_k^* = \underset{i:C(i)=k}{\operatorname{argmin}} \sum_{C(j)=k} \|x_i - x_j\|_2^2.$$

   Therefore, $m_k = x_{i_k^*}$, $k = 1,2,...,K$ are the current estimates of the medoids.

2. Assign each observation to the cluster whose medoid, m_i, $i \in 1,2...,K$, is closest (based on the Euclidean distance), that is,

$$C(i) = \underset{1\le k\le K}{\operatorname{argmin}} \|x_i - m_k\|_2^2.$$

3. Repeat step 1 and 2 until the cluster assignments stop changing.

An alternative algorithm is the **Partitioning Around Medoids** algorithm (**PAM**) proposed by Kauffman and Rousseeuw (1990) [20]. Instead of searching for the medoid of each cluster, it provisionally exchanges each center i_k with an observation which is not currently a center and selects the exchanges (until there are no better ones) that produce the greatest reduction in

$$\underset{C,\{i_k\}_1^K}{min} \sum_{k=1}^{K}\sum_{C(i)=k} d_{ii_k}.$$

2.4 Hierarchical Clustering

Pre-specifying the number of clusters to be produced by the partitioning method can sometimes be a disadvantage rather than an advantage. Therefore, there is an alternative approach known as Hierarchical Clustering that does not require such specifications. The only thing the user has to specify is the dissimilarity measure between the observations. These methods produce a hierarchical representation in which each cluster is the result of the union of clusters at the next lower level. At the lowest level, there are as many clusters as observations and at the highest level, there is only one cluster which contains all of the observations. Hierarchical Clustering can be subdivided into two types [17]:

- **Agglomerative Clustering**, also known as *Bottom-Up Clustering*. It is the most common type of Hierarchical Clustering, which consists in considering each observation as one single cluster at the beginning and at each level successively merging the most similar groups into a single one. The two groups that are merged together are the two groups of observations with the smallest intergroup dissimilarity.

- **Divisive Clustering**, also known as *Top-Down Clustering*. It is the inverse of Agglomerative Clustering and starts at the top considering only one cluster with all the observations in it and at each level recursively splits the most heterogeneous cluster of observations into two new clusters.

Each level of the hierarchy represents a particular grouping of the data into disjoint clusters. There are $N-1$ levels in the hierarchy and the total ensemble represents an ordered sequence of such groupings.

A feature that all agglomerative and some divisive methods have is that the dissimilarity between merged clusters is monotone increasing with the level of the fusion. The earlier this union occurs, the more similar are the groups of observations to each other. On the other hand, the later this union occurs, the more dissimilar are the groups of observations to each other. The lowest level in which each observation is a cluster is plotted at zero height. This tree-based representation of the groups, called **dendrogram** is the graphical result of these methods.

A dendrogram produces in a graphical format a complete description of the Hierarchical Clustering. The height of the dendrogram, represented in the vertical axis, indicates how different two groups are. As stated before, the lowest the height, the more similar the observations, and vice versa. However, the similarity of two observations can not be deduced from the proximity along the horizontal axis, unlike the vertical axis.

Hence, the clusters are obtained when the dendrogram is cut horizontally at a determined height. The vertical lines that intersect the dendrogram establish the disjoint clusters. The height of the dendrogram can be seen as the K in the partitioning clustering because they determine the number of clusters obtained.

Nevertheless, the term hierarchical means that the clusters obtained by cutting the dendrogram at some height are nested within the cluster obtained by cutting the dendrogram at some higher height. This cluster hierarchy might be unrealistic for some data sets and some interpretations should be taken with caution because the summary of the data is only valid if this hierarchical structure is satisfied. Since hierarchical methods impose this hierarchical structure whether or not the data follows it, the *cophenetic correlation coefficient* measures it.

The **cophenetic dissimilarity**, C_{ij}, between two observations, (i, j), is the height of the dendrogram where the groups that the observations lie together, join into the same cluster. It is used to determine how similar are two different groups of observations in order to merge them into the same cluster. The **cophenetic correlation coefficient** is the correlation between the $\binom{N}{2} = N(N-1)/2$ pairwise observation dissimilarities, d_{ij}, input to the algorithm and their corresponding cophenetic dissimilarities, C_{ij}. This coefficient measures the quality of the dendrogram to preserve the pairwise distances between the original unmodeled data points.

Therefore, the dendrogram should be viewed as a description of the clustering structure rather than a graphical summary due to the difference between cophenetic dissimilarities from a data set and their general dissimilarities, which is usually very distant [7], [13].

2.4.1 Agglomerative Clustering

Agglomerative Clustering groups objects in clusters based on their similarity. It works in a bottom-up manner. First, a dissimilarity measure between each pair of observations must be defined. At the beginning, every observation is considered its own cluster. The two clusters that are most similar to each other are merged into a single cluster, so there is a $n-1$ cluster left. Next, the two clusters that are most similar to each other are fused again. Therefore, pairs of clusters are successively fused until all clusters have been merged into one single cluster that contains every observation of the data set.

Previously, the concept of dissimilarity between a pair of observations was defined, but now, this term has to be extended to a pair of groups of observations. This is what is known as **linkage**. Let G and H be two different clusters and $|G|$ and $|H|$ the number of observations of each cluster. There are some common types of linkages [7]:

- **Complete linkage**. Also known as *Furthest-Neighbor* technique, takes the maximal intercluster dissimilarity. After computing all the pairwise dissimilarities between the observations of two different clusters, it takes the most dissimilar or the furthest pair:

$$d_{CL}(G,H) = \max_{i\in G, j\in H} d_{ij}.$$

- **Single linkage**. Also known as *Nearest-Neighbor* technique, takes the minimal intercluster dissimilarity. After computing all the pairwise dissimilarities between the observations of two different clusters, it takes the closest or the nearest pair:

$$d_{SL}(G,H) = \min_{i\in G, j\in H} d_{ij}.$$

- **Average linkage**. It takes the mean intercluster dissimilarity. After computing all the pairwise dissimilarities between the observations of two different clusters, it takes the average of the dissimilarities:

$$d_{AL}(G,H) = \frac{1}{|G||H|}\sum_{i\in G}\sum_{j\in H} d_{ij}.$$

- **Centroid linkage**. It takes the dissimilarity between the centroid of the two different clusters:

$$d_{Centroid}(G,H) = d(\mu_G,\mu_H) = d\left(\frac{1}{|G|}\sum_{i\in G} x_i, \frac{1}{|H|}\sum_{j\in H} x_j\right).$$

- **Ward's minimum variance method**. It takes the total within-cluster variance. After computing the total within-cluster variance of all clusters, it merges the two of them with minimum between-cluster distance.

If all the observations within a cluster are close together as compared with observations in different clusters, the clusters are *compact*. However, if the clusters are not compact, the results between the different types of linkages will differ. An important concept to understand what effect produces the different types of linkages is the **diameter** of a group of observations. It is the largest dissimilarity among its members:

$$D_G = \max_{i,j\in G} d_{ij}.$$

Single linkage tends to combine observations linked by a series of close intermediate observations. The clusters produced are usually rare in compactness, since they lack being similar to one another. Therefore, single linkage produces clusters with very large diameters.

Complete linkage is the opposite. Instead of being rare in compactness, it is rare in closeness, since it tends to produce compact clusters with small diameters. Sometimes, observations of one cluster can be closer to other cluster observations than observations of its own cluster.

Average linkage tries to balance single and complete linkage producing relatively compact and relatively far apart clusters.

Hence, complete linkage, average linkage and Ward's method are generally preferred over the other linkages types because they tend to exhibit more balanced dendrograms [13].

Once the dissimilarity measure has been chosen, the algorithm that provides the hierarchical clustering dendrogram is extremely easy to compute. The resulting dendrogram will depend quite strongly on the type of linkage used. It can be computed as follows:

Algorithm 3 Hierarchical Clustering algorithm

1. Considering the n observations, measure all the $\binom{n}{2} = n(n-1)/2$ pairwise dissimilarities, and treat each observation as its own cluster.
2. For $i = n, n-1, \ldots, 2$:
   (a) Fuse the pair of clusters that are less dissimilar within all pairwise inter-cluster dissimilarities among the i clusters. The dissimilarity between these pairs of clusters is the height in the dendrogram where the fusion takes place.
   (b) Compute the new pairwise inter-cluster dissimilarities among the $i-1$ remaining cluster.

2.5 Interpretation

2.5.1 Standardization

Before performing clustering and prior to the computing of the dissimilarity between the observations, one should acknowledge whether the variables should be standardized or not. This means considering if the variables should be scaled to have a standard deviation of one and mean of zero or not.

If the variables are measured in different scales, they should be standardized to avoid algorithms to depend on irrelevant changes in the measurement scale. However, standardizing the variables adds same weight to them, independently to the original variance, which might not always be appropriate. If the variables are not standardized, the dissimilarity measure will rely especially on the variables with higher values and the result will change completely if the scale is changed [27].

2.5.2 Determining the Optimal Number of Clusters

In order to compute K-means or K-medoids, the number of clusters have to be determined before starting the algorithm. There are many methods to select the optimal number of clusters, but the answer is not straightforward nor unique. There is no simple answer, therefore any solution that gives some interesting features have to be taken into account. Knowing the exact number of clusters is important because if the number of clusters given is fewer than the optimal value, the result produced by the algorithm does not capture the important aspects or the essence of the underlying data. However, if the assumed K value is greater than the optimal value, then the algorithm builds a model that will represent unnecessary associations between data points.

This issue is traditionally overcome by trial-and-error, which is both computationally inefficient and time-consuming. There are several methods available to identify the optimal number of clusters for a given data set. These methods can be divided into two classes:

- **Direct methods**: These methods try to optimize a cluster criterion (e.g., *elbow* and *silhouette* methods).
- **Statistical testing methods**: These methods compare evidence against null hypothesis (e.g., *gap statistic* method).

There are other methods to compute the optimal number of clusters but only a few provide reliable and accurate results such as the Elbow, Average Silhouette, and Gap Statistic method. However, between these methods, the results might differ when applied to the same data set [25].

- The **Average silhouette method** consists of calculating a silhouette value for every data point, the mean of which is used to find the optimal number of clusters. It measures how well each observation lies in its respective cluster compared to other clusters. Therefore, a high average silhouette indicates that the clusterization is good because it represents how similar a data point is to its own cluster when compared to all the other clusters or cluster centroids. The range varies from -1 to $+1$. Hence, high values mean the clustering structure is appropriate and low or negative values means that the cluster structure is not proper. The silhouette value is defined as [29]:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} = \begin{cases} 1 - a(i)/b(i), & \text{if } a(i) < b(i), \\ 0, & \text{if } a(i) = b(i), \\ b(i)/a(i) - 1, & \text{if } a(i) > b(i). \end{cases}$$

  - $a(i) = \frac{1}{|C_k| - 1} \sum_{j \in C_k, i \neq j} d(i, j)$ for an observation $i \in C_k$, where $|C_k|$ is the number of observations of the cluster k and $d(i, j)$ the distance between the observations i and j of the cluster C_k. It is the average dissimilarity of i to all other objects of C_k.
  - $b(i) = \min_{k \neq j} \frac{1}{|C_j|} \sum_{j \in Cj} d(i, j)$. It is the smallest mean distance of $i \in C_k$ to all points in any other cluster C_j, of which $i \notin C_j$.

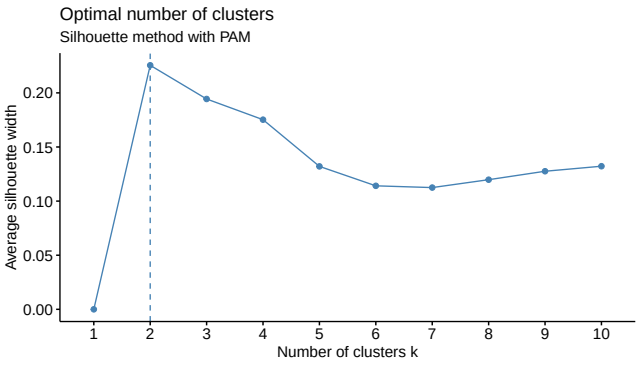


Figure 2.1: Scree plot showing the silhouette method with PAM. It can be observed that the optimal number of clusters suggested by the silhouette method is 2 since the average silhouette width is the highest. This graphic has been constructed using the data set introduced in chapter 3.

- The **Elbow method** is a method whose purpose is to minimize the total intra-cluster variation (or total within-cluster sum of squares). To do so, the variance (within-cluster sum of squared errors) is plotted against the number of clusters. The scree plot is key to choosing how many clusters are retained. It can be seen that the first clusters introduce a lot of variance, i.e information, but at some point, the information provided by the clusters will decrease, making graphic changes in the scree plot. Therefore, the optimal number of clusters is found where the graph changes its form, making something similar to an "*elbow*". That is why this method is known as the "*elbow criterion*". Nevertheless, this method can be ambiguous and not appropriate for all cases.

- The **Gap Statistic** is a technique that compares the total within-cluster variation, W_k, for different values of k, and their expected values under the null reference distribution of the data, W^*_{ki}. The difference between these two values is known as **gap value**, $Gap(k)$. The value that maximizes the gap statistic is an estimate of the optimal number of clusters because it means the clustering structure is very far away from the random uniform distribution of points.

  The null hypothesis assumes that the model has a single cluster, and the alternative hypothesis assumes the model has $k > 1$ components. The chosen distribution is the uniform, because among all unimodal distributions, it is the most likely to produce erroneous clusters by the gap test.

For the observed data and the reference data, which should be generated, the total intra-cluster variation is computed using different values of k, therefore for each k, there are k different clusters generated with the observations. The Gap Statistic for a given k is defined as follows [33]:

$$Gap(k) = \frac{1}{B}\sum_{i=1}^{B} \log(W_{ki}^*) - \log(W_k)$$

where B is the choosen number of reference data sets with random uniform distribution, k is the number of clusters, $W_k = \sum_{i=1}^{k} W(C_{k,i})$ is the total intra-cluster variation for the observed data sets where $C_{k,i}$ is the i-th cluster when k cluster have been chosen, and W_{ki}^* is the total intra-cluster variation for the reference data sets.

Therefore, the number of clusters should be the smallest value of k such that the gap statistic is within one standard deviation of the gap at $k+1$, s_{k+1}, that is

$$Gap(k) \geq Gap(k+1) - s_{k+1}.$$

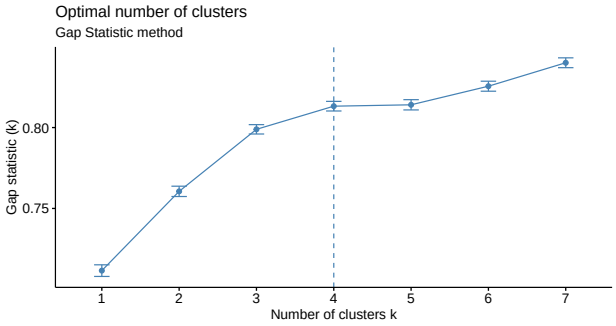


Figure 2.2: Scree plot showing the number of clusters against the Gap Statistic. It can be seen that the optimal number of clusters suggested by the Gap Statistic method is 4. This graphic has been constructed using the data set introduced in chapter 3.

2.5.3 Assessing Clustering Tendency

Before applying any Clustering method, it is important to determine if the data set contains inherent clusters or not. If the answer to this question is positive, a cluster analysis can be applied. However, if it is negative, any Clustering method should be applied because even if the data does not contain any clusters, Clustering algorithms will locate and specify clusters. Therefore, any Clustering method should be applied blindly. Hence, before computing any method, the *clustering tendency* or the *feasibility* of the cluster analysis should be evaluated.

The **Hopkins statistic** is a statistical method to assess the clustering tendency. It measures the probability that a given data set is generated by a uniform data distribution and are thus uniformly randomly distributed. The algorithm can be computed as follows [17]:

Algorithm 4 Hopkins Statistic

1. Let X be the set of n data points of the real data set.
2. Generate a set Y of n uniformly randomly distributed data points with the same variation as the original real data set X.
3. For each point, $y_i \in Y$ compute the distance between its nearest neighbor $x_j \in X$ and the point, and do the same with $x_i \in X$, that is, $w_i = dist(x_i, x_j)$ and $u_i = dist(y_i, x_k)$.
4. Calculate the Hopkins statistic (H) as follows: $H = \dfrac{\sum_{i=1}^{n} u_i}{\sum_{i=1}^{n} w_i + \sum_{i=1}^{n} u_i}$.

If X were uniformly distributed, then $\sum_{i=1}^{n} u_i$ and $\sum_{i=1}^{n} w_i$ would be close to each other, and thus H would be about 0.5. Hence, if the value of Hopkins statistic is close to zero (an acceptable threshold is 0.3), the data set is significantly a clusterable data. In other case, the data set X is uniformly distributed.

2.5.4 Choosing the Best Clustering Algorithms

Before applying any Clustering method, it is necessary to find the one that works better for each case. Therefore, the performance of each Clustering method is evaluated using different metrics [2]. For all the measures let n and m be the total number of rows and columns of the data set respectively:

- **Internal measures** assess the quality of the clusters using inner information without using any external information. These measures are:
  - **Connectivity** indicates the degree of connectedness of the clusters, as determined by the k-nearest neighbors. Let $nn_{i(j)}$ the j-th nearest neighbor of observation i, and let $x_{i,nn_{i(j)}}$ be zero if i and j are in the same cluster and $1/j$ otherwise. For a given cluster assignment $\mathscr{C}$ $= \{C_1, C_2, ..., C_K\}$, the connectivity should be minimized and is defined as:
    $$Conn(\mathscr{C}) = \sum_{i=1}^{n} \sum_{j=1}^{p} x_{i,nn_{i(j)}}$$
    where p is a parameter giving the number of nearest neighbors to use.
  - **Silhouette Coefficient**. It is exactly the same method as explained in section 2.5.2
  - **Dunn Index** is the ratio of the smallest distance between observations not in the same cluster to the largest intra-cluster distance. It should be maximized and can be computed as follows:
    $$D(\mathscr{C}) = \frac{\min\limits_{C_i \neq C_j i \in C_i, j \in C_j} d_{ij}}{\max\limits_{C_m} diam(C_m)}$$
    where $diam(C_m)$ is the maximum distance between observations in the cluster C_m.
- **Stability measures** remove each column, one at a time, before computing the clustering algorithm and compares the clusters, one by one, with the complete cluster result to evaluate the robustness. Let $C_{i,0}$ the original cluster i, $C_{i,l}$ the cluster with column l removed and K the number of clusters. Smaller values correspond to higher consistent cluster results. These measures include:
  - The **Average Proportion of Non-overlap (APN)** measures the average proportion of observations that are not in the same cluster by clustering made with the full data and clustering made with the full data except for one variable. It is computed as:
    $$APN(K) = \frac{1}{mn} \sum_{i=1}^{n} \sum_{l=1}^{m} \left(1 - \frac{|C_{i,0} \cap C_{i,l}|}{|C_{i,0}|}\right).$$
  - The **Average Distance (AD)**, in contrast to the APN, measures the average distance between observations placed in the same cluster by clustering based on the full data and clustering based on the full data except one variable. It is defined as:
    $$AD(K) = \frac{1}{mn} \sum_{i=1}^{n} \sum_{l=1}^{m} \frac{1}{|C_{i,0}||C_{i,l}|} \left[\sum_{i \in C_{i,0}, j \in C_{i,l}} dist(i,j)\right].$$
  - The **Average Distance between Means (ADM)** measures the average distance between cluster centers for observations placed in the same cluster by clustering based on the full data and clustering based on the full data except one variable. The ADM can be computed as follows:
    $$ADM(K) = \frac{1}{mn} \sum_{i=1}^{n} \sum_{l=1}^{m} dist(\bar{x}_{C_{i,0}}, \bar{x}_{C_{i,l}}),$$
    where $\bar{x}_{C_{i,0}}$ is the mean of the observations in the cluster that contains the observation i, when clustering is based on the full data, and $\bar{x}_{C_{i,l}}$ similarly defined.
  - The **Figure of Merit (FOM)** measures the average intra-cluster variance of the removed variable, where the clustering is based on the remaining columns. For the left-out column l:
    $$FOM(l,K) = \sqrt{\frac{1}{n} \sum_{K=1}^{K} \sum_{i \in C_k(l)} dist(x_{i,l}, \bar{x}_{C_{k(l)}})},$$
    $x_{i,l}$ is the value of the i-th observation in th l-th column in cluster $C_k(l)$ and $\bar{x}_{C_{k(l)}}$ is its average.

Chapter 3

Biomechanics' Applications. Stride Profiles.

One of the goals of this end-of-degree project is to apply the previously introduced techniques and concepts to real data. Specifically to use the knowledge to compare stride samples that represent the principal runner stride profiles depending on their running patterns, and create different clusters in which athletes with similar patterns can be grouped together.

This is going to be obtained through a procedure to compute clusters for the different stride profiles represented in the new features constructed from the running set of variables, which is going to be introduced. This allows to classify runners into different groups depending on their running patterns.

The methods used to construct the different stride profiles are the two statistical Unsupervised Learning techniques introduced in the first two chapters of this work: PCA and Clustering.

On the one hand, Principal Component Analysis derives a low dimensional set of features from a larger set of variables trying to lose the least amount of information possible. Since there are 11 variables that measure the running pattern, it is interesting to represent them with fewer features.

On the other, the aim is grouping together athletes with the same running pattern, therefore clusters have to be made in order to classify athletes into one cluster or another depending on their characteristics.

This analysis was performed using R Statistical Software [28]. Principal Component Analysis was performed and K-means, K-medoids and Hierarchical Clustering were obtained using *cluster* [22], *factoextra* [19] and *FactoMineR* [21] R packages.

This project arises from the need of constant renovation and searching of greatness that many sports are demanding since nowadays, sports is a field which undergoes continuous development. High performance athletes look for the newest garments with the best technologies, the best training sessions or the best supplements in the market. Every slight detail counts and, therefore, every aspect of an athlete needs to be rigorously tracked.

For runners, it is essential to learn the ins and outs of running faster. Hence, gathering the most amount of information about their running technique is vital. Otherwise, they would not be able to improve and become the best athletes in their field. In addition, they can work on the way they perform their technique placing the focus on their weakest points aiming to prevent injuries [11].

3.1 Methodology of the Data Set

The biomechanical study has been conducted by Dr. Luis Enrique Roche[1], who has shared this data set in order to illustrate the previously mentioned statistical techniques.

The study population consists of 1976 completely anonymous observations, considering each observation an athlete running at a certain pace. Note, that a single athlete could provide several observations

[1] PhD biomedicine, MSc Sport performance. DPT. DPM. Researcher and professor in San Jorge University. Villanueva de Gallego, Saragossa, Spain.

determined by different running paces. For example, one runner could appear 3 times; the first one running at 10km/h, the second running at 14km/h, and the third one running at 18km/h. These individuals are long and middle distance runners who run at a "comfortable" pace, usually the pace they train and sometimes the pace at which they would run a race.

These observations are gathered from long and middle distance runners, from different genders and levels, including amateur and professional runners. The pace is between $10km/h$ and $22km/h$ and it is always an integer. The collect data has been tracked by different sensors such as 3D motion capture systems and RunScribe pods. 3D motion capture systems utilize 8 infra-red cameras surrounding the athlete, with reflective markers attached to key body segments, thus capturing high resolution data from all angles, whereas RunScribe pod is a sensor that collects real motion data that explains cadence patterns. The study was conducted in the Innova Biomechanics' S.L.[2] studio and every athlete ran in a treadmill.

These variables tracked during the biomechanical study form the columns of the data set, which is given in table form, and the rows of the table are formed by the observations, which are the runners running at a certain pace.

3.2 Background of the Data Set

It is essential to have a basic knowledge of biomechanics to follow the present document. Key concepts such as Biomechanics or some biomechanics variables are described in the glossary at the beginning of this work. Understanding these concepts makes it easier to follow the ideas and conclusions this work provides.

Some of these definitions may be ambiguous and depend on the interpretation the author wants to convey. Since the following study has been tracked with a running foot pod called *RunScribe*, the previous concepts have been extracted from their metrics' definitions [24].

Moreover, since this experiment was conducted by Dr. Luis Enrique Roche and colleagues, the rest of the definitions and complementary information provided in the glossary are based on different works published by his research group. Therefore, if the reader is interested in more detailed explanations, further information and definitions can be found in [5], [6], [12].

Before computing PCA, it is interesting to understand what each variable represents and how they are correlated with each other. There are 11 original variables: *Pace, Step length, Step Rate, Flight Ratio (Flight Time)* (It is indifferent to choose *Contact Ratio* since it is the opposite), *Step Angle, Vertical stiffness, Leg stiffness, Stance Excursion, Pronation Excursion, Impact Gs* and *Braking Gs*.

It should be pointed out that all the variables are numeric and divided by the weight of the individual to make them comparable, except *Leg* and *Vertical Stiffness*, which are dependent on the weight of each individual. This is a limitation of the study and in future research they should be measured divided by the weight. However, these features are two of the most studied variables since they help to describe how, when running, the human body behaves like a simplified spring mass system [1], [23]. Therefore, although having limitations, these features are kept as the information they provide should be taken into account due to its relevance.

To begin with, *Speed* is a particularly interesting variable in this study. It is the only variable that can be controlled by the experiment conductor. As opposed to the rest of the variables measured in this experiment, the runner can run at a desirable pace. It is as simple as setting the treadmill at that speed or controlling it with a GPS watch. However, it is not that simple to set the runner a specific flight time or step length, just to give an example, and succeed.

As stated before, *Speed* is calculated as the product of *Stride Rate* and *Stride Length*, therefore is expected to be highly correlated with them. However, even *Speed* depends on these two variables, they do not behave proportionally when *Speed* increases or decreases. There are runners that are *Step Rate* dominant and others are *Stride Length* dominant. Nevertheless, more experienced runners tend to have higher *Step Rates* than beginners at similar speeds. In the same way, as *Speed* increases, *Contact Time* decreases [30], which implies that *Flight Time* increases.

[2]Business specialized in investigation, innovation, development, transference and education in Biomechanics.

Vertical stiffness increases with *Speed* and *Stride Rates*. Elite runners tend to have higher *Vertical Stiffness* which is associated with a lower oxygen cost. On the contrary, *Leg Stiffness* remains relatively constant with increasing velocity and is not strongly related to the aerobic demand and fatigue [32].

In addition, the values of these variables sometimes depend on the level of the runner. For example, runners with shorter *Stride Lengths* and higher *Step Rates* tend to have lower impact forces and *Impact Gs* at foot strike. More efficient runners tend to have a higher *Flight Ratio* and experienced runners tend to have shorter ground *Contact Time* [24].

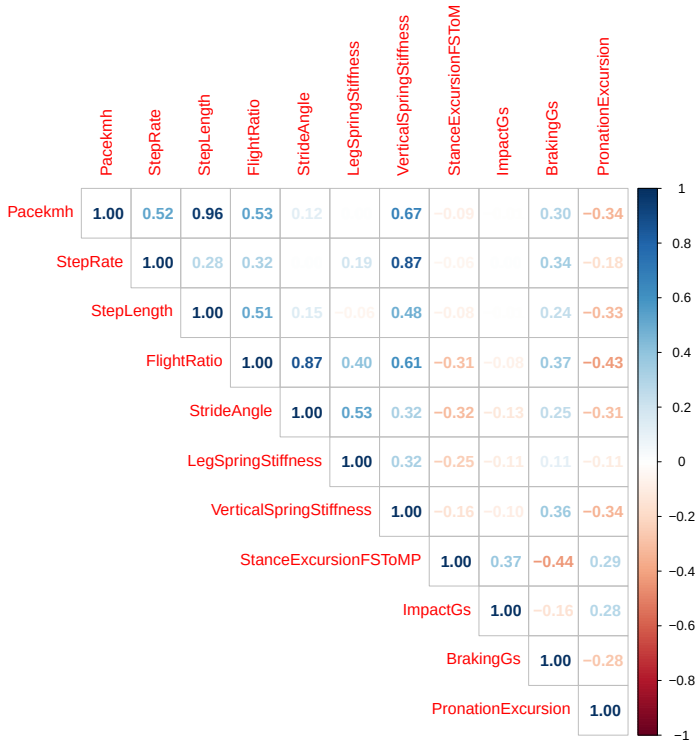


Figure 3.1: Pearson's Correlation matrix of the running variables.

The concordance between what the theory predicts and the collected data can be seen in the Figure 3.1. First, *Speed* seems to be highly correlated with *Flight Time*, *Step Rate*, *Stride Length* and *Vertical Stiffness*. There are other connections between other variables as well. *Flight Ratio* is complementary to *Contact Ratio*, since when the runner is not touching the ground, he is "flying". *Stride Angle* seems to be correlated with *Flight Ratio* and *Leg Stiffness*. This is because *Flight Time* and *Stride Angle* are variables associated with better running efficiency. It seems that optimal execution of *Stride Angle* allows runners to minimize contact time during ground contact, which increases *Flight Time*. Hence, an optimal *Stride Angle* makes the runner "fly" during more time, which is translated into a better running economy [31].

3.3 Principal Component Analysis and Clustering Application

Diverse procedures are going to be applied into the data to obtain different results and analyze what each object represents. Every step is going to be carefully explained and described for the reader to understand the different conclusions.

The first step before applying some of the previously mentioned statistical techniques is the data cleansing. Since the data is measured by sensors that can fail, it should be analyzed to detect possible irregularities or mistakes.

Once the data is clean, the PCA can be applied. This step consists on applying the function *prcomp*, which performs a PCA using the singular value decomposition on the given data matrix and returns the results as an object of class *prcomp*.

```
PSA11var <- prcomp(Datosfinal, center = TRUE,scale. = TRUE)
```

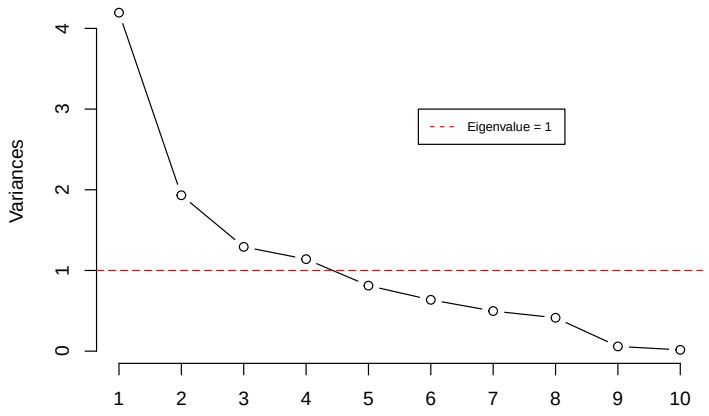
Datosfinal is the data after removing all the values that are corrupt. *Center = TRUE* is a logical value that indicates the variables are shifted to be zero centered. *Scale. = TRUE* is a logical value that indicates the variables are shifted to have unit variance. As stated in the subsection 1.4.1 of the first chapter, in this case it is recommended to standardize the variables since there are no variables more important than others and all of them are measured in different units of measurement.

Once the PCA function is performed, one should determine how many components are to be retained. There are many ways to compute the optimal number of components, three of them are explained in the section 1.4.4. Therefore, to compute the optimal number, more than one technique is going to be applied in order to get an agreement or a consensual answer. First, the function *get_eigenvalue* shows the eigenvalues associated to each Principal Component, the Proportion of Variance and the Cumulative Proportion of Variance. Deciding how many components to retain depends on what is considered an optimal value of the Cumulative Proportion of Variance on the area involved:

```
eig.val2 <- get_eigenvalue(PSA11var)
eig.val2

##         eigenvalue variance.percent cumulative.variance.percent
## Dim.1   4.194986843      38.13624403                    38.13624
## Dim.2   1.931994154      17.56358322                    55.69983
## Dim.3   1.292548637      11.75044215                    67.45027
## Dim.4   1.140261672      10.36601520                    77.81628
## Dim.5   0.811926279       7.38114799                    85.19743
## Dim.6   0.636263630       5.78421481                    90.98165
## Dim.7   0.496990813       4.51809830                    95.49975
## Dim.8   0.413942805       3.76311641                    99.26286
## Dim.9   0.058643755       0.53312505                    99.79599
## Dim.10  0.015895152       0.14450138                    99.94049
## Dim.11  0.006546262       0.05951147                   100.00000
```

Besides the Cumulative Variance Percentage, the Average Eigenvalue Approach is crucial to determine the final number. The function works as follow:

```
screeplot(PSA11var, type = "l", main = "Average Eigenvalue Approach Scree plot")
abline(h = 1, col="red", lty=5)
legend(5.8,3, legend=c("Eigenvalue = 1"), col=c("red"), lty=8, cex=1)
```



As it can be seen in the Figure 3.2, 4 eigenvalues are higher than 1, and therefore according to the Average Eigenvalue Approach criterion, 4 Principal Components should be chosen. Moreover, in the Cumulative Proportion of Variance it is observable that with 4 components, more than 75% of the information was explained, which is a reasonable percentage in biomechanics, since it is an experimental science and the reduction goes from 11 variables to 4. Therefore, from here to the end, 4 Principal Components are going to be considered.

Figure 3.2: Average Eigenvalue Approach Scree Plot.

Once the number of PCs has been decided, it is essential to understand what each component represents.

The different Principal Components can be plotted with the function *fviz_contrib*. This function supports a first argument of different classes such as *prcomp*. In *axes*, the Principal Component to be plotted should be specified and *top* asks about the number of variables that contribute to the Principal Component to be plotted.

```
fviz_contrib(PSA11var, choice = "var", axes = 1, top = 6)
```

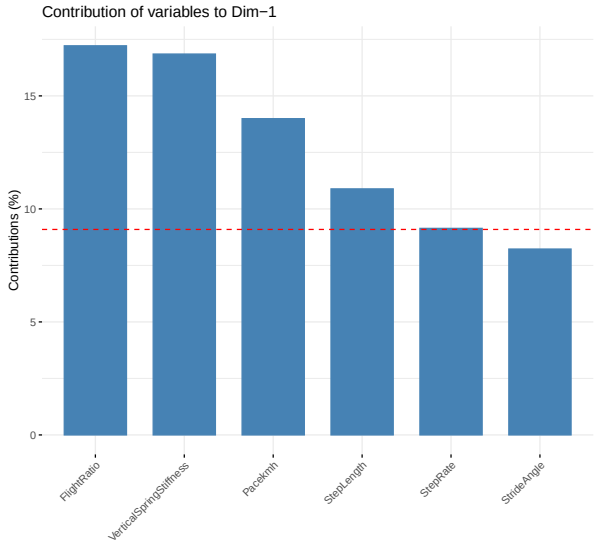


Figure 3.3: Contribution of the variables to the First Principal Component.

The First Principal Component is the component that explains the majority of the variance in the data set, almost a 40% of the total, which is more than half of the information that all the Cumulative Variance Percent explains of the 4 Principal Components chosen. Hence, it is the main Principal Component and the one that provides the most amount of information. The 5 variables that more contribute to the First Principal Component are *Flightratio, VerticalStiffnes, Pace, StepLength* and *StepRate*. It is readily seen that this component is the **Speed** component, since all of these variables are intimately correlated with the *Speed* of the athlete and therefore, it explains the metrics related to the velocity of the runner.

```
fviz_contrib(PSA11var, choice = "var", axes = 2, top = 6)
```

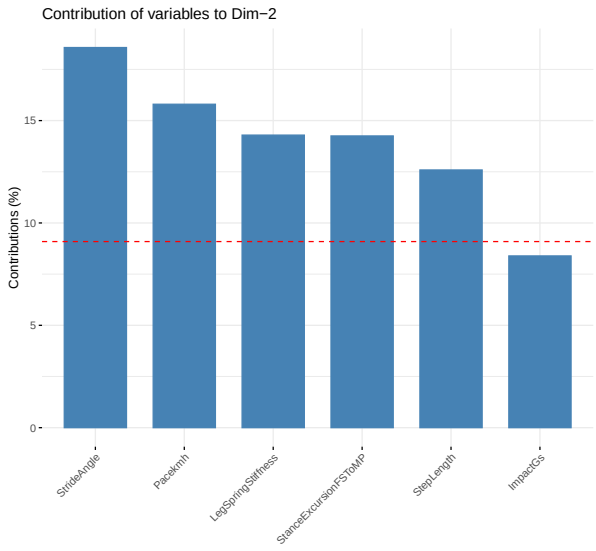


Figure 3.4: Contribution of the variables to the Second Principal Component.

The Second Principal Component explains almost 20% of the total variance, which is half of the information that the First Principal Component provides. The four main contributing variables are *StrideAngle, Pace, LegStiffness* and *StanceExcursion*. This component is the **Spring** component, since 3 of the 4 variables mentioned before (all except pace) are related to the bounce and push of a runner when running. These variables explain the running stage when the runner is pushing again after landing. They give the output angle (SA), the part of the feet where the runner lands (SE) which is crucial to propel at step, and the stiffness of the leg (Kleg) which is the variable that explains how well a runner recycles the energy applied to the ground in each stride and determines if the runner bounces quickly or caves losing speed and force.

```
fviz_contrib(PSA11var, choice = "var", axes = 3, top = 6)
```

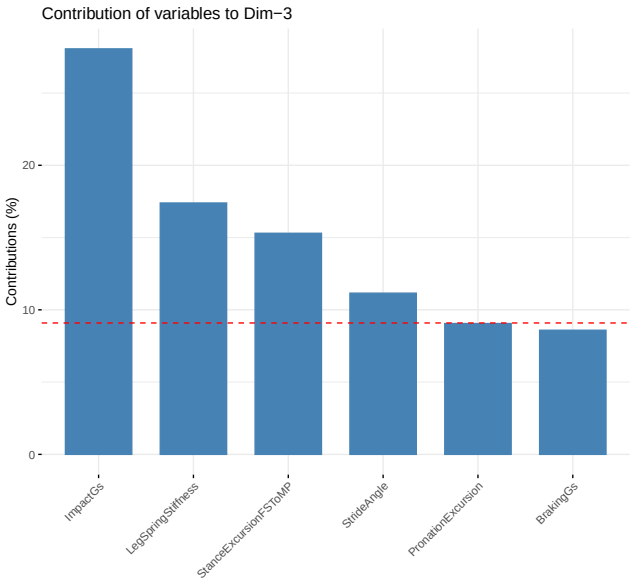


Figure 3.5: Contribution of the variables to the Third Principal Component.

The Third Principal Component is the one that explains the phase between the flight time and the takeoff. This component is the **Shock** component, since it groups the variables that are more related to the landing of the athlete. These variables are *ImpactGs, LegStiffnes, StanceExcursion, PronationExcursion*, and *BrakingGs*. They measure the forces the athlete experiments when landing. For example, the force wasted vertically and horizontally represented by *ImpactGs* and *BrakingGs* respectively. The force absorption by the leg and the taking off is measured by the *LegStiffness*, since the leg works as a string [1], [23]. The *Stance Excursion*, depending on the part of the feet the athlete lands, makes him lose more energy or, on the contrary, it helps to propel him efficiently. The *Pronation Excursion* is the angle of pronation of the athlete at landing which is crucial to absorb forces.

```
fviz_contrib(PSA11var, choice = "var", axes = 4, top = 6)
```

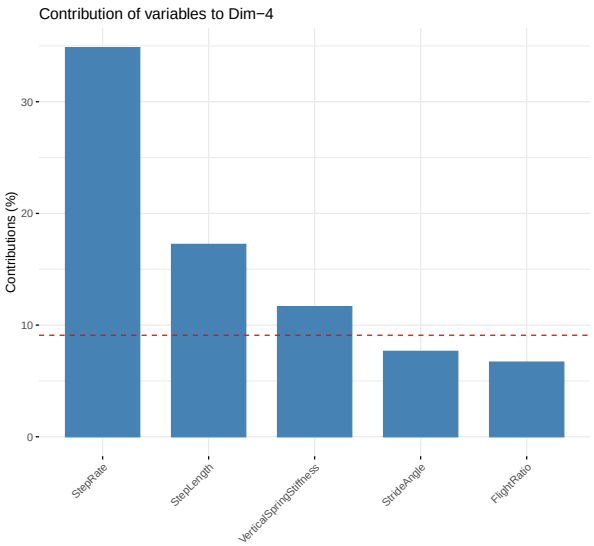


Figure 3.6: Contribution of the variables to the Fourth Principal Component.

The Fourth Principal Component is the most unevenly distributed component. The variable that more contributes is *Step Rate*, followed by *Step Length*. At first, seeing these variables again after they contribute to the First Principal Component might be confusing. However, as explained in section 3.2, since Pace is SR multiplied by SL, two runners might run at the same pace and have different values for these variables. For example, both athletes run at 300 *m/min*, the first one steeping 150 *times/min* with a SL of 2 *m* and the second one steeping 300 *times/min* with a SL of 1 *m*. Therefore, this component explains the case when the pace is faster due to an increase in the SR and not in the SL. That is, if the first runner of the example increases his speed to 400 *m/min*, he will increase his SR and not that much of his SL. For instance, he would step 190 *times/min* with a SL of 2.1 *m*. To summarize, both variables are increased, but Step Rate predominates over Step Length.

Before choosing the number of clusters and the algorithm to use, it is important to determine whether the data set contains meaningful clusters or not because if any Clustering method is applied blindly, no matter if there are no meaningful clusters in the data, every Clustering method will return clusters because it is what they are supposed to do, as explained in section 2.5.3. Therefore, before computing

any clustering method, one should assess the cluster tendency or the feasibility of the clustering analysis. Therefore, the *Hopkins* statistics can be computed as follows:

```
df <- scale(Datosfinal)
hopkins(df, n = nrow(df)-1)

## [1] 0.16155654
```

The data has been scaled to have standard deviation of one and mean of zero since the variables are measured in different scales and the variables should have the same weight, see section 2.5.1. The value of Hopkins statistic is 0.16, which is far below the threshold 0.5. Hence, the data set can be assumed to not be uniformly distributed, i.e., it contains meaningful clusters and clustering methods can be applied.

Therefore, the next step is to decide which algorithm should be applied to compute how many clusters are in the data. In section 2.5.4 some measures to compare clustering algorithms are shown.

The function *clValid* measures internal and stability measures. For this function, the algorithms to compare the measures, *c("hierarchical","kmeans","pam")*, should be established, the number of clusters that could be chosen, *nClust = 2:6*, and the type of validation, *= "internal"* or *= "stability"*, depending on the measure wanted to perform. The function can be applied as follows:

```
clmethods <- c("hierarchical","kmeans","pam")
intern<-clValid(df, nClust = 2:6, clMethods=clmethods, validation = "internal")
optimalScores(intern)

##              Score        Method       Clusters
## Connectivity 2.9290  hierarchical         2
## Dunn         0.3230  hierarchical         2
## Silhouette   0.5026  hierarchical         2
```

Hierarchical Clustering with 2 clusters performs the best in each case with the internal measures.

```
stab<-clValid(df, nClust = 2:6, clMethods = clmethods, validation = "stability")
optimalScores(stab)

##           Score        Method      Clusters
## APN   0.006991741  hierarchical        3
## AD    3.418018733       pam            6
## ADM   0.041655792  hierarchical        2
## FOM   0.821602199       kmeans         6
```

For the APN and ADM measures, Hierarchical Clustering with 3 and 2 clusters respectively gives the best score. For the other measures, K-means and PAM with 6 clusters has the best score. Therefore, there is no agreement on which is the best method to follow.

Since the decision about what algorithm to choose is not decided by an objective way by using a formalized procedure, it should be chosen assessing other criteria.

Clustering methods provide a graphic representation of the clusters, which is usually helpful in the data analysis of identifying the respective groups. Since the dendrogram of the Hierarchical Clustering does not provide information about the variables, the chosen method should be K-means or K-medoids. These methods provide two-dimensional scatterplots of the data. However, since there are 11 variables, the data could be plotted in $\binom{11}{2} = 55$ different graphics. We are interested in the two-dimensional scatterplot that captures the most amount of information from the data, therefore the best procedure is to perform PCA and plot the first two Principal Component score vectors [13]. Moreover, since the interpretation of the two PCs is known, the explanation about what represents each cluster depending on the position they take up can be elucidated.

The Hierarchical Clustering procedure applied to the data can be found in the the appendix B.1.

The first step before computing K-means, (K-medoids procedure is similar and can be found in the appendix B.2) is determining how many clusters are optimal. Some methods explained in section 2.5.2 such as the Elbow method and the Gap Statistic are applied as follows:

```
fviz_nbclust(df, kmeans, method = "wss", k.max = 8) +
  geom_vline(xintercept = 3, linetype = 2)+labs(subtitle = "Elbow method")
```

```
gap_stat <-(clusGap(df, FUN = kmeans, nstart = 25, K.max = 7, B = 50))
print(gap_stat, method = "Tibs2001SEmax")+labs(subtitle="Gap Statistic method")
fviz_gap_stat(gap_stat, maxSE = list(method = "Tibs2001SEmax", SE.factor = 1))+
    labs(subtitle = "Gap Statistic method")
```

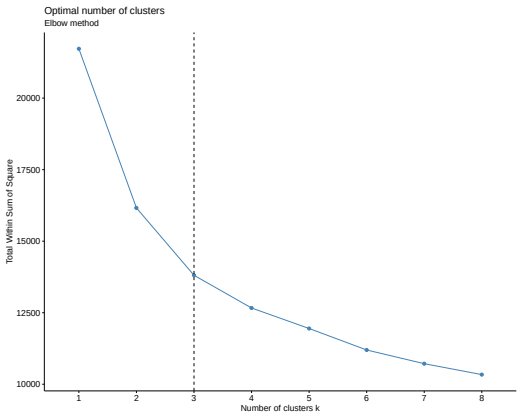


Figure 3.7: Elbow method to determine the optimal number of clusters.

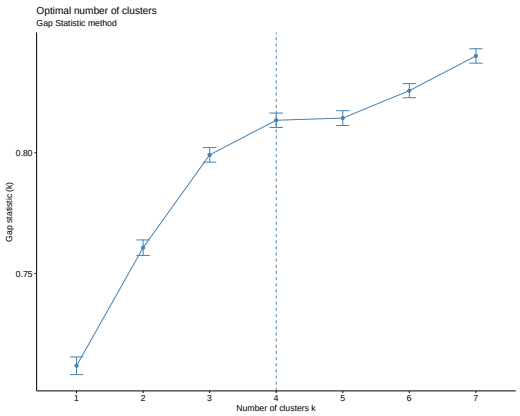


Figure 3.8: Gap Statistic method to determine the optimal number of cluster.

These two methods suggest that 3 and 4 clusters are the optimal number of clusters in K-means. However, in appendix B.3 it can be seen that using the function *NbClust* providing 30 indices for determining the number of clusters suggests that 3 clusters is the best option.

To get that big doubt out, the two graphics are going to be plotted. Since there is no single or correct answer, the choice boils down to select the clusterization that gives a better graphical representation.

```
km.res4 <- eclust(df, "kmeans", k = 4, nstart = 25, graph = FALSE)
fviz_cluster(km.res4, geom = "point", ellipse.type = "norm",
             palette = "jco", ggtheme = theme_minimal())
km.res <- eclust(df, "kmeans", k = 3, nstart = 25, graph = FALSE)
fviz_cluster(km.res, geom = "point", ellipse.type = "norm",
             palette = "jco", ggtheme = theme_minimal())
```

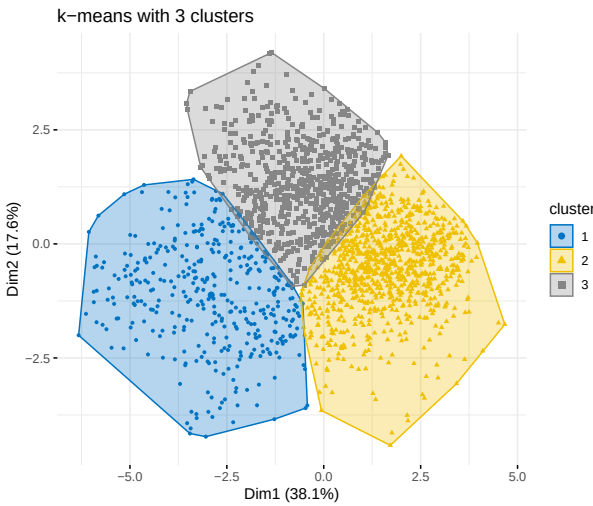


Figure 3.9: Representation of the 3 clusters applying k-means.

Figure 3.10: Representation of the 4 clusters applying k-means.

3.4 Analysis of the Data: Results and Conclusions

Since the beginning of the work, it has been pointed out that some aspects of these statistical methods are subjective. In the previous section, the different methods available to choose the number of clusters were not decisive and therefore, other considerations should determine how many clusters are optimal. In this case, graphics could be used as a tool to help determine the final number of clusters.

Unlike in figure 3.10, in which it is difficult to ensure where each cluster has its boundary, in Figure 3.9 there are 3 clusters with a minimal overlapping. Many observations seem to be erroneously grouped into their clusters since they are out of their cluster border. Therefore, it is preferable to choose the 3-cluster graphic with no observations overlaying each other.

Finally, the interpretation of what each cluster in the graphic represents deserves some attention. Other scatterplots that represent the data into different PCs score vectors can be found in appendix C. In this case, the horizontal axis represents the First Principal Component (*Speed* component) and the vertical axis represents the Second Principal Component (*Spring* component). Writing the PCs as linear combinations of the variables helps to analyze the influence of each variable in each PC:

$$PC1 = -0.37 \times Pace - 0.30 \times SR - 0.33 \times SL - 0.41 \times FR - 0.29 \times SA - 0.19 \times Kleg - 0.41 \times Kvert + 0.20 \times SE + 0.1 \times IGs - 0.27 \times BGs + 0.28 \times PE.$$

$$PC2 = -0.40 \times Pace - 0.29 \times SR - 0.35 \times SL + 0.16 \times FR + 0.43 \times SA + 0.37 \times Kleg - 0.17 \times Kvert - 0.38 \times SE - 0.29 \times IGs + 0.1 \times BGs - 0.1 \times PE.$$

The positive or negative sign of the coefficients indicates the direction that a given variable in that PC is going to take on a single dimension vector. Loadings close to 0 do not play an important role in explaining the variability along the PC and therefore, bigger loadings have greater impacts in that Principal Component. Finally, the scores of that PC indicate that individuals with negative PC tend to have greater values of the negative loadings, and lower values of the positive, whereas individuals with greater values of the positive loadings, and lower values of the negative, tend to have positive PC.

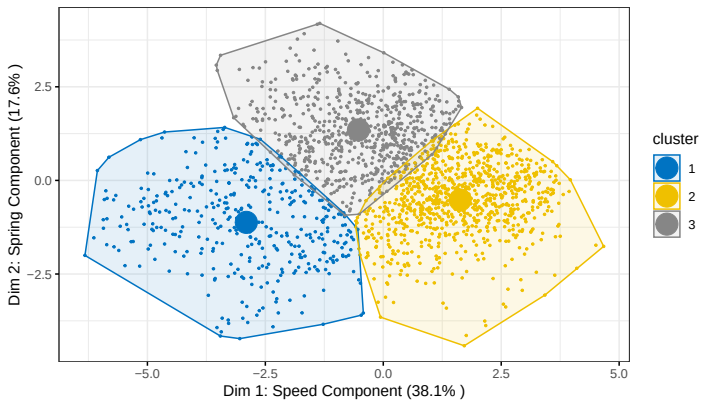


Figure 3.11: The graphical representation could be considered suitable since the observations only belong to one cluster and the overlapping is minimal. The horizontal axis is the *Speed* component that represents almost a 40% of the total, and the vertical axis is the *Spring* component that represents almost a 20% of the total. There are 3 clusters that represent 3 different stride profiles.

- **The blue cluster**: It is the cluster associated with the fastest runners since the values in the horizontal axis (*Speed* Component) are the most negative (The speed loadings in the PC1 are negative). The same happens in the vertical axis (*Spring* Component), therefore the individuals of this cluster tend to have with high Kvert (smaller values mean higher stiffness), lower SA and bigger SE. Hence, they are very reactive with a mainly horizontal displacement .

- **The gray cluster**: It is the cluster of the runners with average pace (The value of the PC1 is almost 0) and highest values in the Spring Component. Therefore, they have bigger SA and smaller Kver and SE, which means the displacement is more vertical than other groups and the Leg Stiffness is lower (they are not as reactive as the other groups).

- **The yellow cluster**: It is the cluster of the slowest athletes, since they have the most positive values of the Speed Component and average Spring Component (the values of the PC2 are around zero).

Bibliography

[1] R. Blickhan, "The spring-mass model for running and hopping," *Journal of Biomechanics*, vol. 22, no. 11, pp. 1217–1227, 1989, ISSN: 0021-9290. DOI: https://doi.org/10.1016/0021-9290(89)90224-8. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0021929089902248.

[2] G. Brock, V. Pihur, S. Datta, and S. Datta, "Clvalid : An r package for cluster validation," *Journal of statistical software*, vol. 25, Mar. 2008. DOI: 10.18637/jss.v025.i04.

[3] R. Cattell, "The description of personality: Basic traits resolved into clusters.," *The Journal of Abnormal and Social Psychology*, vol. 38, no. 4, pp. 476–506, Oct. 1943. DOI: 10.1037/h0054116.

[4] H. Driver and A. Kroeber, "Quantitative expression of cultural relationships," *University of California Publications in American Archaeology and Ethnology*, vol. 31, no. 4, pp. 211–256, 1932. [Online]. Available: http://digitalassets.lib.berkeley.edu/anthpubs/ucb/text/ucp031-005.pdf.

[5] C. T. Farley and O. González, "Leg stiffness and stride frequency in human running," *Journal of Biomechanics*, vol. 29, no. 2, pp. 181–186, 1996. DOI: 10.1016/0021-9290(95)00029-1.

[6] F. García-Pinillos, A. Cartón-Llorente, D. Jaén-Carrillo, *et al.*, "Does fatigue alter step characteristics and stiffness during running?" *Gait & amp; Posture*, vol. 76, pp. 259–263, 2020. DOI: 10.1016/j.gaitpost.2019.12.018.

[7] T. Hastie, J. Friedman, and R. Tisbshirani, *The elements of Statistical Learning: Data Mining, Inference, and prediction*, Second. Springer, 2008.

[8] H. Hatze, *Letter: The meaning of the term "biomechanics"*. J Biomech, Mar. 1974.

[9] J. G. Hay, *The biomechanics of sports techniques*. Prentice Hall, 1973.

[10] H. Hotelling, "Analysis of a complex of statistical variables into principal components.," *Journal of educational psychology*, vol. 24, no. 6, pp. 417–441, 1933. DOI: https://doi.org/10.1037/h0071325.

[11] Á. Infante Ojeda, Y. Flores Labrada, and D. A. Fuentes Varona, "Los fundamentos técnicos de las carreras de fondo y medio fondo (revisión)," *Revista científica Olimpia*, vol. 14, no. 42, pp. 109–118, Mar. 2017. [Online]. Available: https://revistas.udg.co.cu/index.php/olimpia/article/view/1278.

[12] D. Jaén-Carrillo, L. E. Roche-Seruendo, L. Felton, A. Cartón-Llorente, and F. García-Pinillos, "Stiffness in running: A narrative integrative review," *Strength & amp; Conditioning Journal*, vol. 43, no. 2, pp. 104–115, 2020. DOI: 10.1519/ssc.0000000000000593.

[13] G. James, D. Witten, T. J. Hastie, and R. J. Tibshirani, *An introduction to statistical learning: With applications in R*, Second. Springer, 2021.

[14] G. F. Jenks, "The data model concept in statistical mapping," in *International Yearbook of Cartography 7*, 1967, pp. 186–190.

[15] I. T. Jolliffe, "Principal component analysis: A beginner's guide - i. introduction and application," *Weather*, vol. 45, no. 10, pp. 375–382, 1990. DOI: 10.1002/j.1477-8696.1990.tb05558.x.

[16] I. T. Jolliffe and J. Cadima, "Principal component analysis: A review and recent developments," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, 2016. DOI: 10.1098/rsta.2015.0202.

[17] A. Kassambara, *Practical guide to cluster analysis in R: Unsupervised machine learning*. STHDA, 2017.

[18] A. Kassambara, *Practical guide to principal component methods in R*. STHDA, 2017.

[19] A. Kassambara and F. Mundt, *Factoextra: Extract and visualize the results of multivariate data analyses*, R package version 1.0.7, 2020. [Online]. Available: https://CRAN.R-project.org/package=factoextra.

[20] L. Kaufman and P. J. Rousseeuw, "Partitioning around medoids (program pam)," in *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, Inc., 1990, pp. 68–125. DOI: 10.1002/9780470316801.ch2. [Online]. Available: http://dx.doi.org/10.1002/9780470316801.ch2.

[21] S. Lê, J. Josse, and F. Husson, "FactoMineR: A package for multivariate analysis," *Journal of Statistical Software*, vol. 25, no. 1, pp. 1–18, 2008. DOI: 10.18637/jss.v025.i01.

[22] M. Maechler, P. Rousseeuw, A. Struyf, M. Hubert, and K. Hornik, *Cluster: Cluster analysis basics and extensions*, R package version 2.1.2 — For new features, see the 'Changelog' file (in the package source), 2021. [Online]. Available: https://CRAN.R-project.org/package=cluster.

[23] T. A. McMahon and G. Cheng, "The mechanics of running: How does stiffness couple with speed?" *Journal of biomechanics*, vol. 23 Suppl 1, pp. 65–78, 1990.

[24] *Metrics*. [Online]. Available: https://runscribe.com/metrics/ (visited on 06/14/2022).

[25] S. Nanjundan, S. Sankaran, C. R. Arjun, and G. P. Anand, *Identifying the number of clusters for k-means: A hypersphere density based approach*, 2019. DOI: 10.48550/ARXIV.1912.00643. [Online]. Available: https://arxiv.org/abs/1912.00643.

[26] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901. DOI: 10.1080/14786440109462720. eprint: https://doi.org/10.1080/14786440109462720. [Online]. Available: https://doi.org/10.1080/14786440109462720.

[27] D. Peña, *Análisis de Datos Multivariantes*. McGraw-Hill, 2002.

[28] R Core Team, *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, 2020. [Online]. Available: https://www.R-project.org/.

[29] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987, ISSN: 0377-0427. DOI: https://doi.org/10.1016/0377-0427(87)90125-7. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0377042787901257.

[30] C. J. de Ruiter, B. van Oeveren, A. Francke, P. Zijlstra, and J. H. van Dieen, "Running speed can be predicted from foot contact time during outdoor over ground running," *PLOS ONE*, vol. 11, no. 9, 2016. DOI: 10.1371/journal.pone.0163023.

[31] J. Santos-Concejero, N. Tam, C. Granados, *et al.*, "Stride angle as a novel indicator of running economy in well-trained runners," *Journal of Strength and Conditioning Research*, vol. 28, no. 7, pp. 1889–1895, 2014. DOI: 10.1519/jsc.0000000000000325.

[32] A. Struzik, K. Karamanidis, A. Lorimer, J. W. Keogh, and J. Gajewski, "Application of leg, vertical, and joint stiffness in running performance: A literature overview," *Applied Bionics and Biomechanics*, vol. 2021, pp. 1–25, 2021. DOI: 10.1155/2021/9914278.

[33] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a data set via the gap statistic," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63, no. 2, pp. 411–423, 2001. DOI: 10.1111/1467-9868.00293.

[34] R. C. Tryon, *Cluster analysis; Correlation Profile and orthometric (factor) analysis for the isolation of unities in mind and personality*. Edwards Brother, Inc., lithoprinters and Publishers, 1939.

[35] S. Valle, W. Li, and S. J. Qin, "Selection of the number of principal components: The variance of the reconstruction error criterion with a comparison to other methods," *Industrial & amp; Engineering Chemistry Research*, vol. 38, no. 11, pp. 4389–4401, 1999. DOI: 10.1021/ie990110i.

[36] J. Zubin, "A technique for measuring like-mindedness.," *The Journal of Abnormal and Social Psychology*, vol. 33, no. 4, pp. 508–516, 1938. DOI: 10.1037/h0055441.ISSN0096-851X.

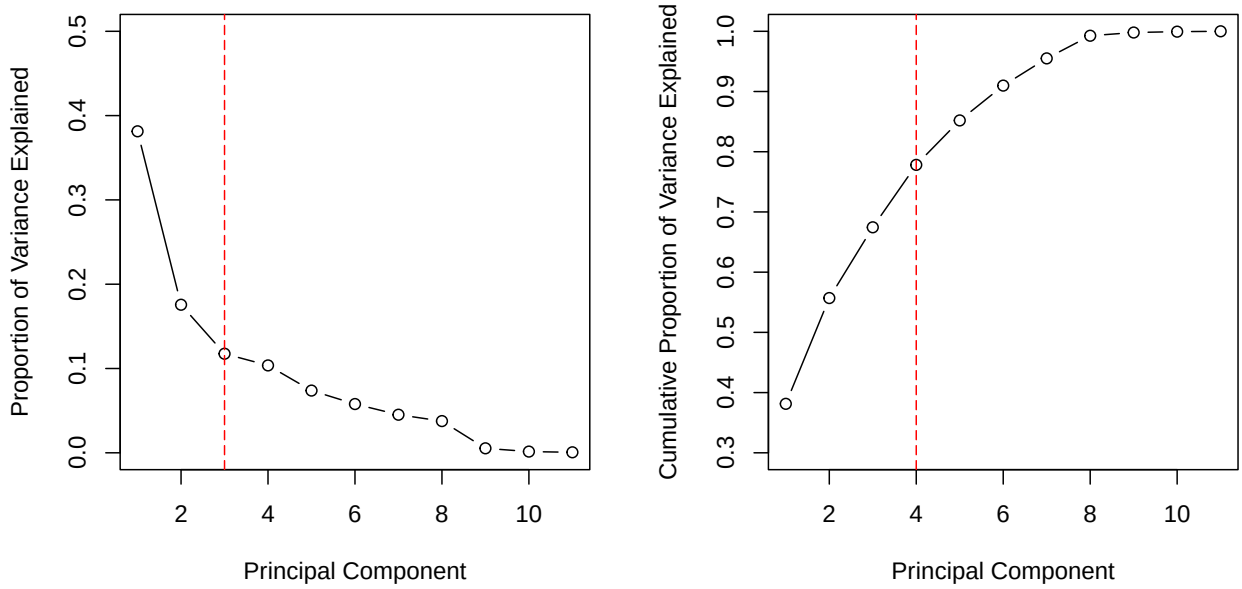
Annexes

Appendix A

Choosing the Number of Components. Graphical Example

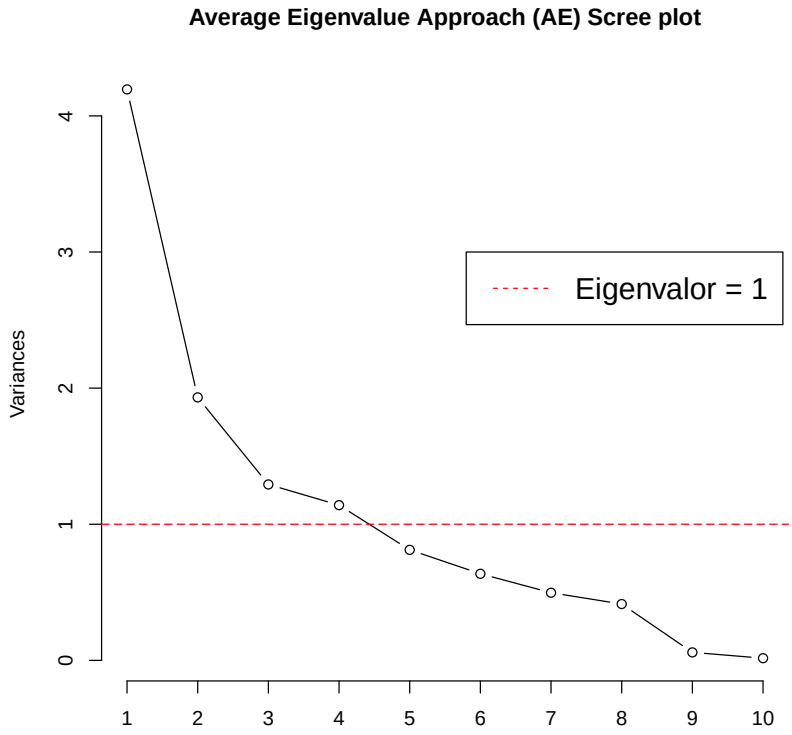
Elbow method based on the Proportion of Variance Explained or Cumulative Proportion of Variance Explained

```
pr.var <- PSA11var$sdev^2
pve <- pr.var / sum(pr.var)
par(mfrow = c(1, 2))
plot(pve , xlab = "Principal Component", ylab = "Proportion of Variance
    Explained", ylim = c(0, 0.5), type = "b")
abline(v = 3, col="red", lty=5)
plot(cumsum(pve), xlab = "Principal Component", ylab = "Cumulative Proportion of
     Variance Explained", ylim = c(0.3, 1), type = "b")
abline(v = 4, col="red", lty=5)
```



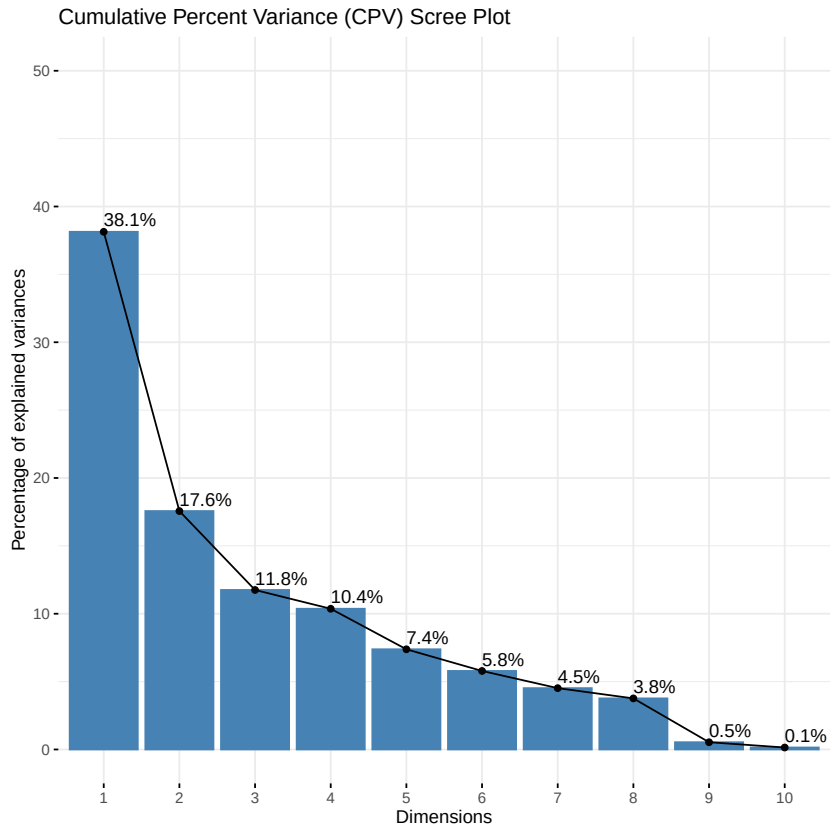
Average Eigenvalue Approach (AE)

```
screeplot(PSA11var, type = "l", main = "Average Eigenvalue Approach (AE) Scree
    plot")
abline(h = 1, col="red", lty=5)
legend(5.8,3, legend=c("Eigenvalor = 1"), col=c("red"), lty=8, cex=1.5)
```



Cumulative Percent Variance (CPV)

```
fviz_eig(PSA11var, main = "Cumulative Percent Variance (CPV) Scree Plot",
    addlabels = TRUE, ylim = c(0, 50))
```



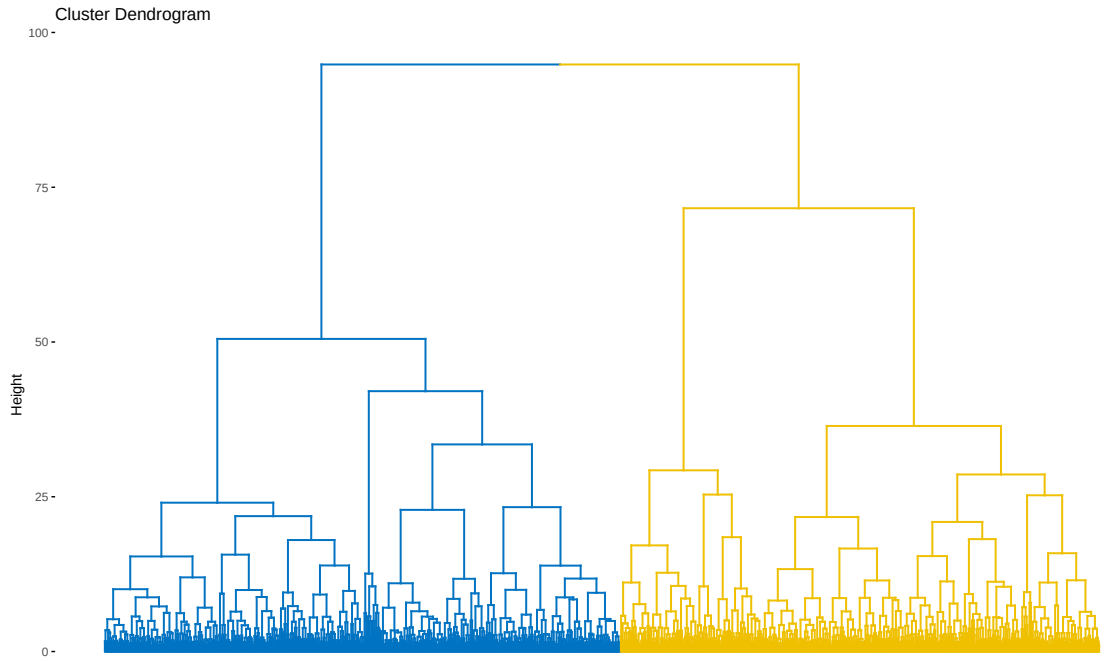
Appendix B

Hierarchical, PAM and K-means Clustering

B.1 Hierarchical Clustering Procedure

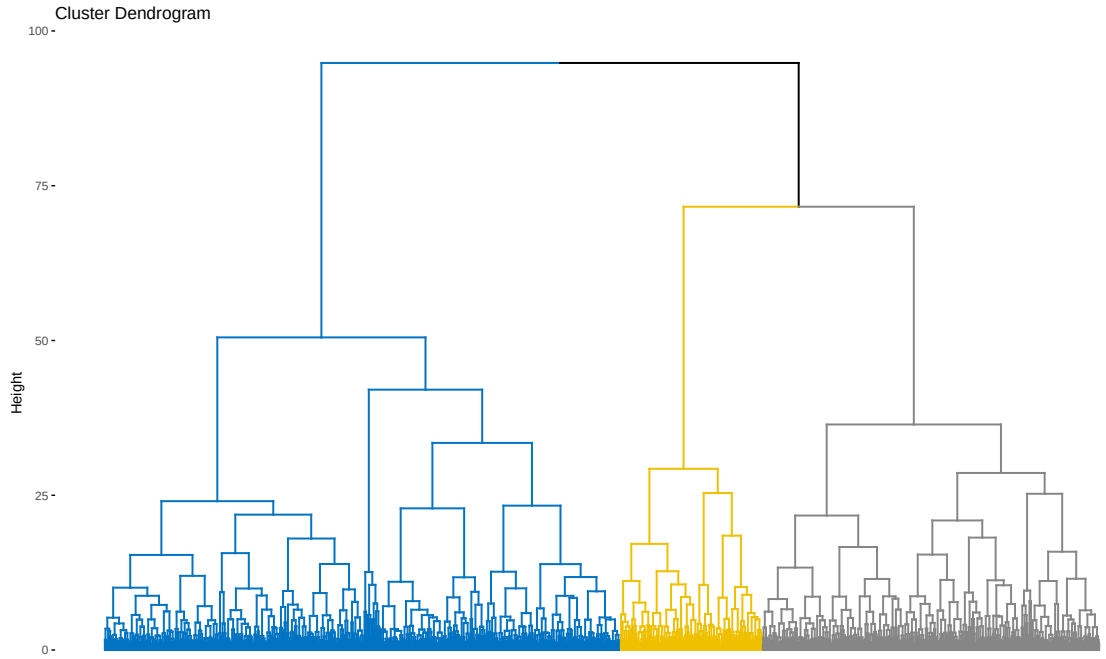
Dendrogram of the Hierarchical Clustering for 2 clusters

```
hc.res <- eclust(df, "hclust", k = 2, hc_metric = "euclidean",
hc_method = "ward.D2", graph = FALSE)
fviz_dend(hc.res, show_labels = FALSE,
palette = "jco", as.ggplot = TRUE)
```



Dendrogram of the Hierarchical Clustering for 3 clusters

```
hc.res1 <- eclust(df, "hclust", k = 3, hc_metric = "euclidean",
hc_method = "ward.D2", graph = FALSE)
# Visualize dendrograms
fviz_dend(hc.res1, show_labels = FALSE,
palette = "jco", as.ggplot = TRUE)
```



Copehenic test to check if the Hierarchical Clustering is a good representation of the data

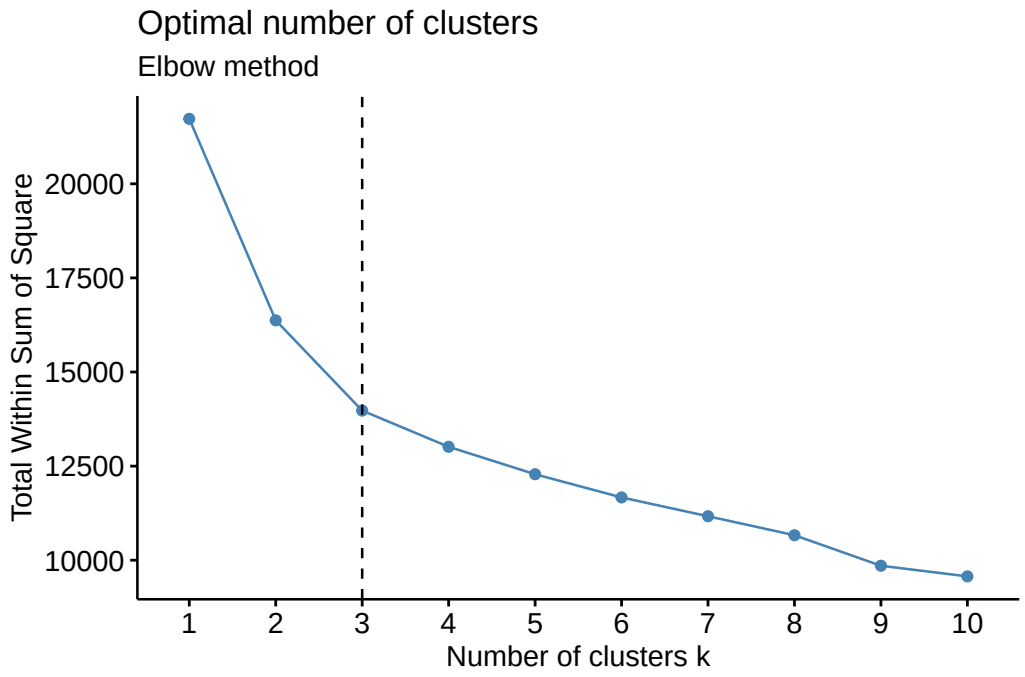
```
res.dist<-dist(df, method="euclidean")

res.coph<-cophenetic(hc.res)
cor(res.dist,res.coph)

## [1] 0.4456151
```

Therefore, since the value is < 0.5, it is not a good representation of the data.

B.2 K-medoids Clustering Procedure

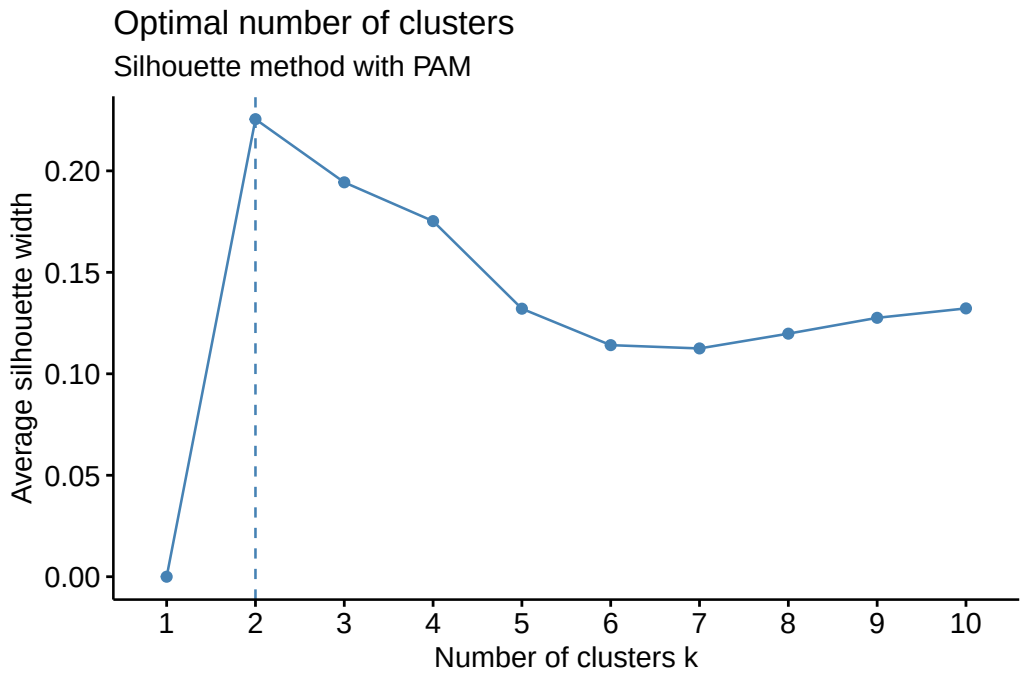
Elbow method with K-medoids to determine the number of clusters to choose

```
fviz_nbclust(df, cluster::pam, method = "wss") +
  geom_vline(xintercept = 3, linetype = 2)+
  labs(subtitle = "Elbow method")
```



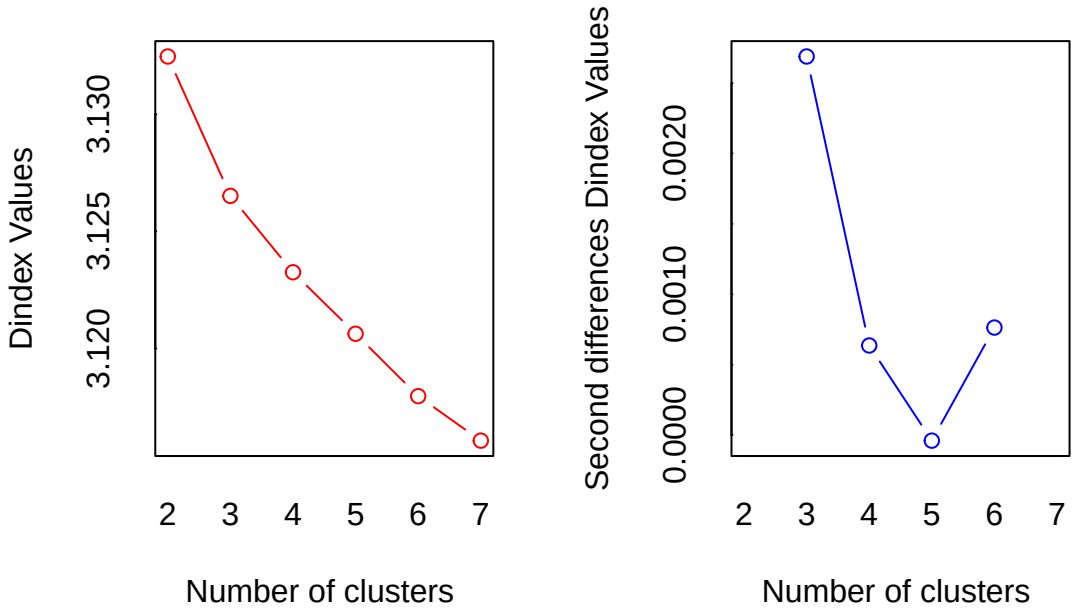
Silhouette method with K-medoids to determine the number of clusters to choose

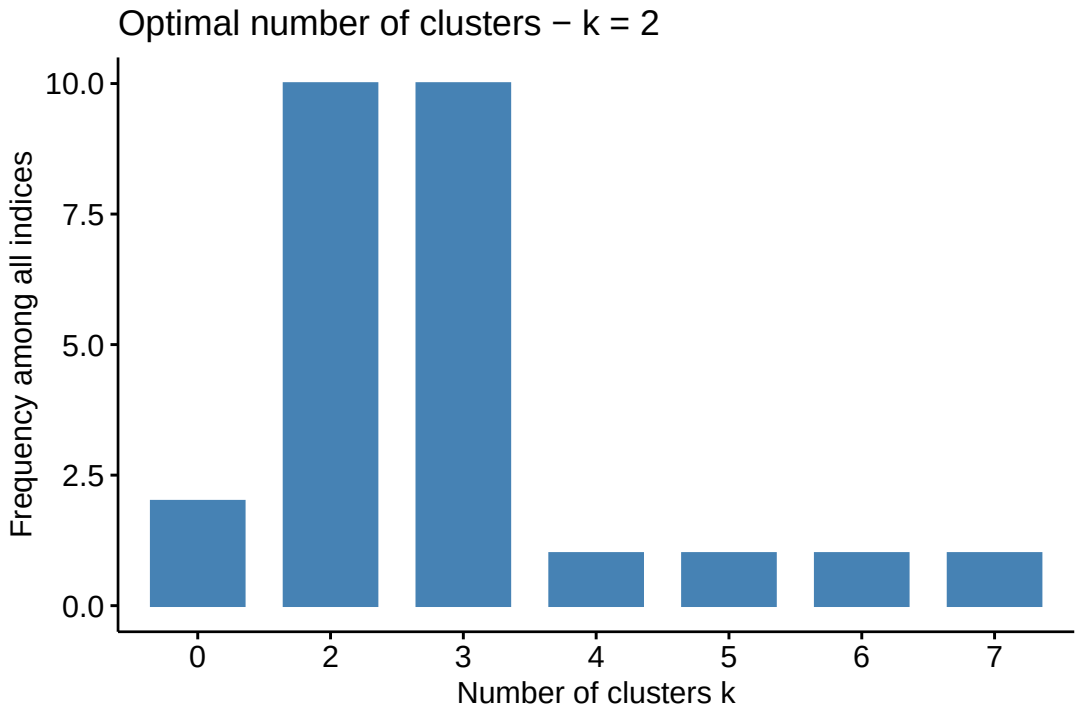
```
fviz_nbclust(df, cluster::pam, method = "silhouette")+
  labs(subtitle = "Silhouette method with PAM")
```



Nb clust with K-medoids to determine the number of clusters to choose

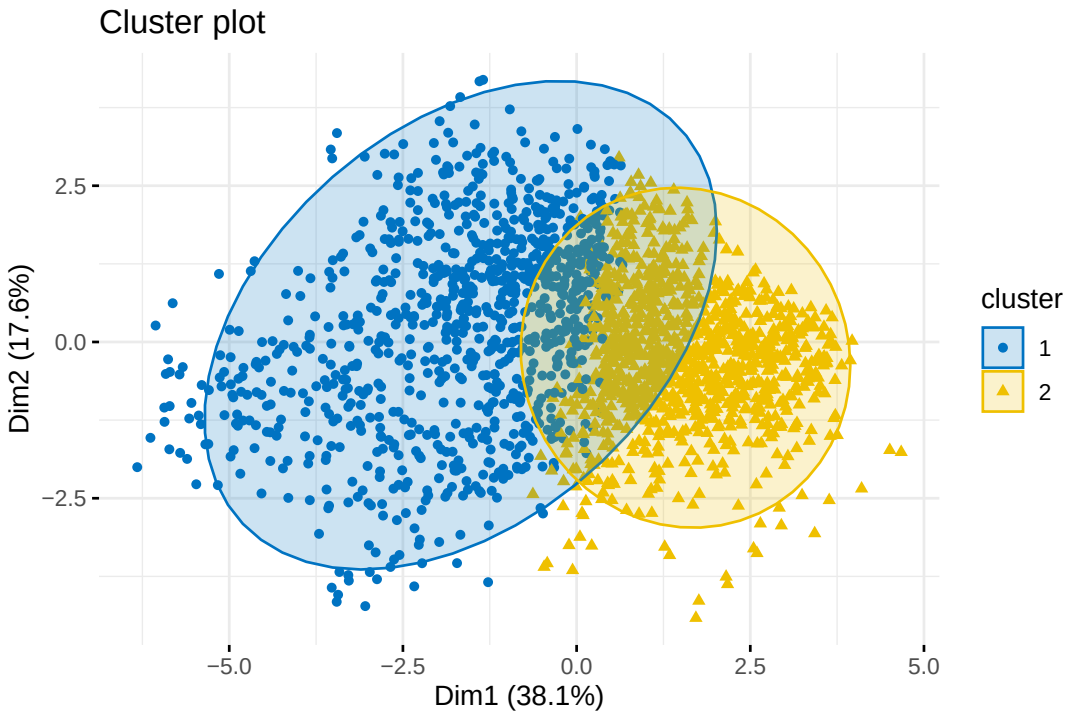
```
nb <- NbClust(df, distance = "euclidean", min.nc = 2,
              max.nc = 7, method = "centroid")
fviz_nbclust(nb)

*** : The Hubert index is a graphical method of determining the number of
    clusters.
                In the plot of Hubert index, we seek a significant knee that
    corresponds to a
                significant increase of the value of the measure i.e the
    significant peak in Hubert
                index second differences plot.

*** : The D index is a graphical method of determining the number of clusters.
                In the plot of D index, we seek a significant knee (the
    significant peak in Dindex
                second differences plot) that corresponds to a significant
    increase of the value of
                the measure.

*******************************************************************
* Among all indices:
* 10 proposed 2 as the best number of clusters
* 10 proposed 3 as the best number of clusters
* 1 proposed 4 as the best number of clusters
* 1 proposed 5 as the best number of clusters
* 1 proposed 6 as the best number of clusters
* 1 proposed 7 as the best number of clusters

                   ***** Conclusion *****

* According to the majority rule, the best number of clusters is  2


*******************************************************************
```





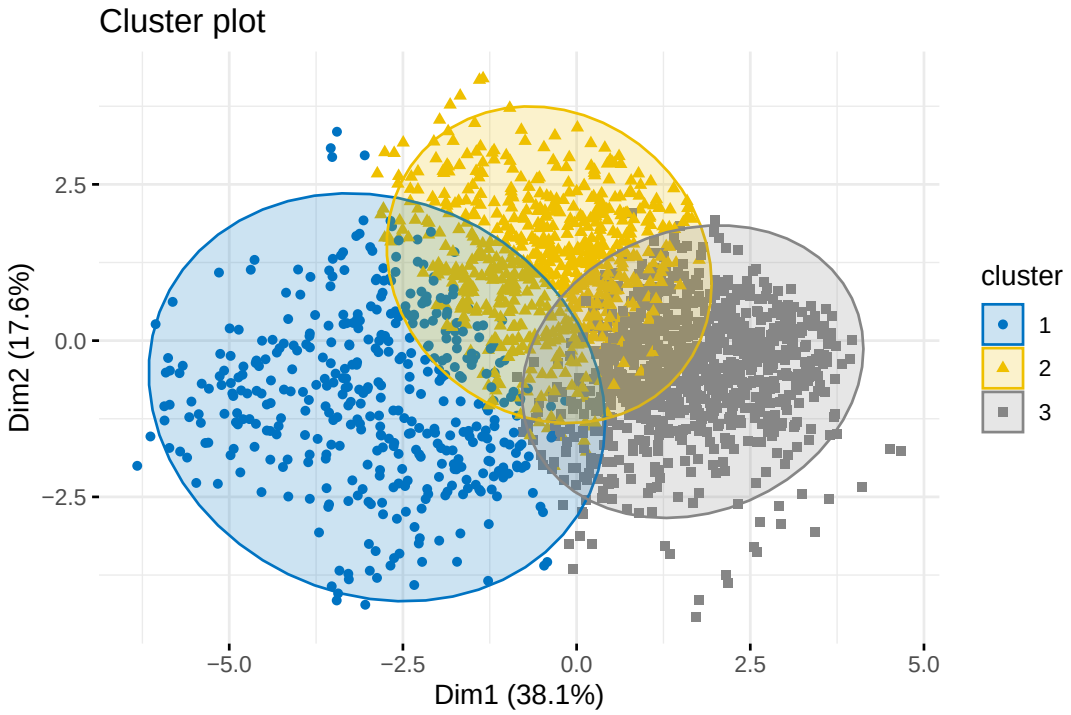
K-medoids Clustering with 2 clusters

```
km.res1 <- eclust(df, "pam", k = 2, nstart = 25, graph = FALSE)
fviz_cluster(km.res1, geom = "point", ellipse.type = "norm",
             palette = "jco", ggtheme = theme_minimal())
```



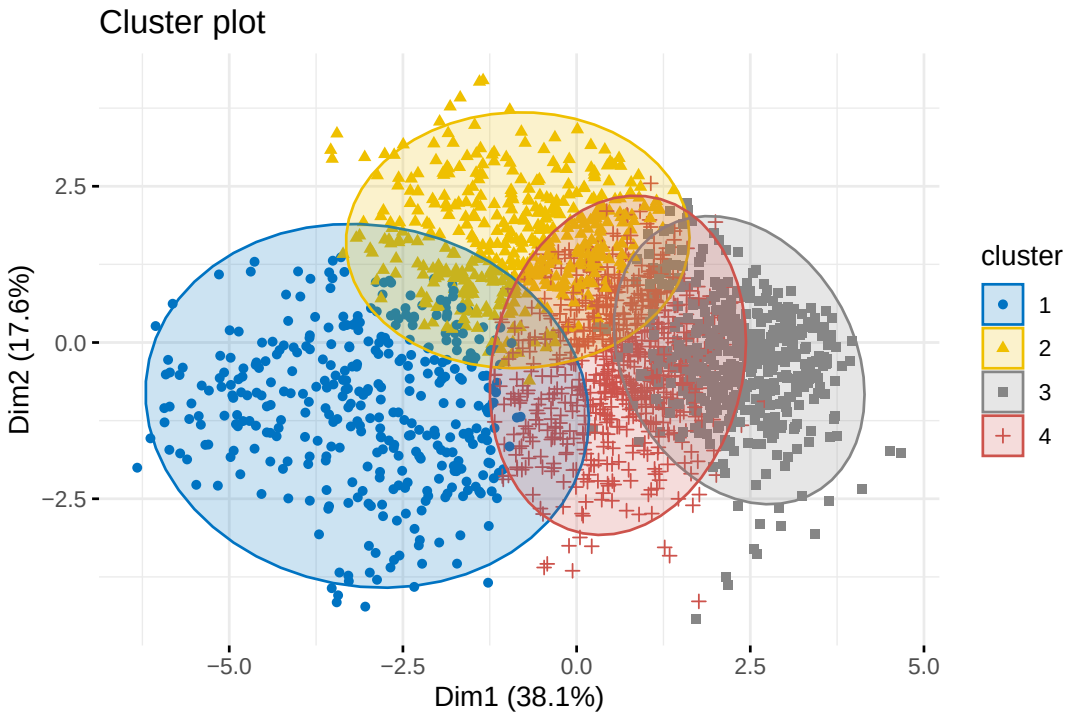
K-medoids Clustering with 3 clusters

```
km.res1 <- eclust(df, "pam", k = 3, nstart = 25, graph = FALSE)
fviz_cluster(km.res1, geom = "point", ellipse.type = "norm",
             palette = "jco", ggtheme = theme_minimal())
```



K-medoids Clustering with 4 clusters

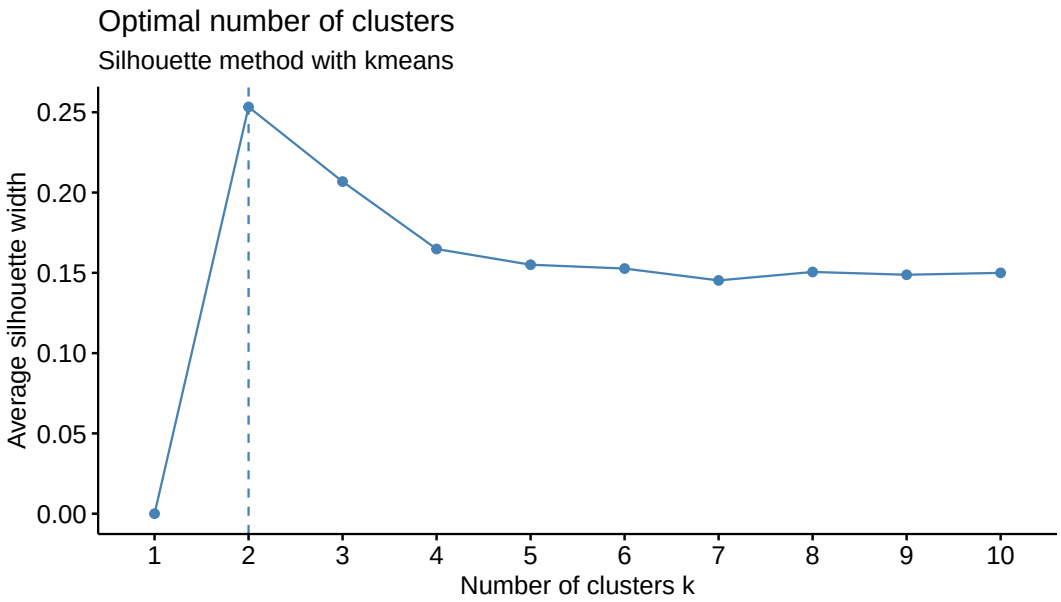
```
km.res1 <- eclust(df, "pam", k = 4, nstart = 25, graph = FALSE)
fviz_cluster(km.res1, geom = "point", ellipse.type = "norm",
             palette = "jco", ggtheme = theme_minimal())
```



B.3 K-Means Clustering Complementary Procedure

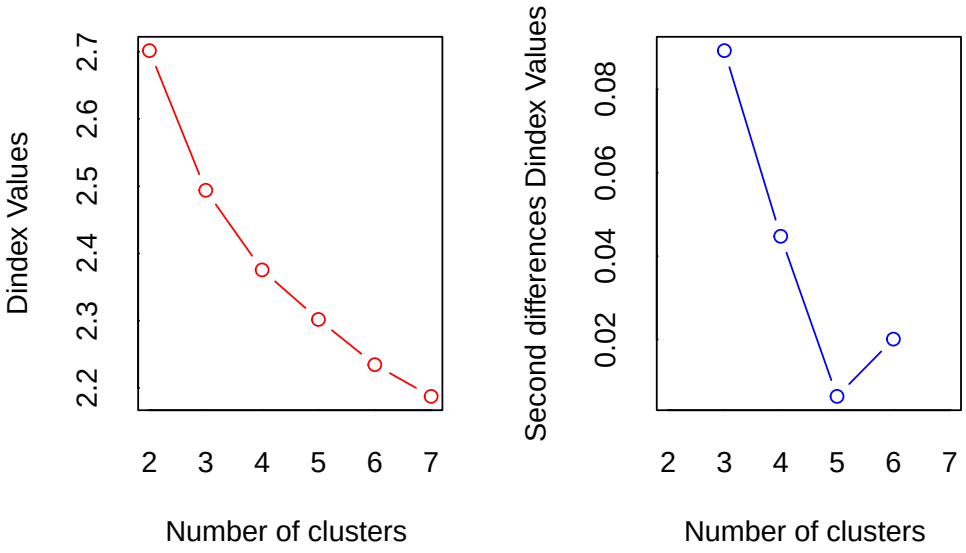
Silhouette method with K-means to determine the number of clusters to choose

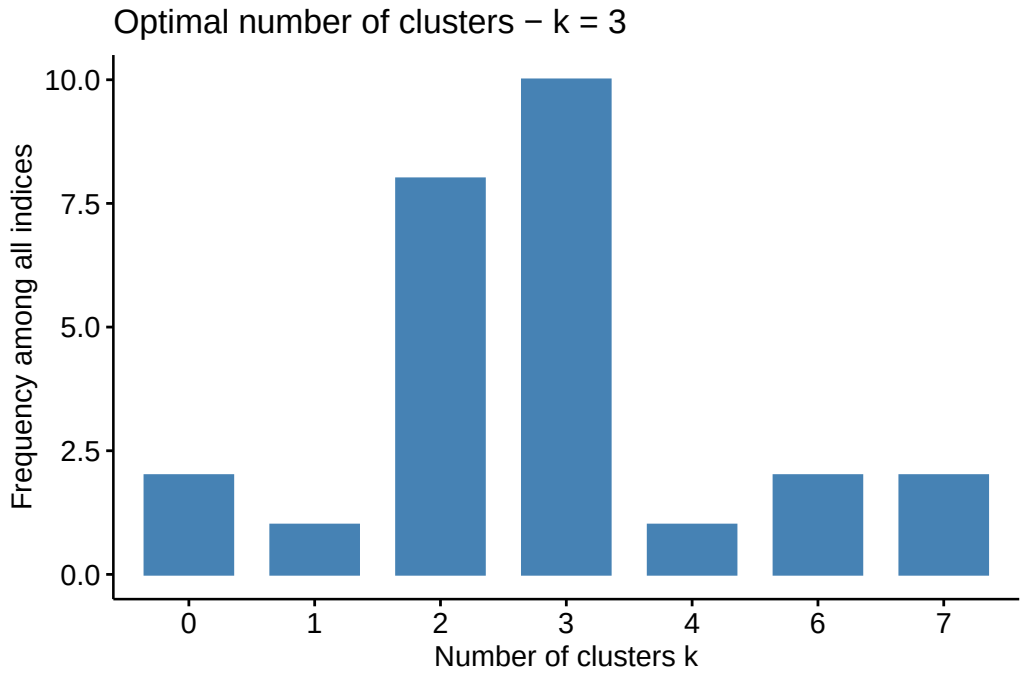
```
fviz_nbclust(df, kmeans, method = "silhouette")+
  labs(subtitle = "Silhouette method with kmeans")
```



Nb clust with K-means to determine the number of clusters to choose

```
nb <- NbClust(df, distance = "euclidean", min.nc = 2,
              max.nc = 7, method = "kmeans")
fviz_nbclust(nb)
*** : The Hubert index is a graphical method of determining the number of
    clusters.
                In the plot of Hubert index, we seek a significant knee that
    corresponds to a
                significant increase of the value of the measure i.e the
    significant peak in Hubert
                index second differences plot.

*** : The D index is a graphical method of determining the number of clusters.
                In the plot of D index, we seek a significant knee (the
    significant peak in Dindex
                second differences plot) that corresponds to a significant
    increase of the value of
                the measure.

*******************************************************************
* Among all indices:
* 8 proposed 2 as the best number of clusters
* 10 proposed 3 as the best number of clusters
* 1 proposed 4 as the best number of clusters
* 2 proposed 6 as the best number of clusters
* 2 proposed 7 as the best number of clusters

                   ***** Conclusion *****

* According to the majority rule, the best number of clusters is  3


*******************************************************************
```


Dindex Values
2.2
2.3
2.4
2.5
2.6
2.7
2
3
4
5
6
7
Number of clusters
Second differences Dindex Values
0.02
0.04
0.06
0.08
2
3
4
5
6
7
Number of clusters

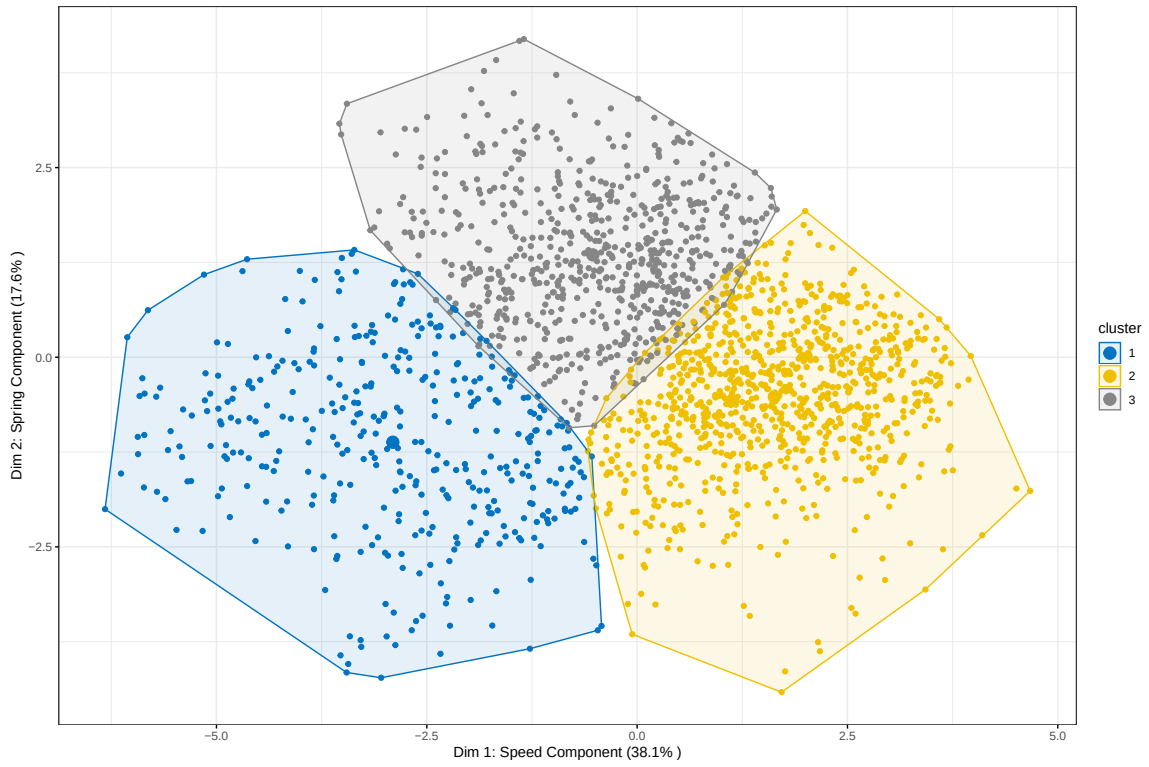

Optimal number of clusters − k = 3
Frequency among all indices
10.0
7.5
5.0
2.5
0.0
0
1
2
3
4
6
7
Number of clusters k

Appendix C

K-means Plots in Different PCs Dimensions

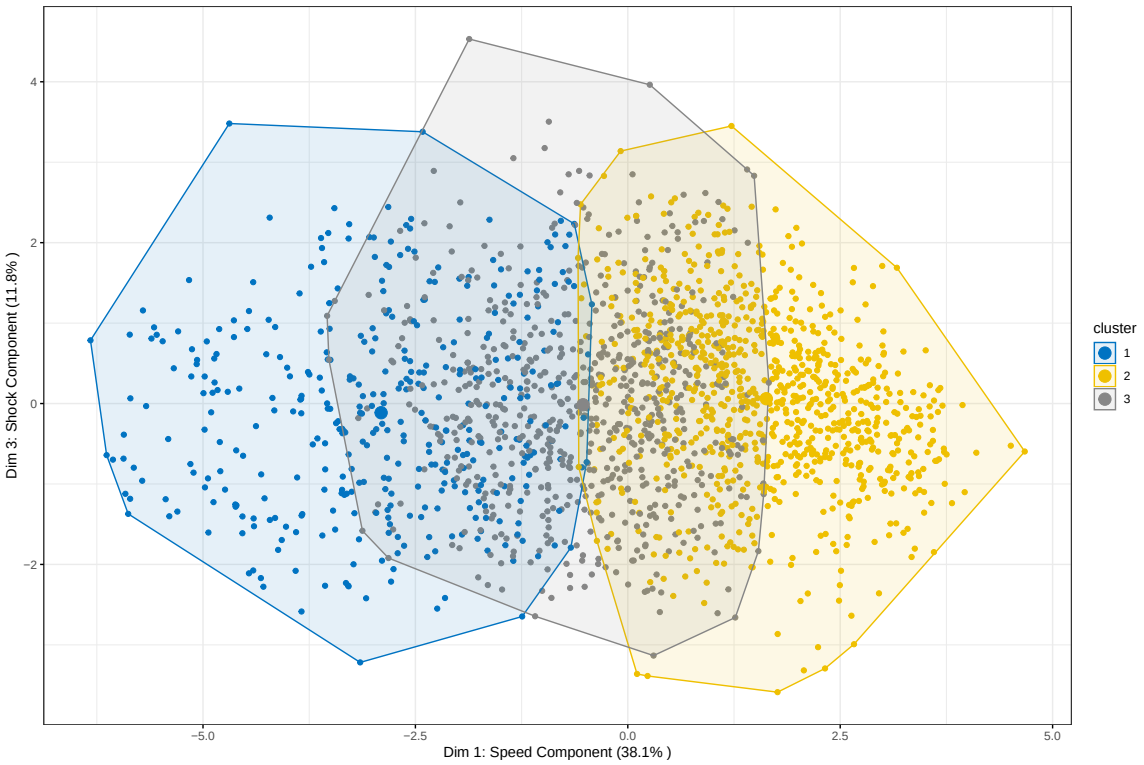
K-means clusters projection onto the PC1: Speed component, PC2: Spring component plane

```
ggscatter(ind.coord, x = "Dim.1", y = "Dim.2",
  color = "cluster", palette = "jco", ellipse = TRUE, ellipse.type = "convex",
  size = 1.5,  legend = "right", ggtheme = theme_bw(),
  xlab = paste0("Dim 1: Speed Component (", variance.percent[1], "% )" ),
  ylab = paste0("Dim 2: Spring Component (", variance.percent[2], "% )" )) +
  stat_mean(aes(color = cluster), size = 4)
```



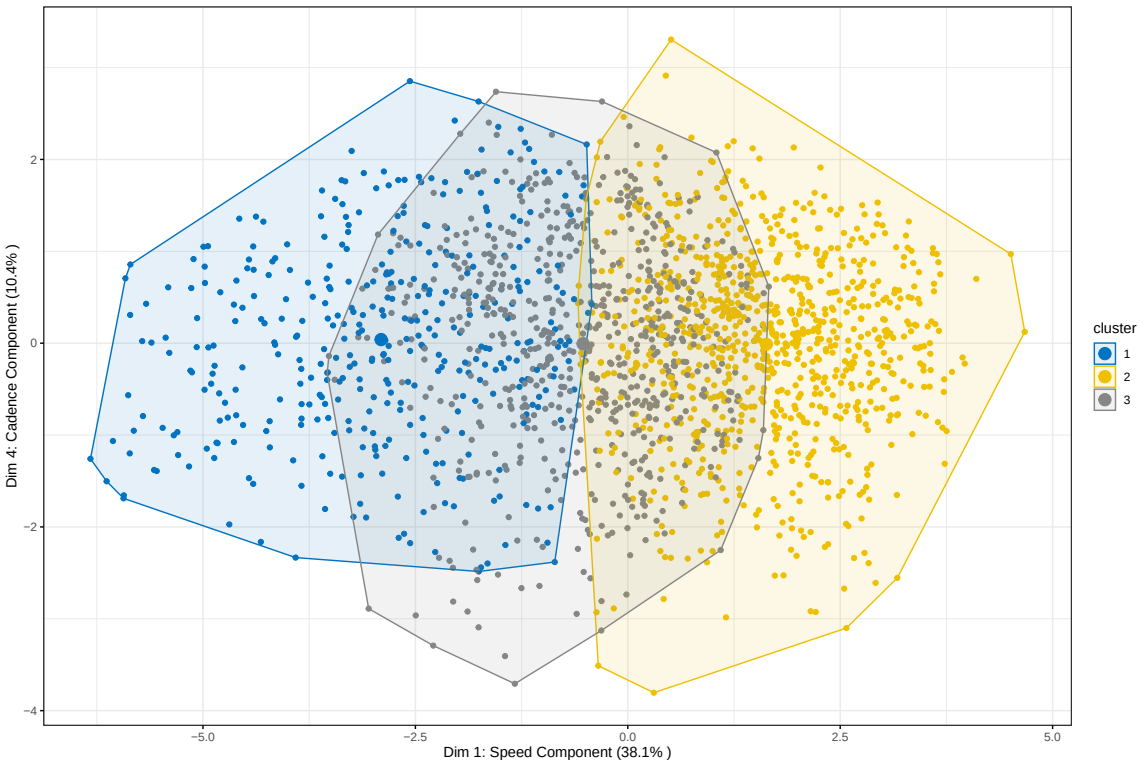
K-means clusters projection onto the PC1: Speed component, PC3: Shock component plane

```
ggscatter(ind.coord, x = "Dim.1", y = "Dim.3",
  color = "cluster", palette = "jco", ellipse = TRUE, ellipse.type = "convex",
  size = 1.5,  legend = "right", ggtheme = theme_bw(),
  xlab = paste0("Dim 1: Speed Component (", variance.percent[1], "% )" ),
  ylab = paste0("Dim 3: Shock Component (", variance.percent[3], "% )" )) +
  stat_mean(aes(color = cluster), size = 4)
```



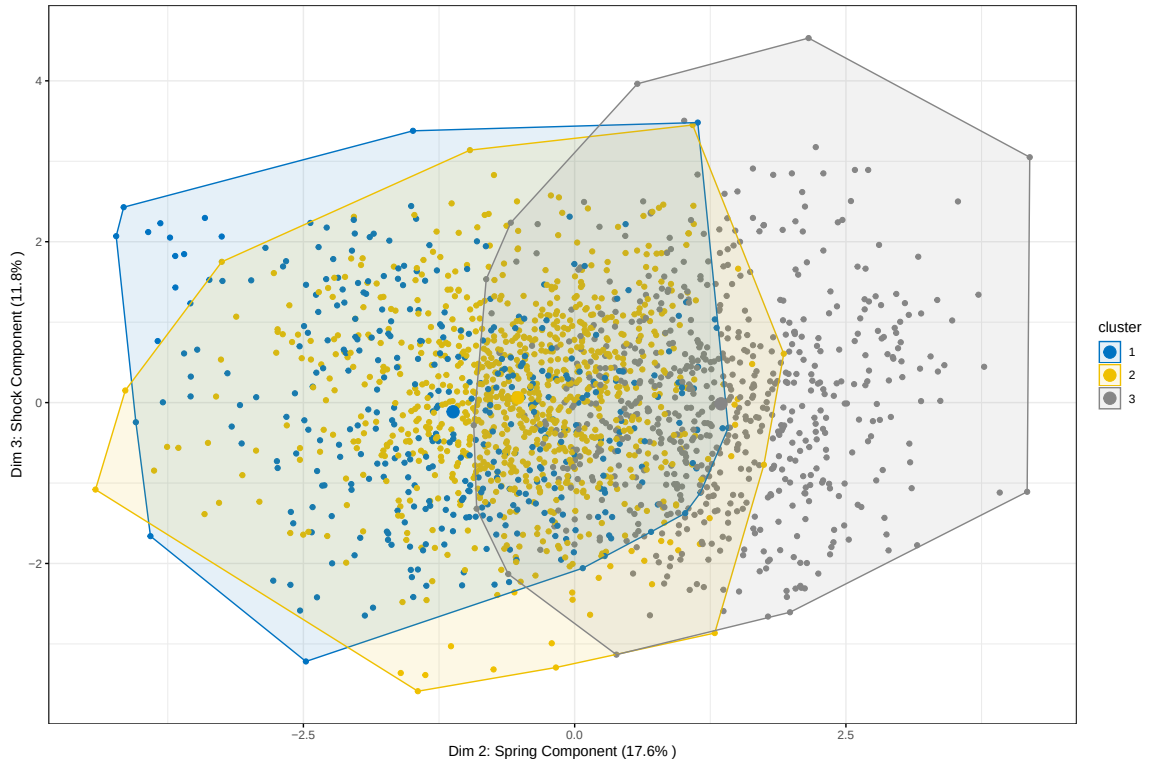
K-means clusters projection onto the PC1: Speed component, PC4: Cadence component plane

```
ggscatter(ind.coord, x = "Dim.1", y = "Dim.4",
  color = "cluster", palette = "jco", ellipse = TRUE, ellipse.type = "convex",
  size = 1.5,  legend = "right", ggtheme = theme_bw(),
  xlab = paste0("Dim 1: Speed Component (", variance.percent[1], "% )" ),
  ylab = paste0("Dim 4: Cadence Component (", variance.percent[4], "% )" )) +
  stat_mean(aes(color = cluster), size = 4)
```



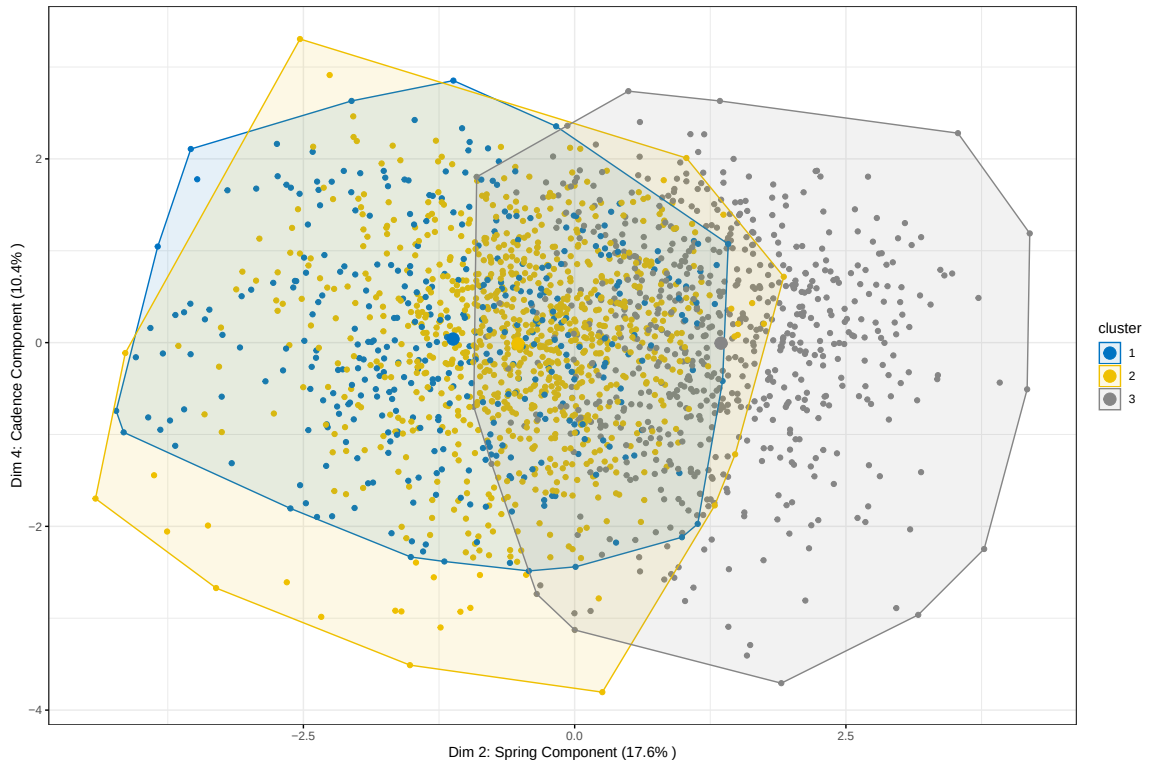
K-means clusters projection onto the PC2: Spring component, PC3: Shock component plane

```
ggscatter(ind.coord, x = "Dim.2", y = "Dim.3",
  color = "cluster", palette = "jco", ellipse = TRUE, ellipse.type = "convex",
  size = 1.5,  legend = "right", ggtheme = theme_bw(),
  xlab = paste0("Dim 2: Spring Component (", variance.percent[2], "% )" ),
  ylab = paste0("Dim 3: Shock Component (", variance.percent[3], "% )" )) +
  stat_mean(aes(color = cluster), size = 4)
```



K-means clusters projection onto the PC2: Spring component, PC4: Cadence component plane

```
ggscatter(ind.coord, x = "Dim.2", y = "Dim.4",
  color = "cluster", palette = "jco", ellipse = TRUE, ellipse.type = "convex",
  size = 1.5,  legend = "right", ggtheme = theme_bw(),
  xlab = paste0("Dim 2: Spring Component (", variance.percent[2], "% )" ),
  ylab = paste0("Dim 4: Cadence Component (", variance.percent[4], "% )" )) +
  stat_mean(aes(color = cluster), size = 4)
```



K-means clusters projection onto the PC3: Shock component, PC4: Cadence component plane

```
ggscatter(ind.coord, x = "Dim.3", y = "Dim.4",
  color = "cluster", palette = "jco", ellipse = TRUE, ellipse.type = "convex",
  size = 1.5,  legend = "right", ggtheme = theme_bw(),
  xlab = paste0("Dim 3: Shock Component (", variance.percent[3], "% )" ),
  ylab = paste0("Dim 4: Cadence Component (", variance.percent[4], "% )" )) +
  stat_mean(aes(color = cluster), size = 4)
```