Project Management Development – Practice and Perspectives
International Scientific Conference on Project Management in the Baltic Countries
Riga, University of Latvia
e-ISSN 2501-0263

Applicability of Multivariant Linear Optimization for Project Process Relevance Modeling

Rosenberger Philipp, FH Campus Wien, Austria
Tick József, Óbuda University, Hungary

Abstract

This article evaluates the applicability of linear regression method for optimization of PMBOK project process relevance factors.

In a first step, published in the article "Relevance of PMBOK v6 Processes for Tailored Agile Project Categories" [ROS19] published at IEEE 13th International Symposium on Applied Computational Intelligence in Timisoara Romania May 29-30, initial relevance factors of all PMBOK version 6 project processes have been postulated, based on scientific literature coverage. These presented relevance factors presented themselves as highly heterogenic. In other words, some project processes seemed to be more critical and complex to manage then others.

Nevertheless, the distribution of these factors is highly individual and until today only based on scientific literature and not on real project management practice. As next step, statistically and mathematically modelling needs to follow, enabling IT project managers to evaluate their own process relevance experience and therefore delivering input data sets for a mathematical optimization model increasing overall project health and success. As a first step towards this goal, this article evaluates the suitability of multivariate linear regression methods for such modelling purposes. Based on a dimensionally reduced example set of data it is proven, that linear models are not usable in this system. They do not reflect the needed balance in covariance of all PMBOK project processes for project success. In addition to this proven result of non-applicability of linear optimization models, the article proposes non-parametric kernel-based optimization as a possible non-linear solution. Furthermore, the requirements of using neuronal network-based modelling and optimization are discussed.

Key words: *SCRUM, IT-Project Management, Agile, PMBOK*
JEL code: M15 (IT-Management)

Introduction

Professional project managers try to standardize and optimize their work by applying project management frameworks in their projects. These frameworks and certifications like the project management body of knowledge (PMBOK) of PMI Organization [PMI17] or PRINCE2 framework of AXELOS [PRI17] Organization provide structure and a collection of project processes to be performed by project managers. Often, these frames give an indication about what to do in a project, but at the same time, do not indicate how much to do certain activities in a project. This factor can make management of agile developed IT project challenging. [PIC07] PMBOK certification just mentions, that it is the responsibility of an experienced project manager to choose the right processes and where to focus on, while often integrating agile practices [WEN16]. PMI organization structures the 49 project processes in different ways like knowledge areas or project phases and thereby gives an indication, which processes to use when in the project, but lacks a clear indication about a specific distribution of relevance. Of course, one could argue,

e-ISSN 2501-0263

that every project is different and unique and therefore relevance of project processes is unique in every project but providing an average value for indication could provide a valuable indication for project managers. In a nutshell: Project frameworks may be well defined regarding what to do but lack information regarding how much to do which activities. This lack of information where to focus on, defines the underlying problem for this research project, of which testing of linear optimization, although only part of the problem-solving process, is the focus of this article.

Basis and approach for this research

Discovering "how much to do" certain activities to optimize project success is the overall goal of this research approach. To achieve this goal a combination of data collection and mathematical optimization will be applied.

Data Collection:

Currently in development, a slim android app will collect data from IT project managers. These practitioners will be asked about their distribution of process relevance in their specific project phases. For example: Do you spend 80% of your focus and work in the project on the PMBOK process "Risk Management Execution"? And the rest on "Planning Stakeholder Management"? Ignoring all other processes? If not, which seems likely, how do you distribute your work and focus in the project? And how successful and healthy is your current project right now?

By asking project managers about their project process specific relevance factors and asking about their current project health, we will get data sets of project process relevance in relation to overall project success. These data sets will have the following form (Note: The data values in bellows table are just dummy data values)

Dataset Upload Number	ProcessNo1	ProcessNo2	ProcessNo3	...	....	ProcessNo46	ProcessNo47	ProcessNo48	ProcessNo49	Project Health Indicator
1	14,8	6,6	2,2	...	...	2,2	0,4	0,4	0,4	2,8
2	15,05	6,1	3,3	...	...	2,0	0,2	1,0	1,0	2,3
3	14,02	8,4	3,3	...	...	5,6	1,5	2,0	1,0	2,2
4	0,3	14,8	8,1	...	...	2,1	0,3	0,4	0,3	2,0
5	14,2	6,8	1,9	...	...	3,5	0,1	2,2	0,1	2,8
6	0,1	0,1	2,5	...	...	4,4	0,8	1,3	1,3	2,6
7	8,7	2,5	1,0	...	...	0,7	0,6	1,3	3,6	2,8
8	2,2	14,8	6,6	...	...	2,2	0,4	0,4	0,4	2,1
9	14,1	6,1	2,0	...	...	2,0	0,2	1,0	0,9	2,9
10	13,95	8,4	1,0	...	...	5,6	1,5	2,0	1,0	2,1
11	14,8	8,1	2,1	...	...	2,1	0,3	0,3	0,4	2,4
12	14,25	6,8	1,9	...	...	3,5	0,1	2,2	5,0	2,7
13	12,52	7,7	1,9	...	...	4,4	0,8	1,3	2,0	2,1
14	8,7	2,5	3,0	...	...	7,0	5,8	7,2	2,7	3,2
15	12,0	0,4	0,4	...	...	2,2	0,4	0,4	2,2	1,7
16	0,2	0,2	0,2	...	...	2,7	0,2	0,2	2,0	2,4
17	0,0	0,0	0,0	...	...	3,0	0,0	0,0	0,8	2,6
18	0,3	0,3	2,1	...	...	0,3	1,2	0,3	2,1	3,1
19	0,1	0,1	0,1	...	...	1,0	1,2	0,7	1,9	2,4

Table 1: ***Dummy data set for optimization***

e-ISSN 2501-0263

Basic conditions for data collection:

- Sum of process factors No.1 to No.49 = 100
- Project Health Indicator <= 4
- Minimal value of each process shall be 0,5
- All values >=0
- One single process <=76

Collected data sets can then be used to mathematically define an optimally distributed process relevance solution. An initial proposal of such a process relevance distribution is already proposed in the IEEE article "Suitability of PMBOK 6th edition for agile-developed IT Projects" [ROS18] . We shall now assume that the data collection has already been finalised and the optimization can be started. Goal of this current research step and focus of this article is to answer the question, whether linear least square optimization method [ALE10] is generally applicable to optimize the dummy data of table 1. To answer this question a three-step approach is applied. After a brief introduction to least square regression method and optimization, three optimization attempts with increasing complexity will demonstrate the applicability of the optimization method.

Introduction to multivariate regression.

The multivariate linear regression approach can be described mathematically the following way:

Equation of linear regression:

$$\mathbf{y = m*x + c} \qquad [1.]$$

where:

- y is the output of the model
- x is the independent variable
- m is the slope of the regression line
- c is the intercept

x and **y** values now get measured by data collection and we try to estimate m and c as best as possible to predict **y** for any given input **x**.

As we use multiple dependent variables, a matrix representation of linear regression is needed to describe the model more compact.

Using matrix notation the equation for linear regression looks like that:

$$\mathbf{Y = X\beta + e} \qquad [2.]$$

where:

- β is the matrix of parameters
- X is the matrix of measured values of these parameters
- e is a factor of predicted error for each measurement in contrast to the the predicted value (we want to minimize that!)

Changing the equation for e results in this equation:

$$\mathbf{e = Y - X\beta} \qquad [3.]$$

so **e** is a function of parameters **β.** Now we want to get the mean square error (MSE) in Matrix form as basis for optimization.

We get MSE be adding all squares of e from all measurements and divide them by the number of observations. In mathematical notation:

$$MSE = \frac{1}{n}\sum_{i=1}^{i=n} e_i^2$$

$$\sum_{i=1}^{i=n} e_i^2 = e_1^2 + e_2^2 + e_3^2 + e_4^2 + ... + e_n^2$$

$$= [e_1\ e_2\ e_3\ e_4\ \dots\ e_n] \times \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ . \\ . \\ . \\ e_n \end{bmatrix} = e^T e \qquad [4.]$$

Now, we can replace **e** with the equation **Y — Xβ** giving us MSE in such a form:

$$\boldsymbol{MSE = \frac{1}{n}(Y - X\beta)^T(Y - X\beta)} \qquad [5.]$$

After some expansion and replacing of that equation, we get for MSE the following equation:

$$MSE = \frac{1}{n}(Y^T Y - 2\beta^T X^T Y + \beta^T X^T X\beta) \qquad [6.]$$

This function is used as objective function in the optimization problem. Most algorithms will then, minimize this function. To get the best parameters in the model. To do this, gradients need to be estimated for gradient descent optimization.

The gradient of MSE is noted as such:

$$\nabla MSE = \frac{1}{n}(\nabla Y^T Y - 2\nabla \beta^T X^T Y + \nabla \beta^T X^T X \beta) \quad [7.]$$

where ∇ is the differential operator for the gradient.

By applying matrix differentiation, we get the following two equations for optimization:

$$= \frac{1}{n}(0 - 2X^T Y + 2X^T X\beta)$$

$$= \frac{2}{n}(X^T X\beta - X^T Y) \quad [8.]$$

The second equation is called Jacobian matrix and is used together with a learning rate (**lr**) to gradually update model parameter.

$$J(\beta) = \frac{2}{n}(X^T X\beta - X^T Y) \quad [9.]$$

The gradient descent method now iteratively updates model parameters by applying the Jacobian matrix and the learning rate:

$$\beta new = \beta old - lr \times J(\beta) \quad [10.]$$

βold is an initial starting point, that needs to be defined to start. This initial vector then gets updated with each iteration. This happens again and again until the **MSE** value gets reduced and becomes flat. **lr**, the learning rate can be seen as the step size for each iteration and shall help to prevent "shooting over the optimum " in other words, the lowest point of the error curve/surface. So by a minimization of the error and collecting new **β** values, we can define a optimum regression solution. [PRA19]

As the basic approach for least square regression and optimization is now clarified, we can apply the method to dummy data in three solution proposals with increasing complexity.

Solution proposal 1: Single variable application of LSQ

As a first, most simple step we use linear regression method with only one single input and output variable.

e-ISSN 2501-0263

Data:

Table 2: ***Single Input Data Set***

Input Variable	0,35	0,95	1,00	0,30	0,10	1,25	3,60	0,35	0,90	1,00	0,40	5,00	2,00	2,70	2,17
Output Variable	0.80	0.50	0.70	0.60	0.70	0.50	0.20	0.50	0.70	0.50	0.50	0.70	0.60	0.70	0.40

Linearization of data:

The Matlab function "polyfit" is linearizing the data points.

[11.]

```
>> p=polyfit(x,y,1)

p =

   -0.0221    0.6398
```

x is the input, **y** is the output and **1** for the digit of the polynome, which is linear and therefore 1. As a result, we get the k value of the linear function with -0.0221 and the d value with 0.6398.

Using the plot command shows the linear function: >> plot(x,y,'o',x,y1,'-')

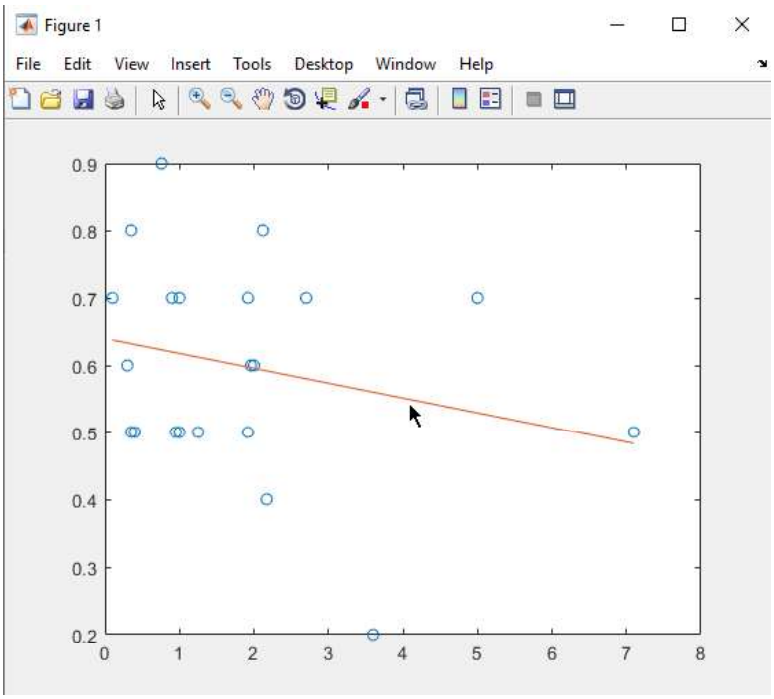

Fig. 1: ***Linear function***
Source: author's construction

Result:

Linear regression works well with a single variable. The optimized solution equals in that dummy data set x = 0.

Solution proposal 2: Multiple variable application of LSQ

As a next step, we increase complexity by applying regression to two input and one output variable:

Data:

Table 3: ***Double Input Data Set***

Input Variable 1	0,40	1,00	2,00	0,40	2,20	1,30	1,30	0,40	1,00	2,00	0,30	2,20	1,30	7,20	0,40	0,20	0,00	0,30	0,70	1,80	0,76
Input Variable 2	0,35	0,95	1,00	0,30	0,10	1,25	3,60	0,35	0,90	1,00	0,40	5,00	2,00	2,70	2,17	1,96	0,76	2,12	1,92	1,92	7,10
Output Variable	0,80	0,50	0,70	0,60	0,70	0,50	0,20	0,50	0,70	0,50	0,50	0,70	0,60	0,70	0,40	0,60	0,90	0,80	0,70	0,50	0,50

Based on these variables, a coefficient matrix using: X=[ones(size(x)) x y x.*y] can be created:

```
>> X=[ones(size(x)) x y x.*y]

X =

    1.0000    0.4000    0.3500    0.1400
    1.0000    1.0000    0.9500    0.9500
    1.0000    2.0000    1.0000    2.0000
    1.0000    0.4000    0.3000    0.1200
    1.0000    2.2000    0.1000    0.2200
    1.0000    1.3000    1.2500    1.6250
    1.0000    1.3000    3.6000    4.6800
    1.0000    0.4000    0.3500    0.1400
    1.0000    1.0000    0.9000    0.9000
    1.0000    2.0000    1.0000    2.0000
    1.0000    0.3000    0.4000    0.1200
    1.0000    2.2000    5.0000   11.0000
    1.0000    1.3000    2.0000    2.6000
    1.0000    7.2000    2.7000   19.4400
    1.0000    0.4000    2.1700    0.8680
    1.0000    0.2000    1.9600    0.3920
    1.0000         0    0.7600         0
    1.0000    0.3000    2.1200    0.6360
    1.0000    0.7000    1.9200    1.3440
    1.0000    1.8000    1.9200    3.4560
    1.0000    0.7600    7.1000    5.3960
```

Then, the regress function can be created as such:

```
>> regress(z,X)

ans =

    0.6857
   -0.0490
   -0.0505
    0.0248

>> z=0.6857-0.0490*x-0.0505*y+0.0248
```

Using the CFToolbox in MATLAB, the regress function can be visualized as such:

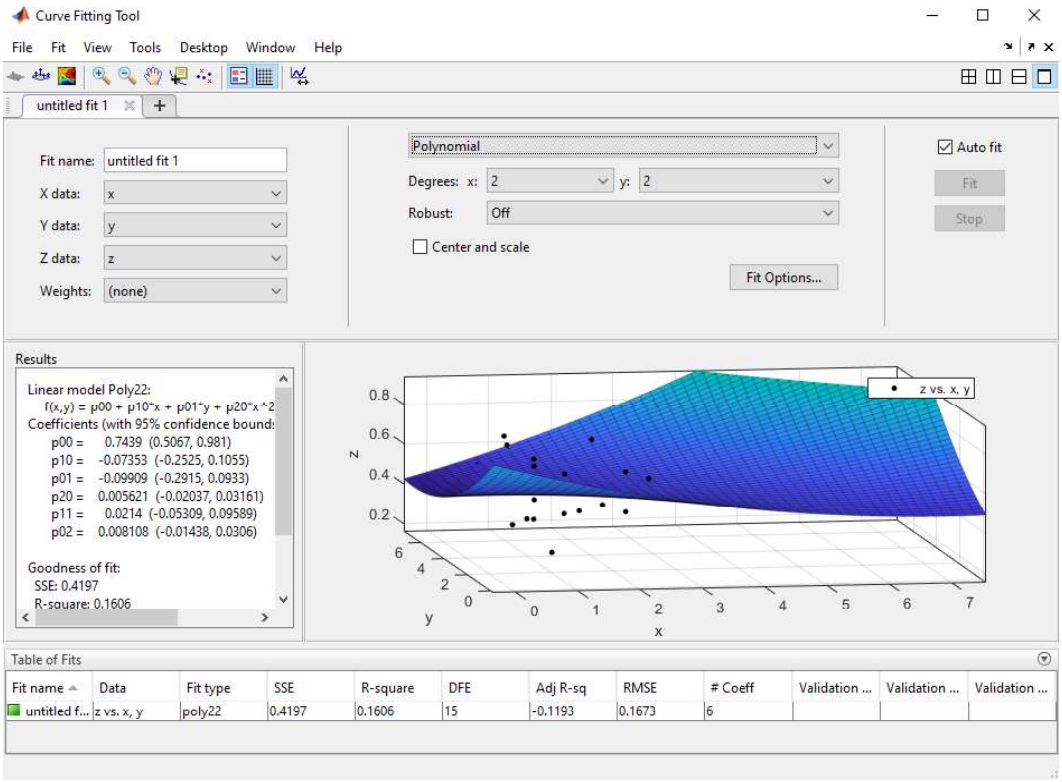

Fig. 2: ***Double input regression***
Source: author's construction

Result:

The graphic clearly shows the regression plane in the three-dimensional system. The maximum is in a corner of the system with a max y value and an x value between 4 and 5. So the maximum of this system is maximizing one input variable as much as possible. Interpreting this result with the goal of the process relevance optimization, the solution found here would not be applicable.

It would mean, that one process of project management should be done as much as possible, based on the existing data. In the practice of project management, focusing on just one process as much as possible would rarely be a good approach. However, this specific interpretation cannot be done, before the actual data of project managers is available. Only then can the results have interpreted correctly.

Solution proposal 3: Multiple variable application of LSQ

As a last step, we use linear regression method with four input and one output variable.

As we have 49 independent variables, we cannot visualize the curves anymore. To get the parameter P of a regression function in form of. Y=P1*x1+P2*x2+…….+Pn*xn we can use the fitlm command in Matlab.

Therefore, inputting the data and defining the variables x1 to x49 by using x1=data(:,1);x2=data(:,2);x3=data(:,3);x4=data(:,4);o=data(:,50); (here only with 4 variables for testing) and o the output, is not necessary. Matlab automatically defines each column as a parameter and the last column as output. Just importing the data and then input the command: mdl=fitlm(data) is enough to get this result:

```
Command Window

>> mdl=fitlm(data)

mdl =


Linear regression model:
    SumofOutputs ~ [Linear formula with 50 terms in 49 predictors]

Estimated Coefficients:
                    Estimate        SE         tStat       pValue
                   __________    ________    __________    _______

    (Intercept)       6.4036       4.2216       1.5169     0.13423
    ProcessNo1     -0.054538      0.04335      -1.2581     0.21294
    ProcessNo2     -0.074031     0.049765      -1.4876     0.14177
    ProcessNo3     -0.037776     0.058033     -0.65094     0.51742
    ProcessNo4      -0.12503     0.089346      -1.3994     0.16651
    ProcessNo5     -0.079918      0.05903      -1.3539     0.18054
    ProcessNo6     -0.024255     0.047506     -0.51057     0.61141
    ProcessNo7      0.019212      0.04586      0.41894     0.67666
    ProcessNo8     -0.048475     0.052277     -0.92726     0.35727
    ProcessNo9     -0.041543     0.049123     -0.84568     0.40088
```

Fig.3: ***Result of multiple regression in MATLAB***
Source: author's construction

As a next step we can use these parameters to define a function, which then needs to be optimized.

e-ISSN 2501-0263

Based on the result of the regression, we can use the parameter of the "Estimate column" to create our objective function for the maximization operation (in our case we use the minimization operation *fmincon* of MATLAB and just invert the output goals to achieve the maximization.

The function for optimization is reduced to 4 processes for a proof of concept.

```
objective = @(x) -0.054538*x(1)-0.074031*x(2)-0.03776*x(3)-0.12503*x(4);
```

The constraints of the function are:

- The sum of all x factors has to be 100
- Each x value has to be between 0 and 100.
- We want to start at an initial "estimation point" we use here the values of our initial process relevance defined by literature research. (in the case below, we randomly chose the values 1,5,5 and 1 for the x values to start.

This optimization can be implemented in MATLAB as follows:

```
objective = @(x) -0.054538*x(1)-0.074031*x(2)-0.03776*x(3)-0.12503*x(4);
% initial guess
x0 = [1,5,5,1];
% variable bounds
lb = 0.0 * ones(4);
ub = 100.0 * ones(4);
% show initial objective
disp(['Initial Objective: ' num2str(objective(x0))])
% linear constraints
A = [1 1 1 1];
b = 100;
Aeq = [];
beq = [];
% optimize with fmincon
%[X,FVAL,EXITFLAG,OUTPUT,LAMBDA,GRAD,HESSIAN]
% = fmincon(FUN,X0,A,B,Aeq,Beq,LB,UB,NONLCON,OPTIONS)
x = fmincon(objective,x0,A,b,Aeq,beq,lb,ub);
% show final objective
disp(['Final Objective: ' num2str(objective(x))])
% print solution
disp('Solution')
disp(['x1 = ' num2str(x(1))])
disp(['x2 = ' num2str(x(2))])
disp(['x3 = ' num2str(x(3))])
disp(['x4 = ' num2str(x(4))])
```

Running this script presents us values for x. therefore optimized process relevance. In our proof of concept all values x1 until x3 are 0 and x4 is 100.

e-ISSN 2501-0263

```
>> main
Initial Objective: -0.73852
Warning: Length of lower bounds is > length(x); ignoring extra bounds.
> In checkbounds (line 27)
  In fmincon (line 318)
  In main (line 24)
Warning: Length of upper bounds is > length(x); ignoring extra bounds.
> In checkbounds (line 41)
  In fmincon (line 318)
  In main (line 24)

Local minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the default value of the optimality tolerance,
and constraints are satisfied to within the default value of the constraint tolerance.

<stopping criteria details>

Final Objective: -12.503
Solution
x1 = 7.1695e-06
x2 = 1.1024e-05
x3 = 5.515c-06
x4 = 100

Optimization completed: The relative first-order optimality measure, 4.082197e-07,
is less than options.OptimalityTolerance = 1.000000e-06, and the relative maximum constraint
violation, 0.000000e+00, is less than options.ConstraintTolerance = 1.000000e-06.

Optimization Metric                                    Options
relative first-order optimality =   4.08e-07       OptimalityTolerance =   1e-06 (default)
relative max(constraint violation) =   0.00e+00    ConstraintTolerance =   1e-06 (default)
```

Fig.4: **Result of multiple regression optimization in MATLAB**
Source: author's construction

Result:

We only need the operations "fitlm" and "fmincon" of MATLAB to optimize our process relevance factors with linear regression if we summarize the 4 health indicators to a single output value. The actual values of the input variables from our dummy data set are as useless as in solution 2. Again, the optimization maximizes only one single input variable, and minimizes all others.

Conclusion

Based on the used dummy data set, the optimization works, but would not be useful for process relevance optimization, which is the overall goal to this research project. The optimum of the used data sets will always be in a corner of our "hyper-plane" and therefor always have one of the 49 processes at 100% and the others at 0%. Of course, this result is based only on the dummy data, which itself is randomly developed. So non- significant behavior was to be expected.

In the future, when a large data set of project management practitioners is available, the optimization approach must be repeated, and the results analyzed again. Only then can a deliberate decision in choosing the best optimization method be made.

Outlook to different optimization algorithms

As the linearity of least square optimization method causes maxima to appear in "corners", nonlinear models could offer better fitting to the actual data. The most promising methods could be non-parametric multivariate models like kernel-based and spline-based regression methods. [WEN19]

Especially because kernel-based models apply smoothing to nonlinear data as shown in a graphic example below. [CAO08]

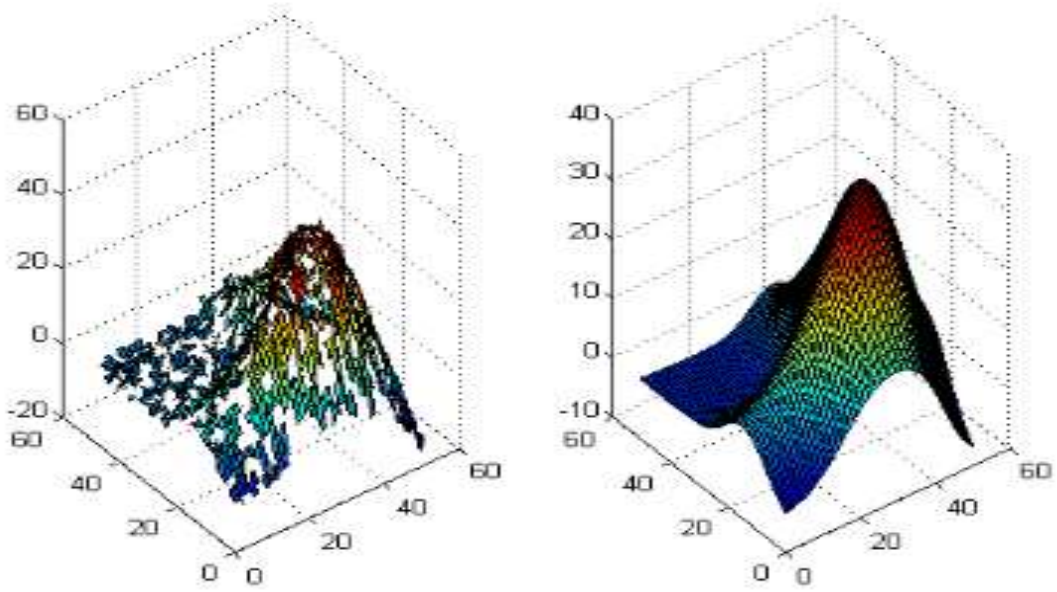

Fig.5: ***Kernel-based smoothing of nonlinear distributions***
Source: author's construction

Another approach to decrease the complexity and co-variance between all 49 processes would be splitting the processes into project phases and applying only project phase specific optimizations with decreased number of processes in each phase.

All these decisions will be finally made when the data from project practitioners is available. For now, it is only obvious, that least square optimization fundamentally works for multiple inputs, but is most likely optimize one single input parameter, which does not fit to the modeled system of project management practice.

References

ALE10 – E.C. Alexopoulos, *Introduction to Multivariate Regression Analysis*, Hippokratia Dec 2010, ISSN 1108-4189

CAO08, Yi Cao, *Multivatiant Kernel Regression and Smoothing*, [Online] Available at: https://de.mathworks.com/matlabcentral/fileexchange/19279-multivariant-kernel-regression-and-smoothing, [Accessed 24.02.2020]

PIC07 - Pichler, R., 2007. Scrum - *Agiles Projektmanagement erfolgreich einsetzen*, 1 edition. ed. dpunkt.verlag, Heidelberg

PMI17 - Project Management Institute, *A Guide To The Project Management Body Of Knowledge (PMBOK-Guide) – Sixth version*, 2017, Project Management Institute, Pennsylvania, USA

PRA19 - Niranjan Pramanik, *Multiple Linear Regression — with math and code*, [Online] Available at: https://towardsdatascience.com/multiple-linear-regression-with-math-and-code-c1052f3c7446 [Accessed 24.02.2020]

ROS18 – Rosenberger P. & Tick J., *Suitability of PMBOK 6th edition for agile-developed IT Projects*, 2018, CINTI 2018 IEEE 18th International Symposium on Computational Intelligence and Informatics

ROS19 - *Relevance of PMBOK v6 Processes for Tailored Agile Project Categories*", IEEE 13th International Symposium on Applied Computational Intelligence in Timisoara Romania May 29-30, 2019

WEN16 - Wendt R.., 2016, *Hybrides Projektmanagement für agile Projekte in mittelständischen und großen Unternehmen*, [Online] Available at: http://www.masventa.eu/fileadmin/user_upload/media/pdf_de/PMI/PMI-SG-Live_2016-12__1_.pdf, [Accessed 24.02.2020]

WEN19 - Wentao Gu et al, *Forecasting Realized Volatility in Financial Markets Based on a Time-Varying Non-Parametric Model*, Journal of Advanced Computational Intelligence and Intelligent Informatics Fuji Technology Press LTD, 2019, ISSN: 1343-0130