Second order polynomial fit using least squares method and Rstudio

Claudio H. González-Rojas, PhD.
Laboratorio de Computación Cuántica, Melipilla,
P.O. Box 9580000, RM, Chile, chgonzalezr@academicos.uta.cl

ABSTRACT

A routine written in R language, in Rstudio environment, is presented. The routine was developed to solve an old problem addressed in the literature: fitting a set of data to a second-degree polynomial function using the least squares method. The criterion used to qualify the goodness of fit was the root mean square statistic of the residuals, i.e. RMSE. The error made was found to be reasonable.
The problem arises when it comes to instructing students how these systems work since using calculation routines either in R or Python could lead to the use to using a black box so that the interested does not understand how the method of adjustment proceeds. Moreover, the author of this work has reviewed statistical texts and mathematics notes from other universities and has found that they all reach the conventional canonical normal equation:

$$a+b\langle x\rangle+c\langle x^2\rangle=\langle y\rangle$$
$$a\langle x\rangle+b\langle x^2\rangle+c\langle x^3\rangle=\langle xy\rangle$$
$$a\langle x^2\rangle+b\langle x^3\rangle+c\langle x^4\rangle=\langle x^2y\rangle$$

Starting from

$$\sigma^2=\frac{1}{N}\sum_i\left(y_i-\left(a+bx_i+cx_i^2\right)\right)^2$$

However, at the resolution stage, they are reduced to a specific numerical problem, and hence their numerical solution at the resolution stage, but the reader misses must take advantage of the opportunity to appreciate a general solution. This work proposes a replacement of the canonical equation notation, its solution using the matrix method, and its explicit algebraic resolution, then writing the solution by means employing routine developed in Rstudio accompanied with convenient data tests.

keywords: second order polynomial fit Rstudio

INTRODUCTION

A linear relationship between two variables x and y is one of the most common, effective and easy assumptions to make when trying to figure out their relationship. Sometimes however, the true underlying relationship is more complex than that, and this is when polynomial regression comes in to help. For example, if you have a data set and you need to fit a polynomial empirical function of order two, it is possible to see in the literature that there are many ways to solve this problem. One of them is through R language[1]. Another way could be using Phyton [2].

The literature has generally found that least squares methods [3] are a prevalent and accepted resource, especially in the linear case. Some of the merits and demerits found in this regard are the following:

Merits

1. The method is mathematically sound.
2. The estimates *a* and *b* are unbiased.
3. The least square method gives trend values for all the years, and it is devoid of all kinds of subjectivity.
4. The algebraic sum of deviations of actual values from trend values is zero, and the sum of the deviations $\sum(y-y_c)^2$ is minimum.

Demerits

1. The least square method is highly mathematical, making it difficult for a non-specialist to understand it.
2. The method needs to be more flexible. If specific new values are included in the given time series, the values of *n*, $\sum x$, $\sum y$,$\sum x2$, and $\sum xy$ would change. Which affects the trend values.
3. It has been assumed that y is only a linear function of time x, which may not be accurate in many situations

A problem arises when trying to explain to students how these systems work since doing so using calculation routines in R or Phyton language could lead to a black box approach so that the interested party does not understand how the adjustment method proceeds. Therefore, a review of the polynomial fitting method has been proposed, using the least squares methodology for the nonlinear case, taking the standard equation to matrix methods, solving its solutions explicitly, writing the solution routine in R language, and finally, making a couple of tests of the algorithm developed in Rstudio.

THEORETICAL FRAMEWORK

The problem to be solved is finding a set of equation-minimizing parameters.

$$\sigma^2 = \frac{1}{N}\sum_i \left(y_i - \left(a + bx_i + cx_i^2\right)\right)^2$$

For these purposes, the fitting polynomial is minimized as follows.

$$\frac{\partial\sigma^2}{\partial a} = \frac{\partial\sigma^2}{\partial b} = \frac{\partial\sigma^2}{\partial c} = 0$$

Applied successively, this leads to the following standard canonical equation:

$$a + b\langle x\rangle + c\langle x^2\rangle = \langle y\rangle$$
$$a\langle x\rangle + b\langle x^2\rangle + c\langle x^3\rangle = \langle xy\rangle$$
$$a\langle x^2\rangle + b\langle x^3\rangle + c\langle x^4\rangle = \langle x^2y\rangle$$

Whenever the following format is used:

$$\sum_{i=1}^{N} 1 = N \quad \frac{1}{N}\sum_{i=1}^{N} x_i = \langle x\rangle \quad \frac{1}{N}\sum_{i=1}^{N} y_i = \langle y\rangle \quad \frac{1}{N}\sum_{i=1}^{N} x_i^2 = \langle x^2\rangle \quad \frac{1}{N}\sum_{i=1}^{N} x_i^3 = \langle x^3\rangle \quad \frac{1}{N}\sum_{i=1}^{N} x_i^4 = \langle x^4\rangle$$

$$\frac{1}{N}\sum_{i=1}^{N} x_i y_i = \langle xy\rangle \quad \frac{1}{N}\sum_{i=1}^{N} x_i^2 y_i = \langle x^2y\rangle$$

The canonical equation converted into a matrix format would be expressed as follows:

$$\begin{bmatrix} 1 & \langle x\rangle & \langle x^2\rangle \\ \langle x\rangle & \langle x^2\rangle & \langle x^3\rangle \\ \langle x^2\rangle & \langle x^3\rangle & \langle x^4\rangle \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} \langle y\rangle \\ \langle xy\rangle \\ \langle x^2y\rangle \end{bmatrix} \qquad /1/$$

Designating A, X and Y according to

$$A = \begin{bmatrix} 1 & \langle x\rangle & x^2 \\ \langle x\rangle & \langle x^2\rangle & \langle x^3\rangle \\ \langle x^2\rangle & \langle x^3\rangle & \langle x^4\rangle \end{bmatrix} \quad X = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad Y = \begin{bmatrix} \langle y\rangle \\ \langle xy\rangle \\ \langle x^2y\rangle \end{bmatrix}$$

Eq. /1/ remains in matrix format

$$A \cdot X = Y \qquad /2/$$

Their solutions are presented below along with the computational algorithm implemented in Rstudio. Provided that A is invertible.

Goodness of fit

The goodness of fit is determined by the statistical indicator, root mean square root value or RMSE, or root mean second moment value of the residual, defined by:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{i=n}\left(\hat{y}_i - y(i)\right)^2} \qquad /3/$$

where:

$\hat{y}_i$ represents the predicted value
y(i) represents the exact value of

While we will define accuracy as the relationship:

$$precision = \left(1 - \frac{RMSE}{100}\right)\cdot 100 \text{ , \%} \qquad /4/$$

In this way we will explore with both descriptors how accurate is the fit of the polynomial function to the real data, that is, how well the real values are explained with respect to the fit function we are using as a model to represent the data.

RESULTS AND DISCUSSION

1.1.Formal development

Eq. /1/ is formally and explicitly developed in the APPENDIX. The column vector's parameters a, b, and c that solve the eq problem. /1/ are the following:

$$a = \frac{\begin{pmatrix} \langle x^2\rangle\langle x^4\rangle - \langle x^3\rangle^2 & \langle x^2\rangle\langle x^3\rangle - \langle x\rangle\langle x^4\rangle & \langle x\rangle\langle x^3\rangle - \langle x^2\rangle^2 \end{pmatrix}}{\left(\langle x^2\rangle - \langle x\rangle^2\right)\langle x^4\rangle + 2\langle x\rangle\langle x^2\rangle\langle x^3\rangle - \langle x^2\rangle^3 - \langle x^3\rangle^2} \begin{pmatrix} \langle y\rangle \\ \langle xy\rangle \\ \langle x^2 y\rangle \end{pmatrix} \qquad /5/$$

$$b=\frac{\begin{pmatrix}\langle x^2\rangle\langle x^3\rangle-\langle x\rangle\langle x^4\rangle & \langle x^4\rangle-\langle x^2\rangle^2 & \langle x\rangle\langle x^2\rangle-\langle x^3\rangle\end{pmatrix}}{\left(\langle x^2\rangle-\langle x\rangle^2\right)\langle x^4\rangle+2\langle x\rangle\langle x^2\rangle\langle x^3\rangle-\langle x^2\rangle^3-\langle x^3\rangle^2}\begin{pmatrix}\langle y\rangle\\ \langle xy\rangle\\ \langle x^2y\rangle\end{pmatrix} \quad /6/$$

$$c=\frac{\begin{pmatrix}\langle x\rangle\langle x^3\rangle-\langle x^2\rangle^2 & \langle x\rangle\langle x^2\rangle-\langle x^3\rangle & \langle x^2\rangle-\langle x\rangle^2\end{pmatrix}}{\left(\langle x^2\rangle-\langle x\rangle^2\right)\langle x^4\rangle+2\langle x\rangle\langle x^2\rangle\langle x^3\rangle-\langle x^2\rangle^3-\langle x^3\rangle^2}\begin{pmatrix}\langle y\rangle\\ \langle xy\rangle\\ \langle x^2y\rangle\end{pmatrix} \quad /7/$$

The novelty is neither the algebraic solution nor its difficulty obtaining it but its relevance as a straightforward solution. The author reviewed the literature, finding that the same standard canonical equation is observed for similar cases. Still, when solving it, the reader is invited to solve this case with a numerical problem. In this case, the well-known Gauss-Seidel elimination technique is applied (Salkuyeh, 2007) and others (Stanimirović & Petković, 2013), achieving its solution. Perhaps the optimal in front of the required numerical solution. However, its explicit resolution could be relegated to the tedious or stilted stages. But if I want to address the students' learning, perhaps emphasizing didactics will be considered relevant even if it is not optimal.

The literature has many methods for solving this type of nonlinear function minimization problem. One of the methodologies of recognized use is the Levenberg-Marquadt algorithm (Levenberg, 1944; Marquardt, 1963). In the numerical context, there are procedures routinized in Phyton (Newville et al., 2016) and R (Nash, 2022). The author thinks the innovation in the present work comes not in the way a well-known problem is solved but in the computational algorithm developed, starting from an explicit solution of a polynomial of order 2, represented in matrix format, which could serve to complement those already presented in the literature using the two previous languages. For example, for demonstration data provided in APPENDIX 2, an algorithm developed in Rstudio by the author is presented below, defined by:

```
# Polynomial fit with three parameters, using least squares method

install.packages('ggplot2')   # Install libraries
install.packages("readr")     # Install libraries
library("ggplot2")            # load library
library("readr")              # load library

# X <- c(0,1,2,3,4,5,6,7,8,9)                  # First data set
# Y <- c(1,3,5,7,9,11,13,15,17,19)
# df <- data.frame(X,Y)

# X <- c(0,1,2,3,4,5,6,7,8,9)                  # Second data set
# Y <- c(1,6,17,34,57,86,121,162,209,262)
# df <- data.frame(X,Y)

# X <- c(0,1,2,3,4,5,6,7,8,9)                  # Third data set
# Y <- c(4,9,18,31,48,69,94,123,156,262)
# df <- data.frame(X,Y)

X <- c(22,31,51,62,86,100,123,133,160,176)    # Fourth data set
```

```
Y <- c(24,29,45,63,80,100,153,175,270,300)
df <- data.frame(X,Y)

n = nrow(df)   # Determining the length of dataframe

X1 = 0     # terms order 1    # Calculus of determinant order 3
s = 0
s = sum(df$X)
X1 = s/n

X2 = 0     # terms order 2
s = sum(df$X^2)
X2 = s/n

X3 = 0     # terms order 3
s = 0
s = sum(df$X^3)
X3 = s/n

X4 = 0     # terms order 4
s = 0
s = sum(df$X^4)
X4 = s/n
XY = 0     # terms crossing
s = 0
s = sum(df$X*df$Y)
XY = s/n

Y = 0     # terms order 1 for Y
s = 0
s = sum(df$Y)
Y = s/n   # Y observed

Y2 = 0    # terms order 2 for Y
s = 0
s = sum(df$Y^2)
Y2 = s/n  # cuadratic average Y observed

X2Y = 0   # terms crossing X2Y
s = 0
s = sum(df$X^2*df$Y)
X2Y = s/n

X1^2     # cuadratic average
X2^2     # cuadratic average order 2
(X2)^3   # cuadratic average order 3
X3^2     # cubic average order 2

det = 0
det = (X2-X1^2)*X4 + 2*X1*X2*X3-X2^3-X3^2
a = 0   # a parameter
a = (X2*X4*Y - X3^2*Y + X2*X3*XY - X1*X4*XY + X1*X3*X2Y - X2^2*X2Y)/det
b = 0   # b parameter
b = (X2*X3*Y - X1*X4*Y + X4*XY - X2^2*XY + X1*X2*X2Y - X3*X2Y)/det
c = 0   # c parameter
c = (X1*X3*Y - X2^2*Y + X1*X2*XY - X3*XY + X2*X2Y - X1^2*X2Y)/det
```

```
Y <- c(df$Y)        #conversion dataframe to a vector Y

y <- 0
for (i in 1:n){
  y[i] = a+b*X[i]+c*X[i]^2
}

Table <- data.frame("X" = X,"Yobs"= Y,"ycalc" = y)

s1 = 0                    # Calculus of RMSE
s2 = 0
DIF = 0
for (i in 1:n){
  s1 = Y[i]               # calculated value
  s2 = y[i]               # observed value
  DIF = abs(s1-s2)^2
  S = DIF/n
}
RMSE = sqrt(S)
Precision=(1-RMSE/100)*100

# create regression plot with customized style

ggplot(data=Table, aes(x=X, y=Y)) + geom_point(color="red", size=3) + geo
m_line(y=y, color="blue", linewidth=1) + labs(x='Name of variable X', y='
Name of variable Y', title='Polynomial fit') + theme(plot.title = element
_text(hjust=1.0, size=20, face='bold'))

print(paste("a =", a))
print(paste("b =", b))
print(paste("c =", c))
print(paste("RMSE =", RMSE))
print(paste("Precision =", Precision))

Table
```

Data and Calculus

We are going to test four data frames. The first and second are ideal, the third contains hard data and the fourth is similar to the third.

Table I. Data set proposed in this work

	Dataset 1		Dataset 2		Dataset 3		Dataset 4	
	X	Y	X	Y	X	Y	X	Y
1	0	0	0	1	0	4	22	24
2	1	10	1	6	1	9	31	29
3	2	20	2	17	2	18	51	45
4	3	30	3	34	3	31	62	63
5	4	40	4	57	4	48	86	80
6	5	50	5	86	5	69	100	100

7	6	60	6	121	6	94	123	153
8	7	70	7	162	7	123	133	175
9	8	80	8	209	8	156	160	270
10	9	90	9	262	9	262	176	300

In following Figures, it is show a plot with the fitting curves corresponding to the presented data with use of R code designed by the author

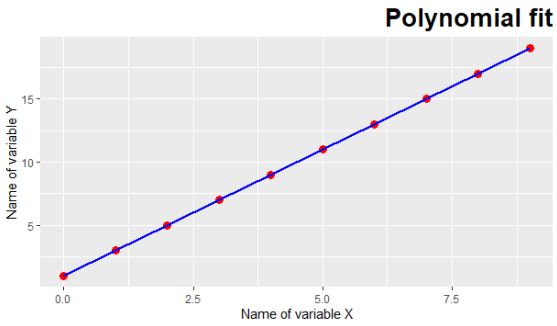

Figure 1. First dataset

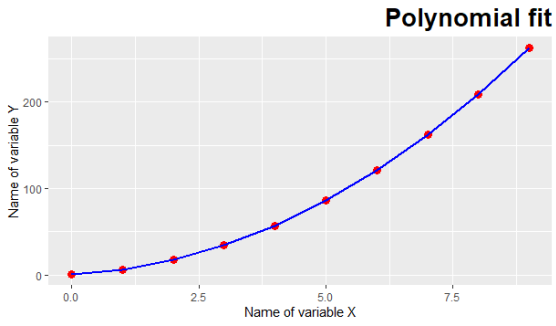

Figure 2. Second dataset

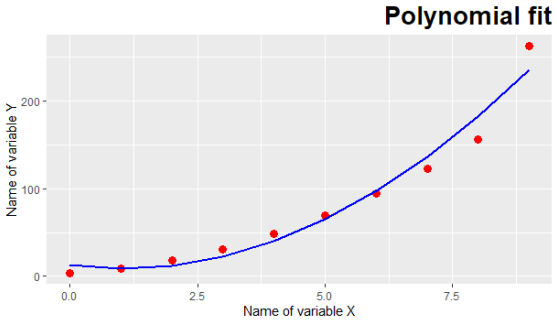

Figure 3. Third dataset

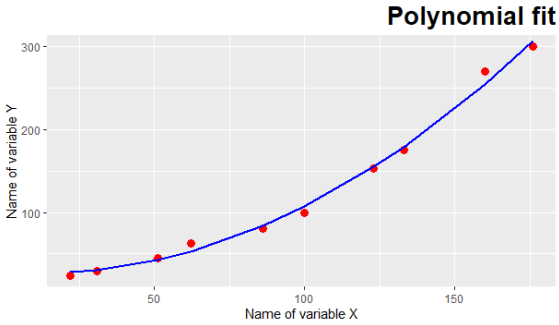

Figure 4. Fourth dataset

Once the adjustment for the present collection of data has been made, the parameters found are the following:

Data	RMSE	Precision, %	Calculated parameters			Proposed model
			a	b	c	
1	1.572853e-14	100	0.99999999	2	0	$y=1+2x$
2	9.706752e-13	100	1.00000000	2.0000000	3.0000000	$y=1+2x+3x^2$
3	8.331164	91.66884	12.7818181	-7.35000000	3.56818181	$y=12.7-0.7x+3.56x^2$
4	2.347853	97.65215	28.1790088	-0.26202628	0.010503686	$y=28.17-0.26x+0.01x^2$

CONCLUDING REMARKS

Although somewhat cumbersome, the explicit development enabled an algebraic solution in a matrix format for a problem already addressed in the literature. The solution is separate

from the central issue in this work. However, the innovation is to show its implementation in R language, independent of the classical numerical algorithms recorded in the literature.

The algorithm developed in Rstudio allowed, given a test DATA, to reach the adjustment parameters according to the initial model and calculate the goodness of fit.

The weakness of the present proposal is perhaps the variety of similar proposals found in the literature, on the one hand, and on the other hand, that the didactics remain complex for students in the idea of presenting the solution of a fitting data problem through a nonlinear function that is explainable in a simple way. Developing it could improve the students' approach and interest in numerical computation.

References and notes

1. Team, R.C., *R language definition.* Vienna, Austria: R foundation for statistical computing, 2000. **3**(1).
2. VanRossum, G. and F. Drake, *The Python Language Reference: Python software foundation Amsterdam*. 2010, Netherlands.
3. Gorry, P.A., *General least-squares smoothing and differentiation by the convolution (Savitzky-Golay) method.* Analytical Chemistry, 1990. **62**(6): p. 570-573.

1. Team, R.C., *R language definition.* Vienna, Austria: R foundation for statistical computing, 2000. **3**(1).
2. VanRossum, G. and F. Drake, *The Python Language Reference: Python software foundation Amsterdam*. 2010, Netherlands.

Gorry, P. A. (1990). General least-squares smoothing and differentiation by the convolution (Savitzky-Golay) method. *Analytical Chemistry, 62*(6), 570-573.

Levenberg, K. (1944). A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics, 2*(2), 164-168.

Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics, 11*(2), 431-441.

Nash, J. C. (2022). Function minimization and nonlinear least squares in R. *Wiley Interdisciplinary Reviews: Computational Statistics*, e1580.

Newville, M., Stensitzki, T., Allen, D. B., Rawlik, M., Ingargiola, A., & Nelson, A. (2016). LMFIT: Non-linear least-square minimization and curve-fitting for Python. *Astrophysics Source Code Library*, ascl: 1606.1014.

Salkuyeh, D. K. (2007). Generalized Jacobi and Gauss-Seidel methods for solving linear system of equations. *NUMERICAL MATHEMATICS-ENGLISH SERIES-, 16*(2), 164.

Stanimirović, P. S., & Petković, M. D. (2013). Gauss–Jordan elimination method for computing outer inverses. *Applied Mathematics and Computation, 219*(9), 4667-4679.

Team, R. C. (2000). R language definition. *Vienna, Austria: R foundation for statistical computing, 3*(1).

VanRossum, G., & Drake, F. L. (2010). *The python language reference*: Python Software Foundation Amsterdam, Netherlands.

Acknowledgment

The author acknowledges ***Laboratorio de Computación Cuántica*** *for dedicating time and lending the facilities during the development of the present work.*

APPENDIX 1

Resolution of the matrix equation

$$A \cdot X = Y$$

Con

$$A = \begin{bmatrix} 1 & \langle x \rangle & x^2 \\ \langle x \rangle & \langle x^2 \rangle & \langle x^3 \rangle \\ \langle x^2 \rangle & \langle x^3 \rangle & \langle x^4 \rangle \end{bmatrix} \quad X = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad Y = \begin{bmatrix} \langle y \rangle \\ \langle xy \rangle \\ \langle x^2 y \rangle \end{bmatrix}$$

First, solve the column vector X containing the unknowns by multiplying the reduced matrix equation by the inverse matrix of A on the left-hand side of the reduced matrix equation.

$$A^{-1}AX = A^{-1}Y$$

$$IX = A^{-1}Y$$

Which equals to

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = A^{-1} \begin{bmatrix} \langle y \rangle \\ \langle xy \rangle \\ \langle x^2 y \rangle \end{bmatrix}$$

Now, the inverse matrix A is calculated, provided that A is invertible, according to,

$$A^{-1} = \frac{adjA}{detA}$$

The determinant of A

$$\det A = \begin{vmatrix} 1 & \langle x \rangle & \langle x^2 \rangle \\ \langle x \rangle & \langle x^2 \rangle & \langle x^3 \rangle \\ \langle x^2 \rangle & \langle x^3 \rangle & \langle x^4 \rangle \end{vmatrix}$$

This is solved using the Sarrus Rule, in which case:

$$\det A = \left(\langle x^2 \rangle - \langle x \rangle^2 \right) \langle x^4 \rangle + 2 \langle x \rangle \langle x^2 \rangle \langle x^3 \rangle - \langle x^2 \rangle^3 - \langle x^3 \rangle^2$$

Meanwhile, the adjunct of A is calculated according to:

$$adjA = \begin{bmatrix} \langle x^2\rangle\langle x^4\rangle - \langle x^3\rangle^2 & \langle x^2\rangle\langle x^3\rangle - \langle x\rangle\langle x^4\rangle & \langle x\rangle\langle x^3\rangle - \langle x^2\rangle^2 \\ \langle x^2\rangle\langle x^3\rangle - \langle x\rangle\langle x^4\rangle & \langle x^4\rangle - \langle x^2\rangle^2 & \langle x\rangle\langle x^2\rangle - \langle x^3\rangle \\ \langle x\rangle\langle x^3\rangle - \langle x^2\rangle^2 & \langle x\rangle\langle x^2\rangle - \langle x^3\rangle & \langle x^2\rangle - \langle x\rangle^2 \end{bmatrix}$$

This leaves us with the inverted matrix A:

$$A^{-1} = \frac{\begin{bmatrix} \langle x^2\rangle\langle x^4\rangle - \langle x^3\rangle^2 & \langle x^2\rangle\langle x^3\rangle - \langle x\rangle\langle x^4\rangle & \langle x\rangle\langle x^3\rangle - \langle x^2\rangle^2 \\ \langle x^2\rangle\langle x^3\rangle - \langle x\rangle\langle x^4\rangle & \langle x^4\rangle - \langle x^2\rangle^2 & \langle x\rangle\langle x^2\rangle - \langle x^3\rangle \\ \langle x\rangle\langle x^3\rangle - \langle x^2\rangle^2 & \langle x\rangle\langle x^2\rangle - \langle x^3\rangle & \langle x^2\rangle - \langle x\rangle^2 \end{bmatrix}}{\left(\langle x^2\rangle - \langle x\rangle^2\right)\langle x^4\rangle + 2\langle x\rangle\langle x^2\rangle\langle x^3\rangle - \langle x^2\rangle^3 - \langle x^3\rangle^2}$$

Solving the parameters a,b and c from the equation:

$$IX = A^{-1}Y$$

The following can be reached

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = A^{-1}\begin{bmatrix} \langle y\rangle \\ \langle xy\rangle \\ \langle x^2y\rangle \end{bmatrix}$$

and operate A^{-1} to the right, it is possible to get to:

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \frac{\begin{bmatrix} \langle x^2\rangle\langle x^4\rangle - \langle x^3\rangle^2 & \langle x^2\rangle\langle x^3\rangle - \langle x\rangle\langle x^4\rangle & \langle x\rangle\langle x^3\rangle - \langle x^2\rangle^2 \\ \langle x^2\rangle\langle x^3\rangle - \langle x\rangle\langle x^4\rangle & \langle x^4\rangle - \langle x^2\rangle^2 & \langle x\rangle\langle x^2\rangle - \langle x^3\rangle \\ \langle x\rangle\langle x^3\rangle - \langle x^2\rangle^2 & \langle x\rangle\langle x^2\rangle - \langle x^3\rangle & \langle x^2\rangle - \langle x\rangle^2 \end{bmatrix}}{\left(\langle x^2\rangle - \langle x\rangle^2\right)\langle x^4\rangle + 2\langle x\rangle\langle x^2\rangle\langle x^3\rangle - \langle x^2\rangle^3 - \langle x^3\rangle^2} \begin{bmatrix} \langle y\rangle \\ \langle xy\rangle \\ \langle x^2y\rangle \end{bmatrix}$$

Whose explicit solutions for parameter a are:

$$a=\frac{\begin{pmatrix}\langle x^2\rangle\langle x^4\rangle-\langle x^3\rangle^2 & \langle x^2\rangle\langle x^3\rangle-\langle x\rangle\langle x^4\rangle & \langle x\rangle\langle x^3\rangle-\langle x^2\rangle^2\end{pmatrix}}{\left(\langle x^2\rangle-\langle x\rangle^2\right)\langle x^4\rangle+2\langle x\rangle\langle x^2\rangle\langle x^3\rangle-\langle x^2\rangle^3-\langle x^3\rangle^2}\begin{pmatrix}\langle y\rangle\\ \langle xy\rangle\\ \langle x^2y\rangle\end{pmatrix}$$

$$a=\frac{\left(\langle x^2\rangle\langle x^4\rangle-\langle x^3\rangle^2\right)\langle y\rangle+\left(\langle x^2\rangle\langle x^3\rangle-\langle x\rangle\langle x^4\rangle\right)\langle xy\rangle+\left(\langle x\rangle\langle x^3\rangle-\langle x^2\rangle^2\right)\langle x^2y\rangle}{\left(\langle x^2\rangle-\langle x\rangle^2\right)\langle x^4\rangle+2\langle x\rangle\langle x^2\rangle\langle x^3\rangle-\langle x^2\rangle^3-\langle x^3\rangle^2}$$

$$a=\frac{\left(\langle x^2\rangle\langle x^4\rangle\langle y\rangle-\langle x^3\rangle^2\langle y\rangle\right)+\left(\langle x^2\rangle\langle x^3\rangle\langle xy\rangle-\langle x\rangle\langle x^4\rangle\langle xy\rangle\right)+\left(\langle x\rangle\langle x^3\rangle\langle x^2y\rangle-\langle x^2\rangle^2\langle x^2y\rangle\right)}{\left(\langle x^2\rangle-\langle x\rangle^2\right)\langle x^4\rangle+2\langle x\rangle\langle x^2\rangle\langle x^3\rangle-\langle x^2\rangle^3-\langle x^3\rangle^2}$$

For parameter B are:

$$b=\frac{\begin{pmatrix}\langle x^2\rangle\langle x^3\rangle-\langle x\rangle\langle x^4\rangle & \langle x^4\rangle-\langle x^2\rangle^2 & \langle x\rangle\langle x^2\rangle-\langle x^3\rangle\end{pmatrix}}{\left(\langle x^2\rangle-\langle x\rangle^2\right)\langle x^4\rangle+2\langle x\rangle\langle x^2\rangle\langle x^3\rangle-\langle x^2\rangle^3-\langle x^3\rangle^2}\begin{pmatrix}\langle y\rangle\\ \langle xy\rangle\\ \langle x^2y\rangle\end{pmatrix}$$

$$b=\frac{\left(\langle x^2\rangle\langle x^3\rangle-\langle x\rangle\langle x^4\rangle\right)\langle y\rangle+\left(\langle x^4\rangle-\langle x^2\rangle^2\right)+\left(\langle x\rangle\langle x^2\rangle-\langle x^3\rangle\right)\langle x^2y\rangle}{\left(\langle x^2\rangle-\langle x\rangle^2\right)\langle x^4\rangle+2\langle x\rangle\langle x^2\rangle\langle x^3\rangle-\langle x^2\rangle^3-\langle x^3\rangle^2}$$

$$b=\frac{\left(\langle x^2\rangle\langle x^3\rangle\langle y\rangle-\langle x\rangle\langle x^4\rangle\langle y\rangle\right)+\left(\langle x^4\rangle\langle xy\rangle-\langle x^2\rangle^2\langle xy\rangle\right)+\left(\langle x\rangle\langle x^2\rangle\langle x^2y\rangle-\langle x^3\rangle\langle x^2y\rangle\right)}{\left(\langle x^2\rangle-\langle x\rangle^2\right)\langle x^4\rangle+2\langle x\rangle\langle x^2\rangle\langle x^3\rangle-\langle x^2\rangle^3-\langle x^3\rangle^2}$$

And finally for parameter C are:

$$c=\frac{\begin{pmatrix}\langle x\rangle\langle x^3\rangle-\langle x^2\rangle^2 & \langle x\rangle\langle x^2\rangle-\langle x^3\rangle & \langle x^2\rangle-\langle x\rangle^2\end{pmatrix}}{\left(\langle x^2\rangle-\langle x\rangle^2\right)\langle x^4\rangle+2\langle x\rangle\langle x^2\rangle\langle x^3\rangle-\langle x^2\rangle^3-\langle x^3\rangle^2}\begin{pmatrix}\langle y\rangle\\ \langle xy\rangle\\ \langle x^2y\rangle\end{pmatrix}$$

$$c=\frac{\left(\langle x\rangle\langle x^3\rangle-\langle x^2\rangle^2\right)\langle y\rangle+\left(\langle x\rangle\langle x^2\rangle-\langle x^3\rangle\right)\langle xy\rangle+\left(\langle x^2\rangle-\langle x\rangle^2\right)\langle x^2y\rangle}{\left(\langle x^2\rangle-\langle x\rangle^2\right)\langle x^4\rangle+2\langle x\rangle\langle x^2\rangle\langle x^3\rangle-\langle x^2\rangle^3-\langle x^3\rangle^2}$$

$$c=\frac{\langle x\rangle\langle x^3\rangle\langle y\rangle-\langle x^2\rangle^2\langle y\rangle+\langle x\rangle\langle x^2\rangle\langle xy\rangle-\langle x^3\rangle\langle xy\rangle+\langle x^2\rangle\langle x^2y\rangle-\langle x\rangle^2\langle x^2y\rangle}{\left(\langle x^2\rangle-\langle x\rangle^2\right)\langle x^4\rangle+2\langle x\rangle\langle x^2\rangle\langle x^3\rangle-\langle x^2\rangle^3-\langle x^3\rangle^2}$$

APPENDIX 2

```
X,Y
0,4
1,9
2,18
3,31
4,48
5,69
6,94
7,123
8,156
9,193
10,234
```