BIT Numer Math (2017) 57:1109–1136
DOI 10.1007/s10543-017-0679-4

A new method for computing a p-solution to parametric interval linear systems with affine-linear and nonlinear dependencies

Iwona Skalna[1] · **Milan Hladík**[2]

Received: 21 March 2017 / Accepted: 3 August 2017 / Published online: 18 August 2017

Abstract We propose a new approach to computing a parametric solution (the so-called p-solution) to parametric interval linear systems. Solving such system is an important part of many scientific and engineering problems involving uncertainties. The parametric solution has many useful properties. It permits to compute an outer solution, an inner estimate of the interval hull solution, and intervals containing the lower and upper bounds of the interval hull solution. It can also be employed for solving various constrained optimisation problems related to the parametric interval linear system. The proposed approach improves both outer and inner bounds for the parametric solution set. In this respect, the new approach is competitive to most of the existing methods for solving parametric interval linear systems. Improved bounds on the parametric solution set guarantees improved bounds for the solutions of related optimisation problems.

Keywords Revised affine forms · Affine–interval iterative methods · Parametric interval linear systems · Parametric solution · Parametric linear programming

Mathematics Subject Classification 15A06 · 15B99 · 65G99 · 68U99

Communicated by Lars Eldén.

✉ Iwona Skalna
skalna@agh.edu.pl

Milan Hladík
milan.hladik@matfyz.cz

1 AGH University of Science and Technology, Kraków, Poland

2 Charles University, Malostranské nám 25, 118 00 Prague, Czech Republic

1 Introduction

Parametric linear equations are encountered in various fields of science such as engineering, physics, computer science, technology, business, economics etc. However, since uncertainty is inherent in the real world, the parameters of such systems are often subject to uncertainty and assumed to vary within prescribed intervals. Therefore, the problem of solving *parametric interval linear systems* (PILS) has gained significant attention over past years.

Most of the existing methods for solving PILS are concerned with determining an outer interval (OI) solution [20,36] to PILS. Jansson [13] was among the first who considered interval systems with a special structure. Rump [34] proposed a fixed point iteration, which was then investigated by Popova (e.g., [30]) and implemented by Popova and Krämer [32]. Iterative methods for solving PILS were also proposed by Kolev [17,20] and El-Owny [8]. The so-called direct methods were given by Skalna [36] and Kolev [22,24]. A monotonicity approach was investigated by Kolev [14], Popova [31], Rohn [33], and Skalna [38]. Parametric versions of Bauer–Skeel and Hansen–Bliek–Rohn methods were discussed by Hladík [11]. Degrauwe et al. [7] developed a method based on a Neumann series. Affine–Interval Gaussian Elimination was investigated by Akhmerov [1]. The problem of computing the tightest OI solution, i.e., the interval hull solution was considered, e.g., by Kolev [21,23,24], and Skalna [39]. There are also few methods for computing an inner estimate of the hull (IEH) solution (see, e.g. [24,34,37,41]).

A more general approach to the problem of solving PILS was developed by Kolev [17,24]. He introduced a new type of solution, called a *parametric solution* or a *p-solution*, which is of the following parametric form $\mathbf{x}(p) = Lp + \mathbf{a}$, where L is a real matrix and $\mathbf{a}$ is an interval column vector. The p-solution has many useful properties [24]. It permits to compute the OI solution, the IEH solution, and what follows the intervals containing the lower and upper bounds of the hull solution. However, the main advantage of the parametric solution $\mathbf{x}(p)$ is that it can be laid as a basis for a new paradigm for solving the following optimisation problem [24]: find the global minimum of a function $g(x, p)$, subject to constraint $A(p)x(p) = b(p)$, $p \in \mathbf{p}$, where $g(x, p)$ is, in the general case, a nonlinear function.

In this paper a new approach to obtaining a p-solution is proposed. The new approach combines an iterative method with revised affine forms [9,19,44]. The obtained p-solution is represented by a revised affine form, and therefore can be used in any other computation involving revised or standard affine forms. Moreover, the obtained p-solution gives improved bounds of both outer solutions and inner estimates of the hull solution, and hence improved bounds for the optimisation problems related to the parametric interval linear systems. The performance of the proposed approach is illustrated in Sect. 6 using several numerical examples.

The rest of the paper is organised as follows. The basic theory on revised affine forms is presented in Sect. 2. Section 3 describes the problem of solving parametric interval linear systems with both affine-linear and nonlinear dependencies. Section 4 introduces a method for obtaining the new form of the p-solution. The convergence results in a more general fashion are stated in Sect. 5, and thorough numerical comparisons with other methods are performed in Sect. 6. The paper ends with concluding remarks.

Notation Throughout the paper italic fonts are used to denote real quantities, and bold italic fonts are used to denote their interval counterparts. The set of all closed intervals is denoted by $\mathbb{IR}$, whereas $\mathbb{IR}^n$ and $\mathbb{IR}^{n\times n}$ stand, respectively, for the set of all interval vectors and the set of all interval matrices. The mid-point $x^c = (\underline{x}+\overline{x})/2$ and the radius $x^{\Delta} = (\overline{x}-\underline{x})/2$ of an interval $\mathbf{x} = [\underline{x}, \overline{x}]$ are applied to interval vectors and matrices componentwise. The minimal absolute value (mignitude) and the maximal absolute value (magnitude) of $\mathbf{x}$ are defined, respectively, as $\langle \mathbf{x} \rangle = \min\{|x| \mid x \in \mathbf{x}\}$ and $|\mathbf{x}| = \max\{|x| \mid x \in \mathbf{x}\}$. The mignitude and magnitude of a vector and the magnitude of an interval matrix are computed componentwise, whereas the mignitude of an interval matrix is called Ostrowski's comparison matrix with entries $\langle \mathbf{A} \rangle_{ij} = \langle \mathbf{A}_{ij} \rangle$ for $i = j$ and $\langle \mathbf{A} \rangle_{ij} = -|\mathbf{A}_{ij}|$ for $i \neq j$. The identity matrix of any size is denoted by I, and for a non-empty bounded set $S \subset \mathbb{R}^n$, its interval hull $\square S = \bigcap\{\mathbf{Y} \in \mathbb{IR}^n \mid S \subseteq \mathbf{Y}\}$. Finally, D_x denotes a diagonal matrix with entries $x_1, \ldots, x_n$ and $\operatorname{sgn}(x)$ the sign vector of $x \in \mathbb{R}^n$, that is, $\operatorname{sgn}(x)_i = 1$ if $x_i \geq 0$ and $\operatorname{sgn}(x)_i = -1$ otherwise.

2 Revised affine forms

Revised affine forms (RAF) [44] were inspired by the generalised intervals (GI) [9,19] and Messine's reduced affine forms (AF1) [25]. They eliminate the main deficiency of the standard affine arithmetic (AA) [3–6], i.e., the gradual increase of noise symbols, which limits the use of AA in practical applications. Revised affine arithmetic (RAA), like standard affine arithmetic, produces guaranteed enclosures for computed quantities, taking into account any uncertainties in the input data as well as all internal truncation and round-off errors, and also keeps track of first-order correlations between quantities involved in a computation.

A revised affine form of length n is defined as a sum of a standard affine form and a term that represents (accumulates) all errors introduced during a computation (including rounding errors), i.e.

$$\hat{x} = x_0 + \sum_{i=1}^{n} x_i \varepsilon_i + x_r[-1, 1]. \tag{2.1}$$

Here, the *central value* x_0 and the *partial deviations* x_i are finite floating-point numbers, the *noise symbols* ε_i are unknown, but assumed to vary within the interval $\mathbf{e} = [-1, 1]$, and $x_r \geqslant 0$ is the radius of the *accumulative error*. The length n equals to the number of initial uncertain parameters and remains unchanged during the whole computation, unless new independent parameters are introduced.

Remark A revised affine form $\hat{x}$ is an affine–interval function of the noise symbols, so it can be written as $\mathbf{x}(e) = e^{\mathrm{T}}x + \mathbf{x}$, where $e = (\varepsilon_1, \ldots, \varepsilon_n)$, and $\mathbf{x} = x_0 + x_r[-1, 1]$.

If an unknown ideal quantity $\tilde{x}$ is represented by the revised affine form $\hat{x}$, then there exist $\varepsilon_i \in [-1, 1]$ $(i = 1, \ldots, n)$ and $\varepsilon_x \in [-1, 1]$, such that $\tilde{x} = x_0 + \sum_{i=1}^{n} x_i\varepsilon_i + x_r\varepsilon_x$. Moreover, every revised affine form $\hat{x}$ implies the range $[\hat{x}] = [x_0 - r_x, x_0 + r_x]$ which, assuming that each ε_i and ε_x vary independently within $\mathbf{e}$, is the smallest interval that contains all possible values of $\hat{x}$, and thus also an unknown ideal quantity $\tilde{x}$

represented by $\hat{x}$. The radius $r_x = \sum_{i=1}^{n} |x_i| + x_r$ is called the *total deviation* of $\hat{x}$ [5]. Conversely, if an ideal quantity $\tilde{x}$ belongs to an interval $\mathbf{x} = [\underline{x}, \overline{x}]$, then $\tilde{x}$ can be represented by the revised affine form $\hat{x} = x^c + x^{\Delta}\varepsilon_k$, where ε_k is a noise symbol not occurring in any previous computations [4,5].

2.1 Revised affine arithmetic

In order to evaluate an expression with RAA, each elementary operation on real numbers must be replaced by a corresponding operation on revised affine forms. Affine-linear operations result straightforwardly in revised affine forms. Put $e = (\varepsilon_1, \ldots, \varepsilon_n)$ and take arbitrary $\alpha, \beta, \gamma \in \mathbb{R}$. Then, for two revised affine forms $\hat{x}$ and $\hat{y}$, it holds

$$\hat{z} = \alpha\hat{x} + \beta\hat{y} + \gamma = z_0 + e^{\mathrm{T}}z + z_r[-1, 1], \tag{2.2}$$

where $z_0 = \alpha x_0 + \beta y_0 + \gamma$, $z_i = \alpha x_i + \beta y_i$, and $z_r = |\alpha|x_r + |\beta|y_r$.

The result of nonlinear operations must be approximated by a revised affine form and the error of this approximation must be taken into account. There are $n+1$ degrees of freedom in the choice of the affine approximation. However, to keep algorithms simple and efficient, only approximations that are themselves linear combinations of input forms are usually considered [4].

Approximations that minimise the maximum absolute error are the subject of Chebyshev approximation theory [4]. The use of Chebyshev approximations is motivated by the fact that they preserve as much information as possible about the relation between $\hat{x}$, $\hat{y}$ and the affine approximation $\hat{z}$ of $z = f(\hat{x}, \hat{y})$ [4].

Multiplication The revised affine form $\hat{z}$, which describes the product of the revised affine forms $\hat{x} = x_0 + e^{\mathrm{T}}x + x_r[-1, 1]$ and $\hat{y} = y_0 + e^{\mathrm{T}}y + y_r[-1, 1]$, is defined here as follows:

$$\hat{z} = x_0y_0 + 0.5(R_{\min} + R_{\max}) + e^{\mathrm{T}}(y_0x + x_0y) + z_r[-1, 1], \tag{2.3a}$$
$$z_r = |y_0|x_r + |x_0|y_r + 0.5(R_{\max} - R_{\min}), \tag{2.3b}$$

where $R_{\min}$, $R_{\max}$ are, respectively, minimum and maximum of the quadratic form $R(e) = \left(\sum_{i=1}^{n} x_i\varepsilon_i + x_r\varepsilon_{n+1}\right) \cdot \left(\sum_{i=1}^{n} y_i\varepsilon_i + y_r\varepsilon_{n+2}\right)$ on $\mathbf{e}^{n+2}$. The number of multiplications in (2.3) is $2n + 5$. According to [4], the extremal values $R_{\min}$ and $R_{\max}$ can be found in $\mathcal{O}(n \log n)$ time. However, as shown in [42], this can be done in $\mathcal{O}(n)$ time. So, the asymptotic time complexity of the multiplication (2.3) is $\mathcal{O}(n)$, which is optimal.

Alternatively, the following formula for multiplication of revised affine forms can be considered (cf. [43]):

$$\hat{z} = x_0y_0 + x^{\mathrm{T}}y + e^{\mathrm{T}}(y_0x + x_0y) + z_r[-1, 1], \tag{2.4a}$$
$$z_r = |y_0|x_r + |x_0|y_r + (\|x\|_1 + x_r) \cdot (\|y\|_1 + y_r) - |x|^{\mathrm{T}}|y|. \tag{2.4b}$$

Division The division of revised affine forms $\hat{x}$, $\hat{y}$ ($0 \notin [\hat{y}]$) is defined here so that the quotient $\hat{x}/\hat{x}$ is close to 1 [9,15], i.e.

$$\frac{\hat{x}}{\hat{y}} = \frac{x_0}{y_0} + \hat{p} \cdot \hat{y}', \tag{2.5}$$

where $\hat{p} = e^{\mathrm{T}}(x - (x_0/y_0)y) + x_r[-1, 1]$, and $\hat{y}'$ is the Chebyshev minimum-error approximation of the reciprocal (it will be discussed later) of $\hat{y}$. Thus, the revised affine form, which describes the quotient of $\hat{x}$, $\hat{y}$ ($0 \notin [\hat{y}]$), is defined by

$$\hat{z} = x_0/y_0 + \hat{z}', \tag{2.6}$$

where $\hat{z}'$ is the affine approximation of the product $\hat{p} \cdot \hat{y}'$ obtained using (2.3). The asymptotic time complexity of division is $\mathcal{O}(n)$.

Nonlinear unary functions Let f be a nonlinear unary function, such that $f \in \mathcal{C}^1$ and f'' has a constant sign on an interval $[a, b]$ (this might look at first glance as a severe limitation, but in fact many of the basic functions have the required properties). Then, the Chebyshev minimum-error approximation of $f(\hat{x})$ is given by [4]

$$f^a(\hat{x}) = \alpha\hat{x} + \beta, \tag{2.7}$$

where $\alpha = \frac{f(b)-f(a)}{b-a}$, $\beta = \frac{b_1+b_2}{2}$, $b_1 = \frac{f(a)b-f(b)a}{b-a}$, $b_2 = f(\tilde{x}) - \alpha\tilde{x}$, and $\tilde{x}$ solves $f'(\tilde{x}) = \alpha$. The error of this approximation is $\delta = \frac{\mathrm{sgn}(f'')(b_1-b_2)}{2}$. So, the best affine form for $z = f(\hat{x})$, as long as the equation $f'(\tilde{x}) = \alpha$ can be solved, is defined as follows:

$$\hat{z} = \alpha x_0 + \beta + \alpha e^{\mathrm{T}}x + z_r[-1, 1], \tag{2.8a}$$
$$z_r = |\alpha|x_r + \delta. \tag{2.8b}$$

The asymptotic time complexity of this approximation is $\mathcal{O}(n)$.

Reciprocal For the reciprocal function, the coefficients in (2.8) take values

$$\begin{aligned}
\alpha &= -1/(ab),\\
\beta &= \begin{cases} (a+b+2\sqrt{ab})/(2ab) & a > 0,\\ (a+b-2\sqrt{ab})/(2ab) & b < 0,\end{cases}\\
\delta &= \begin{cases} (a+b-2\sqrt{ab})/(2ab) & a > 0,\\ -(a+b+2\sqrt{ab})/(2ab) & b < 0,\end{cases}
\end{aligned}$$

and the point which solves $f'(\tilde{x}) = \alpha$ is

$$\tilde{x} = \begin{cases} \sqrt{ab} & a > 0,\\ -\sqrt{ab} & b < 0.\end{cases}$$

3 Parametric interval linear systems

A parametric interval linear system is defined as the following family of real parametric linear systems

$$A(p)x = b(p), \;\; p \in \mathbf{p} \in \mathbb{IR}^K, \tag{3.1}$$

where K is the number of system parameters, $\mathbf{p}_k = [\underline{p}_k, \overline{p}_k]$ $(k = 1, \ldots, K)$ determine the range for uncertain parameters p_k, $A(p) = [A_{ij}(p)] \in \mathbb{R}^{n\times n}$, and $b(p) = (b_1(p), \ldots, b_n(p))^{\mathrm{T}}$. The parametric interval linear system (3.1) is often written as

$$A(\mathbf{p})x = b(\mathbf{p}), \tag{3.2}$$

to underline its strong relationship with interval linear systems [27].

The entries A_{ij}, and b_i, for $i, j = 1, \ldots, n$, are in general some real valued (continuous) functions of p (usually expressed as closed-form expressions). Depending on whether these functions are affine-linear or nonlinear, parametric interval linear systems with affine-linear and nonlinear dependencies can be distinguished.

In the affine-linear case, matrix $A(p)$ and vector $b(p)$, $p \in \mathbf{p}$, can be represented as follows:

$$A(p) = A^{(0)} + \sum_{k=1}^{K} A^{(k)} p_k, \tag{3.3a}$$

$$b(p) = b^{(0)} + \sum_{k=1}^{K} b^{(k)} p_k, \tag{3.3b}$$

where $A^{(k)} \in \mathbb{R}^{n\times n}$ and $b^{(k)} \in \mathbb{R}^n$, for $k = 0, \ldots, K$. Affine-linear dependencies are by far not easy to handle, but to deal with nonlinear dependencies, some sophisticated tools for bounding the range of a function on a box are required in addition. As will be shown later, with revised affine forms, both types of dependencies can be treated in a unified manner.

3.1 Solutions to parametric interval linear systems

A solution to a parametric interval linear system (3.2) can be defined in various ways. However, usually the so-called *united*[1] *parametric solution set* is considered. It is defined as a set of solutions to all real systems from the family (3.1), i.e.

$$S(\mathbf{p}) = \left\{x \in \mathbb{R}^n \mid \exists p \in \mathbf{p} : A(p)x = b(p)\right\} \tag{3.4}$$

The problem of computing $S(\mathbf{p})$ is, in general case, an NP-hard problem. Therefore, most of the methods for solving parametric interval linear systems produce various types of interval solutions. The best interval solution is simply the hull of the parametric solution set, $\mathbf{x}^H = \square S(\mathbf{p})$. But determining the hull of the solution set is also an NP-hard problem, unless some specific conditions (e.g., monotonicity of the solution with

[1] In what follows, the adjective "united" will be omitted as no other solutions are considered here.

respect to the parameters) are fulfilled. Computationally cheaper methods usually approximate the hull solution, producing the so-called *outer interval (OI) solution*, i.e., an interval vector $\mathbf{x}^* \in \mathbb{IR}^n$, such that $S(\mathbf{p}) \subseteq \mathbf{x}^*$. There are also few methods that produce *inner estimation of the hull (IEH) solution* [24,30,34], i.e. an interval vector $\boldsymbol{\xi}$, such that $\boldsymbol{\xi} \subseteq \mathbf{x}^H$.

A more general approach to the problem of solving PILS was developed by Kolev [24]. He introduced a new type of solution, called a *parametric solution* or a *p-solution*, which is of the following parametric form $\mathbf{x}(p) = Lp + \mathbf{a}$, where $L \in \mathbb{R}^{n\times K}$ and $\mathbf{a} \in \mathbb{IR}^n$. The p-solution can be useful for solving various problems related to parametric interval linear systems, as already mentioned in the introduction.

4 Main result

This section presents a new approach to obtaining a p-solution to parametric interval linear systems with both affine-linear and nonlinear dependencies.

4.1 Affine transformation

Each parameter $p_k \in \mathbf{p}_k$, for $k = 1, \ldots, K$, can be represented by the revised affine from $\hat{p}_k = p_k^c + p_k^{\Delta}\varepsilon_k$, where ε_k is a noise symbol that varies independently from other noise symbols. Substituting the vector $\hat{p}$ for p in $A(p)$ and $b(p)$ gives the following system:

$$A(\hat{p})x = b(\hat{p}), \quad \hat{p} = (\hat{p}_1, \ldots, \hat{p}_K). \tag{4.1}$$

If there are only affine-linear dependencies in (4.1), then (4.1) can be written as

$$\left(A^{(0)} + \sum_{k=1}^{K} A^{(k)}\left(p_k^c + p_k^{\Delta}\varepsilon_k\right)\right)x = b^{(0)} + \sum_{k=1}^{K} b^{(k)}\left(p_k^c + p_k^{\Delta}\varepsilon_k\right),\ \varepsilon_k \in \mathbf{e}, \tag{4.2}$$

or equivalently as

$$C(e)x = c(e),\ e \in \mathbf{e}^K, \tag{4.3}$$

where, for each $e \in \mathbf{e}^K$,

$$C(e) = C^{(0)} + \sum_{k=1}^{K} C^{(k)}\varepsilon_k, \tag{4.4a}$$

$$c(e) = c^{(0)} + \sum_{k=1}^{K} c^{(k)}\varepsilon_k, \tag{4.4b}$$

with

$$C^{(0)} = A^{(0)} + \sum_{k=1}^{K} A^{(k)}p_k^c, \quad C^{(k)} = A^{(k)}p_k^{\Delta}, \tag{4.5a}$$

$$c^{(0)} = b^{(0)} + \sum_{k=1}^{K} b^{(k)}p_k^c, \quad c^{(k)} = b^{(k)}p_k^{\Delta}. \tag{4.5b}$$

Proposition 4.1 *Assuming exact computation, the solution set of the system* (3.1) *equals to the solution set of the system* (4.3).

In the case of nonlinear dependencies, after substituting $\hat{p}_k$ for p_k $(k = 1, \ldots, K)$ in (3.1), the respective operations on revised affine forms must be performed. Then, the following parametric interval linear system with affine-linear dependencies is obtained:

$$\mathbf{C}(e)x = \mathbf{c}(e), \tag{4.6}$$

where

$$\mathbf{C}(e) = C^{(0)} + \sum_{k=1}^{K} C^{(k)}\varepsilon_k + C^r[-1, 1] = \sum_{k=1}^{K} C^{(k)}\varepsilon_k + \mathbf{C}, \tag{4.7a}$$

$$\mathbf{c}(e) = c^{(0)} + \sum_{k=1}^{K} c^{(k)}\varepsilon_k + c^r[-1, 1] = \sum_{k=1}^{K} c^{(k)}\varepsilon_k + \mathbf{c}. \tag{4.7b}$$

Obviously, the system (4.3) is a special case of (4.6), where $C^r = 0$ and $c^r = 0$.

Proposition 4.2 *Let the system* (4.6) *be an affine transformation of the system* (3.1). *Then, the solution set of the system* (3.1) *is included in the solution set of the system* (4.6).

Proof The inclusion follows directly from the *fundamental invariant of affine arithmetic* (FIAA), which can be straightforwardly extended to revised affine forms. The FIAA states that at any stable instant in an AA computation (i.e., at any time when the algorithm is not performing an AA operation), there is a single assignment of values from the interval **e** to each of the noise symbols in use at the time that makes the value of every affine form equal to the value of the corresponding quantity in the ideal computation [4]. □

The transformation from (3.1) to (4.6) causes some loss of information, i.e., an enlargement of the solution set, but instead significantly simplifies computation, and what is more important enables PILS with affine-linear and nonlinear dependencies to be treated in a unified manner. In view of this, the theory presented in the remainder of this paper concerns parametric interval linear systems with affine-linear dependencies, unless stated otherwise.

4.2 Preconditioning

Preconditioning, in the context of solving linear systems, refers to multiplying both sides of a linear system with a nonsingular matrix R. The main reason for preconditioning is to obtain another system with more favourable properties. Preconditioning is especially useful for iterative solvers, however it is performed by most of the methods for solving parametric interval linear systems.

Here, the system (4.6) is preconditioned with the inverse of the mid-point matrix, $R = (C^c)^{-1}$, which, from some point of view (cf. [12]), is an optimal preconditioner [40]. Preconditioning transforms the system (4.6) into a new system

$$\mathbf{V}(e)x = \mathbf{v}(e), \tag{4.8}$$

where $\mathbf{V}(e) = (C^c)^{-1}\mathbf{C}(e)$ and $\mathbf{v}(e) = (C^c)^{-1}\mathbf{c}(e)$. The solution set of the system (4.8) is larger than the solution set of the system (4.6), however the system (4.8) usually has better convergence properties.

Proposition 4.3 *The solution set of the system* (4.6) *is included in the solution set of the system* (4.8).

4.3 Computing a parametric solution

Now, the p-solution is computed in the following way. First, a rigorous enclosure $\mathbf{x}^0 \in \mathbb{IR}^n$ for the solution set of the system (4.8) is computed. The elements $\mathbf{x}_i^0$ $(i = 1, \ldots, n)$ of $\mathbf{x}^0$ are then replaced by the revised affine forms: $\hat{x}_i^0 = (x_i^0)^c + (x_i^0)^\Delta[-1, 1]$. Finally, an iterative method is used to improve the initial enclosure.

Here, the *Interval–affine Gauss–Seidel iteration* (IAGSI) is used to obtain the p-solution. It combines the revised affine forms with the Gauss–Seidel (GS) iteration (for the parametric version of GS see, e.g., [12,30]), which is one of the most known stationary iterative methods for solving linear systems. However, we would like to underline that the Gauss–Seidel iterative scheme can be replaced by any other iterative scheme for solving linear systems, which might result in even better bounds.

For the preconditioned system (4.8), the Interval–affine Gauss–Seidel iteration takes the form:

$$\mathbf{x}^0(e) = \hat{x}^0, \quad \mathbf{x}^{i+1}(e) = \mathbf{L}(e)^{-1}\left(\mathbf{v}(e) - \mathbf{U}(e)\mathbf{x}^i(e)\right), \tag{4.9}$$

where $\mathbf{L}(e) = \left[\mathbf{V}_{j\mu}(e)\right]_{j\leqslant\mu}$, and $\mathbf{U}(e) = \left[\mathbf{V}_{j\mu}(e)\right]_{j>\mu}$. The element-based formula can be obtained by taking the advantage of the lower triangular form of $\mathbf{L}(e)$. Thus, for $j = 1, \ldots, n$,

$$\mathbf{x}_j^0(e) = \hat{x}_j^0, \tag{4.10a}$$

$$\mathbf{x}_j^{i+1}(e) = \frac{\mathbf{v}_j(e) - \sum_{\mu<j}\mathbf{V}_{j\mu}(e)\mathbf{x}_\mu^{i+1}(e) - \sum_{\mu>j}\mathbf{V}_{j\mu}(e)\mathbf{x}_\mu^i(e)}{\mathbf{V}_{jj}(e)}. \tag{4.10b}$$

Clearly, for each $i \geqslant 1$, $\mathbf{x}^i(e)$ is a vector of revised affine forms (revised affine vector), the range of which is the interval vector $\mathbf{x}^i = ([\mathbf{x}_1^{i+1}(e)], \ldots, [\mathbf{x}_n^{i+1}(e)])$. The distance between the interval vectors $\mathbf{x}^i$ and $\mathbf{x}^{i+1}$, which will be used in the stopping criterion of the IAGSI, is assessed here using the following formula [24]:

$$q(\mathbf{x}^i, \mathbf{x}^{i+1}) = \max\left\{\max_{j=1,\ldots,n}\left|\underline{x}_j^i - \underline{x}_j^{i+1}\right|, \max_{j=1,\ldots,n}\left|\overline{x}_j^i - \overline{x}_j^{i+1}\right|\right\}. \tag{4.11}$$

Proposition 4.4 *For any* $\mathbf{x}^i(e)$*,* $i \geqslant 1$*, defined by* (4.10) *we have:*

i. $[\mathbf{x}^i(e)]$ *is an outer interval solution to (3.1),*
ii. *the revised affine vector* $\mathbf{x}^i(e)$ *determines a p-solution to the system (3.1).*

Proof i. If $\tilde{x}$ is a solution to the system (4.8), then there exist $V(e) \in \mathbf{V}(e)$ and $v(e) \in \mathbf{v}(e)$, such that

$$V(e)\tilde{x} = v(e). \tag{4.12}$$

Assuming that $V_{jj}(e) \neq 0$, the Eq. (4.12) can be written as

$$\begin{aligned} \tilde{x}_j &= \frac{1}{V_{jj}(e)}\left(v_j(e) - \sum_{\mu\neq j} V_{j\mu}(e)\tilde{x}_\mu\right) \\ &\in \frac{1}{\mathbf{V}_{jj}(e)}\left(\mathbf{v}_j(e) - \sum_{\mu\neq j} \mathbf{V}_{j\mu}(e)\hat{x}^0_\mu\right) \equiv \mathbf{x}'_j(e). \end{aligned} \tag{4.13}$$

The interval vector $[\mathbf{x}'_j(e)]$ is a new outer interval enclosure for $\tilde{x}$. Since, in the j-th step, an enclosure for $\tilde{x}_j$ and new enclosures for $\tilde{x}_1, \ldots, \tilde{x}_{j-1}$ are already available, they can be used in the right-hand side of (4.13). This gives

$$\tilde{x}_j \in \frac{1}{\mathbf{V}_{jj}(e)}\left(\mathbf{v}_j(e) - \sum_{\mu<j} \mathbf{V}_{j\mu}(e)\mathbf{x}_\mu(e) - \sum_{\mu>j} \mathbf{V}_{j\mu}(e)\hat{x}^0_\mu\right) = \mathbf{x}^1_j(e). \tag{4.14}$$

This means that for each $i \geqslant 1$, the interval vector $[\mathbf{x}^i(e)]$ is an outer interval enclosure for the solution set of the system (4.8). On account of Propositions 4.2 and 4.3, $[\mathbf{x}^i(e)]$ is also an outer interval enclosure for the solution set of the system (3.1).
ii. Follows directly from the fact that the range of $\mathbf{x}^i(e)$ yields an outer interval solution to (3.1). □

Each revised affine vector $\mathbf{x}^i(e)$, $i \geqslant 1$, can be written as

$$\mathbf{x}^i(e) = \sum_{k=1}^{K} (x^i)^{(k)}\varepsilon_k + \mathbf{x}^i, \tag{4.15}$$

where $(x^i)^{(k)} \in \mathbb{R}^n$, $\mathbf{x}^i \in \mathbb{IR}^n$, or, equivalently, as

$$\mathbf{x}^i(e) = L^i e + \mathbf{x}^i, \tag{4.16}$$

where $L^i \in \mathbb{R}^{n\times K}$ with vectors $(x^i)^{(k)}$ as columns. Now, let

$$\lambda^i_j(e) = L^i_j e, \;\; e \in \mathbf{e}^K, \quad j = 1, \ldots, n, \tag{4.17}$$

where $L^i_j \in \mathbb{R}^K$ is the j-th row of the matrix L^i, and denote

$$\lambda^{i,\min}_j = \min\left\{\lambda^i_j(e), e \in \mathbf{e}^K\right\} = -\sum_{k=1}^{K}\left|(x^i)^{(k)}_j\right|, \quad j = 1, \ldots, n, \tag{4.18a}$$

$$\lambda_j^{i,\max} = \max\left\{\lambda_j^i(e), e \in \mathbf{e}^K\right\} = \sum_{k=1}^{K}\left|(x^i)_j^{(k)}\right|, \quad j = 1, \ldots, n. \tag{4.18b}$$

Theorem 4.1 *Let $\mathbf{x}^i(e) = \sum_{k=1}^K (x^i)^{(k)}\varepsilon_k + \mathbf{x}^i$, for $i \geqslant 1$, be the revised affine vector defined by (4.10), and let $\lambda_j^{i,\min}$, $\lambda_j^{i,\max}$ be given by (4.18). Then, $\boldsymbol{\xi}$, such that*

$$\boldsymbol{\xi}_j = \begin{cases} [\lambda_j^{i,\min} + \overline{x}_j^i, \lambda_i^{i,\max} + \underline{x}_j^i], & \lambda_j^{i,\min} + \overline{x}_j^i \leqslant \lambda_j^{i,\max} + \underline{x}_j^i, \\ \varnothing, & \text{otherwise}, \end{cases} \tag{4.19}$$

is an inner estimation of the hull solution to the system (3.1) and the system (4.8).

Proof If $\tilde{x}$ is a solution to the system (3.1), then there exists $p \in \mathbf{p}$, such that $\tilde{x} = A(p)^{-1}b(p)$. So, $\tilde{x}$ is a function of p, i.e., $\tilde{x} = \tilde{x}(p)$. Clearly, p can be written as $p = p^c e + p^\Delta$, and hence, for $j = 1, \ldots, n$, $\tilde{x}_j$ is a function of e, i.e., $\tilde{x}_j = f_j(e)$. From the construction of the iteration (4.10) it follows that, for $i = 1, \ldots, n$,

$$f_j(e) \in \lambda_j^i(e) + \mathbf{x}_j^i, \tag{4.20}$$

and hence

$$\lambda_j^i(e) + \underline{x}_j^i \leqslant f_j(e) \leqslant \lambda_j^i(e) + \overline{x}_j^i. \tag{4.21}$$

Denote the interval hull of the solution set of the system (3.1) as

$$\mathbf{x}^H = \left(\left[x_1^{\min}, x_1^{\max}\right], \ldots, \left[x_n^{\min}, x_n^{\max}\right]\right), \tag{4.22}$$

where

$$x_j^{\min} = \min\{\tilde{x}_j(p) \mid p \in \mathbf{p}\}, \tag{4.23a}$$
$$x_j^{\max} = \max\{\tilde{x}_j(p) \mid p \in \mathbf{p}\}. \tag{4.23b}$$

On account of (4.20), the following inequalities hold true:

$$x_j^{\min} = \min\left\{f_j(e) \mid e \in \mathbf{e}^K\right\} \leqslant \lambda_j^{i,\min} + \overline{x}_j^i, \tag{4.24a}$$
$$x_j^{\max} = \max\left\{f_j(e) \mid e \in \mathbf{e}^K\right\} \geqslant \lambda_j^{i,\max} + \underline{x}_j^i, \tag{4.24b}$$

hence $\boldsymbol{\xi} \subseteq \mathbf{x}^H$, which was to be proved. □

Corollary 4.1 *Let $\mathbf{x}^* := [x^*(e)]$ and $\boldsymbol{\xi}$ be, respectively, an OI solution and IEH solution to the problem (4.8). Then*

$$\underline{x}_i^H \in \left[\underline{x}_j^*, \underline{\xi}_j\right], \quad \overline{x}_j^H \in \left[\overline{x}_j^*, \overline{\xi}_j\right]. \tag{4.25}$$

As can be seen, the p-solution obtained using the Interval–affine Gauss–Seidel iteration permits determining an outer interval solution $\mathbf{x}^*$, an inner estimation of the hull solution $\boldsymbol{\xi}$ as well as intervals containing the endpoints of each component of the hull solution $\mathbf{x}^H$ related to the system (3.1).

4.4 Algorithm of the Interval–affine Gauss–Seidel iteration

The sketch of the algorithm which implements the interval–affine Gauss–Seidel iteration (4.10) is presented in Algorithm 1. The stopping criterion of the iteration is based on the formula (4.11). So, the iteration stops if the distance $q\left([\mathbf{x}^i(e)], [\mathbf{x}^{i+1}(e)]\right) < \epsilon$, where ϵ is a tolerance parameter. Typically $10^{-6} < \epsilon < 10^{-3}$, but this really depends on the problem being solved and the cost of the iterations. The asymptotic time complexity of the algorithm is $\mathcal{O}(n^3K + n^2mM)$, where n is the size of the problem, K is the number of uncertain parameters (noise symbols), m is the cost of multiplication of revised affine forms, and M is the number of iterations taken by the algorithm.

Algorithm 1 Interval–affine Gauss–Seidel iteration

```
Require: Parametric interval linear system $A(p)x = b(p)$, $p \in \mathbf{p} \in \mathbb{IR}^K$
Ensure: Parametric solution $\mathbf{x}^*(e)$
1: Substitute $\hat{p}_k = p^c + p^\Delta \varepsilon_k$ for $p_k$, $k = 1, \dots, K$ and perform the respective operations on RAF
2: $R \approx \left(C^{(0)}\right)^{-1}$ {numerically computed inverse of the mid-point matrix}
3: $\mathbf{V}(e) = R\mathbf{C}(e)$
4: $\mathbf{v}(e) = R\mathbf{c}(e)$
5: Set $\mathbf{x}^0(e)$ to an OI solution computed using some verified method {starting point}
6: repeat
7:     for $j = 1$ to $n$ do
8:         $\mathbf{x}_j^{i+1}(e) = \left(\mathbf{v}_j(e) - \sum_{\mu<j} \mathbf{V}_{j\mu}(e)\mathbf{x}_\mu^{i+1}(e) - \sum_{\mu>j} \mathbf{V}_{j\mu}(e)\mathbf{x}_\mu^i(e)\right) / \mathbf{V}_{jj}(e)$
9:     end for
10: until $\left(q\left([\mathbf{x}^i(e)], [\mathbf{x}^{i+1}(e)]\right) < \epsilon\right)$
11: $\mathbf{x}^*(e) = \mathbf{x}^{i+1}(e)$
12: return $\mathbf{x}^*(e)$
```

Compared with the parametric Gauss–Seidel iteration, the interval–affine version is expected to perform better. This is indeed confirmed in Examples 6.2 and 6.3. However, there are some situations when affine division overestimates interval division, in which case parametric Gauss-Seidel iteration might produce tighter enclosures. This is illustrated in Example 6.1. Therefore, both methods are theoretically incomparable. Practical numerical performance is studied in Sect. 6.

5 Convergence of iterative methods

In this section, we discuss convergence properties for iterative methods that use revised affine forms. We consider a general class of interval operators, including Krawczyk, Gauss–Seidel or Jacobi iterations as particular examples. Denote by

$$\mathbf{W}(e) = \sum_{k=1}^{K} W^{(k)} e_k + \mathbf{W},$$

$$\mathbf{w}(e) = \sum_{k=1}^{K} w^{(k)} e_k + \mathbf{w},$$
$$\mathbf{x}(e) = \sum_{k=1}^{K} x^{(k)} e_k + \mathbf{x}$$

a matrix and two vectors in revised affine forms, respectively. Consider the iterations

$$\mathbf{x}(e) \mapsto \mathbf{w}(e) + \mathbf{W}(e)\mathbf{x}(e), \tag{5.1}$$

where the right-hand side is computed by revised affine arithmetic. Consider also the standard multiplication of revised affine forms

$$\begin{aligned}\mathbf{W}(e)\mathbf{x}(e) := \sum_{k=1}^{K} \left(W^{(k)}x^c + W^c x^{(k)}\right) e_k + W^c x^c + \left(|W^c| x^{\Delta} + W^{\Delta}|x^c|\right)[-1, 1]\\ + \left(\sum_{k=1}^{K} |W^{(k)}| + W^{\Delta}\right)\left(\sum_{k=1}^{K} |x^{(k)}| + x^{\Delta}\right)[-1, 1].\end{aligned}$$

Notice that the term

$$\left(\sum_{k=1}^{K} |W^{(k)}| + W^{\Delta}\right)\left(\sum_{k=1}^{K} |x^{(k)}| + x^{\Delta}\right)[-1, 1]$$

is a simple enclosure for the residual value

$$\mathbf{r}(e) = \left(\sum_{k=1}^{K} W^{(k)} e_k + W^{\Delta}[-1, 1]\right)\left(\sum_{k=1}^{K} x^{(k)} e_k + x^{\Delta}[-1, 1]\right),$$

and it can be replaced by tighter enclosures. However, the theory derived below holds for the simple enclosure as well as for any better one.

Denote

$$R := \sum_{k=1}^{K} \left|W^{(k)}\right| + W^{\Delta}.$$

Theorem 5.1 *If* $\rho(R + |\mathbf{W}|) < 1$, *then the iterations* (5.1) *converge to a unique fixed point for each* $\mathbf{x}(e)$.

Proof Denote

$$\tilde{W}_x := \left(\begin{array}{ccccc} W^c & & & W^{(1)} & \\ & \ddots & & \vdots & \\ & & W^c & V^{(K)} & \\ & & & W^c & \\ RD_{\operatorname{sgn}(x^{(1)})} & \dots & RD_{\operatorname{sgn}(x^{(K)})} & W^{\Delta}D_{\operatorname{sgn}(x^c)} & R + |W^c| \end{array}\right),$$

$$\tilde{x} := \begin{pmatrix} x^{(1)} \\ \vdots \\ x^{(K)} \\ x^c \\ x^{\Delta} \end{pmatrix}, \quad \tilde{w} := \begin{pmatrix} w^{(1)} \\ \vdots \\ w^{(K)} \\ w^c \\ w^{\Delta} \end{pmatrix}, \quad \tilde{W} := \begin{pmatrix} |W^c| & & & |W^{(1)}| & \\ & \ddots & & \vdots & \\ & & |W^c| & |W^{(K)}| & \\ & & & |W^c| & \\ R & \dots & R & W^{\Delta} & R + |W^c| \end{pmatrix}.$$

The affine multiplication can be formulated as

$$\mathbf{V}(e)\mathbf{x}(e) = \sum_{k=1}^{K} (W^{(k)}x^c + W^c x^{(k)})e_k + W^c x^c + [-|W^c|, |W^c|]x^{\Delta} + [-W^{\Delta}, W^{\Delta}]|x^c| + \sum_{k=1}^{K} [-R, R]x^{(k)} + [-R, R]x^{\Delta}.$$

Thus, the iteration (5.1) can be reformulated as follows in terms of the coefficients of the revised affine forms

$$\tilde{x} \mapsto \tilde{w} + \tilde{W}_x \tilde{x} \subseteq \tilde{w} + [-\tilde{W}, \tilde{W}]\tilde{x}. \tag{5.2}$$

Now, due to the assumption $\rho(R + |\mathbf{W}|) < 1$ there is a Perron vector $u > 0$, such that $(R + |\mathbf{W}|)^{\mathrm{T}} u < 1$. Since

$$\begin{pmatrix} |W^c| & & & |W^{(1)}| & \\ & \ddots & & \vdots & \\ & & |W^c| & |W^{(K)}| & \\ & & & |W^c| & \\ R & \dots & R & W^{\Delta} & R + W^{\Delta} \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} u \\ \vdots \\ u \end{pmatrix} \le \begin{pmatrix} (R + |\mathbf{W}|)^{\mathrm{T}} u \\ \vdots \\ (R + |\mathbf{W}|)^{\mathrm{T}} u \end{pmatrix} < \begin{pmatrix} u \\ \vdots \\ u \end{pmatrix},$$

the spectral radius of $\tilde{W}$ is strictly less than 1.

Now, consider two affine forms $\mathbf{x}(e)$ and $\mathbf{y}(e)$, which are equivalently represented by their coefficient vectors $\tilde{x}$ and $\tilde{y}$, respectively. For their images, it holds

$$\left| \tilde{w} + \tilde{W}_x \tilde{x} - \tilde{w} - \tilde{W}_y \tilde{y} \right| = \left| \tilde{W}_x \tilde{x} - \tilde{W}_y \tilde{y} \right|.$$

Due to the property

$$R D_{\mathrm{sgn}(x)} x - R D_{\mathrm{sgn}(y)} y = R|x| - R|y| = R(|x| - |y|) \le R|x - y|$$

we have

$$\left| \tilde{W}_x \tilde{x} - \tilde{W}_y \tilde{y} \right| \le \tilde{W} \left| \tilde{x} - \tilde{y} \right|.$$

Therefore, by [2], the iterations form the so called $\tilde{W}$-contraction, and converge for each initial setting. □

The above iterations involved Krawczyk type of iterative methods. Now, we extend the results to cover also Jacobi and Gauss–Seidel-type iterations. Let

$$\mathbf{U}(e) = \sum_{k=1}^{K} U^{(k)} e_k + \mathbf{U},$$

and consider the iterations

$$\mathbf{x}(e) \mapsto \mathbf{U}(e)\left(\mathbf{w}(e) + \mathbf{W}(e)\mathbf{x}(e)\right). \tag{5.3}$$

For any matrix in the affine form $\mathbf{W}(e)$ denote

$$R_W := \sum_{k=1}^{K} |W^{(k)}| + W^{\Delta}.$$

Theorem 5.2 *The iterations* (5.3) *converge if*

$$\rho(R_U + |\mathbf{U}|) < 1 \quad \textit{and} \quad \rho(R_W + |\mathbf{W}|) < 1.$$

Proof It follows from Theorem 5.1 and its proof by applying it twice on the contracting interval operations

$$\mathbf{x}(e) \mapsto \mathbf{y}(e) := \mathbf{w}(e) + \mathbf{W}(e)\mathbf{x}(e), \quad \mathbf{y}(e) \mapsto \mathbf{x}(e) := \mathbf{U}(e)\mathbf{y}(e).$$

□

Consider the alternative iterations to (5.3)

$$\mathbf{x}(e) \mapsto \mathbf{U}(e)\mathbf{w}(e) + (\mathbf{U}(e)\mathbf{W}(e))\mathbf{x}(e). \tag{5.4}$$

and further denote $\mathbf{G}(e) := \mathbf{U}(e)\mathbf{W}(e)$. Now, it directly follows that:

Corollary 5.1 *The iterations* (5.4) *converge if* $\rho(R_G + |\mathbf{G}|) < 1$.

For the system (4.8), which reads $\mathbf{V}(e)x = \mathbf{v}(e)$, the Krawczyk iteration has the particular form of

$$\mathbf{x}(e) \mapsto \mathbf{v}(e) + (I_n - \mathbf{V}(e))\mathbf{x}(e).$$

We can generalise the properties of the Krawczyk iteration as follows.

Theorem 5.3 *Let* $\mathbf{x}(e)$ *be such that*

$$\mathbf{v}(e) + (I_n - \mathbf{V}(e))\mathbf{x}(e) \subseteq \operatorname{int} \mathbf{x}(e) \tag{5.5}$$

for each $e \in \mathbf{e}^n$. *Then:*

i. $\mathbf{V}(e)$ *is regular for each* $e \in \mathbf{e}^n$,
ii. $\mathbf{x}(e)$ *is an enclosure for the solution set of* (4.8) *for each* $e \in \mathbf{e}^n$.

Proof It directly follows from the properties of the Krawczyk iteration for standard interval systems of equations applied to a fixed $e \in \mathbf{e}^n$; see [35]. □

In order to employ this observation, we need to check for (5.5). The following lines give a recipe how to do it efficiently.

Proposition 5.1 *Let* $\mathbf{x}(e), \mathbf{y}(e)$ *be given. Then* $\mathbf{x}(e) \subseteq \operatorname{int} \mathbf{y}(e)$ *for each* $e \in \mathbf{e}^n$ *if and only if*

$$\overline{x} + \sum_{k=1}^{K} \left|x^{(k)} - y^{(k)}\right| < \overline{y},$$
$$\underline{x} - \sum_{k=1}^{K} \left|x^{(k)} - y^{(k)}\right| > \underline{y}.$$

Proof The maximum value of

$$\mathbf{x}(e) - \mathbf{y}(e) = \sum_{k=1}^{K} \left(x^{(k)} - y^{(k)}\right) e_k + \mathbf{x} - \mathbf{y}$$

is attained for $x := \overline{x}$, $y := \underline{y}$ and $e_k := \operatorname{sgn}(x^{(k)} - y^{(k)})$. Its value must be negative, from which the first condition follows. The second condition is proved analogously. □

6 Numerical experiments

This section shows the potential of the new p-solution. All the computations were carried out using author's own C++ software on a PC computer with CPU 2.50 GHz (Intel(R) Core(TM) i5-4200M), 16Gb of memory, under Microsoft Windows 10 Pro system.

6.1 Solving parametric interval linear systems

Several methods for solving parametric interval linear systems are compared here, mainly in terms of the quality of the computed interval bounds. Whenever the hull solution $\mathbf{x}^H$ is available, the quality of the bounds $\mathbf{x}$ is judged by computing the average percentage by which $\mathbf{x}$ overestimates the hull. Thus, the overestimation measure is defined as follows:

$$O_\omega(\mathbf{x}, \mathbf{x}^H) = \left(1 - \frac{1}{n} \sum_{i=1}^{n} \frac{(x_i^H)^\Delta}{x_i^\Delta}\right) \cdot 100\%. \tag{6.1}$$

The average underestimation of the hull by an IEH solution is provided only when all entries are non-empty. If the hull solution is not available, the results are compared using the *relative sum of radii* (AM denotes the arithmetic mean):

$$\frac{AM(\mathbf{x})}{AM(\mathbf{y})} = \frac{\sum_{i=1}^{n} x_i^{\Delta}}{\sum_{i=1}^{n} y_i^{\Delta}}, \tag{6.2}$$

where $\mathbf{x}$ denotes the bounds computed using one of the considered methods and $\mathbf{y}$ denotes the IAGSI bounds. The best results are highlighted in tables with italics.

The following abbreviations are used to address the methods: PGSI for Parametric Gauss-Seidel Iteration (cf. [12,30]), KI for Kolev's iterative method [24], SVFPI for self-verified Rump's fixed point iteration [30] (which is probably the most recognised method for solving PILS with affine linear dependencies), KDM for Kolev's direct method [16], SDM for Skalna's direct method [36], NP for the Neumaier–Pownuk method [28] (which is known as the best parametric method known so far [29]), PBSM for the parametric Bauer–Skeel method [11], PHBRM for the parametric Hansen–Rohn–Bliek method [11], IAGE for the Interval–affine Gaussian Elimination [1], and DMSO for Degrauwe's method of second order [7].

The comparison of some of the above mentioned methods can be found in the literature. Kolev [16] proved that SDM and KDM yield the same OI solution. Hladík [11] pointed out that PBS gives the same enclosures as SDM and that on average the PBSM gives tighter enclosures that the PHBR method. However, on the occasion, we want to present a comprehensive overview of the most known methods.

In all the presented examples, we use the SDM to obtain the initial box for the IAGSI and PGSI, however any other method for solving parametric interval linear systems can be used instead. Using a more efficient method (see, e.g., [16]) might improve the efficiency of the IAGSI, but not the final result. The performed numerical experiments showed that the quality of the solution provided by IAGSI and PGSI does not depend on the initial enclosure. The stopping criterion parameter $\varepsilon = 1.0e - 8$.

Example 6.1 This example was originally presented in [28]:

$$\begin{pmatrix} p_1 & \frac{1}{2} - p_2 \\ 1 + p_1 & p_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 6 \\ 6 \end{pmatrix}, \quad p \in \begin{pmatrix} \left[\frac{1}{2}, \frac{3}{2}\right] \\ \left[\frac{1}{2}, \frac{3}{2}\right] \end{pmatrix}. \tag{6.3}$$

In [29], there are given the results of the three methods: NP [28], PBSM, and SVFPI. Table 1 reports these results together with the results of the IAGSI and the remaining methods. The average overestimation of the hull, computed using the formula (6.1), is included as well. The hull can be easily computed in this case based on the observation that the extremal values of the solutions $x_1(p_1, p_2)$, $x_2(p_1, p_2)$ are attained at the vertices of the box $[0.5, 1.5] \times [0.5, 1.5]$. The combinatorial approach yields:

$$\mathbf{x}^H = ([3.15789, 12], [-24, -1.263157]).$$

Table 1 shows that the proposed here approach brings significant improvement of the bounds with respect to other considered methods, except the PGSI, which turned out to be the best in this case (with the least number of iterations). The overestimation of IAGSI bounds is almost two times less than the overestimation of the NP, PBSM, SVFPI, PHBRM, KDM, SDM, and DGSO bounds, and almost one and a half times less than the KI bounds. However, it is much worse than the PGSI, which stems from that affine computation is only usually much accurate than interval computation. Here, the affine division lacks accuracy and gives much worse result than the interval division. Table 1 shows as well that despite the PHBR method is on average less accurate than PBS (and thus also SDM and KDM), in this case the lower bound for x_1 and the upper bound for x_2 produced by the HBR method are better than those produced by the PBS method. The IAGE method turned out to be the worst out of all the considered methods. Table 2 presents inner estimate of the hull solution. As we can see, the IAGSI was the only one that was able to produce a significant result. The computational times (Table 3) of all the considered methods are pretty much the same.

Example 6.2 This example shows that in some cases, the IAGSI method with the multiplication formula (2.4) yields really tight bounds for the parametric solution set.

$$\begin{pmatrix} 3 & p & p \\ p & 3 & p \\ p & p & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \tag{6.4}$$

Table 1 Results for Example 6.1: the outer interval solution ($\mathbf{x}^*$) obtained using the most known methods for solving parametric interval linear systems and the IAGSI; average overestimation of the hull solution by the OI solutions computed using the formula (6.1)

Method	$\mathbf{x}_1^*$	$O(\mathbf{x}_1^*, \mathbf{x}_1^H)(\%)$	$\mathbf{x}_2^*$	$O(\mathbf{x}_2^*, \mathbf{x}_2^H)$ (%)
PBSM, SVFPI, KDM, SDM, DGSO	[−3, 12]	41	[−24, 18]	46
PHBRM	[−1.5, 19.5]	58	[−45, 15]	62
IAGE	[−93.65, 177.36]	97	[−197.99, 45.44]	91
NP	[−5.87, 14.87]	57	[−34.23, 28.23]	64
KI	[−1.8651, 12]	36	[−24, 14.9912]	42
PGSI	[1.0909, 12]	19	[−24, 0]	5
IAGSI	[−0.20351, 12]	28	[−24, 6.4503]	25

Table 2 Results for example 6.1: the inner estimation of the hull solution $\mathbf{x}$

Method	$\mathbf{x}_1$	$\mathbf{x}_2$
IAGSI	[5.5909, 6.20553]	–
KI	–	–
SVFPI	–	–

Table 3 Results for Example 6.1: the number of iterations and computational times (in seconds)

Method	#Iter	Time
PBSM	–	0.001
KDM	–	0.001
SDM	–	0.001
PHBRM	–	0.001
DGSO	–	0.001
SVFPI	43	0.002
KI	76	0.001
PGSI	2	0.001
IAGSI	7	0.002

where $p \in [0, 1 + \delta]$. The IAGSI result (OI and IEH solutions), the hull solution and the overestimation for $\delta = 0$ are presented in Table 4. In this case the hull solution can be computed using the fact that, for each i, $x_i(p) = \frac{p^2-6p+9}{2p^3-9p^2+27}$. It is not hard to check that, for $\delta \geqslant 0$, $x_i'([0, 1 + \delta]) \leqslant 0$. So, $x_{i,\min} = x_i(1 + \delta)$ and $x_{i,\max} = x_i(0) = 0.333333$.

Table 6 presents the comparison of the overestimation produced by the remaining methods for $\delta = 0.0, 0.2, 0.6, 0.8$. As can be seen, the IAGSI gave the best bounds. The PGSI method didn't improve the initial enclosures (see Table 5) in any of the considered cases.

Table 7 presents the underestimation of the hull solution by IEH solution obtained from the p-solution. We can see that the IAGSI significantly outperforms other methods. The computational times and number of iterations for iterative methods are given in Table 8.

Table 4 Results for Example 6.2 ($\delta = 0$) obtained using the IAGSI method: OI solution ($\mathbf{x}$), IEH ($\mathbf{y}$), the hull solution ($\mathbf{x}^H$) and the overestimation

$\mathbf{x}$	$\mathbf{y}$	$\mathbf{x}^H$	$O_\omega(\mathbf{x}, \mathbf{x}^H)(\%)$	$O_\omega(\mathbf{x}^H, \mathbf{y})(\%)$
[0.199941508, 0.333338464]	[0.200185226, 0.333094746]	[0.2, 0.333333]	0.000	0.3
[0.199941508, 0.333338464]	[0.200185226, 0.333094746]	[0.2, 0.333333]	0.000	0.3
[0.199941508, 0.333338464]	[0.200185226, 0.333094746]	[0.2, 0.333333]	0.000	0.3

Table 5 Initial enclosures for Example 6.2 obtained using the SDM method

$\delta = 0.0$	$\delta = 0.2$	$\delta = 0.6$	$\delta = 0.8$
[0.153846, 0.346154]	[0.11904, 0.35714]	[0.02798, 0.4068]	[−0.04902, 0.46569]
[0.153846, 0.346154]	[0.11904, 0.35714]	[0.02798, 0.4068]	[−0.04902, 0.46569]
[0.153846, 0.346154]	[0.11904, 0.35714]	[0.02798, 0.4068]	[−0.04902, 0.46569]

Table 6 Results for Example 6.2: the average overestimation of the hull solution by the OI solution

Method	$\delta = 0$ (%)	$\delta = 0.2$ (%)	$\delta = 0.6$ (%)	$\delta = 0.8$ (%)
PHBRM	68.0	71.2	79.6	84.6
PBSM, SDM, KDM, SVFPI	30.7	37.8	54.6	64.7
IAGE	0.2	0.5	2.4	4.7
DGSO	39.2	48.8	68.4	77.9
KI	13.3	16.4	25.6	33.1
PGSI	30.7	37.8	54.6	64.7
IAGSI	0.0	0.1	0.8	1.9

Table 7 Results for Example 6.2: the average underestimation of the hull solution by the IEH solution

Method	$\delta = 0$ (%)	$\delta = 0.2$ (%)	$\delta = 0.6$ (%)	$\delta = 0.8$ (%)
IAGSI	0.3	0.7	2.7	5.2
SVFPI	56.7	77.0		
KI	21.8	28.2	47.2	64.7

Table 8 Results for Example 6.2: the number of iterations and computational times (in seconds)

Method	$\delta = 0$		$\delta = 0.2$		$\delta = 0.6$		$\delta = 0.8$	
	Time	#Iter	Time	#Iter	Time	#Iter	Time	#Iter
PHBRM	0.001	–	0.001	–	0.001	–	0.001	
PBSM	0.001	–	0.001	–	0.001	–	0.001	
SDM	0.001	–	0.001	–	0.001	–	0.001	
IAGE	0.001	–	0.001	–	0.001	–	0.001	
KDM	0.001	–	0.001	–	0.001	–	0.001	
DGSO	0.001	–	0.001	–	0.001	–	0.001	
SVFPI	0.002	7	0.002	8	0.003	12	0.003	17
KI	0.001	9	0.001	12	0.002	19	0.002	27
PGSI	0.002	1	0.002	1	0.002	1	0.002	1
IAGSI	0.005	10	0.006	12	0.007	17	0.01	23

Example 6.3 One of the main advantages of the proposed approach is that it can be used straightforwardly to solve PILS with nonlinear dependencies. Consider the parametric interval linear system

$$\begin{pmatrix} -(p_1 + p_2)/p_4 & p_5 \\ p_2 p_4 & p_3/p_5 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} p_1^2 p_2 \\ p_4^2 + p_5 \end{pmatrix}$$

For $\delta = 0.05$, the entries of the transformed preconditioned matrix $\mathbf{V}$ and the right-hand vector $\mathbf{v}$ are as follows (coefficients are given to the fifth decimal place):

$$\begin{aligned}
\mathbf{V}_{11} &= -3.05136 - 0.18179\varepsilon_1 - 0.14139\varepsilon_2 + 0.42418\varepsilon_4 + 0.061745\mathbf{U}\\
\mathbf{V}_{12} &= 3.5 + 0.07\varepsilon_5 + 4.44089e - 016\mathbf{U}\\
\mathbf{V}_{21} &= 0.25 + 0.035\varepsilon_2 + 0.035\varepsilon_4 + 0.0049\mathbf{U}\\
\mathbf{V}_{22} &= 2.03261 + 0.18179\varepsilon_3 - 0.28279\varepsilon_5 + 0.03769\mathbf{U}\\
\mathbf{v}_1 &= 0.50203 + 0.09\varepsilon_1 + 0.07028\varepsilon_2 + 0.01491\mathbf{U}\\
\mathbf{v}_2 &= 0.63113 + 0.175\varepsilon_4 + 0.01767\varepsilon_5 + 0.01119\mathbf{U}
\end{aligned} \tag{6.5}$$

Table 9 presents the initial enclosures obtained using the SDM method. The overestimation and underestimation for $\delta = 0.05, 0.1, 0.13$ are given in Tables 10 and 11, respectively whereas the computational times and the number of iterations are shown in Table 12. We provide the results of the KI and KDM methods, however, we would like to underline that these methods cannot be used straightforwardly, since they require some specific representation of the parametric interval linear system. Whereas other methods can be used without any additional effort.

Table 9 Initial enclosures for Example 6.3 obtained using the SDM method

$\delta = 0.05$	$\delta = 0.1$	$\delta = 0.13$
$[-0.2600389, 0.595909]$	$[-1.077753, 1.394349]$	$[-2.493651, 2.792468]$
$[0.0697597, 0.509929]$	$[-0.30954, 0.883971]$	$[-0.921001, 1.489605]$

Table 10 Results for Example 6.3: the average overestimation of the hull solution by the OI solution

Method	$\delta = 0.05$ (%)	$\delta = 0.1$ (%)	$\delta = 0.13$ (%)
PHBRM	49	66	79
PBSM, SDM, SVFPI, KDM	36	59	75
IAGE	25	47	64
KI	31	55	71
DGSO	34	58	74
PGSI	26	50	70
IAGSI	23	43	60

Table 11 Results for Example 6.3: the average underestimation of the hull solution by the IEH solution

Method	$\delta = 0$ (%)	$\delta = 0.1$ (%)	$\delta = 0.13$ (%)
IAGSI	34	43	–
SVFPI	51	–	–
KI	40	–	–

Table 12 Results for Example 6.3: the number of iterations and computational times (in seconds)

Method	$\delta = 0.05$		$\delta = 0.1$		$\delta = 0.13$	
	#Iter	Time	#Iter	Time	#Iter	Time
PHBRM		0.001		0.001		0.001
PBSM		0.001		0.001		0.001
SDM		0.001		0.001		0.001
IAGE		0.001		0.001		0.001
KDM		0.001		0.001		0.001
DGSO		0.001		0.001		0.001
SVPI	8	0.001	15	0.001	25	0.001
KI	11	0.002	25	0.003	48	0.004
PGSI	2	0.001	2	0.001	2	0.001
IAGSI	6	0.004	10	0.004	15	0.005

Table 13 Results for Example 6.4: the average overestimation of the hull solution by the OI solution

Method	$\delta = 0.05$ (%)	$\delta = 0.1$ (%)	$\delta = 0.12$ (%)	$\delta = 0.14$ (%)
PHBRM	71	83	89	95
PBSM, SDM, SVFPI, KDM	35	65	77	89
IAGE	41	71	82	93
KI	25	54	68	84
DGSO	47	76	85	93
PGSI	32	61	74	88
IAGSI	24	50	62	77

Example 6.4 Consider the parametric interval linear system

$$\begin{pmatrix} 1/p_1 - 2 & \sqrt{p_1 p_2^2} & p_1^3 & -4 \\ -p_1^2 p_2 + 4 & 1/(p_1^2 + p_2^2) & \sqrt{3 + p_3} & 3p_3 p_4 - 1 \\ 3 & (p_1 - p_3) p_2 & \sqrt{p_2 p_3} & p_1 p_2 p_5 \\ p_4 p_5 - p_1 & (2p_4 - p_3)^2 p_2^2 & p_2 p_3 & p_2 p_3 p_4 + p_5^2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} p_1 \\ p_1^2 - p_2 p_3 \\ -2p_3 \\ -2 \end{pmatrix},$$

where $p_1 \in 1.2\boldsymbol{\delta}$, $p_2 \in 0.8\boldsymbol{\delta}$, $p_3 \in 0.51\boldsymbol{\delta}$, $p_4 \in 2.51\boldsymbol{\delta}$, $p_5 \in 1.01\boldsymbol{\delta}$, $\boldsymbol{\delta} = [1-\delta, 1+\delta]$.

The results for $\delta = 0.05, 0.1, 0.12, 0.14$ are presented in Tables 13, 14 and 15. For the comparison purposes, we use an approximation of the hull solution obtained using a global optimisation software.

Example 6.5 In this example we consider parametric interval linear systems with affine-linear dependencies defined by:

Table 14 Results for Example 6.4: the average underestimation of the hull solution by the IEH solution

	$\delta = 0.05$ (%)	$\delta = 0.1$	$\delta = 0.12$	$\delta = 0.14$
IAGSI	31	–	–	–
SVFPI	39	–	–	–
KI	18	–	–	–

Table 15 Results for Example 6.4: the number of iterations and computational times (in seconds)

Method	$\delta = 0.05$		$\delta = 0.1$		$\delta = 0.12$		$\delta = 0.14$	
	#Iter	time	#Iter	Time	#Iter	Time	#Iter	Time
PHBRM, PBSM, SDM IAGE, DGSO	–	0.003	–	0.003	–	0.003	–	0.003
SVFPI	7	0.004	14	0.004	21	0.005	41	0.006
KI	9	0.011	24	0.011	40	0.011	93	0.011
PGSI	2	0.006	2	0.006	2	0.006	2	0.006
IAGSI	7	0.012	12	0.016	17	0.021	33	0.037

$$\begin{aligned} A^{(k)} &= (k+1)\cdot L, \\ b^{(k)} &= 1, \end{aligned} \tag{6.6}$$

where $L \in \mathbb{R}^n$ is the n-dimensional Lehmer matrix [26] (case A) or n-dimensional Parter matrix [26] (case B). The Lehmer matrix is a symmetric positive definite matrix such that $A_{ij} = i/j$ for $j \geqslant i$. The Parter matrix is a Cauchy matrix and a Toeplitz matrix with singular values near π such that $A_{ij} = 1/(i - j + 0.5)$.

The parameters of the system are subjected to tolerances $\mathbf{p}_k = [1-\delta, 1+\delta]$, $k = 1, \ldots, K$. We compare here three methods which turned out to be the best out of all considered methods, i.e. the IAGSI, PGSI, and KI, with respect to the computational times and tightness of resulting enclosures for various n, K and δ. We provide as well the results for the SDM, since it is used to obtain the initial enclosure for IAGSI and PGSI. The results are presented in Tables 16, 17, 18 and 19. As can be seen, the IAGSI always yields better bounds, but it is the most time consuming. This is however not surprising, because better bounds are usually most costly, unless we can take advantage from some specific conditions. Moreover, affine arithmetic (used by IAGSI) is more expensive than interval arithmetic (used by PGSI), which in turn is more expensive than floating-point arithmetic (used by KI).

Example 6.6 In this example we consider the tridiagonal parametric interval linear system

$$\begin{pmatrix} p_2 & p_1 & & & \\ p_1 & p_2 & p_1 & & \\ & p_1 & p_2 & & \\ & & \ddots & \ddots & p_1 \\ & & & p_1 & p_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \cdots \\ x_n \end{pmatrix} = \begin{pmatrix} -p_1 - p_2 \\ -p_1 - p_2 \\ -p_1 - p_2 \\ \cdots \\ -p_1 - p_2 \end{pmatrix} \tag{6.7}$$

Table 16 Results for Example 6.5 (case A): the relative sum of radii (6.2)

n	K	δ	IAGSI	PGSI	KI	SDM
10	10	0.5	1	1.27	1.11	1.27
10	10	0.6	1	1.33	1.12	1.33
50	20	0.5	1	1.27	1.13	1.27
100	20	0.5	1	1.27	1.13	1.27
100	20	0.8	1	1.09	1.19	1.45
100	30	0.8	1	1.07	1.20	1.44

Table 17 Results for Example 6.5 (case A): CPU times (in s) and the number of iterations taken by iterative methods (given in parentheses)

n	K	δ	IAGSI	PGSI	KI	SDM
10	10	0.5	0.02 (2)	0.02 (1)	0.01 (13)	0.01
10	10	0.6	0.02 (2)	0.02 (1)	0.01 (18)	0.01
50	20	0.5	20.62 (2)	21.78 (1)	1.19 (13)	12.91
100	20	0.5	160.79 (2)	181.74 (1)	3.89 (13)	106.80
100	20	0.8	168.86 (2)	196.67 (2)	9.76 (41)	112.70
100	30	0.8	212.90 (2)	271.86 (2)	20.67 (41)	153.99

Table 18 Results for Example 6.5 (case B): relative sum of radii (6.2)

n	K	δ	IAGSI	PGSI	KI	SDM
10	10	0.5	1	1.27	1.11	1.27
10	10	0.6	1	1.33	1.12	1.33
50	20	0.5	1	1.27	1.13	1.27
100	20	0.5	1	1.27	1.13	1.27
100	20	0.8	1	1.09	1.19	1.45
100	30	0.8	1	1.07	1.20	1.44

Table 19 Results for Example 6.5 (case B): CPU times (in s) and the number of iterations taken by iterative methods (given in parentheses)

n	K	δ	IAGSI	PGSI	KI	SDM
10	10	0.5	0.112 (2)	0.101 (1)	0.012 (14)	0.064
10	10	0.6	0.119 (2)	0.110 (1)	0.016 (19)	0.073
50	20	0.5	20.825 (2)	17.743 (1)	0.889 (12)	10.814
100	20	0.5	162.768 (2)	183.618 (1)	3.699 (12)	111.277
100	20	0.8	173.595 (2)	190.25 (2)	6.835 (39)	114.337
100	30	0.8	179.08 (2)	193.149 (2)	5.966 (38)	121.957

Table 20 Results for Example 6.6: the relative sum of radii (6.2)

n	K	IAGSI	KI	PGSI	SDM
10	10	1	1.01	1.05	1.05
10	20	1	1.03	1.10	1.10
10	50	1	1.12	1.30	1.30
60	10	1	1.03	1.07	1.07
60	20	1	1.07	1.13	1.13
60	50	1	1.23	1.31	1.31
100	10	1	1.05	1.09	1.09
100	20	1	1.10	1.18	1.18
100	50	1	1.15	1.16	1.16
200	10	1	1.13	1.21	1.21
200	20	1	1.30	1.33	1.33

Table 21 Results for Example 6.6: CPU times (in s) and the number of iterations

n	K	IAGSI	KI	PGSI	SDM
10	10	0.05 (4)	0 (3)	0.04 (1)	0.02 (1)
10	20	0.07 (4)	0 (5)	0.05 (1)	0.02 (1)
10	50	0.08 (5)	0 (15)	0.06 (1)	0.02 (1)
60	10	4.51 (5)	0.06 (5)	5.18 (1)	1.09 (1)
60	20	5.24 (5)	0.07 (9)	6.69 (1)	0.96 (1)
60	50	7.17 (8)	0.16 (40)	7.31 (2)	0.59 (2)
100	10	20.09 (6)	0.3 (6)	50.43 (1)	4.12 (1)
100	20	27.03 (7)	0.52 (12)	62.06 (2)	3.95 (2)
100	50	34.62 (70)	1.71 (491)	73.73 (2)	2.64 (2)
200	10	163.23 (9)	1.95 (16)	25.97 (1)	24.58 (1)
200	20	186.84 (26)	3.85 (149)	26.5 (1)	25.98 (1)

where $p_1 \in [100-\delta, 100+\delta]$, $p_2 \in [1-\delta/100, 1+\delta/100]$. We solve the system for different dimensions and tolerances. Tables 20 and 21 present the obtained results. As we can see, IAGSI yields the tightest enslosures, but KI is the winner with respect to time complexity, which is not surprising for the reasons explained in the previous example.

Example 6.7 (Parametric interval linear programming problem) Another advantage of the proposed approach is that it can be used to solve various constrained optimisation problems related to the linear interval parametric systems. In this example, we consider the following parametric linear programming (PLP) problem [10,24]: given a linear parametric objective function

$$l(x, p) = \sum_{i=1}^{n} x_i(p) \tag{6.8}$$

Table 22 Results for Example 6.7: The overestimation of the hull solution l^H to the PLP problem (6.8), (6.9) by the outer interval solution l

Method	$l(\rho = 0.3)$	$O\left(l, l^H\right)$ (%)	$l(\rho = 0.6)$	$O(l^H)$ (%)
IAGSI	[−1.80170, −0.71739]	49	[−4.03384, 1.31046]	77
[18]	[−1.84730, −0.63430]	54	[−5.85463, 3.35040]	87

and the constraint equation

$$\begin{pmatrix} p_1 & p_2+1 & -p_3 \\ p_2+1 & -3 & p_1 \\ 2-p_3 & 4p_2+1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 2p_1 \\ p_3-1 \\ -1 \end{pmatrix}, \tag{6.9}$$

find bounds l on

$$l^*(A(p), b(p), \mathbf{p}) = \left\{ l = \sum_{i=1}^{n} x_i \mid A(p)x = b(p),\ p \in \mathbf{p} \right\}. \tag{6.10}$$

The involved parameter vectors (boxes) of variable width, which depends on a parameter ρ, have the following form

$$\mathbf{p}(\rho) = p^c + \rho \cdot p^{\Delta}[-1, 1], \tag{6.11}$$

where $p^c = (0.5, 0.5, 0.5)$, $p^{\Delta} = (0.5, 0.5, 0.5)$. Each parameter $p_i \in \mathbf{p}_i$ is replaced by the respective revised affine form $\hat{p}_i = p^c + p^{\Delta}\varepsilon_i$. Then, the Interval–affine Gauss–Seidel iteration is used to obtain the p-solution (4.16), and the outer interval enclosure for l^* is computed from the formula:

$$l = \sum_{j=1}^{3} \left| \sum_{i=1}^{3} L_{ij} \right| [-1, 1] + \sum_{i=1}^{3} \mathbf{a}_i. \tag{6.12}$$

The results for $\rho = 0.3$ and $\rho = 0.6$ are presented in Table 22. Additionally, the result from [18], which is so far the best known outer interval solution for the considered PLP problem, is presented as well. It can be seen that the proposed approach yields smaller bounds.

7 Conclusions

A new approach to computing the p-solution of the parametric interval linear system was presented. The obtained p-solution can be useful in solving parametric interval linear systems with both affine-linear and nonlinear dependencies. The p-solution permits to compute outer interval solution, inner estimate of the hull solution and thus also intervals containing the lower and upper bound of the hull solution. The obtained results show that the proposed parametric iteration with revised affine forms might bring significant improvement of both inner and outer bounds of the parametric solution set. Moreover, the proposed approach can also be useful in solving various constrained optimisation problems related to the parametric interval linear system and other problems involving affine forms. There is also a possibility to further improve the results by combining different iterative schemes with different formulae for arithmetic operations on affine forms.

Acknowledgements M. Hladík was supported by the Czech Science Foundation Grant P402/13-10660S.

References

1. Akhmerov, R.R.: Interval–affine Gaussian algorithm for constrained systems. Reliab. Comput. **11**, 323–341 (2005)
2. Alefeld, G., Herzberger, J.: Introduction to Interval Computations. Computer Science and Applied Mathematics. Academic Press, New York (1983)
3. Comba, J.L.D., Stolfi, J.: Affine arithmetic and its applications to computer graphics. In: Proceedings of SIBGRAPI'93 VI Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens (Recife, BR), pp. 9–18 (1993)
4. de Figueiredo, L.H., Stolfi, J.: Self-Validated Numerical Methods and Applications. Brazilian Mathematics Colloquium Monograph. IMPA, Rio de Janeiro (1997)
5. de Figueiredo, L.H., Stolfi, J.: An introduction to affine arithmetic. TEMA Trends Math. Appl. Comput. **4**(3), 297–312 (2003)
6. de Figueiredo, L.H., Stolfi, J.: Affine arithmetic: concepts and applications. Numer. Algorithms **37**(1–4), 147–158 (2004)
7. Degrauwe, D., Lombaert, G., De Roeck, G.: Improving interval analysis in finite element calculations by means of affine arithmetic. Comput. Struct. **88**, 247–254 (2010)
8. El-Owny, H.: Outer interval solution of linear systems with parametric interval data. Int. J. Sci. Eng. Res. **4**(9), 1432–1436 (2013)
9. Hansen, E.R.: A generalized interval arithmetic. In: Nickel, K. (ed.) Interval mathematics: proceedings of the international symposium Karlsruhe, West Germany, May 20–24, 1975. Lecture Notes in Computer Science, pp. 7–18. Springer, Berlin (1975)
10. Hladík, M.: Optimal value range in interval linear programming. Fuzzy Optim. Decis. Mak. **8**(3), 283–294 (2009)
11. Hladík, M.: Enclosures for the solution set of parametric interval linear systems. Int. J. Appl. Math. Comput. Sci. **22**(3), 561–574 (2012)
12. Hladík, Milan: Optimal preconditioning for the interval parametric Gauss–Seidel method. In: M. Nehmeier et al. (ed.) Scientific Computing, Computer Arithmetic and Validated Numerics: 16th International Symposium, SCAN 2014, Würzburg, Germany, vol. 9553 of LNCS, pp. 116–125, Springer (2016)
13. Jansson, C.: Interval linear systems with symmetric matrices, skew-symmetric matrices and dependencies in the right hand side. Computing **46**(3), 265–274 (1991)
14. Kolev, L.: Worst-case tolerance analysis of linear dc and ac electric circuits. IEEE Trans. Circuits Syst. I Fundam. Theory Appl. **49**(12), 1–9 (2002)
15. Kolev, L.: An improved interval linearization for solving nonlinear problems. Numer. Algorithms **37**, 213–224 (2004)
16. Kolev, L.: A direct method for determining a p-solution of linear parametric systems. J. Appl. Comput. Math. **5**(1), 1–5 (2016)
17. Kolev, L: Iterative algorithms for determining a p-solution of linear interval parametric systems. In: Advanced Aspects of Theoretical Electrical Engineering, Sofia, Bulgaria, pp. 15.09–16.09 (2016)
18. Kolev, L.: A new class of iterative interval methods for solving linear parametric systems. Reliab. Comput. **22**, 26–46 (2016)
19. Kolev, L.V.: Automatic computation of a linear interval enclosure. Reliab. Comput. **7**, 7–18 (2001)

20. Kolev, L.V.: A method for outer interval solution of linear parametric systems. Reliab. Comput. **10**(3), 227–239 (2004)
21. Kolev, L.V.: Solving linear systems whose elements are non-linear functions of intervals. Reliab. Comput. **10**(3), 227–239 (2004)
22. Kolev, L.V.: Improvement of a direct method for outer solution of linear parametric systems. Reliab. Comput. **12**(3), 193–202 (2006)
23. Kolev, L.V.: Componentwise determination of the interval hull solution for linear interval parameter systems. Reliab. Comput. **20**(1), 1–24 (2014)
24. Kolev, L.V.: Parameterized solution of linear interval parametric systems. Appl. Math. Comput. **246**, 229–246 (2014)
25. Messine, F.: New affine forms in interval branch and bound algorithms. Technical Report R2I 99-02, Université de Pau et des Pays de l'Adour (UPPA), France (1999)
26. https://www.mathworks.com/help/matlab/ref/gallery.html
27. Neumaier, A.: Interval Methods for Systems of Equations. Cambridge University Press, Cambridge (1990). xvi+255
28. Neumaier, A., Pownuk, A.: Linear systems with large uncertainties, with applications to truss structures. Reliab. Comput. **13**, 149–172 (2007)
29. Popova, E.D.: Improved enclosure for some parametric solution sets with linear shape. Comput. Math. Appl. **68**(9), 994–1005 (2014)
30. Popova, E.D.: On the solution of parametrised linear systems. In: von Gudenberg, J.W., Krämer, W. (eds.) Scientific Computing, Validated Numerics, Interval Methods, pp. 127–138. Kluwer, London (2001)
31. Popova, E.D.: Computer-assisted proofs in solving linear parametric problems. In: 12th GAMM/IMACS International Symposium on Scientific Computing. Computer Arithmetic and Validated Numerics, SCAN 2006, pp. 35–35, Duisburg(2006)
32. Popova, E.D., Krämer, W.: Inner and outer bounds for the solution set of parametric linear systems. J. Comput. Appl. Math. **199**(2), 310–316 (2007)
33. Rohn, J.: A method for handling dependent data in interval linear systems. Technical Report 911, Institute of Computer Science, Academy of Sciences of the Czech Republic, Prague (2004). https://asepactivenode.lib.cas.cz/arl-cav/en/contapp/?repo=crepo1&key=20925094170
34. Rump, S.M.: Verification methods for dense and sparse systems of equations. In: Herzberger, J. (ed.) Topics in Validated Computations, pp. 63–136. Elsevier, Amsterdam (1994)
35. Rump, S.M.: Verification methods: rigorous results using floating-point arithmetic. Acta Numer. **19**, 287–449 (2010)
36. Skalna, I.: A method for outer interval solution of systems of linear equations depending linearly on interval parameters. Reliab. Comput. **12**(2), 107–120 (2006)
37. Skalna, I.: Evolutionary optimization method for approximating the solution set hull of parametric linear systems. In: Lecture Notes in Computer Science, vol. 4310, pp. 361–368 (2007)
38. Skalna, I.: On checking the monotonicity of parametric interval solution of linear structural systems. In: Lecture Notes in Computer Science, vol. 4967, pp. 1400–1409 (2008)
39. Skalna, I.: A global optimization method for solving parametric linear systems whose input data are rational functions of interval parameters. In: Lecture Notes in Computer Science, vol. 6068, pp. 475–484 (2010)
40. Skalna, I.: Strong regularity of parametric interval matrices. Linear Multilinear Algebra (2017). doi:10.1080/03081087.2016.1277687
41. Skalna, I., Duda, J.: A comparison of metaheurisitics for the problem of solving parametric interval linear systems. In: Lecture Notes in Computer Science, vol. 6046, pp. 305–312 (2011)
42. Skalna, I., Hladík, M.: A new algorithm for Chebyshev minimum-error multiplication of reduced affine forms. Numer. Algorithms. doi:10.1007/s11075-017-0300-6
43. Sun, R., Zhang, Y., Cui, A.: A refined affine approximation method of multiplication for range analysis in word-length optimization. EURASIP J. Adv. Signal Process. **36**, 2014 (2014)
44. Vu, X.-H., Sam-Haroud, D., Faltings, B.: A generic scheme for combining multiple inclusion representations in numerical constraint propagation. Technical Report No. IC/2004/39, Swiss Federal Institute of Technology in Lausanne (EPFL), Lausanne (2004)