Proceedings of the 2nd VNC; VNC-2020;
2020 2nd National Conference on "Technical Advancements for Social Upliftment", 04th April, 2020
Vidyavardhini's College of Engineering & Technology (VCET), Vasai (INDIA)

B-spline Krawczyk Approach for Solving Polynomial Systems

D. D. Gawali
Instrumentation Engineering
Vidyavardhini's College of Engineering and Technology
Vasai, India
deepak.gawali@vcet.edu.in

Ahmed Zidna
LGIPM
University of Lorraine
Metz, France
ahmed.zidna@univ-lorraine.fr

P. S. V. Nataraj
Systems and Control Engineering
Indian Institute of Technology, Bombay
Mumbai, India
nataraj@sc.iitb.ac.in

***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 propose an algorithm for obtaining 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 Krawczyk operator.

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

Keywords — Polynomial B-spline, finding roots, Interval Krawczyk, Polynomial systems.

I. Introduction

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.

In [1], [2] the authors proposed several root-finding algorithms to find the solutions to a system of polynomial equations. In [3]–[6] the authors use interval methods for solving systems of nonlinear algebraic equations. The approach of interval methods guaranteed interval enclosures to all the zeros of the polynomial systems can be obtained using interval branch and bound methods. Unfortunately, this approach often requires repeated evaluation of the polynomial functions, which is a time-consuming operation.

Pruning operator such as Hansen-Sengupta, interval Newton, Krawczyk, etc. can be introduced to reduce the number of iterations. However, the evaluation of interval enclosures for these operators requires derivatives. Finding derivatives of polynomial systems using interval methods is also a time-consuming process. Again, in [7], [8] the authors combine Krawczyk operator and subdivision for solving nonlinear system of polynomial equations in B-spline and Bernstein basis respectively.

An algorithm was proposed based on B-spline expansion approach combined with B-spline Krawczyk operator to find the solutions of a system of polynomial equations i.e. roots. The B-spline coefficient computation algorithm was proposed in [9] for solving global optimization problems. We combine the advantages of the B-spline Krawczyk algorithm, and the B-spline coefficient algorithm to propose a new algorithm to solve a system of nonlinear polynomial equations.

In the B-spline expansion approach, the objective function polynomial in power form is transformed into the polynomial B-spline form of the same degree, (m). Then, the B-spline coefficients give a bound on the range of the objective function.

This paper is organized as follows: In section 2, we give a brief introduction about the B-spline expansion of multivariate power form polynomial and subdivision procedure. In section 3, we explain the interval Krawczyk operator and the B-spline Krawczyk operator algorithm. In section 4, we propose an algorithm for solving the system of the polynomial equation which includes the B-spline Krawczyk operator for pruning the bounds. In section 5, we illustrate the use of the proposed algorithm for solving a system of nonlinear polynomial equations by considering two numerical examples. We compare the performance of our proposed algorithm with the INTLAB based solver. Finally, in the last section, we conclude.

II. B-spline Form

In this section, we first give the notations and definitions, along with some useful properties of B-spline polynomials. Then, we explain the subdivision procedure for domain box.

A. *B-spline form*

Let $s \in \mathbb{N}$ be the number of variables and $x = (x_1, x_2, ..., x_s) \in \mathbb{R}^s$. A multi-index I is defined as $I = (i_1, i_2, ..., i_s) \in \left(\mathbb{N} \cup \{0\}\right)^s$ and multi-power x^I is defined as $x^I = (x_1^{i_1}, x_2^{i_2}, ..., x_s^{i_s})$. Given a multi-index $\mathrm{N} = (n_1, n_2, ..., n_s)$ and an index r, we define $\mathrm{N}_{r,-l} = (n_1, ..., n_{r-1}, n_r - l, n_{r+1}, ...,$

$n_s)$, where $0 \le n_r - l \le n_r$. Inequalities $I \le \mathrm{N}$ for multi-indices are meant component-wise, i.e. $i_l \le n_l$, $l = 1, 2, ..., s$. With $I = (i_1, ..., i_{r-1}, i_r, i_{r+1}, ..., i_s)$ we associate the index $I_{r,l}$ given by $I_{r,l} = (i_1, ..., i_{r-1}, i_r + l, i_{r+1}, ..., i_s)$, where $0 \le i_r + l \le n_r$. A real bounded and closed interval $\mathbf{x}_r$ is defined as $\mathbf{x}_r \equiv [\underline{\mathbf{x}_r}, \overline{\mathbf{x}_r}]$: $= [\inf \mathbf{x}_r = \min \mathbf{x}_r, \sup \mathbf{x}_r = \max \mathbf{x}_r] \in \mathbb{IR}$, where $\mathbb{IR}$ denotes the set of compact intervals. Let wid $\mathbf{x}_r$ denotes the width of $\mathbf{x}_r$, that is width $\mathbf{x}_r := \overline{\mathbf{x}_r} - \underline{\mathbf{x}_r}$. We follow the procedure by [10]–[12] to obtain the B-spline representation of a multivariate polynomial of degree N, in order to derive bounds for its range over an s-dimensional box $x = (x_1, x_2, ..., x_s)$,

$$p(x) = \sum_{I \le \mathrm{N}} a_I x^I, \, x \in \mathbb{R}^s. \qquad (1)$$

B. Univariate case [12]

We consider a univariate polynomial

$$p(x) := \sum_{t=0}^{n} a_t x^t, \, x \in [a, b], \qquad (2)$$

to be expressed in terms of the B-spline basis of the space of polynomial splines of degree $m \ge n$ (i.e. order $m+1$). In the following, we give some preliminary results about the construction of B-spline bases. First of all, we consider the following uniform grid partition

$$\mathbf{u} = \{x_0 < x_1 < \ldots < x_{k-1} < x_k\}, \qquad (3)$$

of the interval $\mathbb{I} = [a, b]$, where $x_i = a + ih, 0 \le i \le k$, and $h = (b-a)/k$. Let $\mathbb{P}_m$ be the space of polynomials of degree at most m. Then the space of splines of degree m and class C^{m-1} on $[a, b]$ associated with $\mathbf{u}$ is defined by

$$\begin{aligned} S_m(\mathbb{I}, \mathbf{u}) = \{ & S \in C^{m-1}(I) : S \,|\, [x_i, x_{i+1}] \in \mathbb{P}_m, \\ & i = 0, ..., k-1\}. \end{aligned} \qquad (4)$$

It is well known that $S_m(\mathbb{I}, \mathbf{u})$ is a linear space of dimension equal to $k+m$ [13]. In order to construct a basis of locally supported splines for $S_m(\mathbb{I}, \mathbf{u})$, some auxiliary knots $x_{-m} \le \cdots \le x_{-1} \le a$ and $b \le x_{k+1} \le \cdots \le x_{k+m}$ are needed. Taking into account that $\mathbf{u}$ is a uniform partition, we choose $x_i := a + ih$ for $i \in \{-m, \ldots, -1\} \cup \{k+1, \ldots, m+k\}$.

$$\begin{aligned} x_{-m} \le \ldots \le x_{-1} \le a = x_0 < x_1 < \ldots < x_{k-1} < x_k = b \\ \le x_{k+1} \le \ldots \le x_{k+m}. \end{aligned} \qquad (5)$$

From the extended partition, a basis $\left(N_i^m\right)_{-m \le i \le k-1}$ of $S_m(\mathbb{I}, \mathbf{u})$ can be defined in terms of divided differences:

$$N_i^m(x) := \left(x_{i+m} - x_i\right)\left[x_i, x_{i+1}, \ldots, x_{i+m+1}\right](\cdot - x)_+^m, \qquad (6)$$

where $(\cdot)_+^m$ stands for the truncated power of degree m. It is easy to prove that

$$N_i^m(x) = \Omega_m\left(\frac{x-a}{h} - i\right), -m \le i \le k-1, \qquad (7)$$

where

$$\Omega_m(x) := \frac{1}{m!}\sum_{\ell=0}^{m+1}(-1)^\ell \binom{m+1}{\ell}(x-\ell)_+^m, \qquad (8)$$

is the B-spline of the degree m associated with the partition of the real line induced by the integer numbers and supported on the interval $[0, m+1]$. The B-splines can be computed by the recurrence formula

$$N_i^m(x) = \gamma_{i,m}(x) N_i^{m-1}(x) + (1 - \gamma_{i+1,m}(x)) N_{i+1}^{m-1}(x), m \ge 1, \quad (9)$$

where

$$\gamma_{i,m}(x) = \begin{cases} \dfrac{x - x_i}{x_{i+m} - x_i}, & \text{if } x_i \le x_{i+m}, \\ 0, & \text{otherwise,} \end{cases} \qquad (10)$$

and

$$N_i^0(x) := \begin{cases} 1, & \text{if } x \in x_i, x_{i+1}), \\ 0, & \text{otherwise.} \end{cases} \qquad (11)$$

It is well known that the set $\{N_i^m\}_{i=-m}^{k-1}$ is a basis for $S_m(\mathbb{I}, \mathbf{u})$ that satisfies interesting properties; for example, each N_i^m is positive on its support and $\{N_i^m\}_{i=-m}^{k-1}$ form a partition of unity.

On the other hand, as $\mathbb{P}_m \subset S_m(\mathbb{I}, \mathbf{u})$, the power basis functions $\{x^r\}_{r=0}^m$ can be expressed in terms of B-splines through the relations

$$x^t = \sum_{j=-m}^{k-1} \pi_j^{(t)} N_j^m(x), \; t = 0, \ldots, m, \qquad (12)$$

where π_j^t are the symmetric polynomials given by

$$\pi_j^{(t)} = \frac{\mathrm{Sym}_t\left(j+1, ..., j+m\right)}{k^t \binom{m}{t}} \; \text{ for } t = 0, 1, \ldots, m. \qquad (13)$$

By substituting (12) into (2) we get

$$\begin{aligned} p(x) = \sum_{t=0}^{n} a_t \sum_{j=-m}^{k-1} \pi_j^{(t)} N_j^m(x) = \sum_{j=-m}^{k-1}\left[\sum_{t=0}^{n} a_t \pi_j^{(t)}\right] N_j^m(x) = \\ \sum_{j=-m}^{k-1} d_j N_j^m(x), \end{aligned} \qquad (14)$$

where

$$d_j \triangleq \sum_{t=0}^{n} a_t \pi_j^{(t)}. \tag{15}$$

C. *Multivariate case*

Now, we derive the B-spline representation of a given multivariate polynomial

$$p(x_1, x_2, ..., x_s) = \sum_{i_1=0}^{n_1} ... \sum_{i_s=0}^{n_s} a_{i_1 ... i_s} x_1^{i_1} ... x_s^{i_s} = \sum_{I \le N} a_I x^I, \tag{16}$$

where $I := (i_1, i_2, ..., i_s)$, and $\mathrm{N} := (n_1, n_2, ..., n_s)$. By substituting (12) for each x^t, (16) can be written as

$$\begin{aligned} & p(x_1, x_2, ..., x_s) = \\ & = \sum_{i_1=0}^{n_1} ... \sum_{i_s=0}^{n_s} a_{i_1 ... i_s} \sum_{j_1=-m_1}^{k_1-1} \pi_{j_1}^{(i_1)} N_{j_1}^{m_1}(x_1) ... \sum_{j_s=-m_s}^{k_s-1} \pi_{j_s}^{(i_s)} N_{j_s}^{m_s}(x_s) \\ & = \sum_{j_1=-m_1}^{k_1-1} ... \sum_{j_s=-m_s}^{k_s-1} \left(\sum_{i_1=0}^{n_1} ... \sum_{i_s=0}^{n_s} a_{i_1 ... i_s} \pi_{j_1}^{(i_1)} .... \pi_{j_s}^{(i_s)} \right) N_{j_1}^{m_1}(x_1) ... N_{j_s}^{m_s}(x_s) \\ & = \sum_{j_1=-m_1}^{k_1-1} ... \sum_{j_s=-m_s}^{k_s-1} d_{j_1 ... j_s} N_{j_1}^{m_1}(x_1) ... N_{j_s}^{m_s}(x_s) \end{aligned} \tag{17}$$

we express p as

$$p(x) = \sum_{I \le N} d_I(x) N_I^{\mathrm{N}}(x), \tag{18}$$

with the coefficients $d_I(x)$ given by

$$d_{j_1, ..., j_s} = \sum_{i_1=0}^{n_1} ... \sum_{i_s=0}^{n_s} a_{i_1 ... i_s} \pi_{j_1}^{(i_1)} .... \pi_{j_s}^{(i_s)}. \tag{19}$$

The B-spline form of a multivariate polynomial *p* is defined by (17). The partial derivative of a polynomial in a particular direction can be found from the B-spline coefficients of the original polynomial on a box $\mathbf{b} \subseteq \mathbf{x}$, the first partial derivative with respect to x_r of a polynomial $p(x)$ in B-spline form is [14]

$$p_r'(\mathbf{b}) = \frac{n_r}{u_{I+n_r+1} - u_{I+1}} \sum_{I \le \mathrm{N}_{r,-1}} \left[d_{I_{r,1}}(\mathbf{b}) - d_I(\mathbf{b}) \right] N_{\mathrm{N}_{r,-1}, I}(x), \quad 1 \le r \le s, x \in \mathbf{b} \tag{20}$$

Where u represents a knot vector. Now, $p_r'(\mathbf{b})$ contains an enclosure of the range of the partial derivative of p on $\mathbf{b}$. In order, the B-spline curve interpolates the end control points and is tangent to the control polygon at its endpoints. One usually duplicate *m+1* time the first and last knot in the knot vector (where *m* is the degree of the B-spline). We use the distribution as follows:

$$\mathbf{u} := \{a = u_{-m} = u_{-m+1} = \cdots = u_{-1} = u_0 < \cdots < u_k = u_{k+1} = \cdots = u_{k+m} = b\}. \tag{21}$$

The distribution of equally spaced knot values as given in (21) is referred to as an open or clamped knot vector. Due to the modification in the distribution of knot values we use the modified form of (13) as follows:

$$\pi_j^{(t)} = \frac{\mathrm{Sym}_t(j+1, ..., j+m)}{\binom{m}{t}}. \tag{22}$$

D. *Range enclosure property*

The following Lemma describes the range enclosure property of the B-spline coefficients.

Lemma 1.

Let *p* be a polynomial of degree N and let $\bar{p}(\mathbf{x})$ denote the range of *p* on the given domain $\mathbf{x}$. Then, for a patch $D(\mathbf{x})$ of B-spline coefficients, it holds

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

Obtaining the B-spline coefficients of multivariate polynomials by transforming the polynomial from power form to B-spline form, provides an enclosure of the range of the multivariate polynomial *p* on $\mathbf{x}$.

E. *B-spline subdivision procedure*

Generally, the range enclosure obtained as per Lemma 1 is over-estimated and can be improved either by subdivision of domain, degree elevation of the B-spline or by increasing the number of B-spline segments. The subdivision is generally more efficient than degree elevation strategy [15], [16] or increasing the number of B-spline segments. Therefore, subdivision strategy is preferred over the latter two. A subdivision in the r^{th} direction $(1 \le r \le s)$ is a bisection perpendicular to this direction. Let

$$\mathbf{x} = [\underline{\mathbf{x}}_1, \overline{\mathbf{x}}_1] \times ... \times [\underline{\mathbf{x}}_r, \overline{\mathbf{x}}_r] \times ... \times [\underline{\mathbf{x}}_s, \overline{\mathbf{x}}_s], \tag{23}$$

be any subbox. Further, suppose that $\mathbf{x}$ is bisected along the r^{th} component direction. Then, two subboxes $\mathbf{x}_A$ and $\mathbf{x}_B$ are generated as

$$\begin{aligned} \mathbf{x}_A &= [\underline{\mathbf{x}}_1, \overline{\mathbf{x}}_1] \times ... \times [\underline{\mathbf{x}}_r, m(\mathbf{x}_r)] \times ... \times [\underline{\mathbf{x}}_s, \overline{\mathbf{x}}_s], \\ \mathbf{x}_B &= [\underline{\mathbf{x}}_1, \overline{\mathbf{x}}_1] \times ... \times [m(\mathbf{x}_r), \overline{\mathbf{x}}_r] \times ... \times [\underline{\mathbf{x}}_s, \overline{\mathbf{x}}_s]. \end{aligned} \tag{24}$$

So here, $m(\mathbf{x}_r)$ denotes the midpoint of $[\underline{\mathbf{x}}_r, \overline{\mathbf{x}}_r]$.

III. B-SPLINE KRAWCZYK OPERATOR ALGORITHM

The proposed B-spline Krawczyk operator algorithm is based on interval Krawczyk pruning operator. This algorithm is introduced to reduce the number of iterations. Interval Krawczyk operator is given by

$$K = y - Mf(y) + I - MJ(\mathbf{x})(\mathbf{x} - y).$$

Where M is a nonsingular precondition real matrix, i.e., $M = \left(\text{mid } J(\mathbf{x})\right)^{-1}$ and J is the real Jacobian matrix computed over the interval $\mathbf{x}$, y is the midpoint of the interval $\mathbf{x}$, i.e., $y = \text{mid}(\mathbf{x})$. The computation of the interval Krawczyk operator needs the evaluation of the nonlinear polynomial equations at the midpoint, $f(y)$ and Jacobian matrix over the interval $\mathbf{x}$. The proposed B-spline Krawczyk operator algorithm can be summarized as follows

Step 1: This algorithm uses a domain box or interval, $\mathbf{x}$ and a cell structure $D_c(\mathbf{x})$, consisting of B-spline coefficients $D_i(\mathbf{x})$ of polynomial systems on the domain $\mathbf{x}$, where $i = 1, 2, \cdots, n,$ and n is the number of polynomial equations.

Step 2: Then we compute the interval midpoint y as $y = \text{mid}(\mathbf{x})$.

Step 3: Next, we compute the polynomial function value at the midpoint for all polynomial functions $f_i(y)$.

Step 4: Using B-spline derivative approach compute derivatives of polynomial systems in all component directions and denote it as interval value, $J(\mathbf{x})$.

Step 5: Compute the determinant of $\text{mid}(J(\mathbf{x}))$ if it is less than ε, then generate two subboxes . Chose the subdivision direction along the longest direction of $\mathbf{x}$ and the subdivision point as the midpoint. Subdivide $\mathbf{x}$ into two subboxes $\mathbf{x}_1$ and $\mathbf{x}_2$ such that $\mathbf{x} = \mathbf{x}_1 \cup \mathbf{x}_2$. Compute the B-spline coefficients for both the subboxes and store both these items $\{\mathbf{x}_1, D(\mathbf{x}_1)\}$ and $\{\mathbf{x}_2, D(\mathbf{x}_2)\}$ into the list $\mathcal{L}$.

Step 6: If the determinant of $\text{mid}(J(\mathbf{x}))$ is greater than ε compute inverse midpoint preconditioner M as $M = \left(\text{mid}(J(\mathbf{x}))\right)^{-1}$.

Step 7: Determine the value of B-spline Krawczyk operator, K as $K = y - Mf(y) + I - MJ(\mathbf{x})(\mathbf{x} - y)$ where I is unity matrix of size $(s \times s)$ and s is the number of variables.

IV. ZERO FINDING ALGORITHM

The algorithm based on B-spline Krawczyk operator for the computation of the roots of a system of nonlinear polynomial equations in s variables which lie within an s dimensional box. The algorithm can be summarized as follows.

Step 1: A cell structure A_c consisting of the polynomial coefficients array a_i of the polynomial in power form, $i = 1, 2, \cdots, n,$ and n is the number of polynomial functions.

Step 2: A cell structure N_c, consisting degree vector $\mathbf{N}_i$, which contains the degree of each variable in a polynomial.

Step 3: Then we compute the B-spline coefficients $D_i(\mathbf{x})$ of the nonlinear polynomials on the initial box $\mathbf{x}$.

Step 4: Initialize a working list $\mathcal{L}$ with the item $\mathcal{L} \leftarrow \{\mathbf{x}, D_i(\mathbf{x})\}$ and a solution list $\mathcal{L}^{Sol}$ to empty list.

Step 5: Start iteration, if $\mathcal{L}$ is empty go to step 14 otherwise pick the last item from $\mathcal{L}$, denote it as $\{\mathbf{b}, D_i(\mathbf{b})\}$ and delete this item entry from the list $\mathcal{L}$.

Step 6: Check the feasibility of the box $\mathbf{b}$, for the enclosure of roots. If $\text{any}(\min(D_i(\mathbf{b}))) > 0$ else if $\text{any}(\max(D_i(\mathbf{b}))) < 0$ then delete this box $\mathbf{b}$ as it does not enclose the roots and go to step 5 else go to step 7.

Step 7: Accepting the new box $\mathbf{b}$ as root. If $\text{width}(\mathbf{b}) < \varepsilon$ then store $\mathbf{b}$ in the list $\mathcal{L}^{Sol}$ and go to step 5 else go to step 8.

Step 8: Compute B-spline Krawczyk operator value K, using B-spline Krawczyk operator algorithm.

Step 9: Next compute the updated bound values as $\mathbf{b}_{\mathbf{new}} = \mathbf{b} \cap K$.

Step 10: Examine the validity of updated bound value. If $\mathbf{b}_{\mathbf{new}} = \varnothing$ then discard the item $\{\mathbf{b}, D_i(\mathbf{b})\}$ and go to step 5 else go to step 11.

Step 11: Subdivision confirmation, if there is a reduction of variable bounds more than 20% in any of the variable direction, evaluated as $\text{any}(\text{width}(\mathbf{b}_{\mathbf{new}})) < 0.8 \times (\text{width}(\mathbf{b}))$ then go to step 12 else go to step 13.

Step 12: Compute the B-spline coefficients over the new contracted bound box $\mathbf{b}_{\mathbf{new}}$ as $D_i(\mathbf{b}_{\mathbf{new}})$ and store item $\{\mathbf{b}_{\mathbf{new}}, D_i(\mathbf{b}_{\mathbf{new}})\}$ into the list $\mathcal{L}$.

Step 13: Generate two items. 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 $\mathbf{b}_2$ then enter both the items $\{\mathbf{b}_1, D(\mathbf{b}_1)\}$ and $\{\mathbf{b}_2, D(\mathbf{b}_2)\}$ into the list $\mathcal{L}$ and go to step 5.

Step 14: Return all the roots found above.

V. NUMERICAL TESTS

The numerical computation is done on a PC Intel i3-370M 2.40 GHz processor, 6 GB RAM, while the algorithms are implemented in MATLAB [17]. An accuracy $\varepsilon = 10^{-03}$ is prescribed for computing the set of roots in each test problem.

We consider the two problems to test and compare the performance of B-spline Krawczyk operator (BKO) over the interval Krawczyk operator (IKO). The performance metrics are taken as the number of iterations and computational time (in seconds). Our MATLAB source code implementation of interval Krawczyk operator based INTLAB [18] solver is made available at [http://bit.ly/37wcB4C] for all two test problems. The MATLAB source code for problem evaluation at roots is made available at [http://bit.ly/2UZNLYB] for the interested reader.

EXAMPLE 1:

This example is taken from [19], [20], the polynomial system is given by

$$\begin{aligned} &1 + x_1 + x_2 + x_3 + x_4 = 0, \\ &x_1 + x_1 x_2 + x_2 x_3 + x_3 x_4 + x_4 = 0, \\ &x_1 x_2 + x_1 x_2 x_3 + x_2 x_3 x_4 + x_3 x_4 + x_4 x_1 = 0, \\ &x_1 x_2 x_3 + x_1 x_2 x_3 x_4 + x_2 x_3 x_4 + x_3 x_4 x_1 + x_4 x_1 x_2 = 0, \end{aligned}$$

and the bounds on the variables are $\mathbf{x}_2 = [0.95, 1.05]$, $\mathbf{x}_3 = [-2.65, -2.6]$, $\mathbf{x}_1 = [0.95, 1.05]$, $\mathbf{x}_4 = [-0.4, -0.37]$.

The results are tabulated in Table I.

TABLE I: Roots value and comparison of performance between BKO and IKO.

Roots			
x_1	x_2	x_3	x_4
1	1	-2.6180	-0.3820

	Number of iterations	**Computation Time (Sec.)**
BKO	4	1.248
IKO	6	1.336

EXAMPLE 2:

This example is taken from [2]. The system of polynomial equations is

$$\begin{aligned} &5x_1^9 - 6x_1^5 x_2^2 + x_1 x_2^4 + 2x_1 x_3 = 0 \\ &-2x_1^6 x_2 + 2x_1^2 x_2^3 + 2x_2 x_3 = 0 \\ &x_1^2 + x_2^2 - 0.265625 = 0 \end{aligned}$$

and the bounds on the variables are $\mathbf{x}_1 = [0.45, 0.5]$, $\mathbf{x}_2 = [0.2, 0.24]$, $\mathbf{x}_3 = [0, 0.03]$.

The results are tabulated in Table II.

TABLE II: Roots value and comparison of performance between BKO and IKO.

Roots		
x_1	x_2	x_3
0.4670	0.2180	0

	Number of iterations	**Computation Time (Sec.)**
BKO	4	0.728
IKO	5	1.1648

VI. CONCLUSION

In this paper, we presented a novel method for finding all roots of a system of nonlinear polynomial equations in s variables which lie within an s dimensional box. We presented two examples to show the superiority of the B-spline Krawczyk operator. It is found in these examples that the proposed algorithm encloses the roots of the polynomial systems with the desired accuracy in the small number of iterations and computation time as compared to the interval Krawczyk operator.

REFERENCES

[1] L. Kolev, “An interval method for global nonlinear analysis,” *IEEE Trans. Circuits Syst. I Fundam. Theory Appl.*, vol. 47, no. 5, pp. 675–683, 2000.

[2] C. Jäger, D. Ratz, K. Iyegyer, and L. Rats, “A combined method for enclosing all solutions of nonlinear systems of polynomial equations,” *Reliab. Comput.*, vol. 1, no. 1, pp. 41–64, 1995.

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

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

[5] R. Hammer, M. Hocks, U. Kulisch, and D. Ratz, *Numerical toolbox for verified computing I: Basic numerical problems theory, algorithms, and pascal-XSC programs*, vol. 21. Springer Science & Business Media, 2012.

[6] P. S. V Nataraj and S. Sondur, “Construction of bode envelopes using REP based range finding algorithms,” *Int. J. Autom. Comput.*, vol. 8, no. 1, pp. 112–121, 2011.

[7] D. Michel and A. Zidna, “Interval-Krawczyk Approach for Solving Nonlinear Equations Systems in B-spline Form,” in *Modelling, Computation and Optimization in Information Systems and Management Sciences*, Springer, 2015, pp. 455–465.

[8] M. Arounassalame, “Analysis of Nonlinear Electrical Circuits Using Bernstein Polynomials,” *Int. J. Autom. Comput.*, vol. 9, no. 1, pp. 81–86, 2012.

[9] D. D. Gawali, A. Zidna, and P. S. V. Nataraj, “Algorithms for unconstrained global optimization of nonlinear (polynomial) programming problems: The single and multi-segment polynomial B-spline approach,” *Comput. Oper. Res.*, vol. 87, 2017.

[10] C. De Boor, *A practical guide to splines*, vol. 27. Springer-Verlag, New York, 1978.

[11] Q. Lin and J. G. Rokne, “Interval approximation of higher order to the ranges of functions.,” *Comput. Math. with Appl.*, vol. 31, no. 7, pp. 101–109, 1996.

[12] Q. Lin and J. G. Rokne, "Methods for bounding the range of a polynomial.," *J. Comput. Appl. Math.*, vol. 58, pp. 193–199, 1995.

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

[14] C. K. Shene, "CS 3621 introduction to computing with geometry notes." 2014.

[15] J. Garloff, "The Bernstein algorithm.," *Interval Comput.*, vol. 6, pp. 154–168, 1993.

[16] M. Zettler and J. Garloff, "Robustness analysis of polynomials with polynomial parameter dependency using Bernstein expansion," *IEEE Trans. Autom. Contr.*, vol. 43(5), pp. 425–431, 1998.

[17] Mathworks Inc., *MATLAB Version 9.2.0.538062 (R 2017 a)*. Inc., Natick, Massachusetts, U.S.A., 2017.

[18] S. M. Rump, "INTLAB-Interval laboratory, the MATLAB toolbox for verified computation, version 10, Hamburg, Germany, 2017." 2017.

[19] A. K. Prakash, "Vectorized interval analysis algorithms and their applications," PhD thesis, Systems and Control Engineering, IIT Bombay, 2002.

[20] V. Stahl, *Interval methods for bounding the range of polynomials and solving systems of nonlinear equations. PhD thesis, Johannes Kepler University, Linz*. 1995.