Analecta Technica Szegedinensia

Vol. 18, No. 3 ISSN 2064-7964 2024

DEVELOPING EXCEL VBA FUNCTIONS FOR THE MATHEMATICAL MODELING OF THREE-DIMENSIONAL VECTORS

[1]*Zoltán Fabulya*, [2]*György Hampel*

[1]University of Szeged Faculty of Engineering, 7 Mars sq., H6724, Szeged, Hungary,
e-mail: fabulya@mk.u-szeged.hu,
[1]University of Szeged Faculty of Engineering, 7 Mars sq., H6724, Szeged, Hungary

Received: 5th June | *Accepted: 19th September*

ABSTRACT

The current computer support for operations involving three-dimensional vectors is insufficient, even in widely used programs like Excel. These programs lack customised features specifically designed for vector operations. However, this limitation can be overcome by using the options provided by Visual Basic for Applications. By creating user functions, we can effectively calculate various results related to the mathematical application of vectors. These functions include determining the magnitude and absolute value of a vector, calculating the angle between two vectors or the cosine of that angle, finding the scalar product and vector product of two vectors, and evaluating the mixed product of three vectors. By incorporating these custom functions into the spreadsheet program, users can easily perform mathematical calculations pertaining to three-dimensional vectors.

Keywords: Excel VBA, programming, vector operations

1. INTRODUCTION

In certain branches of mathematics, there are certain topics that require extensive calculations. This is also true for linear algebra, where the manipulation of three-dimensional vectors and their operations holds significant importance in practical applications. However, there is a lack of computer support in this regard, as widely used spreadsheet programs, such as Excel, do not offer such functionalities [1].
Our objective was to develop functions that can compute the outcomes of operations performed on three-dimensional vectors using the Visual Basic for Applications extension, which enables programming within Excel. By doing so, we aim to expand the range of functions available in the spreadsheet program and provide a familiar approach to perform vector operations [2].

2. MATERIALS AND METHODS

We utilise Microsoft Excel 2021 along with Visual Basic for Application (VBA), which offers the programming environment necessary to create functions [3] [4] [5]. These functions are designed to compute a result based on the of vector values in the array variables present in their argument. Microsoft Excel is a versatile spreadsheet program that can be applied in various scenarios such as financial calculations [6], survey evaluations [7], automation of statistical hypothesis tests [8] [9], and even in aiding the coordination of industrial processes [10].

2.1. Materials

The vectors are inputted using algebraic notation, consisting of three coordinate values that represent the vector's components along specific directions within three-dimensional space, indicating the magnitude of the vector in each direction (1).

$$\bar{a} = (a_1; a_2; a_3)$$

where:

$\bar{a}$ – vector; a_1, a_2, a_3 – vector components, coordinates

The vector's coordinates can be denoted using a real number, allowing us to determine the outcome of vector operations based on their coordinate values. By adding vectors (2), subtracting them (3), scaling them by a scalar (4), or finding the vector product of two vectors (5), we obtain a resultant vector using the following formulas:

$$\bar{a} + \bar{b} = (a_1; a_2; a_3) + (b_1; b_2; b_3) = (a_1 + b_1; a_2 + b_2; a_3 + b_3) \tag{2}$$

$$\bar{a} - \bar{b} = (a_1; a_2; a_3) - (b_1; b_2; b_3) = (a_1 - b_1; a_2 - b_2; a_3 - b_3) \tag{3}$$

$$\lambda \cdot \bar{a} = \lambda \cdot (a_1; a_2; a_3) = (\lambda \cdot a_1; \lambda \cdot a_2; \lambda \cdot a_3) \tag{4}$$

$$\begin{gathered} \bar{a} \times \bar{b} = (a_1; a_2; a_3) \times (b_1; b_2; b_3) = \\ = (a_2 \cdot b_3 - a_3 \cdot b_2; -a_1 \cdot b_3 + a_3 \cdot b_1; a_1 \cdot b_2 - a_2 \cdot b_1) \end{gathered} \tag{5}$$

where:
$\lambda \in \mathbb{R}$ – scalar

There exist various operations involving vectors that yield a scalar, rather than another vector, as a result. These operations include the absolute value of a vector (6), the scalar product of two vectors (7), the mixed product of three vectors (8), the cosine of the angle formed by two vectors (9), and the enclosed value (10). These quantities can be computed using the following mathematical formulas:

$$|\bar{a}| = |(a_1; a_2; a_3)| = \sqrt{a_1^2 + a_2^2 + a_3^2} \tag{6}$$

$$\bar{a} \cdot \bar{b} = (a_1; a_2; a_3) \cdot (b_1; b_2; b_3) = a_1 \cdot b_1 + a_2 \cdot b_2 + a_3 \cdot b_3 \tag{7}$$

$$\begin{gathered} \bar{a} \cdot \left(\bar{b} \times \bar{c}\right) = (a_1; a_2; a_3) \cdot \left((b_1; b_2; b_3) \times (c_1; c_2; c_3)\right) = \\ = a_1 \cdot (b_2 \cdot c_3 - b_3 \cdot c_2) - a_2 \cdot (b_1 \cdot c_3 - b_3 \cdot c_1) + \\ + a_3 \cdot (b_1 \cdot c_2 - b_2 \cdot c_1) \end{gathered} \tag{8}$$

$$cos\varphi = \frac{\bar{a} \cdot \bar{b}}{|\bar{a}| \cdot |\bar{b}|} \tag{9}$$

$$\varphi = arccos \frac{\bar{a} \cdot \bar{b}}{|\bar{a}| \cdot |\bar{b}|} \tag{10}$$

2.2. The Excel VBA add-in

The Visual Basic for Applications add-in offers the opportunity to create custom functions that can be utilised in the same manner as other Excel functions. When programming, Excel provides an integrated program development environment that uses the Visual Basic programming language. The variables included in the function arguments store data from three-dimensional arrays, specifically, the coordinates of vectors. Certain data within the array variables can be accessed by using their index. For example, the first coordinate (a_1) of a vector ($\bar{a}$) can be found at the single value index (a(1)). Therefore, the index value must be specified in brackets following the variable name. The function's result is formed by utilising the variables specified in the arguments. To achieve this, a value assignment statement must be used with mathematical expressions which can be represented using the standard mathematical operation symbols (operators).

3. RESULTS AND DISCUSSION

Our objective was to develop a program consisting of functions that can compute the results of vector operations. However, it is important to consider that the result of certain operations may be utilised in subsequent operations. Basic operations, such as vector addition, vector subtraction, and scalar

multiplication, do not necessitate the creation of a separate function because they can be directly formulated within the spreadsheet cell. These formulas differ from conventional ones in that the result is not a single value in a single cell, but rather an array of three cells representing the three coordinates of the resultant vector.
The scalar product of two vectors can be obtained by employing the following function:

```
Public Function VectorScalarProduct(a, b)
    s = 0
    For i = 1 To 3
        s = s + a(i) * b(i)
    Next i
    VectorScalarProduct = s
End Function
```

The function is named `VectorScalarProduct` and it calculates the result by operating on two vectors stored in its arguments (array variables *a* and *b*) according to (7). To compute the result, the program required the elementary summation algorithm, where, by increasing the initial 0 value of a variable by multiplying the coordinates with the same index (`s=s+a(i)*b(i)`), we get the result, the value of the function (`VectorScalarProduct=s`).
The absolute value of a vector (6) can be derived from the result of the scalar product.

$$|\bar{a}| = \sqrt{\bar{a}\cdot\bar{a}} = (\bar{a}\cdot\bar{a})^{\frac{1}{2}} \tag{11}$$

By leveraging this relationship, the VectorAbs function can be effortlessly developed:

```
Public Function VectorAbs(a)
    VectorAbs = VectorScalarProduct(a, a) ^ 0.5
End Function
```

The caret symbol (^) serves as the exponentiation operator, utilised to deduce the root from the result of the scalar product.
The cosine of an angle formed by two vectors can be computed through (9), which requires the scalar product result of the vectors and their magnitudes. The function is named `VectorCosFi`:

```
   Public Function VectorCosFi(a, b)
       numerator = VectorScalarProduct(a, b)
       denominator = VectorAbs(a) * VectorAbs(b)
       If denominator = 0 Then
           VectorCosFi = 0
       Else
           VectorCosFi = numerator / deniminator
       End If
   End Function
```

Initially, the computation involves determining the values of both the numerator (`numerator`) and denominator (`denominator`) values in the fraction that ultimately yields the final outcome. It is crucial to note that the denominator of the fraction cannot equate to zero, necessitating a verification process. The denominator can solely assume a zero value if one of the vectors is a zero vector, resulting in an absolute value of zero. Consequently, the scalar product of the vectors also equals zero, serving as a prerequisite and adequate condition for the vectors to be orthogonal. Consequently, the cosine of the angle formed by the

vectors also assumes a zero value (`VectorCosFi=0`) in cases where they are perpendicular. In scenarios where the denominator is nonzero, the fraction's value (`numerator/denominator`) determines the final outcome.
To determine the angle formed by two vectors, the cosine of the angle can be utilised. The outcome is derived using the arc cosine (`Acos`) worksheet function.

```
Public Function VectorFi(a, b)
     CosFi = VectorCosFi(a, b)
     VectorFi = WorksheetFunction.Acos(CosFi)
End Function
```

To define the vector product of two vectors, it is necessary to utilise a function that yields a vector as its output. To achieve this, an array variable (`t(1 To 3)`) must be created, where the three elements can be accessed through their respective indices. The two vectors to be multiplied (`a, b`) serve as the arguments for this function. The program for this function is as follows:

```
Public Function VectorVectorProduct(a, b)
    Dim t(1 To 3)
    t(1) = a(2) * b(3) - a(3) * b(2)
    t(2) = -a(1) * b(3) + a(3) * b(1)
    t(3) = a(1) * b(2) - a(2) * b(1)
    VectorVectorProduct = Application.Transpose(t)
End Function
```

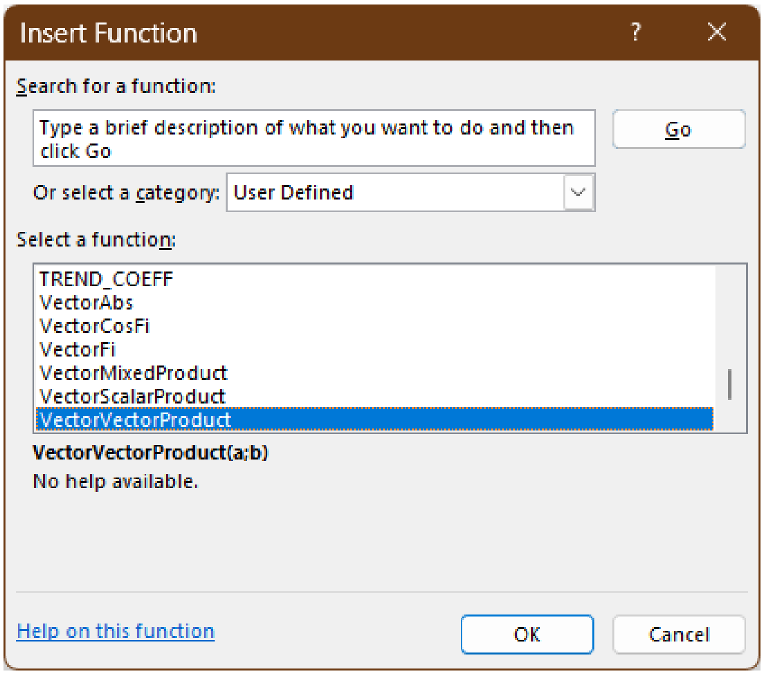
The `VectorVectorProduct` function requires the calculation of the three coordinates of the resulting vector. Typically, vectors are interpreted as columns, so an array represented as a row in Excel is transposed to become a column. This columnar array then represents the resulting vector of the vector product. If there is a need to utilise this result in a different function, it can be transposed back into an array variable that can be interpreted as a data line. This process is essential for the `VectorMixedProduct` function, which involves the mixed product of three vectors:

```
Public Function VectorMixedProduct(a, b, c)
    Dim t As Variant
    t=Application.Transpose(VectorVectorProduct(b,c))
    VectorMixedProduct = VectorScalarProduct(a, t)
End Function
```

To create the mixed product, the initial step involves creating a vector product, followed by using the outcome of this process to calculate the final scalar product. Therefore, it is satisfactory to utilise the functions that have already been created.

3.1. Using our custom functions

The custom-made functions can be found in the user-defined category in the Insert Function dialogue box (Fig. 1). Having selected the desired function, its arguments are entered in the usual way. When using a function whose result is a vector, one must first select the location required for the result (an array of three cells below each other), and then complete the creation of the formula with the CTRL-SHIFT-ENTER key combination (or a simple ENTER in Excel versions published after 2018).



Insert Function

Search for a function: Type a brief description of what you want to do and then click Go

Or select a category: User Defined

Select a function:

TREND_COEFF
VectorAbs
VectorCosFi
VectorFi
VectorMixedProduct
VectorScalarProduct
VectorVectorProduct

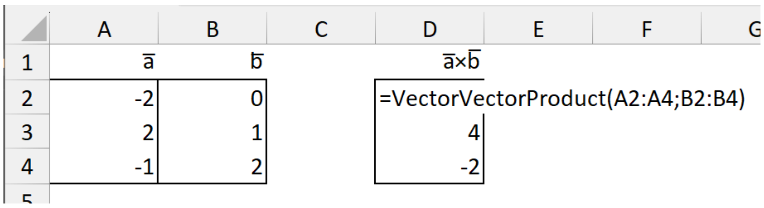
VectorVectorProduct(a;b)

No help available.

Figure 1. The dialogue box of Insert function

Fig. 2 shows the part of the calculator in which the calculation of the vector product ($\bar{a} \times \bar{b}$) of two vectors is done by typing with the function called VectorVectorProduct. The range D2:D4 must be selected for the result. In the argument of the function, there is a reference to the two vectors to be multiplied as A2:A4 and B2:B4. In Excel versions prior to 2018, the creation of the array formula must be closed with the CTRL-SHIFT-ENTER key combination, for later versions pressing ENTER is enough to do the calculations.

Figure 2: Using a function by typing



	A	B	C	D	E	F	G
1	$\bar{a}$	$\bar{b}$		$\bar{a}\times\bar{b}$			
2	-2	0		=VectorVectorProduct(A2:A4;B2:B4)			
3	2	1		4			
4	-1	2		-2			

4. CONCLUSIONS

The prepared functions can be used in Excel in the conventional manner to perform mathematical calculations on three-dimensional vectors. These functions will not remain solely accessible within the workbook where they were initially created and saved, but will also be accessible if the workbook is opened during our work. Additionally, there is the possibility of developing an extension to facilitate the user-friendly utilisation of these functions without the need to open the file.

REFERENCES

[1] B. L. Matteson, Microsoft Excel Visual Basic Programmer's Guide, Microsoft Press, Washington, 1995

[2] G. Kovalcsik, Az Excel programozása / Programming in Excel, Computerbooks, Budapest, 2005
[3] M., Alexander, D., Kusleika, Excel 2016 Power Programming with VBA, John Wiley & Sons, Inc., Hoboken, New Jersey, 2016
[4] R., Mansfield, Mastering VBA for Microsoft Office 365, 2019 Edition, John Wiley & Sons, Inc., Hoboken, New Jersey, 2019 https://doi.org/10.1002/9781119579366
[5] J., Walkenbach, Excel 2016 Bible (1st ed.), John Wiley & Sons, Inc., Hoboken, New Jersey, 2015
[6] B. Zsótér, I. Túri, Economical calculations related to a smoking technology investment of a pork processing plant. Annals of Faculty of Engineering Hunedoara – International Journal of Engineering 15 (4) (2017), pp. 57-61.
[7] B. Zsótér, A. Tóth, Examination of statisfaction related to investments (2006-2011) accomplished by the local council in Abony, Analecta Technica Szegedinensia 8 (1) (2014), pp. 33-37. https://doi.org/10.14232/analecta.2014.1.33-37
[8] Gy. Hampel, Excel alkalmazása normális eloszlás tesztelésére Shapiro-Wilk próbával / Using Excel to Test Normal Distribution with Shapiro-Wilk Test, Jelenkori társadalmi és gazdasági folyamatok / Current social and economic processes, 13 (1-2) (2018), pp. 77-82. https://doi.org/10.14232/jtgf.2018.1-2.77-82
[9] Gy. Hampel, Egymintás t-próba programozható kialakítása Excel VBA környezetben / Designing Programmable One-Sample t-Test in Excel VBA Environment, Jelenkori társadalmi és gazdasági folyamatok / Current social and economic processes, 13 (3-4) (2018b), pp. 169-175. https://doi.org/10.14232/jtgf.2018.3-4.169-175
[10] Z. Fabulya, Hőkezelési folyamatok összehangolása Excel VBA szolgáltatásokkal / Coordination of Heat Treatment Processes with Excel VBA Services, Jelenkori társadalmi és gazdasági folyamatok / Current social and economic processes, 12 (4) (2017), pp. 19-25. https://doi.org/10.14232/jtgf.2017.4.19-25