Пакет IntLinIncR3
Руководство пользователя

Содержание:

1 О пакете

Необходимое программное обеспечение — MATLAB®.
Основой алгоритма служит метод граничных интервалов [1].
Автор пакета IntLinIncR3 и метода граничных интервалов — И.А. Шарая
(Институт вычислительных технологий СО РАН, Новосибирск).
Пакет распространяется свободно. Исходные коды открыты.
Первая версия была опубликована 14 января 2014 г.
Адреса, с которых можно скачать последнюю версию пакета:
http://interval.ict.nsc.ru/Programing,
http://interval.ict.nsc.ru/sharaya/irash.html.

2 Назначение

Пакет `IntLinIncR3` предназначен для визуализации различных множеств решений интервальных и точечных (т.е. не интервальных, а обычных) систем отношений. Перечислим эти системы и множества.

Интервальные системы:

1) множество формальных решений интервального включения

$$\boldsymbol{C}x \subseteq \boldsymbol{d} \tag{1}$$

в арифметике Каухера, в котором
$\boldsymbol{C} = [\underline{C}, \overline{C}] \in \mathbb{KR}^{m\times 3}$ — интервальная матрица (заданы концы $\underline{C}$ и $\overline{C}$);
$x \in \mathbb{R}^3$ — вещественный вектор (неизвестен);
$\boldsymbol{d} = [\underline{d}, \overline{d}] \in \mathbb{K}\overline{\mathbb{R}}^m$ — интервальный вектор (заданы концы $\underline{d}$ и $\overline{d}$);
$m \in \mathbb{N}$ — натуральное число;
$\mathbb{KR} = \{[\underline{z}, \overline{z}] \mid \underline{z}, \overline{z} \in \mathbb{R}\}$ – интервалы Каухера (в отличие от классических интервалов $\mathbb{IR} = \{[\underline{z}, \overline{z}] \mid \underline{z}, \overline{z} \in \mathbb{R},\ \underline{z} \le \overline{z}\}$, для интервалов Каухера нет требования $\underline{z} \le \overline{z}$);
$\mathbb{K}\overline{\mathbb{R}} = \{[\underline{z}, \overline{z}] \mid \underline{z}, \overline{z} \in \overline{\mathbb{R}}\}$ – интервалы Каухера над расширенной числовой осью $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$;
умножение $\boldsymbol{C}$ на x — стандартное для арифметики Каухера;
включение "$\subseteq$" – задается парой неравенств $\underline{\boldsymbol{C}x} \ge \underline{d}$ и $\overline{\boldsymbol{C}x} \le \overline{d}$, каждое из которых понимаем покомпонентно, а $\underline{\boldsymbol{C}x}$ и $\overline{\boldsymbol{C}x}$ обозначают левый и правый концы интервального вектора $\boldsymbol{C}x = [\underline{\boldsymbol{C}x}, \overline{\boldsymbol{C}x}]$;

2) все возможные множества AE-решений интервальной системы уравнений

$$\boldsymbol{A}x = \boldsymbol{b}, \quad \boldsymbol{A} \in \mathbb{IR}^{m\times 3},\ \boldsymbol{b} \in \mathbb{IR}^m,\ m \in \mathbb{N}; \tag{2}$$

3) все возможные множества кванторных решений для интервальной системы неравенств вида

$$\boldsymbol{A}x \ge \boldsymbol{b}, \quad \boldsymbol{A} \in \mathbb{IR}^{m\times 3},\ \boldsymbol{b} \in \mathbb{IR}^m,\ m \in \mathbb{N}, \tag{3}$$

или вида

$$\boldsymbol{A}x \le \boldsymbol{b}, \quad \boldsymbol{A} \in \mathbb{IR}^{m\times 3},\ \boldsymbol{b} \in \mathbb{IR}^m,\ m \in \mathbb{N}; \tag{4}$$

4) различные множества кванторных решений для смешанной интервальной системы уравнений и неравенств

$$\boldsymbol{A}x\ \sigma\ \boldsymbol{b},\quad \boldsymbol{A}\in\mathbb{IR}^{m\times 3},\ \boldsymbol{b}\in\mathbb{IR}^{m},\ \sigma\in\{=,\geq,\leq\}^{m},\ m\in\mathbb{N};\tag{5}$$

речь идет только о тех кванторных решениях, в описании которых строкам с отношением "=" соответствует AE-порядок кванторных приставок.

Точечные системы:

1) множество решений системы

$$Ax+B|x|\geq c,\quad A,B\in\mathbb{R}^{m\times 3},\ c\in\mathbb{R}^{m},\ m\in\mathbb{N};\tag{6}$$

2) множество решений системы

$$|Ax-c|\leq B|x|+d,\quad A,B\in\mathbb{R}^{m\times 3},\ c,d\in\mathbb{R}^{m},\ m\in\mathbb{N};\tag{7}$$

3) множество решений смешанной системы линейных уравнений, неравенств и двусторонних линейных неравенств

$$\begin{cases} A_{(1)}x= b_{(1)}, & A_{(1)}\in\mathbb{R}^{m_1\times 3}, & b_{(1)}\in\mathbb{R}^{m_1}, & m_1\in\mathbb{N}\cup\{0\},\\ b_{(2)}\leq A_{(2)}x, & A_{(2)}\in\mathbb{R}^{m_2\times 3}, & b_{(2)}\in\mathbb{R}^{m_2}, & m_2\in\mathbb{N}\cup\{0\},\\ A_{(3)}x\leq b_{(3)}, & A_{(3)}\in\mathbb{R}^{m_3\times 3}, & b_{(3)}\in\mathbb{R}^{m_3}, & m_2\in\mathbb{N}\cup\{0\},\\ b_{(4)}\leq A_{(4)}x\leq b_{(5)}, & A_{(4)}\in\mathbb{R}^{m_4\times 3}, & b_{(4)},b_{(5)}\in\mathbb{R}^{m_4}, & m_4\in\mathbb{N}\cup\{0\}, \end{cases}\tag{8}$$

где $m_1+m_2+m_3+m_4>0$.

В [2] показано, что каждое из перечисленных множеств решений можно представить как множество формальных решений включения (1). Поэтому центральное место в пакете занимает визуализация именно этого множества, что отражено в названии пакета `IntLinIncR3` – Interval Linear Inclusion. Последние буквы `R3` обозначают русскоязычную версию для случая трех неизвестных (т.е. для $x\in\mathbb{R}^3$).

Замечание. Пакет `IntLinIncR3` ориентирован на иллюстрирование простых примеров (в публикациях, обучении и т.п.), поэтому наиболее правильно и безотказно он работает при условии, что все начальные данные целочисленны и лежат в диапазоне $[-10^2,10^2]$.

3 Структура пакета

Главная функция пакета — `CxindR3`. Она предназначена для визуализации множества решений включения (1).

Функции, используемые в главной:

`AddV`,	`ChooseDrawingBox`,	`DrawHedrons`,
`BoundaryIntervals`,	`ClearRows`,	`Intervals2Path`,
`ChangeVariables`,	`ClearZeroRows`,	`NonRepeatRows`.

Пакет содержит также вспомогательные функции. Они упрощают применение пакета к задачам, эквивалентным включению (1). Выбор вспомогательной функции зависит от системы отношений (2)–(7), которую надо обработать, а для интервальных систем — еще и от типа решений системы. Эта зависимость отражена в именах вспомогательных функций.

Имена вспомогательных функций для интервальных систем

система	тип решений				
	слабое	допусковое	управляемое	сильное	кванторное
(2) $\boldsymbol{A}x = \boldsymbol{b}$	`EqnWeakR3`	`EqnTolR3`	`EqnCtlR3`	`EqnStrongR3`	`EqnAEssR3`
(3) $\boldsymbol{A}x \geq \boldsymbol{b}$	`GeqWeakR3`	`GeqTolR3`	`GeqCtlR3`	`GeqStrongR3`	`GeqQtrR3`
(4) $\boldsymbol{A}x \leq \boldsymbol{b}$	`LeqWeakR3`	`LeqTolR3`	`LeqCtlR3`	`LeqStrongR3`	`LeqQtrR3`
(5) $\boldsymbol{A}x \ \sigma \ \boldsymbol{b}$	`MixWeakR3`	`MixTolR3`	`MixCtlR3`	`MixStrongR3`	`MixQtrR3`

Определения указанных в таблице типов решений, кроме кванторных, приведены в руководстве пользователя к пакету `IntLinIncR2`. Полный комплект определений имеется в [2], он обобщает терминологию из [3, 4]. Заметим, что вспомогательные функции `EqnAEssR3` и `MixQtrR3` предназначены только для тех кванторных решений, в описании которых строкам с отношением "=" соответствует AE-порядок кванторных приставок.

Для точечных систем имеется две вспомогательные функции: функция `Abs1R3` предназначена для системы (6) с одной операцией взятия абсолютного значения, а функция `Abs2R3` — для системы (7) с двумя такими операциями.

Пусковыми будем называть главную и вспомогательные функции пакета. Аргументы пусковых функций описаны в их телах. Чтобы посмотреть это описание в командном окне MATLAB, используйте команду `help`, например,

```
>> help EqnWeakR3
```

4 Свойства визуализируемых множеств

Обозначим визуализируемое множество решений через H, а через po_k (от англ. *piece in k-th orthant*) — пересечение H с k-м ортантом, $k \in \{1, 2, \ldots, 8\}$.

Непустое множество $M \subseteq \mathbb{R}^n$ называют *ограниченным*, если можно указать такое число $\lambda \in \mathbb{R}$, что все точки множества M удалены от начала координат на расстояние не большее λ. Пустое множество считают ограниченным. **Всякое множество** po_k**, как и** H **в целом, может быть ограниченным или неограниченным в зависимости от исходных данных системы.**

Полиэдром в $\mathbb{R}^n$ будем называть подмножество пространства $\mathbb{R}^n$, которое представимо как множество решений системы линейных неравенств

$$Ax \geq b, \qquad A \in \mathbb{R}^{m \times n},\ x \in \mathbb{R}^n,\ b \in \mathbb{R}^m,\ m, n \in \mathbb{N};$$

политопом — ограниченный полиэдр, а *полиэдральным множеством* — объединение конечного числа полиэдров. Заметим, что само пространство $\mathbb{R}^n$ — полиэдр, а пустое множество — политоп. **Все множества** po_k**,** $k = 1, \ldots, 8$**, — полиэдры, а** H **— полиэдральное множество.**

Множество называется *связным*, если из любой его точки можно, образно говоря, пройти в любую другую его точку, не выходя из самого множества. *Компонентой связности* называется такое связное подмножество, которое максимально по включению. **Каждое** po_k **связно, поскольку выпукло. Множество** H **не обязано быть связным и может иметь до 8 компонент связности.** (Множество H имеет 8 компонент связности в случае, когда все po_k, $k = 1, \ldots, 8$, непустые и попарно не пересекаются.)

Размерностью непустого *полиэдра* считают размерность его аффинной оболочки. Размерность пустого множества примем равной -1. Полиэдр будем называть *телесным*, если он имеет внутренние точки, и *тощим* в противном случае. **Отдельный кусок** po_k **множества** H **может иметь размерность от** -1 **(пустое множество) до 3 (телесный полиэдр).**

Множество H **может иметь полость** (определение полости и примеры множеств с полостью даны в разделе 6.6).

В $\mathbb{R}^n$ гиперплоскость P называется *опорной* к замкнутому множеству M, если P имеет с M хотя бы одну общую точку, и M содержится в замкнутом полупространстве с границей P. Пересечение полиэдра с опорной гиперплоскостью назовем *опорой* полиэдра. Очевидно, что опоры полиэдра — полиэдры.

Каждое множество po_k (как и всякий полиэдр в $\mathbb{R}^3$) **может иметь опоры размерности от** 0 **до** 2: опоры размерности 0 — это *вершины*, опоры размерности 1 — *ребра*, опоры размерности 2 — *грани*.

Точку пространства $\mathbb{R}^3$ будем называть *ориентиром* (*множества* H), если она служит вершиной какого-нибудь множества po_k, $k \in \{1, 2, \ldots, 8\}$. **Отсутствие ориентиров эквивалентно пустоте множества** H**.**

5 Обозначения в рисунках

Для работы с рисунками множества H в $\mathbb{R}^3$ введем следующие определения:

Брус обрезки — это такой брус (т.е. прямоугольный параллелепипед с ребрами параллельными координатным осям), пересечение которого с множеством решений будет показано на рисунке в результате работы пакета `IntLinIncR3`. Брусы обрезки бывают двух типов в зависимости от способа обрезки: брус *автоматической* обрезки пакет `IntLinIncR3` выбирает автоматически, а брус *принудительной* обрезки пользователь задает сам.

Обозначим через $\widetilde{\mathrm{po}_k}$ пересечение множества po_k с брусом обрезки.

Реальная грань — грань множества $\widetilde{\mathrm{po}_k}$, лежащая на границе множества решений H.

Грань автоматической обрезки — грань множества $\widetilde{\mathrm{po}_k}$, лежащая на границе бруса автоматической обрезки.

Грань принудительной обрезки — грань множества $\widetilde{\mathrm{po}_k}$, лежащая на границе бруса принудительной обрезки.

Грань от ортанта — грань множества $\widetilde{\mathrm{po}_k}$, возникающая от пересечения множества решений H с k-м ортантом. Такая грань не лежит ни на границе множества H, ни на границе бруса обрезки.

Обозначения в рисунках:

- • — ориентир,
- ■ — реальная грань,
- ■ — грань автоматической обрезки,
- ■ — грань принудительной обрезки.

Грани от ортантов на рисунках не визуализируются.

6 Рекомендации по подготовке и анализу рисунков

О том, как в зависимости от типа решений систем (1)–(8) выбирать и запускать главную или вспомогательную функцию пакета, подробно объяснено в руководстве пользователя к пакету `IntLinIncR2` для случая двух неизвестных. Поэтому в данном руководстве мы на этом останавливаться не будем. Отметим только, что случай трех неизвестных отличается наличием у пусковых функций входных аргументов `OrientPoints`, `transparency` и `varargin` и отсутствием выходных аргументов `P1`, `P2`, `P3`, `P4`.

В процессе работы пакета `IntLinIncR3` пользователю становится доступна следующая информация о множестве решений H:

1) сообщения в командном окне, среди которых всегда присутствует сообщение о числе ориентиров;
2) список ориентиров множества решений как выходной аргумент пусковой функции;
3) рисунок множества решений.

Число и список ориентиров — это объективные геометрические характеристики всего множества решений. Они не зависят от того, какая его часть представлена на рисунке. Способы получения пользователем списка ориентиров — это стандартные способы доступа к выходным аргументам функций в MATLAB, поэтому на них мы тоже останавливаться не будем.

Гораздо сложнее ситуация с рисунками. Если в случае двух неизвестных рисунок, созданный пакетом `IntLinIncR2`, дает полное представление о геометрии множества решений, то в случае трех неизвестных рисунок, созданный пакетом `IntLinIncR3`, — это всего лишь заготовка для анализа геометрии множества решений и для получения репрезентативного рисунка. Чтобы

- правильно подобрать рисунок-заготовку,
- проанализировать по нему устройство множества решений
- и сделать репрезентативный рисунок, который дает четкое представление о строении множества и годится для сохранения в “плоском” формате,

надо использовать не только стандартные инструменты MATLAB, но и специальные знания о пакете `IntLinIncR3`. Поэтому основную часть данного руководства занимают рекомендации по подготовке и анализу рисунков в пакете `IntLinIncR3`. Наиболее существенные из них отмечены знаком “**!**”.

6.1 Инструменты подготовки и анализа рисунков

У пользователя пакета `IntLinIncR3` есть два набора инструментов для подготовки и анализа рисунков множества решений:

- инструменты просмотра рисунков (Data Exploration Tools) в MATLAB, среди которых масштабирование (Zoom), поворот (Rotate 3D), освещение (Scene Light) и другие;
- *аргументы просмотра* в пусковых функциях пакета `IntLinIncR3`, к которым относятся обязательные входные аргументы `OrientPoints` и `transparency`, а также дополнительный входной аргумент `varargin`.

Аргумент `OrientPoints` предназначен для визуализации ориентиров. Он может принимать значения:

1 — ориентиры на рисунке обозначены точками,
0 — ориентиры на рисунке не видны.

Аргумент `transparency` регулирует прозрачность реальных граней множества решений. Его значения:

1 — реальные грани прозрачны,
0 — реальные грани непрозрачны.

Дополнительный входной аргумент `varargin` дает возможность пользователю задать брус принудительной обрезки множества решений как 6 чисел: `xb`, `xe`, `yb`, `ye`, `zb`, `ze`. При наличии такой шестерки чисел в списке входных аргументов пусковой функции брус обрезки равен $[\texttt{xb}, \texttt{xe}] \times [\texttt{yb}, \texttt{ye}] \times [\texttt{zb}, \texttt{ze}]$, а при отсутствии — пакет сам выбирает брус обрезки.

При первой визуализации множества решений для получения полного и правильного представления о геометрии этого множества надо установить следующие значения входных аргументов пусковой функции: !

```
>> OrientPoints = 1;
>> transparency = 1;
```

дополнительный аргумент `varargin` не использовать. Такие значения аргументов просмотра будем называть *стартовыми*. При последующих вызовах пусковой функции значения аргументов просмотра можно менять, чтобы получить более красивый рисунок (аргументы `OrientPoints`, `transparency`), или выделить его фрагмент (аргумент `varargin`).

6.2 Как сделать бриллиант

Опишем типичный путь получения хорошего рисунка множества решений.

Пример 1 (Диамант)**.** Нужен рисунок допускового множества решений системы

$$\begin{pmatrix} 3.5 & [0,2] & [0,2] \\ [0,2] & 3.5 & [0,2] \\ [0,2] & [0,2] & 3.5 \end{pmatrix} x = \begin{pmatrix} [-1,1] \\ [-1,1] \\ [-1,1] \end{pmatrix}.$$

Порядок действий.

1) *Ввод исходных данных системы.* В этом примере рассматривается система уравнений (2) с

$$\underline{A} = \begin{pmatrix} 3.5 & 0 & 0 \\ 0 & 3.5 & 0 \\ 0 & 0 & 3.5 \end{pmatrix}, \ \overline{A} = \begin{pmatrix} 3.5 & 2 & 2 \\ 2 & 3.5 & 2 \\ 2 & 2 & 3.5 \end{pmatrix}, \ \underline{b} = \begin{pmatrix} -1 \\ -1 \\ -1 \end{pmatrix}, \ \overline{b} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}.$$

Введем последовательно эти данные:

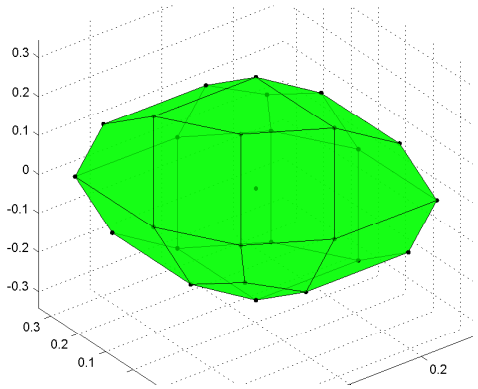
```
>> infA = [ 3.5 0 0; 0 3.5 0; 0 0 3.5 ];
>> supA = [ 3.5 2 2; 2 3.5 2; 2 2 3.5 ];
>> infb = [ -1; -1; -1 ];
>> supb = [ 1; 1; 1 ];
```

2) *Вызов пусковой функции со стартовыми значениями аргументов просмотра.* Чтобы получить объемный рисунок-заготовку, дающий правильное представление о геометрии множества решений, запустим вспомогательную функцию `EqnTolR3` со стартовыми значениями аргументов просмотра:

```
>> OrientPoints = 1;
>> transparency = 1;
>> EqnTolR3(infA,supA,infb,supb,OrientPoints,transparency);
```

Получим:

```
Число ориентиров = 27
```

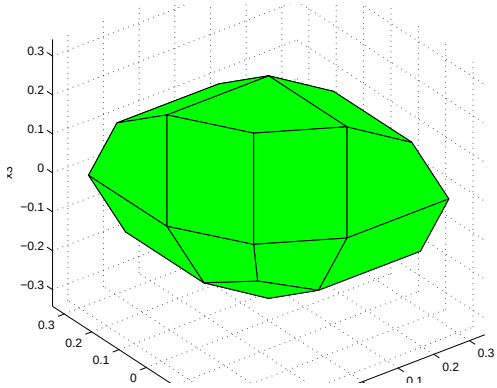


3) *Подбор аргументов просмотра.* Анализ рисунка показывает, что множество решений ограниченно (см. раздел 6.3), не имеет полости (см. раздел 6.6), и все непустые po_k в нем телесны. Такие множества более удачно смотрятся без ориентиров, а отключение прозрачности реальных граней для таких множеств — дело вкуса.

Запустим еще раз функцию `EqnTolR3`, но уже без визуализации ориентиров и без прозрачности реальных граней:

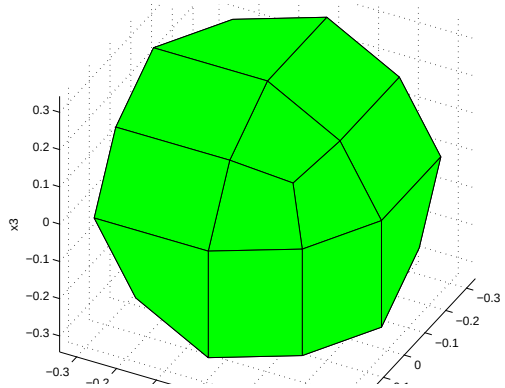
```
>> EqnTolR3(infA,supA,infb,supb,0,0);
```

Новый рисунок выглядит так. ⟶
На нем множество решений напоминает булыжник.

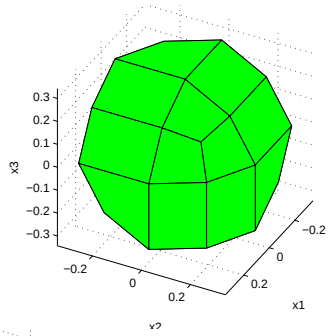


4) *Обработка рисунка.* Чтобы сделать из булыжника бриллиант, воспользуемся стандартными инструментами просмотра рисунков в MATLAB:

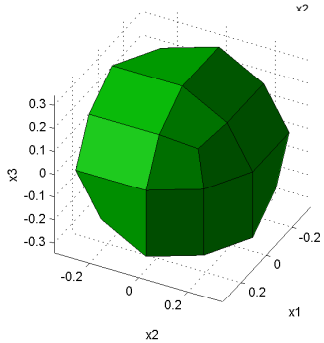
повернем (инструмент Rotate 3D) ↓



уменьшим в размере (инструмент Zoom) ↓



и включим свет (Scene Light). ⟶
Такой рисунок годится для сохранения в “плоском” формате (eps, pdf и т.п.).



Замечание. Все рисунки, которые приведены далее, после выдачи пакетом `IntLinIncR3` обработаны инструментами просмотра рисунков в MATLAB.

6.3 Неограниченные множества

Данный раздел посвящен двум средствам пакета `IntLinIncR3` для создания адекватных рисунков неограниченных множеств. Эти средства — автоматическая обрезка и визуализация ориентиров.

6.3.1 Автоматическая обрезка

Стандартный прием, к которому требуется прибегать в компьютерных системах при визуализации неограниченных множеств, заключается в указании диапазона по координатам. В результате мы видим только часть множества, ограниченную этим диапазоном.

Пакет `IntLinIncR3` тоже может визуализировать часть множества решений в предварительно заданном диапазоне координат. Такой диапазон координат мы называем *брусом принудительной обрезки*. Он задается дополнительным аргументом `varargin`. На рисунке грани, которые появляются в результате принудительной обрезки, прозрачны и желтого цвета, в отличие от реальных граней множества решений, которые имеют зеленый цвет и регулируемую прозрачность.

Стандартный прием имеет два недостатка:

1) пользователь должен сам провести предварительный анализ визуализируемого множества и выбрать подходящий диапазон координат,
2) рисунок в заданном диапазоне координат не позволяет судить о глобальных свойствах множества, в частности о неограниченности.

Преимущество пакета `IntLinIncR3` при визуализации неограниченных множеств в том, что указывать брус принудительной обрезки не обязательно. Если брус принудительной обрезки не задан, пакет `IntLinIncR3` сам выбирает диапазон координат для отрисовки множества решений, причем делает это так, чтобы в выбранном диапазоне была отражена вся существенная информация о множестве решений. Такой диапазон координат мы называем *брусом автоматической обрезки*. В случае автоматической обрезки пакет `IntLinIncR3` выполняет следующие действия:

- находит множество всех ориентиров,
- выбирает брус обрезки шире интервальной оболочки множества ориентиров,
- рисует пересечение множества решений с брусом обрезки.

На рисунке грани, возникающие от пересечения неограниченного множества решений с брусом автоматической обрезки, всегда прозрачные и имеют красный цвет.

В отличие от принудительной, автоматическая обрезка не требует предварительной работы пользователя и сохраняет информацию о глобальных свойствах множества решений.

Теперь приведем рекомендации, которые позволят пользователю по рисунку, полученному при автоматической обрезке, различать ограниченные и неограниченные множества решений.

1) Прежде всего отметим, что задача распознавания (не)ограниченности ! H сводится к распознаванию (не)ограниченности множеств po_k. Множество H ограниченно, если ограниченны все po_k, $k = 1, 2, \ldots, 8$. Если хоть одна из компонент po_k неограниченна, то H неограниченно.

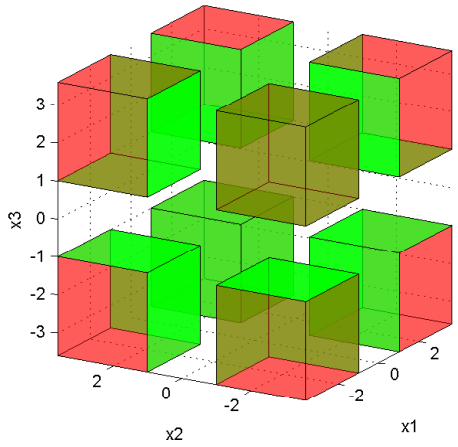
2) Если множество po_k имеет красную грань, то оно неограниченно. !

Пример 2. Объединенное множество решений для системы уравнений

$$\begin{pmatrix} [-1,1] & 0 & 0 \\ 0 & [-1,1] & 0 \\ 0 & 0 & [-1,1] \end{pmatrix} x = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

неограниченно и состоит из 8 трехгранных углов. Чтобы получить его рисунок, введите команды

```
infA = [ -1 0 0; 0 -1 0; 0 0 -1 ];
supA = [  1 0 0; 0  1 0; 0 0  1 ];
infb = [ 1; 1; 1 ];
supb = [ 1; 1; 1 ];
EqnWeakR3(infA,supA,infb,supb,0,1);
```



3) Если множество po_k телесно и все грани у него зеленые, то оно ограничено.

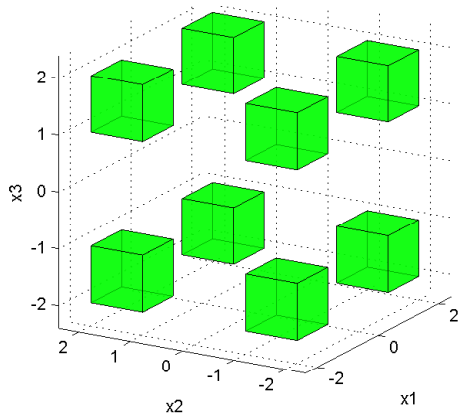
Пример 3. Объединенное множество решений системы уравнений

$$\begin{pmatrix} [-1,1] & 0 & 0 \\ 0 & [-1,1] & 0 \\ 0 & 0 & [-1,1] \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} x = \begin{pmatrix} 1 \\ 1 \\ 1 \\ [-2,2] \\ [-2,2] \\ [-2,2] \end{pmatrix}$$

ограниченно и состоит из 8 кубов.

Для получения рисунка введите команды

```
infA = [ -1 0 0; 0 -1 0; 0 0 -1; eye(3) ];
supA = [  1 0 0; 0  1 0; 0 0  1; eye(3) ];
infb = [ 1; 1; 1; -2; -2; -2 ];
supb = [ 1; 1; 1;  2;  2;  2 ];
EqnWeakR3(infA,supA,infb,supb,0,1);
```

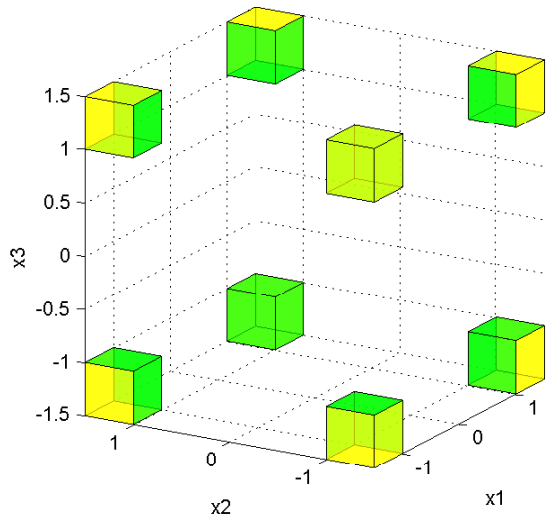


4) Задание дополнительного аргумента `varargin` часто мешает установить (не)ограниченность множества решений.

Пример 4. Если в Примерах 2 и 3 запустить пакет с указанием бруса принудительной обрезки $[-1.5, 1.5] \times [-1.5, 1.5] \times [-1.5, 1.5]$ командой

```
EqnWeakR3(infA,supA,infb,supb,0,1,-1.5,1.5,-1.5,1.5,-1.5,1.5);
```

рисунки будут одинаковы:



6.3.2 Визуализация ориентиров

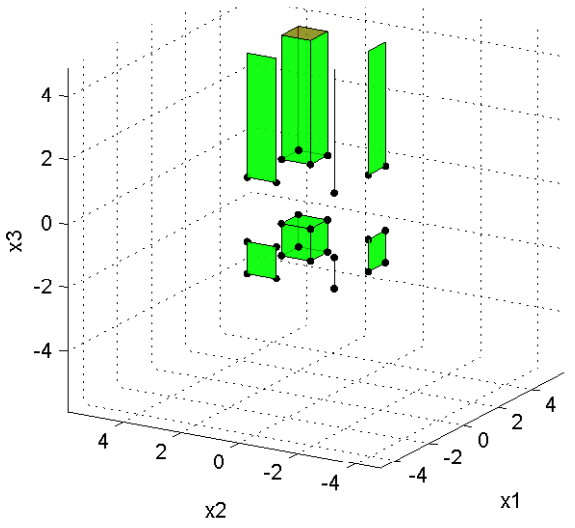
В Примерах 2 и 3 мы намеренно дали аргументу `OrientPoints` значение 0, чтобы показать, что иногда (например, когда все непустые множества po_k телесны) можно обойтись без прорисовывания ориентиров. В общем же случае, только сочетание автоматической обрезки с визуализацией ориентиров дает **критерий ограниченности множества** po_k: Множество po_k ограниченно тогда и только тогда, когда на рисунке, полученном при автоматической обрезке и визуализации ориентиров, либо множество po_k совсем не имеет ребер, либо на каждом его ребре обозначено по две вершины. !

Пример 5. Объединенное множество решений системы отношений

$$\begin{pmatrix} [-1,1] & 0 & 0 \\ 0 & [-1,1] & 0 \\ 0 & 0 & [-1,1] \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} x \begin{pmatrix} = \\ = \\ = \\ = \\ = \\ \geq \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \\ [-1,2] \\ [-1,2] \\ -2 \end{pmatrix}$$

состоит из ограниченных (отрезок, два квадрата, куб) и неограниченных кусков (луч, две полуполосы и полубесконечная квадратная призма). Рисунок можно получить с помощью команд

```
infA = [ -eye(3); eye(3) ];
supA = [  eye(3); eye(3) ];
infb = [ 1; 1; 1; -1; -1; -2 ];
supb = [ 1; 1; 1;  2;  2; -2 ];
relations=['=';'=';'=';'=';'=';'>'];
MixWeakR3(infA,supA,infb,supb,relations,1,1);
```



Итак, чтобы по рисунку множества решений можно было определить его ограниченность, надо дать пакету возможность автоматически выбрать брус обрезки (для этого просто не следует задавать аргумент `varargin`) и нарисовать ориентиры (т.е. установить аргумент `OrientPoints` равным 1).

6.4 Тощие, но важные

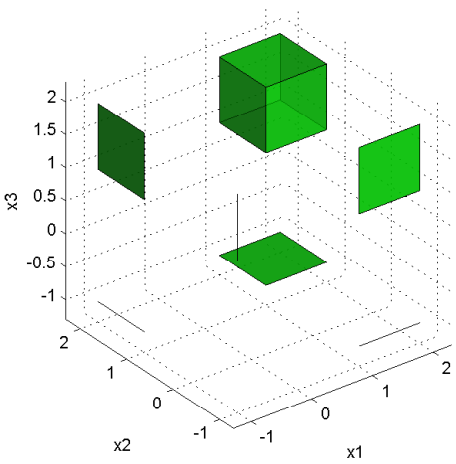
Чтобы на рисунке, созданном пакетом `IntLinIncR3`, тощие множества po_k отображались правильно и недвусмысленно, аргумент `OrientPoints` должен иметь значение 1. В противном случае, мы не увидим изолированных точек, которые являются отдельными компонентами связности множества H, и не сможем отличить ограниченные тощие множества po_k от неограниченных (отрезок от луча или прямой, плоский угол от треугольника, и т.п.)

!

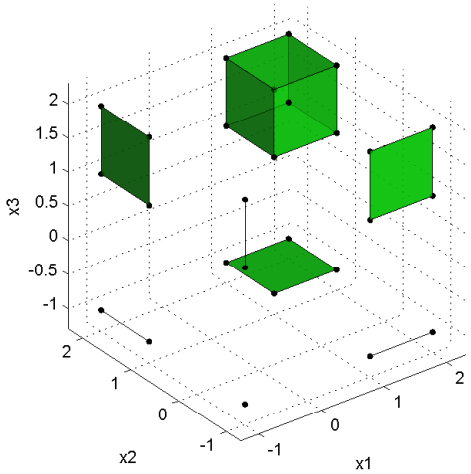
Пример 6. Объединенное множество решений системы уравнений

$$\begin{pmatrix} [-1,1] & 0 & 0 \\ 0 & [-1,1] & 0 \\ 0 & 0 & [-1,1] \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} x = \begin{pmatrix} 1 \\ 1 \\ 1 \\ [-1,2] \\ [-1,2] \\ [-1,2] \end{pmatrix}$$

состоит из точки, трех отрезков, трех квадратов и куба. На левом рисунке точка не видна, а отрезки и квадраты изображены так, что их ограниченность вызывает сомнения. На правом рисунке все составляющие множества решений изображены правильно и интерпретируются однозначно.



`OrientPoints` $= 0$



`OrientPoints` $= 1$

(Для ввода данных и запуска пакета используйте Пример 3.)

6.5 Пустое множество и все пространство

В данном разделе рассмотрим как по рисункам и сообщениям пакета `IntLinIncR3` судить о пустоте множества решений или о его совпадении со всем пространством $\mathbb{R}^3$.

6.5.1 Пустое множество

Пустота множества решений H означает пустоту всех его пересечений po_k, $k = 1, \ldots, 8$, с отдельными ортантами. Каждый полиэдр po_k не содержит прямых, поэтому его пустота равносильна отсутствию вершин. В целом, множество H пусто тогда и только тогда, когда не имеет ориентиров. В применении к пакету `IntLinIncR3` полученное теоретическое утверждение преобразуется в следующую рекомендацию по распознаванию пустоты множества H

Множество решений пусто тогда и только тогда, когда пакет не создает рисунка и выдает сообщение !

```
Число ориентиров = 0
Множество решений пусто
```

Эта рекомендация не зависит от того, при каких допустимых значениях аргументов просмотра были вызваны пусковые функции, но, как и другие рекомендации данной инструкции, опирается на то, что пакет правильно нашел геометрию множества решений.

Пример 7. Очевидно, что уравнение $\begin{pmatrix} 0 & 0 & 0 \end{pmatrix} x = 1$ не имеет решений в $\mathbb{R}^3$. Чтобы посмотреть, как его обрабатывает пакет `IntLinIncR3`, наберите

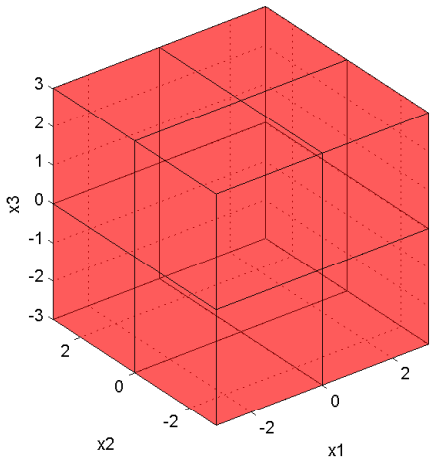
```
uC = [ 0 0 0 ];
oC = uC;
ud = 1 ;
od = ud;
CxindR3(uC,oC,ud,od,0,1);
```

6.5.2 Все пространство

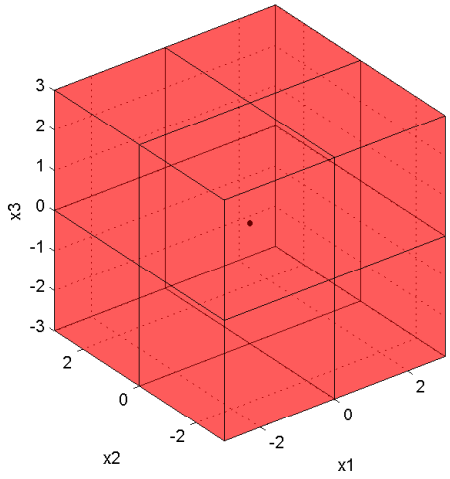
Множество решений совпадает со всем пространством $\mathbb{R}^3$ в том и только в том случае, когда, запустив пакет без дополнительного аргумента `varargnin`, мы получаем сообщение !

```
Число ориентиров = 1
```

и рисунок с прозрачным красным кубом. При `OrientPoints` $= 1$ начало координат отмечено на рисунке как ориентир, а значение аргумента `transparency` не влияет на картинку.



`OrientPoints` $= 0$



`OrientPoints` $= 1$

Пример 8. Объединенное множество решений интервального неравенства

$$\big([-1,1]\ \ [-1,1]\ \ [-1,1]\big)\, x \geq 0$$

совпадает со всем пространством. Чтобы получить его рисунки, надо ввести

```
infA = [ -1 -1 -1 ];
supA = [  1  1  1 ];
infb = [ 0 ];
supb = [ 0 ];
GeqWeakR3(infA,supA,infb,supb,0,0);
```

и поварьировать значения последних двух аргументов функции `GeqWeakR3`.

6.5.3 А это что?

Есть два рисунка, которые могут быть ошибочно истолкованы как пустота множества решений или совпадение множества решений со всем пространством, — это “пустой белый брус” (система координат без каких-либо объектов) и “пустой желтый брус”.

Что означает “пустой белый брус”?

Прежде всего следует подчеркнуть, что “пустой белый брус” никогда не означает пустоты множества решений или совпадения множества решений со всем пространством. !

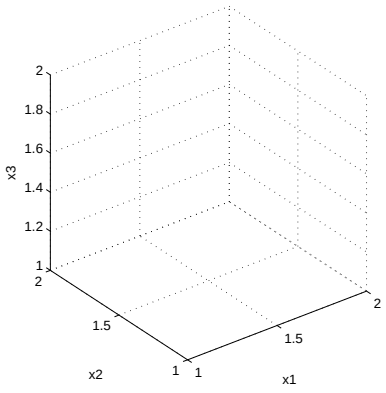
Если на рисунке вы видите систему координат, в которой ничего нет, то обратите внимание на аргументы `varargnin` и `OrientPoints`. Аргумент `varargnin` задает для множества решений брус принудительной обрезки. Поэтому когда пакет запущен с дополнительным аргументом `varargnin`, на рисунке мы видим только ту часть множества решений, которая попала в этот брус. Аргумент `OrientPoints` помогает увидеть или скрыть ориентиры множества решений. В зависимости от значений этих аргументов есть три пути правильной интерпретации “пустого белого бруса”.

1) Если аргумент `varargnin` задан и `OrientPoints` равен 1, то “пустой белый брус” означает, что брус принудительной обрезки целиком лежит в дополнении к множеству решений.

Пример 9. В примере Диамант (см. раздел 5.2, Пример 1) укажем брус принудительной обрезки $[1,2]\times[1,2]\times[1,2]$ и установим значение аргумента `OrientPoints` равным 1 в аргументах функции `EqnTolR3`:

```
>> EqnTolR3(infA,supA,infb,supb,1,0,1,2,1,2,1,2);
```

Получим рисунок ⟶



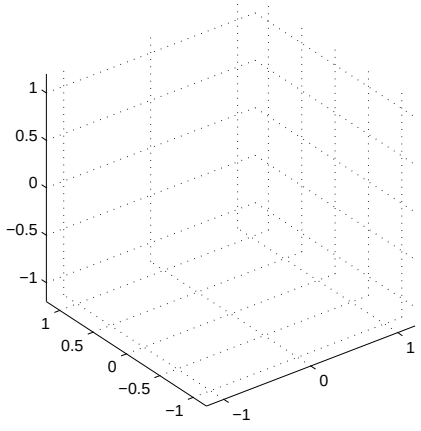
2) Если аргумент `varargnin` задан и `OrientPoints` $= 0$, то установите значение аргумента `OrientPoints` равным 1 и перезапустите пакет, чтобы проверить наличие изолированных точек множества решений в брусе принудительной обрезки.

3) Если аргумент `varargnin` не задан, "пустой белый брус" означает, что аргумент `OrientPoints` равен 0 и множество решений состоит из изолированных точек. Для правильного просмотра множества решений запустите пакет со значением `OrientPoints` равным 1.

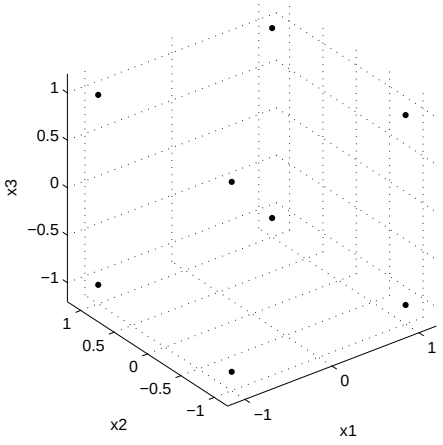
Пример 10. Объединенное множество решений уравнения

$$\begin{pmatrix} [-1,1] & 0 & 0 \\ 0 & [-1,1] & 0 \\ 0 & 0 & [-1,1] \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} x = \begin{pmatrix} 1 \\ 1 \\ 1 \\ [-1,1] \\ [-1,1] \\ [-1,1] \end{pmatrix}$$

состоит из восьми точек, но при `OrientPoints` $= 0$ выглядит как "пустой белый брус":



`OrientPoints` $= 0$



`OrientPoints` $= 1$

(Для ввода данных и запуска пакета используйте Пример 3.)

Что означает “пустой желтый брус”?

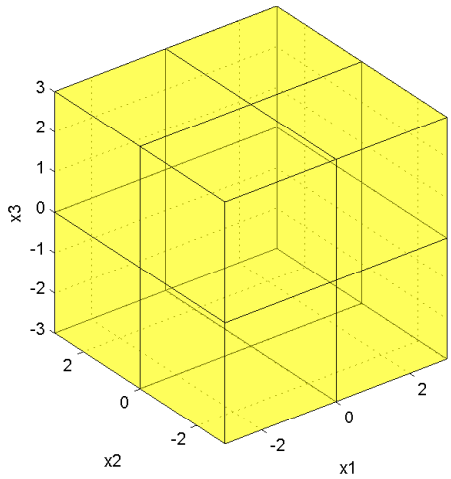
Отметим, что “пустой желтый брус” никогда не означает пустоты множества решений или совпадения множества решений со всем пространством. !

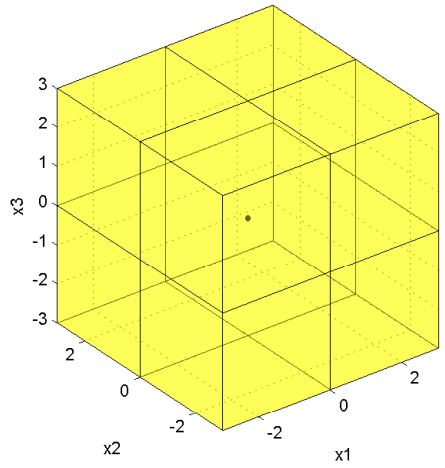
Если на рисунке вы видите “пустой желтый брус”, значит, пакет был запущен с дополнительным аргументом `varargnin` и брус обрезки, указанный в этом аргументе, целиком лежит в множестве решений. Начало координат в “пустом желтом брусе” может быть выделено как ориентир (если оно попало в брус принудительной обрезки и `OrientPoints` $= 1$).

Пример 11. Множество решений двустороннего неравенства $-5 \le x_1 \le 5$ целиком содержит брус $[-3,3] \times [-3,3] \times [-3,3]$. Вводя начальные данные

```
uC = [ 1 0 0 ];
oC = uC;
ud = -5;
od = 5;
```

и запуская функцию `CxindR3` с указанием бруса принудительной обрезки $[-3,3] \times [-3,3] \times [-3,3]$, в зависимости от значений аргумента `OrientPoints` получаем рисунки






```
CxindR3(uC,oC,ud,od,0,0,-3,3,-3,3,-3,3);
CxindR3(uC,oC,ud,od,0,1,-3,3,-3,3,-3,3);
```

```
CxindR3(uC,oC,ud,od,1,0,-3,3,-3,3,-3,3);
CxindR3(uC,oC,ud,od,1,1,-3,3,-3,3,-3,3);
```

В этом примере “пустой желтый брус” очень похож на рисунок всего пространства $\mathbb{R}^3$ и отличается только желтым цветом.

6.6 Как убедиться в наличии полости

Будем говорить, что множество решений H имеет *полость*, если начало координат не лежит в H, а всякий луч, выходящий из начала координат, пересекает H.

Полость представляет собой ограниченное полиэдральное множество. Оно не обязано быть выпуклым, но всегда звёздное (с каждой точкой x содержит весь отрезок $[0,1]x$) и потому связное. Начало координат является внутренней точкой полости. Образно говоря, полость — это домик для начала координат, стены домика состоят из реальных граней множества решений, а дверей и окон нет. В данном разделе мы обсудим настройки аргументов просмотра для выявления полости по рисунку.

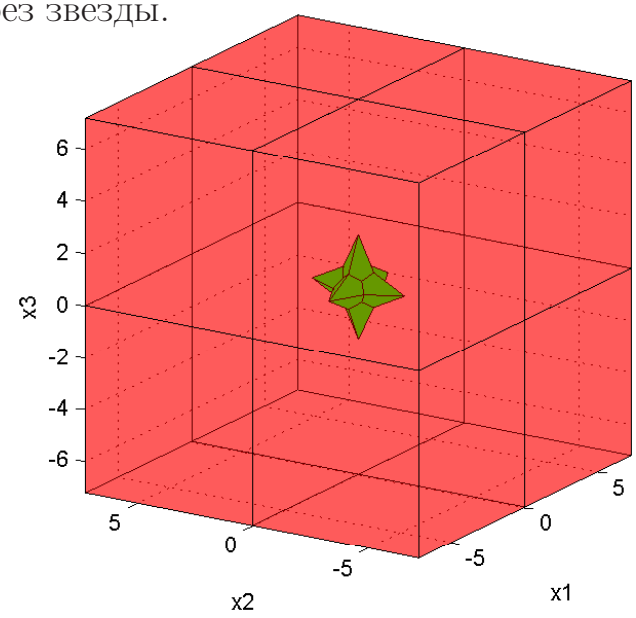
1) Использование автоматической обрезки (т.е. отсутствие дополнительного аргумента `varargin`), даже при произвольных значениях аргументов `OrientPoints` и `transparency`, делает очевидным наличие или отсутствие полости у достаточно широкого класса множеств решений.

Пример 12 (Пространство без звезды)**.** Управляемое множество решений системы интервальных уравнений

$$\begin{pmatrix} [-1,1] & [-2,2] & [-2,2] \\ [-2,2] & [-1,1] & [-2,2] \\ [-2,2] & [-2,2] & [-1,1] \end{pmatrix} x = \begin{pmatrix} 2 \\ 2 \\ 2 \end{pmatrix}$$

представляет собой пространство $\mathbb{R}^3$ без звезды.
Чтобы получить его рисунок,
введите команды

```
supA = [1 2 2; 2 1 2; 2 2 1];
infA = -supA;
infb = 2*ones(3,1);
supb = infb;
EqnCtlR3(infA,supA,infb,supb,0,0);
```



2) Если дополнить автоматическую обрезку прозрачностью реальных граней (`transparency` $= 1$), то класс множеств решений, для которых мы сможем по рисунку определить наличие или отсутствие полости, существенно расширится.

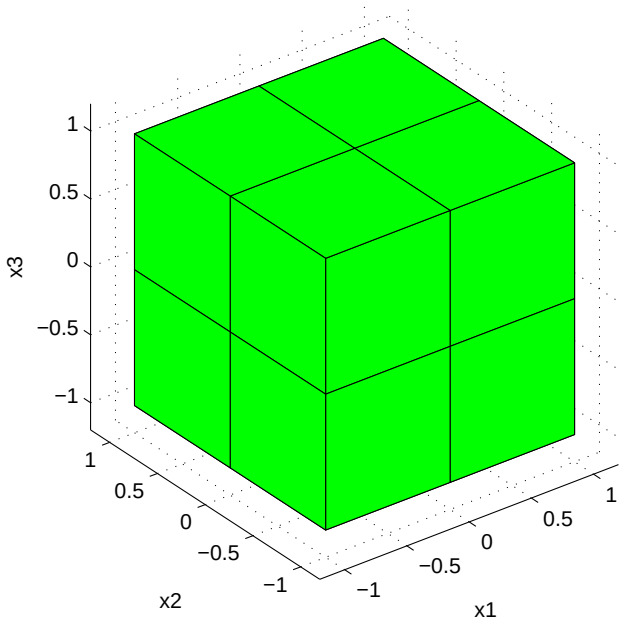
Пример 13 (Куб без зеркальной пирамиды)**.** Объединенное множество решений системы интервальных уравнений

$$\begin{pmatrix} [-1,1] & [-1,1] & [-1,1] \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} x = \begin{pmatrix} 0.5 \\ [-1,1] \\ [-1,1] \\ [-1,1] \end{pmatrix}$$

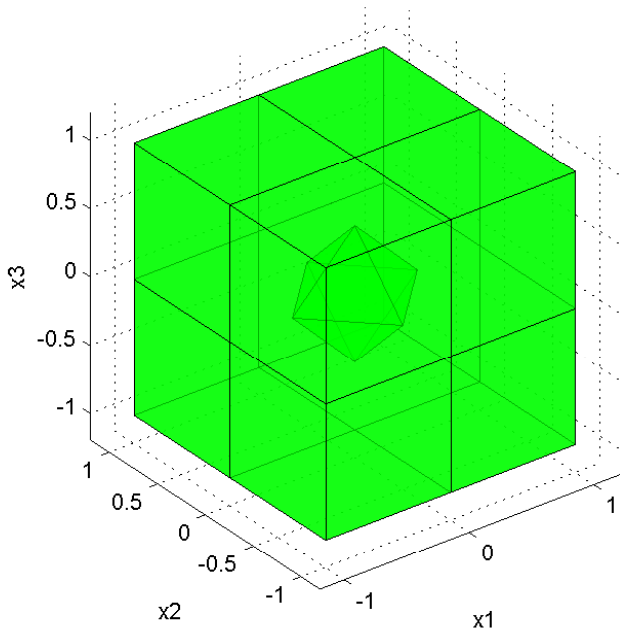
представляет собой куб без зеркальной пирамиды. Чтобы оценить влияние аргумента `transparency`, введите команды

```
infA = [-ones(1,3); eye(3)];
supA = [ones(1,3); eye(3)];
infb = [.5; -ones(3,1)];
supb = [.5; ones(3,1)];
EqnWeakR3(infA,supA,infb,supb,0,0);
```

Затем измените значение последнего аргумента функции `EqnWeakR3` с 0 на 1 и сравните полученные картинки:



`transparency` $= 0$



`transparency` $= 1$

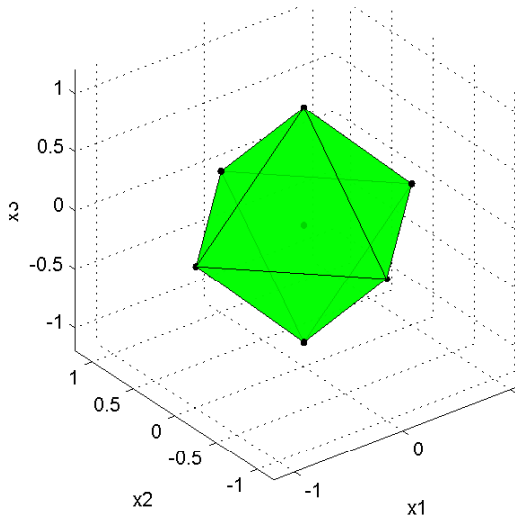
3) Использование полного набора стартовых аргументов просмотра позволяет однозначно определить по рисунку наличие или отсутствие полости практически для всех множеств решений.

В частности, полезно следующее правило: если при стартовых аргументах просмотра начало координат на рисунке обозначено как ориентир, то полости у множества решений нет.

Пример 14 (Зеркальная пирамида)**.** Допусковое множество решений для интервального уравнения

$$\begin{pmatrix}[-1,1] & [-1,1] & [-1,1]\end{pmatrix} x = \begin{pmatrix}[-1,1]\end{pmatrix}$$

представляет собой зеркальную пирамиду:

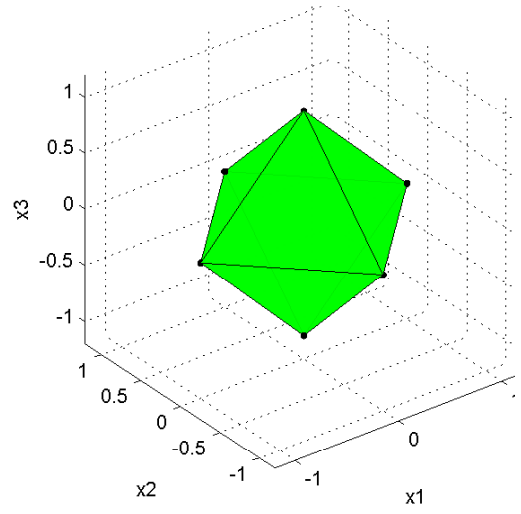



```
infA = [-ones(1,3)];
supA = [ones(1,3)];
infb = -1 ;
supb =  1 ;
EqnTolR3(infA,supA,infb,supb,1,1);
```

Пример 15 (Граница зеркальной пирамиды)**.** Множеством AE-решений системы

$$\begin{pmatrix}[-1,1]^{\forall} & [-1,1]^{\forall} & [-1,1]^{\forall} \\ [-1,1]^{\exists} & [-1,1]^{\exists} & [-1,1]^{\exists}\end{pmatrix} x = \begin{pmatrix}[-1,1]^{\exists} \\ 1^{\exists}\end{pmatrix}$$

служит граница зеркальной пирамиды:




```
infA = [-ones(2,3)];
supA = [ones(2,3)];
infb = [ -1; 1 ];
supb = [  1; 1 ];
Aq = [ 'A' 'A' 'A'; 'E' 'E' 'E' ];
bq = [ 'E' ; 'E' ];
EqnAEssR3(infA,supA,Aq,infb,supb,bq,1,1);
```

4) Наконец, в арсенале пользователя имеется дополнительный аргумент `varargin`, который дает возможность визуализировать пересечение множества решений H с желаемым брусом.

Брус принудительной обрезки, задаваемый аргументом `varargin`, можно использовать для того, чтобы получить представление о пересечении множества решений с отдельным ортантом $\mathcal{O}_k$. Для этого:

1) поместим одну из вершин бруса в начало координат,
2) противоположную вершину выберем внутри ортанта $\mathcal{O}_k$ на таком удалении, чтобы в брус с запасом попали все ориентиры из ортанта $\mathcal{O}_k$ (информацию об ориентирах множества решений в ортанте, можно извлечь из рисунка, полученного при стартовых значениях аргументов просмотра, или из списка ориентиров),
3) выбранный брус предста́вим как дополнительный аргумент `varargin`.

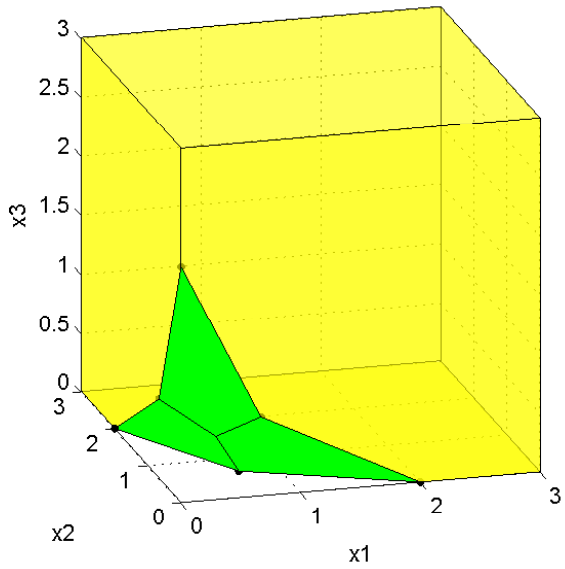
Если рисунок, полученный при стартовых значениях аргументов просмотра, вызывает сомнения в наличии/отсутствии полости у множества решений, рекомендуем следующий порядок действий, который позволит это сомнение разрешить. В пусковой функции установите значение аргумента `OrientPoints` равным 1, а значение дополнительного аргумента `varargin` варьируйте так, чтобы посмотреть на пересечение множества решений с каждым ортантом. Множество решений H имеет полость тогда и только тогда, когда для всех $k = 1, \ldots, 8$, на рисунке пересечения множества H с ортантом $\mathcal{O}_k$ выполнены оба следующих условия:

- начало координат не обозначено как ориентир,
- каждая координатная ось содержит ориентир.

Пример 16. В Примерах 12–15 каждый столбец матрицы симметричен относительно нуля, поэтому множество решений симметрично относительно каждой координатной плоскости. Чтобы убедиться в наличии/отсутствии полости у такого множества, достаточно увидеть его пересечение только с одним ортантом. Для определенности возьмем положительный ортант. В пусковой функции каждого из Примеров 12–15 установим значение аргумента `OrientPoints` равным 1 и укажем подходящий брус принудительной обрезки `varargin`:

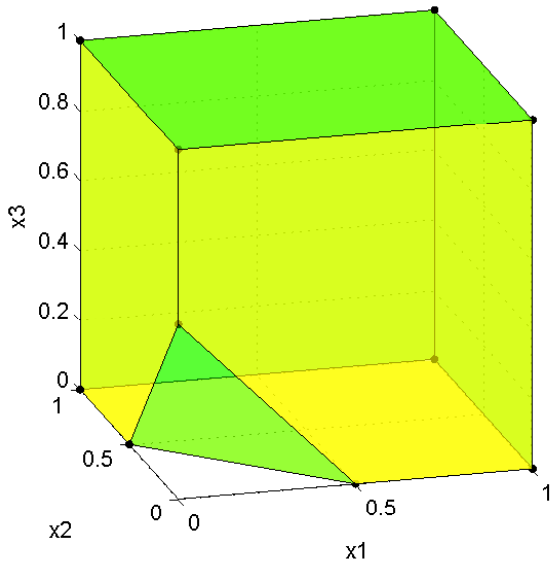
Пространство без звезды

```
EqnCtlR3(infA,supA,infb,supb,1,0,0,3,0,3,0,3);
```



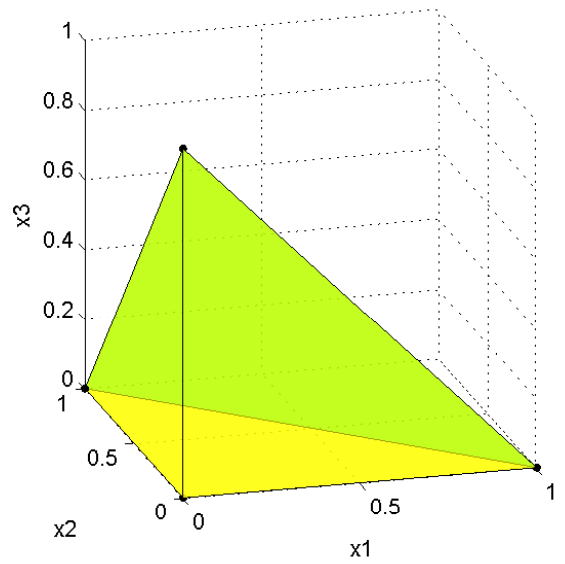
Куб без зеркальной пирамиды

```
EqnWeakR3(infA,supA,infb,supb,1,1,0,1,0,1,0,1);
```



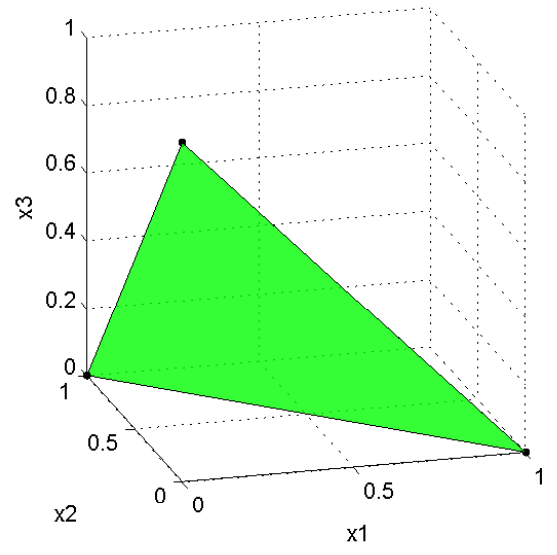
Зеркальная пирамида

```
EqnTolR3(infA,supA,infb,supb,1,1,0,1,0,1,0,1);
```



Граница зеркальной пирамиды

```
EqnAEssR3(infA,supA,Aq,infb,supb,bq,1,1,0,1,0,1,0,1);
```



Из рисунков видно, что все множества решений, кроме зеркальной пирамиды, имеют полость.

7 Порядок действий по установке и использованию пакета IntLinIncR3

1. Скачайте файл-архив в кодировке UTF8 или cp1251:
   interval.ict.nsc.ru/Programing/MCodes/IntLinIncR3_UTF8.zip,
   interval.ict.nsc.ru/Programing/MCodes/IntLinIncR3_cp1251.zip.
2. Разархивируйте его в отдельную папку.
3. Обеспечьте доступ к этой папке из MATLAB.
4. В командном окне MATLAB вводите данные для систем (1)–(8) и вызывайте пусковые функции в соответствии с этим руководством и руководством пользователя к пакету IntLinIncR2.

8 Список литературы

[1] Шарая И.А. Метод граничных интервалов и визуализация полиэдральных множеств — *готовится к печати в журнале «Вычислительные технологии»*

[2] Шарая И.А. Бескванторные описания для интервально-кванторных линейных систем // *Труды института математики и механики УрО РАН.* — 2014. — Т. 20, №2. — С. 311–323.
— (http://interval.ict.nsc.ru/sharaya/Papers/trIMM14.pdf)

[3] Шарый С.П. *Конечномерный интервальный анализ.* — http://interval.ict.nsc.ru/Library/InteBooks/SharyBook.pdf

[4] Рон И. Разрешимость систем интервальных линейных уравнений и неравенств // *Задачи линейной оптимизации с неточными данными* / Фидлер М., Недома Й., Рамик Я., Рон И., Циммерманн К. — М.–Ижевск: НИЦ «Регулярная и хаотическая динамика», Институт компьютерных исследований, 2008. — Глава 2. — С. 64–117.
— (Электронная версия книги на английском языке доступна по адресу: http://interval.ict.nsc.ru/Library/InteBooks/InexactLP.pdf)

И.А. Шарая
Институт вычислительных технологий СО РАН

1 сентября 2014 г.