STUDIA INFORMATICA 2011
Volume 32 Number 2A (96)

Tomasz PŁUCIENNIK, Marcin MICHALAK
Silesian University of Technology

SPLICO – SPLINE DESCRIPTION OF CLOSED CONTOURS

Summary. This paper deals with approximation of a two-dimensional closed curve. The notion of *SpliCo* is introduced that is based on spline functions. Spline functions are a common method of the regression function estimation. As the closed curve cannot be described as the function (from the mathematical point of view) a modification of the standard method must be done. This paper describes three following models and each of them makes it possible to describe a contour more smoothly and accurate. The best model of *SpliCo* is compared with the standard B-spline model.

Keywords: approximation, function smoothness, estimation of the regression function, spline functions

SPLICO – OPIS ZAMKNIĘTYCH KONTURÓW ZA POMOCĄ FUNKCJI SKLEJANYCH

Streszczenie. Niniejszy artykuł porusza problem aproksymacji dwuwymiarowej krzywej zamkniętej. Nazwa *SpliCo* została wprowadzona ze względu na to, że rozwiązanie oparte jest na funkcjach sklejanych. Funkcje sklejane są typową metodą estymacji funkcji regresji. Zamknięta krzywa nie może, z matematycznego punktu widzenia, być przedstawiona jako funkcja, więc do tego celu należy zaproponować modyfikację standardowej metody aproksymacji. Artykuł ten przedstawia trzy kolejne modele aproksymacji, każdy dający bardziej poprawny i wizualnie gładszy opis konturu.

Słowa kluczowe: aproksymacja, estymacja funkcji regresji, funkcje sklejane, gładkość funkcji

1. Introduction

Estimation of the regression function is one of the subproblems from the discipline called machine learning [11, 12]. The aim of the evaluation of regression function is to find some dependencies between variables in the observed data set. The same task may be also described as the approximation of the known (unknown) function with the other one. There are two main groups of methods of the regression function estimation: parametrical and nonparametrical. Models with a well defined functional form belong to parametrical methods and are described with finite number of free parameters which values must be established (usually as the result of the optimization). Nonparametrical methods are also described by some free parameters but their results are not given in the analytical form. These methods do not make any assumptions about the functional form of described dependence. They only describe dependence but do not explain its nature. Very common nonparametric regression functions estimators are spline functions [1, 4], radial basis functions [6], additive (and generalized additive) models [5] or kernel estimators [8, 13] with Support Vector Machines [2]. This short article describes the application of modified spline functions into approximation of closed curves (contours). First part of the paper describes the context of the problem and explains the nature of spline function regression. Afterwards three models of spline closed curve approximation are described. Finally results of experiments, including the comparison of *SpliCo* and B-splines, are shown and some final conclusions complete the paper.

2. Problem Description

The closed curve is given as a series of points in two-dimensional space. The demanded result should be an equation representing the curve as accurately as possible. This means that the sequence of points representing the curve will be approximated by a function. Since the curve is closed there are at least two points where its shape cannot be represented as a function. This is why the curve must be divided into fragments which will be approximated separately. The result of this approximation will be a group of splines. The point of two splines connection is called a knot.

The above calculations may seem trivial but there is a requirement that the shape of the approximated spline curve should be smooth. This requirement is not ensured in the neighbourhoods of the knots. Therefore a special modification of each consequent spline must be considered.

3. Spline Regression

One of the method of nonparametrical regression are spline functions [1]. With this method an unknown regression function is divided into several functions in the given knots. If we assume that the domain of the spline function is divided with K knots (what means that there will be $K + 1$ different functions) and particular functions are polynomials with degree equal to q the spline function equation simplifies and takes the following form:

$$F(x) = \sum_{i=1}^{q} \alpha_{i-1} x^{i-1} + \sum_{k=1}^{K} \beta_k \left(x - t_k\right)_+^q , \tag{1}$$

where t represents knots, β represents additional factors, m_+ means the positive value of the m also defined as $m_+ = \frac{m + |m|}{2}$ and q is the degree of the approximating polynomial. The error is generally based on the difference between the input and output functions so the approximation minimizes the norm $||f(x) - F(x)||$, usually using the root mean squared error as the objective function.

4. *SpliCo* Definitions

As it was described in the previous section spline functions can be named as the analytical description of the unknown dependence. The main aim of this article is to create a "spline language" for closed curves description. It is obvious that a closed curve cannot be represented by a single $y = f(x)$ function . Its shape simply prevents it by having more than one value for a single argument. But a closed curve may be described as the connection of several non-closed curves that may be functions from the mathematical point of view.

This section describes three approaches for spline contour description. Each of them is based on the assumption that the single spline is the polynomial which order is equal to four:

$$F(x) = \sum_{i=1}^{4} \alpha_{i-1} x^{i-1} , \tag{2}$$

where α represents unknown factors of the approximation function. This form of $F(x)$ is simple to be calculated and is often used for this purpose. It can have both convex and concave parts which makes it ideal for most cases. The factor before the highest powers of x can be equal to zero if a lower degree polynomial is sufficient. This covers the quadratic and linear functions.

4.1. Spline Approximation

Spline approximation extends typical task of approximation by dividing arguments into intervals in selected knots. Created fragments are approximated separately. Usually, in typical spline approximation, knots are equidistant from each other. However in case of a closed curve the most intuitive solution is to place them in points where the curve shape stops being a function. In this kind of points (x_0, y_0) the tangent line of the curve given by the equation:

$$y - y_0 = f'(x_0)(x - x_0) \tag{3}$$

will be vertical what means the value of the tangent line direction coefficient will equal infinity:

$$|f'(x_0)| = \infty . \tag{4}$$

Fig. 1. shows an example of choosing the spline knots for a closed curve.

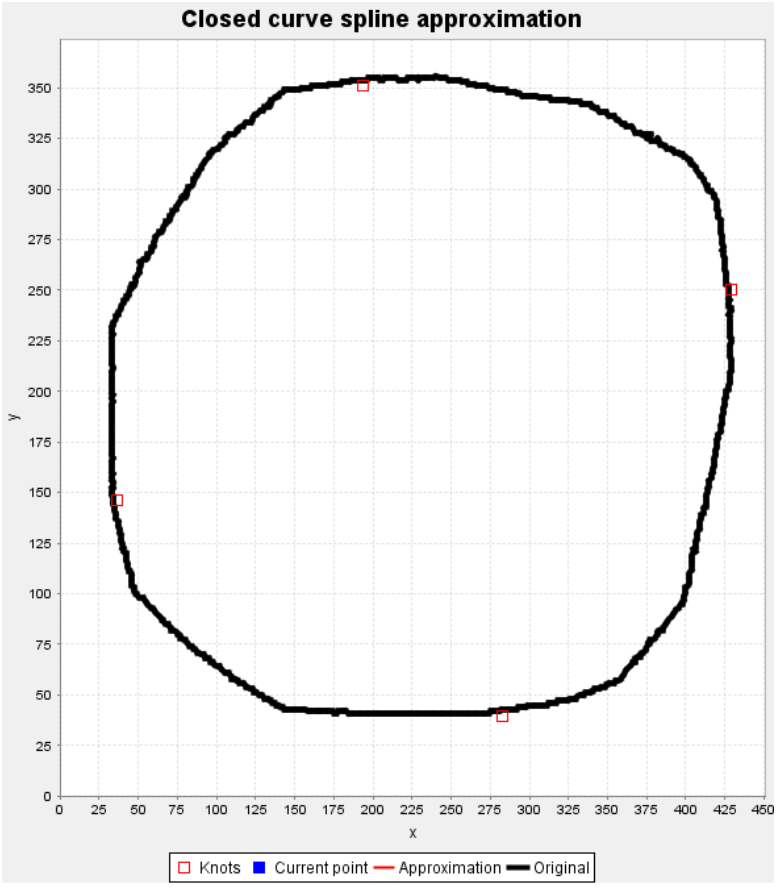

Fig. 1. Example of choosing spline knots
Rys. 1. Przykład wyboru węzłów

However the above description is only a speculation. The shape of the whole curve has the influence on errors so the created solution should decide about the knots on its own. In this paper it is assumed that at the beginning the number and position of knots are given. There is nothing said whether they will give satisfying result. Approximation can be done in

iterations until a certain stop condition is fulfilled e.g. number of iteration is exceeded, the error is lower than the given threshold or the error is not declining. In the proposed solution knots are changed so that the absolute value of the derivative in every knot will grow during iterations. In geometric terms the tangent lines in moving knots will tend to be as close to being vertical as possible. Iterations are conducted as long as the error declines because this method should give the most accurate results.

4.2. Assurance of Smoothness – Smoothing Addend

The results of the approximation for every spline joined together create the approximated shape of the curve. However in the neighbourhood of the spline knots the smoothness of the new curve is not assured. A modification of the spline polynomials is required to achieve a continuity of the curve. A special addend could make the values of every two functions equal in their common knot. Additionally the curve shape could be differentiable which will make sure that no sharp edges are present but this is very hard to compute. Furthermore knots placed on the curve in places where tangent line will be vertical complicates even the smoothness assurance.

The mathematician definition of smoothness say that function is called smooth of class C^n if all derivatives of the function up to and including order n exist and are continuous [10]. The class C^1 ensures that a function do not have discontinued parts and is differentiable in every point and therefore displayed function looks smooth. Since polynomials have smooth derivatives up to any order the interesting part is the knot between two splines. It requires that left-hand and right-hand derivatives in the knot point (or two left-hand or right-hand derivatives in case of a point in whose neighbourhood the shape is not a function) are equal. Therefore so called smoothing terms are used in spline approximation e.g. in [9]. According to the Eq. (1) and to the assumption of the number of knots (parameter that is set at start) and the polynomial degree we obtain the following spline formula:

$$F(x) = \sum_{i=1}^{4} \alpha_{i-1} x^{i-1} + \sum_{k=1}^{K} \beta_k \left(x - t_k\right)_+^4 \tag{5}$$

If we assume that the approximated shape is a function and splines are calculated from left to right then every next spline will have a polynomial addend based on previous knots (for $x > t_k$). Previous spline will shape the new one making the output smoother. K first splines will create the result.

4.3. Assurance of Smoothness – Geometric Construction

The results of implemented solution still have issues with curve smoothness as shown in the results chapter. To create a fully smooth shape a geometric algorithm will be proposed. In this algorithm approximation is done without any smoothing addend. Returned polynomials almost surely do not have matching endings nor create a smooth closed curve. The algorithm uses a fragment of a circle and (if necessary) a fragment of a straight line to create a hyphen filing the gap between two splines. The algorithm steps for a pair of neighbour splines are as follows (refer to Fig. 2 for details):

1. Remove ends of the splines near the spline knot with a chosen range. Approximation error is growing near the ends of the approximated function interval. Save new ends of the splines: *S1* and *S2*.
2. Construct tangent lines to the two splines in *S1* and *S2*.
3. Calculate tangents crossing point *P*.
4. Check whether point *P* coordinates values are between coordinate values of points S1 and *S2* in at least one dimension (e.g. near the points of the curve with minimal and maximal *x* coordinates values). If this condition is not satisfied then extend the range of spline ends cutting and go to 1.
5. Find the distance *a* between *P* and the closest from *S1* and *S2*. Mark the longer segment from point *P* to *S1* or *S2*.
6. Calculate point *T* on the longer segment which distance from *P* is *a*.
7. The remaining part of the longer segment of length *b* will be put into result – part of a straight line. New ends are one of the *S1* and *S2* points (point *S*) and point *T*.
8. Calculate perpendicular lines to tangent lines in corresponding point *S* and *T*.
9. Since there are two segments *SP* and *PT* with equal length, the new lines have a crossing point *R*, which is a centre point of a circle with radius *r*.
10. Add to the result the part of the circle between point *S* and *T*, which is closer to point *P*.

Additionally the implementation should return the hyphen points in order imposed by splines order, e.g. counter clockwise from *S1* to *S2*. Although inserted circle part is not a function the use of tangent lines assures smooth shape of the connected splines and hyphens.

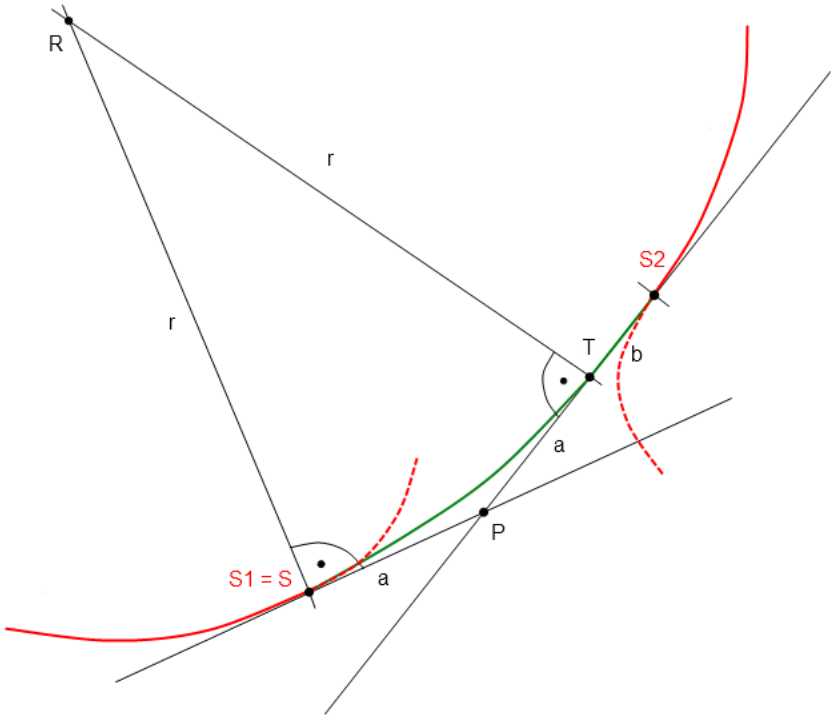

Fig. 2. Smoothing hyphen geometric construction
Rys. 2. Konstrukcja łącznika wygładzającego

5. Experiments and Results

5.1. Application

Contours come from a raster map with isolines. The only interesting part of the map were closed isolines and they were taken into the further preprocessing step. With the usage of standard image processing methods (thresholding etc.) five isolines were extracted and saved into separate black-white bitmap files. The extraction of contour pixels were performed with the usage of the 8-way chain code that is used to describe direction to move over pixels of the image [3].

5.2. Quality Measure

The approximation error is the sum of errors for every spline. There is also a special factor expanding the error on the basis of information about knots and the degree of the polynomial approximation function:

$$totalError = \sum_{k=1}^{K}\left[error + \lambda\left(\sum_{i=1}^{4} xPower_{i-1} + K \right)\right], \tag{6}$$

where:

- *error* is the error that origins from the spline function itself: for the polynomial part it is the root mean squared error (*RMSE*) and for circular part it is the average Euclidean distance of the point from the arc,
- λ is a given penalty factor for high power exponents in the spline function and bigger number of knots,
- *xPower* is the power exponent of given occurrence of x in the spline polynomial function (where factors are not equal to zero),
- K is the number of knots.

5.3. Results

Figure 3 shows an example of the same calculations done with no smoothing, smoothing addend and geometric smoothing. Note that the result in the first two figures is not very satisfying. Table 1 shows approximation results for five contours.

Fig. 3. Example results (contour 1) from left to right: with no smoothing, with addend, with geometric construction
Rys. 3. Przykładowe wyniki (kontur 1) od lewej: bez wygładzania, ze składnikiem wygładzającym, z wygładzaniem geometrycznym

Table 1

Summarized results of the approximation

Contour	No smoothing	Smoothing addend	Geometric smoothing
1	42.64	34.24	40.35
2	39.16	33.46	67.98
3	36.08	31.24	59.48
4	24.45	27.04	36.48
5	24.35	36.84	25.99

We may see that from the qualitative point of view the most advanced model does not give satisfying results. But it also may be seen (Fig. 3) that from the utilical point of view the third seems as smooth as it should.

5.4. Comparison with B-splines

B-spline parametric curves can also be used for approximation [1]. The approximation of the five isolines was therefore tested using a MATLAB-derived procedure creating B-spline curves. This procedure is able to calculate the best positions of knots and control points of the B-spline and apart from the points does not require any other parameters [7]. The results from MATLAB were then compared to *SpliCo* approximation in terms of error and calculation time. Since it will be the best compare both algorithms under the same conditions, the proposed precautions were conducted:

- RMSE errors are assessed for both algorithms (in case of *SpliCo* $\lambda = 0$) based on the distance of an output point to the closest of the input points (the easiest way for a closed curve),
- run time of both algorithms was measured on the same machine, not aggravated with any other time-consuming calculations, taking into account only algorithms itself not i.e. drawing results.

SpliCo was run with standard parameters (four start knots evenly distributed, default error growth threshold) except of the mentioned error modifications. Results of approximation for first two contours are shown in fig. 4. and fig. 5. Table 2 shows the RMSE errors and run times for both methods.

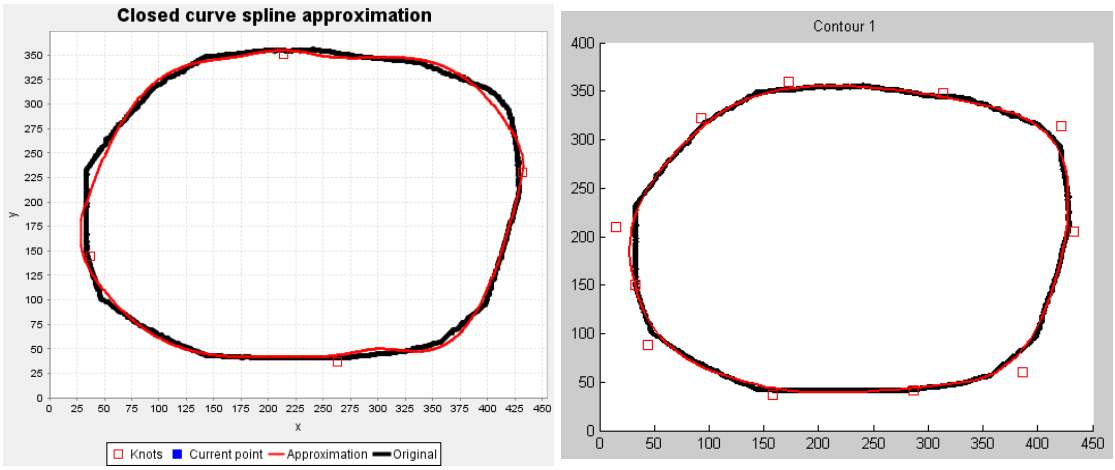

Fig. 4. Results of the approximation using *SpliCo* (left) and B-splines (right) for the first contour

Rys. 4. Wyniki aproksymacji z wykorzystaniem *SpliCco* (po lewej) i funkcji B-sklejanych (po prawej) dla pierwszego konturu

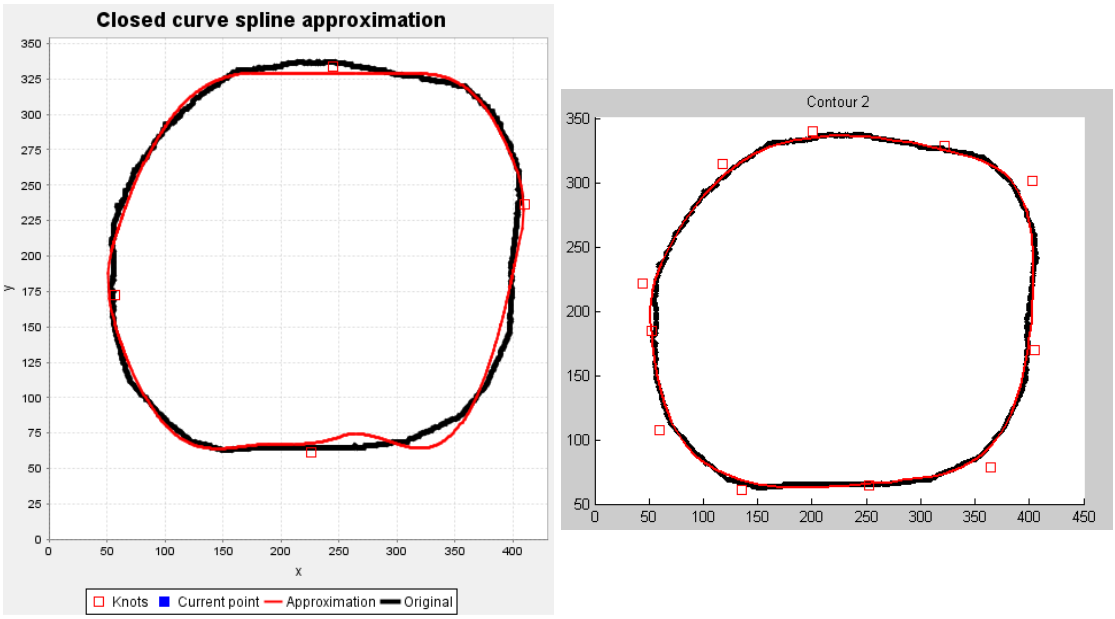

Fig. 5. Results of the approximation using *SpliCo* (left) and B-splines (right) for the second contour

Rys. 5. Wyniki aproksymacji z wykorzystaniem *SpliCo* (po lewej) i funkcji B-sklejanych (po prawej) dla drugiego konturu

Table 2

Comparison of the two methods for the test data

Contour	*SpliCo*		B-spline	
	RMSE error	Run time [s]	RMSE error	Run time [s]
1	3.0366	0.407	3.3748	296.083
2	3.8374	0.215	3.0551	258.522
3	2.6524	0.095	2.7523	213.673
4	1.8204	0.054	1.8004	150.362
5	1.5571	0.023	1.3709	111.742

The error values for the both of algorithms are comparable with the small advantage for the B-splines. However execution times are radically longer in case of the B-splines. At this moment it is relevant to mention that MATLAB is an interpreted script executed on the Java Virtual Machine and the *SpliCo* algorithm was implemented directly in Java and also executed on the Java Virtual Machine. Nevertheless the additional strain (the script interpreting) will not slow down the application 1000 times. The most important factor here is the calculation of the knots and control points of the B-spline. *SpliCo* algorithm finds the potential knots in the initial part of the algorithm so finding the optimal solution has the smaller complexity.

In *SpliCo* results limitations of use of the polynomials are visible by presence of certain "waves". They can be reduced by experimenting with knots position or adding/removing knots. In general, higher number of knots increases the probability of "waving" curve, but too small number of knots increases the error.

Both compared solutions give satisfying results, however *SpliCo* is simpler and faster.

6. Conclusions and Further Work

In this article the method of closed curves approximation with the usage of spline functions was presented. Starting from the original spline functions following modifications were introduced like smoothing addend and geometric construction. Final *SpliCo* fulfils the condition of being smooth and accurate though it is not obvious when the approximation error is observed. It points that the standard method of approximation error evaluation (*RMSE*) should be replaced with the more appropriate method. The *RMSE* fails as the quality function near regions where the slope of the tangent of *SpliCo* takes values close to right angle. It causes that the *SpliCo* is relatively close to training points from the Euclidean distance point of view, but the difference between $f(x)$ and $F(x)$ is big.

Also the choosing the start knots has impact on the result. The smoothing addend modifies the curves equation. The smoothing hyphens are outputted between the splines. Output range for every polynomial is therefore shrank to ending points of the neighbour hyphens. The total error grows when geometric smoothing is applied because the method does not take into account the knot position nor the shape of the approximated curve. It is simply trying to connect two splines. It suggests to define the new condition of choosing the starting knots positions and the number of knots. It is worth to notice that *SpliCo* finds the spline model significantly faster than full adaptive B-splines model.

SpliCo should be useful as the mathematical description of isolines, especially as the base of three-dimensional maps generating. It is also common that three-dimensional grids need the analytical form of contours as the input argument. *SpliCo* should be attractive for this purpose also.

Acknowledgements

This work was supported by the European Community from the European Social Fund.

BIBLOGRAPHY

1. de Boor C.: A Practical Guide to Splines. Springer-Verlag, 1978.
2. Boser B. E., Guyon I. M., Vapnik V. N.: A training algorithm for optimal margin classifiers. Proc. of the 5th Annu. Workshop on Comput. Learn. Theory, 1992, p. 144÷152.
3. Freeman H.: On the Encoding of Arbitrary Geometric Configurations. IRE Trans. on Electron. Computers EC-10, 1961, p. 260÷268.

4. Friedman J. H.: Multivariate Adaptive Regression Splines. Ann. of Stat. 19, 1991, p. 1÷141.
5. Hastie T. J., Tibshirani R. J.: Generalized Additive Models. Chapman & Hall/CRC, 1990.
6. Hastie T. J., Tibshirani R. J., Friedman J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, 2001.
7. Hunyadi L.: B-splines Matlab toolbox, http://www.mathworks.com/matlabcentral/fileexchange/27374-b-splines.
8. Nadaraya E. A.: On estimating regression. Theory of Probab. and its Appl. 9, 1964, p. 141÷142.
9. Pottmann H., Leopoldseder S., Hofer M.: Approximation with Active B-spline Curves and Surfaces. PG '02: Proc of the 10th Pac. Conf. on Computer Graph. and Appl., 2002, p. 8÷25.
10. Shikin E. V.: Handbook and Atlas of Curves. CRC Press, 1995.
11. Taylor J. S., Cristianini N.: Kernel Methods for Pattern Analysis. Cambridge University Press, 2004.
12. Vapnik V. N.: Statistical Learning Theory. Wiley, 1988.
13. Watson G. S.: Smooth Regression Analysis. Sankhya - The Indian J. of Stat. 26, 1964, p. 359÷372.

Recenzent: Prof. dr hab. inż. Ewa Piętka

Wpłynęło do Redakcji 12 stycznia 2011 r.

Omówienie

Artykuł opisuje nowe podejście do modelowania krzywych zamkniętych. Algorytm *SpliCo* (Spline Contour) pozwala na opisanie konturów na płaszczyźnie za pomocą odpowiednio zmodyfikowanych funkcji sklejanych. Ponieważ, z matematycznego punktu widzenia, nie jest możliwe opisanie takiej krzywej funkcją postanowiono podzielić problem opisania całego konturu jednocześnie na kilka mniejszych zadań, w których poszukuje się analitycznego opisu fragmentu konturu. Do tego celu zastosowano podejście znane z wykorzystania funkcji sklejanych do estymacji funkcji regresji.

W artykule przedstawiono dwa sposoby (analityczny i geometryczny) rozwiązania problemu nieciągłości i nieróżniczkowalności konturu w węzłach złączeń. Jako rozwiązanie referencyjne potraktowano wykorzystanie wielomianu czwartego rzędu jako funkcji sklejanej.

Jako punkt wyjścia dla rozwiązania analitycznego posłużyło równanie (1). Równanie to pozwala zachować ciągłość, jak również różniczkowalność odpowiedniego rzędu w punktach, w których następuje sklejenie dwóch wielomianów. Nie uwzględnia ono jednak sklejenia w węzłach początkowym i końcowym, które to w przypadku konturu stanowią ten sam punkt oraz w węzłach, w sąsiedztwie których krzywa nie jest funkcją.

Geometryczne rozwiązanie problemu opiera się na konstrukcji fragmentu okręgu lub okręgu i prostej wypełniającego przerwę między dwoma sąsiednimi wynikami aproksymacji (rys. 2). Powstały fragment ma na swoich końcach identyczne styczne, jak wspomniane zaproksymowane funkcje w tych samych punktach.

W artykule zaproponowano również funkcję oceniającą jakość znalezionego dopasowania konturu sklejanego (równanie (6)). Uwzględnia ona jednocześnie samą dokładność modelu, a także jego złożoność. Złożoność ta jest bowiem reprezentowana przez składnik będący liczbą węzłów (K), a także stopniem użytych wielomianów ($xPower$). Stała λ pozwala na odpowiednie sterowanie siłą wpływu złożoności modelu sklejanego na ocenę dokładności dopasowania. Mimo że użycie geometrycznej konstrukcji gładkiej krzywej aproksymującej daje największe błędy dopasowania (tabela 1), wygląd wyniku jest najbliższy oczekiwanemu (rys. 3).

Porównując czas generowania konturu z użyciem *SpliCo* oraz w pełni adaptacyjnej wersji krzywych B-sklejanych okazuje się, że użycie nowo zaproponowanego algorytmu pozwala skrócić czas o kilka rzędów wielkości.

Addresses

Tomasz PŁUCIENNIK: Silesian University of Technology, Institute of Computer Science, Akademicka 16, 44-100 Gliwice, Poland, tomek.pluciennik@gmail.com.
Marcin MICHALAK: Silesian University of Technology, Institute of Computer Science, Akademicka 16, 44-100 Gliwice, Poland, Marcin.Michalak@polsl.pl.