International Journal of Applied Engineering & Technology

BOUNDING ZEROS OF POLYNOMIAL SYSTEMS USING B-SPLINE EXPANSION AND HANSEN-SENGUPTA CONTRACTOR

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

ABSTRACT

Engineering applications such as computer-aided design, robotics, and electrical network requires an efficient computational technique of finding all roots of a system of nonlinear polynomial equations in s variables which lie within an s dimensional box. We are proposing an algorithm to obtain the roots of the polynomial system, it is based on the following technique:

1) Transformation of the original nonlinear algebraic equations into polynomial B-spline form;

2) Includes a pruning step using B-spline Hansen-Sengupta operator.

We compare the performance of the proposed B-spline Hansen-Sengupta operator with that of Interval Hansen-Sengupta operator using numerical examples, providing the superiority of the proposed approach.

Keywords: Polynomial B-spline form, Polynomial systems, Hansen-Sengupta operator.

I. INTRODUCTION

In [1][2]the authors proposed several root-finding algorithms for Finding solutions to the system of polynomial equations. In [3][4][5][6][7] the authors use interval methods to solve algebraic nonlinear equations. Methods addressing intervals approach provides interval enclosures with all roots of the polynomial systems using interval branch and bound strategy. Unfortunately, these methods also involves regular evaluation of polynomial functions and time-consuming [7].

To reduce the number of iterations operators like Newton, Krawczyk and Hansen-Sengupta are used for pruning the search space. Whereas to obtain interval enclosures for these pruning operators involves computation of derivatives [7]. Finding polynomial system derivatives using interval methods is similarly a time-consuming process. Again, to solve polynomial systems in [7][8] the authors combine Krawczyk operator and subdivision in B-spline and Bernstein basis respectively.

An algorithm was proposed based on B-spline expansion approach in combination with B-spline Hansen-Sengupta contractor to obtain the zeros of a polynomial systems i.e. roots of polynomials. The B-spline coefficient computation algorithm was suggested in [9] for global optimization. We are considering a new algorithm to solve a system of nonlinear polynomial equations by combining the B-spline Hansen-Sengupta algorithm, and the B-spline coefficient computation algorithm given in [9]. The idea behind B-spline expansion is to obtain polynomial B-spline form representation of given power form polynomial to obtain the bound on its range [9][10][11].

This paper is organized as follows: In the next section we give a brief introduction about the B-spline expansion of multivariate power form polynomial along range enclosure property and subdivision procedure. The interval Hansen-Sengupta operator is presented in section 3. In section 4, we present main zero finding algorithm to solve the system of the polynomial equation and use the B-spline Hansen-Sengupta operator algorithm for pruning the bounds. In section 5, we demonstrate the use of the suggested algorithm to solve a system of nonlinear polynomial equations by considering two numerical examples. The performance of proposed zero finding algorithm is compared with solver based on the INTLAB software. Finally, in the last section, we conclude.

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 [12][13] for B-spline expansion. Let $\varphi(t_1,\cdots t_l)$ be a multivariate polynomial in l real variables with highest degree $(m_1+\cdots m_l)$, (1).

$$\varphi(t_1,\cdots t_l)=\sum_{s_1=0}^{m_1}\cdots\sum_{s_l=0}^{m_l}a_{s_1\cdots s_l}t_1^{s_1}\cdots t_l^{s_l}. \qquad (1)$$

2.1 Univariate polynomial

Lets consider univariate polynomial case first, (2)

$$\varphi(t)=\sum_{s=0}^{m}a_s t^s,\ t\in[p,q], \qquad (2)$$

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, $\mathbf{u}$:

$$\mathbf{u}:=\{t_0<t_1<\cdots<t_{k-1}<t_k\}, \qquad (3)$$

Where $t_i:=p+iy,\ 0\leq i\leq k,\ k$ 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

$$\psi_d(I,\mathrm{u}):=\{\psi\in C^{d-1}(I):\psi|[t_i,t_{i+1}]\in \mathrm{P}_d, i=0,\cdots,k-1\}. \qquad (4)$$

Since $\psi_d(I,\mathrm{u})$ is $(k+d)$ dimension linear space [14]. Therefore to construct basis of splines supported locally for $\psi_d(I,\mathrm{u}),$ we use few extra knots $t_{-d}\leq\cdots\leq t_{-1}\leq p$ and $q\leq t_{k+1}\leq\cdots\leq t_{k+d}$ at the ends in knot vector. These types of knot vectors are known as *Open* or *Clamped* knot vectors, (5). Since knot vector $\mathbf{u}$ is uniform grid partition, we choose $t_i:=p+iy$ for $i\in\{-d,\cdots,-1\}\cup\{k+1,\cdots,k+d\}$,

$$\mathbf{u}:=\{t_{-d}\leq\cdots\leq t_{-1}\leq p=t_0<t_1<\cdots<t_{k-1}<q=t_k\leq t_{k+1}\leq\cdots\leq t_{k+d}\}. \qquad (5)$$

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:

$$B_i^d(t):=(t_{i+d}-t_i)[t_i,t_{i+1},\cdots,t_{i+d+1}](.-t)_+^d, \qquad (6)$$

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

$$B_i^d(t):=\Omega_d\left(\frac{t-a}{h}-i\right),-d\leq i\leq k-1, \qquad (7)$$

where

$$\Omega_d(t):=\frac{1}{d!}\sum_{i=0}^{d+1}(-1)^i\binom{d+1}{l}(t-l)_+^d, \qquad (8)$$

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

$$B_i^d(t) := \gamma_{i,d}(t) B_i^{d-1}(t) + (1 - \gamma_{i+1,d}(t)) B_{i+1}^{d-1}(t), d \geq 1, \quad (9)$$

where

$$\gamma_{i,d}(t) = \begin{cases} \dfrac{t - t_i}{t_{i+d} - t_i}, & \text{if } t_i \leq t_{i+d}, \\ 0, & \text{otherwise}, \end{cases} \quad (10)$$

and

$$B_i^0(t) := \begin{cases} 1, & \text{if } t \in [t_i, t_{i+1}), \\ 0, & \text{otherwise.} \end{cases} \quad (11)$$

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 (2) can be represented in term of B-spline using following relation

$$t^s := \sum_{v=-d}^{k-1} \pi_v^{(s)} B_v^d(t), s = 0, \cdots, d, \quad (12)$$

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

$$\pi_v^{(s)} := \frac{\mathrm{Sym}_s(v+1, \cdots, v+d)}{k^s \binom{d}{s}}, \ s = 0, \cdots, d. \quad (13)$$

Then by substituting (12) in (2) we get B-spline extension of power form polynomial (2) as follows:

$$\varphi(t) := \sum_{s=0}^m a_s \sum_{v=-d}^{k-1} \pi_v^{(s)} B_v^d(t) = \sum_{v=-d}^{k-1} \left[\sum_{s=0}^m a_s \pi_v^{(s)} \right] B_v^d(t) = \sum_{v=-d}^{k-1} d_v B_v^d(t), \quad (14)$$

where

$$d_v := \sum_{s=0}^m a_s \pi_v^{(s)}. \quad (15)$$

2.2 Multivariate polynomial case

Lets consider next multivariate power form polynomial (16) for B-spline expansion

$$\varphi(t_1, \cdots t_l) := \sum_{s_1=0}^{k_1} \cdots \sum_{s_l=0}^{k_l} a_{s_1 \cdots s_l} t_1^{s_1} \cdots t_l^{s_l} = \sum_{\mathbf{s} \leq \mathbf{k}} a_{\mathbf{s}} t^{\mathbf{k}}, \quad (16)$$

where $\mathbf{s} := (s_1, \cdots, s_l)$ and $\mathbf{k} := (k_1, \cdots, k_l)$. By substituting (12) for each t^s, (16) can be written as

$$\varphi(t_1,t_2,...,t_l)=\sum_{s_1=0}^{m_1}...\sum_{s_s=0}^{m_l}a_{s_1...s_l}\sum_{v_1=-d_1}^{k_1-1}\pi_{v_1}^{(s_1)}B_{v_1}^{d_1}(t_1)...\sum_{v_l=-d_l}^{k_l-1}\pi_{v_l}^{(s_l)}B_{v_l}^{d_l}(t_l),$$

$$=\sum_{v_1=-d_1}^{k_1-1}...\sum_{v_l=-d_l}^{k_s-1}\left(\sum_{s_1=0}^{m_1}...\sum_{s_l=0}^{m_l}a_{s_1...s_l}\pi_{v_1}^{(s_1)}...\pi_{v_l}^{(s_l)}\right)B_{v_1}^{d_1}(t_1)...B_{v_l}^{d_l}(t_l),\ (17)$$

$$=\sum_{v_1=-d_1}^{k_1-1}...\sum_{v_l=-d_l}^{k_l-1}d_{v_1...v_l}B_{v_1}^{d_1}(t_1)...B_{v_l}^{d_l}(t_l),$$

we can write (17) as

$$\varphi(t):=\sum_{\mathrm{v}\le\mathrm{k}}d_{\mathrm{v}}B_{\mathrm{v}}^{\mathrm{k}}(t). \qquad (18)$$

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

$$d_{v_1...v_l}=\sum_{s_1=0}^{m_1}...\sum_{s_l=0}^{m_l}a_{s_1...s_l}\pi_{v_1}^{(s_1)}...\pi_{v_l}^{(s_l)}. \qquad (19)$$

The B-spline expansion of (16) is given by (17). 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 (20),

$$\varphi_r'(\mathbf{y})=\frac{m_r}{\mathbf{u}_{\mathbf{s}+m_r+1}-\mathbf{u}_{\mathbf{s}+1}}\times\sum_{I\le\mathbf{m}_{r,-1}}\left[d_{\mathbf{s}_{r,1}}(\mathbf{y})-d_{\mathbf{s}}(\mathbf{y})\right]B_{\mathbf{m}_{r,-1},\mathbf{s}}(t),1\le r\le l,t\in\mathbf{y}, \qquad (20)$$

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 (13) for symmetric polynomial and used closed or periodic knot vector(3). Due to change in knot vector from (3) to (5) we propose new form of (13) as follows,

$$\pi_v^{(s)}:=\frac{\mathrm{Sym}_s(v+1,\cdots,v+d)}{\binom{d}{s}}. \qquad (21)$$

2.3 B-spline range enclosure property

$$\varphi(t):=\sum_{i=1}^{m}d_iB_i^d(t),t\in\mathbf{y}. \qquad (22)$$

Let (22) be a B-spline expansion of polynomial $q(t)$ in power form and $\bar{q}(\mathbf{y})$ 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

$$\bar{q}(\mathbf{y})\subseteq D(\mathbf{y})=[\min D(\mathbf{y}),\max D(\mathbf{y})]. \qquad (23)$$

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 $\mathbf{y}_\mathbf{A}$ and $\mathbf{y}_\mathbf{B}$ are generated as follows

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

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

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

III. B-SPLINE HANSEN-SENGUPTA OPERATOR ALGORITHM

The interval Newton operator is given in [15] as

$$\mathbf{N}\left(\mathbf{p}, \mathbf{y}, \check{y}\right) = \check{y} - \frac{p(\check{y})}{\mathbf{p}'(\mathbf{y})}. \qquad (24)$$

Let $p : \mathbf{y} = [\underline{y}, \overline{y}] \to \square$ be a continuously differentiable multivariate polynomial on $\mathbf{y}$, let that there exists $y^* \in \mathbf{y}$ such that $p\left(y^*\right) = 0$, and suppose that $\check{y} \in \mathbf{y}$. Then, since the mean value theorem implies

$$0 = p\left(y^*\right) = p\left(\check{y}\right) + p'(\xi)\left(y^* - \check{y}\right),$$

therefore $y^* = \check{y} - \dfrac{p\left(\check{y}\right)}{p'(\xi)}$ for some $\xi \in \mathbf{y}$. If $\mathbf{p}'(\mathbf{y})$ is any interval extension of the derivative of p over $\mathbf{y}$, then

$$y^* \in \check{y} - \frac{p\left(\check{y}\right)}{\mathbf{p}'(\mathbf{y})}, \quad \check{y} \in \mathbf{y}. \qquad (25)$$

Because of (25), any solution of $p(y) = 0$ that are in $\mathbf{y}$ must also be in $\mathrm{N} = \left(\mathbf{p}, \mathbf{y}, \check{y}\right)$ and therefore (25) is the basis of the univariate Newton method (24).

The univariate Newton method (24) can be extended as a Multivariate Newton method which execute an iteration equation similar to equation (24).

Suppose now that $y \in \square^s$ and $f(y) \in \square^n$ (continuously differentiable nonlinear) polynomial equations in s unknowns, and let that $\check{y} \in \square^S$. Then a basic formula for multivariate Newton method is

$$\mathbf{N}\left(f, \mathbf{y}, \check{y}\right) = \check{y} + \mathbf{w}, \qquad (26)$$

where $\mathbf{w}$ is a vector of interval bounding all zeros w of system $Aw = -f\left(\check{y}\right)$, as $A \in \mathbf{f}'(\mathbf{y})$, such that $\mathbf{f}'(\mathbf{y})$ is the

Jacobi matrix f interval extension over $\mathbf{y}$.

As interval Newton operator given by(26), we can write as follows:

(27)

$$\mathbf{f}'(\mathbf{x})\left(\mathbf{N}\left(\mathbf{f},\mathbf{x},\overset{\vee}{x}\right)-\overset{\vee}{x}\right)=-f\left(\overset{\vee}{x}\right).$$

Preconditioning equation (27) with Y, as midpoint inverse of an interval extension of the Jacobi matrix $\mathbf{f}'(\mathbf{x})$, i.e. $Y=\left\{\text{mid}\,\mathbf{f}'(\mathbf{x})\right\}^{-1}$ gives

$$0=p\left(y^*\right)=p\left(\overset{\vee}{y}\right)+p'(\xi)\left(y^*-\overset{\vee}{y}\right),$$

therefore $y^*=\overset{\vee}{y}-\dfrac{p\left(\overset{\vee}{y}\right)}{p'(\xi)}$ for some $\xi\in\mathbf{y}$. If $\mathbf{p}'(\mathbf{y})$ is any interval extension of the derivative of p over $\mathbf{y}$, then

(28)

$$y^*\in\overset{\vee}{y}-\frac{p\left(\overset{\vee}{y}\right)}{\mathbf{p}'(\mathbf{y})},\quad \overset{\vee}{y}\in\mathbf{y}.$$

Because of (28), any solution of $p(y)=0$ that are in $\mathbf{y}$ must also be in $\mathrm{N}=\left(\mathbf{p},\mathbf{y},\overset{\vee}{y}\right)$ and therefore (28) is the basis of the *univariate* Newton method (24).

The *univariate* Newton method (24) can be extended as a *Multivariate* Newton method which execute an iteration equation similar to equation (24).

Suppose now that $y\in\square^{s}$ and $f(y)\in\square^{n}$ (continuously differentiable nonlinear) polynomial equations in s unknowns, and let that $\overset{\vee}{y}\in\square^{S}$. Then a basic formula for multivariate Newton method is

$$\mathbf{N}\left(f,\mathbf{y},\overset{\vee}{y}\right)=\overset{\vee}{y}+\mathbf{w}, \qquad (29)$$

where $\mathbf{w}$ is a vector of interval bounding all zeros w of system $Aw=-f\left(\overset{\vee}{y}\right)$, as $A\in\mathbf{f}'(\mathbf{y})$, such that $\mathbf{f}'(\mathbf{y})$ is the Jacobi matrix f interval extension over $\mathbf{y}$.

We can write (29) as follows:

(30)

$$\mathbf{f}'(\mathbf{x})\left(\mathbf{N}\left(\mathbf{f},\mathbf{x},\overset{\vee}{x}\right)-\overset{\vee}{x}\right)=-f\left(\overset{\vee}{x}\right).$$

Preconditioning equation (30) with Y, as midpoint inverse of an interval extension of the Jacobi matrix $\mathbf{f}'(\mathbf{x})$, i.e. $Y=\left\{\text{mid}\,\mathbf{f}'(\mathbf{x})\right\}^{-1}$ gives

(31)

$$Y\,\mathbf{f}'(\mathbf{x})\left(\mathbf{N}\left(\mathbf{f},\mathbf{x},\overset{\vee}{x}\right)-\overset{\vee}{x}\right)=-Yf\left(\overset{\vee}{x}\right).$$

Changing the notation $\mathbf{N}\left(\mathbf{f},\mathbf{x},\overset{\vee}{x}\right)$ to $\mathbf{H}\left(\mathbf{f},\mathbf{x},\overset{\vee}{x}\right)$ and defining,

$$M=Y\,\mathbf{f}'(\mathbf{x}),b=Yf\left(\overset{\vee}{x}\right),$$

the interval Gauss-Seidel procedure proceeds component by component to give the iteration

(32)

$$\mathbf{H}\left(\mathbf{f},\mathbf{x}^k,\overset{\vee}{x}^k\right)_i=\overset{\vee}{x}{}_i^k-\frac{b_i+\sum_{j=1}^{i-1}Y_{ij}\left(\mathbf{x}^{k+1}-\overset{\vee}{x}^{k+1}\right)+\sum_{j=i+1}^{n}Y_{ij}\left(\mathbf{x}^{k+1}-\overset{\vee}{x}^{k+1}\right)}{Y_{ii}},$$

(33)

$$\mathbf{x}_i^{k+1}=\mathbf{H}\left(\mathbf{f},\mathbf{x}^k,\overset{\vee}{x}^k\right)_i\cap\mathbf{x}_i^k,$$

for $k=0,1,\cdots,n$ and $\overset{\vee}{x}^k\in\mathbf{x}^k$.

In this iteration after the ith component of $\mathbf{H}\left(\mathbf{f},\mathbf{x}^k,\overset{\vee}{x}^k\right)$ is computed using (32), the intersection (33) is performed.

The result is then used to calculate subsequent component of $\mathbf{H}\left(\mathbf{f},\mathbf{x}^k,\overset{\vee}{x}^k\right)$.

We now present the algorithm for bounding zeros of polynomial systems similar to [16],

Algorithm 3.1: Subdivision Algorithm for Solving a Polynomial Systems

Input: Here A_c is a cell structure containing the coefficients array a_I of the polynomials in the power form. N_c is a cell structure, containing degree vector, N_I which contains degree of each variable in polynomial function. Initial bound $\mathbf{x}$ of each variable and tolerance limit ò.

Output: The zero(s) of f in $\mathbf{x}$ or $\{\varnothing\}$ as no solution exists in $\mathbf{x}$.

Begin Algorithm

1 {Compute the B-spline coefficients}

Compute the B-spline coefficients $D_i(\mathbf{x})$ of given n polynomials on the initial box $\mathbf{x}$, where $i=1,2,\cdots,n.$ (Use algorithms given in [9])

2 {Initialize iteration number}

Set $k=0,\ \mathbf{x}^{(0)}=\mathbf{x}.$

3 {Compute $f(\overset{\vee}{x})$ }

Choose $\check{x} = \text{mid}(\mathbf{x}^{(k)})$ and obtain the value of $f(\check{x})$ directly from the B-spline coefficient value at the vertex of $\text{mid}(\mathbf{x}^{(k)})$.

4 {Compute $\mathbf{f}'(\mathbf{x})$ }

Use the B-spline coefficients of f on $\mathbf{x}^{(k)}$, to compute the B-spline coefficients of all the first partial derivatives of f on $\mathbf{x}^{(k)}$ via (21). From the minimum and maximum B-spline coefficients of the first derivative, construct their range enclosure interval, and form the interval Jacobian matrix $\mathbf{f}'(\mathbf{x})$.

5 {Compute the precondition matrix Y }

Compute the preconditioning matrix Y as

$Y = \left\{\text{mid}\,\mathbf{f}'\left(\mathbf{x}^{k}\right)\right\}^{-1}$.

6 {B-spline Hansen-Sengupta operator}

Compute the value of B-spline Hansen-Sengupta operator H and update the solution,

Set $M = Y \times \mathbf{f}'(\mathbf{x})$, $b = Y \times f\left(\check{x}\right)$ and $n = s$.

for i=1 to n do

if i == 1 then

$$H(i) = \check{x}(i) - \frac{b(i) + M(i,2:n) \times \left\{\mathbf{x}(2:n) - \check{x}(2:n)\right\}}{M(i,i)},$$

$$\mathbf{x}^{(k)}(i) = H(i) \cap \mathbf{x}^{(k)}(i),$$

else

$$H(i) = \check{x}(i) - \frac{b(i) + \phi + \gamma}{M(i,i)},$$

where

$$\phi = M(i,1:i-1) \times \left\{\mathbf{x}(i:i-1) - \check{x}(i:i-1)\right\}$$

and

$$\gamma = M(i,i+1:n) \times \left\{\mathbf{x}(i,i+1:n) - \check{x}(i,i+1:n)\right\},$$

$$\mathbf{x}^{(k)}(i) = H(i) \cap \mathbf{x}^{(k)}(i),$$

end

end

7 {Return $\{\varnothing\}$ }

If $\mathbf{x}^{(k)} = 0$, then return $\{\varnothing\}$ as solution and *exit* algorithm.

8 {Termination}

If $\mathbf{x}^{(k)} < \grave{o}$, then return $\mathbf{x}^{(k+1)}$ as solution and *exit* algorithm.

9 Set $k = k+1$ and **go to step 3.**

End Algorithm

IV. NUMERICAL RESULTS

We consider the two problems to test and compare the performance of B-spline Hansen-Sengupta operator (BHSO) over the interval Hansen-Sengupta operator (IHSO). The performance metrics are taken as the number of iterations and computational time (in seconds). Our MATLAB source code implementation of interval Hansen-Sengupta operator based on INTLAB.

Example 1: This example is taken from [17]. This is a problem with 4 variables. The polynomial systems is given by

$$x_1 + x_2 + x_3 + x_4 + 1 = 0,$$
$$x_1 + x_1x_2 + x_2x_3 + x_3x_4 + x_4 = 0,$$
$$x_1x_2 + x_1x_2x_3 + x_2x_3x_4 + x_3x_4 + x_1x_4 = 0,$$
$$x_1x_2x_3 + x_1x_2x_3x_4 + x_2x_3x_4 + x_3x_4x_1 + x_1x_2x_4 = 0.$$

and the bounds on the variables are

$$x_1 = [0.95, 1.05],\ x_2 = [0.95, 1.05],\ x_3 = [-2.65, -2.6],\ x_4 = [-0.4, -0.37].$$

The results of algorithm are tabulated in Table 1.

Table 1: Roots of Example 1.

Roots	
x_1	1
x_2	1
x_3	-2.6180
x_4	-0.3819

Table 2: Comparison of performance between BHSO and IHSO.

	Number of Iterations	**Computation Time (Sec.)**
BHSO	15	3.77
IHSO	4	3.06

Example 2: This example is taken from [17]. This is a problem with 5 variables. The polynomial systems is given by

$$x_1 + x_2 + x_3 + x_4 + x_5 + 1 = 0,$$
$$x_1 + x_1x_2 + x_2x_3 + x_3x_4 + x_4x_5 + x_5 = 0,$$
$$x_1x_2 + x_1x_2x_3 + x_2x_3x_4 + x_3x_4x_5 + x_4x_5 + x_1x_5 = 0,$$
$$x_1x_2x_3 + x_1x_2x_3x_4 + x_2x_3x_4x_5 + x_3x_4x_5 + x_1x_4x_5 + x_1x_2x_5 = 0.$$
$$x_1x_2x_3x_4 + x_1x_2x_3x_4x_5 + x_2x_3x_4x_5 + x_3x_4x_5x_1 + x_1x_2x_4x_5 + x_1x_2x_3x_5 = 0.$$

and the bounds on the variables are

$$x_1 = [0.95, 1.05],\ x_2 = [-3.75, -3.70],\ x_3 = [-0.28, -0.25],\ x_4 = [0.95, 1.01],\ x_5 = [0.95, 1.01].$$

From Table 3 we observe that the existing interval Newton operator require 4 iterations to bound the roots of the polynomial systems with the accuracy of $\grave{o}=10^{-06}$. The proposed B-spline Newton operator algorithm computes the result in 7 iterations within the same accuracy. The computational time required for the proposed B-spline Newton operator is 1.23 seconds, whereas the interval Newton operator method requires computational time 1.44 seconds. The results of algorithm are tabulated in Table 3.

Table 3: Roots of Example 2.

Roots	
x_1	1
x_2	-3.7320
x_3	-0.2679
x_4	1
x_5	1

Table 4: Comparison of performance between BHSO and IHSO.

	Number of Iterations	**Computation Time (Sec.)**
BHSO	14	6.01
IHSO	4	3.83

V. CONCLUSION

In this paper we presented an algorithm for contracting the search domain using B-spline Hansen-Sengupta. The computational examples demonstrate that the algorithm suggested quite effectively solves the polynomial system but requires more number of iterations due to over estimation in range enclosure of the first partial derivatives of the original polynomial.

REFERENCES

[1] Kolev L. An interval method for global nonlinear analysis. IEEE Trans Circuits Syst I Fundam Theory Appl 2000;47:675–83.

[2] Jäger C, Ratz D, iyegyer K, Rats L. A combined method for enclosing all solutions of nonlinear systems of polynomial equations. Reliab Comput 1995;1:41–64. https://doi.org/10.1007/BF02390521.

[3] Hansen E, Walster G. Global optimization using interval analysis: revised and expanded. vol. 264. Marcel Dekker, New York; 2004.

[4] Moore R. E. Methods and Applications of Interval Analysis. SIAM, Philadelphia; 1979.

[5] Hammer R, Hocks M, Kulisch U, Ratz D. Numerical Toolbox for Verified Computing I, Basic Numerical Algorithms, Theory, Algorithms, and Pascal-XSC Programs. Number 21 in Springer Series in Computational Mathematics 1991.

[6] Nataraj P.S.V., Sondur S. Construction of bode envelopes using REP based range finding algorithms. Int J Autom Comput 2011;8:112–21.

[7] Arounassalame M. Analysis of Nonlinear Electrical Circuits Using Bernstein Polynomials. Int J Autom Comput 2012;9:81–6.

[8] Michel D, Zidna A. Interval-Krawczyk Approach for Solving Nonlinear Equations Systems in B-spline Form. Model. Comput. Optim. Inf. Syst. Manag. Sci., Springer; 2015, p. 455–65.

[9] Gawali D.D., Zidna A., Nataraj P.S.V.. Algorithms for unconstrained global optimization of nonlinear (polynomial) programming problems: The single and multi-segment polynomial B-spline approach. Comput Oper Res 2017;87. https://doi.org/10.1016/j.cor.2017.02.013.

[10] Gawali D., Zidna A., Nataraj P.S.V.. Solving nonconvex optimization problems in systems and control: A polynomial B-spline approach. vol. 359. 2015. https://doi.org/10.1007/978-3-319-18161-5_40.

[11] Gawali D.D., Patil B.V., Zidna A., Nataraj P.S.V.. A B-Spline Global Optimization Algorithm for Optimal Power Flow Problem. vol. 991. 2020. https://doi.org/10.1007/978-3-030-21803-4_6.

[12] Lin Q., Rokne J.G. Interval approximation of higher order to the ranges of functions. Computers Mathematics with Applications 1996;31:101–9.

[13] Lin Q., Rokne J.G.. Methods for bounding the range of a polynomial. J Comput Appl Math 1995;58:193–9.

[14] DeVore R.A., Lorentz G.G.. Constructive approximation. vol. 303. Springer Science & Business Media, Berlin; 1993.

[15] Kearfott R.B.. Encyclopedia of Optimization, Springer US; 2009, p. 1763–6.

[16] Nataraj P.S.V., Arounassalame M.. An interval Newton method based on the Bernstein form for bounding the zeros of polynomial systems. Reliab Comput 2011;15(2):185–212.

[17] Verschelde J. The PHC Pack, the Database of Polynomial Systems. University of Illinois, Mathematics Department, Chicago, IL, 2001.