Proceedings of the Federated Conference on Computer Science and Information Systems pp. 743–747 DOI: 10.15439/2017F416 ISSN 2300-5963 ACSIS, Vol. 11

A Database Performance Polynomial Multiple Regression Model

Artur Nowosielski[1]

[1] Findwise Sp. z o.o.
ul. Widok 16/3, 00-023 Warsaw, Poland
Email: artnowo@gmail.com

Piotr A. Kowalski[2,3], Piotr Kulczycki[2,3]

[2] Faculty of Physics and Applied Computer Science
AGH University of Science and Technology
al. Mickiewicza 30, 30-059 Cracow, Poland
[3]Systems Research Institute
Polish Academy of Sciences
ul. Newelska 6, 01-447 Warsaw, Poland
Email: {pakowal,kulczycki}@ibspan.waw.pl

***Abstract*—Modelling of a database performance depending on numerous factors is the first step towards its optimization. The linear regression model with optional parameters was created. Regression equation coefficients are optimized with the Flower Pollination metaheuristic algorithm. The algorithm is executed with numerous possible execution parameter combinations and results are discussed. Potential obstacles are discussed and alternative modelling approaches are mentioned.**

I. Introduction

THIS article presents advances in the research introduced at FedCSIS 2015 DS-RAIT [1] and then presented at FedCSIS 2016 8th WSC [2]. The research relies on benchmarking the column-oriented database management system CODB. Model coefficients (weights) are optimized by the Flower Pollination Algorithm (FPA) [3]. Section II presents a sequence of steps that result in two regression equations without concrete weight values. Section III discusses how a model performance varies as a result of selecting different combinations of algorithm execution parameters in both variants. It also contains both model regression equations after supplying coefficients that achieved the highest accuracy. An outcome of the research is a mathematical model that expresses a database performance. It is created on the basis of empirical data collected by benchmarking the CODB database. In contrary to the the previous study [2], technology-oriented factors were not covered. Study has shown that such components have low impact on overall performance. Thus, further research is focused on data features and features of request sequence issued against a database management system.

II. Model construction

The modelling process was conducted as a sequence of activities described in next paragraphs. Firstly, a database benchmark was executed numerous times with different, but regular, settings in order to isolate data about specific factors' influence on overall performance. Then, a data concerning specific factors was visualised as a scatter plot and curves in a three-dimensional space. Function shapes for specific factors are recognized by a manual visual graph assessment. The model has been split into two variants at this stage. The second variant enriches the first one by including an information about proportion between database operations. The first variant skips this feature, whereas a second one uses it, in a form of a compositional variable components supplied directly to the regression equation. Both variants are compared to each other in terms of model accuracy. Second benchmarking round used random values obtained from the uniform distribution as input variables. Splitting the empirical data collection onto two stages was intended to ensure high quality of input data. A model formula with coefficients (weights) was coined as the result of both stages. Coefficients are optimized for the minimum error, i.e. the higher accuracy.

Model is created with the multiple regression technique with a priori known explanatory variable distributions. This method is based on a well-known and widely used linear regression model [4], with multiple input variables and a single output variable. Input parameters are called independent or explanatory variables, whereas an outcome is a dependent or explained variable. Created model is linear, although the explanatory variables are handled with polynomial functions. The linearity relates to the linearity of model coefficients in the model equation. The regression analysis has been chosen for the analysis because of its simplicity and straightforwardness.

The general regression formula for n-dimensional independent variables vector X and dependent variable Y is:

$$Y = w_n X_n + w_{n-1} X_{n-1} + ... + w_1 X_1 + w_0 + \epsilon, \qquad (1)$$

where ϵ denotes an error term, w is a $n+1$-dimensional coefficient vector, especially with the random term w_0. The error term is discussed in section III.

Model construction started from two fundamental factors acting as independent variables: a number of values (v) and a number of columns (c). Benchmark execution time t was set as an explained variable. Database benchmarking, that is a source of training data used for a model coefficient optimization, was limited by a number of constraints, such as maximum values for a number of columns and a number of values. A number of columns and a number of values both must be higher than 0 for obvious reasons. Both parameters are integers. A polynomial that expresses time depending from number of values will be referred to as $t(v)$, whereas time from number of columns will be $t(c)$. In this research, t was measured as a time of execution of 20 000 database operations. Initial column family state was 20 000, 30 000 or 40 000 tuples already existing at a start of measurement. A number of initially existing records depend on the ratio between different types of database operations, described in the next part of this paper. Regardless of parameters, each test has been executed 4 times. Values used for testing are randomly generated strings with lengths randomized from range $[100, 10000]$ with unified distribution. The formula 1 with supplied parameters is:

$$\begin{aligned} t &= t(v) + t(c) + w_0 + \epsilon \\ &= w_n v^{dv} + w_{n-1} v^{dv-1} + ... + w_{n-dv} v \\ &\quad + w_{n-dv-1} c^{dc} + w_{n-dv-2} c^{dc-1} + ... + w_1 c \\ &\quad + w_0 + \epsilon, \end{aligned} \tag{2}$$

where $n+1$ is a number of coefficients, w_i constitute model coefficients, especially with the w_0 random term. dv and dc are polynomial degrees for $t(v)$ and $t(c)$ polynomial functions, respectively.

The first variable, a number of values, denotes how many values are read or written while working with database. Although it may appear that usually this number is indefinite, this is not true for each case. There are many cases that have not only definite, but really low number of possible values. Such cases include, among others, gender, city or country, which are usually taken from dictionaries. The model assumes that in case of analysed field there is a finite value set. For the sake of model construction, a range of $[1, 500]$ was used as the v parameter domain.

The second factor, c, denotes a number of columns involved in given request. For the write or delete requests it is a number of columns that are modified, whereas for the read request it is a number of columns that consists of a read tuple. Similarly as in case of number of values, a range of $[1, 500]$ was used as a domain with similar assumptions. This range is supposed to handle most of typical Create-Retrieve-Update-Delete (CRUD) use cases.

In order to asses a general shape (a polynomial function degree) of functions for each parameter, an intermediate value within presumed domains were chosen and benchmarked extensively. Research started with the following values:

$$c, v \in \{1, 100, 200, 300, 400, 500\}.$$

A ratio between read and write operations was $\frac{2}{1}$, that is 67% of operations were read, and 33% were write. The very early result examination displayed that a shape varies more for lower values, than for higher values of c, so for the sake of a shape assessment, a value density has been increased in the lower part of the range:

$$c \in \{1, 5, 10, 30, 50, 100, 200, 300, 400, 500\}.$$

The first graph lets to extract first conclusions about the shape. The general trend is that a performance improves as the number of different values increase. But it is not the only observable trend. Results are more stable when the number of values increase. Standard deviation for $v = 1$ is 10.10 (including values that are hidden on the graph), for $v = 200$ it falls down to 0.22 and for $v = 500$ it is only 0.11. When it comes to a number of columns, operations time decreases as a number of columns grows, but does not approach 0 asymptotically. Somewhere between 200 and 300 columns (depending on v) it starts to grow again. Probably this marks a moment when there is too many columns to be handled by operating system I/O smoothly and jumps between numerous files starts to be a visible cost. For lower v values, a higher dispersion for low c values is observable, than for higher ones. However, the impact is lower than in case of low v values.

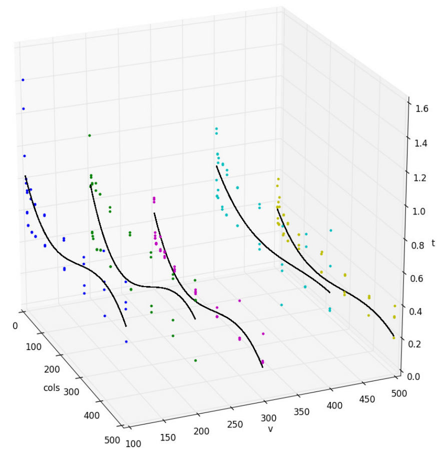

Fig. 1. A scatter plot of $t(v, c)$ with $\frac{R}{Wi} = \frac{2}{1}$ with 3rd order polynomial regression curves and points for chosen v values ($v = 1$ removed for clarity)

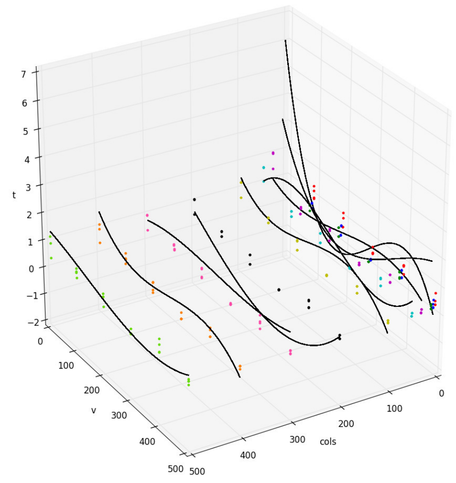

Fig. 2. A scatter plot of $t(v, c)$ with $\frac{R}{Wi} = \frac{2}{1}$ with 3rd order polynomial regression curves and points for chosen c values ($c = 1$ removed for clarity; please note the reversed graph orientation)

Figure 1 presents data points for $t(c)$ for constant v values with curves that present a supposed function shape for each value. Just as the main model, these functions were constructed using simple regression with FPA-optimized weights but they are used exclusively for presentation purposes. The trend is visible, with growing v values curves are smoother and almost linear near the end of the scope. This is an indication of a higher result stability for higher values. The same was repeated for different v with constant c values and presented on Figure 2. Conclusions are similar to those for Figure 1.

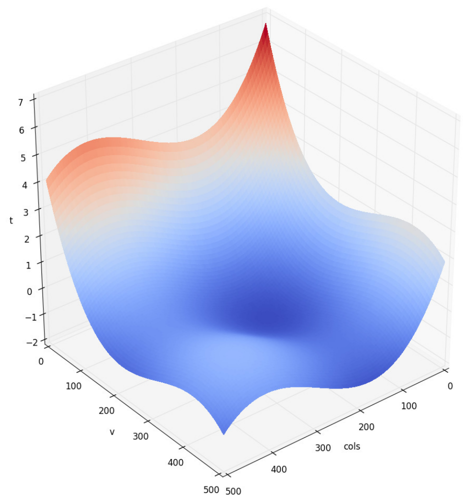

Fig. 3. A $t(v, c)$ with $\frac{R}{Wi} = \frac{2}{1}$ plane with weights ($c \in \{1, 5\}$ removed from graph for clarity; please note the reversed graph orientation)

On the basis of presented plots, a polynomial degree for both $t(c)$ and $t(v)$ was heuristically estimated as 3. This results in 7-dimensional optimization task for the algorithm, because the refined regression formula has 7 coefficients:

$$t = w_6 v^3 + w_5 v^2 + w_4 v + w_3 c^3 + w_2 c^2 + w_1 c + w_0 + \epsilon. \quad (3)$$

Besides number of columns and number of values, a very important performance determinant is a ratio between a different kinds of database operations. Three different kinds of operations were identified: O_R - read (fetching a tuple of values identified by a common key), O_{Wi} - insertion and O_{Wd} - deletion. These values compose a typical compositional data [10], which is a set of variables linked together so that they sum up to a constant value. The most intuitive representation for a ratio between different exclusive values of the same feature is a percentage, so a constant sum constraint is $O = O_R + O_{Wi} + O_{Wd} = 100$. Using a compositional data in regression model has a serious drawback. In order to ensure that regression model will be free from a noise, explanatory variables should be linearly independent from each other. This is not true for composite elements. When compositional data is to be used in regression model, it should be transformed to a set of abstract components that are not correlated, for example with the principal component analysis (PCA) or its' internal dependence should be weakened by removing one or more components. However, for the sake of this research, composite components were put directly into model, because there are only three components and removing even one of them would cause a model to infer on incomplete data. In order to monitor and control potential model accuracy degradation, results from model variants with and without compositional variable was compared.

In the first data discovery phase, 10 permutations of $R/Wi/Wd$ values were considered, with each component $\in \{0\%, 33.(3)\%, 66.(6)\%, 100\%\}$ so that the sum was always 100%. Assuming 240 tests executed for each of 10 proportions, it gives 2400 data points in total. For the majority operation ratios a plane has more or less common shape, similar to the one presented on the Figure 3. Rapid execution time growth in the lower parts of both crucial parameters is clearly visible as a red peak in the back right graph corner. There are areas of significantly higher results in the central and higher part of value number range. They have one feature in common: delete operation has dominated in these benchmark executions (100% or 67%). This is is unintuitive given the CODB storage architecture [2]. However, for a low column count (as in discussed cases), this may require to rewrite a high number of identifiers in order to move free space to the end of the area occupied by the value record. This is the most probable reason for exceptionally high execution times for test cases with high deletion ratio.

As it was mentioned previously, operation sorts ratios are expressed in percents, so their domain are integers from range $[0, 100]$. The O components were put directly into model equation:

$$\begin{aligned} t &= t(O) + t(v) + t(c) + w_0 + \epsilon \\ &= w_9 O_R + w_8 O_{Wi} + w_7 O_{Wd} + w_6 v^3 + w_5 v^2 + w_4 v \\ &\quad + w_3 c^3 + w_2 c^2 + w_1 c + w_0 + \epsilon. \end{aligned} \quad (4)$$

III. Model Optimization and Results

In this section, the main optimization goal is to minimize an error. The residual sum of squares (RSS; also known as sum of squared error, SSE) [11] metric has been chosen to measure error value. It is calculated as a sum of error term values from each sample:

$$RSS = \sum_{i=0}^{n} (\epsilon_i) = \sum_{i=0}^{n} (t_i - m_i)^2, \quad (5)$$

where t_i is actual value of i-th benchmarked case, m_i is a corresponding model value and n constitutes a number of benchmark results. The residual standard error (RSE) is presented as an auxiliary error metric. Its advantage over the RSS metric is that it is expressed in similar orders of magnitude as the original data which makes it directly comparable to actual results. The RSE can be calculated on the basis of the RSS:

$$RSE = \sqrt{\frac{RSS}{n - p - 1}} \quad (6)$$

where n is a number of samples (2400 and 4800 for the first and second modelling round) and p is number of parameters in each sample (7 and 10, for model without and with operations component, respectively). The $n-p-1$ value is called degrees of freedom and is commonly used metric in statistics.

The FPA [3] was used to optimize model coefficients. The optimization goal was to minimize the RSS. Execution parameters include a number of iterations, a number of flowers (solutions) and a switch probability. Number of iterations denote how many times a simulated pollination will be performed. Number of solutions defines how many solutions will

TABLE I
MODEL COEFFICIENT SEARCH RANGES

Coefficient	Initial range	Refined range
$w0$	$[-100, 100]$	$[-100, 100]$
$w1$	$[-100, 100]$	$[-10, 10]$
$w2$	$[-100, 100]$	$[-2, 2]$
$w3$	$[-100, 100]$	$[-1, 1]$
$w4$	$[-100, 100]$	$[-10, 10]$
$w5$	$[-100, 100]$	$[-2, 2]$
$w6$	$[-100, 100]$	$[-1, 1]$
$w7$	$[-100, 100]$	$[-100, 100]$
$w8$	$[-100, 100]$	$[-100, 100]$
$w9$	$[-100, 100]$	$[-100, 100]$

be handled in each iteration. Just like in case of the number of iterations, the bigger the value is the better performance is. A switch probability defines a probability of the random long pollination. This parameter defines a compromise between a local-optimum protection and a close result space exploration.

For each model coefficient, search range was initially defined as $[-100, 100]$. Initial results displayed that particular coefficients tend to converge to specific order of magnitude. A rule of thumb is that the order of magnitude is reversed proportional to given coefficient's degree, for example a random term is expressed in unities or at most tens, whereas v^3 weight always felt into 10^{-6}. Table I presents refined ranges.

After search range refinements, the model coefficient optimization phase has been performed. Table II presents results from the first phase for a Cartesian product of three algorithm execution parameters value sets: number of iterations in $\{1000, 2000, 3000\}$, number of solutions in $\{100, 200\}$ and switch probability in $\{0.2, 0.5\}$. These values where chosen arbitrarily on the basis of previous tries. Besides these basic execution parameters, FPA has other parameters that were not modified, default values are used. Their impact on model performance has been analysed in [12]. The *RSS* and *RSE* columns present the best (the least), error value achieved in algorithm executions with given parameter values. The *RSS diff* and *RSE diff* columns present how the error value has changed after compositional variable insertion to the regression equation. Difference is calculated as:

$$d = 100\% \cdot \frac{v2}{v1} - 100, \tag{7}$$

where $v1$ and $v2$ before and after values. A positive number indicates error growth, and a negative indicates a decrease, so that the lower value the better.

Table III presents the model performance after coefficients recalculation with randomly collected data set. For each parameters combination, a percentage improvement or regression in relation to performance from the corresponding row from Table II is presented in brackets in the same cells as RSS and RSE values. In both tables and both variants, a reversed dependency of the error from a number of iterations is visible yet weak. This conforms intuitive predictions that the more iterations the better, as the FPA algorithm is by design protected from result degradation. As p is growing, model accuracy falls down dramatically. For the simpler variant 1 the best results were obtained with almost all the parameter combinations, regardless of iterations or solutions number. The more complex variant 2 required at least 2000 iterations to get the best result. In most cases, variant 1 is less accurate than variant 2. This happens despite theoretical risk of disturbances caused by mutual correlation of three variables. It is likely that a disturbance introduced by the correlated compositional variable components into the regression model makes less damage to the accuracy than a partial lack of information. A difference between phases 1 and 2 is much higher than anticipated and requires further investigation, because there are many possible reasons. More evenly distributed data than in the 1st phase was expected to increase model accuracy, but such a low RSE may indicate overfitting to the train data caused by a lack of a cross validation.

Coefficients obtained with the best solution from table III were put into equations (3) and (4) resulting with:

$$\begin{aligned} t &= -4.586E-8 \cdot v^3 + 2.873E-5 \cdot v^2 - 1.873E-3 \cdot v \\ &\quad + 4.674E-9 \cdot c^3 + 4.536E-6 \cdot c^2 - 6.034E-4 \cdot c \\ &\quad + 0.544 + \epsilon \end{aligned} \tag{8}$$

as the model variant 1 and:

$$\begin{aligned} t &= t(O) + t(v) + t(c) + w_0 \\ &= -0.301 \cdot O_R - 0.304 \cdot O_{Wi} - 0.286 \cdot O_{Wd} \\ &\quad - 5.004E-8 \cdot v^3 + 2.883E-5 \cdot v^2 - 1.460E-3 \cdot v \\ &\quad + 3.513E-9 \cdot c^3 + 5.398E-6 \cdot c^2 - 7.249E-4 \cdot c \\ &\quad + 29.800 + \epsilon \end{aligned} \tag{9}$$

for the variant 2. Both weights vectors were taken from $3000/200/0.2$ executions with $RSS = 2478$ and $RSS = 2179$ for variant 1 and 2, respectively.

IV. SUMMARY

This paper presents a mathematical model of a column-oriented database performance. Mandatory explanatory variables of the model are a number of columns and a number of different values present in a database and requests issued against it. As an optional component, explanatory variables set includes information about percentage share of different kinds of database CRUD operations. Data was collected in two phases. The first stage collected data necessary to assess function shape for particular factors whereas the second increased statistical value of the model input data. Number of columns and values explanatory variables model were assessed as third-order polynomial, which resulted in regression equation with 7 coefficients. A variant with operation ratios increased a number of coefficients to 10. Both problems were optimized with the FPA for minimal RSS. The switch probability parameter p turned out to have a significant impact on the model accuracy, making a model generated with $p > 0.5$ much less accurate, especially with lower iteration counts. The best results were obtained with $p = 0.2$. An impact of the remaining parameters, iteration and solution counts, turned out to be lower, but still observable.

In the future, the model may benefit from trying out other metaheuristic algorithms, such as the Krill Herd Algorithm

TABLE II
THE FPA-OPTIMIZED MODEL PERFORMANCE - 1ST PHASE

Iterations	Solutions	Switch probability	Variant 1: without O_x		Variant 2: with O_x			
			RSS	RSE	RSS	RSS diff	RSE	RSE diff
1000	100	0.2	61 088	5.05	57 168	-6.42%	4.89	-3.21%
		0.5	71 530	5.47	119 884	67.60%	7.08	29.55%
	200	0.2	61 090	5.05	57 185	-6.39%	4.89	-3.19%
		0.5	82 665	5.88	91 973	11.26%	6.21	7.25%
2000	100	0.2	61 088	5.05	57 143	-6.46%	4.89	-3.23%
		0.5	61 088	5.05	57 143	-6.46%	4.89	-3.23%
	200	0.2	61 088	5.05	57 143	-6.46%	4.89	-3.23%
		0.5	61 088	5.05	57 144	-6.46%	4.89	-3.23%
3000	100	0.2	61 088	5.05	57 143	-6.46%	4.89	-3.23%
		0.5	61 088	5.05	57 143	-6.46%	4.89	-3.23%
	200	0.2	61 088	5.05	57 143	-6.46%	4.89	-3.23%
		0.5	61 088	5.05	57 143	-6.46%	4.89	-3.23%

TABLE III
THE FPA-OPTIMIZED MODEL PERFORMANCE - 2ND PHASE

Iterations	Solutions	Switch probability	Variant 1: without O_x		Variant 2: with O_x			
			RSS	RSE	RSS	RSS diff	RSE	RSE diff
1000	100	0.2	2478 (-95.94%)	0.72 (-85.75%)	2321 (-95.94%)	-6.34%	0.70 (-85.69%)	-2.78%
		0.5	7318 (-89.77%)	1.24 (-77.32%)	353 448 (194.82%)	4729%	8.59 (21.33%)	592.74%
	200	0.2	2479 (-95.94%)	0.72 (-85.75%)	2630 (-95.40%)	6.09%	0.74 (-84.87%)	2.78%
		0.5	12 119 (-85.34%)	1.59 (-72.95%)	816 355 (787.60%)	6636%	13.06 (110.31%)	721.38%
2000	100	0.2	2478 (-95.94%)	0.72 (-85.75%)	2179 (-96.19%)	-12.10%	0.67 (-86.30%)	-6.94%
		0.5	2478 (-95.94%)	0.72 (-85.75%)	2181 (-96.18%)	-12.02%	0.67 (-86.30%)	-6.94%
	200	0.2	2478 (-95.94%)	0.72 (-85.75%)	2179 (-96.19%)	-12.10%	0.67 (-86.30%)	-6.94%
		0.5	2478 (-95.94%)	0.72 (-85.75%)	2192 (-96.16%)	-11.54%	0.68 (-86.09%)	-5.56%
3000	100	0.2	2478 (-95.94%)	0.72 (-85.75%)	2179 (-96.19%)	-12.10%	0.67 (-86.30%)	-6.94%
		0.5	2478 (-95.94%)	0.72 (-85.75%)	2179 (-96.19%)	-12.10%	0.67 (-86.30%)	-6.94%
	200	0.2	2478 (-95.94%)	0.72 (-85.75%)	2179 (-96.19%)	-12.10%	0.67 (-86.30%)	-6.94%
		0.5	2478 (-95.94%)	0.72 (-85.75%)	2181 (-96.18%)	-11.99%	0.67 (-86.30%)	-6.94%

[13] [14]. Numeric optimization is also one of the most typical appliances of evolutionary and genetic algorithms [5]. A compositional value transformation such as Additive/Centered/Isometric Log ratio Transformation (ALR, CLR, ILR) [15] should be used against the operations compositional variable. This should improve model accuracy and let to perform analysis without comparing both model variants. Trying out different modelling techniques, like non-parametric methods [8] [9] and other prediction models, such as Radial Basis Function neural networks [6] [7], may improve accuracy. The model is intended to be a foundation for a database performance optimization, which means it should be as accurate and sophisticated as possible. However, it needs to maintain simplicity to be executed with satisfying performance. This balance between accuracy and execution time is crucial for the considered application.

REFERENCES

[1] A. Nowosielski, P. A. Kowalski, and P. Kulczycki, "The column-oriented database partitioning optimization based on the natural computing algorithms," in *2015 Federated Conference on Computer Science and Information Systems, FedCSIS 2015, Łódź, Poland, September 13-16, 2015*, 2015. doi: 10.15439/2015F262 pp. 1035–1041. [Online]. Available: http://dx.doi.org/10.15439/2015F262

[2] A. Nowosielski, P. A. Kowalski, and P. Kulczycki, "The column-oriented data store performance considerations," in *Computer Science and Information Systems (FedCSIS), 2016 Federated Conference on*. IEEE, 2016, pp. 877–881.

[3] X.-S. Yang, "Flower Pollination Algorithm for Global Optimization," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2012, vol. 7445 LNCS, pp. 240–249. ISBN 9783642328930. [Online]. Available: http://link.springer.com/10.1007/978-3-642-32894-7_27

[4] C. R. Rao and H. Toutenburg, "Linear models," in *Linear models*. Springer, 1995, pp. 23–24.

[5] C. Blum and X. Li, "Swarm Intelligence in Optimization," *Swarm Intelligence Introduction and Applications*, pp. 43–85, 2008. doi: 10.1007/978-3-540-74089-6

[6] T. Santhanam and A. C. Subhajini, "Radial Basis Function Neural Network."

[7] S. E. VT and Y. C. Shin, "Radial basis function neural network for approximation and estimation of nonlinear stochastic dynamic systems," *IEEE Transactions on Neural Networks*, vol. 5, no. 4, pp. 594–603, 1994.

[8] L. Xu, A. Krzyżak, and A. Yuille, "On radial basis function nets and kernel regression: statistical consistency, convergence rates, and receptive field size," *Neural Networks*, vol. 7, no. 4, pp. 609–628, 1994.

[9] P. Kulczycki, "Kernel Estimators in Systems Analysis," *WNT, Warsaw*, 2005.

[10] V. Egozcue and J. J. Tolosana, "Lecture Notes on Compositional Data Analysis," vol. 962, no. 2003, p. 96, 2007. [Online]. Available: http://dugi-doc.udg.edu/handle/10256/297

[11] S. Khan, "Predictive distribution of regression vector and residual sum of squares for normal multiple regression model," *Communications in Statistics-Theory and Methods*, vol. 33, no. 10, pp. 2423–2441, 2005.

[12] S. Łukasik and P. A. Kowalski, "Study of Flower Pollination Algorithm for Continuous Optimization," in *Intelligent Systems'2014*. Springer, 2015, pp. 451–459. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-11313-5_40

[13] A. H. Gandomi and A. H. Alavi, "Krill herd: A new bio-inspired optimization algorithm," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 12, pp. 4831–4845, 2012. doi: 10.1016/j.cnsns.2012.05.010. [Online]. Available: http://dx.doi.org/10.1016/j.cnsns.2012.05.010

[14] P. A. Kowalski and S. Łukasik, "Experimental Study of Selected Parameters of the Krill Herd Algorithm," in *Intelligent Systems'2014*. Springer, 2015, pp. 473–485. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-11313-5_42

[15] K. Hron, P. Filzmoser, and K. Thompson, "Linear regression with compositional explanatory variables," *Journal of applied statistics*, vol. 39, no. 5, pp. 1115–28, 2012. doi: 10.1080/0266476YYxxxxxxxx. [Online]. Available: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2712304&tool=pmcentrez&rendertype=abstract