Proc. Estonian Acad. Sci. Phys. Math., 2000, **49**, 1, 21–27
https://doi.org/10.3176/phys.math.2000.1.03

ON THE CONSTRUCTION OF SMOOTHING SPLINES BY QUADRATIC PROGRAMMING

Natalia BUDKINA

Department of Mathematics, University of Latvia, Rainis Blvd 19, LV-1586 Riga, Latvia

Received 15 October 1998, in revised form 8 June 1999

Abstract. The problem of minimization of a smoothing functional under inequality constraints, which has a solution in the form of a natural spline, is reduced to the problem of quadratic programming with a positive semidefinite matrix. Using the results of quadratic programming, we obtain the modified simplex method for the solution of this problem by adding–removing interpolating knots of a spline.

Key words: smoothing spline, quadratic programming, simplex method.

1. INTRODUCTION

We consider the problem of approximation of an unknown function g by the information $z = (z_1, \dots, z_n)$ given at knots $\triangle_n : a \le t_1 < t_2 < \dots < t_n \le b$ when the information is imprecise:

$$|g(t_i) - z_i| \le \varepsilon_i, \; i = 1, \dots, n.$$

As approximation of g we take the solution of the minimization problem

$$J(f) = \int_a^b (f^{(q)}(t))^2 \, dt \longrightarrow \min_{\substack{f \in \mathbf{W}_2^q[a,b], \\ |f(t_i) - z_i| \le \varepsilon_i, \; i=1,\dots,n,}} \tag{1}$$

where $\mathbf{W}_2^q[a, b]$ is the Sobolev space.

If $n \le q$, then any polynomial p of degree $(q - 1)$, which satisfies the conditions $p(t_i) = z_i, i = 1, \dots, n$, gives the solution of the problem (1). If $n > q$ and no algebraic polynomial of degree $(q - 1)$ satisfies the inequalities

$|p(t_i) - z_i| \leq \varepsilon_i, i = 1, \ldots, n$, then the unique solution of the problem (1) exists. This solution is a natural spline of degree $(2q-1)$, which minimizes the functional $J(f)$ under the restraints (see, e.g., [1]). We assume in the sequel the uniqueness of the solution of (1).

Let $S_q(\triangle_n)$ be the space of all natural splines of degree $(2q-1)$ over the grid $\triangle_n$. It is known that $s \in S_q(\triangle_n)$ if and only if s can be written as

$$s(t) = \sum_{j=0}^{q-1} a_j t^j + \sum_{j=1}^{n} \frac{d_j}{(2q-1)!}(t-t_j)_+^{2q-1},$$

where the coefficients d_j satisfy the equalities

$$\sum_{j=1}^{n} d_j t_j^k = 0, \quad k = 0, \ldots, q-1.$$

If we denote by $s_i \in S_q(\triangle_n)$ the spline interpolating the value $(\delta_{ij})_{j=1}^n$ (δ_{ij} is the Kronecker symbol), then $s_1, \ldots, s_n$ constitute a basis of the space $S_q(\triangle_n)$ and any spline $s \in S_q(\triangle_n)$ can be written in the form

$$s(t) = \sum_{i=1}^{n} y_i s_i(t), \tag{2}$$

where $y_i = s(t_i)$.

2. THE SMOOTHING PROBLEM AS THE PROBLEM OF QUADRATIC PROGRAMMING

Taking into account that the solution of the problem (1) is a spline, we can restrict the class of functions $\mathrm{W}_2^q[a,b]$ by $S_q(\triangle_n)$ and restate the problem (1) as follows:

$$J(s) = \int_a^b (s^{(q)}(t))^2\, dt \longrightarrow \min_{\substack{s \in S_q(\triangle_n), \\ |s(t_i) - z_i| \leq \varepsilon_i,\, i=1,\ldots,n.}} \tag{3}$$

Differently from the problem (1) we have now the minimization problem in the n-dimensional space. Let us rewrite the smoothing functional J as a function of n variables. Any spline s can be uniquely defined by the n-dimensional vector of its values [see (2)]. Let us consider the new parameters of a spline s

$$h_j = s(t_j) - (z_j - \varepsilon_j), \; j = 1, \ldots, n. \tag{4}$$

We shall obtain an expression for the smoothing functional J with respect to h by using a well-known equality.

Lemma 2.1 ([1], p. 153). *For any function $f \in \mathbf{W}_2^q[a,b]$ and any natural spline s there holds*

$$\int_a^b f^{(q)}(t)s^{(q)}(t)\,dt = (-1)^q \sum_{i=1}^n d_i f(t_i).$$

Here d_i are the coefficients of the spline s.

Let us express a spline s with respect to h by using (2) and (4):

$$s(t) = \sum_{j=1}^n (z_j - \varepsilon_j + h_j)s_j(t). \tag{5}$$

We get its coefficients

$$d_i = \sum_{j=1}^n (z_j - \varepsilon_j + h_j)d_{ij}, \tag{6}$$

where $(d_{ij})_{i=1}^n$ are the coefficients of the basis spline s_j. Now, by Lemma 2.1 we obtain

$$J(h) = \int_a^b (s^{(q)}(t))^2\,dt = (-1)^q \sum_{i=1}^n (z_i - \varepsilon_i + h_i)d_i = (-1)^q \sum_{i=1}^n \sum_{j=1}^n h_i h_j d_{ij}$$

$$+2(-1)^q \sum_{i=1}^n \sum_{j=1}^n h_i d_{ij}(z_j - \varepsilon_j) + (-1)^q \sum_{i=1}^n \sum_{j=1}^n (z_i - \varepsilon_i)(z_j - \varepsilon_j)d_{ij}.$$

We introduce the matrix $D = ((-1)^q d_{ij})_{i,j=1,\dots,n}$ and the vector $c = (c_1, \dots, c_n)$, where $c_i = (-1)^q \sum\limits_{j=1}^n (z_j - \varepsilon_j)d_{ij}$, to rewrite the problem (3) in the matrix form

$$J(h) = hDh^T + 2ch^T \quad \longrightarrow \quad \min_{h \in \mathbf{R}_+^n,\, h \le 2\varepsilon}. \tag{7}$$

Note that $h \le 2\varepsilon$ means $h_i \le 2\varepsilon_i$, $i = 1, \dots, n$.

The problem (7) is a problem of quadratic programming under linear restrictions.

Lemma 2.2. *The matrix D is symmetric and positive semidefinite.*

Proof. The transformations of the expressions for $d_{ij} = d_i(s_j)$ and $d_{ji} = d_j(s_i)$ on the basis of Lemma 2.1

$$d_{ij} = \sum_{k=1}^n d_k(s_j)s_i(t_k) = (-1)^q \int_a^b s_j^{(q)}(t)s_i^{(q)}(t)\,dt,$$

$$d_{ji} = \sum_{k=1}^{n} d_k(s_i) s_j(t_k) = (-1)^q \int_a^b s_i^{(q)}(t) s_j^{(q)}(t)\, dt$$

prove the equality $d_{ij} = d_{ji}$.

The inequality $hDh^T \geq 0$ for any vector $h \in \mathbf{R}^n$ is proved by the identity

$$hDh^T = \int_a^b (s_h^{(q)}(t))^2\, dt, \tag{8}$$

where s_h is the interpolating spline for a vector h. The equality (8) is obtained by direct transformation

$$hDh^T = (-1)^q \sum_{i=1}^{n} h_i \sum_{j=1}^{n} h_j d_{ij} = (-1)^q \sum_{i=1}^{n} h_i d_i(s_h) = \int_a^b (s_h^{(q)}(t))^2\, dt.$$

3. THE EQUIVALENT PROBLEM OF "ALMOST" LINEAR PROGRAMMING

We use the results of quadratic programming for the solution of our problem in the form (7). We start with the Lagrangian function

$$F(h, \lambda) = hDh^T + 2ch^T + \lambda(h - 2\varepsilon)^T,$$

where $\lambda = (\lambda_1, \ldots, \lambda_n)$ is a vector of Lagrange multipliers.

A necessary condition for h to be a solution of the minimum problem (7) (see [2], p. 119) is the existence of some λ such that h and λ satisfy the conditions

$$\nabla_h F(h, \lambda) \geq 0, \qquad \nabla_h F(h, \lambda) h^T = 0, \qquad h \geq 0;$$

$$\nabla_\lambda F(h, \lambda) \leq 0, \qquad \nabla_\lambda F(h, \lambda) \lambda^T = 0, \qquad \lambda \geq 0.$$

Taking into account that

$$\nabla_h F(h, \lambda) = 2Dh^T + 2c + \lambda, \quad \nabla_\lambda F(h, \lambda) = h - 2\varepsilon,$$

by introducing slack nonnegative variables $\mu = (\mu_1, \ldots, \mu_n)$ and $\bar{h} = (\bar{h}_1, \ldots, \bar{h}_n)$:

$$\mu_i = 2(Dh^T)_i + 2c_i + \lambda_i, \quad \bar{h}_i = 2\varepsilon_i - h_i, \quad i = 1, \ldots, n,$$

we have the system of equations

$$\begin{gathered} 2Dh^T + 2c^T + \lambda - \mu = 0, \\ h + \bar{h} = 2\varepsilon,\ \mu^T h = 0,\ \lambda^T \bar{h} = 0. \end{gathered} \tag{9}$$

The existence of a nonnegative solution of this system follows from the existence of a solution of the problem (7).

The solution of the nonlinear system (9) is reduced to the solution of the "almost" linear programming problem of minimization of an auxiliary nonnegative variable u:

$$u \longrightarrow \min \tag{10}$$

$$\begin{gathered} 2Dh^T + 2c^T + \lambda - \mu + uE = 0, \\ h + \bar{h} = 2\varepsilon, \ \ \mu^T h = 0, \ \ \lambda^T \bar{h} = 0, \\ h \geq 0, \bar{h} \geq 0, \lambda \geq 0, \mu \geq 0, u \geq 0, \end{gathered}$$

where E is any vector with 0, 1, and -1. The existence of a nonnegative solution of (9) implies that zero is the solution of the problem (10).

Theorem 3.1. *Let the problem* (1) *have the unique solution. Then it is equivalent to the problem* (10), *i.e.*

– *the problem* (10) *has the unique solution too,*

– *the solution of* (1) *determines the solution of* (10) *by* (4) *and* (9),

– *the solution of* (10) *determines the solution of* (1) *by* (5).

Proof. As was proved, the problem (1) can be reduced to the problem (10), i.e. the vector h, which is obtained by (4) using the solution of (1), together with the additional variables $\bar{h}, \lambda, \mu$ [uniquely determined by h by (9)] satisfies the system of restrictions (10) when $u = 0$. To prove the uniqueness of this solution of (10), let us take any combination of $h, \bar{h}, \lambda, \mu$ satisfying (10) with $u = 0$. The values of h determine a spline $s \in S_q(\triangle_n)$ by (6). To prove that this spline gives a solution of (1) we check the conditions

$$\begin{aligned} d_i = 0 \quad & \text{if} \quad |s(t_i) - z_i| < \varepsilon_i, \\ (-1)^q d_i \geq 0 \quad & \text{if} \quad s(t_i) - z_i = -\varepsilon_i, \\ (-1)^q d_i \leq 0 \quad & \text{if} \quad s(t_i) - z_i = \varepsilon_i. \end{aligned} \tag{11}$$

As is known ([3], p. 66), these conditions are necessary and sufficient for s to be a solution of the problem (1).

According to (6), we have

$$(-1)^q d_i = (-1)^q \sum_{j=1}^{n} h_j d_{ij} + (-1)^q \sum_{j=1}^{n} (z_j - \varepsilon_j) d_{ij} = (Dh^T + c^T)_i.$$

Then, from (10) with $u = 0$, we obtain

$$(-1)^q d_i = \mu_i - \lambda_i, \ i = 1, \ldots, n.$$

Now it is easy to verify the conditions (11):
If $|s(t_i) - z_i| < \varepsilon_i$, i.e. $h_i \neq 0$, $\bar{h}_i \neq 0$, then $\lambda_i = \mu_i = 0$ and so $(-1)^q d_i = 0$.
If $s(t_i) - z_i = -\varepsilon_i$, i.e. $h_i = 0$, $\bar{h}_i = 2\varepsilon_i$, then $\lambda_i = 0$, $\mu_i \geq 0$ and so $(-1)^q d_i \geq 0$.
If $s(t_i) - z_i = \varepsilon_i$, i.e. $h_i = 2\varepsilon_i$, $\bar{h}_i = 0$, then $\lambda_i \geq 0$, $\mu_i = 0$ and so $(-1)^q d_i \leq 0$.

Therefore any solution of the problem (10) gives the unique solution of the problem (1). This proves also the uniqueness of the solution of (10).

4. CONSTRUCTION BY THE WOLFE–DAUGAVET METHOD

The problem (10) differs from problems of linear programming in having two simple nonlinear conditions $\mu^T h = 0$ and $\lambda^T \bar{h} = 0$. We solve this problem by using a modification of the simplex method which takes those conditions into account and which Wolfe and Daugavet suggested for such type of problems ([4]).

We give a short description of this algorithm. As an initial solution we take a spline which passes through the lower and the upper boundaries of the restrictions, i.e. $s(t_i) = z_i \pm \varepsilon_i$, $i = 1, \ldots, n$. Let $I_1 = \{i : s(t_i) = z_i - \varepsilon_i\}$, $I_2 = \{i : s(t_i) + \varepsilon_i\}$; note that $I_1 \cup I_2 = \{1, \ldots, n\}$. This choice corresponds to $h_i = 0$ for $i \in I_1$ and $\bar{h}_i = 0$ for $i \in I_2$. Then we have the initial values $\lambda_i = 0$ for $i \in I_1$ and $\mu_i = 0$ for $i \in I_2$. To obtain the values for other variables we consider

$$\rho_i = 2(Dh^T + c^T)_i,\ i = 1, \ldots, n, \text{ and } I_3 = \{i : \rho_i \geq 0\},\ I_4 = \{i : \rho_i < 0\}.$$

We take

$$u = \max\{|\rho_i| : i \in (I_1 \cap I_4) \cup (I_2 \cap I_3)\} = |\rho_{i_0}|, \quad \lambda_{i_0} = 0,\ \mu_{i_0} = 0;$$

$$\mu_i = 2(Dh^T + c^T)_i + u \text{ for } i \in I_1 \cap I_4; \quad \lambda_i = -2(Dh^T + c^T)_i + u \text{ for } i \in I_2 \cap I_3;$$

$$\mu_i = 2(Dh^T + c^T)_i \text{ for } i \in I_1 \cap I_3; \quad \lambda_i = -2(Dh^T + c^T)_i \text{ for } i \in I_2 \cap I_4.$$

The initial solution in the case $h_i = 2\varepsilon_i$, $\bar{h}_i = 0$, $i = 1, \ldots, n$ (the initial spline passes through the upper boundaries in all knots) is shown in Table 1.

Table 1. The initial table of the algorithm

	$\bar{h}_i, i \in I_3$	$\bar{h}_i, i \in I_4$	$\mu_i, i \in I_3 \backslash \{i_0\}$	$\mu_i, i \in I_4$	μ_{i_0}	λ_{i_0}
$h_i, i \in I_3$	–1	0	0	0	0	0
$h_i, i \in I_4$	0	–1	0	0	0	0
$\lambda_i, i \in I_3 \backslash \{i_0\}$	$\cdots$	$\cdots$	1	0	0	0
$\lambda_i, i \in I_4$	$\cdots$	$\cdots$	0	1	–1	1
u	$\cdots$	$\cdots$	0	0	–1	1

Each step of the method is a transformation of this table and hence also of the corresponding spline by adding and removing interpolating knots.

There are three possibilities of the location of h_i and $\bar{h}_i$ in the table (note that h_i and $\bar{h}_i$ cannot be in the upper part of the table simultaneously):

– h_i is in the upper part, it means that the spline passes through the upper boundary of the restriction in the knot t_i: $s(t_i) = z_i + \varepsilon_i$;

– $\bar{h}_i$ is in the upper part, it means that the spline passes through the lower boundary of the restriction in the knot t_i: $s(t_i) = z_i - \varepsilon_i$;

– h_i and $\bar{h}_i$ are in the lower part, it means that the spline passes between the boundaries of the restriction in the knot t_i: $|s(t_i) - z_i| < \varepsilon_i$.

Transformations of the table imply the following changes of the spline:

– if a new variable h_i or $\bar{h}_i$ appears in the upper part, then the spline is attracted to the lower boundary (if h_i appears) or to the upper boundary (if $\bar{h}_i$ appears) in the knot t_i;

– in another case, if a new variable h_i or $\bar{h}_i$ appears in the lower part, then the spline is released in the knot t_i.

The algorithm completes its work when the variable u appears in the upper part of the table. As was proved by Daugavet (see [4]), this occurs in a finite number of steps when the matrix D is positive semidefinite.

REFERENCES

1. Laurent, P. J. *Approximation et optimisation*. Hermann, Paris, 1972.
2. Seaty, T. L. *Nonlinear Mathematics*. McCraw-Hill, New York, 1964.
3. Ignatov, M. I. and Pevnij, A. B. *Natural Splines of Several Variables*. Nauka, Leningrad, 1991 (in Russian).
4. Daugavet, V. A. Modification of Wolfe's method. *Zh. Vychishl. Mat. Mat. Fiz.*, 1981, **21**, 504–508 (in Russian).

SILUVATE SPLAINIDE LEIDMINE RUUTPLANEERIMISÜLESANNET LAHENDADES

Natalia BUDKINA

Tõketega silumisülesanne, mis on teatavasti samaväärne ruutplaneerimisülesandega, on esitatud kujul, kus minimiseeritavat funktsionaali iseloomustav maatriks on sümmeetriline ja mittenegatiivne. Kasutades minimiseerimisülesande Lagrange'i funktsiooni omadusi on näidatud ruutplaneerimisülesande samaväärsust lineaarse funktsionaaliga minimiseerimisülesandega, milles kitsendused on lineaarsed või sisaldavad ainult tundmatute vektorite skalaarkorrutisi. Selliseid lineaarplaneerimisülesannetele lähedasi ülesandeid saab lahendada Wolfe'i–Daugaveti meetodiga, mis antud juhul annab lahendi lõpliku arvu sammudega.