Constructing a Vector With a Specific Pearson Correlation

Ermin Hodžić

Nov 2023

This paper will describe a solution to a problem that I have encountered while constructing synthetic data for a research project, to run simulations on. Without going into much detail, a part of the data that I needed to generate consisted of an n-dimensional vector and a separate matrix with n columns. Rather than generating data of the vector and the matrix uniformly at random, it would have been useful to decide (randomly) on a certain Pearson correlation coefficient of each row of the matrix with the vector. Once a row has been assigned a target correlation coefficient, we would like to generate values in that row which reflect the decided correlation coefficient.

More formally, given a vector $\vec{v}$ and a target Pearson correlation coefficient value P, the goal is to construct a vector $\vec{u}$ such that $r(\vec{u}, \vec{v}) = P$, where $r(\vec{u}, \vec{v})$ denotes the Pearson correlation coefficient between vectors $\vec{u}$ and $\vec{v}$.

Requirements. I will assume that the reader possesses a very basic understanding of vectors in a vector space, such as what vectors are and their basic properties, as well as the rules (and visualization) of their addition and subtraction. Additionally, I will assume that the reader is familiar with the inner product (or dot product) of vectors, and its relation to the angle between vectors (or cosine similarity in higher dimensions) and the norm, or magnitude, of the vector.

Contents

1 Pearson Correlation Coefficient and the Angle Between Two Vectors

Given a pair of samples x and y of size n, i.e. each consisting of n data points, the *Pearson Correlation Coefficient* is defined as:

$$r(x,y) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{(\sum_{i=1}^{n}(x_i - \bar{x})^2)(\sum_{i=1}^{n}(y_i - \bar{y})^2)}}$$

Here, x_i and y_i represent the i-th data point of each sample, and $\bar{x}$ and $\bar{y}$ represent the average value in each sample. By subtracting the average values from every part of both x and y, we are *centering* them, and making the new average value to be 0. Let us denote the centered samples as x_c and y_c, where $x_{ci} = x_i - \bar{x}$ and $y_{ci} = y_i - \bar{y}$. Then the above formula can be rewritten as:

$$r(x,y) = \frac{\sum_{i=1}^{n} x_{ci} y_{ci}}{\sqrt{(\sum_{i=1}^{n} {x_{ci}}^2)(\sum_{i=1}^{n} {y_{ci}}^2)}}$$

Those familiar with the *inner product* from analytical geometry should recognize that this equation now directly reflects the formula for the cosine of the angle between two n-dimensional vectors:

$$\cos(\sphericalangle(x_c, y_c)) = \frac{\langle x_c, y_c \rangle}{||x_c|| \cdot ||y_c||} = \frac{\langle x_c, y_c \rangle}{\sqrt{\langle x_c, x_c \rangle} \cdot \sqrt{\langle y_c, y_c \rangle}} = \frac{\sum_{i=1}^{n} x_{ci} y_{ci}}{\sqrt{(\sum_{i=1}^{n} {x_{ci}}^2)(\sum_{i=1}^{n} {y_{ci}}^2)}}$$

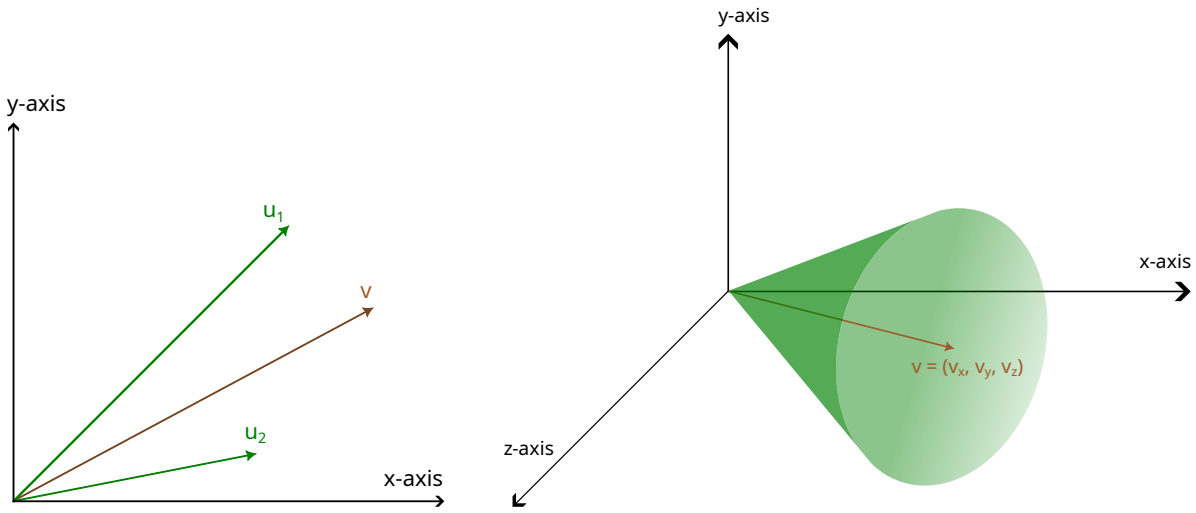
Thus, the Pearson correlation coefficient of a pair of samples is just the cosine of the angle between their centered selves in vector space. This means that the original problem definition:

> ***Problem definition 1:*** Given a vector $\vec{v}$ and a target Pearson correlation coefficient value P, the goal is to construct a vector $\vec{u}$ such that $r(\vec{u}, \vec{v}) = P$.

can be reformulated to an equivalent definition as follows:

> ***Problem definition 1(b):*** Given a vector $\vec{v}$ and a target Pearson correlation coefficient value P, the goal is to construct a centered vector $\vec{u}_c$ such that $cos(\sphericalangle(\vec{u}_c, \vec{v}_c)) = P$.

The above reformulation reveals two important parts of our general problem: (i) Find a vector at a certain angle relative to another (centered) vector; and (ii) Ensure that the found vector is centered. In the rest the text, it will first be explained how to solve the first part – finding a vector at a given angle, and then it will be explained how to ensure that it is centered.



(a) 2D – 2 possible directions

(b) 3D – infinitely many possible directions along the surface of the cone

Figure 1: Depiction of the possible (directions of) solutions for the problem of finding a vector with a certain angle relative to v.

2 Constructing a Vector at a Given Angle

Problem definition 2: Given a vector $\vec{v}$ and an angle λ, the goal is to construct a vector $\vec{u}$ such that $\sphericalangle(\vec{u}, \vec{v}) = \lambda$.

Figure 1 depicts the problem in the case of 2 and 3 dimensions. In Figure 1(a), we see that there are two possibilities for the direction of such a solution vector u in the two-dimensional space – even though the number of possible solution vectors is infinite, because there are infinitely many possible lengths of such vectors, there are only two possibilities for their direction that would fit the given angle. Figure 1(b) shows that the problem has an infinite number of directions along which a solution vector could be selected in the three-dimensional space. Every vector drawn from the centre of the coordinate system and along the exterior of the green-shaded cone is going to have the same angle relative to the vector v. In higher dimensional spaces, the set of solution directions remains infinite.

Before describing the procedure with which we can find a solution to our problem, we will need to go through some concepts of analytical geometry and trigonometry.

2.1 Projection of a Vector Onto Another

Given two vectors $\vec{a}$ and $\vec{b}$, the goal is to compute a vector $\vec{x}$ which is an orthogonal projection of $\vec{a}$ onto $\vec{b}$. Geometrically, an orthogonal projection of $\vec{a}$ onto $\vec{b}$ would represent a vector that is identical to $\vec{b}$ in direction (and thus lies on it), and is oriented the same way as $\vec{a}$ is, relative to $\vec{b}$. Its length is determined by the point at which a line that is drawn from the endpoint of $\vec{a}$ touches $\vec{b}$ orthogonally, and there exists only one such point. Thus, a projection of a vector onto another is unique.

Since $\vec{x}$ lies on $\vec{b}$, it has an identical direction, and thus we know that its coordinates are going to be proportional to coordinates of the unit vector $\vec{b_o}$, scaled by some scalar value that represents the length of the projection and determines its orientation.

$$\vec{x} = c \cdot \vec{b_o} = c \cdot \frac{\vec{b}}{||b||}$$

The question that remains is how to find c and compute the length and orientation of the projection, and for that we need to refer to trigonometry. Given a vector in 2D space, let us denote the angle that the vector creates with the x-axis by λ, and let us consider the right-triangle that is created by the x-axis, the vector, and the vertical line drawn

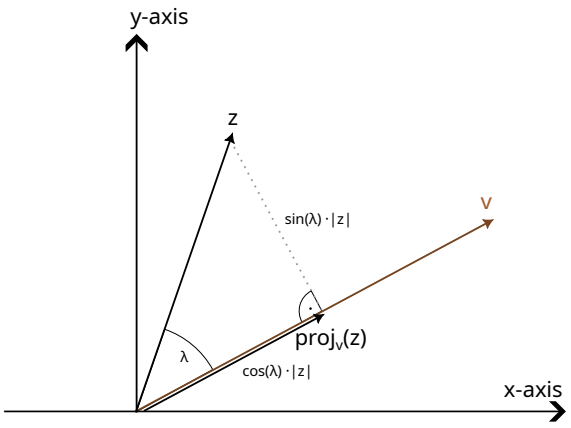


Figure 2: Projection of the vector $\vec{z}$ onto $\vec{v}$. The direction of the projection is the same as the direction of $\vec{v}$. The length and orientation of the projection is determined by the cosine of the angle between the vectors.

from the end of the vector orthogonally down onto the x-axis. We know from trigonometry that $\cos(\lambda)$ is equal to the ratio of the length of the side of the right-triangle that lies on the x-axis, and the hypotenuse (hypotenuse is in this case the vector). In other words, cosine of an angle in a right-triangle tells us exactly how the length of the non-hypotenuse side of the angle (the one that creates the angle together with the hypotenuse) scales with the length of the hypotenuse. Thus, the length of the orthogonal projection of the vector onto the x-axis is going to be equal to the length of the vector itself multiplied by $\cos(\lambda)$.

Back to our problem, we are considering an orthogonal projection onto another vector, not the x-axis. However, we still have an angle between two vectors that form a right-triangle (because one of the vectors is an orthogonal projection of the other, connecting their ends with another vector would create a right angle), and thus the length of the projection-vector is still going to scale with the length of the projected vector by a factor that is equal to the cosine of the angle between them. Thus, the scaling factor of the length of $\vec{x}$ with the length of $\vec{a}$ is given by the cosine of the angle between $\vec{a}$ and $\vec{b}$, which gives us the value of $c = \cos(\sphericalangle(\vec{a}, \vec{b})) \cdot ||a||$.

$$\vec{x} = c \cdot \frac{\vec{b}}{||b||} = \cos(\sphericalangle(\vec{a}, \vec{b})) \cdot ||a|| \cdot \frac{\vec{b}}{||b||} = \frac{\langle \vec{a}, \vec{b} \rangle \cdot ||a||}{||a|| \cdot ||b||} \cdot \frac{\vec{b}}{||b||} = \frac{\langle \vec{a}, \vec{b} \rangle}{||b||^2} \cdot \vec{b}$$

$$\implies \quad \vec{x} = \frac{\langle \vec{a}, \vec{b} \rangle}{\langle \vec{b}, \vec{b} \rangle} \cdot \vec{b}$$

We explained this process in 2D space, but does it hold in higher-dimensional vector space as well? The reason that the answer is "yes" is that the formula depends only on the cosine of the angle between two vectors and the length of the one being projected, and both of them are well-defined in vector space of any finite dimension. Thus this same formula works for any two n-dimensional vectors.

2.2 Solving The Problem

Now we will go through the procedure of generating a vector at a given angle. We will explore a simple case of the 2D space and $\lambda < \frac{\pi}{2}$, and subsequently show that the solution generalizes to any higher dimension, as well as how to handle the case when $\lambda > \frac{\pi}{2}$. Figure 3 depicts this case.

As explained in Figure 1(b), the solution space in higher dimensions is the surface of a (hyper)cone, and any vector along that surface will be a solution. We will thus start by generating a random vector $\vec{z}$ in order to pick a random direction along the surface. The solution $\vec{u}$ can then be determined by finding the point where an orthogonal line drawn from $\vec{z}$ towards $\vec{v}$ hits the surface of the (hyper)cone. $\vec{u}$ can thus be represented as a sum of its two components: (i) the orthogonal projection of $\vec{z}$ onto $\vec{v}$, which we denote as $\vec{z_1}$ on Figure 3; (ii) and the vector $\vec{z_2}$ which is orthogonal to $\vec{v}$, has its beginning at the endpoint of $\vec{z_1}$, is aimed at the endpoint of $\vec{z}$, and its length extends exactly to the point of touching the (hyper)cone.

We can obtain vector $\vec{z_1}$ using the projection formula given in the previous section, as:

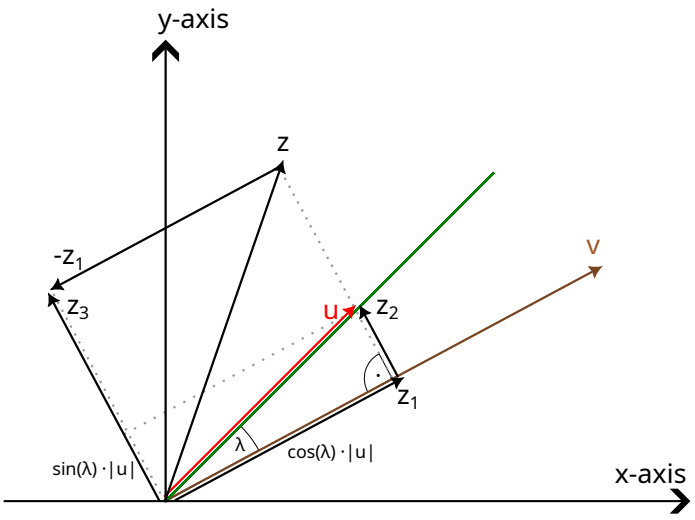


Figure 3: **Outline of the approach for finding some vector $\vec{u}$ that has an angle λ relative to a given vector $\vec{v}$, in 2 dimensions.** $\vec{u}$ can be represented as the result of the sum of the projection of some random vector $\vec{z}$ onto $\vec{v}$, and a vector that is orthogonal to that projection in the same direction that vector $\vec{z}$ came from, and whose length matches the sine-side of the right-triangle formed by $\vec{u}$ and the projection. Such an orthogonal vector is easily obtain by subtracting the projection vector from the vector $\vec{z}$.

$$\vec{z_1} = \frac{\langle \vec{z}, \vec{v} \rangle}{\langle \vec{v}, \vec{v} \rangle} \cdot \vec{v}$$

The direction and orientation of $\vec{z_2}$ is going to be identical to the vector $\vec{z_3}$ which we can obtain by subtracting $\vec{z_1}$ from $\vec{z}$. Then we can obtain $\vec{z_2}$ by scaling the unit-vector of $\frac{\vec{z_3}}{||\vec{z_3}||}$ by magnitude of $\vec{z_2}$ that we have to somehow calculate.

$$\vec{z_3} = \vec{z} - \vec{z_1}$$
$$\vec{z_2} = \frac{||\vec{z_2}||}{||\vec{z_3}||} \cdot \vec{z_3}$$

Using right-angle trigonometry, we know that the magnitudes of $\vec{z_1}$ and $\vec{z_2}$ are in proportion to the magnitude of $\vec{u}$, scaled by $\cos(\lambda)$ and $\sin(\lambda)$. We thus have:

$$||\vec{z_2}|| = \sin(\lambda) \cdot ||\vec{u}|| \quad \wedge \quad ||\vec{z_1}|| = \cos(\lambda) \cdot ||\vec{u}|| \quad \Longrightarrow \quad ||\vec{z_2}|| = \frac{\sin(\lambda) \cdot ||\vec{z_1}||}{\cos(\lambda)}$$

$$\vec{z_2} = \frac{||\vec{z_2}||}{||\vec{z_3}||} \cdot \vec{z_3} \quad \Longrightarrow \quad \vec{z_2} = \frac{\sin(\lambda) \cdot ||\vec{z_1}||}{\cos(\lambda) \cdot ||\vec{z_3}||} \vec{z_3}$$

Then the solution is given by $\vec{u} = \vec{z_1} + \vec{z_2}$.

Now, let us consider scenarios that deviate from what we see on Figure 3. **First**, we assumed that $\lambda < \frac{\pi}{2}$, but what if that is not the case? It is simple – we flip the orientation of vector $\vec{v}$ to turn it into $-\vec{v}$ (just negate every coordinate), and instead of looking for a solution with the angle λ, we look for one with the angle $\pi - \lambda$. **The second** matter to consider is that it is possible that the randomly-generated vector $\vec{z}$ is not going to be in the positive direction of $\vec{v}$; i.e. the sign of correlation of $\vec{z}$ with $\vec{v}$ is going to be the opposite of the cosine of λ, and thus the opposite of what we want. This is solved by simply "turning the vector around" by negating the coordinates of $\vec{z}$ in case that $\cos(\lambda) \cdot \cos(\sphericalangle(\vec{z}, \vec{v})) < 0$. **The third** matter to consider is when the angle $\lambda = \frac{\pi}{2}$. In this case, we are looking for any vector that is orthogonal to $\vec{v}$, and we can simply return the vector $\vec{z_3}$ as the solution. **Lastly**, we have explained the procedure for the 2D case, but none of the formulas used depend on the number of dimensions, and the procedure can be applied to any n-dimensional vector $\vec{v}$ and any given angle.

The following pseudocode implements the algorithm described above.

Algorithm 1: vectorWithCosineSimilarity(V, targetCosAngle)

Input: A vector V, a cosine similarity value targetCosAngle
Output: A random vector whose cosine similarity with V is equal to $targetCosAngle$

if *(targetCosAngle* < 0*)* **then**
    flip(V)
    targetCosAngle = -targetCosAngle
Z = randomVector(V.numDimensions)
if *(targetCosAngle* $* cos(angle(Z, V)) < 0$*)* **then**
    flip(Z)
$Z_1 = V * (\langle Z, V\rangle / \langle V, V\rangle)$
$Z_3 = Z - Z_1$
if *(targetCosAngle* $==$ *0)* **then**
    return Z_3
targetSinAngle = sqrt(1 - targetCosAngle * targetCosAngle)
Z_2 = Z_3 * (sqrt($\langle Z_1, Z_1\rangle$) * targetSinAngle) / (sqrt($\langle Z_3, Z_3\rangle$) * targetCosAngle)
$U = Z_1 + Z_2$
return U

3 Solving the General Problem

As a reminder, here is the problem definition that we ultimately need to solve:

> ***Problem definition 1(b):*** Given a vector $\vec{v}$ and a target Pearson correlation coefficient value P, the goal is to construct a centered vector $\vec{u}_c$ such that $cos(\sphericalangle(\vec{u}_c, \vec{v}_c)) = P$.

The procedure described in the previous section can be used to construct some vector $\vec{u}$ such that $\cos(\sphericalangle(\vec{v}, \vec{u})) = P$, but unfortunately that does not guarantee that $r(\vec{v}, \vec{u}) = P$, because the procedure is not set to produce a centered vector. As a reminder, the Pearson correlation coefficient is equal to the cosine of the angle between *centered* vectors. When we center a vector, we subtract the average value from each of its coordinates, and thus change the direction of the vector. So what we are really looking for is some *centered* vector $\vec{u}_c$, with a specific (cosine of the) angle relative to the *centered* vector $\vec{v}_c$. Thus, giving $\vec{v}_c$ as the input to the algorithm of the previous section would solve one half of the issue.

Whether a vector is centered or not is determined only by its direction, and not its orientation or length, as shown below:

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \quad \Longrightarrow \quad \forall \alpha \in \mathbb{R} : \overline{\alpha x} = \alpha \cdot \overline{x}$$

$$\vec{x}_c = (x_1 - \overline{x}, x_n - \overline{x}, ..., x_n - \overline{x})$$

$$\Longrightarrow \overline{x_c} = \frac{1}{n}\sum_{i=1}^{n} x_{ci} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x}) = \frac{1}{n}\sum_{i=1}^{n} x_i - \frac{1}{n}\sum_{i=1}^{n}\overline{x} = \overline{x} - \frac{n \cdot \overline{x}}{n} = 0$$

$$\Longrightarrow \forall \alpha \in \mathbb{R} : \overline{\alpha x_c} = \alpha \cdot \overline{x_c} = 0$$

Since a projection of a vector onto another retains the direction of the target vector, that means that $\vec{z_1}$ (referring to Figure 3 now) is going to be centered if $\vec{v}$ is centered. Similarly, $\vec{z_2}$ is going to be centered if $\vec{z_3}$ is centered, since $\vec{z_2}$ is just $\vec{z_3}$ multiplied by a scalar. The consequence of this is that then $\vec{u} = \vec{z_1} + \vec{z_2}$ would also be centered, because of the following:

$$\overline{x} = 0 \wedge \overline{y} = 0 \quad \Longrightarrow \quad \overline{x + y} = \frac{1}{n}\sum_{i=1}^{n}(x_i + y_i) = \frac{1}{n}\sum_{i=1}^{n} x_i + \frac{1}{n}\sum_{i=1}^{n} y_i = \overline{x} + \overline{y} = 0$$

Then the final question to answer is how to ensure that $\vec{z_3}$ is centered. For that, observe that $\vec{z_3} = \vec{z} - \vec{z_1}$. From that, and the equations just above, it then follows that $\vec{z_3}$ will be centered if $\vec{z}$ and $\vec{z_1}$ are both centered. Thus, we only need to center the randomly-generated vector z in our algorithm, and we will obtain a correct solution to our original problem.

The pseudocode of the final algorithm is given below.

Algorithm 2: vectorWithPearsonCorrelation(V, P)

Input: A vector V, a target Pearson correlation factor P
Output: A random vector whose Pearson correlation with V is equal to P

V_c = center(V)
targetCosAngle = P
if *(targetCosAngle* < 0*)* **then**
    flip(V_c)
    targetCosAngle = -targetCosAngle
Z = randomVector(V_c.numDimensions)
Z_c = center(Z)
if *(targetCosAngle* $* cos(angle(Z_c, V_c)) < 0$*)* **then**
    flip(Z_c)
$Z_1 = V_c$ * ($\langle Z_c, V_c \rangle$ / $\langle V_c, V_c \rangle$)
$Z_3 = Z_c - Z_1$
if *(targetCosAngle* $== 0$*)* **then**
    return Z_3
targetSinAngle = sqrt(1 - targetCosAngle * targetCosAngle)
$Z_2 = Z_3$ * (sqrt($\langle Z_1, Z_1 \rangle$) * targetSinAngle) / (sqrt($\langle Z_3, Z_3 \rangle$) * targetCosAngle)
$U_c = Z_1 + Z_2$
return U_c