Global Optimization using Polynomial B-spline Form

Deepak Gawali
Systems & Control Engineering Department, Indian Institute Technology, Bombay
ddgawali2002@gmail.com

Abstract:

Many problems in engineering can be formulated as constrained optimization problems with multivariate polynomial objective functions. We propose algorithms based on polynomial B-spline form for constrained global optimization of multivariate polynomial functions. The proposed algorithms are based on a branch-and-bound framework. We tested the proposed basic constrained global optimization algorithms by considering test problems from systems analysis. The obtained results agree with those reported in literature.

Keywords: *Polynomial B-spline, System analysis, Constrained optimization.*

I. INTRODUCTION

A key problem in system analysis is to determine the minimum distance of a point to the surface defined by a polynomial constraint $f(x)=0$. We can pose it as the constrained optimization problem

$$\rho^* = \min_{x\in R^n} \| x \|_2^2$$
$$\text{s.t. } f(x)=0.$$

Most methods in literature for solving the minimum distance problem are based on LMI relaxation techniques [1][2]. These methods are based on a suitable representation of the polynomials in homogeneous forms. Generally the minimum distance problem reduces to constrained global optimization of nonlinear programming problems (NLP) is the study of how to find the best (optimum) solution to a problem. The constrained global optimization of NLPs is stated as follows:

$$\begin{aligned} &\min_{x\in \mathbf{x}} f(x) \\ &\text{s.t. } g_i(x) \le 0, i=1,2,\ldots,p \qquad (1)\\ &\quad h_j(x)=0,\ j=1,2,\ldots,q \end{aligned}$$

Branch-and-bound framework is commonly used for solving constrained global optimization problems [3]. For instance, several interval methods [4][5] use this framework to find the global minimum of a given NLP. In this work, we propose B-spline based algorithms for solving nonconvex nonlinear multivariate polynomial programming problems in systems and control, where the objective function f and constraints $(g_i \& h_j)$ are limited to being *polynomial* functions. The polynomial objective function and constraints in power form are transformed into the polynomial B-spline form [6][7]. Then, the B-spline coefficients provide a bound on the range of the objective function and constraints.

In this paper, we investigate three applications of the basic constrained global optimization algorithm: the robust stability analysis problem, the minimum distance problem, and the domain of attraction problem. These problems are reduced to strict inequalities (or equations) involving multivariate polynomials and solved using the proposed algorithm for constrained global optimization.

The merits of the proposed approach are: (i) it avoids evaluation of the objective function and constraints; (ii) an initial guess to start optimization is not required; only an initial search box bounding the region of interest; (iii) it guarantees that the global minimum is found to a user-specified accuracy, and (iv) prior knowledge of stationary points is not required.

II. BACKGROUND: POLYNOMIAL B-SPLINE FORM

Firstly, we present brief review of B-spline form, which is used as inclusion function to bound the range of multivariate polynomial in power from. The B-spline form is then used as basis of main zero finding algorithm in section 3.

We follow the procedure given in [7],[6] for B-spline expansion. Let $\varphi(t_1,\cdots t_l)$ be a multivariate polynomial in l real variables with highest degree $(m_1+\cdots m_l)$, (2).

$$\varphi(t_1,\cdots t_l)=\sum_{s_1=0}^{m_1}\cdots\sum_{s_l=0}^{m_l} a_{s_1\cdots s_l} t_1^{s_1}\cdots t_l^{s_l}. \tag{2}$$

2.1 Univariate polynomial

Lets consider univariate polynomial case first, (3)

$$\varphi(t)=\sum_{s=0}^{m} a_s t^s,\ t\in[p,q], \tag{3}$$

for degree d (i.e. order d+1) B-spline expansion where $d \geq m$, on compact interval I=[p,q]. We use $\psi_d(I,\mathbf{u})$ to represent the space of splines of degree d on the uniform grid partition known as *Periodic* or *Closed* knot vector, $\mathbf{u}$:

$$\mathbf{u} := \{t_0 < t_1 < \cdots < t_{k-1} < t_k\}, \tag{4}$$

Where $t_i := p + iy,\ 0 \le i \le k,\ k$ denotes B-spline segments and $y := (q-p)/k.$

Let $\mathbf{P}_d$ reflects the space of degree d splines. We then denote the space of degree d splines with C^{d-1} continuous on $[p,q]$ and defined on $\mathbf{u}$ as

$$\psi_d(I,\mathrm{u}) := \{\psi \in C^{d-1}(I) : \psi\,|\,[t_i, t_{i+1}] \in \mathrm{P}_d, i = 0,\cdots,k-1\}. \tag{5}$$

Since $\psi_d(I,\mathrm{u})$ is $(k+d)$ dimension linear space [8]. Therefore to construct basis of splines supported locally for $\psi_d(I,\mathrm{u}),$ we use few extra knots $t_{-d} \le \cdots \le t_{-1} \le p$ and $q \le t_{k+1} \le \cdots \le t_{k+d}$ at the ends in knot vector. These types of knot vectors are known as *Open* or *Clamped* knot vectors, (6). Since knot vector $\mathbf{u}$ is uniform grid partition, we choose $t_i := p + iy$ for $i \in \{-d,\cdots,-1\} \cup \{k+1,\cdots,k+d\},$

$$\mathbf{u} := \{t_{-d} \le \cdots \le t_{-1} \le p = t_0 < t_1 < \cdots < t_{k-1} < q = t_k \le t_{k+1} \le \cdots \le t_{k+d}\}. \tag{6}$$

The B-spline basis $\{B_i^d(t)\}_{i=1}^{k-1}$ of $\psi_d(I,\mathrm{u})$ is defined in terms of divided differences:

$$B_i^d(t) := (t_{i+d} - t_i)[t_i, t_{i+1}, \cdots, t_{i+d+1}](.-t)_+^d, \tag{7}$$

where $(.)_+^d$ represent the truncated power of degree d. This can be easily proven that

$$B_i^d(t) := \Omega_d\left(\frac{t-a}{h} - i\right), -d \le i \le k-1, \tag{8}$$

where

$$\Omega_d(t) := \frac{1}{d!}\sum_{i=0}^{d+1}(-1)^i \binom{d+1}{l}(t-l)_+^d, \tag{9}$$

$B_i^d(t) := (t_{i+d} - t_i)[t_i, t_{i+1}, \cdots, t_{i+d+1}](.-t)_+^d,$ is the polynomial B-spline of the degree d. The B-spline basis can be computed by a recursive relationship that is known as *Cox-deBoor* recursion formula

$$B_i^d(t) := \gamma_{i,d}(t) B_i^{d-1}(t) + (1 - \gamma_{i+1,d}(t)) B_{i+1}^{d-1}(t), d \ge 1, \tag{10}$$

where

$$\gamma_{i,d}(t) = \begin{cases} \dfrac{t-t_i}{t_{i+d}-t_i}, & \text{if } t_i \le t_{i+d}, \\ 0, & \text{otherwise,} \end{cases} \tag{11}$$

and

$$B_i^0(t) := \begin{cases} 1, & \text{if } t \in [t_i, t_{i+1}), \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

The set of spline basis $\{B_i^d(t)\}_{i=1}^{k-1}$ satisfies following interesting properties:

1. Each $B_i^d(t)$ is positive on its support $[t_i, t_{i+d+1}]$.
2. Set of spline basis $\{B_i^d(t)\}_{i=1}^{k-1}$ exhibits a partition of unity, i.e. $\sum_{i=1}^{k-1} B_i^d(t) = 1.$

The power basis functions $\{t^r\}_{r=0}^{m}$ in power form polynomial (3) can be represented in term of B-spline using following relation

$$t^s := \sum_{v=-d}^{k-1} \pi_v^{(s)} B_v^d(t), s = 0, \cdots, d, \tag{13}$$

and the symmetric polynomial $\pi_v^{(s)}$ defined as

$$\pi_v^{(s)} := \frac{\mathrm{Sym}_s(v+1, \cdots, v+d)}{k^s \binom{d}{s}}, \ s = 0, \cdots, d. \tag{14}$$

Then by substituting (13) in (3) we get B-spline extension of power form polynomial (3) as follows:

$$\varphi(t) := \sum_{s=0}^{m} a_s \sum_{v=-d}^{k-1} \pi_v^{(s)} B_v^d(t) = \sum_{v=-d}^{k-1} \left[\sum_{s=0}^{m} a_s \pi_v^{(s)} \right] B_v^d(t) = \sum_{v=-d}^{k-1} d_v B_v^d(t), \tag{15}$$

where

$$d_v := \sum_{s=0}^{m} a_s \pi_v^{(s)}. \tag{16}$$

2.2 Multivariate polynomial case

Lets consider next multivariate power form polynomial (17) for B-spline expansion

$$\varphi(t_1,\cdots t_l):=\sum_{s_1=0}^{k_1}\cdots\sum_{s_l=0}^{k_l}a_{s_1\cdots s_l}t_1^{s_1}\cdots t_l^{s_l}=\sum_{\mathbf{s}\le\mathbf{k}}a_{\mathbf{s}}t^{\mathbf{k}}, \tag{17}$$

where $\mathbf{s}:=(s_1,\cdots,s_l)$ and $\mathbf{k}:=(k_1,\cdots,k_l)$. By substituting (13) for each t^s, (17) can be written as

$$\begin{aligned}\varphi(t_1,t_2,...,t_l)&=\sum_{s_1=0}^{m_1}...\sum_{s_s=0}^{m_l}a_{s_1...s_l}\sum_{v_1=-d_1}^{k_1-1}\pi_{v_1}^{(s_1)}B_{v_1}^{d_1}(t_1)...\sum_{v_l=-d_l}^{k_l-1}\pi_{v_l}^{(s_l)}B_{v_l}^{d_l}(t_l),\\&=\sum_{v_1=-d_1}^{k_1-1}..\sum_{v_l=-d_l}^{k_s-1}\left(\sum_{s_1=0}^{m_1}...\sum_{s_l=0}^{m_l}a_{s_1...s_l}\pi_{v_1}^{(s_1)}...\pi_{v_l}^{(s_l)}\right)B_{v_1}^{d_1}(t_1)...B_{v_l}^{d_l}(t_l),\\&=\sum_{v_1=-d_1}^{k_1-1}...\sum_{v_l=-d_l}^{k_l-1}d_{v_1...v_l}B_{v_1}^{d_1}(t_1)...B_{v_l}^{d_l}(t_l),\end{aligned} \tag{18}$$

we can write (18) as

$$\varphi(t):=\sum_{\mathrm{v}\le\mathrm{k}}d_{\mathrm{v}}B_{\mathrm{v}}^{\mathrm{k}}(t). \tag{19}$$

where $\mathrm{v}:=(v_1,\cdots,v_l)$ and d_{v} is B-spline coefficient given as

$$d_{v_1...v_l}=\sum_{s_1=0}^{m_1}...\sum_{s_l=0}^{m_l}a_{s_1...s_l}\pi_{v_1}^{(s_1)}...\pi_{v_l}^{(s_l)}. \tag{20}$$

The B-spline expansion of (17) is given by (18). The derivative of polynomial can be found in a particular direction using the values of d_{v} i.e. B-spline coefficients of original polynomial for $\mathbf{y}\subseteq I$, the derivative of a polynomial $\varphi(t)$ with respect to t_r in polynomial B-spline form is (21),

$$\varphi_r'(\mathbf{y})=\frac{m_r}{\mathbf{u}_{\mathbf{s}+m_r+1}-\mathbf{u}_{\mathbf{s}+1}}\times\sum_{I\le\mathbf{m}_{r,-1}}\left[d_{\mathbf{s}_{r,1}}(\mathbf{y})-d_{\mathbf{s}}(\mathbf{y})\right]B_{\mathbf{m}_{r,-1},\mathbf{s}}(t),1\le r\le l,t\in\mathbf{y}, \tag{21}$$

where $\mathbf{u}$ is a knot vector. The partial derivative $\varphi_r'(\mathbf{y})$ now includes range enclosure for derivative of φ on $\mathbf{y}$. Lin and Rokne proposed (14) for symmetric polynomial and used closed or periodic knot vector (4). Due to change in knot vector from (4) to (6) we propose new form of (14) as follows,

$$\pi_v^{(s)} := \frac{\text{Sym}_s\left(v+1,\cdots,v+d\right)}{\begin{pmatrix} d \\ s \end{pmatrix}}. \tag{22}$$

2.3 B-spline range enclosure property

$$\varphi(t) := \sum_{i=1}^{m} d_i B_i^d(t), t \in \mathbf{y}. \tag{23}$$

Let (23) be a B-spline expansion of polynomial $q(t)$ in power form and $\overline{q}(\mathbf{y})$ denotes the range of the power form polynomial on subbox $\mathbf{y}$. The B-spline coefficients are collected in an array $D(\mathbf{y}) := (d_i(\mathbf{y}))_{i\in\Re}$ where $\Re := \{1,\cdots,m\}$. Then for $D(\mathbf{y})$ it holds

$$\overline{q}(\mathbf{y}) \subseteq D(\mathbf{y}) = [\min D(\mathbf{y}), \max D(\mathbf{y})]. \tag{24}$$

The range of the minimum and the maximum value of B-spline coefficients of multivariate polynomial B-spline expansion provides an range enclosure of the multivariate polynomial q on $\mathbf{y}$.

2.4 Subdivision procedure

We can improve the range enclosure obtained by B-spline expansion using subdivision of subbox $\mathbf{y}$. Let

$$\mathbf{y} := \left[\underline{\mathbf{y}}_1, \overline{\mathbf{y}}_1\right] \times \cdots \times \left[\underline{\mathbf{y}}_r, \overline{\mathbf{y}}_r\right] \times \cdots \times \left[\underline{\mathbf{y}}_l, \overline{\mathbf{y}}_l\right],$$

represent the box to be subdivided in the r th direction $(1 \le r \le l)$. Then two subboxes $\mathbf{y_A}$ and $\mathbf{y_B}$ are generated as follows

$$\mathbf{y_A} := \left[\underline{\mathbf{y}}_1, \overline{\mathbf{y}}_1\right] \times \cdots \times \left[\underline{\mathbf{y}}_r, m(\mathbf{y}_r)\right] \times \cdots \times \left[\underline{\mathbf{y}}_l, \overline{\mathbf{y}}_l\right],$$
$$\mathbf{y_B} := \left[\underline{\mathbf{y}}_1, \overline{\mathbf{y}}_1\right] \times \cdots \times \left[m(\mathbf{y}_r), \overline{\mathbf{y}}_r\right] \times \cdots \times \left[\underline{\mathbf{y}}_l, \overline{\mathbf{y}}_l\right],$$

where $m(\mathbf{y}_r)$ is a midpoint of $[\underline{\mathbf{y}}_r, \overline{\mathbf{y}}_r]$.

ISSN- 2394-5125 VOL 07, ISSUE 19, 2020

III. BASIC B-SPLINE CONSTRAINED GLOBAL OPTIMIZATION ALGORITHM SUMMARY

The basic B-spline algorithm for constrained global optimization of multivariate nonlinear polynomials, is similar to the one described in [9]. The algorithm can be summarized as follows.

Step 1: The basic algorithm uses the polynomial coefficients array of the objective function, A_o, the inequality constraints, A_{g_i} and the equality constraints, A_{h_j}. These coefficient arrays are stored in a cell structure A_c.

Step 2: A cell structure N_c contains degree vectors N, N_{g_i} and N_{h_j}, $i=0,\ldots,p$, $j=0,\ldots,q$, where these degree vector represents the degree of each variable occurring in objective function, the inequality constraints and the equality constraints respectively.

Step 3: The vector degree is used to compute the B-spline segment number, as the B-spline is constructed with the number of segments equal to order of the B-spline plus one. The vectors K_o, K_{g_i}, and K_{h_j} are computed using degree vectors N, N_{g_i} and N_{h_j} as $K=\mathrm{N}+2$ and stored in a cell structure K_c.

Step 4: Then we compute the B-spline coefficients of the objective, inequality and equality constraint polynomials on the initial search box $\mathbf{x}$. We store them in arrays $D_o(\mathbf{x})$, $D_{g_i}(\mathbf{x})$ and $D_{h_j}(\mathbf{x})$, respectively.

Step 5: We initialize the current minimum estimate $\tilde{p}$ to the maximum B-spline coefficient of the objective function on $\mathbf{x}$, i.e. $\tilde{p}=\max D_o(\mathbf{x})$.

Step 6: Next, we initialize a flag vector F with each component to zero as $F:=(F_1,\ldots,F_p,F_{p+1},\ldots,F_{p+q})=(0,\ldots,0)$. The flag vector F is used to make the algorithm more efficient. Consider, i^{th} inequality constraint is satisfied for x in a the box $\mathbf{b}$, i.e. $g_i(x)\le 0$ for $x\in\mathbf{b}$. Then there is no need to check again $g_i(x)\le 0$ for any subbox $\mathbf{b}_0\subseteq\mathbf{b}$. The same holds true for $h_j(x)$. To handle this information, we use flag vector $F=(F_1,\ldots,F_p,, F_{p+1},\ldots,F_{p+q})$ where the components F_f, takes the value 0 or 1, as follows

a) $F_f=1$ if the f^{th} inequality or equality constraint is satisfied for the box.

b) $F_f=0$ if the \$ f^{th} inequality or equality constraint has not yet been verified for the box.

Step 7: Initialize a working list L with the item $\mathrm{L}\leftarrow\{\mathbf{x},D_o(\mathbf{x}),D_{g_i}(\mathbf{x}),D_{h_j}(\mathbf{x}),F\}$, and a solution list L^{sol} to the empty list.

Step 8: Sort the list L in descending order of $(\min D_o(\mathbf{x}))$.

Step 9: Start iteration. If L is empty go to Step 14. Otherwise pick the last item from L, denote it as $\{\mathbf{b},D_o(\mathbf{b}),D_{g_i}(\mathbf{b}),D_{h_j}(\mathbf{b}),F\}$, and delete this item entry from L.

Step 10: Perform the cut-off test. As mentioned in Lemma 2, the minimum and maximum B-spline coefficients provide the range enclosure of the function. Let $\tilde{p}$ be the current minimum estimate, and $\{\mathbf{b}, D(\mathbf{b})\}$ be the current item for processing, for which $\tilde{p} \leq \min D(\mathbf{b})$. Then, this item surely cannot contain the global minimizer and can be discarded. Discard the item $\{\mathbf{y}, D_o(\mathbf{y}), D_{g_i}(\mathbf{y}), D_{h_j}(\mathbf{y}), F\}$ if $\min D_o(\mathbf{y}) > \tilde{p}$ and return to Step 9.

Step 11: Subdivision decision. If

$$(\text{wid}\,\mathbf{b}) \text{ and } (\max D_o(\mathbf{b}) - \min D_o(\mathbf{b})) < \epsilon$$

then add the item $\{\mathbf{b}, \min D_0(\mathbf{b})\}$ to L^{sol} and go to step 9. Else go to Step 12. Here ϵ is a tolerance number.

Step 12: Generate two sub boxes. Choose the subdivision direction along the longest direction of $\mathbf{b}$ and the subdivision point as the midpoint. Subdivide $\mathbf{b}$ into two subboxes $\mathbf{b}_1$ and $\mathbf{b}_2$ such that $\mathbf{b} = \mathbf{b}_1 \cup \mathbf{b}_2$.

Step 13: For $r = 1, 2$

1. Set $F^r := (F_1^r, \ldots, F_p^r, F_{p+1}^r, \ldots, F_{p+q}^r) = F$
2. Compute the B-spline coefficient arrays of objective and constraints polynomial on the box $\mathbf{b}_r$ and compute corresponding B-spline range enclosure $\mathrm{D}_o(\mathbf{b}_r), \mathrm{D}_{g_i}(\mathbf{b}_r)$,and $\mathrm{D}_{h_j}(\mathbf{b}_r)$ for objective and constraints polynomial.
3. Set $\tilde{p}_{local} = \min(\mathrm{D}_o(\mathbf{b}_r))$.
4. If $\tilde{p}_{local} > \tilde{p}$ go to sub Step 9.
5. for $i = 1, \ldots, p$ if $F_i = 0$ then
    a. If $\mathrm{D}_{g_i}(\mathbf{b}_r) > 0$ then go to sub Step 6.
    b. If $\mathrm{D}_{g_i}(\mathbf{b}_r) \leq 0$ then set $F_i^r = 1$.
6. for $j = 1, \ldots, q$ if $F_{p+j} = 0$ then
    a. If $0 \notin \mathrm{D}_{h_j}(\mathbf{b}_r)$ then go to sub Step 9.
    b. If $\mathrm{D}_{h_j}(\mathbf{b}_r) \subseteq [-\epsilon_{zero}, \epsilon_{zero}]$ then set $F_{p+j}^r = 1$.
7. If $F^r = (1, \ldots, 1)$ then set $\tilde{p} := \min(\tilde{p}, \max(\mathrm{D}_o(\mathbf{b}_r)))$.
8. Enter $\{\mathbf{b}_r, D_o(\mathbf{b}_r), D_{g_i}(\mathbf{b}_r), D_{h_j}(\mathbf{b}_r), F^r\}$ into the list L .
9. End (of *r*-loop)

Step 14: Set the global minimum to the current minimum estimate, $\hat{p} = \tilde{p}$.

Step 15: Find all those items in L^{sol} for which $\min D_o(\mathbf{b}) = p$. The first entries of these items are the global minimizer(s) $\mathbf{z}^{(i)}$.

ISSN- 2394-5125 VOL 07, ISSUE 19, 2020

Step 16: Return the global minimum $\hat{p}$ and all the global minimizers $\mathbf{z}^{(i)}$ found above.

IV. NUMERICAL RESULTS

The computations are done on a PC Intel i3-370M 2.40 GHz processor, 6 GB RAM, while the algorithms are implemented in MATLAB [11]. An accuracy $\epsilon = 10^{-6}$ is prescribed for computing the global minimum and minimizer(s). The time in second required to solve the problems is reported. This problem is from [1][12]. Consider the state-space system

$$\dot{z} = A(x)z(t),$$

where $z \in \mathbb{R}^n$ is the state vector and $x = (x_1, x_2, ..., x_n)' \in \mathbb{R}^n$ is the vector of uncertain parameters. Assuming $A(0)$ to be a *Hurwitz matrix*, the l_2 parametric stability margin is given by

$$\rho_2 = \sqrt{\rho^*} = \sqrt{min\{\rho_R, \rho_I\}}.$$

Where ρ_R is the solution of the equality constrained optimization problem

$$\rho_R = \min_{x \in \mathbb{R}^n} x_1^2 + x_2^2$$
$$\text{s.t. } \det[A(x)] = 0,$$

and ρ_I is the solution of another equality constrained optimization problem

$$\rho_I = \min_{x \in \mathbb{R}^n} x_1^2 + x_2^2$$
$$\text{s.t. } H_{n-1}[A(x)] = 0.$$

If $A(x)$ is a polynomial in x, then this minimum distance problem becomes a quadratic optimization problem. For the particular example reported in [12], we have

$$\det[A(x)] = -3x_1^3 - 7x_1^2x_2 - 2x_1x_2^2 - 2x_2^3 - 4x_1^2 + x_2^2 + 2x_1 + 2x_2 - 1,$$
$$H_{n-1}[A(x)] = -8x_1^3 - 4x_1o^2x_2 - 2x_1x_2^2 - 28x_1^2 + x_1x_2 - 3x_2 - 22x_1 - 7x_2 + 8,$$
$$\mathbf{x}_1 = [0, 0.5], \mathbf{x}_2 = [0, 0.5].$$

The proposed algorithm finds the global minimum to the first constrained optimization problem as

$$\rho_R = 0.2083,$$

while it finds the global minimum to the second constrained optimization problem as

$$\rho_I = 0.0664.$$

The global minimum of the stability margin is therefore

$$\rho^* = \min\{\rho_R, \rho_I\} = 0.0664,$$

giving the l_2 parametric stability margin as

$$\rho_2 = \sqrt{\rho^*} = 0.2576.$$

These results agree with those reported in [1][12]. The first problem is solved in 83.33 seconds and second in 81.48 seconds.

V. CONCLUSION

We proposed a constrained global optimization algorithm to solve the minimum distance problem using polynomial B-spline form as an inclusion function to bound the range of nonlinear multivariate polynomial function. The algorithm does not need any linearization or relaxation techniques and solves the problem to specified accuracy.

REFERENCES

[1] Floudas CA, Pardalos PM. *Handbook of Test Problems in Local and Global Optimization.* Kluwer Academic Publishers, Dordrecht, The Netherlands; 1999.

[2] Horst R, Pardalos PM. *Handbook of Global Optimization*. Kluwer Academic, Netherlands; 1995.

[3] Hansen E, Walster G. *Global Optimization Using Interval Analysis: Revised and Expanded*. Vol 264. (2, ed.). Marcel Dekker, New York; 2004.

[4] Jaulin L. Applied Interval Analysis: with Examples in Parameter and State Estimation. *Robust Control Robot Springer*. 2001

[5] Stahl V. *Interval Methods for Bounding the Range of Polynomials and Solving Systems of Nonlinear Equations. PhD Thesis, Johannes Kepler University, Linz*.; 1995.

[6] Lin Q, Rokne JG. Interval approximation of higher order to the ranges of functions. *Comput Math with Appl*. 1996;31(7):101-109.

[7] Lin Q, Rokne JG. Methods for bounding the range of a polynomial. *J Comput Appl Math*. 1995;58:193-199.

[8] DeVore RA, Lorentz GG. *Constructive Approximation*. Vol 303. Springer Science & Business Media, Berlin; 1993.

[9] Patil B. V., Nataraj P. S. V., Bhartiya S. Global optimization of mixed-integer nonlinear (polynomial) programming problems: the {B}ernstein polynomial approach. *Computing*. 2012;94(2-4):325-343.

ISSN- 2394-5125 VOL 07, ISSUE 19, 2020

[10] Mathworks Inc. *MATLAB Version 8.0.0.783 ({R} 2012 B)*. Inc., Natick, Massachusetts, United States; 2012.

[1] Henrion, D., Lasserre JB. Solving Nonconvex Optimization Problems. *Control Syst IEEE*. 2004;(24):72-83.

[2] Chesi G, Garulli A, Tesi A, Vicino A. Solving quadratic distance problems: an LMI-based approach. *IEEE Trans Automat Contr*. 2003;48(2):200-212.

[3] Horst R, Pardalos PM. *Handbook of Global Optimization*. Kluwer Academic, Netherlands; 1995.

[4] Hansen E, Walster G. *Global Optimization Using Interval Analysis: Revised and Expanded*. Vol 264. (2, ed.). Marcel Dekker, New York; 2004.

[5] Jaulin L. Applied Interval Analysis: with Examples in Parameter and State Estimation. *Robust Control Robot Springer*. 2001;1.

[6] Stahl V. *Interval Methods for Bounding the Range of Polynomials and Solving Systems of Nonlinear Equations. PhD Thesis, Johannes Kepler University, Linz*.; 1995.

[7] Lin Q, Rokne JG. Interval approximation of higher order to the ranges of functions. *Comput Math with Appl*. 1996;31(7):101-109.

[8] Lin Q, Rokne JG. Methods for bounding the range of a polynomial. *J Comput Appl Math*. 1995;58:193-199.

[9] DeVore RA, Lorentz GG. *Constructive Approximation*. Vol 303. Springer Science & Business Media, Berlin; 1993.

[10] Patil B. V., Nataraj P. S. V., Bhartiya S. Global optimization of mixed-integer nonlinear (polynomial) programming problems: the Bernstein polynomial approach. *Computing*. 2012;94(2-4):325-343.

[11] Mathworks Inc. *MATLAB Version 8.0.0.783 (R 2012 B)*. Inc., Natick, Massachusetts, United States; 2012.

[12] Chesi G, Tesi A, Vicino A, Genesio R. An LMI approach to constrained optimization with homogeneous forms. *Syst Control Lett*. 2001;42(1):11-19.