It is important to note that usually the set of inverses of matrices belonging to **A** is not an interval matrix, hence taking of the hull in the above formula is in general necessary.

If the matrix **A** is an M-matrix, or $\underline{\mathbf{A}}, \overline{\mathbf{A}}$ are regular and $\underline{\mathbf{A}}^{-1}, \overline{\mathbf{A}}^{-1} \geq 0$, we have simply $\mathbf{A}^{-1} = [\overline{\mathbf{A}}^{-1}, \underline{\mathbf{A}}^{-1}] \geq 0$. Interval matrices which are regular and for which $\mathbf{A}^{-1} \geq 0$ are called *inverse positive*. M-matrices are inverse positive, but H-matrices in general are not.

3. Systems of linear interval equations

Let us consider a linear interval system of equations with an interval coefficient matrix $\mathbf{A} \in \mathbb{IR}^{n\times n}$ and an interval right-hand vector $\mathbf{b} \in \mathbb{IR}^n$:

$$\mathbf{Ax} = \mathbf{b} \tag{6}$$

The solution set of Eq. (6) is usually defined as:

$$\sum(\mathbf{A}, \mathbf{b}) = \{\tilde{\mathbf{x}} \in \mathbb{R}^n \mid (\exists \tilde{\mathbf{A}} \in \mathbf{A})(\exists \tilde{\mathbf{b}} \in \mathbf{b})\tilde{\mathbf{A}}\tilde{\mathbf{x}} = \tilde{\mathbf{b}}\}. \tag{7}$$

It is sometimes called a *united solution set* and denoted by $\sum_{\exists\exists}(\mathbf{A}, \mathbf{b})$, as there are other possible definitions of solutions to Eq. (6), see e.g. [32, 33].

Usually the set $\sum(\mathbf{A}, \mathbf{b})$ is not an interval vector, and can be of quite complicated shape (in general, not necessarily convex, connected, or bounded). It is connected and bounded if the matrix **A** is regular. In this case, it constitutes an n-dimensional polyhedron which is a sum of at most 2^n convex polyhedrons obtained as intersections of the set $\sum(\mathbf{A}, \mathbf{b})$ with every of the 2^n orthants of the solution space $Ox_1...x_n$. The *convex hull* conv $\sum(\mathbf{A}, \mathbf{b})$ of this set is a minimal convex polyhedron containing $\sum(\mathbf{A}, \mathbf{b})$; as can be easily seen, the vertices of the convex hull constitute a subset of vertices of the solution set.

Another important and useful characterisation of the solution set, valid also for singular matrices **A**, was given by Oettli and Prager [22] (another proof of the formula was later given by Rohn [26]):

$$\sum(\mathbf{A}, \mathbf{b}) = \{\tilde{x} \in \mathbb{R}^n \mid \left|\check{\mathbf{A}}\tilde{\mathbf{x}} - \check{\mathbf{b}}\right| \leq (\mathrm{rad}\mathbf{A})|\tilde{\mathbf{x}}| + \mathrm{rad}\ \mathbf{b}\}. \tag{8}$$

Calculating (and representing) the solution set $\sum(\mathbf{A}, \mathbf{b})$ may be quite hard and impractical, especially for larger n. Therefore, for many practical purposes we are satisfied with the *interval enclosure* of the set. The smallest (tightest) enclosure is the *hull* of the set:

$$\mathrm{hull} \sum(\mathbf{A}, \mathbf{b}) = [\inf \sum(\mathbf{A}, \mathbf{b}),\ \sup \sum(\mathbf{A}, \mathbf{b})].$$

Obviously, hull $\sum(\mathbf{A}, \mathbf{b})$ = hull conv $\sum(\mathbf{A}, \mathbf{b})$. This enclosure is also hard to calculate in general case (see the next section), hence a number of more effective algorithms producing less exact enclosures has been devised. Some of them are described and used in the sequel to solve certain linear mechanical problems.

3.1. Finding the exact hull of the solution set

There are several methods for obtaining the hull which is the *exact* (called also *optimal*) interval enclosure of the solution set $\sum(\mathbf{A}, \mathbf{b})$. However, it was proven that finding the hull is an inherently exponential complexity problem, see Sec. 5.1.2 for details. Fortunately, this is the worst-case behaviour – for many practical problems some of the algorithms of this class exhibit much better performance [2, 26].

3.1.1. Enumerating combinations of endpoints of interval coefficients (CEIC)

This method of calculating the hull, called also a *combinatorial method* [25], is very simple and easy to implement, hence often used as a reference algorithm during experiments with implementation and application of other, more intricate algorithms.

The CEIC algorithm is based on the theorem by Hartfiel [9] which states (after reformulation into our notation) that for a regular interval matrix $\mathbf{A}$:

$$\operatorname{conv} \sum(\mathbf{A}, \mathbf{b}) = \operatorname{conv} \sum(\partial\mathbf{A}, \partial\mathbf{b}),$$

where:

$$\sum(\partial\mathbf{A}, \partial\mathbf{b}) = \{\tilde{\mathbf{x}} \in \mathbb{R}^n \mid (\exists\tilde{\mathbf{E}} \in \partial\mathbf{A})(\exists\tilde{\mathbf{e}} \in \partial\mathbf{b})\tilde{\mathbf{E}}\tilde{\mathbf{x}} = \tilde{\mathbf{e}}\}. \tag{9}$$

Since for every S holds hull conv S = hull S, then hull $\sum(\mathbf{A}, \mathbf{b})$ = hull $\sum(\partial\mathbf{A}, \partial\mathbf{b})$. That is, the method works by computing standard numerical solutions for all 2^t = card $\partial\mathbf{A}$·card $\partial\mathbf{b}$ real systems of equations for all combinations of endpoints of the interval elements of the matrix $\mathbf{A}$ and vector $\mathbf{b}$ (where $t = t_{\mathbf{A}} + t_{\mathbf{b}}$ is the number of thick interval coefficients in them), and returning the interval envelope of the resulting set of solutions.

Obviously, the algorithm is of exponential complexity: in the worst case, when all intervals in $\mathbf{A}$ and $\mathbf{b}$ are thick, $t = n^2 + n$ and the algorithm must solve 2^{n^2+n} linear systems of n equations, hence its practical value is small. However, when interval elements constitute a small fraction of the coefficients of the system (e.g., for sparse matrices or limited uncertainty in system parameters) it may become more useful than other algorithms of this type described in the subsequent sections.

As $\sum(\partial\mathbf{A}, \partial\mathbf{b}) \subseteq \sum(\mathbf{A}, \mathbf{b})$ and $\sum(\partial\mathbf{A}, \partial\mathbf{b})$ contains all extremal points of the solution set $\sum(\mathbf{A}, \mathbf{b})$, a Monte-Carlo random sampling of the set $\sum(\partial\mathbf{A}, \partial\mathbf{b})$ has good chances to show qualitatively the overall shape of the solution set $\sum(\mathbf{A}, \mathbf{b})$.

3.1.2. The Rohn sign-accord algorithm (RSA)

One way of improving the CEIC algorithm is to find a method to filter out as many as possible of those elements of $\partial\mathbf{A}$ and $\partial\mathbf{b}$ which do not lead to solutions occupying the extremal points of $\sum(\mathbf{A}, \mathbf{b})$. Such a filtering scheme was indeed found by Rohn [20, 26].

Let $J = \{\mathbf{j} \in \mathbb{R}^n \mid |\mathbf{j}| = (1, 1, ..., 1)^T\}$ denotes a set of all n-component vectors with components equal to +1 or −1. Obviously, card $J = 2^n$. For any vector $\mathbf{v} = (v_1, v_2, ..., v_n)^T$, let $\mathbf{D}_{\mathbf{v}} = \operatorname{diag}(v_1, v_2, ..., v_n)$ denotes a diagonal $n \times n$ matrix with components of $\mathbf{v}$ along the diagonal. Then, let us form the following matrices:

$$\begin{cases} \mathbf{A}_{\mathbf{rc}} = \check{\mathbf{A}} - \mathbf{D}_{\mathbf{r}}(\operatorname{rad}\mathbf{A})\mathbf{D}_{\mathbf{c}}, \\ \mathbf{b}_{\mathbf{r}} = \check{\mathbf{b}} + \mathbf{D}_{\mathbf{r}} \operatorname{rad} \mathbf{b}, \end{cases}$$

where $\mathbf{r}, \mathbf{c} \in J$. Obviously, using the definition of midpoint and radius of an interval matrix:

$$(\mathbf{A}_{\mathbf{rc}})_{ij} = \begin{cases} \underline{a}_{ij} & \text{if } r_i c_j = 1, \\ \overline{a}_{ij} & \text{if } r_i c_j = -1, \end{cases}$$

$$(\mathbf{b}_{\mathbf{r}})_i = \begin{cases} \overline{b}_i & \text{if } r_i = 1, \\ \underline{b}_i & \text{if } r_i = -1, \end{cases}$$

hence $\mathbf{A}_{\mathbf{rc}}$ and $\mathbf{b}_{\mathbf{r}}$ are boundary matrices: $\mathbf{A}_{\mathbf{rc}} \in \partial\mathbf{A}$, $\mathbf{b}_{\mathbf{r}} \in \partial\mathbf{b}$. Now putting:

$$\sum(\mathbf{A}_{\mathbf{rc}}, \mathbf{b}_{\mathbf{r}}) = \{\tilde{\mathbf{x}} \in \mathbb{R}^n \mid (\exists\mathbf{r}, \mathbf{c} \in J)\mathbf{A}_{\mathbf{rc}}\tilde{\mathbf{x}} = \mathbf{b}_{\mathbf{r}}\}, \tag{10}$$

we see that $\sum(\mathbf{A_{rc}}, \mathbf{b_r}) \subseteq \sum(\partial\mathbf{A}, \partial\mathbf{b}) \subseteq \sum(\mathbf{A}, \mathbf{b})$. Moreover, as proven by Rohn [20, 26], we ha again, for any regular matrix $\mathbf{A}$, conv $\sum(\mathbf{A_{rc}}, \mathbf{b_r})$ = conv $\sum(\mathbf{A}, \mathbf{b})$, hence also hull $\sum(\mathbf{A_{rc}}, \mathbf{b_r})$ hull $\sum(\mathbf{A}, \mathbf{b})$. Since the number of different pairs of vectors $\mathbf{r}, \mathbf{c} \in J$ equals $2^n \cdot 2^n = 2^{2n}$, we ha card $\sum(\mathbf{A_{rc}}, \mathbf{b_r}) \leq 2^{2n}$ (some of the solutions for different pairs of vectors $\mathbf{r}, \mathbf{c}$ may be the same However, since we do not know in advance which solutions will repeat in $\sum(\mathbf{A_{rc}}, \mathbf{b_r})$, finding th set, and hence the hull, would still require solving (exactly) 2^{2n} linear systems of n equations. Tha though still exponential with the size of the problem, is already a huge improvement in compariso with the worst-case complexity of the CEIC algorithm, namely by a factor of $2^{n^2+n}/2^{2n} = 2^{n^2-n}$

However, still further improvement is possible. Namely, as Rohn showed in [26], each vertex o conv $\sum(\mathbf{A}, \mathbf{b})$ satisfies the equation $\mathbf{A_{rc}}\tilde{\mathbf{x}} = \mathbf{b_r}$ where $\mathbf{c} = \operatorname{sgn} \tilde{\mathbf{x}}$ (i.e., $\mathbf{D_c}\tilde{\mathbf{x}} \geq 0$). Hence, if we coul somehow find for every vector $\mathbf{r} \in J$ such a vector $\mathbf{c} \in J$ that the solution $\tilde{\mathbf{x}}$ to the linear syste of equations $\mathbf{A_{rc}}\tilde{\mathbf{x}} = \mathbf{b_r}$ will have the property sgn $\tilde{\mathbf{x}} = \mathbf{c}$ (for the purpose of this analysis we ca safely assume that sgn $0 = 1$), then the total number of linear systems to solve would be only 2^n (i.e., the cardinality of J). This leads to the following algorithm:

<u>*Rohn Sign-Accord* algorithm (RSA):</u>

For every $\mathbf{r} \in J$ do:

Step 0: Select a $\mathbf{c} \in J$ (recommended: $\mathbf{c} = \operatorname{sgn}(\check{\mathbf{A}}^{-1}\mathbf{b_r})$).
Step 1: Solve $\mathbf{A_{rc}}\tilde{\mathbf{x}} = \mathbf{b_r}$.
Step 2: If sgn $\tilde{\mathbf{x}} = \mathbf{c}$, register $\tilde{\mathbf{x}}$ and go to next $\mathbf{r}$, otherwise:
Step 3: Find $k = \min\{j \mid \operatorname{sgn} \tilde{\mathbf{x}}_j \neq \mathbf{c}_j\}$.
Step 4: Set $\mathbf{c}_k = -\mathbf{c}_k$ and go to *Step 1*.

The algorithm finds all 2^n vertices of conv $\sum(\mathbf{A}, \mathbf{b})$ in a finite number of steps regardless of the choice of initial vector $\mathbf{c} \in J$ in *Step 0*. However, the recommendation included there is quite important. As it was shown by Rohn [26], while the worst-case complexity of the algorithm is still of the order of 2^{2n} (with the inner loop [*Step 1* – *Step 4*] traversed 2^n times for every $\mathbf{r}$), starting with the recommended value of $\mathbf{c}$ leads for many types of problems to much smaller complexity, often as low as 2^n (with the inner loop traversed only once for every $\mathbf{r}$). The hull of $\sum(\mathbf{A}, \mathbf{b})$ is then easily obtained as the hull of the 2^n solutions registered at *Step 2* of the algorithm. In fact, compiling the whole set of solutions is not necessary for calculation of the hull, as it can be made "on the fly", by simply updating current minimal and maximal values of vector $\tilde{\mathbf{x}}$ components with every new solution found at *Step 2*.

The algorithm can be also extended into a more complicated form capable of testing for regularity of the matrix $\mathbf{A}$ along the way [20, 26].

A problem with this algorithm may arise for systems in which components of some endpoint solution(s) are near to zero. Then possible roundoff errors may produce wrong value for sgn $\tilde{\mathbf{x}}$ at *Step 2*, which may lead to infinite looping of the algorithm [20].

3.1.3. The linear programming method (LPM)

As the problem of finding the hull is in fact a problem of finding extremal values of some set of numbers, one may try to formulate it as an optimisation problem – minimisation (or maximisation) of appropriate objective function subject to appropriate constraints. The resulting linear programming method (LPM) has been first formulated in [21] and then used, among others, in [12, 23, 25].

The derivation of the method starts from Oettli and Prager [22] characterisation of the solution set $\sum(\mathbf{A}, \mathbf{b})$ (see Eq. (8)):

$$\tilde{\mathbf{x}} \in \sum(\mathbf{A}, \mathbf{b}) \Leftrightarrow \left|\check{\mathbf{A}}\tilde{\mathbf{x}} - \check{\mathbf{b}}\right| \leq (\operatorname{rad}\mathbf{A})|\tilde{\mathbf{x}}| + \operatorname{rad}\mathbf{b}.$$