ISSN: 2752-3829 Vol. 2 No.2, (December, 2022)

*Stochastic Modelling and Computational Sciences*

# SUBDIVISION ALGORITHMS FOR SOLVING A POLYNOMIAL SYSTEMS: EMPIRICAL COMPARISONS

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

## ABSTRACT

*This paper deals with the methods based on subdivision for the solution of systems of nonlinear polynomial equations based predicates for the Newton, Hansen-Sengupta, and Krawczyk contractor.*

*The proposed algorithms of polynomial B-spline forms predicates for obtaining the solutions of polynomial system, is based on following technique:*

*1) transformation of the original nonlinear algebraic equations into polynomial B-spline form; 2) includes a pruning step using polynomial B-spline predicate.*

*We solved two numerical examples with proposed algorithms. The performance of proposed algorithms is compared with INTLAB solver based predicates. In algorithm suggested the value of polynomial B-spline predicate is obtained from B-spline coefficient. This approach avoids the repeated computation of function value and the derivative. We compare the performance of interval arithmetic based predicates with its polynomial B-spline form is directly obtained using B-spline coefficients.*

*Keywords: Nonlinear polynomial systems, Polynomial B-spline form, Interval analysis, Interval Newton operator, Hansen-Sengupta operator, Krawczyk operator.*

## I. INTRODUCTION

Finding the solutions of systems of nonlinear equations is a very important problem in scientific computing, constraint logic programming, geometric modeling, engineering, etc. A system of polynomial equations given by

$$f(x) = 0, \qquad (1)$$

where $f = (f_1, f_2, \cdots, f_n)$, and each $f_i$ is a $s$ dimensional polynomial of independent variables $x = (x_1, x_2, \cdots, x_s)$. The solutions can be obtained in a variety of ways, for instance, by directly solving the set of polynomial equations [1] or by resolving a simplified set of functions obtained by transforming the nonlinear polynomial equations [2][3]. Interval analysis provides a powerful solution tool based on the Newton method. The method, called the interval Newton method [4][5][6] provides guaranteed enclosures to the zeros of a given system of nonlinear equations. We compare three interval arithmetic based predicates (the interval Newton, Hansen-Sengupta and Krawczyk operators) with predicates based on their polynomial B-spline forms. The Newton-Raphson iteration is extended in the three operators we consider, the interval Newton operator due to Moore [7], Krawczyk's operator [8] and Hansen and Sengupta [9].

In literature, according to the tightness in the output interval of these operators they are ordered. Let "$\succ$" define the ordering and for two operators R, S that $R \succ S$ if and only if $\mathrm{R}(\mathbf{y}) \subseteq \mathrm{S}(\mathbf{y})$ for each interval that they process on. As in [6]:

Interval Newton $\succ$ Hansen-Sengupta $\succ$ Karwczyk.

Thus, the Newton operator provides the tightest output interval, and we will expect it to perform more frequently than the other operators on a given set of inputs. However, we note that, in practice, there has not been an examination of their results.

In [1][2], the authors proposed several root finding algorithms for the solving systems of nonlinear polynomial equations. In [5] [10] [11] the authors use interval methods for bounding zeros of systems of nonlinear polynomial equations. The approach of interval computation guaranteed an interval that contains all zeros of the system of polynomial equations can be assured using branch and bound strategy. Generally, interval branch and bound methods are time consuming because they requires evaluation of the polynomial functions during each iteration.

Narrowing operators like Hansen-Sengupta, Newton, and Krawczyk can be introduced for pruning the search space. The interval enclosures for these narrowing operators requires evaluation of polynomial function derivatives during each iteration. Finding polynomial function derivatives using interval methods is often time-consuming. Again, in [12] the authors combine Krawczyk contractor and domain subdivision for bounding zeros of systems of nonlinear polynomial equations in B-spline and Bernstein form respectively.

We present an algorithm based on polynomial B-spline form of interval arithmetic based predicates such as Newton, Krawczyk, and Hansen-Sengupta for bounding zeros of systems of polynomial equations. The proposed algorithm combine the advantages of the interval arithmetic based predicates and the B-spline coefficient computation algorithm given in [13][14][15] used for unconstrained optimization problems.

We use B-spline expansion approach to obtain estimate for the range of polynomial in power form. On expanding the polynomial in power form into polynomial B-spline form the minimum and the maximum value of B-spline coefficients provides the bound on the range of polynomial in power form. To obtain tight bounds on the range enclosure we increase number of segments of B-spline as shown in figure 1.

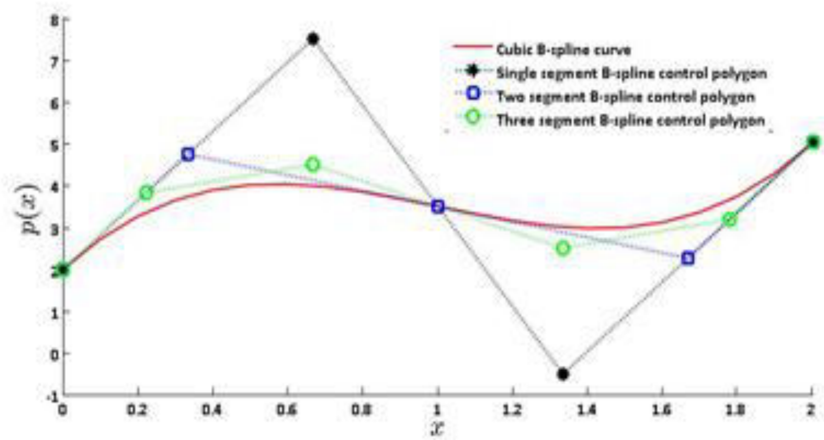


**Figure 1**: Improvement in the range enclosure of univariate polynomial by increasing the number of segments of B-spline.

The computational complexity of B-spline coefficients computation as given in [13] is $O((m+k)^s m)$. Therefore, to minimize the computation time a B-spline with single segments is a best option for bounding zeros of systems of polynomial equations.

This paper is organized as follows. Section 2, gives an overview of B-spline expansion and domain subdivision approach. In section 3, we explain an interval arithmetic based predicates Newton operator, Hansen-Sengupta operator, and Krawczyk operator and propose subdivision algorithm for solving the system of polynomial equation. In section 4, we illustrate the use of the proposed algorithm for solving two numerical examples. We compare the performance of proposed algorithm with INTLAB based solver for interval arithmetic based predicates. 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 [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).

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

### *2.1 Univariate polynomial*

Lets consider univariate polynomial case first, (3)

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

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, $\mathbf{u}$:

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

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

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

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 $t_{-d}\leq\cdots\leq t_{-1}\leq p$ 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*,* (6). Since knot vector $\mathbf{u}$ is uniform grid partition, we choose $t_i:=p+iy$ for $i\in\{-d,\cdots,-1\}\cup\{k+1,\cdots,k+d\},$

$$\mathbf{u}:=\{t_{-d}\leq\cdots\leq t_{-1}\leq p=t_0<t_1<\cdots<t_{k-1}<q=t_k\leq t_{k+1}\leq\cdots\leq t_{k+d}\}. \qquad (6)$$

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:

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

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

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

where

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

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

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

where

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

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 $\left\{B_i^d\left(t\right)\right\}_{i=1}^{k-1}$ satisfies following interesting properties:

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

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

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

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

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

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

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

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

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

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

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

$$= \sum_{v_1=-d_1}^{k_1-1}\dots\sum_{v_l=-d_l}^{k_s-1}\left(\sum_{s_1=0}^{m_1}\dots\sum_{s_l=0}^{m_l} a_{s_1\dots s_l}\pi_{v_1}^{(s_1)}\dots\pi_{v_l}^{(s_l)}\right) B_{v_1}^{d_1}(t_1)\dots B_{v_l}^{d_l}(t_l), \quad (18)$$

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

we can write (18) as

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

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

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

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 $d_{\mathrm{v}}$ 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),

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

where $\mathbf{u}$ is a knot vector. The partial derivative $\varphi_r^{'}(\mathbf{y})$ now includes range enclosure for derivative of $\varphi$ 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,

$$\pi_v^{(s)} := \frac{\mathrm{Sym}_s(v+1,\cdots,v+d)}{\binom{d}{s}}. \qquad (22)$$

### 2.3 B-spline Range Enclosure Property

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

Let (23) 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

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

### 2.4Subdivision 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. INTERVAL ARITHMETIC BASED PREDICATES AND ALGORITHM

In this section we explain interval arithmetic based predicates Newton operator, Krawczyk operator and Hansen-Sengupta operator and present subdivision algorithm for solving a polynomial systems.

### 3.1Newton operator

The interval Newton operator is given in [19] as

$$\mathbf{N}\left(\mathbf{p}, \mathbf{y}, \check{y}\right) = \check{y} - \frac{p(\check{y})}{\mathbf{p}'(\mathbf{y})}. \quad (25)$$

Let $p: \mathbf{y} = [\underline{y}, \overline{y}] \to \mathbb{R}$ 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

$$y^* \in \check{y} - \frac{p\left(\check{y}\right)}{\mathbf{p}'(\mathbf{y})}, \quad \check{y} \in \mathbf{y}. \quad (26)$$

Because of (26), 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 (26) is the basis of the *univariate* Newton method (25).

The *univariate* Newton method (25) can be extended as a *Multivariate* Newton method which execute an iteration equation similar to equation (25).

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

$$\mathbf{N}\left(f, \mathbf{y}, \check{y}\right) = \check{y} + \mathbf{w}, \tag{27}$$

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 $\mathbf{y}$. Therefore obtaining the interval vector $\mathbf{w}$ bounding the solution set to the interval linear system in (27) is an important step in multivariate interval method,

$$\mathbf{f}'(\mathbf{y})\mathbf{w} = -f\left(\check{y}\right).$$

From (25) and (27), the interval vector $\mathbf{w}$ is given by

$$-\frac{f\left(\check{y}\right)}{\mathbf{f}'(\mathbf{y})} = \mathbf{w}.$$

Thus the interval linear system form of multivariate Newton method is given as

$$\mathbf{f}'(\mathbf{y}) \times \mathbf{w} = -f\left(\check{y}\right), \tag{28}$$

It is necessary to precondition the system (28) by a point matrix $Y \in \mathbb{R}^{n \times n}$ given by the inverse of the midpoint matrix of an interval extension of the Jacobi matrix $\mathbf{f}'(\mathbf{y})$, i.e. $Y = \left\{\text{mid}\,\mathbf{f}'(\mathbf{y})\right\}^{-1}$.

$$A \times \mathrm{w} = B, \tag{29}$$

where $A = Y \times \mathbf{f}'(\mathbf{y})$ and $B = -Y \times f\left(\check{y}\right)$.

Then to compute sharper bounds on $\mathbf{w}$ the interval Gauss-Seidel method [20] or the interval hull method [21] can be used to solve the system (29). The components of $\mathbf{N}\left(\mathbf{f}, \mathbf{y}, \check{y}\right)$ is given by (27). Then the intersection $\mathbf{y} \cap \mathbf{N}\left(\mathbf{f}, \mathbf{y}, \check{y}\right)$ results in contracted domain of $\mathbf{y}$.

Interval Newton method requires repeated evaluation of the (polynomial) function at $\check{y} \in \mathbf{y}$ to compute $f(\check{y})$, which can be time consuming operation. Moreover, interval computations are used for finding $\mathbf{f}'(\mathbf{y})$ to compute the precondition matrix $Y$, which apart from time consuming , often give quite pessimistic results.

The B-spline Newton method can alleviate some of these difficulties. In this method, it is quite simple and straightforward to compute $f(\check{y})$, if we choose $\check{y}$ to be any vertex point of $\mathbf{y}$, then $f(\check{y})$ is given directly by the B-spline coefficient value at $\check{y}$. This obviates the need to evaluate the system of polynomial at $\check{y}$ as done in the interval Newton method. In proposed method the B-spline coefficients of the first partial derivatives are simply obtained as the differences of coefficients of the original polynomial $f$ (21).

### 3.2 Krawczyk Operator

The proposed B-spline Krawczyk operator algorithm is based on interval Krawczyk pruning operator. This algorithm is used to decrease the number of iterations. Krawczyk interval contractor is given in [12] as

$$\mathbb{K} = m_x - \mathbb{C}f(m_x) + \mathbb{I} - \mathbb{C}\mathbb{J}(\boldsymbol{x})(\boldsymbol{x} - m_x)$$

Where $\mathbb{C}$ is a nonsingular precondition real matrix, i.e., $\mathbb{C} = \left(\mathrm{mid}\,\mathbb{J}(\boldsymbol{x})\right)^{-1}$ and J is the real Jacobian matrix computed over the interval $\mathbf{x}$, $m_{\mathbf{x}}$ is the center of the interval $\mathbf{x}$ i.e. $m_{\mathbf{x}} = \mathrm{mid}(\mathbf{x})$. The computation of the Krawczyk interval contractor requires the evaluation of the nonlinear polynomial equations at the midpoint, and Jacobian matrix over the interval.

In B-spline Krawczyk method the evaluation of the nonlinear polynomial equations at midpoint can be computed just by subdivision, and the Jacobian value of the nonlinear polynomial equations can be computed by evaluating the partial derivative in all the component directions. Computation of partial derivatives using B-spline coefficients requires only the B-spline coefficients of nonlinear polynomial equations as given in (21).

The intersection between B-spline Krawczyk operator $K$ and the initial domain $\mathbf{x}$ gives the new contracted domain $\mathbf{x}_{new}$, as $\mathbf{x}_{new} = \mathbf{x} \cap K$. The convergence of the solution bounds with B-spline Krawczyk operator will be much faster than interval Krawczyk operator.

### 3.3 Hansen-Sengupta Operator

As interval Newton operator given by (27), we can write as follows:

$$\mathbf{f}'(\mathbf{x})\left(\mathbf{N}\left(\mathbf{f},\mathbf{x},\check{x}\right) - \check{x}\right) = -f\left(\check{x}\right). \qquad (30)$$

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

$$Y\,\mathbf{f}'(\mathbf{x})\left(\mathbf{N}\left(\mathbf{f},\mathbf{x},\check{x}\right) - \check{x}\right) = -Yf\left(\check{x}\right). \qquad (31)$$

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

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

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

$$\mathbf{H}\left(\mathbf{f},\mathbf{x}^k,\check{x}^k\right)_i = \check{x}_i^k - \frac{b_i + \sum_{j=1}^{i-1} Y_{ij}\left(\mathbf{x}^{k+1} - \check{x}^{k+1}\right) + \sum_{j=i+1}^{n} Y_{ij}\left(\mathbf{x}^{k+1} - \check{x}^{k+1}\right)}{Y_{ii}}, \qquad (32)$$

$$\mathbf{x}_i^{k+1} = \mathbf{H}\left(\mathbf{f},\mathbf{x}^k,\check{x}^k\right)_i \cap \mathbf{x}_i^k, \qquad (33)$$

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

In this iteration after the $i$th 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 $\mathbf{H}\left(\mathbf{f},\mathbf{x}^k,\overset{\vee}{x}^k\right)$. Neumaier [6] shows that this operator yields a tighter enclosure than the Krawczyk operator. Next we present subdivision algorithm for solving a polynomial systems similar to [22].

**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 $\epsilon$. |
| **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}<br>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 [13]) |
| 2 | {Initialize iteration number}<br>Set $k=0,\ \mathbf{x}^{(0)}=\mathbf{x}.$ |
| 3 | {Compute $f(\overset{\vee}{x})$ }<br>Choose $\overset{\vee}{x}=\mathrm{mid}(\mathbf{x}^{(k)})$ and obtain the value of $f(\overset{\vee}{x})$ directly from the B-spline coefficient value at the vertex of $\mathrm{mid}(\mathbf{x}^{(k)})$ . |
| 4 | {Compute $\mathbf{f}'(\mathbf{x})$ }<br>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 $Y$ }<br>Compute the preconditioning matrix $Y$ as<br>$Y=\left\{\mathrm{mid}\,\mathbf{f}'\left(\mathbf{x}^k\right)\right\}^{-1}.$ |

| | |
|---|---|
| 6 | {Compute the value of B-spline predicates and update solution} |
| **BNO** | { B-spline Newton operator}<br>Solve the linear interval system<br>$Y \times \mathbf{f}'(\mathbf{x}) \times \mathbf{v} = -Y \times f\left(\check{x}\right)$,<br>and obtain $\mathbf{N}\left(\mathbf{f}, \mathbf{x}^{(k)}, \check{x}\right)$ to update the solution as<br>$\mathbf{x}^{(k)} = \mathbf{x}^{(k)} \cap \mathbf{N}\left(\mathbf{f}, \mathbf{x}^{(k)}, \check{x}\right)$. |
| **BKO** | { B-spline Krawczyk operator}<br>Compute the value of B-spline Krawczyk operator *k* as,<br>$K = f(\check{x}) - Yf(\check{x}) + (I - Y\,\mathbf{f}'(\mathbf{x}))(\mathbf{x} - \check{x})]$.<br>and update the solution as<br>$\mathbf{x}^{(k)} = \mathbf{x}^{(k)} \cap K$. |
| | |
| **BHSO** | { B-spline Hansen-Sengupta operator}<br>Compute the value of B-spline Hansen-Sengupta operator *H* and update the solution,<br>Set $M = Y \times \mathbf{f}'(\mathbf{x})$, $b = Y \times f\left(\check{x}\right)$ and $n = s$.<br>**for i=1 to n do**<br>**if i == 1 then**<br>$H(i) = \check{x}(i) - \dfrac{b(i) + M(i, 2:n) \times \left\{\mathbf{x}(2:n) - \check{x}(2:n)\right\}}{M(i,i)}$,<br>$\mathbf{x}^{(k)}(i) = H(i) \cap \mathbf{x}^{(k)}(i)$,<br>**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 $\{\varnothing\}$ }<br>If $\mathbf{x}^{(k)} = 0$, then return $\{\varnothing\}$ as solution and *exit* algorithm. |
| 8 | {Termination}<br>If $\mathbf{x}^{(k)} < \epsilon$, 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 from [23]to test and compare the performance of three interval arithmetic based predicates (the interval Newton, Hansen-Sengupta and Krawczyk operator) with predicates based on their polynomial B-spline forms. The performance metrics are taken as the number of iterations and computational time (in seconds). Table 2 and Table 5 shows that except for B-spline Hansen-Sengupta operator the performance of polynomial B-spline predicates for interval Newton and Krawczyk operator is more efficient than the interval Newton and Krawczyk operator, because polynomial B-spline predicates avoids the repeated evaluations of polynomials and derivatives. Whereas polynomial B-spline predicates requires more number of iterations than interval arithmetic predicates because the bounds on the range of polynomials provided by B-spline coefficients is over estimated.

As In Table 1 and Table 4 we summarize some representative numerical results. In each numerical tests, the iterations was terminated when the width of each final box bounding a solution was less than $10^{-06}$. The width, $w$ of a box with components $\mathrm{x}_i = [a_i, b_i]$ $(i = 1, \cdots, n)$ is defined to be

$$w = \max_{1 \le i \le n} (b_i - a_i).$$

As shown in Table 3 and Table 6, interval Newton, Hansen-Sengupta and Krawczyk operators required almost same number of iterations in each numerical tests with different computational time because these three operators do not have equal computational costs.

Our MATLAB source code implementation of interval arithmetic predicates using INTLAB [23]solver is made available at [https://bit.ly/34Kb4Ix] for all two test problems. The MATLAB source code for problem evaluation at roots is made available at [bit.ly/34jiiTB] for the interested reader.

***Example 1:*** This example is taken from [23]. This is a problem with 4 variables. The polynomial systems is given by

$$x_1 + x_2 + x_3 + x_4 + 1 = 0,$$
$$x_1 + x_1 x_2 + x_2 x_3 + x_3 x_4 + x_4 = 0,$$
$$x_1 x_2 + x_1 x_2 x_3 + x_2 x_3 x_4 + x_3 x_4 + x_1 x_4 = 0,$$
$$x_1 x_2 x_3 + x_1 x_2 x_3 x_4 + x_2 x_3 x_4 + x_3 x_4 x_1 + x_1 x_2 x_4 = 0.$$

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**: A Comparison of performance between BNO, BHSO and BKO.

| | **Number of Iterations** | **Computation Time (Sec.)** |
|---|---|---|
| **BNO** | 5 | 1.71 |
| **BHSO** | 15 | 3.77 |
| **BKO** | 18 | 1.26 |

**Table 3**: A Comparison of interval arithmetic based predicates for problem 1.

| | **Number of Iterations** | **Computation Time (Sec.)** |
|---|---|---|
| **INO** | 4 | 2.23 |
| **IHSO** | 4 | 3.06 |
| **IKO** | 6 | 2.33 |

***Example 2:*** This example is taken from [23]. This is a problem with 5 variables. The polynomial systems is given by

$$x_1 + x_2 + x_3 + x_4 + x_5 + 1 = 0,$$
$$x_1 + x_1 x_2 + x_2 x_3 + x_3 x_4 + x_4 x_5 + x_5 = 0,$$
$$x_1 x_2 + x_1 x_2 x_3 + x_2 x_3 x_4 + x_3 x_4 x_5 + x_4 x_5 + x_1 x_5 = 0,$$
$$x_1 x_2 x_3 + x_1 x_2 x_3 x_4 + x_2 x_3 x_4 x_5 + x_3 x_4 x_5 + x_1 x_4 x_5 + x_1 x_2 x_5 = 0.$$
$$x_1 x_2 x_3 x_4 + x_1 x_2 x_3 x_4 x_5 + x_2 x_3 x_4 x_5 + x_3 x_4 x_5 x_1 + x_1 x_2 x_4 x_5 + x_1 x_2 x_3 x_5 = 0.$$

and the bounds on the variables are

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

The results of algorithm are tabulated in Table 4.

**Table 4**: Roots of Example 2.

| **Roots** | |
|---|---|
| $x_1$ | 1 |
| $x_2$ | -3.7320 |
| $x_3$ | -0.2679 |
| $x_4$ | 1 |
| $x_5$ | 1 |

**Table 5**: Comparison of performance between BNO, BHSO and BKO.

| | **Number of Iterations** | **Computation Time (Sec.)** |
|---|---|---|
| **BNO** | 7 | 1.23 |
| **BHSO** | 14 | 6.01 |
| **BKO** | 17 | 1.73 |

**Table 6**: A Comparison of interval arithmetic based predicates for problem 2.

| | **Number of Iterations** | **Computation Time (Sec.)** |
|---|---|---|
| **INO** | 4 | 1.44 |
| **IHSO** | 4 | 3.83 |
| **IKO** | 6 | 3.65 |

## V. CONCLUSION

In this paper we implemented subdivision algorithms for solving a polynomial systems using predicates based on polynomial B-spline form and we measured their performance with the interval arithmetic based predicates (the interval Newton, Hansen-Sengupta, and Krawczyk operator). Except for B-spline Hansen-Sengupta operator the performance of B-spline Newton operator and B-spline Krawczyk operator is more efficient than interval Newton and the Krawczyk operator in terms of computation time performance metrics and though in theory the Krawczyk operator is the weakest test, practically it might be a viable choice because it is computational efficient and easy to implement.

## REFERENCES

[1] 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(1):41-64. doi:10.1007/BF02390521

[2] Kolev L.. An interval method for global nonlinear analysis. *IEEE Trans Circuits Syst I Fundam Theory Appl*. 2000;47(5):675-683.

[3] Kolev L. V.. An improved method for global solution of non-linear systems. *Reliab Comput*. 1999;5(2):103-111.

[4] Hansen E., Walster G.W.. *Global Optimization Using Interval Analysis Second Edition, }evised and Expanded*. Vol 264. MARCEL DEKKER, INC. New York; 2004.

[5] Moore R.E.. *Methods and Applications of Interval Analysis.* SIAM, U.S.A.; 1979.

[6] Neumaier A.. *Interval Methods for Systems of Equations*. Vol 37. Cambridge university press; 1990.

[7] Moore R.E.. *Interval Analysis*. Prentice-Hall, Englewood Cliffs, New Jersey; 1966.

[8] Krawczyk R.. Newton-algorithmen zur bestimmung von nullstellen mit fehlerschranken. *Computing*. 1969;4(3):187-201.

[9] Hansen E.R.. On solving systems of equations using interval arithmetic. *Math Comput*. 1968;22(102):374-384.

[10] Hammer R., Hocks M., Kulisch U., Ratz D.. *Numerical Toolbox for Verified Computing I: Basic Numerical Problems Theory, Algorithms, and Pascal-XSC Programs*. Vol 21. Springer Science & Business Media; 2012.

[11] Nataraj P.S.V., Sondur S. Construction of bode envelopes using REP based range finding algorithms. *Int J Autom Comput*. 2011;8(1):112-121.

[12] Arounassalame M. Analysis of Nonlinear Electrical Circuits Using Bernstein Polynomials. *Int J Autom Comput*. 2012;9(1):81-86.

[13] 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. doi:10.1016/j.cor.2017.02.013

[14] Gawali D., Zidna A,. Nataraj P. S. V. Solving Nonconvex Optimization }roblems in Systems and Control: A Polynomial B-spline Approach. In: *Modelling, Computation and Optimization in Information Systems and Management Sciences*. Springer; 2015:467-478.

[15] Gawali D.D., Patil B. V., Zidna A., Nataraj P.S.V. A B-Spline Global Optimization Algorithm for Optimal Power Flow Problem. In: *World Congress on Global Optimization*. ; 2019:58-67.

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

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

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

[19] Kearfott RB. Encyclopedia of Optimization. In: Springer US; 2009:1763-1766.

[20] Kearfott RB. *Rigorous Global Search: Continuous Problems.* Vol 13. Springer Science & Business Media, Berlin; 2013.

[21] Neumaier A. A simple derivation of the Hansen-Bliek-Rohn-Ning-Kearfott enclosure for linear interval equations. *Reliab Comput*. 1999;5(2):131-136.

[22] 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.

[23] Verschelde J. The PHC Pack, the Database of Polynomial Systems. Published online 2001.