GENERALIZED INTERPOLATION WITH POLYNOMIAL FUNCTIONS [1]

Elena Pelican and Constantin Popa [2]

Abstract

Least squares data fitting is an important task in many fields of applied mathematics ([3, 4]). Essentially, it means to find an element from a a given class of functions which best approximates a given set of points in the real plan, by also preserving their shape. In this paper we use for such an aproximation, classical and Bernstein polynomials. The (generally inconsistent) least sqaures problems so obtained are solved by both a Kaczmarz-like projection method, and an approximate orthogonalization technique (previously developed by the one of the authors in [1, 2]). Numerical experiments and comparisons are also provided.

1 Introduction

We shall present in this section the generalized interpolation (for short, **GI**) problem with respect to the linear subspace of real polynomial functions of degree less or equal than a given value $n \geq 1$, denoted by $\mathcal{P}_n$. We shall suppose that a basis $\{P_1, P_2, \dots, P_{n+1}\}$ is known in $\mathcal{P}_n$. With these assumptions we can state the **GI** problem as follows: if $N_1(x_1, y_1)$,...,$N_m(x_m, y_m)$ are $m \geq 1$ given points (nodes) in the real plane $I\!R^2$, find a polynomial $f \in \mathcal{P}_n$, such that the "orthogonal" distances $d_i, i = 1, ..., m$ defined by (see (3) and Figure (1))

$$d_i = \text{dist}\{N_i(x_i, y_i), (x_i, f(x_i)\} = |y_i - f(x_i)| \tag{1}$$

satisfy

$$\sum_{i=1}^{m} d_i^2 = \min! \tag{2}$$

Because $f \in \mathcal{P}_n$, we have

$$f(x) = a_1 P_1 + ... + a_{n+1} P_{n+1}, \tag{3}$$

which introduced in (1)-(2) gives

$$\sum_{i=1}^{m} \left[\sum_{j=1}^{n+1} a_j P_j(x_i) - y_i \right]^2 = \min! \tag{4}$$

Now, if we define the $m \times (n+1)$ matrix A, m-vector b and $(n+1)$-vector u by

$$A = \begin{bmatrix} P_1(x_1) & P_2(x_1) & P_3(x_1) & . & . & . & P_{n+1}(x_1) \\ P_1(x_2) & P_2(x_2) & P_3(x_2) & . & . & . & P_{n+1}(x_2) \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ P_1(x_m) & P_2(x_m) & P_3(x_m) & . & . & . & P_{n+1}(x_m) \end{bmatrix}, \tag{5}$$

$$b = (y_1, y_2, \dots, y_m)^t), \ u = (a_1, a_2, \dots a_{n+1})^t, \tag{6}$$

[1]The paper was supported by the CEEX-05-D11-25/2005 Grant.
[2]"Ovidius" University of Constantza, Faculty of Mathematics and Computer Science, Romania

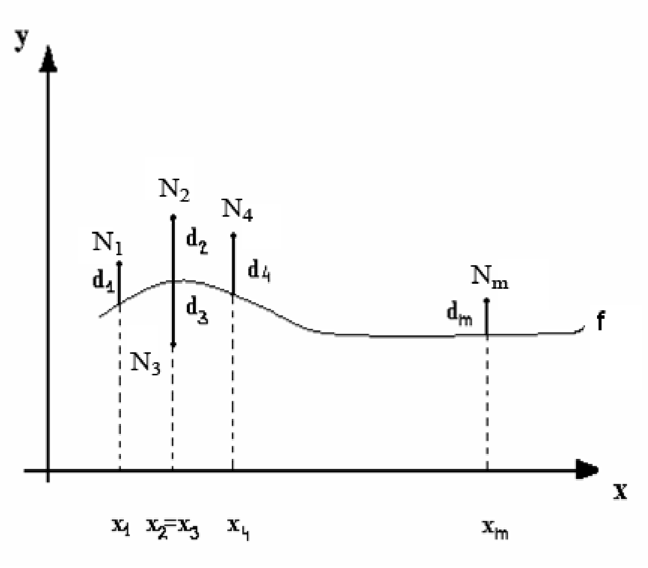


Figure 1: Generalized interpolation, $f \in \mathcal{P}_n$

the problem (2) (or (4)) becomes: find $u \in I\!R^{n+1}$ such that

$$\|Au - b\|^2 = \min! \tag{7}$$

i.e. is a classical *least-squares problem.*

Remark 1. *Usually the number* m *of the given data* $N_1, \dots, N_m$ *exceeds the number of fitting function parameters* $(a_1, a_2, \dots, a_{n+1}$*, in our case), i.e the problem* (7) *is overdetermined. Moreover, the abscissas* $x_1, \dots, x_m$ *of the given points* $N_1, \dots, N_m$ *have no special properties (i.e. may exist points* N_i *and* N_j*,* $i \neq j$ *with the same abscissas* $x_i = x_j$*). This means that the matrix* A *in* (5) *will have no more full row or column rank, thus* (7) *will be an inconsistent and rank-defficient least-squares problem.*

2 Classical and Bernstein polynomials

We used the following two different basis $\{P_1, P_2, \dots, P_{n+1}\}$ in $\mathcal{P}_n$:

I. Standard polynomials

$$P_1(x) = 1, P_i(x) = x^{i-1}, i = 2, \dots, n+1,\ x \in [a, b]\,; \tag{8}$$

II. Bernstein-like polynomials

$$P_i(x) = C_n^{i-1} x^{i-1} (1-x)^{n-i+1},\ i = 1, \dots, n+1,\ x \in [0, 1] \tag{9}$$

Remark 2. *In practical applications, the interval* $[a, b]$ *from* (8) *is defined such that*

$$\min_{1 \leq i \leq m} x_i = a < b = \max_{1 \leq i \leq m} x_i \tag{10}$$

where $x_1, \dots, x_m$ *are the abscissas of the points* $N_1, \dots, N_m$ *from* (1).

Remark 3. *The polynomials in* (9) *are restricted to the unit interval* $[0, 1] \subset I\!R$*. For an arbitrary interval* $[a, b]$ *(e.g. as in* (10)*) we consider the translation* φ

$$\varphi : [a, b] \to [0, 1]\,,\ \varphi(x) = \frac{x - a}{b - a} \tag{11}$$

and define the basis $\{\hat{P}_1, \ldots, \hat{P}_m\}$ *by*

$$\hat{P}_i(x) = P_i(\varphi(x)), \ x \in [a, b] \tag{12}$$

We have the following results.

Proposition 1. *Let* $\mathcal{P}_n([a, b])$ *be the vector space of the restrictions* $f|_{[a,b]}$ *of the elements* $f \in \mathcal{P}_n$ *with the basis* $\{P_1, \ldots, P_{n+1}\}$ *from* (9). *Then* $\{\hat{P}_1, \ldots, \hat{P}_m\}$ *is a basis in* $\mathcal{P}_n([a, b])$.

Proof. It will suffice to prove our proposition for $[a, b] = [0, 1]$. Similar arguments will prove it for an arbitrary interval $[a, b] \subset I\!R$. Let

$$\alpha_1 P_1(x) + \ldots + \alpha_{n+1} P_{n+1}(x) = 0, \ \forall x \in [0, 1] \tag{13}$$

and suppose that it exists an index $i \in \{1, \ldots m\}\}$ such that $\alpha_i \neq 0$. Then from (13) we get

$$E_1(x) + \alpha_i + E_2(x) = 0, \ \forall x \in (0, 1) \tag{14}$$

with

$$E_1(x) = \sum_{j=1}^{i-1} \alpha_j \frac{C_n^{j-1}}{C_n^{i-1}} \frac{(1-x)^{i-j}}{x^{i-j}}, \ E_2(x) = \sum_{j=i+1}^{n+1} \alpha_j \cdot \frac{C_n^{j-1}}{C_n^{i-1}} \frac{x^{j-i}}{(1-x)^{j-i}} \tag{15}$$

$x \in (0, 1)$, thus

$$\lim_{x \to 0, x > 0} E_1(x) = \lim_{x \to 1, x < 1} E_2(x) = \pm\infty; \ \lim_{x \to 0, x > 0} E_2(x) = \lim_{x \to 1 x < 1} E_1(x) = 0, \tag{16}$$

from which we obtain that

$$\alpha_1 = \ldots = \alpha_{i-1} = \alpha_{i+1} = \ldots = \alpha_{n+1} = 0 \tag{17}$$

and together with (14) we get $\alpha_i = 0$. But, this contradicts our initial assumption about α_i. It rests that all the α_i 's from the linear combination (13) are zero and the proof is complete.

Corollary *The two basis from* (8) *and* (12) *generate the finite dimensional vector space* $\mathcal{P}_n([a, b])$.

Remark 4. *The Bernstein-like polynomial (on* $[0, 1]$*),* $B_n(x) = \sum_{i=1}^{n+1} a_i P_i(x)$ *(with* $P_i(x)$ *from* (9)*) is a convex combination of the numbers* a_i*. For this we expect a "better shape preserving" by using them for the* **GI** *problem (see the next section of the paper).*

3 Numerical experiments

We used as numerical iterative solvers for the problem (7) the *Kaczmarz Extended* **(KE)** method and the *Right hand side Kovarik* **(rhs-Ko)** method. In what follows, we shall briefly describe the above mentioned methods. For detailes, see [1] and [2].

1. <u>Kaczmarz Extended **(KE)**</u>

   Let $a_i, \ i = 1, \ldots, m, \ \alpha_j, \ j = 1, \ldots, n$ be the i-th row and the j-th column in A respectively, and b_i the i-th component for b; assume that $a_i \neq 0, \ \alpha_j \neq 0, \ (\forall) \ i = 1, \ldots, m, \ j = 1, \ldots, n$. We define the mappings $f_i(b; \cdot), \ F(b; \cdot) : I\!R^n \to I\!R^n$ and $\varphi_j, \ \phi : I\!R^m \to I\!R^m$ by

$$f_i(b; u) = u - \frac{< u, a_i > -b_i}{\|a_i\|^2} a_i,$$

$$F(b;u) = (f_1 \circ \ldots \circ f_m)(b;u),\ u \in I\!R^n,$$

$$\varphi_j(y) = y - \frac{< y, \alpha_j >}{\|\alpha_j\|^2}\alpha_j,\ j = 1, \ldots, n,$$

$$\phi(y) = (\varphi_1 \circ \ldots \circ \varphi_n)(y),\ y \in I\!R^m.$$

Let $y^0 = b,\ x^0 \in I\!R^n$ (arbitrary) and $u^k \in I\!R^n$ already computed. The next iteration, u^{k+1}, is obtained as follows

$$y^{k+1} = \phi(y^k),$$

$$b^{k+1} = b - y^{k+1},\ u^{k+1} = F(b^{k+1}; u^k),\ k \geq 0.$$

In [??] it is proven that we have the following convergence result.
Result. Under the above assumption, we have that it exists $\lim_{k\to\infty} u^k = u^* \in \mathrm{LSS(A;b)}$; for $u^0 = 0,\ u^* = u_{LS}$.

2. Right hand side Kovarik **(rhs-KO)**

$$A_0 = A,\ b_0 = b$$

$$\text{for } k = 0, 1, ... \text{ do}$$

$$H_k = I - A_k A_k^t;\ \Gamma_k = I + \frac{1}{2}H_k$$

$$A_{k+1} = \Gamma_k A_k,\ b^{k+1} = \Gamma_k b^{(k)}$$

In [1] it is proven that we have the following convergence result.
Result. With the following scalling

$$A = A \cdot \frac{1}{\sqrt{\|AA^t\|_\infty + 1}};\ b = b \cdot \frac{1}{\sqrt{\|AA^t\|_\infty + 1}},$$

we have

$$\exists \lim_{k\to\infty} A_k^t b^k = u_{LS}.$$

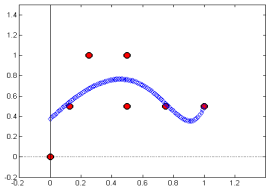
We considered in our experiments the following set of points

Problem 1. Input data							
x	0.0	0.125	0.25	0.5	0.5	0.75	1.0
y	0.0	0.5	1.0	1.0	0.5	0.5	0.5

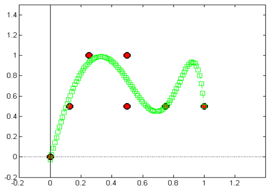
Problem 2									
x	0.0	0.1	0.3	0.35	0.45	0.45	0.6	0.8	0.9
y	0.2	0.1	0.3	0.8	0.6	0.8	0.3	0.9	1.0

The tests will be run for $\dim(\mathcal{P}_n) = 11$ for Problem 1 and $\dim(\mathcal{P}_n) = 13$ for Problem 2.

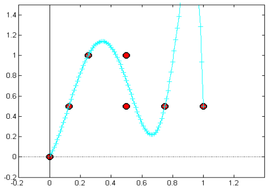
As it can be noticed, the results obtained by MATLAB's function are the worst in both cases, since it obtains graphs with very high "amplitutes", and in comparison the Berstein polynomials give us a better "shape" of the graph, as it was expected. With respect to the solver, in this particular cases, the **rhs-KO** is better than the **KE**.



Interpolation using **KE** solver

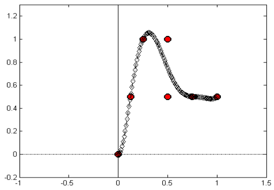


Interpolation using **rhs-KO** solver

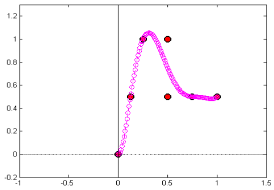


Interpolation using **MATLAB** function

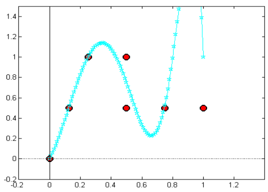
Figure 2: Problem 1, Standard polynomials



Interpolation using **KE** solver

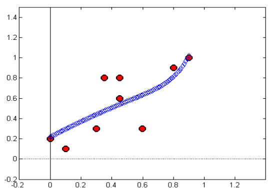


Interpolation using **rhs-KO** solver

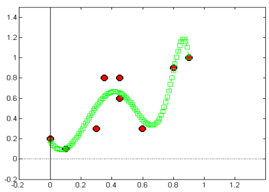


Interpolation using **MATLAB** function

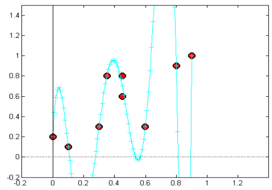
Figure 3: Problem 1, Bernstein polynomials



Interpolation using **KE** solver

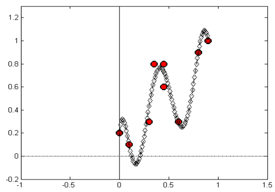


Interpolation using **rhs-KO** solver

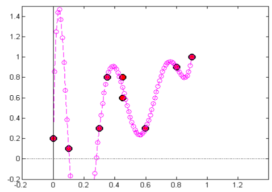


Interpolation using **MATLAB** function

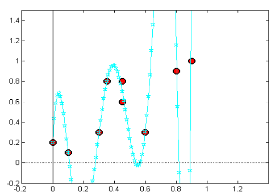
Figure 4: Problem 2, Standard polynomials



Interpolation using **KE** solver



Interpolation using **rhs-KO** solver



Interpolation using **MATLAB** function

Figure 5: Problem 2: Bernstein polynomials

REFERENCES

[1] Popa, C., *Extensions of block-projections methods with relaxation parameters to inconsistent and rank-defficient least-squares problems*, B I T **38**(1) (1998), 151-176.

[2] Popa, C., *Kovarik-like algorithms for Tikhonov regularization of dense inverse problems*, Lehrstuhlbericht **06-1**(2006), Institut für Informatik, Lehrstuhl für Informatik 10 (Systemsimulation), Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany.

[3] Watson G. A., *Least-squares fitting of circles and ellipses to measured data*, B I T **39(1)** (1999), 176-191.

[4] Varah, J.M., *Least squares data fitting with implicit functions*, B I T **36**(4) (1996), 842-854.