Polynomial regression analysis

“Least square regression analysis to fit points to a polynomial of degree m”

Brief description:

This work is based on the realization of a program to perform a least square regression analysis and in this way to generate a series of values to solve a polynomial of unknown degree "m", to finally obtain the polynomial. In this way, the program is general and depends on the type of polynomial to be calculated, so it is adapted in a generic way.

On the other hand, it is necessary to talk about the language used. In this case, Fortran has been used, which is a language used for mathematical calculation and has a great variety of applications for scientific interest.

Throughout this work we will explain the algorithm used (method) and then all the parts of the code used, together with certain results used to verify its viability.

Algorithm:

The objective of polynomial regression is to model a nonlinear relationship between the independent and dependent variables (technically, between the independent variable and the conditional mean of the dependent variable). In this way, a set of points is generated which give a solution to the polynomial in question (Fig. 1).

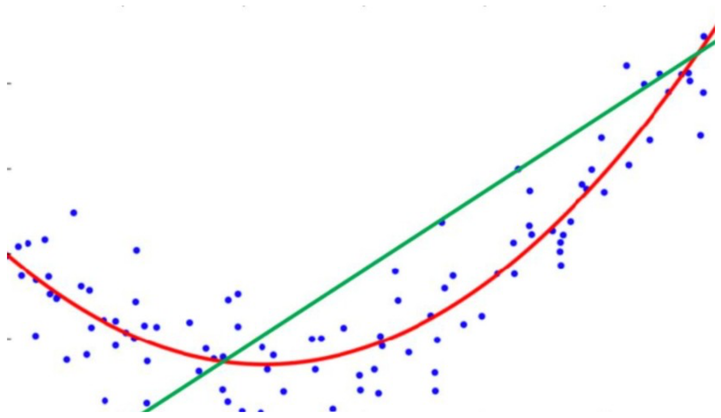


Fig. 1: *Example of a polynomial regression*

The calculation in the case of this work has been carried out from three fundamental steps: the reading of the points given by the user in order to generate the polynomial in question, the generation of the extended matrix by calculating the necessary summations for the polynomial of degree m and finally the resolution of the linear system using the procedure of Gaussian elimination with pivoting and back-substitution.

At the end, the created codes have been produced using the programming language called Fortran, which is a language used for mathematical calculation and, at the same time, very used in the scientific field. Throughout this work we define the method previously mentioned and, at the same time, the type of code used, in order to understand how it works. The whole set of variables and structures are explained together with the reason for the use of each one.

Implementation of the code:

This program is divided into nine distinct parts, the first six parts are defined as subroutines, which have a specific function, followed by the general program, which generates and organizes the entire calculation, and finally, the function of the polynomial of degree m is defined (Fig. 2):

$$y = a_m x^m + a_{m"1} x^{m"1} + \cdots + a_1 x + a_{\$} x^{\$} + a_{\%}$$

Fig. 2: *Least square regression polynomial of degree m (equation)*

The program starts with the subroutine pertaining to the reading of data with which the regression is performed. In this case the data must be located in columns, since the do loop used is created using the read command so that the variable j in each iteration reads each row as two data.

Next, the subroutine used for the creation of the extended matrix for the calculation of the polynomial in question is introduced. First the dimensions of the matrix are defined and then a do loop is started with another loop inside it, which moves through the matrix changing the values of j i k which correspond to the coefficients of each term of the matrix. Then at each of the points the summation corresponding to the corresponding degree is performed to find the value of the term in each position. Finally, the last column is accumulated for each row from a do loop.

The following subroutine corresponds to the resolution of the linear system. It is based on the resolution of the linear system from the application of Gaussian elimination, which reduces the matrix to the row echelon form, in order to solve it. First, two do loops are created to go through all the positions within the matrix and then a constant multiplier is sought for the subtraction between rows. Then from the formula explained in the code we eliminate certain rows in the calculation to finally find the divisor and to be able to equal elements to 1. Then we introduce two more do loops to introduce values equal to zero in the left pivots, the procedure is very similar to the previous one described.

The last two routines that are missing before the main program belong to the calculation of the parameter R2, to determine the feasibility of the obtained fitting and the second one belongs to the creation of ouputs to obtain the coefficients of the polynomial itself and at the same time the data to plot the obtained function. In this case the corresponding definitions are used for the calculation of R2, using two variables which belong to the sum of squared residuals and the total sum of squares.

The main program, which organizes the whole global procedure of the calculation, is based on calling the subroutines explained above and, at the same time, writing the results produced at the end of the calculation. It is worth mentioning

It should be noted that during the process the user is asked to enter a name for each of the outputs and also the degree of the polynomial (m) to be calculated.

Finally, the function to determine each value corresponding to the coefficients of the polynomial plotted is defined.

Discussion of results (outputs):

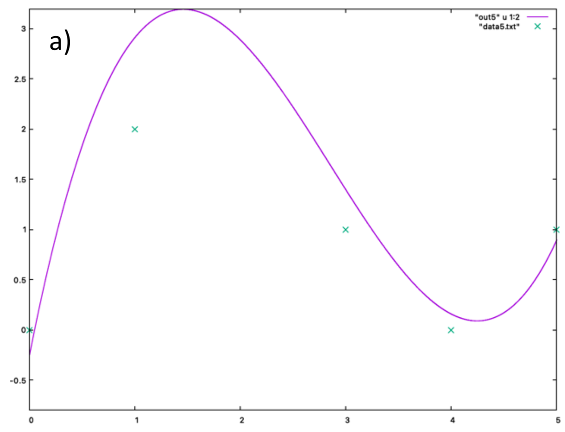
In the last section of the work, a series of calculations have been carried out with previously given values of different points. For each of the sets of points, the corresponding polynomials were plotted and the R2 parameter was calculated (Table 1). In this way, the feasibility of the program has been observed.

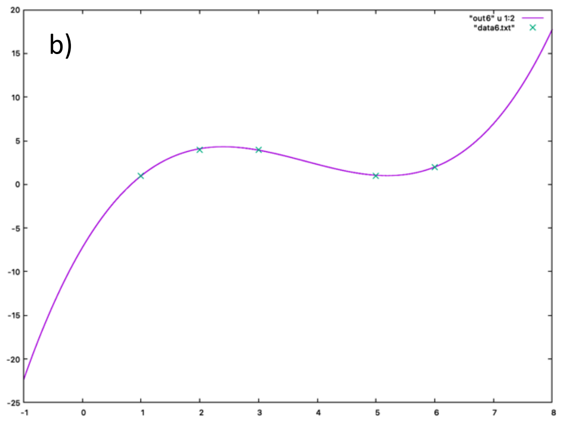
On the other hand, it is necessary to comment on the structure of the inputs. This structure must be formed by two columns in which the points of the space are defined with two variables: "x" in the first column and "y" in the second column. The inputs must be structured in this way since the file reading procedure in the subroutine is programmed in this way.

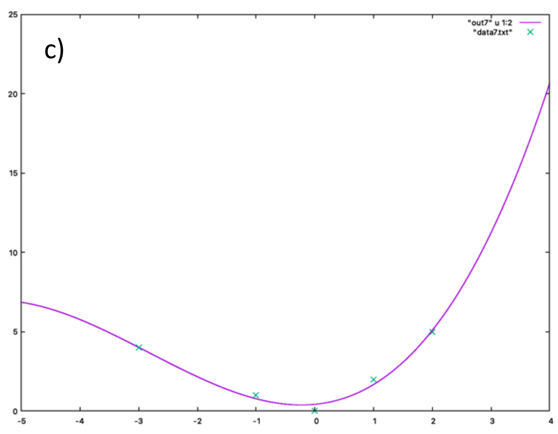
Next, we proceed to comment the results obtained in each of the exercises and see if such fitting has been correct:

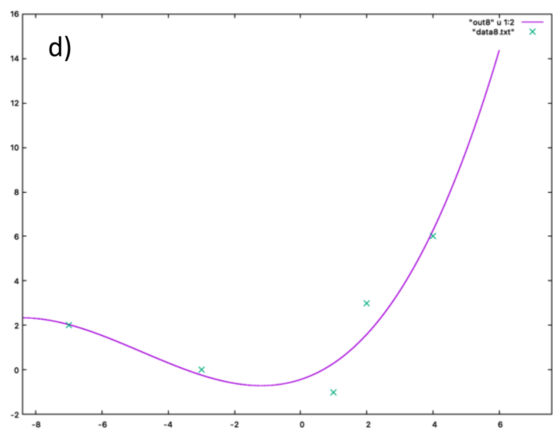
Degree:	Values:	R^2
3	5	0,79482
3	6	0,99817
3	7	0,97994
3	8	0,87387
2	9	0,99514

Table 1: *R^2 values for each fitting (5 to 9)*









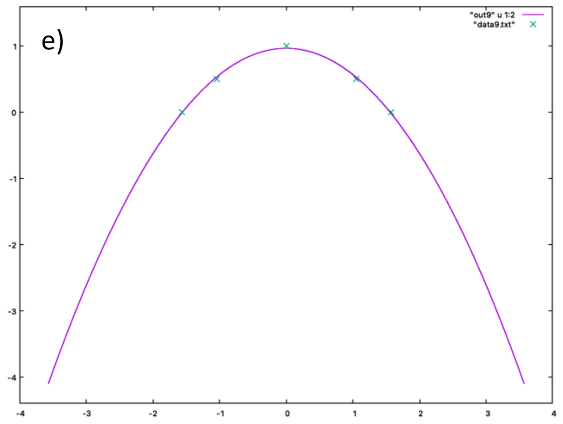


Fig. 3: *Plots and values for different tests a(5), b(6), c(7), d(8), e(9)*

It can be observed that the values of R2 (Table 1) for the calculated polynomials is very close to 1 except for cases 5 and 8. It should be noted that all calculations have not given any errors and the corresponding graphs (Fig 3) are optimally adjusted to the values of the given points, although if the points differ much from the general trend, they give a not so correct approximation of the polynomial, a larger number of points would be needed. As a conclusion it can be said that the program works correctly and does not give any errors.