

CHAPTER 31

Higher order arithmetic sequences and applications

  10.56238/pacfdnsv1-031

Jhon Kenedy Moura Chagas

Faculty of Agronomy and Veterinary Medicine,
UnB/Brasilia
E-mail: kenedymc@gmail.com

Josimar da Silva Rocha

Academic Department of Mathematics, UTFPR/Cornélio
Procópio
E-mail: jrocha@utfpr.edu.br

ABSTRACT

In everyday life it is common to deal with some situations that involve the need to determine a general rule to describe them. To help in the description of phenomena that involve quantities that can be expressed in terms of polynomial functions, there is a need to find instruments that facilitate obtaining such polynomial functions from the data analyzed in the process. An easy way to obtain polynomial functions

from a sequence of real numbers is through the study of Higher Order Arithmetic Sequences. If the sequence obtained from the data collected through an experiment is polynomial, then this sequence is called an arithmetic sequence of order k , where k is the degree of the polynomial. With the study of higher order arithmetic sequences it is possible to obtain from a sequence of real numbers both the formula that describes this sequence and the formula for the sum of the first n terms of this sequence. With this, we will be able to obtain general formulas that allow us to obtain an estimate for the behavior of different phenomena. Thus, through the bibliographic survey, consulting the reference [1], the research project began with weekly meetings.

Keywords: Sequences, Polynomials, Pascal's Triangle, Newton's binomial, Functions.

1 INTRODUCTION

A higher-order arithmetic sequence is a sequence in which the formula that defines it is a polynomial function. In this way, we say that a sequence of order n is a polynomial function of degree n .

With the study of higher order arithmetic sequences, it is possible to obtain from a sequence of real numbers both the formula that describes the sequence and the formula for the sum of the first n terms of this sequence. Thus, we will be able to obtain general formulas that allow us to obtain an estimate for the behavior of different phenomena. For more information see [3].

Goals

The research objectives were aimed at:

- Introduce the concept of higher order arithmetic sequence;
- Find properties of higher order arithmetic sequences;
- Look for examples that illustrate and justify the study and application of higher-order arithmetic sequences.

2 METHODOLOGY

Initially, we sought to study various contents that served as prerequisites for the study of arithmetic sequences, such as arithmetic progressions, Pascal's triangle, Newton's binomial, among others. For this purpose, the reference [1] was basically used.

Through the introductory study, the need arose to study mathematical techniques of induction to obtain general results that were contemplated in the demonstrations of the main results. This was made possible by consulting [4].

Bibliographic research was necessary, we used, for example, [3], to obtain applications involving higher order arithmetic sequences through phenomena whose behavior can be translated through a polynomial function.

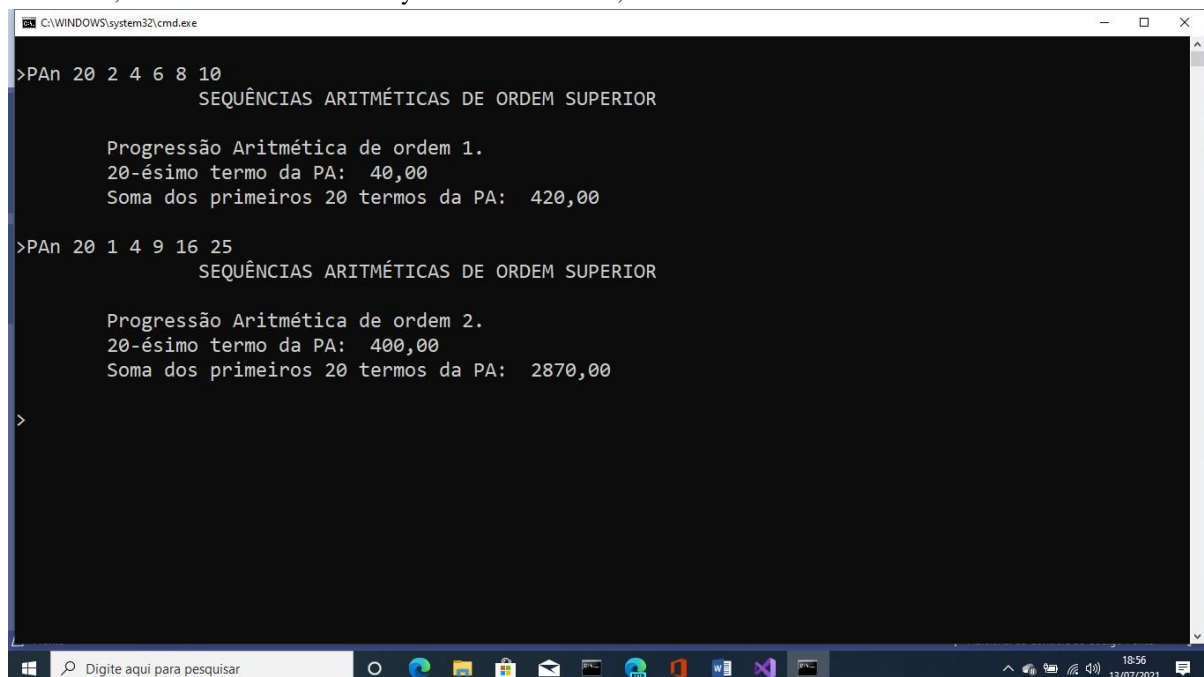
A study of programming in the C language using [2] was also necessary for the development of a software to integrate part of the knowledge about arithmetic progressions of higher order, and, to reach this end, we consulted [6]. A version of this program can be obtained in the appendix of this work. In figure 1 we have a screenshot of the program execution.

Figure 1 - Example of use of the developed program –

Caption: PA (arithmetical progression)

>PAn 20 2 4 6 8 10 - Higher order arithmetic sequences - Arithmetic progression of order 1 - 20th term of the Arithmetic Progression: 40,00 - Sum of the first twenty terms of PA: 420,00

>PAn 20 1 4 9 16 25 - Higher order arithmetic sequences - Arithmetic progression of order 2 - 20th term of the Arithmetic Progression: 400,00 - Sum of the first twenty terms of PA: 2870,00



```
C:\WINDOWS\system32\cmd.exe
>PAn 20 2 4 6 8 10
      SEQUÊNCIAS ARITMÉTICAS DE ORDEM SUPERIOR

Progressão Aritmética de ordem 1.
20-ésimo termo da PA: 40,00
Soma dos primeiros 20 termos da PA: 420,00

>PAn 20 1 4 9 16 25
      SEQUÊNCIAS ARITMÉTICAS DE ORDEM SUPERIOR

Progressão Aritmética de ordem 2.
20-ésimo termo da PA: 400,00
Soma dos primeiros 20 termos da PA: 2870,00

>
```

3 RESULTS AND DISCUSSION

First, we created a special notation to designate each term in the difference table. In $a_n^{(c)}$, according to the table of differences, (c) indicates in which line the term is found and n indicates the position of the term in the indicated line. This has since been used to refer to sequence terms used in the context of the table of differences.

Soon after we define what we call the table of differences. This will be used to obtain from the sequence $(a_n^{(1)})$ the difference between its terms that will be taken as another sequence, $(a_n^{(2)})$, doing this for $k+1$ times to obtain the sequence $(a_n^{(k+1)})$. Thus, the term $a_n^{(c)}$ will be obtained recursively by equation (1).

$$a_n^{(c)} = a_{n+1}^{(c-1)} - a_n^{(c-1)} \quad (1)$$

Hence, a numerical sequence (a_n) is an arithmetic sequence of order k if the sequence $(a_n^{(k+1)})$ obtained from the difference table is constant for some non-zero constant.

Through discussions we arrived at the determination of a formula for the general term of an arithmetic sequence of order k . This is expressed according to equation (2).

$$a_n = \sum_{j=0}^k \binom{n-1}{j} a_1^{(j+1)} \quad (\text{two})$$

Using the properties seen in [1] and [5] we were able to demonstrate a formula that describes the behavior of the sum of the first n terms of an arithmetic sequence of order k . Equation 3 represents the sum formula.

$$S_n = \sum_{j=0}^k \binom{n}{j+1} a_1^{(j+1)} \quad (3)$$

To calculate the binomial coefficients present so far in equations (2) and (3) we use equation (4), and this equation is valid for all natural n, p .

$$\binom{n}{p} = \frac{n \cdot (n-1) \cdot (n-2) \cdot \dots \cdot (n-(p-1))}{p!} \quad (4)$$

After obtaining general formulas for a_n and S_n , we started to study cases that led us to a diversity of points that could be developed during the research.

We notice that the sum of any row of the table of differences, starting from the second, with the first element of the previous row results in the term $a_{n+1}^{(1)}$ as described in equation 5.

$$S_n^{(2)} + a_1^{(1)} = a_{n+1}^{(1)}$$

Having a sequence (a_1, a_2, \dots, a_n) as an arithmetic sequence of order k we can say that the inverse sequence $(a_m, a_{m-1}, \dots, a_1)$ is also an arithmetic sequence of the same order.

From what has been said, we can say that by permuting the n elements of any sequence, as long as it is an arithmetic sequence of order k , there is at least one more sequence that is an arithmetic sequence of order k . There may be more others of the same order or not. For this phenomenon, we could not establish general rules that explain this behavior. We just looked at a few examples.

To guarantee that from the table of differences a given sequence is an arithmetic sequence of order k we only need $k+1$ terms.

If more or less terms are used, the sequence may or may not be a higher-order arithmetic sequence with an order greater or less than k .

Finally, other properties and aspects of lesser relevance were also studied during the research. So the above can be seen as the most important.

4 CONCLUSIONS

In addition to the results that we already expected to obtain, which refer to the general term and the formula for the sum of the first n terms, we were able to obtain conclusions that show the existence of higher order arithmetic sequences and lead us to perceive their application in several problems.

Equations (2) and (3) were demonstrated by studying the properties described in [5] and by the induction techniques presented in [4]. Equation (2), the general term formula of a higher-order arithmetic sequence, can be used either to find any n term of the sequence or the polynomial that describes this sequence. Thus, a conclusion that could be obtained was that the polynomial has the same degree of the sequence it describes, that is, both have degree k .

When we refer to equation (3), the formula for the sum of the first n terms of an arithmetic sequence of higher order, we can conclude that the degree of the polynomial that generates it is a degree greater than that of the sequence, that is, the sequence has degree k and the sum polynomial has degree $k+1$.

We were unable to demonstrate and obtain general cases for some phenomena, which leaves room for other researchers to continue to develop this theme and establish rules and explanations for the behavior of these phenomena.

REFERENCES

- [1] IEZZI, Gelson et al. **Mathematics - Single Volume**. São Paulo: Current, 2007.
- [2] KLANDRES, Lars; KRIS, Jamsa. **Programming in C/C++: "The Bible"**. São Paulo: Makron Books do Brasil Ltda, 1999.
- [3] LOPES, Luis. **Progression Handbook**. Rio de Janeiro: Interciencia, 1998.
- [4] MORGADO, Jose. Induction and Mathematical Induction. **SPM Bulletin No. 17 of June 1990**. Available at: < http://nautilus.fis.uc.pt/bspm/revistas/17/023_034.150.pdf > . Accessed on: 07 Nov. of 2011.
- [5] RODRIGUES, FW **Newton's Binomial and the Venus de Milo**. RPM 30, SBM, 1996, pgs. 34-41.
- [6] RUGGIERO, Marcia A. Gomes; LOPES, Vera Lucia da Rocha. **Calculation Numerical: Theoretical and Computational Aspects**. Sao Paulo: McGraw-Hill, 1998.

Appendix

```
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

// This program must receive a list of numeric parameters (n, x, y, z, ..., w)
// where n is an integer and (x, y, z, ..., w) we have them from an AP,
// printing the Arithmetic Progression order, the nth term and the sum of the
// n first terms of Arithmetic Progression.

//Function that calculates a binomial number:
double binomial( int number1 , int number2 ) {
    if ( number2 > number1 ) return 0;
    else if ( numero2 == 1) return numero1;
    else if ( numero2 == 0) return 1;
    else if ( numero2 == numero1) return 1;
    else return binomio(numero1 - 1, numero2) + binomio(numero1 - 1, numero2 - 1);
}

//Function that calculates the elements of the difference table:
double seq( double * vector , int line , int column ) {
    if ( line == 0) return vector [ column ];
    else return seq( vector , row - 1, column + 1) - seq( vector , row - 1, column );
}

//Function that finds the Arithmetic Progression order:
int order( double * vector , int dim ) {
    int i;
    for ( i = 0; i < dim ; i++)
        if (seq( vector , dim - i, 0) != 0) break ;
    return dim - i;
}

//Function that calculates the nth term of an Arithmetic Progression:
double term( double * vector , int dim , int number ) {
    double x = 0;
    int i;
    for ( i = 0; i <= dim ; i++)
        x = x + binomial( number - 1, i) * seq( vector , i, 0);
    return x;
}

// Function that determines the sum of the first k terms of an Arithmetic Progression:
double sum( double * vector , int dim , int number ) {
    double x = 0;
    int i;
    for ( i = 0; i <= dim ; i++)
        x = x + binomial( number , i + 1) * seq( vector , i, 0);
    return x;
}

//main program:
int main( int argc , char * argv [ ] ) {
    //Configuration of local Language and accent parameters:
    setlocale( LC_ALL , "Portuguese" );
    //For the program to continue, the user must enter at least two numbers as parameters:
    if ( argc > 2) {
        double * vector;
        int i;

        //Dynamic memory allocation for Arithmetic Progression2 terms:
        vector = ( double *)malloc(( argc -1) * sizeof ( double ));
        //Filling a vector with Arithmetic Progression terms:
        for ( i = 0; i < argc - 2; i++)
            vector[i] = atof( argv [i + 2]);
        //Printing the results:
        i = atoi( argv [1]);
        printf( "\t\tHIGH ORDER ARITHMETIC SEQUENCES\n\n" );
        printf( "\tArithmetic Progression of order %d.\n" , order(vector, argc - 3));
        printf( "\t%d-th term of AP: %.2f\n" , i, term(vector, argc - 3, i));
        printf( "\tSum of first %d terms of AP: %.2f\n" , i, sum(vector, argc - 3, i));
        //Release dynamically allocated memory:
        free(vector);
    }
    else printf( "Enter at least two numeric parameters!\n" );
    return 0;
}
```