Министерство Образования Российской Федерации
Новосибирский Государственный Университет
Механико-математический Факультет
Кафедра «Математическое моделирование»

Дипломная работа на тему:

Сравнительный анализ модификации Рона в методе дробления параметров

Выполнил: студент гр. 5132 Людвин Д. Ю.
Научный руководитель: д.ф.-м.н Шарый С. П.

Новосибирск–2010

Оглавление

1 Введение и постановка задачи

В работе рассматривается задача внешнего интервального оценивания объединённого множества решений интервальных систем линейных алгебраических уравнений (ИСЛАУ) вида

$$\begin{cases} \boldsymbol{a}_{11}x_1 + \boldsymbol{a}_{12}x_2 + \ldots + \boldsymbol{a}_{1n}x_n = \boldsymbol{b}_1, \\ \boldsymbol{a}_{21}x_1 + \boldsymbol{a}_{22}x_2 + \ldots + \boldsymbol{a}_{2n}x_n = \boldsymbol{b}_2, \\ \quad\vdots \qquad\quad \vdots \qquad\quad \ddots \qquad\quad \vdots \qquad\quad \vdots \\ \boldsymbol{a}_{n1}x_1 + \boldsymbol{a}_{n2}x_2 + \ldots + \boldsymbol{a}_{nn}x_n = \boldsymbol{b}_n \end{cases} \tag{1}$$

с интервальными коэффициентами $\boldsymbol{a}_{ij}$ и интервальными правыми частями $\boldsymbol{b}_i$, $i, j = 1, 2, \ldots, n$, или, кратко,

$$\boldsymbol{A}x = \boldsymbol{b}, \tag{2}$$

где $\boldsymbol{A} = (\,\boldsymbol{a}_{ij})$ — интервальная $n \times n$-матрица и $\boldsymbol{b} = (\,\boldsymbol{b}_i)$ — n-вектор. Системы (1)–(2) мы мыслим как семейства обычных точечных (неинтервальных) линейных систем $Ax = b$ той же структуры с матрицами $A \in \boldsymbol{A}$ и векторами $b \in \boldsymbol{b}$.

Обтединённым множеством решений интервальной линейной системы уравнений будем называть множество

$$\Xi(\boldsymbol{A}, \boldsymbol{b}) \;=\; \big\{\, x \in \mathbb{R}^n \mid (\exists\, A \in \boldsymbol{A})(\exists\, b \in \boldsymbol{b})(\, Ax = b\,) \,\big\}, \tag{3}$$

образованное всевозможными решениями точечных систем $Ax = b$ с $A \in \boldsymbol{A}$ и $b \in \boldsymbol{b}$. Для интервальных систем уравнений существуют другие множества решений, более адекватные тем или иным практическим ситуациям. Поскольку мы не рассматриваем их в нашей работе, то в дальнейшем будем называть (3) термином «множество решений».

Всюду далее интервальная матрица $\boldsymbol{A}$ предполагается неособенной, т. е. содержащей только неособенные (невырожденные) точечные матрицы. Тогда множество решений $\Xi(\boldsymbol{A}, \boldsymbol{b})$ системы (1)–(2) ограничено. Нас будет интересовать задача оптимального внешнего интервального оценивания этого множества решений:

> Найти интервальный вектор $\boldsymbol{U} \subset \mathbb{IR}^n$, имеющий наименьшую возможную ширину и содержащий множество решений $\Xi(\boldsymbol{A}, \boldsymbol{b})$ интервальной системы уравнений $\boldsymbol{A}x = \boldsymbol{b}$. (4)

Задача внешнего оценивания множества решений ИСЛАУ является одной из классических задач интервального анализа, различным аспектам решения которой с начала 60-х годов по настоящее время посвящено значительное количество монографий и статей.

Вычисление для множества решений внешних покоординатных оценок с заданной точностью является NP-трудной задачей [11, 12, 20]. В силу трудноразрешимости этой задачи соответствующие численные методы являются экспоненциально трудными и по своей структуре близки переборным алгоритмам дискретной оптимизации.

В работе рассматривается класс эффективных вычислительных алгоритмов для нахождения решений задачи (4), называемый *методами дробления параметров* (или PPS-методами[1]) [4, 5, 23, 24]. Их основная идея – представить «внешнюю задачу» как оптимизационную и применить для ее решения интервальные методы глобальной оптимизации. Задачу поиска нижних оценок множества решений ИСЛАУ можно переформулировать в виде

$$\boxed{\begin{array}{c}\text{Найти} \quad \min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\},\ \nu = 1, 2, \ldots, n, \\ \text{либо как можно более точные их оценки снизу.}\end{array}} \tag{5}$$

Это эквивалентно задаче глобальной оптимизации величины

$$\min\{\, x_\nu \mid Ax = b\}$$

как функции параметров $A \in \boldsymbol{A}$ и $b \in \boldsymbol{b}$.

При поиске верхних оценок множества решений ИСЛАУ будем учитывать, что

$$\max\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\} = -\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, -\boldsymbol{b})\}.$$

Метод дробления параметров заключается в последовательном улучшении оценки $\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$ посредством дробления системы $\boldsymbol{A}x = \boldsymbol{b}$ на две ИСЛАУ-потомка, получающиеся из этой системы рассечением на концы одного интервального элемента либо в матрице $\boldsymbol{A}$, либо в векторе $\boldsymbol{b}$. Так можно делать в силу теоремы Бека-Никеля точные значения $\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$ и $\max\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}, \nu = 1, 2, \ldots, n$, достигаются на решениях точечных систем уравнений $Ax = b$ таких, что матрица A и вектор b образованы концами интервальных элементов из $\boldsymbol{A}$ и $\boldsymbol{b}$ соответственно. Итерационная процедура улучшения оценки строится аналогично широко известному методу «ветвей и границ».

[1] PPS — Partitioning Parameter Set

Результат теоремы Бека-Никеля был значительно усилен И.Роном [21], уточнившим множество вершин матрицы $A \in \boldsymbol{A}$ и вектора $b \in \boldsymbol{b}$, на которых достигаются минимальное и максимальное значения компонент точек из множества $\Xi(\boldsymbol{A}, \boldsymbol{b})$ решений интервальной системы $\boldsymbol{A}x = \boldsymbol{b}$.

Целью данной работы является реализация метода дробления параметров с использованием модификации, основанной на методике Рона, апробация разработанных алгоритмов на тестовых примерах и сравнительный анализ полученных результатов.

Обозначения

Всюду в тексте интервалы и другие интервальные величины (векторы, матрицы и др.) обозначаются жирным шрифтом, например, $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, …, $\boldsymbol{b}$, $\boldsymbol{x}$, тогда как неинтервальные (точечные) величины никак специально не выделяются. Арифметические операции с интервальными величинами — это операции классической интервальной арифметики $\mathbb{IR}$. Под векторами (точечными или интервальными) понимаются вектор-столбцы.

Другие обозначения

$\mathbb{IR}$	классическая интервальная арифметика;
$\mathbb{IR}^n$	множество n-мерных векторов с элементами из $\mathbb{IR}$;
$\underline{\boldsymbol{a}}$, $\inf \boldsymbol{a}$	левый конец интервала $\boldsymbol{a}$;
$\overline{\boldsymbol{a}}$, $\sup \boldsymbol{a}$	правый конец интервала $\boldsymbol{a}$;
$\lvert\boldsymbol{a}\rvert$	абсолютная величина (магнитуда) интервала $\boldsymbol{a}$;
$\langle\boldsymbol{a}\rangle$	мигнитуда интервала $\boldsymbol{a}$;
$\langle\boldsymbol{A}\rangle$	компарант интервальной матрицы $\boldsymbol{A}$;
$\operatorname{mid}\boldsymbol{a}$	середина (медиана) интервала $\boldsymbol{a}$;
$\operatorname{wid}\boldsymbol{a}$	ширина интервала $\boldsymbol{a}$;
$\operatorname{rad}\boldsymbol{a}$	радиус интервала $\boldsymbol{a}$;
Ξ	объединённое множество решений;
$\boldsymbol{A}^{-1}$	«обратная» интервальная матрица;
I	единичная матрица соответствующих размеров;
$\operatorname{int} X$	топологическая внутренность множества X;
$\operatorname{conv} X$	выпуклая оболочка множества X;
$\operatorname{diag}\{z_1, \ldots, z_n\}$	диагональная $n \times n$-матрица с элементами $z_1, \ldots, z_n$ по главной диагонали;
$\rho(A)$	спектральный радиус матрицы A.

К интервальным векторам и матрицам все введённые выше операции за исключением операции «$\langle\cdot\rangle$» — взятия мигнитуды — будут применяться покомпонентно и поэлементно, так что если, к примеру, $\boldsymbol{a} = (\boldsymbol{a}_i)$ — интервальный вектор, то $\operatorname{mid}\boldsymbol{a}$ — это вещественный вектор $(\operatorname{mid}\boldsymbol{a}_i)$.

2 Метод дробления параметров для интервальных линейных систем уравнений

2.1 Стратегия дробления параметров

Пусть дана интервальная линейная система уравнений $\boldsymbol{A}x = \boldsymbol{b}$, где $\boldsymbol{A}$ — интервальная $n \times n$-матрица, содержащая лишь неособенные вещественные матрицы, и $\boldsymbol{b}$ — интервальный n-вектор.

Нас интересует задача нахождения интервальной оболочки множества решений $\Xi(\boldsymbol{A}, \boldsymbol{b})$ системы $\boldsymbol{A}x = \boldsymbol{b}$. Поскольку нам будет удобно находить отдельные покомпонентные оценки для множества решений, то переформулируем задачу в виде: найти $\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$ и $\max\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$, $\nu = 1, 2, \ldots, n$, либо как можно более точные их оценки снизу и сверху соответственно.

Очевидно $\max\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\} = -\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, -\boldsymbol{b})\}$, поэтому мы сосредоточимся на вычислении оптимальных нижних покоординатных оценок множества $\Xi(\boldsymbol{A}, \boldsymbol{b})$, т.е. как можно более точных оценок снизу для $\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$, $\nu = 1, 2, \ldots, n$.

Существует ряд интервальных методов (например, интервальный метод Гаусса, интервальный метод Гаусса-Зейделя, метод Кравчика и т. д.), позволяющих получить внешнюю интервальную оценку множества $\Xi(\boldsymbol{A}, \boldsymbol{b})$. Данные методы позволяют вычислить интервальный вектор, гарантированно содержащий множество решений ИСЛАУ, но не обеспечивают его оптимальность.

Обозначим через *Encl* какой-нибудь фиксированный метод внешнего оценивания множества решений ИСЛАУ и в дальнейшем будем называть его *базовым*. Пусть $\mathit{Encl}(\boldsymbol{A}, \boldsymbol{b}) \in \mathbb{IR}^n$ — интервальный вектор, получаемый с помощью этого метода, так что

$$\mathit{Encl}(\boldsymbol{A}, \boldsymbol{b}) \supseteq \Xi(\boldsymbol{A}, \boldsymbol{b}).$$

Пусть $\Upsilon(\boldsymbol{A}, \boldsymbol{b})$ — нижний конец ν-ой компоненты ($\nu = 1, 2, \ldots, n$) внешней интервальной оценки множества решений, получаемой методом *Encl*, т.е.

$$\Upsilon(\boldsymbol{A}, \boldsymbol{b}) := \underline{(\mathit{Encl}(\boldsymbol{A}, \boldsymbol{b}))_\nu}.$$

Основа PPS-методов — адаптивное дробление области параметров интервальной системы. В случае решения ИСЛАУ (1)–(2) процесс измельчения элементов матрицы $\boldsymbol{A}$ и вектора правых частей $\boldsymbol{b}$ можно значительно упростить, если учесть следующий результат.

Теорема Бека-Никеля [7, 17]. *Для любого* $\nu \in \{1, 2, \ldots, n\}$ *точные покоординатные оценки точек из объединённого множества решений — экстремальные значения* $\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$ *и* $\max\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$ *— достигаются на решениях крайних точечных систем уравнений* $Ax = b$, *т. е. таких, что матрица* A *и вектор* b *образованы концами интервальных элементов из* $\boldsymbol{A}$ *и* $\boldsymbol{b}$ *соответственно.*

Потребуем от базового метода *Encl* удовлетворения следующему условию: оценка $\Upsilon(\boldsymbol{A}, \boldsymbol{b})$ монотонна по включению относительно матрицы $\boldsymbol{A}$ и вектора $\boldsymbol{b}$, т.е. для всех $\boldsymbol{A}', \boldsymbol{A}'' \in \mathbb{IR}^{n\times n}$ и $\boldsymbol{b}', \boldsymbol{b}'' \in \mathbb{IR}^n$ при $\boldsymbol{A}' \subseteq \boldsymbol{A}''$ и $\boldsymbol{b}' \subseteq \boldsymbol{b}''$ верно неравенство

$$\Upsilon(\boldsymbol{A}', \boldsymbol{b}') \geq \Upsilon(\boldsymbol{A}'', \boldsymbol{b}''). \tag{6}$$

В дальнейшем будем обозначать ИСЛАУ-потомки, получающиеся из интервальной системы $\boldsymbol{Q}x = \boldsymbol{r}$ рассечением на концы одного интервального элемента либо в матрице $\boldsymbol{Q}$, либо в векторе $\boldsymbol{r}$, через $\boldsymbol{Q}'x = \boldsymbol{r}'$ и $\boldsymbol{Q}''x = \boldsymbol{r}''$. Здесь $\boldsymbol{Q}'$ и $\boldsymbol{Q}''$ — матрицы, полученные из $\boldsymbol{Q}$ заменой элемента $\boldsymbol{q}_{ij}$ на $\underline{\boldsymbol{q}}_{ij}$ и $\overline{\boldsymbol{q}}_{ij}$ соответственно, а $\boldsymbol{r}'$ и $\boldsymbol{r}''$ — векторы, полученные из $\boldsymbol{r}$ заменой элемента $\boldsymbol{r}_i$ на $\underline{\boldsymbol{r}}_i$ и $\overline{\boldsymbol{r}}_i$ соответственно.

Решив две интервальных «системы-потомка» $\boldsymbol{Q}'x = \boldsymbol{r}$ и $\boldsymbol{Q}''x = \boldsymbol{r}$, можно получить в силу свойства (6) более точную оценку снизу для искомого $\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{Q}, \boldsymbol{r})\}$ в виде

$$\min\{\, \Upsilon(\boldsymbol{Q}', \boldsymbol{r}), \Upsilon(\boldsymbol{Q}'', \boldsymbol{r})\}.$$

Аналогичный эффект имеет и распадение в векторе правых частей $\boldsymbol{r}$ какого-нибудь интервального элемента $\boldsymbol{r}_i$ на концы $\underline{\boldsymbol{r}}_i$ и $\overline{\boldsymbol{r}}_i$.

Процедуру улучшения оценки для $\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{Q}, \boldsymbol{r})\}$ посредством дробления параметров можно повторить по отношению к системам-потомкам $\boldsymbol{Q}'x = \boldsymbol{r}'$ и $\boldsymbol{Q}''x = \boldsymbol{r}''$, затем снова разбить потомков и улучшить оценку и т.д. Данную процедуру можно оформить в соответствии с хорошо известным методом «ветвей и границ»:

во-первых, все возникающие в процессе дробления исходной ИСЛАУ системы $\boldsymbol{Q}x = \boldsymbol{r}$, где $\boldsymbol{Q} = (\,\boldsymbol{q}_{ij}) \subseteq \boldsymbol{A}$ и $\boldsymbol{r} = (\,\boldsymbol{r}_i) \subseteq \boldsymbol{b}$, вместе с их оценками $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$ организуем в некоторый список $\mathcal{L}$,

во-вторых, дроблению каждый раз будем подвергать лишь ту из интервальных систем-потомков $\boldsymbol{Q}x = \boldsymbol{r}$, которая обеспечивает наименьшую на данный момент оценку $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$.

Образующие список $\mathcal{L}$ записи упорядочим по возрастанию значений оценки $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$. Первую запись списка, а также соответствующую ИСЛАУ $\boldsymbol{Q}x = \boldsymbol{r}$ и наименьшую в списке оценку $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$ будем называть *ведущими* на данном шаге.

Если T — общее количество интервальных (с ненулевой шириной) элементов в матрице $\boldsymbol{A}$ и векторе $\boldsymbol{b}$ правых частей исходной системы (в общем случае $T \leq (n+1)n$), то описанный алгоритм остановится не более чем через 2^T шагов.

Традиционно ведущие брусы параметров на каждом шаге дробятся по самой длинной грани. Этот способ дробления обеспечивает сходимость использующих его методов (из-за его конечности). Однако в целях обеспечения наиболее быстрой сходимости алгоритма на его первых шагах рекомендуется рассекать ведущие брусы по элементам, на которых достигается максимум величин

$$\left|\frac{\partial x_\nu(\boldsymbol{Q}, \boldsymbol{r})}{\partial q_{ij}}\right| \cdot \operatorname{wid} \boldsymbol{q}_{ij}, \quad \left|\frac{\partial x_\nu(\boldsymbol{Q}, \boldsymbol{r})}{\partial r_i}\right| \cdot \operatorname{wid} \boldsymbol{r}_i, \qquad i, j \in \{1, 2, \ldots, n\}, \tag{7}$$

т. е. по элементам, на которых достигается максимум произведения модуля интервальной оценки производной решения на ширину соответствующего интервала.

2.2 Тест на монотонность

Пусть дана интервальная система линейных алгебраических уравнений $\boldsymbol{Q}x = \boldsymbol{r}$, и нам известны

$$\frac{\partial x_\nu(\boldsymbol{Q}, \boldsymbol{r})}{\partial q_{ij}} \qquad \text{и} \qquad \frac{\partial x_\nu(\boldsymbol{Q}, \boldsymbol{r})}{\partial r_i}$$

— интервальные расширения соответствующих частных производных

$$\frac{\partial x_\nu(Q, r)}{\partial q_{ij}} \qquad \text{и} \qquad \frac{\partial x_\nu(Q, r)}{\partial r_i}$$

от ν-ой компоненты вектора решения системы $\boldsymbol{Q}x = \boldsymbol{r}$ по ij-му элементу матрицы Q и j-му элементу вектора r. Если интервальная $n \times n$-матрица $\tilde{\boldsymbol{Q}}$,

и интервальный n-вектор $\tilde{\boldsymbol{b}}$ образованы из элементов

$$\tilde{\boldsymbol{q}}_{ij} = \begin{cases} [\underline{\boldsymbol{q}}_{ij}, \underline{\boldsymbol{q}}_{ij}] & \text{при } \dfrac{\partial x_\nu(\boldsymbol{Q}, \boldsymbol{r})}{\partial q_{ij}} \geq 0, \\ [\overline{\boldsymbol{q}}_{ij}, \overline{\boldsymbol{q}}_{ij}] & \text{при } \dfrac{\partial x_\nu(\boldsymbol{Q}, \boldsymbol{r})}{\partial q_{ij}} \leq 0, \\ \boldsymbol{q}_{ij}, & \text{при } \operatorname{int}\dfrac{\partial x_\nu(\boldsymbol{Q}, \boldsymbol{r})}{\partial q_{ij}} \ni 0, \end{cases} \tag{8}$$

$$\tilde{\boldsymbol{r}}_i = \begin{cases} [\underline{\boldsymbol{r}}_i, \underline{\boldsymbol{r}}_i] & \text{при } \dfrac{\partial x_\nu(\boldsymbol{Q}, \boldsymbol{r})}{\partial r_i} \geq 0, \\ [\overline{\boldsymbol{r}}_i, \overline{\boldsymbol{r}}_i] & \text{при } \dfrac{\partial x_\nu(\boldsymbol{Q}, \boldsymbol{r})}{\partial r_i} \leq 0, \\ \boldsymbol{r}_i, & \text{при } \operatorname{int}\dfrac{\partial x_\nu(\boldsymbol{Q}, \boldsymbol{r})}{\partial r_i} \ni 0, \end{cases} \tag{9}$$

то, очевидно,

$$\min\{\, x_\nu \mid x \in \Xi(\tilde{\boldsymbol{Q}}, \tilde{\boldsymbol{r}})\} = \min\{\, x_\nu \mid x \in \Xi(\boldsymbol{Q}, \boldsymbol{r})\}.$$

Поскольку количество существенно интервальных (с ненулевой шириной) элементов в $\tilde{\boldsymbol{Q}}$ и $\tilde{\boldsymbol{r}}$ может быть значительно меньшим, чем в $\boldsymbol{Q}$ и $\boldsymbol{r}$, то переходя от исходной ИСЛАУ $\boldsymbol{Q}x = \boldsymbol{r}$ к решению системы $\tilde{\boldsymbol{Q}}x = \tilde{\boldsymbol{r}}$, мы, тем самым, упрощаем задачу вычисления $\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{Q}, \boldsymbol{r})\}$.

Известно, что если $Y = (y_{ij})$ — обратная матрица для $Q = (q_{ij})$, то производные решения вещественной линейной системы $Qx = r$ по ее коэффициентам определяются формулами [2]

$$\frac{\partial x_\nu}{\partial q_{ij}} = -y_{\nu i} x_i, \qquad \frac{\partial x_\nu}{\partial r_i} = y_{\nu i}.$$

Следовательно, в случае, когда $\boldsymbol{Y} = (\boldsymbol{y}_{ij})$ — «обратная интервальная матрица» для $\boldsymbol{Q}$, т. е.

$$\boldsymbol{Y} \supseteq \{\, Q^{-1} \mid Q \in \boldsymbol{Q}\},$$

а $\boldsymbol{x}_j$ — j-ая компонента некоторого интервального вектора $\boldsymbol{x}$, такого что $\boldsymbol{x} \supseteq \Xi(\boldsymbol{Q}, \boldsymbol{r})$, то мы можем принять следующие интеральные оценки производных

$$\frac{\partial x_\nu(\boldsymbol{Q},\boldsymbol{r})}{\partial q_{ij}} = -\boldsymbol{y}_{\nu i}\boldsymbol{x}_j, \qquad \frac{\partial x_\nu(\boldsymbol{Q},\boldsymbol{r})}{\partial r_i} = \boldsymbol{y}_{\nu i}. \tag{10}$$

Таким образом, перед дроблением ведущей ИСЛАУ целесообразно выполнить исследование ее на монотонность и овеществление («сжатие») соответствующих интервальных элементов на основе (8)-(9). Тогда в дробимой ИСЛАУ $\boldsymbol{Q}x = \boldsymbol{r}$ ненулевую ширину будут иметь лишь те элементы, производные по которым оцениваются интервалами, содержащими внутри себя нуль.

2.3 Вычислительная схема алгоритма

Алгоритм, представленный в Табл. 1, описывает метод дробления параметров для внешнего оценивания объединённого множества решений ИСЛАУ. Для начала работы этого алгоритма необходимо

- найти предварительные внешние оценки для множества решений исходной системы и «обратной интервальной матрицы», т. е. вычислить $\boldsymbol{x} = \mathit{Encl}(\boldsymbol{A},\boldsymbol{b}) \supseteq \Xi(\boldsymbol{A},\boldsymbol{b})$ и $\boldsymbol{Y} \supseteq \boldsymbol{A}^{-1}$,
- назначить точность оценки $\varepsilon > 0$,
- присвоить $\boldsymbol{Q} \leftarrow \boldsymbol{A}$ и $\boldsymbol{r} \leftarrow \boldsymbol{b}$,
- положить $\Upsilon(\boldsymbol{Q},\boldsymbol{r}) \leftarrow \underline{\boldsymbol{x}}_\nu$ и $\omega \leftarrow +\infty$,
- инициализировать рабочий список $\mathcal{L}$ записью $(\boldsymbol{Q},\boldsymbol{r},\Upsilon(\boldsymbol{Q},\boldsymbol{r}),\boldsymbol{Y},\boldsymbol{x})$.

В процессе выполнения алгоритма мы будем поддерживать список $\mathcal{L}$, состоящий из записей вида

$$(\boldsymbol{Q},\boldsymbol{r},\Upsilon(\boldsymbol{Q},\boldsymbol{r}),\boldsymbol{Y},\boldsymbol{x}),$$

где

$\boldsymbol{Q}$ — интервальная $n \times n$-матрица, $\boldsymbol{Q} \subseteq \boldsymbol{A}$;

$\boldsymbol{r}$ — интервальный n-вектор, $\boldsymbol{r} \subseteq \boldsymbol{b}$;

$\Upsilon(\boldsymbol{Q},\boldsymbol{r}) = \underline{(\mathit{Encl}(\boldsymbol{Q},\boldsymbol{r}))_\nu}$ — нижний конец ν-ой компоненты внешней интервальной оценки множества решений системы, получаемой методом Encl;

$\boldsymbol{Y}$ — внешняя оценка «обратной интервальной матрицы» $\boldsymbol{Q}^{-1}$;

$\boldsymbol{x} = \mathit{Encl}(\boldsymbol{Q},\boldsymbol{r})$ — внешняя интервальная оценка множества решений системы, полученная на текущем шаге алгоритма.

На каждом итерационном шаге по формулам (10) вычисляем интервальные расширения для производных решения ИСЛАУ и «сжимаем» в соответствии с (8)–(9) те элементы матрицы $\boldsymbol{Q}$ и вектора $\boldsymbol{r}$, на которых была выявлена монотонная зависимость ν-ой компоненты вектора решения от элементов матрицы системы или вектора её правой части.

Рассекаем ведущую ИСЛАУ по интервальному элементу $\boldsymbol{h}$, которому соответствует наибольшая из величин (7), на две системы-потомка $\boldsymbol{Q'}x = \boldsymbol{r'}$ и $\boldsymbol{Q''}x = \boldsymbol{r''}$. Соответствующие этим системам записи $(\boldsymbol{Q'}, \boldsymbol{r'}, \Upsilon(\boldsymbol{Q'}, \boldsymbol{r'}), \boldsymbol{Y'}, \boldsymbol{x'})$ и $(\boldsymbol{Q''}, \boldsymbol{r''}, \Upsilon(\boldsymbol{Q''}, \boldsymbol{r''}), \boldsymbol{Y''}, \boldsymbol{x''})$ заносим в список $\mathcal{L}$ в порядке возрастания значений третьего поля.

Условием остановки метода дробления параметров является «овеществление» ведущей ИСЛАУ $\boldsymbol{Q}x = \boldsymbol{r}$, когда матрица $\boldsymbol{Q}$ и вектор $\boldsymbol{r}$ — точечные (неинтервальные). Заметим однако, что некоторые итерационные методы внешнего оценивания множества решений (например, метод Гаусса-Зейделя) дают точную оценку $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$ даже в том случае, когда часть элементов в $\boldsymbol{Q}$ и $\boldsymbol{r}$ могут быть интервалами. Поэтому достаточным условием оптимальности вычисленной оценки $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$, т.е. для того, чтобы она была равна $\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$, является выполнение следующего «условия точности»: $\Upsilon(Q, r) = (Q^{-1}r)_\nu$ для всех $Q \in \mathbb{R}^{n\times n}$ и $r \in \mathbb{R}^n$.

Наряду с оценкой $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$ для интервальных систем $\boldsymbol{Q}x = \boldsymbol{r}$, порождаемых алгоритмом, вычисляются ещё и величины $\Upsilon(\operatorname{mid} \boldsymbol{Q}, \operatorname{mid} \boldsymbol{r})$. Очевидно, что

$$\Upsilon(\operatorname{mid} \boldsymbol{Q}, \operatorname{mid} \boldsymbol{r}) \geq \Upsilon(\boldsymbol{Q}, \boldsymbol{r}),$$

и значения $\Upsilon(\operatorname{mid} \boldsymbol{Q}, \operatorname{mid} \boldsymbol{r})$ приближают $\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$ сверху: если для каждого шага алгоритма определять

$$\omega = \min \Upsilon(\operatorname{mid} \boldsymbol{Q}, \operatorname{mid} \boldsymbol{r}) \tag{11}$$

по всем интервальным линейным системам $\boldsymbol{Q}x = \boldsymbol{r}$, когда-либо побывавшим в списке $\mathcal{L}$ до этого шага, то

$$\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\} \leq \omega.$$

С другой стороны, если $\boldsymbol{Q}x = \boldsymbol{r}$ — ведущая ИСЛАУ, то

$$\Upsilon(\boldsymbol{Q}, \boldsymbol{r}) \leq \min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\},$$

и потому одним из критериев остановки алгоритма может служить достижение требуемой малости величины $(\omega - \Upsilon(\boldsymbol{Q}, \boldsymbol{r}))$.

Заметим, что запись $(\boldsymbol{Q}, \boldsymbol{r}, \Upsilon(\boldsymbol{Q}, \boldsymbol{r}), \boldsymbol{Y}, \boldsymbol{x})$, удовлетворяющая на некотором шаге условию

$$\Upsilon(\boldsymbol{Q}, \boldsymbol{r}) > \omega,$$

при дальнейшем выполнении алгоритма уже никогда не станет ведущей, поэтому такая запись либо вообще не вносится в список, либо удаляется при чистке списка. Чистка списка проводится после изменения (уменьшения) параметра ω.

Таблица 1. Алгоритм внешнего оценивания множества решений ИСЛАУ методом дробления параметров.

DO WHILE ((ведущая оценка $\Upsilon(\boldsymbol{Q},\boldsymbol{r})$ неточна) и $(\omega - \Upsilon(\boldsymbol{Q},\boldsymbol{r}) > \varepsilon)$)

по формулам (10) вычисляем интервальные расширения производных $\frac{\partial x_\nu(\boldsymbol{Q},\boldsymbol{r})}{\partial q_{ij}}$ и $\frac{\partial x_\nu(\boldsymbol{Q},\boldsymbol{r})}{\partial r_i}$ по элементам $\boldsymbol{q}_{ij}$ и $\boldsymbol{r}_i$ с ненулевой шириной;

сжимаем в соответствии с (8)-(9) в $\boldsymbol{Q}$ и $\boldsymbol{r}$ элементы, на которых выявлена монотонная зависимость x_ν от q_{ij} и r_i ;

ищем в ведущей ИСЛАУ $\boldsymbol{Q}x = \boldsymbol{r}$ интервальный элемент $\boldsymbol{h}$, которому соответствует наибольшее из произведений (7);

порождаем интервальные системы-потомки $\boldsymbol{Q'}x = \boldsymbol{r'}$ и $\boldsymbol{Q''}x = \boldsymbol{r''}$:

если $\boldsymbol{h} = \boldsymbol{q}_{kl}$ для некоторых $k, l \in \{1, 2, \ldots, n\}$, то полагаем

$\boldsymbol{q}'_{ij} \leftarrow \boldsymbol{q}''_{ij} \leftarrow \boldsymbol{q}_{ij}$ для $(i,j) \neq (k,l)$, $\boldsymbol{q}'_{kl} \leftarrow \underline{\boldsymbol{q}}_{kl}, \boldsymbol{q}''_{kl} \leftarrow \overline{\boldsymbol{q}}_{kl}$,

$\boldsymbol{r'} \leftarrow \boldsymbol{r''} \leftarrow \boldsymbol{r}$;

если $\boldsymbol{h} = \boldsymbol{r}_k$ для некоторого $k \in \{1, 2, \ldots, n\}$, то полагаем

$\boldsymbol{r}'_i \leftarrow \boldsymbol{r}''_i \leftarrow \boldsymbol{r}_i$ для $i \neq k$, $\boldsymbol{r}'_k \leftarrow \underline{\boldsymbol{r}}_k, \boldsymbol{r}''_k \leftarrow \overline{\boldsymbol{r}}_k$,

$\boldsymbol{Q'} \leftarrow \boldsymbol{Q''} \leftarrow \boldsymbol{Q}$;

вычисляем брусы $\boldsymbol{x'} \leftarrow \mathit{Encl}(\boldsymbol{Q'}, \boldsymbol{r'})$ и $\boldsymbol{x''} \leftarrow \mathit{Encl}(\boldsymbol{Q''}, \boldsymbol{r''})$;

вычисляем оценки $\Upsilon(\boldsymbol{Q'}, \boldsymbol{r'})$ и $\Upsilon(\boldsymbol{Q''}, \boldsymbol{r''})$;

вычисляем внешние оценки для обратных интервальных матриц $\boldsymbol{Y'} \supseteq (\boldsymbol{Q'})^{-1}$ и $\boldsymbol{Y''} \supseteq (\boldsymbol{Q''})^{-1}$

вычисляем оценки $\Upsilon(\mathrm{mid}\, \boldsymbol{Q'}, \mathrm{mid}\, \boldsymbol{r'})$ и $\Upsilon(\mathrm{mid}\, \boldsymbol{Q''}, \mathrm{mid}\, \boldsymbol{r''})$,

полагаем $\mu \leftarrow \min\{\, \Upsilon(\mathrm{mid}\, \boldsymbol{Q'}, \mathrm{mid}\, \boldsymbol{r'}), \Upsilon(\mathrm{mid}\, \boldsymbol{Q''}, \mathrm{mid}\, \boldsymbol{r''})\}$;

удаляем из $\mathcal{L}$ бывшую ведущую запись $(\boldsymbol{Q}, \boldsymbol{r}, \Upsilon(\boldsymbol{Q}, \boldsymbol{r}), \boldsymbol{Y}, \boldsymbol{x})$;

если $\Upsilon(\mathrm{mid}\, \boldsymbol{Q'}, \mathrm{mid}\, \boldsymbol{r'}) \leq \omega$, то заносим запись $(\boldsymbol{Q'}, \boldsymbol{r'}, \Upsilon(\boldsymbol{Q'}, \boldsymbol{r'}), \boldsymbol{Y'}, \boldsymbol{x'})$ в список $\mathcal{L}$ в порядке возрастания значений третьего поля;

если $\Upsilon(\mathrm{mid}\, \boldsymbol{Q''}, \mathrm{mid}\, \boldsymbol{r''}) \leq \omega$, то заносим запись $(\boldsymbol{Q''}, \boldsymbol{r''}, \Upsilon(\boldsymbol{Q''}, \boldsymbol{r''}), \boldsymbol{Y''}, \boldsymbol{x''})$ в список $\mathcal{L}$ в порядке возрастания значений третьего поля;

если $\omega > \mu$, то полагаем $\omega \leftarrow \mu$ и чистим список $\mathcal{L}$: удаляем из него все такие записи $(\boldsymbol{Q}, \boldsymbol{r}, \Upsilon(\boldsymbol{Q}, \boldsymbol{r}), \boldsymbol{Y}, \boldsymbol{x})$, что $\Upsilon(\boldsymbol{Q}, \boldsymbol{r}) > \omega$;

END DO

3 Модификация Рона

3.1 Методика Рона

Известно, что объединённое множество решений интервальной линейной алгебраической системы имеет следующую элегантную характеризацию Оеттли-Прагера:

Теорема Оеттли-Прагера [18]. *Точка* $x \in \mathbb{R}^n$ *принадлежит множеству* $\Xi(\boldsymbol{A}, \boldsymbol{b})$ *решений системы* $\boldsymbol{A}x = \boldsymbol{b}$ *тогда и только тогда, когда*

$$|(\operatorname{mid} \boldsymbol{A})x - \operatorname{mid} \boldsymbol{b}| \leq \operatorname{rad} \boldsymbol{A} \cdot |x| + \operatorname{rad} \boldsymbol{b}. \tag{12}$$

Суть теоремы Оеттли-Прагера состоит в том, что множество решений $\Xi(\boldsymbol{A}, \boldsymbol{b})$ описывается простым векторным неравенством (12).

Значения x, обращающие (12) в равенство, будем называть *экстремальными решениями* неравенства Оеттли-Прагера. Множество экстремальных решений обладает рядом замечательных свойств, выводом которых мы сейчас займёмся.

Для вектора $x \in \mathbb{R}^n$ условимся обозначать через $\operatorname{Sgn}(x)$ диагональную матрицу

$$\operatorname{diag}\{\operatorname{sgn} x_1, \operatorname{sgn} x_2, \ldots, \operatorname{sgn} x_n\},$$

образованную знаками компонент вектора x. Так как $|r| = r \cdot \operatorname{sgn} r$ для любого вещественного числа r, то система уравнений

$$|(\operatorname{mid} \boldsymbol{A})x - \operatorname{mid} \boldsymbol{b}| = \operatorname{rad} \boldsymbol{A} \cdot |x| + \operatorname{rad} \boldsymbol{b},$$

определяющая экстремальные решения, может быть преобразована к виду

$$\operatorname{Sgn}\left((\operatorname{mid} \boldsymbol{A})x - \operatorname{mid} \boldsymbol{b}\right) \cdot \left((\operatorname{mid} \boldsymbol{A})x - \operatorname{mid} \boldsymbol{b}\right) = \operatorname{rad} \boldsymbol{A} \cdot |x| + \operatorname{rad} \boldsymbol{b},$$

или

$$\operatorname{Sgn}(y)\left((\operatorname{mid} \boldsymbol{A})x - \operatorname{mid} \boldsymbol{b}\right) = \operatorname{rad} \boldsymbol{A} \cdot |x| + \operatorname{rad} \boldsymbol{b},$$

где $y = \left((\operatorname{mid} \boldsymbol{A})x - \operatorname{mid} \boldsymbol{b}\right)$.

Учитывая, что $(\operatorname{Sgn}(y))^{-1} = \operatorname{Sgn}(y)$, получим

$$(\operatorname{mid} \boldsymbol{A})x - \operatorname{mid} \boldsymbol{b} = \operatorname{Sgn}(y) \cdot \operatorname{rad} \boldsymbol{A} \cdot |x| + \operatorname{Sgn}(y) \cdot \operatorname{rad} \boldsymbol{b}.$$

После группировки членов в уравнении имеем

$$(\operatorname{mid} \boldsymbol{A})x - \operatorname{Sgn}(y) \cdot \operatorname{rad} \boldsymbol{A} \cdot |x| = \operatorname{mid} \boldsymbol{b} + \operatorname{Sgn}(y) \cdot \operatorname{rad} \boldsymbol{b},$$

или

$$((\operatorname{mid} \boldsymbol{A}) - \operatorname{Sgn}(y) \cdot \operatorname{rad} \boldsymbol{A} \cdot \operatorname{Sgn}(x)) \cdot x = \operatorname{mid} \boldsymbol{b} + \operatorname{Sgn}(y) \cdot \operatorname{rad} \boldsymbol{b}.$$

Условимся обозначать через $\mathcal{E}$ множество n-векторов с компонентами ± 1. Очевидно, что множество $\mathcal{E}$ имеет мощность 2^n. Для заданных векторов $\sigma, \tau \in \mathcal{E}$ определим матрицы T_σ, $A^{\sigma\tau} = \{a_{ij}^{\sigma\tau}\}$ и вектор $b^\sigma = \{b_i^\sigma\}$ следующим образом

$$T_\sigma = \operatorname{diag}\{\sigma_1, \ldots, \sigma_n\},$$

$$A^{\sigma\tau} = (\operatorname{mid} \boldsymbol{A}) - \operatorname{Sgn}(y) \cdot \operatorname{rad} \boldsymbol{A} \cdot \operatorname{Sgn}(x) = \operatorname{mid} \boldsymbol{A} - T_\sigma \cdot \operatorname{rad} \boldsymbol{A} \cdot T_\tau,$$

$$b^\sigma = \operatorname{mid} \boldsymbol{b} + \operatorname{Sgn}(y) \cdot \operatorname{rad} \boldsymbol{b} = \operatorname{mid} \boldsymbol{b} + T_\sigma \cdot \operatorname{rad} \boldsymbol{b}.$$

Из этого определения следует, что

$$a_{ij}^{\sigma\tau} = \begin{cases} \overline{\boldsymbol{a}}_{ij}, & \text{если } \sigma_i \tau_j = -1, \\ \underline{\boldsymbol{a}}_{ij}, & \text{если } \sigma_i \tau_j = 1, \end{cases} \tag{13}$$

$$b_i^\sigma = \begin{cases} \overline{\boldsymbol{b}}_i, & \text{если } \sigma_i = 1, \\ \underline{\boldsymbol{b}}_i, & \text{если } \sigma_i = -1. \end{cases} \tag{14}$$

Теорема Рона об экстремальных решениях [6]. *Пусть $n \times n$-матрица $\boldsymbol{A}$ неособенна и пусть $\boldsymbol{b}$ — интервальный n-вектор. Тогда для каждого $\sigma \in \mathcal{E}$ уравнение*

$$\operatorname{mid} \boldsymbol{A} \cdot x - T_\sigma \cdot \operatorname{rad} \boldsymbol{A} \cdot |x| = b^\sigma \tag{15}$$

имеет единственное решение x^σ, принадлежащее $\Xi(\boldsymbol{A}, \boldsymbol{b})$, и справедливо равенство

$$\operatorname{conv} \Xi(\boldsymbol{A}, \boldsymbol{b}) = \operatorname{conv}\{\, x^\sigma \mid \sigma \in \mathcal{E}\}. \tag{16}$$

Заметим, что вследствие (13) и (14) матрица $A^{\sigma\tau}$ и вектор b^σ образованы наборами концов элементов $\boldsymbol{A}$ и $\boldsymbol{b}$ соответственно, поэтому система (15) является крайней точечной системой для исходной интервальной системы $\boldsymbol{A}x = \boldsymbol{b}$.

Таким образом, объединённое множество $\Xi(\boldsymbol{A}, \boldsymbol{b})$ решений системы содержит однозначно определяемый конечный набор экстремальных решений, причём число 2^n экстремальных решений значительно меньше числа 2^{n^2+n} всех крайних решений. Выпуклая оболочка множества экстремальных решений совпадает с выпуклой оболочкой всего множества решений.

Если матрица $\boldsymbol{A}$ неособенна, то для каждого $\sigma \in \mathcal{E}$, экстремальное решение x^σ может быть рассчитано по конечному алгоритму, который называется «знакосогласующим». Целью данного алгоритма является достижение «согласованности знаков» векторов τ и x, т. е. $\tau_j x_j \geq 0$ для каждого j (подробнее см. [19]). Из (16) получаем

$$\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\} = \min\{\, x_\nu^\sigma \mid \sigma \in \mathcal{E}\},$$

где $\nu = 1, \ldots, n$, что в комбинации со знакосогласующим алгоритмом дает конечную процедуру вычисления оптимальной внешней оценки множества решений $\Xi(\boldsymbol{A}, \boldsymbol{b})$.

3.2 Модификация Рона в методе дробления параметров

Описанный выше метод дробления параметров можно модифицировать, если использовать результаты И.Рона, уточнившие множество вершин матрицы $\boldsymbol{A}$ и вектора $\boldsymbol{b}$, на которых достигаются оптимальные внешние оценки множества решений ИСЛАУ [4].

Для неособенной матрицы $\boldsymbol{A}$ экстремальные решения могут достигаться лишь на множестве 4^n матриц $A^{\sigma\tau}$ и ассоциированных с ними векторов b^σ, т. е.

$$\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\} = \min_{\sigma,\tau \in \mathcal{E}} \left((A^{\sigma\tau})^{-1} b^\sigma\right)_\nu, \tag{17}$$

$$\max\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\} = \max_{\sigma,\tau \in \mathcal{E}} \left((A^{\sigma\tau})^{-1} b^\sigma\right)_\nu \tag{18}$$

для каждого индекса $\nu = 1, 2, \ldots, n$.

Из (17)-(18) следует, что в процессе дробления исходной ИСЛАУ мы должны отслеживать концы задействованных интервальных элементов матрицы системы и вектора правых частей. Для этого с каждой интервальной линейной системой $\boldsymbol{Q}x = \boldsymbol{r}$ мы связываем

1) вспомогательную целочисленную $n \times n$-матрицу $W = \{w_{ij}\}$, элементы которой равны ± 1 или 0, такую что

$$w_{ij} = \begin{cases} -1, & \text{если} \quad \boldsymbol{q}_{ij} = \overline{\boldsymbol{a}}_{ij}, \\ 0, & \text{если} \quad \boldsymbol{q}_{ij} = \boldsymbol{a}_{ij}, \\ 1, & \text{если} \quad \boldsymbol{q}_{ij} = \underline{\boldsymbol{a}}_{ij}, \end{cases} \tag{19}$$

и

2) вспомогательные целочисленные n-векторы $s = \{s_i\}$ и $t = \{t_j\}$ с компонентами ± 1 или 0, такие что

$$w_{ij} = s_i t_j \tag{20}$$

для всех $i, j = 1, 2, \ldots, n$, и

$$s_i = \begin{cases} -1, & \text{если} \quad \boldsymbol{r}_i = \underline{\boldsymbol{b}}_i, \\ 0, & \text{если} \quad \boldsymbol{r}_i = \boldsymbol{b}_i, \\ 1, & \text{если} \quad \boldsymbol{r}_i = \overline{\boldsymbol{b}}_i. \end{cases} \tag{21}$$

Матрицу W и векторы s, t будем называть *контрольными*. Значения t_j определяются через матрицу W и вектор s посредством соотношения (20). Контрольные векторы s и t являются «приближениями» к векторам σ и τ, входящим в равенства (17) – (18), а контрольная матрица $W = s\,t^{\top}$ — это «приближение» к матрице $\left(\sigma\,\tau^{\top}\right)$. В начале работы алгоритма элементы W, s и t инициализируем нулями, далее в процессе дробления исходной ИСЛАУ они перевычисляются.

На каждом шаге алгоритма с учетом значений контрольных матрицы W и вектора s выполняется процедура дробления интервального элемента $\boldsymbol{h}$ ведущей ИСЛАУ $\boldsymbol{Q}x = \boldsymbol{r}$. Если $\boldsymbol{h}$ есть $\boldsymbol{q}_{kl}$ из матрицы $\boldsymbol{Q}$ ($\boldsymbol{r}_k$ из вектора правой части $\boldsymbol{r}$), то в случае $w_{kl} = 0$ ($s_k = 0$) порождается две системы-потомка $\boldsymbol{Q}'x = \boldsymbol{r}'$ и $\boldsymbol{Q}''x = \boldsymbol{r}''$. Если $w_{kl} = \pm 1$ ($s_k = \pm 1$), то оставляем в рабочем списке только одного потомка (какого именно, зависит от знака элемента w_{kl} (s_k)). Алгоритм порождения систем-потомков представлен в Табл. 2.

Заметим, что в новой процедуре дробления мы отбрасываем лишь те системы, которые либо не принадлежат множеству точечных систем $\{A^{\sigma\tau}x = b^{\sigma} \mid \sigma, \tau \in \mathcal{E}\}$, либо не содержат такие системы. Поэтому отбрасывание систем-потомков не нарушает то свойство ведущих оценок $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$, что они приближают искомый $\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$ снизу (здесь предполагается, что оценка $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$ монотонна по включению относительно матрицы системы и вектора правых частей).

После дробления ведущей ИСЛАУ необходимо вычислить контрольные матрицы и векторы для систем-потомков. Пусть в результате дробления порождаются две системы-потомка $\boldsymbol{Q}'x = \boldsymbol{r}'$ и $\boldsymbol{Q}''x = \boldsymbol{r}''$, определенные в Табл. 2. Обозначим соответствующие им новые контрольные матрицы через W' и W'', а пары новых контрольных векторов — через s', t' и s'', t''.

Новые контрольные матрицы и векторы вычисляем следующим образом:

1. присваиваем $W'' \leftarrow W' \leftarrow W$, $s'' \leftarrow s' \leftarrow s$, $t'' \leftarrow t' \leftarrow t$;

Таблица 2. Порождение систем-потомков.

IF (рассекается $\boldsymbol{q}_{kl}$) THEN
  IF ($w_{kl} = 0$) THEN
    порождаем системы $\boldsymbol{Q}'x = \boldsymbol{r}'$ и $\boldsymbol{Q}''x = \boldsymbol{r}''$ так, что
      $\boldsymbol{q}'_{ij} \leftarrow \boldsymbol{q}''_{ij} \leftarrow \boldsymbol{q}_{ij}$ для $(i,j) \neq (k,l)$,
      $\boldsymbol{q}'_{kl} \leftarrow \underline{\boldsymbol{q}}_{kl}$, $\boldsymbol{q}''_{kl} \leftarrow \overline{\boldsymbol{q}}_{kl}$, $\boldsymbol{r}' \leftarrow \boldsymbol{r}'' \leftarrow \boldsymbol{r}$;
  ELSE
    порождаем систему $\boldsymbol{Q}'x = \boldsymbol{r}'$ так, что
      $\boldsymbol{r}' \leftarrow \boldsymbol{r}$, $\boldsymbol{q}'_{ij} \leftarrow \boldsymbol{q}_{ij}$ для $(i,j) \neq (k,l)$,
      $$\boldsymbol{q}'_{kl} \leftarrow \begin{cases} \underline{\boldsymbol{q}}_{kl} & \text{для} \quad w_{kl} = 1, \\ \overline{\boldsymbol{q}}_{kl} & \text{для} \quad w_{kl} = -1; \end{cases}$$
  END IF
END IF

IF (рассекается $\boldsymbol{r}_k$) THEN
  IF ($s_k = 0$) THEN
    порождаем системы $\boldsymbol{Q}'x = \boldsymbol{r}'$ и $\boldsymbol{Q}''x = \boldsymbol{r}''$ так, что
      $\boldsymbol{Q}' \leftarrow \boldsymbol{Q}'' \leftarrow \boldsymbol{Q}$,
      $\boldsymbol{r}'_i \leftarrow \boldsymbol{r}_i$ для $i \neq k$, $\boldsymbol{r}'_k \leftarrow \underline{\boldsymbol{r}}_k$, $\boldsymbol{r}''_k \leftarrow \overline{\boldsymbol{r}}_k$;
  ELSE
    порождаем систему $\boldsymbol{Q}'x = \boldsymbol{r}'$ так, что
      $\boldsymbol{Q}' \leftarrow \boldsymbol{Q}$, $\boldsymbol{r}'_i \leftarrow \boldsymbol{r}_i$ для $i \neq k$,
      $$\boldsymbol{r}'_k \leftarrow \begin{cases} \underline{\boldsymbol{r}}_k & \text{для} \quad s_k = -1, \\ \overline{\boldsymbol{r}}_k & \text{для} \quad s_k = 1; \end{cases}$$
  END IF
END IF

2. (a) если рассечённым элементом был $\boldsymbol{q}_{kl}$ из матрицы $\boldsymbol{Q}$, то присваиваем $w'_{kl} \leftarrow 1$ и $w''_{kl} \leftarrow -1$;
   (b) если рассечённым элементом был $\boldsymbol{r}_k$ из вектора $\boldsymbol{r}$, то присваиваем $s'_k \leftarrow -1$ и $s''_k \leftarrow 1$;
3. если хотя бы один из объектов семейства (W', s', t') изменился, перевычисляем остальные два, используя соотношение (20); эту процедуру повторяем, пока изменения не прекратятся; аналогично поступаем с семейством объектов (W'', s'', t'').

В Табл. 3 представлена полная алгоритмическая схема вычисления контрольных матриц W', W'', и векторов s', t', s'', t'' в случае дробления ведущей системы $\boldsymbol{Q}x = \boldsymbol{r}$ на две системы-потомка $\boldsymbol{Q}'x = \boldsymbol{r}'$ и $\boldsymbol{Q}''x = \boldsymbol{r}''$.

Если ведущая система порождает одного потомка $\boldsymbol{Q}'x = \boldsymbol{r}'$, то контрольные матрица и векторы остаются неизменными, поэтому можно присвоить $W' \leftarrow W$, $s' \leftarrow s$, $t' \leftarrow t$.

Как было отмечено выше, значения любого из объектов семейства (W', s', t') можно перевычислить вследствие изменения двух других объектов. Для этого используется соотношение (20) и множества K', Λ', Ω', индексов элементов вектора s', вектора t' и матрицы W' соответственно, которые изменяли свои значения (с 0 на ± 1) на текущем шаге процедуры перевычисления. Очевидно,

$$\mathrm{K}' \subset \{1, 2, \ldots, n\},\ \Omega' \subset \{1, 2, \ldots, n\},$$

$$\Omega' \subset \{(i, j) \mid i, j \in \{1, 2, \ldots, n\}\}.$$

Процедуры перевычисления вектора s' и контрольной матрицы W' приведены в Табл. 4 и 5 соответственно. Процесс перевычисления вектора t' выполняется по схеме алгоритма Табл. 4 с единственным отличием: цикл «`DO FOR` $l \in \Lambda'$» во втором `IF`-операторе заменяется на «`DO FOR` $k \in \mathrm{K}'$».

Отметим одно замечательное свойство контрольной матрицы W, вытекающее из соотношения (20). В каждой её 2×2-подматрице любой элемент равен произведению остальных трех элементов

$$w_{i_1 j_1} w_{i_1 j_2} w_{i_2 j_2} = w_{i_2 j_1}, \tag{22}$$

$$w_{i_1 j_2} w_{i_2 j_1} w_{i_2 j_2} = w_{i_1 j_1}, \tag{23}$$

$$w_{i_1 j_1} w_{i_2 j_1} w_{i_2 j_2} = w_{i_1 j_2}. \tag{24}$$

Данные равенства могут быть полезными для уточнения контрольной матрицы W. Например, пусть мы намереваемся рассечь ведущую интервальную

Таблица 3. Перевычисление контрольных матриц и векторов.

$W'Change \leftarrow$`false`; $s'Change \leftarrow$ `false`; $t'Change \leftarrow$`false`;
$W''Change \leftarrow$ `false`; $s''Change \leftarrow$ `false`; $t''Change \leftarrow$`false`;
`IF` ((рассекается $\boldsymbol{q}_{kl}$ из Q) `AND` ($\boldsymbol{q}_{kl}$ рассекается на два конца))`THEN`
    $s'_k \leftarrow -1$; $s''_k \leftarrow 1$; $s'Change \leftarrow$`true`; $s''Change \leftarrow$ `true`;
`END IF`
`IF` ((рассекается $\boldsymbol{r}_k$ из r) `AND` ($\boldsymbol{r}_k$ рассекается на два конца))`THEN`
    $w'_{kl} \leftarrow 1$; $w''_{kl} \leftarrow -1$; $W'Change \leftarrow$`true`; $W''Change \leftarrow$ `true`;
`END IF`
`DO WHILE` ($W'Change$ `OR`$s'Change$ `OR` $t'Change$)
    `IF` ($s'Change$ `OR` $t'Change$) `THEN`
        Пытаемся перевычислить матрицу W' согласно (20);
        `IF` (W' изменилась) `THEN` $W'C \leftarrow$ `true` `ELSE` $W'C \leftarrow$ `false` `END IF`
    `END IF`
    `IF` ($W'Change$ `OR` $t'Change$) `THEN`
        Пытаемся перевычислить вектор s' согласно (20);
        `IF` (s' изменился) `THEN` $s'C \leftarrow$ `true` `ELSE` $s'C \leftarrow$ `false` `END IF`
    `END IF`
    `IF` ($W'Change$ `OR` $s'Change$) `THEN`
        Пытаемся перевычислить вектор t' согласно (20);
        `IF` (t' изменился) `THEN` $t'C \leftarrow$ `true` `ELSE` $t'C \leftarrow$ `false` `END IF`
    `END IF`
    $W'Change \leftarrow W'C$; $s'Change \leftarrow s'C$; $t'Change \leftarrow t'C$;
`END DO`
`DO WHILE` ($W''Change$ `OR`$s''Change$ `OR` $t''Change$)
    `IF` ($s''Change$ `OR` $t''Change$) `THEN`
        Пытаемся перевычислить матрицу W'' согласно (20);
        `IF` (W'' изменилась) `THEN` $W''C \leftarrow$ `true` `ELSE` $W''C \leftarrow$ `false` `END IF`
    `END IF`
    `IF` ($W''Change$ `OR` $t''Change$) `THEN`
        Пытаемся перевычислить вектор s'' согласно (20);
        `IF` (s'' изменился) `THEN` $s''C \leftarrow$ `true` `ELSE` $s''C \leftarrow$ `false` `END IF`
    `END IF`
    `IF` ($W''Change$ `OR` $s''Change$) `THEN`
        Пытаемся перевычислить вектор t'' согласно (20);
        `IF` (t'' изменился) `THEN` $t''C \leftarrow$ `true` `ELSE` $t''C \leftarrow$ `false` `END IF`
    `END IF`
    $W''Change \leftarrow W''C$; $s''Change \leftarrow s''C$; $t''Change \leftarrow t'C$;
`END DO`

Таблица 4. Перевычисление s'.

```
IF (W'Change) THEN
    DO FOR (k, l) ∈ Ω'
        IF (t'_l ≠ 0) s'_k ← w'_kl/t'_l
    END DO
END IF
IF (t'Change) THEN
    DO FOR (l) ∈ Λ'
        DO FOR k = 1 TO n
            IF (s'_k = 0 AND w'_kl ≠ 0) s'_k ← w'_kl/t'_l
        END DO
    END DO
END IF
```

Таблица 5. Перевычисление W'.

```
IF (s'Change) THEN
      DO FOR k ∈ K'
            DO FOR l = 1 TO n
                  IF (t'_l ≠ 0) w'_kl ← s'_k t'_l
            END DO
      END DO
IF (t'Change) THEN
      DO FOR l ∈ Λ'
            DO FOR k = 1 TO n
                  IF (s'_k ≠ 0) w'_kl ← s'_k/t'_l
            END DO
      END DO
END IF
```

систему $\boldsymbol{Q}x = \boldsymbol{r}$ по элементу $\boldsymbol{q}_{kl}$, в то время как соответствующий элемент w_{kl} матрицы W равен нулю. При этом согласно обычному правилу дробления мы должны были бы породить две системы-потомка. Но разумно сделать попытку доопределить w_{kl}, поискав 2×2-подматрицы в W, имеющие ненулевыми все элементы за исключением w_{kl}. Если такая подматрица в W найдется, то мы присваиваем элементу w_{kl} значение произведения остальных трех элементов. Алгоритм уточнения контрольной матрицы на основе соотношений (22)–(24) представлен в Табл. 6.

Возникающие в процессе дробления системы-потомки организуем в список $\mathcal{L}$, состоящий из записей вида

$$(\boldsymbol{Q}, \boldsymbol{r}, \Upsilon(\boldsymbol{Q}, \boldsymbol{r}), W, s, t, \boldsymbol{Y}, \boldsymbol{x}), \tag{25}$$

где

$\boldsymbol{Q}$ — интервальная $n \times n$-матрица, $\boldsymbol{Q} \subseteq \boldsymbol{A}$,
$\boldsymbol{r}$ — интервальный n-вектор, $\boldsymbol{r} \subseteq \boldsymbol{b}$,
$\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$ — нижний конец ν-ой компоненты внешней интервальной оценки множества решений $\Xi(\boldsymbol{Q}, \boldsymbol{r})$,
W, s, t — контрольные матрица и векторы,
$\boldsymbol{Y}$ — обратная интервальная матрица, такая что $\boldsymbol{Y} \supseteq \{Q^{-1} \mid Q \in \boldsymbol{Q}\}$,
$\boldsymbol{x}$ — интервальный n-вектор, такой что $\boldsymbol{x} \supseteq \Xi(\boldsymbol{Q}, \boldsymbol{r})$.

Как и в обычном методе дробления параметров записи, для которых справедливо неравенство

$$\Upsilon(\boldsymbol{Q}, \boldsymbol{r}) > \omega,$$

либо вообще не заносятся в список, либо удаляются из него во время чистки (см. §2.3).

На каждом шаге алгоритма проверяем монотонную зависимость решения системы от её коэффициентов в соответствии с процедурой, описанной в §2.2. На основе соотношений (19), (21) изменяем с 0 на ± 1 значения элементов контрольных матрицы W и вектора s, соответствующие тем элементам в $\boldsymbol{Q}$ и $\boldsymbol{r}$, на которых была выявлена монотонность.

Для дробления ведущей ИСЛАУ используем стратегию, рассмотренную §2.1. Ведущие брусы параметров рассекаем по элементам, на которых достигается максимум величин (7), т.е. по элементам, которым соответствует наибольшее произведение модуля интервальной оценки производной решения на ширину соответствующего интервала.

Таблица 6. Уточнение W.

```
DO FOR i = 1 TO n
    DO FOR j = 1 TO n
        IF (i ≠ k AND j ≠ l) THEN
            IF (w_ij ≠ 0 AND w_il ≠ 0 AND w_kj ≠ 0) THEN
                w_kl ← w_ij w_il w_kj;
                EXIT
            END IF
        END IF
    END DO
END DO
```

Критерием остановки алгоритма служит достижение точности оценки $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$ или требуемой малости величины $(\omega - \Upsilon(\boldsymbol{Q}, \boldsymbol{r})$ (см. §2.3).

Итоговая схема алгоритма метода дробления параметров с модификацией Рона приведена в Табл. 7. Для начала его работы необходимо

- найти предварительные внешние оценки для множества решений исходной системы и «обратной интервальной матрицы», т. е. вычислить $\boldsymbol{x} = \mathit{Encl}(\boldsymbol{A}, \boldsymbol{b}) \supseteq \Xi(\boldsymbol{A}, \boldsymbol{b})$ и $\boldsymbol{Y} \supseteq \boldsymbol{A}^{-1}$,
- назначить точность оценки $\varepsilon > 0$,
- присвоить $\boldsymbol{Q} \leftarrow \boldsymbol{A}$ и $\boldsymbol{r} \leftarrow \boldsymbol{b}$,
- положить $\Upsilon(\boldsymbol{Q}, \boldsymbol{r}) \leftarrow \underline{\boldsymbol{x}}_{\nu}$ и $\omega \leftarrow +\infty$,
- положить $W \leftarrow 0$, $s \leftarrow 0$, $t \leftarrow 0$,
- инициализировать рабочий список $\mathcal{L}$ записью $(\boldsymbol{Q}, \boldsymbol{r}, \Upsilon(\boldsymbol{Q}, \boldsymbol{r}), W, s, t, \boldsymbol{Y}, \boldsymbol{x})$.

Таблица 7. Алгоритм оптимального внешнего оценивания множества решений ИСЛАУ методом дробления параметров с модификацией Рона.

DO WHILE ((ведущая оценка $\Upsilon(\boldsymbol{Q},\boldsymbol{r})$ неточна) и $(\omega - \Upsilon(\boldsymbol{Q},\boldsymbol{r}) > \varepsilon)$)

по формулам (10) вычисляем интервальные расширения производных $\frac{\partial x_\nu(\boldsymbol{Q},\boldsymbol{r})}{\partial q_{ij}}$ и $\frac{\partial x_\nu(\boldsymbol{Q},\boldsymbol{r})}{\partial r_i}$ по элементам $\boldsymbol{q}_{ij}$ и $\boldsymbol{r}_i$ с ненулевой шириной;

сжимаем в соответствии с (8)-(9) в $\boldsymbol{Q}$ и $\boldsymbol{r}$ элементы, на которых выявлена монотонная зависимость x_ν от q_{ij} и r_i ;

соответствующие элементы в контрольных матрице W и векторе s изменяем с 0 на ± 1 на основе (19), (21);

ищем в ведущей ИСЛАУ $\boldsymbol{Q}x = \boldsymbol{r}$ интервальный элемент $\boldsymbol{h}$, которому соответствует наибольшее из произведений (7);

пытаемся уточнить матрицу W процедурой из Табл. 6;

порождаем одну или две ИСЛАУ-потомка $\boldsymbol{Q'}x = \boldsymbol{r'}$ и $\boldsymbol{Q''}x = \boldsymbol{r''}$ процедурой из Табл. 2;

если порождены две системы-потомка, перевычисляем матрицы W', W'' и векторы s', s'', t', t'' процедурами из Табл. 3-5, иначе присваиваем $W' \leftarrow W$, $s' \leftarrow s$, $t' \leftarrow t$;

вычисляем вектор $\boldsymbol{x'} \leftarrow Encl(\boldsymbol{Q'},\boldsymbol{r'})$ и, возможно, $\boldsymbol{x''} \leftarrow Encl(\boldsymbol{Q''},\boldsymbol{r''})$;

присваиваем оценку $v' \leftarrow \Upsilon(\boldsymbol{Q'},\boldsymbol{r'})$ и, возможно, $v'' \leftarrow \Upsilon(\boldsymbol{Q''},\boldsymbol{r''})$;

вычисляем оценку обратной интервальной матрицы $\boldsymbol{Y'} \supseteq (\boldsymbol{Q'})^{-1}$ и, возможно, $\boldsymbol{Y''} \supseteq (\boldsymbol{Q''})^{-1}$;

вычисляем оценку $\Upsilon(\mathrm{mid}\,\boldsymbol{Q'}, \mathrm{mid}\,\boldsymbol{r'})$ и, возможно, $\Upsilon(\mathrm{mid}\,\boldsymbol{Q''}, \mathrm{mid}\,\boldsymbol{r''})$ и присваиваем $\mu \leftarrow \min\{\,\Upsilon(\mathrm{mid}\,\boldsymbol{Q'}, \mathrm{mid}\,\boldsymbol{r'}), \Upsilon(\mathrm{mid}\,\boldsymbol{Q''}, \mathrm{mid}\,\boldsymbol{r''})\}$;

удаляем из списка $\mathcal{L}$ бывшую ведущую запись $(\boldsymbol{Q},\boldsymbol{r},v,W,s,t,\boldsymbol{Y},\boldsymbol{x})$;

если $v' \le \omega$, то помещаем запись $(\boldsymbol{Q'},\boldsymbol{r'},v',W',s',t',\boldsymbol{Y'},\boldsymbol{x'})$ в список $\mathcal{L}$ в порядке возрастания значений третьего поля;

если ведущая ИСЛАУ породила две системы-потомка и $v'' \le \omega$, то помещаем запись $(\boldsymbol{Q''},\boldsymbol{r''},v'',W'',s'',t'',\boldsymbol{Y''},\boldsymbol{x''})$ в список $\mathcal{L}$ в порядке возрастания значений третьего поля;

если $\omega > \mu$, то полагаем $\omega \leftarrow \mu$ и чистим список $\mathcal{L}$: удаляем из него все такие записи $(\boldsymbol{Q},\boldsymbol{r},v,W,s,t,\boldsymbol{Y},\boldsymbol{x})$, что $v > \omega$;

END DO

4 Базовый алгоритм внешнего оценивания множества решений

Пусть задана интервальная линейная система уравнений $\boldsymbol{A}x = \boldsymbol{b}$ с неособенной интервальной матрицей $\boldsymbol{A} \in \mathbb{IR}^{n\times n}$ и интервальным вектором правой части $\boldsymbol{b} \in \mathbb{IR}^n$. Рассмотрим ряд методов внешнего интервального оценивания объединённого множества решений $\Xi(\boldsymbol{A}, \boldsymbol{b})$, которые могут быть использованы в качестве базовых при поиске оптимального решения ИСЛАУ методом дробления параметров.

4.1 Интервальный метод Гаусса

Интервальный метод Гаусса является интервальным аналогом хорошо известного в линейной алгебре метода исключения Гаусса, состоящего в преобразовании матрицы системы к верхнему треугольному виду (прямой ход) и последовательном вычислении значений неизвестных (обратный ход). Алгоритм интервального метода такой же, как и в вещественном случае, с той лишь разницей, что он оперирует интервальными величинами с помощью операций интервальной арифметики (описание метода см. [15, 4]). Из свойства монотонности интервальной арифметики $\mathbb{IR}$ по включению следует, что результат выполнения интервального метода Гаусса содержит объединённое множество решений ИСЛАУ.

Нетривиальным моментом реализации интервального метода Гаусса является деление на диагональные элементы: оно оказывается невозможным, если эти интервальные элементы содержат нуль. При выполнении прямого хода алгоритма можно использовать перестановку строк матрицы системы для выбора диагонального элемента с наибольшей мигнитудой. Именно, сначала среди элементов первого столбца матрицы выбираем элемент с наибольшей мигнитудой, перемещаем его, если это необходимо, на крайнее верхнее положение перестановкой строк и вычитаем получившуюся после перестановки первую строку из остальных строк, домножив её на величину, равную отношению первого элемента каждой из этих строк к первому элементу первой строки. После этого первую строку и первый столбец мысленно вычёркиваем и совершаем указанные выше преобразования над оставшейся матрицей.

Работоспособность интервального метода Гаусса зависит от свойств интервальной матрицы системы. Если матрица $\boldsymbol{A}$ является H-матрицей, то интервальный метод Гаусса применим к интервальной линейной системе $\boldsymbol{A}x = \boldsymbol{b}$ для любого $\boldsymbol{b} \in \mathbb{IR}^n$ (теорема Алефельда [4]). В случае, когда матрица $\boldsymbol{A}$ есть

M-матрица и справедливо хотя бы одно из условий $\boldsymbol{b} < 0$, $0 \in \boldsymbol{b}$, $\boldsymbol{b} > 0$, то результатом работы интервального метода Гаусса будет оптимальная внешняя оценка множества решений.

Для придания интервальной матрице системы требуемых свойств можно применить так называемое *предобуславливание* — одновременное домножение матрицы $\boldsymbol{A}$ и вектора правой части $\boldsymbol{b}$ слева на некоторую точечную матрицу, например, обратную среднюю матрицу $(\operatorname{mid} \boldsymbol{A})^{-1}$. Таким образом, вместо исходной системы получаем *предобусловленную интервальную систему*

$$\left((\operatorname{mid} \boldsymbol{A})^{-1}\boldsymbol{A}\right) x = (\operatorname{mid} \boldsymbol{A})^{-1}\boldsymbol{b},$$

матрица которой является H-матрицей, если в исходной ИСЛАУ интервальная матрица $\boldsymbol{A}$ сильно неособена. В этом случае предобуславливание обратной средней матрицей гарантирует работу интервального метода Гаусса [16].

Отметим, что качество внешних оценок объединённого множества решений ИСЛАУ, получаемых интервальным методом Гаусса, не очень высокое и ухудшается с возрастанием размерности матрицы системы и увеличением радиусов её интервальных элементов.

4.2 Интервальный метод Гаусса-Зейделя

Интервальный метод Гаусса-Зейделя является итерационной процедурой для уточнения внешней оценки множества решений. Пусть $\boldsymbol{x}^{(k)} \in \mathbb{IR}^n$ — некоторое приближение множества решений ИСЛАУ $\boldsymbol{A}x = \boldsymbol{b}$ с матрицей $\boldsymbol{A} = (\boldsymbol{a}_{ij})$, в которой элементы главной диагонали не содержат нуля, т. е. $0 \notin \boldsymbol{a}_{ii}$ для $i = 1, 2, \ldots, n$. Уточненную оценку $\tilde{\boldsymbol{x}}$ находим следующим образом

$$\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i^{(k)} \cap \left(\boldsymbol{b}_i - \sum_{j=1}^{i-1} \boldsymbol{a}_{ij}\tilde{\boldsymbol{x}}_j - \sum_{j=i+1}^{n} \boldsymbol{a}_{ij}\boldsymbol{x}_j^{(k)} \right) / \boldsymbol{a}_{ii} \ , \quad i = 1, 2, \ldots, n.$$

Если расстояние между векторами $\boldsymbol{x}^{(k)}$ и $\tilde{\boldsymbol{x}}$ больше заданной малой величины $\varepsilon > 0$, полагаем

$$\boldsymbol{x}^{(k+1)} \leftarrow \tilde{\boldsymbol{x}}, \quad k = 0, 1, 2, \ldots.$$

В этом итерационном процессе компоненты нового внешнего приближения множества решений находим последовательно друг за другом посредством пересечения старой и новой оценок, причём уточнённые новые компоненты сразу же привлекаются для расчёта следующих компонент нового приближения.

Интервальный метод Гаусса-Зейделя применяют обычно после предобуславливания системы. Сходимость метода зависит от свойств матрицы системы. Если матрица $\boldsymbol{A}$ системы является H-матрицей, то любой достаточно широкий начальный интервальный вектор $\boldsymbol{x} \in \mathbb{IR}^n$ улучшается методом Гаусса-Зейделя (теорема Ноймайера). Если $\boldsymbol{A}$ является M-матрицей, то метод Гаусса-Зейделя сходится к оптимальной интервальной оценке множества решений $\Xi(\boldsymbol{A}, \boldsymbol{b})$, будучи начатым из любого начального приближения (теорема Барта-Нудинга).

4.3 Метод Кравчика и его модификация

Пусть интервальная линейная система $\boldsymbol{A}x = \boldsymbol{b}$ предобусловлена обратной средней матрицей $C = (\operatorname{mid} \boldsymbol{A})^{-1}$. Предположим, что интервальный вектор $\boldsymbol{x}^{(k)}$ ограничивает множество решений ИСЛАУ, т. е. $\boldsymbol{x}^{(k)} \supseteq \Xi(\boldsymbol{A}, \boldsymbol{b})$. Тогда для любой точечной матрицы $\tilde{A} \in \boldsymbol{A}$ и точечного вектора $\tilde{b} \in \boldsymbol{b}$ справедливо

$$\tilde{A}^{-1}\tilde{b} = C\tilde{b} + (I - C\tilde{A})\tilde{A}^{-1}\tilde{b} \in C\boldsymbol{b} + (I - C\boldsymbol{A})\boldsymbol{x}^{(k)},$$

и можно заключить, что

$$\text{из } \Xi(\boldsymbol{A}, \boldsymbol{b}) \subseteq \boldsymbol{x}^{(k)} \text{ следует } \Xi(\boldsymbol{A}, \boldsymbol{b}) \subseteq (C\boldsymbol{b} + (I - C\boldsymbol{A})\boldsymbol{x}^{(k)}) \cap \boldsymbol{x}^{(k)}. \qquad (26)$$

Отсюда получим итерационный метод Кравчика [10]

$$\boldsymbol{x}^{(k+1)} \leftarrow (C\boldsymbol{b} + (I - C\boldsymbol{A})\boldsymbol{x}^{(k)}) \cap \boldsymbol{x}^{(k)}, \;\; k = 0, 1, 2, \ldots, \qquad (27)$$

где в качестве начального приближения $\boldsymbol{x}^{(0)}$ можно взять брус

$$\boldsymbol{x}^{(0)} = ([-\alpha, \alpha], \ldots, [-\alpha, \alpha])^{\top},$$

$$\text{с } \alpha = \frac{\|C\boldsymbol{b}\|_\infty}{1 - \beta} \text{ и } \beta = \|I - C\boldsymbol{A}\|_\infty < 1.$$

Итерационный процесс (27) будет остановлен в том случае, когда растояние между векторами $\boldsymbol{x}^{(k+1)}$ и $\boldsymbol{x}^{(k)}$, полученными на текущем и предыдущем шагах алгоритма, будет незначительно отличаться от нуля.

Рассмотрим теперь модификацию метода Кравчика, которая реализована в пакете INTLAB — популярном свободно распространяемом интервальном расширении MATLAB'а [9].

Умножим обратную среднюю матрицу $C = (\operatorname{mid} \boldsymbol{A})^{-1}$ на средний вектор $\operatorname{mid} \boldsymbol{b}$ правой части ИСЛАУ и вычислим точечное решение $x_s = C \cdot \operatorname{mid} \boldsymbol{b}$.

Применив (26) к внешней оценке $\boldsymbol{d}^{(k)}$ множества решений $\Xi(\boldsymbol{A}, \boldsymbol{b} - \boldsymbol{A}x_s)$, получим модифицированный метод Кравчика

$$\boldsymbol{d}^{(k+1)} \leftarrow (C(\boldsymbol{b} - \boldsymbol{A}x_s) + (I - C\boldsymbol{A})\boldsymbol{d}^{(k)}) \cap \boldsymbol{d}^{(k)}, \quad k = 0, 1, 2, \ldots \tag{28}$$

Итерационный процесс (28) осуществляется с помощью так называемого «эпсилон-раздутия» (ε-раздутия) [13]. Оно состоит в том, что некоторый интервал $\boldsymbol{d}$, полученный на текущем шаге алгоритма, заменяется на объемлющий интервал $\boldsymbol{d}_\varepsilon$, отличающийся от него на некоторый малый параметр ε. Это можно сделать, например, следующим образом

$$\boldsymbol{d}_\varepsilon = \boldsymbol{d} + [-\varepsilon, \varepsilon] \operatorname{rad} \boldsymbol{d} + [-\eta, \eta]\, e,$$

где ε, η — некоторые малые положительные вещественные числа, $e = (1, 1, \ldots, 1)^\top$.

Значения ε и η выбираются, исходя из эвристических соображений. На основе своего вычислительного опыта Румп в [22] предложил в качестве η брать наименьшее положительное вещественное число, которое может быть представлено в ЭВМ, и $\varepsilon = 0.1$. Тогда итерационная процедура (28) примет вид:

$$\boldsymbol{d}^{(k+1)} \leftarrow C(\boldsymbol{b} - \boldsymbol{A}x_s) + (I - C\boldsymbol{A})(\boldsymbol{d}^{(k)} + 0.1[-1, 1] \operatorname{rad} \boldsymbol{d} + [-\eta, \eta]\, e),$$

$$k = 0, 1, 2, \ldots.$$

Критерием остановки данной процедуры служит условие: $\boldsymbol{d}^{(k+1)} \subseteq \boldsymbol{d}^{(k)}$.

Поскольку

$$\begin{aligned} x_s + C(\boldsymbol{b} - \boldsymbol{A}x_s) &= x_s + C\boldsymbol{b} - C(\boldsymbol{A}x_s) \\ &= C\boldsymbol{b} - (C\boldsymbol{A})x_s + x_s \\ &= C\boldsymbol{b} + (I - C\boldsymbol{A})x_s, \end{aligned}$$

и

$$\begin{aligned} x_s + C(\boldsymbol{b} - \boldsymbol{A}x_s) + (I - C\boldsymbol{A})\boldsymbol{d}^{(k)} &= C\boldsymbol{b} + (I - C\boldsymbol{A})x_s + (I - C\boldsymbol{A})\boldsymbol{d}^{(k)} \\ &\supseteq C\boldsymbol{b} + (I - C\boldsymbol{A})(x_s + \boldsymbol{d}^{(k)}) \\ &= C\boldsymbol{b} + (I - C\boldsymbol{A})\boldsymbol{y}^{(k)}, \end{aligned}$$

где $\boldsymbol{y}^{(k)} = x_s + \boldsymbol{d}^{(k)}$, то окончательно имеем

$$\boldsymbol{y}^{(k+1)} = x_s + \boldsymbol{d}^{(k+1)} \supseteq (C\boldsymbol{b} + (I - C\boldsymbol{A})\boldsymbol{y}^{(k)}) \cap \boldsymbol{y}^{(k)}, \qquad k = 0, 1, 2, \ldots$$

4.4 Процедура Хансена-Блика-Рона

Опишем процедуру Хансена-Блика-Рона так, как она изложена А. Ноймайером в [14]. Процедура основана на следующем результате.

Теорема. *Пусть в интервальной линейной системе* $\boldsymbol{A}x = \boldsymbol{b}$ *матрица* $\boldsymbol{A} = (\boldsymbol{a}_{ij}) \in \mathbb{IR}^{n\times n}$ *является интервальной* H*-матрицей и пусть*

$$u_i = (\langle \boldsymbol{A}\rangle^{-1}|\boldsymbol{b}|)_i, \qquad d_i = (\langle \boldsymbol{A}\rangle^{-1})_{ii}, \tag{29}$$

$$\alpha_i = \langle \boldsymbol{a}_{ii}\rangle - 1/d_i, \qquad \beta_i = u_i/d_i - |\boldsymbol{b}_i|, \tag{30}$$

где $\langle \boldsymbol{A}\rangle$ *— компарант матрицы* $\boldsymbol{A}$, $i = 1, 2, \ldots, n$. *Тогда множество решений* $\Xi(\boldsymbol{A}, \boldsymbol{b})$ *содержится в интервальном векторе* $\boldsymbol{x} = (\boldsymbol{x}_i)$ *с компонентами*

$$\boldsymbol{x}_i = \frac{\boldsymbol{b}_i + \beta_i[-1, 1]}{\boldsymbol{a}_{ii} + \alpha_i[-1, 1]}, \qquad i = 1, 2, \ldots, n. \tag{31}$$

Заметим, что необходимо знать строгую оценку сверху для $\langle \boldsymbol{A}\rangle^{-1}$, чтобы далее найти оценки сверху для α_i и β_i. Известно, что если $\boldsymbol{A}$ является H-матрицей, то $\langle \boldsymbol{A}\rangle^{-1}$ — неотрицательная матрица. Следовательно, верхняя оценка B матрицы $\langle \boldsymbol{A}\rangle^{-1}$ может быть выражена через некоторую оценку $\tilde{B}$ матрицы $\langle \boldsymbol{A}\rangle^{-1}$ и векторы $v, w \in \mathbb{R}^n$, удовлетворяющие неравенству $I - \langle \boldsymbol{A}\rangle\tilde{B} \le \langle \boldsymbol{A}\rangle vw^\top$, следующим образом

$$B = \tilde{B} + vw^\top. \tag{32}$$

Из определения H-матрицы следует, что существует вектор $v > 0$ такой, что $u = \langle \boldsymbol{A}\rangle v > 0$. Чтобы этот вектор v удовлетворял (32), необходимо взять вектор w с компонентами вида

$$w_k = \max_i \frac{-R_{ik}}{u_i},$$

где $R = \langle \boldsymbol{A}\rangle\tilde{B} - I$.

Теперь остается найти вектор v. Предположим, что существует $\tilde{u}$, такой что $v = \tilde{B}\tilde{u} \approx \langle \boldsymbol{A}\rangle^{-1}\tilde{u} > 0$, тогда $\boldsymbol{A}$ есть H-матрица и, если $u = \langle \boldsymbol{A}\rangle v \approx \tilde{u}$ — положительный вектор, то оценка $\tilde{B}$ достаточно точна. Поскольку $\langle \boldsymbol{A}\rangle^{-1}$ неотрицательная матрица, то можно в качестве $\tilde{u}$ взять единичный вектор $(1, \ldots, 1)$, чтобы обеспечить выполнение условия $\langle \boldsymbol{A}\rangle^{-1}\tilde{u} > 0$.

Вычислительный опыт свидетельствует, что процедура Хансена-Блика-Рона всегда дает результаты заведомо не хуже, чем метод Кравчика.

5 Способы обработки списка

В представленных выше алгоритмах дробления параметров (Табл. 1, Табл. 7) рабочий список $\mathcal{L}$ упорядочивается по возрастанию оценки $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$, так что первая запись списка является одновременно и ведущей. Можно использовать другой способ обработки списка — вообще его не структурировать, т. е. реализовать в виде кучи [1].

Достоинство первого способа — это простота обращения к ведущей записи. Однако она достигается ценой определенных затрат на поддержание упорядоченности списка, что приводит к необходимости его частичного просмотра. Во втором способе достаточно просто добавлять новые записи в список. Но чтобы найти в нём ведущую запись, нужно просмотреть весь список.

Заметим, что чистка списка от бесперспективных записей осуществляется проще, если список упорядочен. Кроме того, просматривать весь неупорядоченный список $\mathcal{L}$ приходится на всех без исключения шагах алгоритма вне зависимости от хода его выполнения. Если же список $\mathcal{L}$ упорядочен, то при занесении в него записей-потомков в худшем случае его приходится просматривать целиком, в лучшем он вообще не нуждается в просмотре, а в среднем на каждом шаге мы должны будем просматривать список $\mathcal{L}$ всё-таки не весь, т. е. в меньшей мере, чем для неупорядоченного варианта.

Ускорение обработки списка $\mathcal{L}$ может быть достигнуто с помощью следующего приема, предложенного П.С. Панковым [3]. В его основе — задание некоторой «пороговой константы» γ, такой что

$$\Upsilon(\boldsymbol{Q}, \boldsymbol{r}) < \gamma < \omega,$$

и «подсписка активных записей»

$$\mathcal{L}_\gamma = \{\, (\boldsymbol{Q}, \boldsymbol{r}, \Upsilon(\boldsymbol{Q}, \boldsymbol{r}), W, s, t, \boldsymbol{Y}, \boldsymbol{x}) \mid \Upsilon(\boldsymbol{Q}, \boldsymbol{r}) < \gamma \} \subseteq \mathcal{L},$$

где ω определяется соотношением (11).

В случае неупорядоченного списка $\mathcal{L}$ (т.е. кучи записей) ясно, что именно в подсписке $\mathcal{L}_\gamma$ (при $\mathcal{L}_\gamma \neq \varnothing$) находится ведущая запись всего списка $\mathcal{L}$, и потому при ее поиске нам достаточно, сэкономив машинное время, ограничиться лишь просмотром $\mathcal{L}_\gamma$. Если же мы придерживаемся варианта упорядоченного рабочего списка $\mathcal{L}$, то по аналогичным причинам эту упорядоченность достаточно поддерживать только в $\mathcal{L}_\gamma$, организовав дополнение $\mathcal{L} \backslash \mathcal{L}_\gamma$ в виде кучи. Если в процессе работы алгоритма подмножество $\mathcal{L}_\gamma$ сделается пустым, то пороговая константа γ перевычисляется, и из $\mathcal{L}$ снова выделяется упорядоченный подсписок $\mathcal{L}_\gamma$.

Совершенно строгих рецептов по выбору величины γ дать, по-видимому невозможно. С одной стороны, с уменьшением γ уменьшается и $\mathcal{L}_\gamma$, и тем бóльшим должен быть наш выигрыш в трудоёмкости на каждом отдельном шаге алгоритма. С другой стороны, если γ слишком мало, то подсписок $\mathcal{L}_\gamma$ быстро исчерпывается, и мы вынуждены часто перевычислять γ и перестраивать $\mathcal{L}$. Руководствуясь отчасти эмпирическими, а отчасти эвристическими соображениями, можно, к примеру, взять $\gamma = \frac{1}{3}(\omega + 2\Upsilon(\boldsymbol{Q}, \boldsymbol{r}))$.

Заметим, что прием Панкова позволяет производить чистку не всего списка $\mathcal{L}$, а только его части, при этом достигается определённый выигрыш в быстродействии, но тратится дополнительная оперативная память.

Очевидно, что количество активных записей в подсписке $\mathcal{L}_\gamma$ изменяется в процессе работы алгоритма и на некоторых его шагах может быть достаточно большим, что увеличивает трудоёмкость его обработки.

Поэтому мы предлагаем ещё один возможный способ — структурировать список $\mathcal{L}$ в виде подсписка $\mathcal{L}_l$ активных записей, имеющего некоторую фиксированную максимальную длину l, и остальной части $\bar{\mathcal{L}} = \mathcal{L} \backslash \mathcal{L}_l$. Первую запись подсписка $\mathcal{L}_l$ считаем ведущей.

Если на текущем шаге алгоритма длина подсписка активных записей меньше максимальной длины l, то записи-потомки помещаем в подсписок $\mathcal{L}_l$ в порядке возрастания оценки $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$.

Если длина $\mathcal{L}_l$ равна l, т.е. подсписок активных записей полностью заполнен, то новая запись в зависимости от содержащейся в ней оценки $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$ может быть внесена либо в $\mathcal{L}_l$, либо в подсписок $\bar{\mathcal{L}}$, организованный в виде кучи. Обозначим через $\Upsilon_{\max}$ максимальную оценку $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$, содержащуюся в записях подсписка $\mathcal{L}_l$. В случае, когда длина $\mathcal{L}_l$ равна l, новая запись с оценкой $\Upsilon(\boldsymbol{Q}, \boldsymbol{r}) < \Upsilon_{\max}$ вносится в подсписок активных записей. При этом происходит его переполнение и лишняя запись помещается в кучу $\bar{\mathcal{L}}$. Записи-потомки с оценками $\Upsilon(\boldsymbol{Q}, \boldsymbol{r}) \geq \Upsilon_{\max}$ при полностью заполненном подсписке $\mathcal{L}_l$ вносятся в $\bar{\mathcal{L}}$.

Если подсписок $\mathcal{L}_l$ активных записей становится пустым, отбираем из $\bar{\mathcal{L}}$ не более l записей с наименьшими оценками $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$ и помещаем их упорядоченно в $\mathcal{L}_l$.

При изменении параметра ω производим чистку бесперспективных записей только из подсписка $\bar{\mathcal{L}}$. Чистка подсписка $\mathcal{L}_l$ активных записей выполняется только в случае, когда $\bar{\mathcal{L}} = \varnothing$.

Заметим, что длина l подсписка активных записей может быть подобрана эмпирически так, чтобы увеличить быстродействие алгоритма.

6 Реализация алгоритмов и численные эксперименты

Представленные в предыдущих параграфах алгоритмы были реализованы в программной системе Matlab, включающей интеравальное расширение INTLAB.

Помимо основной схемы алгоритма внешнего оценивания множества решений ИСЛАУ методом дробления параметров с модификацией Рона, приведенного в Табл. 7, были реализованы его модификации, различающиеся базовым алгоритмом *Encl*, а также способом организации и обработки списка.

Базовый алгоритм *Encl* был реализован на основе

- метода Кравчика,
- модифицированного метода Кравчика с использованием «эпсилон-раздутия»,
- интервального метода Гаусса,
- интервального метода Гаусса-Зейделя,
- процедуры Хансена-Блика-Рона.

В качестве алгоритма *Encl* мы вопользовались также процедурой `verifylss` из пакета INTLAB [9]. Посредством данной процедуры $\boldsymbol{x} =$ `verifylss`$(\boldsymbol{A}, \boldsymbol{b})$ может быть получена внешняя оценка множества решений ИСЛАУ $\boldsymbol{A}x = \boldsymbol{b}$. Процедура состоит из двух этапов, первый из которых реализует модифицированный метод Кравчика, второй — процедуру Хансена-Блика-Рона. Если на первом этапе интервальная оценка $\boldsymbol{x}$ решения ИСЛАУ не будет найдена в течение семи итераций, то выполняется второй этап, т. е. решение будет получено с помощью процедуры Хансена-Блика-Рона.

Отметим, что во всех базовых алгоритмах применялось предобуславливание ИСЛАУ обратной средней матрицей. Соответствующие базовые алгоритмы использовались также для внешнего оценивания «обратной интервальной матрицы», которая используется в тесте на монотонность и при выборе рассекаемого интервального элемента.

Были реализованы следующие способы организации и обработки списка:

- упорядочение списка $\mathcal{L}$ по возрастанию оценки $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$, так что первая запись списка является одновременно и ведущей; чистка бесперспективных записей производится только при изменении параметра ω (см. Табл. 7);

- список $\mathcal{L}$ организован в виде кучи; ведущая запись, для которой оценка $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$ минимальна, определяется при просмотре всего списка; чистка бесперспективных записей осуществляется только при изменении параметра ω;

- в списке $\mathcal{L}$ выделяется упорядоченный подсписок $\mathcal{L}_l$ активных записей, имеющий некоторую фиксированную максимальную длину l; дополнение $\bar{\mathcal{L}} = \mathcal{L} \backslash \mathcal{L}_l$ организовано в виде кучи; ведущей записью является первая запись $\mathcal{L}_l$; при изменении параметра ω производим чистку подсписка $\bar{\mathcal{L}}$, чистка подсписка $\mathcal{L}_l$ выполняется только в случае, когда $\bar{\mathcal{L}} = \varnothing$;

- способ, предложенный Панковым.

Представленные алгоритмы были апробированы на следующих тестовых интервальных линейных системах уравнений.

Пример 1. Интервальная линейная система Ноймайера, предложенная в [15]:

$$\begin{pmatrix} \theta & [0,2] & \cdots & [0,2] \\ [0,2] & \theta & \cdots & [0,2] \\ \vdots & \vdots & \ddots & \vdots \\ [0,2] & [0,2] & \ldots & \theta \end{pmatrix} x = \begin{pmatrix} [-1,1] \\ [-1,1] \\ \vdots \\ [-1,1] \end{pmatrix} \tag{33}$$

где θ — неотрицательный вещественный параметр. Отметим, что матрица системы чётного порядка n неособенна при $\theta > n$, а матрица нечётного порядка n неособенна при $\theta > \sqrt{n^2 - 1}$. При приближении θ к границам неособенности размеры объединённого множества решений ИСЛАУ (33) неограниченно возрастают. Варьируя θ, можно получить набор тестов для проверки рассмотренных алгоритмов оптимального решения «внешней задачи» ИСЛАУ.

Пример 2. Интервальная линейная система уравнений, рассмотренная в [4], [24]:

$$\begin{pmatrix} [n-1,N] & [\alpha-1,1-\beta] & \cdots & [\alpha-1,1-\beta] \\ [\alpha-1,1-\beta] & [n-1,N] & \cdots & [\alpha-1,1-\beta] \\ \vdots & \vdots & \ddots & \vdots \\ [\alpha-1,1-\beta] & [\alpha-1,1-\beta] & \ldots & [n-1,N] \end{pmatrix} x = \begin{pmatrix} [1-n,n-1] \\ [1-n,n-1] \\ \vdots \\ [1-n,n-1] \end{pmatrix} \tag{34}$$

где n — размерность системы ($n \geq 2$), $0 < \alpha \leq \beta \leq 1$, N — вещественное число, не меньшее $n-1$. Варьируя значения α, β, n и N нетрудно получить широкий набор ИСЛАУ для тестирования разработанных алгоритмов. Когда

β уменьшается, приближаясь к нулю, матрица системы (34) становится всё более близкой к особенной, а множество решений неограниченно увеличивается в размерах. Изменяя отношение α и β, можно модифицировать форму множества решений.

Можно показать [4], что оптимальными покомпонентными оценками множества $\tilde{\Xi}$ решений системы (34) являются

$$\begin{aligned} \min\{\, x_i \mid x \in \tilde{\Xi} \,\} &= -1/\alpha, \\ \max\{\, x_i \mid x \in \tilde{\Xi} \,\} &= 1/\alpha, \quad i = 1, 2, \ldots, n, \end{aligned}$$

причём они не зависят от конкретного значения N.

Пример 3. Интервальная линейная система Тофта [25]

$$\boldsymbol{A}x = \boldsymbol{b}, \tag{35}$$

где матрица системы

$$\boldsymbol{A} = \begin{pmatrix} [1-r,1+r] & 0 & \cdots & 0 & [1-r,1+r] \\ 0 & [1-r,1+r] & \cdots & 0 & [2-r,2+r] \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & [1-r,1+r] & [n-1-r,n-1+r] \\ [1-r,1+r] & [2-r,2+r] & \cdots & [n-1-r,n-1+r] & [n-r,n+r] \end{pmatrix},$$

и вектор правой части

$$\boldsymbol{b} = \begin{pmatrix} [1-R,1+R] \\ [1-R,1+R] \\ \vdots \\ [1-R,1+R] \end{pmatrix}.$$

Здесь r, R — положительные вещественные числа.

Матрица данной системы является интервализацией известной тестовой матрицы вычислительной линейной алгебры из справочника [8]:

$$\begin{pmatrix} 1 & 0 & \cdots & 0 & 1 \\ 0 & 1 & \cdots & 0 & 2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & n-1 \\ 1 & 2 & \cdots & n-1 & n \end{pmatrix},$$

7 Сравнительный анализ результатов

7.1 Влияние базового алгоритма

Мы провели ряд тестовых расчётов с интервальной системой Ноймайера (Пример 1), меняя размерность системы n, значение её параметра θ, а также базовый алгоритм *Encl* внешней оценки множества решений интервальной системы.

В Табл. 8 представлены результаты численных экспериментов — время (в сек.) нахождения оптимальной нижней оценки первой компоненты множества решений ИСЛАУ.

Базовые алгоритмы обозначены в таблице следующим образом: **K** — метод Кравчика, **MK** — модифицированный метод Кравчика, **G** — интервальный метод Гаусса, **GS** — интервальный метод Гаусса-Зейделя, **HBR** — процедура Хансена-Блика-Рона, **V** — процедура `verifylss` из пакета INTLAB.

Проанализировав полученные результаты, можно сделать несколько выводов.

Несложно заметить, что наихудшую скорость мы наблюдаем при работе программы, в которой в качестве базового алгоритма реализован итерационный метод Кравчика. Несколько быстрее работает программа, если базовый алгоритм основан на модификации метода Кравчика, а также интервальных методах Гаусса и Гаусса-Зейделя. Применение этих алгоритмов как базовых, скорее всего, нежелательно.

Процедура `verifylss` из пакета INTLAB показала вполне сносные результаты. Однако мы имеем наилучшую скорость работы, если в качестве базового алгоритма используем процедуру Хансена-Блика-Рона.

Таким образом, при выборе базового алгоритма следует предпочесть процедуру Хансена-Блика-Рона.

Заметим, что при варьировании параметра θ, т.е. изменении свойств интервальной матрицы $\boldsymbol{A}$, время работы программы меняется довольно значительно.

Для характеристики свойств интервальной матрицы системы мы использовали следующие величины:

1) $\rho = \rho\left(|(\operatorname{mid} \boldsymbol{A})^{-1}| \cdot \operatorname{rad} \boldsymbol{A}\right)$ — спектральный радиус матрицы $|(\operatorname{mid} \boldsymbol{A})^{-1}| \cdot \operatorname{rad} \boldsymbol{A}$;

2) $\Delta\sigma = \sigma_{\min}(\operatorname{mid} \boldsymbol{A}) - \sigma_{\max}(\operatorname{rad} \boldsymbol{A})$ — разность между наименьшим и наибольшим сингулярными числами матриц $\operatorname{mid} \boldsymbol{A}$ и $\operatorname{rad} \boldsymbol{A}$ соответственно.

Таблица 8. Сравнение базовых алгоритмов (Пример 1).

Параметр θ м-цы системы	Время работы программы на основе базовых алгоритмов					
	K	**MK**	**G**	**GS**	**HBR**	**V**
$n = 5$						
10	3.3141	2.8475	2.8144	2.8184	1.0465	2.0098
14	0.3866	0.3267	0.7009	0.7143	0.2847	0.3380
18	0.2692	0.2590	0.5691	0.6146	0.2217	0.2703
22	0.2102	0.2290	0.5003	0.5710	0.1953	0.2385
26	0.2125	0.1995	0.4993	0.5395	0.1890	0.2076
30	0.2144	0.1973	0.4997	0.5478	0.1883	0.2046
$n = 8$						
16	390.2556	103.0986	303.9039	289.9082	65.0992	89.6327
20	26.0022	7.0169	17.1965	17.3720	5.4327	6.1442
24	3.8117	2.6047	6.3058	6.5445	2.0045	1.8814
28	1.9779	1.7448	4.7304	4.8175	1.5000	1.4086
32	1.3841	1.3575	3.4790	3.5781	1.1043	1.0383
36	1.0880	1.3010	2.7362	2.7911	0.8382	0.7896
40	0.8941	0.9629	2.2655	2.2837	0.6887	0.6495
44	0.8449	0.8131	2.1070	2.0919	0.6416	0.6045
48	0.7438	0.7638	1.9481	1.9712	0.5898	0.5574
$n = 10$						
22	3282.7079	610.1702	1142.0173	1028.6021	166.8741	520.3718
26	100.6111	30.1484	84.3959	82.1482	25.4255	26.21199
30	20.6401	12.9534	34.3599	34.1455	10.2935	9.47044
34	11.9682	6.3594	18.3697	18.3715	5.3928	4.95621
38	6.5616	4.9949	14.2529	14.2307	4.1852	3.8525
42	5.4355	3.7195	11.0988	11.0751	3.2568	3.0034
46	3.8658	3.1829	8.6940	8.6610	2.5558	2.4238
50	2.8362	2.5957	8.4592	8.4089	2.4747	2.2906
54	2.6810	2.5162	7.0172	6.9698	2.0580	1.8986
58	2.2866	2.3118	5.8132	5.7508	1.7045	1.5730

Напомним, что сингулярными числами матрицы $A \in \mathbb{R}^{n\times n}$ называются неотрицательные квадратные корни из собственных чисел матрицы $AA^\top$.

Рассмотрение указанных выше величин основывается на следующих признаках неособенности интервальных матриц.

Признак Риса-Бека. *Пусть интервальная $n \times n$-матрица $\boldsymbol{A}$ такова, что* $\operatorname{mid} \boldsymbol{A}$ *неособенна и*

$$\rho\left(|(\operatorname{mid} \boldsymbol{A})^{-1}| \cdot \operatorname{rad} \boldsymbol{A}\right) < 1.$$

Тогда $\boldsymbol{A}$ неособенная.

Теорема. *Пусть интервальная $n \times n$-матрица $\boldsymbol{A}$ такова, что* $\operatorname{mid} \boldsymbol{A}$ *неособенная и*

$$\max_{1\le j\le n} \left(\operatorname{rad} \boldsymbol{A} \cdot |(\operatorname{mid} \boldsymbol{A})^{-1}|\right)_{jj} \ge 1.$$

Тогда $\boldsymbol{A}$ особенная.

Признак Румпа. *Пусть для нтервальной $n \times n$-матрицы $\boldsymbol{A}$ имеет место*

$$\sigma_{\max}(\operatorname{rad} \boldsymbol{A}) < \sigma_{\min}(\operatorname{mid} \boldsymbol{A}),$$

то она неособенная.

Признак Рона-Рекса. *Если для нтервальной матрицы $\boldsymbol{A}$ имеет место*

$$\sigma_{\min}(\operatorname{rad} \boldsymbol{A}) \ge \sigma_{\max}(\operatorname{mid} \boldsymbol{A}),$$

то она особенная.

В Табл. 9 приведены значения рассматриваемых характеристик ρ и $\Delta\sigma$ матрицы Ноймайера при различных значениях её диагонального параметра θ и размерности n.

Отметим, что при близких к нулю значениях ρ и достаточно больших $\Delta\sigma$ время работы программы мало. Однако оно экспоненциально возрастает при приближении матрицы системы к границам неособенности, т. е. при стремлении ρ к единице и $\Delta\sigma$ к нулю.

На Рис. 1, 2 мы показали зависимость времени работы программы от характеристик ρ и $\Delta\sigma$ интервальной матрицы $\boldsymbol{A}$ размерности $n = 5$ для различных базовых алгоритмов.

Для тестирования разработанного алгоритма и его модификаций мы использовали также интервальную систему Шарого (Пример 2). Провели ряд

Таблица 9. Характеристики ρ и $\Delta\sigma$ матриц Ноймайера.

Параметр θ	Характеристики ρ и $\Delta\sigma$	
	$\rho\left(\lvert(\mathrm{mid}\ \boldsymbol{A})^{-1}\rvert \cdot \mathrm{rad}\ \boldsymbol{A}\right)$	$\sigma_{\min}(\mathrm{mid}\ \boldsymbol{A}) - \sigma_{\max}(\mathrm{rad}\ \boldsymbol{A})$
$n = 5$		
10	0.5397	5
14	0.359	9
18	0.2674	13
22	0.2125	17
26	0.176	21
30	0.1501	25
$n = 8$		
16	0.5884	8
20	0.4503	12
24	0.3633	16
28	0.3037	20
32	0.2605	24
36	0.2279	28
40	0.2024	32
44	0.1819	36
48	0.1652	40
$n = 10$		
24	0.5392	14
28	0.4423	18
32	0.374	22
36	0.3235	26
40	0.2846	30
44	0.2539	34
48	0.2291	38
52	0.2086	42
56	0.1914	46
60	0.1767	50

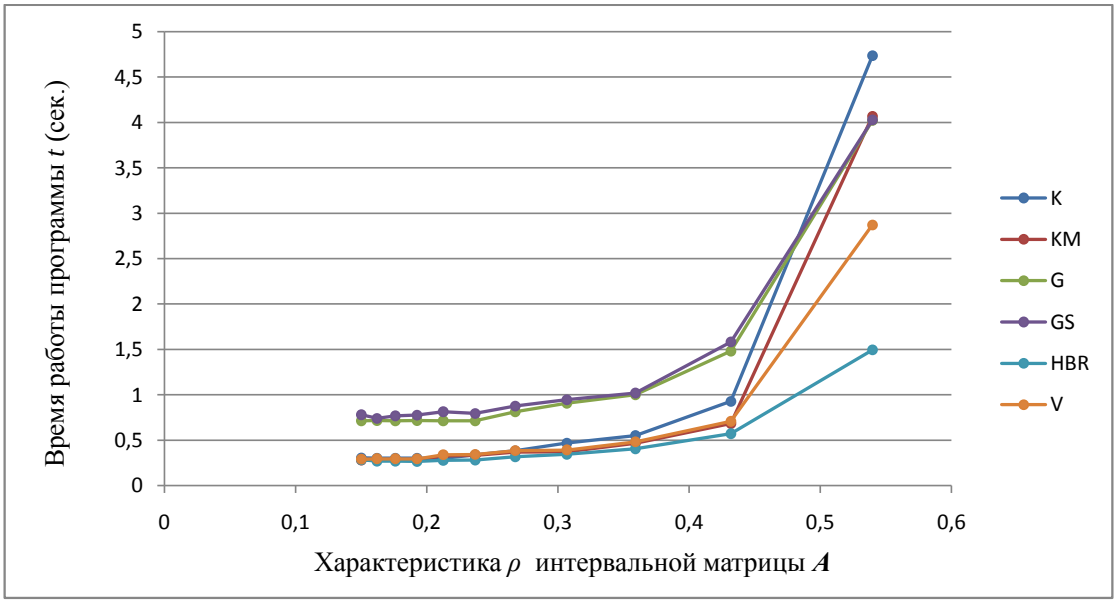


Рис. 1. Зависимость времени работы программы от характеристики ρ матрицы Ноймайера.

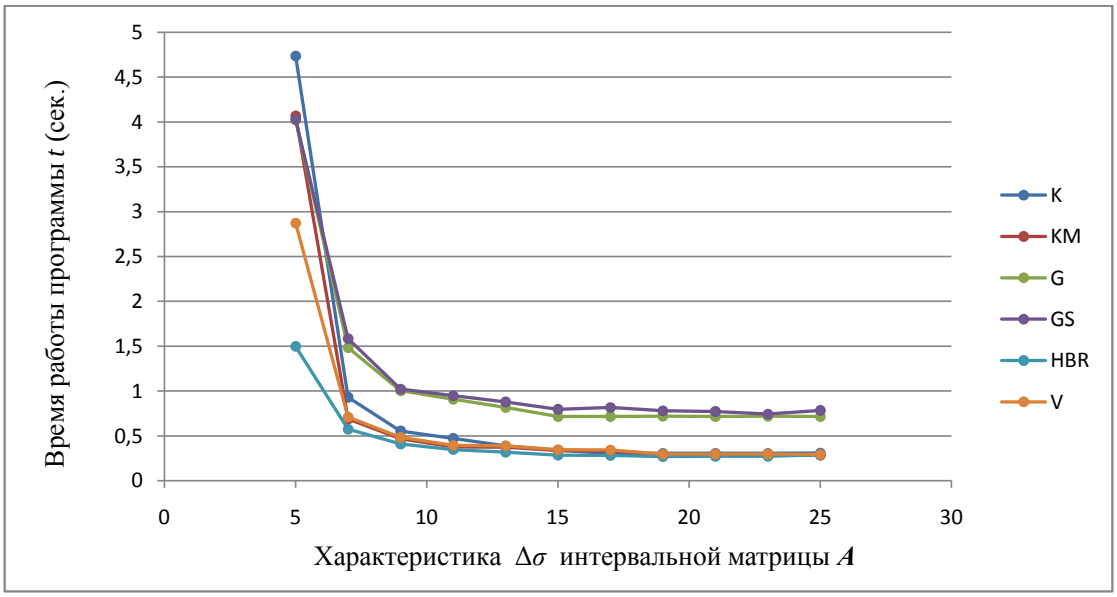


Рис. 2. Зависимость времени работы программы от характеристики $\Delta\sigma$ матрицы Ноймайера.

численных экспериментов при разных значениях размерности n матрицы системы, её параметров α, β, N и базовых методов внешней оценки множества решений интервальной системы.

В Табл. 10 приведены характеристики ρ и $\Delta\sigma$ интервальной матрицы системы, а также время нахождения оптимальной нижней оценки первой компоненты множества решений ИСЛАУ на основе разных базовых алгоритмов (**G** — интервальный метод Гаусса, **GS** — интервальный метод Гаусса-Зейделя, **HBR** — процедура Хансена-Блика-Рона, **V** — процедура `verifylss` из пакета INTLAB). На Рис. 3 изображены диаграммы, показывающие эффективность работы модификаций программы при разных значениях параметров и размерности интервальной системы Шарого.

Анализируя полученные результаты, можно сделать следующие выводы. Очень медленно работает программа, использующая в качестве базового метод Кравчика и его модификацию (мы даже не приводим эти данные в Табл. 10). Наилучшую скорость работы имеем в случае, когда базовым методом является поцедура Хансена-Блика-Рона.

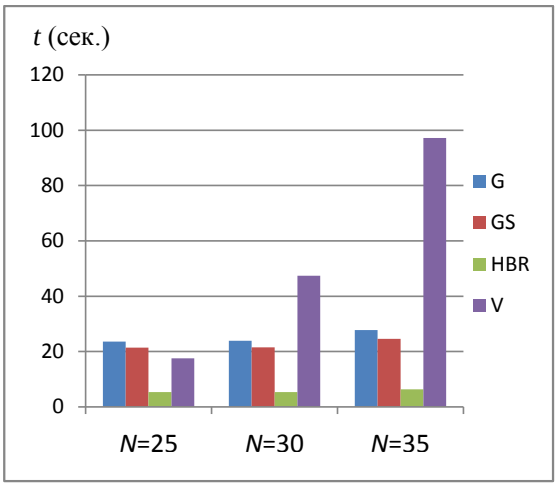
Отметим, что скорость работы программы на основе процедуры `verifylss` из пакета INTLAB значительно падает при возрастании размерности интервальной системы. Модификации на основе базовых методов Гаусса и Гаусса-Зейделя показывают достаточно близкие по эффективности результаты.

Из Табл. 10 видно, что при фиксированных значениях размерности n и параметров α, β возрастание значения параметра N не приводит к изменению характеристики $\Delta\sigma$ интервальной матрицы системы (при этом характеристика ρ варьирует незначительно). Поэтому скорость работы программы, основанной на базовых методах Гаусса, Гаусса-Зейделя и процедуре Хансена-Блика-Рона мало меняется. Однако модификация программы, использующая процедуру `verifylss` оказывается весьма чувствительной к изменению парметра N интервальной матрицы $\boldsymbol{A}$ (см. Рис. 3).

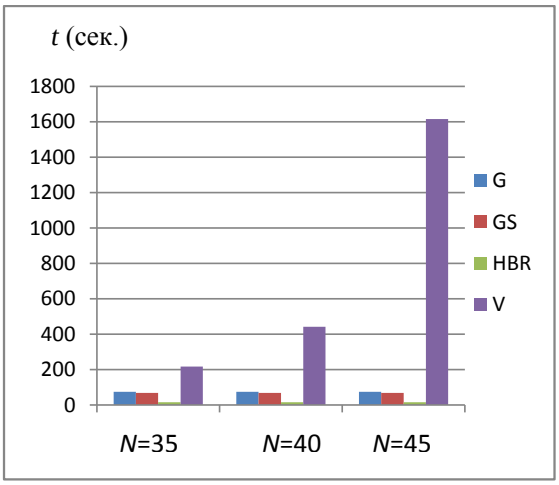
Далее мы провели апробацию разработанных алгоритмов и их модификаций на тестовой интервальной системе, описанной в Примере 3 (см. Табл. 11). Сравнивая результаты работы алгоритмов, основанных на разных базовых методах, можно сделать выводы, аналогичные тем, что были получены ранее.

Таким образом, на ряде тестовых примеров мы апробировали шесть базовых методов для использования их в разработанном алгоритме дробления параметров с модификацией Рона. Среди них наиболее эффективной оказалась процедура Хансена-Блика-Рона. Время работы алгоритма нахождения оптимальной внешней оценки множества решений ИСЛАУ зависит как от её раз-

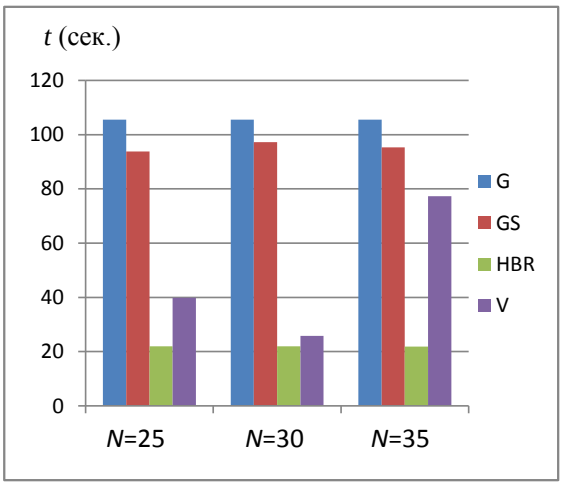
мерности, так и от близости системы к границам неособенности. Напомним, что свойства интервальной матрицы ИСЛАУ мы характеризовали с помощью величин $\rho = \rho\left(|(\mathrm{mid}\ \boldsymbol{A})^{-1}| \cdot \mathrm{rad}\ \boldsymbol{A}\right)$ и $\Delta\sigma = \sigma_{\min}(\mathrm{mid}\ \boldsymbol{A}) - \sigma_{\max}(\mathrm{rad}\ \boldsymbol{A})$.



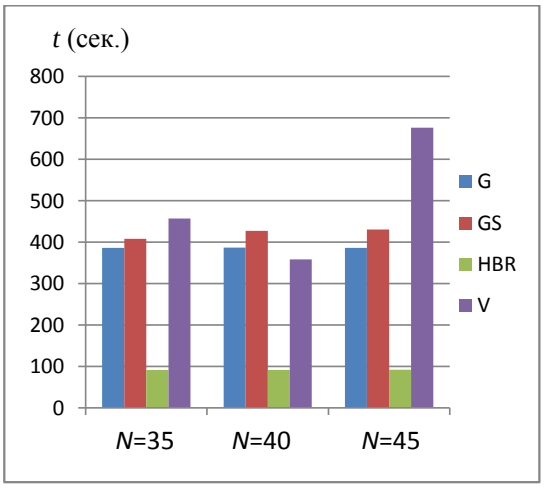
а) $n = 20$, $\alpha = 0.6$, $\beta = 0.8$



б) $n = 30$, $\alpha = 0.6$, $\beta = 0.8$



в) $n = 20$, $\alpha = 0.4$, $\beta = 0.6$



г) $n = 30$, $\alpha = 0.4$, $\beta = 0.6$

Рис. 3. Эффективность алгоритма при разных значениях параметров и размерности интервальной системы Шарого.

Таблица 10. Сравнение базовых алгоритмов (Пример 2).

Параметр N интервальной м-цы системы	Характеристика м-цы системы		Время работы программы на основе базовых алгоритмов			
	ρ	$\Delta\sigma$	**G**	**GS**	**HBR**	**V**
$\alpha = 0.4,\ \beta = 0.6,\ n = 10$						
15	0.6757	3.6	11.2251	9.0547	2.3568	3.5789
20	0.7353	3.6	10.8238	9.3226	2.3415	3.8539
25	0.7764	3.6	10.8116	9.2115	2.3225	3.8556
$\alpha = 0.4,\ \beta = 0.6,\ n = 20$						
25	0.6219	7.6	105.5093	93.826	21.9358	39.8465
30	0.6637	7.6	105.558	97.2705	21.9596	25.7558
35	0.6972	7.6	105.5546	95.3369	21.8803	77.2881
$\alpha = 0.4,\ \beta = 0.6,\ n = 30$						
35	0.6014	11.6	386.1295	407.671	91.4351	456.9929
40	0.6329	11.6	386.5222	426.9776	91.4321	358.4377
45	0.6598	11.6	385.8236	430.4134	92.3635	676.3317
$\alpha = 0.6,\ \beta = 0.8,\ n = 10$						
15	0.515	5.4	3.6593	3.5828	1.0812	2.3131
20	0.6029	5.4	7.9158	6.4509	1.6228	1.8764
25	0.6646	5.4	7.9047	6.3774	1.6264	1.9198
$\alpha = 0.6,\ \beta = 0.8,\ n = 20$						
25	0.4328	11.4	23.6313	21.4671	5.3091	17.544
30	0.4956	11.4	23.8926	21.5409	5.3101	47.3643
35	0.5458	11.4	27.7844	24.5736	6.2993	97.1439
$\alpha = 0.6,\ \beta = 0.8,\ n = 30$						
35	0.4021	17.4	74.7105	68.0642	16.1886	216.3686
40	0.4494	17.4	74.6486	67.9989	16.1929	441.9265
45	0.4897	17.4	74.6718	67.6438	16.1883	1616.238
$\alpha = 0.3,\ \beta = 0.7,\ n = 10$						
15	0.7353	2.7	13.5425	12.4836	3.3733	5.4107
20	0.7874	2.7	13.4357	12.8645	3.2593	4.1701
25	0.8224	2.7	13.4576	12.5255	3.3605	4.1867
$\alpha = 0.3,\ \beta = 0.7,\ n = 20$						
25	0.6868	5.7	157.6021	151.1048	34.3943	34.0165
30	0.7246	5.7	159.0196	144.8433	34.4162	34.0653
35	0.7543	5.7	161.887	141.4928	34.3464	37.1676
$\alpha = 0.3,\ \beta = 0.7,\ n = 30$						
35	0.6679	8.7	712.6947	656.571	149.9291	145.1066
40	0.6969	8.7	704.7249	670.453	150.615	146.7324
45	0.7212	8.7	712.2973	647.9654	151.1542	147.519

Таблица 11. Сравнение базовых алгоритмов (Пример 3).

Параметры м-цы системы $r = R$	Характеристика м-цы системы ρ	Время работы программы на основе базовых алгоритмов			
		G	**GS**	**HBR**	**V**
$n = 5$					
0.1	0.2281	0.4809	0.3112	0.2219	0.3785
0.2	0.4562	0.4228	0.3755	0.1162	0.1992
0.3	0.6844	1.1617	1.1511	0.3071	0.6881
$n = 10$					
0.1	0.2028	0.8131	0.8108	0.2000	0.2166
0.2	0.4056	2.1854	1.9216	0.4755	0.4353
0.3	0.6083	4.0831	5.3451	0.8205	1.3991
$n = 15$					
0.1	0.1998	1.7712	1.7698	0.4197	0.4104
0.2	0.3997	5.8617	5.1018	1.1540	0.8388
0.3	0.5995	13.8343	16.3355	2.5947	4.6909
$n = 20$					
0.1	0.1991	3.1070	3.0944	0.6960	0.6787
0.2	0.3983	9.1261	7.7893	1.5719	1.7288
0.3	0.5974	41.1623	43.1199	7.5761	11.4031
$n = 25$					
0.1	0.199	4.8638	4.8054	1.0628	1.5929
0.2	0.3979	14.1299	12.084	1.6945	2.0167
0.3	0.5969	90.4678	104.553	16.2730	27.1298
$n = 30$					
0.1	0.1989	10.9694	10.5463	1.5572	2.2389
0.2	0.3979	22.8073	19.8282	3.4505	4.1108
0.3	0.5968	216.7871	231.5724	39.0575	60.4330

7.2 Эффективность процедуры работы со списком

В § 5 были описаны четыре способа организации и обработки списка записей, в которых хранится информация о системах-потомках, полученных в результате дробления параметров:

1) список $\mathcal{L}$ организован в виде кучи;

2) записи списка $\mathcal{L}$ упорядочены по возрастанию оценки $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$;

3) в списке $\mathcal{L}$ выделяется упорядоченный подсписок $\mathcal{L}_l$ активных записей, имеющий некоторую фиксированную максимальную длину, а остальные записи организованы в виде кучи;

4) способ, предложенный Панковым.

Соответствующие модификации алгоритма оптимального внешнего оценивания множества решений ИСЛАУ, реализующие данные способы работы со списком, были апробированы на тестовых системах (Примеры 1-3). Результаты численных экспериментов (время (в сек.) нахождения оптимальной нижней оценки первой компоненты множества решений ИСЛАУ) представлены в Табл. 12. В качестве базового алгоритма использовалась процедура Хансена-Блика-Рона.

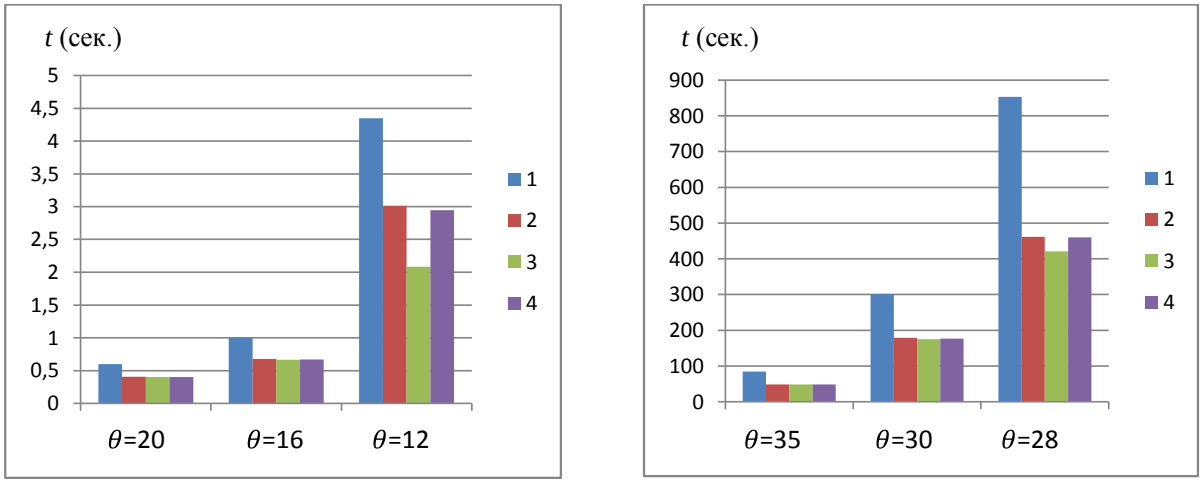
Наименее эффективным оказался способ организации списка в виде кучи (способ 1). Очевидно, на каждом шаге алгоритма достаточно много времени требуется на просмотр всего списка с целью поиска в нем ведущей записи.Скорость обработки списка значительно повышается, если список упорядочивается по возрастанию оценки $\Upsilon(\boldsymbol{Q}, \boldsymbol{r})$ (способ 2).

Заметим, что в системе MATLAB принято хранить каждый массив, независимо от его размерности, как вектор-столбец. Этот вектор образован объединением (конкатенацией) столбцов исходного массива. При добавлении или удалении элементов происходит изменение размеров и структуры массива, что требует определенных временных затрат. Поэтому имеет смысл поддерживать упорядоченность, а также производить чистку не всего списка $\mathcal{L}$, а некоторой его части — подсписка активных записей (способ 3 и 4).

Если сравнивать способы 3 и 4, то следует отдать предпочтение способу 3, при котором фиксируется максимальная длина подсписка активных записей. В способе 4, предложенном Панковым, длина подсписка активных записей определяется пороговой константой γ и может быть достаточно большой на некоторых шагах алгоритма, что приводит к снижению его быстродействия особенно в случае, когда матрица системы близка к границам неособенности.

Таблица 12. Эффективность алгоритмов организации и обработки списка.

Параметры и размерность интервальной системы		Время работы программы			
		1	**2**	**3**	**4**
Интервальная система Ноймайера					
$n = 6$	$\theta = 12$	4.3471	3.0131	2.0826	2.9439
	$\theta = 16$	1.0002	0.6776	0.6672	0.6704
	$\theta = 20$	0.5968	0.404	0.3998	0.4006
$n = 8$	$\theta = 16$	119.7672	73.6024	65.0392	72.8448
	$\theta = 20$	10.2414	5.4527	5.4327	5.427
	$\theta = 24$	3.8593	2.0369	2.0045	2.0314
$n = 10$	$\theta = 20$	5815.058	2323.042	1407.043	2235.154
	$\theta = 22$	336.5059	185.925	166.8741	184.0659
	$\theta = 24$	67.6053	41.3492	40.8864	41.1543
$n = 12$	$\theta = 28$	853.4038	461.6243	420.766	460.1469
	$\theta = 30$	301.1894	178.8155	175.215	176.6263
	$\theta = 35$	84.595	48.6436	48.0037	48.689
Интервальная система Шарого					
$n = 6$	$N = 20, n = 10$	5.7397	3.4281	3.2599	3.2751
	$N = 30, n = 20$	67.0141	35.0381	34.4162	35.5234
	$N = 40, n = 30$	297.9704	151.3018	150.615	151.3033
Интервальная система Тофта					
$n = 8$	$n = 10$	1.3404	0.856	0.8205	0.8579
	$n = 20$	13.3581	7.7729	7.5761	7.7841
	$n = 30$	71.6254	39.9219	39.0575	39.9303



а) $n = 6$ *б*) $n = 12$

Рис. 4. Эффективность алгоритмов организации и обработки списка.

По результатам тестовых экспериментов с системой Ноймайера при разных размерностях ($n = 6$ и $n = 12$) и значениях параметра θ построены диаграммы, изображенные на Рис. 4. Напомним, что при возрастании θ увеличивается характеристика ρ матрицы системы и уменьшается характеристика $\Delta\sigma$ (см. Табл. 9), т.е. матрица ИСЛАУ приближается к границам неособенности. При этом, как видно из Рис. 4, разница в быстродействии различных способов обработки списка, становится более заметной.

7.3 Сравнение процедур оптимального внешнего оценивания множества решений ИСЛАУ

В этом параграфе приведены результаты сравнительного анализа работы двух алгоритмов оптимального внешнего оценивания множества решений интервальной линейной системы уравнений $\boldsymbol{A}x = \boldsymbol{b}$:

- процедуры `verintervalhull` из пакета VERSOFT, основанной на методе Рона (см. §3.1)и написанной самим И.Роном;
- процедуры `linppsr`, реализующей метод дробления параметров с модификацией Рона, где в качестве базового метода используется процедура Хансена-Блика-Рона, а список записей организован и обрабатывается по способу 3 (см. §7.2).

В таблице 13 представлены результаты тестовых расчетов с интервальными системами (Примеры 1–3) при различных размерностях и значениях параметров матрицы ИСЛАУ.

Таблица 13. Эффективность алгоритмов `verintervalhull` и `linppsr`.

Параметры и размерность интервальной системы		Время работы алгоритма	
		`linppsr`	`verintervalhull`
Интервальная система Ноймайера			
$n = 5$	$\theta = 10$	8.0122	3.0176
	$\theta = 20$	1.6426	3.0192
	$\theta = 30$	1.6419	3.0254
$n = 10$	$\theta = 25$	617.2384	144.113
	$\theta = 30$	197.024	144.226
	$\theta = 45$	51.4229	144.8156
	$\theta = 60$	31.065	145.0121
$n = 12$	$\theta = 30$	2943.612	1084.5562
	$\theta = 50$	246.8048	1084.4765
	$\theta = 70$	99.3087	1085.5551
	$\theta = 90$	57.0493	1084.0012
Интервальная система Шарого			
$\alpha = 0.4, \beta = 0.6$	$n = 10, N = 15$	42.5282	144.7172
	$n = 20, N = 25$	882.344	более 8 ч.
	$n = 30, N = 35$	5483.106	более 8 ч.
$\alpha = 0.6, \beta = 0.8$	$n = 10, N = 15$	17.4779	142.084
	$n = 20, N = 25$	209.2503	более 8 ч.
	$n = 30, N = 35$	969.2808	более 8 ч.
Интервальная система Тофта			
$r = 0.2$	$n = 10$	5.7502	9.1158
	$n = 20$	56.5243	1321.2495
	$n = 30$	161.3937	более 8 ч.

Заметим, что задача вычисления оптимальных внешних оценок множества решений интервальной системы NP-трудна. Время работы обоих алгоритмов экспоненциально возрастает с увеличением размерности n системы.

Скорость работы процедуры `verintervalhull` не зависит от свойств интервальной матрицы $\boldsymbol{A}$ решаемой системы и остается неизменной при различных значениях её параметров. Быстродействие алгоритма `linppsr` снижается, если матрица $\boldsymbol{A}$ близка к границам неособенности. Однако при больших размерностях n процедура `linppsr` намного эффективнее, чем `verintervalhull`.

Как было показано в §3.1, искомые оценки $\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$ и $\max\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$, $\nu = 1, 2, \ldots, n$, достигаются на множестве не более чем 2^n экстремальных решений, т. е. верхняя оценка сложности метода Рона равна 2^n. Сложность методов дробления параметров без модификации Рона пропорциональна 2^{n^2} в худшем случае. Немотря на это, менее трудоемким оказывается алгоритм дробления параметров за счет того, что на каждом

шаге алгоритма существенно используется информация, полученная в ходе выполнения предыдущих шагов, т.е. алгоритм является адаптивным.

Важная особенность метода дробления параметров состоит ещё в том, что он порождает последовательность приближенных оценок для $\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$ и $\max\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$, $\nu = 1, 2, \ldots, n$, снизу и сверху соответственно. Поэтому в случае труднорешаемой задачи, когда размерность системы достаточно велика или матрица системы близка к границам неособенности, процесс работы алгоритма `linppsr` может быть прерван до своего полного завершения. При этом полученные оценки, будучи не оптимальными, могут служить более или менее точными приближениями искомых $\min\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$ и $\max\{\, x_\nu \mid x \in \Xi(\boldsymbol{A}, \boldsymbol{b})\}$, $\nu = 1, 2, \ldots, n$.

Далее мы приводим результат вычисления оптимальной оценки множества решений интервальной системы Тофта размерности $n = 20$ с параметрами $r = R = 0.2$

$$\begin{pmatrix} [0.8, 1.2] & 0 & \cdots & 0 & [0.8, 1.2] \\ 0 & [0.8, 1.2] & \cdots & 0 & [1.8, 2.2] \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & [0.8, 1.2] & [18.8, 19.2] \\ [0.8, 1.2] & [1.8, 2.2] & \cdots & [18.8, 19.2] & [19.8, 20.2] \end{pmatrix} x = \begin{pmatrix} [0.8, 1.2] \\ [0.8, 1.2] \\ \vdots \\ [0.8, 1.2] \end{pmatrix}.$$

Интервальный вектор оптимальных покомпонентных оценок имеет вид:

$$\boldsymbol{x} = \begin{pmatrix} [\ 0.5656, 1.4429] \\ [\ 0.4820, 1.3709] \\ [\ 0.3989, 1.2981] \\ [\ 0.3162, 1.2248] \\ [\ 0.2337, 1.1510] \\ [\ 0.1513, 1.0768] \\ [\ 0.0691, 1.0021] \\ [-0.0195, 0.9272] \\ [-0.1413, 0.8520] \\ [-0.2626, 0.7766] \\ [-0.3832, 0.7011] \\ [-0.5034, 0.6256] \\ [-0.6206, 0.5501] \\ [-0.7348, 0.4730] \\ [-0.8472, 0.3948] \\ [-0.9578, 0.3162] \\ [-1.0664, 0.2370] \\ [-1.1730, 0.1573] \\ [-1.2775, 0.0771] \\ [\ 0.0568, 0.1015] \end{pmatrix}.$$

1 Приложение. Код программы.

```
function solv=LinPPS_Ron(A,b,sizeL,epsilon,ncomp)
%  LinPPS_Ron Процедура оптимального внешнего оценивания объединенного
%             множества решений интервальной линейной системы уравнений,
%             основанная на методе дробления параметров с модификацией Рона.
%
%                solv=LinPPS_Ron(A,b,sizeL,epsilon)
%
%  solv     - интервальный вектор верхних и нижних оценок множества решений
%             системы A*x=b или вектор неопределенных значений NaN (если
%             система несовместна);
%  A        - квадратная интервальная матрица ИСЛАУ;
%  b        - интервальный вектор правой части ИСЛАУ;
%  sizeL    - длина подсписка активных записей;
%  epsilon  - точность оценки
%  ncomp    - номер оцениваемой компоненты решения (не указывается,
%             если оценивается весь вектор решения).
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
global nu omega n W s t Wch sch Omega K L Lk sizeL Q r x
n=length(A);
[n,n1]=size(A);
m1=length(b);
if n~=n1
    error('матрица A не квадратная')
  end
if n~=m1
    error('размерности A и b не равны')
end
  A=intval(A);
  b=intval(b);

InvA=inv_HBR(A);
solv=repmat(infsup(NaN,NaN),n,1);

for m=1:2
if isequal(m,2)                %  при m=1 поиск нижней оценки
    b=-b;                      %  при m=2 поиск верхней оценки
end
if (nargin==5)
    nmin=ncomp;
    nmax=ncomp;
else
    nmin=1;
    nmax=n;
end


for nu=nmin:nmax
x=HBR(A,b);              %  внешняя интервальная оценка мн-ва решений
if all(isnan(x))
    error('система несовместна');
end

omega=Inf;
                           %  первая(ведущая)запись в списке:
L(1).gamma=inf(x(nu));     %  нижний конец nu-ой компоненты внешней
                           %  интервальной оценки мн-ва решений
L(1).Q=A;                  %  интервальная матрица системы
L(1).r=b;                  %  интервальный вектор правых частей
L(1).Y=InvA;               %  обратная матрица системы
L(1).x=x;                  %  внешняя интервальная оценка мн-ва решений

          %  начальные (нулевые) контрольные матрицы и их запись в список:
L(1).W=zeros(n,n);                      %  матрица W
L(1).s=zeros(n,1);                      %  вектор-столбец s
L(1).t=zeros(1,n);                      %  вектор-строка t
```

```
while any(any(sup(L(1).Q)-inf(L(1).Q)~=0))  %&& abs(omega-L(1).gamma)>epsilon
    Omega=[];
    K=[];
    Wch=false;
    sch=false;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  проверка на монотонность
for i=1:n
dQ(i,1:n)=-L(1).Y(nu,i)*L(1).x(1:n); %  dQ - интервальные оценки производных
end                                   %       решения по коэф-м ситемы

    for i=1:n
    for j=1:n
        if dQ(i,j)>=0
              L(1).Q(i,j)=inf(L(1).Q(i,j));
              L(1).W(i,j)=1;
        end
        if dQ(i,j)<=0
            L(1).Q(i,j)=sup(L(1).Q(i,j));
            L(1).W(i,j)=-1;
        end
    end
   end
for i=1:n               %  dR=L(1).Y(nu,:) - интервальные оценки производных
    if L(1).Y(nu,i)>=0   %                   решения по коэф-м правых частей
        L(1).r(i)=inf(L(1).r(i));
        L(1).s(i)=-1;
    end
    if L(1).Y(nu,i)<=0
        L(1).r(i)=sup(L(1).r(i));
        L(1).s(i)=1;
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  поиск элемента, которому соответствует max произведения оценки
%                                   производной на ширину интевала
%  z1 - найденный элемент матрицы системы
%  im1 jm1 - его индексы
[c,I]=max(mag(dQ).*diam(L(1).Q));
[z1,jm1]=max(c);
im1=I(jm1);

%z2 - найденный элемент вектора правых частей
%im2 - его индекс
[z2,im2]=max(mag(L(1).Y(nu,1:n)).*(diam(L(1).r))');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  уточнение контрольной матрицы W с помощью 2x2 подматриц
out=false;
for i=1:n
    for j=1:n
        if (i~=im1 && j~=jm1)
            if (L(1).W(i,j)~=0 && L(1).W(i,jm1)~=0 && L(1).W(im1,j)~=0)
                L(1).W(im1,jm1)=L(1).W(i,j)*L(1).W(i,jm1)*L(1).W(im1,j);
                out=true;
                break
            end
        end
    end
if out
    break
end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  дробление на потомки
    W=L(1).W;
    s=L(1).s;
    t=L(1).t;
    Q=L(1).Q;
    r=L(1).r;

if z1>z2
    switch W(im1,jm1)
       case 0     %  случай 2-х потомков Q' и Q"
         Q(im1,jm1)=inf(Q(im1,jm1));
         W(im1,jm1)=1;
         Omega{length(Omega)+1}=[im1 jm1];
         Wch=true;
         m1=split(1,1);
         W=L(1).W;
         s=L(1).s;
         t=L(1).t;
         Omega=[];
         K=[];

         Q(im1,jm1)=sup(L(1).Q(im1,jm1));
         W(im1,jm1)=-1;
         Omega{length(Omega)+1}=[im1 jm1];
         Wch=true;
         m2=split(1,1);
         mu=min(m1(nu),m2(nu));
            case 1     %  случай 1-го потомка Q'
                Q(im1,jm1)=inf(Q(im1,jm1));
                m1=split(0,1);
                mu=m1(nu);
            case -1     %  случай 1-го потомка Q"
                Q(im1,jm1)=sup(Q(im1,jm1));
                m1=split(0,1);
                mu=m1(nu);
        end
    else
        switch s(im2)
            case 0  %  случай 2-х потомков r' и r"
                r(im2)=inf(r(im2));
                s(im2)=-1;
                K(length(K)+1)=im2;
                sch=true;
                m1=split(1,0);
                s=L(1).s;
                W=L(1).W;
                t=L(1).t;
                Omega=[];
                K=[];

                r(im2)=sup(L(1).r(im2));
                s(im2)=1;
                K(length(K)+1)=im2;
                sch=true;
                m2=split(1,0);
                mu=min(m1(nu),m2(nu));
            case 1      %  случай 1-го потомка r'
                r(im2)=sup(r(im2));
                m1=split(0,0);
                mu=m1(nu);
            case -1     %  случай 1-го потомка r"
                r(im2)=inf(r(im2));
                m1=split(0,0);
                mu=m1(nu);

            end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 L(1)=[];                                    %  удаление ведущей записи из списка
 if  omega>mu
   omega=mu;                                 %  вычисление нового значения omega
 end
                                             %  чистка списка
tL=length(L);                                 %  длина списка активных записей L
tLk=length(Lk);                               %  длина списка в виде кучи Lk

if ~isempty(L)&& ~any(any(sup(L(1).Q)-inf(L(1).Q)))
   for i=1:t
     Lk=transfer(Lk,L,tLk+i,i);
   end
L=[];
end

if isempty(L) && ~isempty(Lk)     %  если L пустой, а Lk не пустой,
    i=1;                          %  то чистка кучи Lk
   while i<=length(Lk)
      if Lk(i).gamma>omega
        Lk(i)=[];
      else i=i+1;
      end
   end
tLk=length(Lk);
   if tLk~=0
      [H,I]=sort([Lk.gamma]);     %  выбор активных записей из кучи
      T=min(sizeL,tLk);
      for j=1:T
        L=transfer(L,Lk,j,I(j));
      end
J=sort(I(1:T));
      for j=1:T
       Lk(J(j)-j+1)=[];
      end
   end
end
tL=length(L);
if ~isempty(L)&& isempty(Lk) %  если куча пуста, то чистка активных записей
    for i=1:t
      if L(i).gamma>omega
        L(i:t)=[];
        break
      end
    end
end

if isempty(L)&& isempty(Lk)
    ocenka=omega;
    break
end
  ocenka=L(1).gamma;
end
sv(m,nu)=ocenka;  %  текущая оценка nu-ой компоненты решения
Lk=[];
L=[];
end
end
for nu=nmin:nmax
    solv(nu)=infsup(sv(1,nu),-sv(2,nu)); %  интервальный вектор оценок
end                                      %  множества решений

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  процедура дробления параметра
function Mn=split(ind,fl)
global n nu omega L Lk sizeL invQ Q r x gamma Wch sch Omega K W s t
invQ=L(1).Y;
x=HBR(Q, r);
gamma=inf(x(nu));
Mn=mid(Q)\mid(r);
```

```
%если оценка gamma<omega и fl=true то изменяем матрицу W

  if gamma<omega && ind
%  перевычисление контрольных матрицы и векторов W, s, t
   tch=false;
   Wc=false;
   sc=false;
   tc=false;
   Lambda=[];

           while (Wch || sch || tch)
           if (sch || tch)


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  процедура изменения контрольной матрицы W
if sch
    for i=1:length(K)
        for j=1:n
            if t(j)~=0
                a=s(K(i))*t(j);
                     if W(K(i),j)~=a
                         Wc=true;
                         Omega{length(Omega)+1}=[K(i) j];
                     end
                W(K(i),j)=a;
            end
        end
    end
end

if tch
    for i=1:length(Lambda)
        for j=1:n
            if s(j)~=0
                a=s(j)*t(Lambda(i));
                     if W(j,Lambda(i))~=a
                         Wc=true;
                         Omega{length(Omega)+1}=[j Lambda(i)];
                     end
                W(j,Lambda(i))=a;
            end
        end
    end
end

           end
           if (Wch || tch)


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  процедура изменения контрольного вектора s
if Wch
    for i=1:length(Omega)
        i1=Omega{i}(1);
i2=Omega{i}(2);
        if t(i2)~=0
                a=W(i1,i2)/t(i2);
                     if s(i1)~=a
                         sc=true;
                         K(length(K)+1)=i1;
                     end
                s(i1)=a;
         end
     end
 end
```

```
if tch
    for i=1:length(Lambda)
        for j=1:n
            if (s(j)==0 && W(j,Lambda(i))~=0)
                a=W(j,Lambda(i))/t(Lambda(i));
                     if s(j)~=a
                         sc=true;
                         K(length(K)+1)=j;
                     end
                s(j)=a;
            end
        end
    end
end
           end

if (Wch || sch)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  процедура изменения контрольного вектора t
if Wch
    for i=1:length(Omega)
        i1=Omega{i}(1);
        i2=Omega{i}(2);
        if s(i1)~=0
                a=W(i1,i2)/s(i1);
                     if t(i2)~=a
                         tc=true;
                         Lambda(length(Lambda)+1)=i2;
                     end
                t(i2)=a;
         end
     end
 end


if sch
    for i=1:length(K)
        for j=1:n
            if (t(j)==0 && W(K(i),j)~=0)
                a=W(K(i),j)/s(K(i));
                     if t(j)~=a
                         tc=true;
                         Lambda(length(Lambda)+1)=j;
                     end
                t(j)=a;
            end
        end
    end
end

end

    Wch=Wc;
    sch=sc;
    tch=tc;
    Wc=false;
    sc=false;
    tc=false;

  end
  end
if gamma<omega
            %организуем новую запись в списке
            tL=length(L);
```

```
if tL<sizeL
   for l=1:tL
     if L(l).gamma>gamma
       if l==1
          l=2;
       end
          L(l+1:tL+1)=L(l:tL);
          L=inlist(L,l,fl);
        break
    end
     if l==tL
       L=inlist(L,tL+1,fl);
     end
   end

else
  for l=1:sizeL
       if L(l).gamma>gamma
          if l==1
             l=2;
          end
          c=length(Lk)+1;
          Lk=transfer(Lk,L,c,sizeL);
          L(l+1:sizeL)=L(l:sizeL-1);
          L=inlist(L,l,fl);
          break
       end
       if l==sizeL
          Lk=inlist(Lk,length(Lk)+1,fl);
       end
  end
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  добавление в список L новой записи
function L=inlist(L,l,fl)
global  Q r invQ gamma x W s t
                    L(l).Q=Q;
                    L(l).r=r;
                    L(l).gamma=gamma;
                    L(l).x=x;
                    L(l).W=W;
                    L(l).s=s;
                    L(l).t=t;
                    if fl
                       L(l).Y=inv_HBR(Q);
                    else
                       L(l).Y=invQ;
                    end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function L1=transfer(L1,L2,k1,k2)
     L1(k1).Q=L2(k2).Q;
     L1(k1).r=L2(k2).r;
     L1(k1).gamma=L2(k2).gamma;
     L1(k1).x=L2(k2).x;
     L1(k1).Y=L2(k2).Y;
     L1(k1).W=L2(k2).W;
     L1(k1).s=L2(k2).s;
     L1(k1).t=L2(k2).t;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  внешняя интервальная оценка мн-ва решений ИСЛАУ
function x = HBR(A,b)
global n
%n = dim(A);
C = inv(mid(A));
A = C*A;
b = C*b;
```

```
dA = diag(A);
A = compmat(A); % Компарант матрицы A
B = inv(A);
v = abss(B*ones(n,1));
setround(-1)
u = A*v;
if ~all(min(u)>0)
error('A не является H-матрицей')
else
dAc = diag(A);
A = A*B-eye(n);
setround(1)
w = max(-A./(u*ones(1,n)));
dlow = v.*w'-diag(B);
dlow = -dlow;
B = B+v*w;
u = B*abss(b);
d = diag(B);
alpha = dAc+(-1)./d;
k = size(b,2);
    if k==1
      beta = u./dlow - mag(b);
      x = (b+midrad(0,beta))./(dA+midrad(0,alpha));
    else
       v=ones(1,k);
       beta = u./(d*v) - mag(b);
       x = ( b + midrad(0,beta) ) ./ ( ( dA + midrad(0,alpha) ) * v );
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  нахождение обратной интервальной матрицы
function r = inv_HBR(a)
  n = dim(a);
  r = HBR(a,speye(n));
```

Литература

[1] Бауэр Ф. Л., Гооз Г. Информатика. — Москва: Издательство «Мир», 1996.

[2] Зорич В. А. Математический анализ. Т. 1. — Москва: Наука, 1981. Т. 2 — Москва: Наука, 1984.

[3] Панков П. С. Алгоритмы доказательства устойчивых утверждений и глобальной оптимизации в ограниченной области. — Фрунзе, 1984. — 13 с. — Депонировано в ВИНИТИ, №5250-84Деп.

[4] Шарый С. П. Конечномерный интервальный анализ. — Электронная книга `http://www-sbras.nsc.ru/interval/Library/InteBooks/SharyBook.pdf`

[5] Шарый С. П. Оптимальное внешнее оценивание множеств решений интервальных систем уравнений. Часть 1 // Вычислительные технологии. — 2002. — Т.7, №6. С. 90–113.

Шарый С. П. Оптимальное внешнее оценивание множеств решений интервальных систем уравнений. Часть 2 // Вычислительные технологии. — 2003. — Т.8, №6. С. 84–109.

[6] Фидлер М., Недома Й., Рамик Я., Рон И., Циммерманн К. Задачи линейной оптимизации с неточными данными. — М. – Ижевск: РХД, 2008.

[7] Beeck H. Über die Struktur und Abschätzungen der Lösungsmenge von linearen Gleichungssystemen mit Intervallkoeffizienten // Computing. —1972. — Vol. 10. — P. 231–244.

[8] Gregory R. T. A collection of matrices for testing computational algorithms. — N.–Y.: Wiley-Interscience, 1978.

[9] Hargreaves G. I. Interval analysis in MATLAB // Manchester Center for Computational Mathematics, 2002. (`http://old.ict.nsc.ru/interval/Programming/INTLABtutor.pdf`)

[10] Krawczyk K. R., Neumaier A. An improved interval Newton operator // J. of Mathematical Analisis and Applications. — 1986. — Vol. 118. — P. 194–201.

[11] Kreinovich V., Lakeyev A. V., Rohn J., Kahl P. Computational complexity and feasibility of data processing and interval computations. — Dordrecht: Kluwer, 1997.

[12] Kreinovich V., Lakeyev A. V., Noskov S. I. Optimal solution of interval linear systems is intractable (NP-hard) // Interval Computations. — 1993.— No 1. — P. 6–14.

[13] Mayer G. Epsilon–inflation in verification algorithms // J. of Computational and Applied Mathematics. — 1995. — Vol. 60. — P. 147–169.

[14] Neumaier A. A simple derivation of Hansen–Bliek–Rohn–Ning–Kearfott enclosure for linear interval equations // Reliable Computing. — 1999. — Vol. 5. — P. 131–136.

[15] Neumaier A. Interval methods for systems of equations. — Cambridge: Cambridge University Press, 1990.

[16] Neumaier A. New techniques for the analysis of linear interval equations // Linear Algebra and its Applications. — 1984. — Vol. 58. — P. 273–325.

[17] Nickel K. Die Überschatzung des Wertebereiches einer Funktion in der Intervallrechnung mit Anwendungen auf lineare Gleichungssysteme // Computing. – 1977. – Vol. 18. – P. 15–36.

[18] Oettli W. On the solution set of linear system with inaccurate coefficients // SIAM J. Numer. Anal. — 1965. — Vol. 2. — P. 115–118

[19] Rohn J. A handbook of results on interval linear problems. `http://www.cs.cas.cz/ rohn`.

[20] Rohn J., Kreinovich V. Comuting exact componentwise bounds on solutions of linear systems with interval data is NP–hard // SIAM Journal on Matrix Analysis and Applications. — 1995. — Vol. 16. — P. 415–420.

[21] Rohn J. Systems of linear interval equations // Linear Algebra and its Applications. — 1989. — Vol. 126. — P. 39–78.

[22] Rump S. M A note on epsilon-inflation // Reliable Computing. — 1998. — Vol. 4. — P. 371–375.

[23] Shary S. P. A new class of algorithms for optimal solution of interval linear systems // Interval Computations. — 1992.— No 2(4). — P. 18–29.

[24] Shary S. P. On optimal solution of interval linear equations // SIAM J. Numer. Analysis. — 1995. — Vol. 32. — No 2. — P. 610–630.

[25] Toft O. Sequential and parallel solution of linear interval equations // Eksamensproject: NI-E-92-04, Numerisk Institute, Danmarks Tekniske Hojskole. Lyngby, 1992.

[26] Verification software in MATLAB/INTLAB. — электронный ресурс, доступный на `http://www.cs.cas.cz/rohn/matlab`.