Министерство образования и науки
Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования
«Новосибирский национальный исследовательский государственный университет»

Механико-математический факультет

Выпускная квалификационная работа на соискание степени бакалавра

Кафедра математического моделирования

Шабанов Артем Андреевич

Алгоритмы искусственной иммунной системы в интервальных методах глобальной оптимизации

Научный руководитель:
доктор физико-математических наук
_____ С. П. Шарый

Новосибирск, 2016 г.

Оглавление

Введение

Предметом данной работы является задача поиска глобального оптимума функции $f\colon \boldsymbol{X} \subseteq \mathbb{R}^n \to \mathbb{R}$, где $\boldsymbol{X}$ – прямоугольный брус из $\mathbb{R}^n$. Формально это можно записать следующим образом:

$$\text{найти } \min_{x \in \boldsymbol{X}} f(x). \tag{1}$$

Здесь и далее будем рассматривать постановки только на минимум, т. к. имеет место соотношение

$$\max f(x) = -\min\bigl(-f(x)\bigr). \tag{2}$$

Задача (1) является актуальной, вследствие того, что часто возникает в различных приложениях (машиностроение, приборостроение и т. д.), но т. к. является $\mathcal{NP}$-трудной, то попытки эффективного решения породили большое число различных методов, которые используют разнообразные подходы (методы спуска, методы неравномерных покрытий, интервальные методы . . .). Структура дерева на Рис. 1 показывает как можно разделить методы оптимизации между собой.

С. П. Шарым в 2005 г. было предложено использовать интервальные стохастические методы глобальной оптимизации. На текущий момент имеется небольшой ряд работ в этой области [6], [7], [10], которые показывают работоспособность и перспективность подобных алгоритмов.

Но, несмотря на полученные ранее результаты, алгоритмы этого класса развиты относительно слабо.

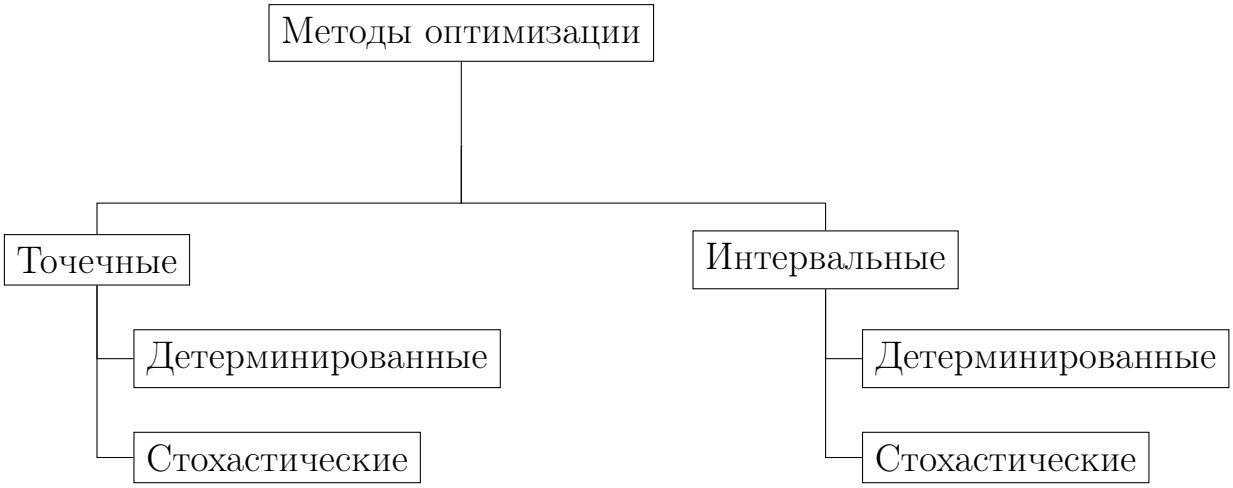


Рис. 1: Классификация методов оптимизации.

Целью данной работы является попытка исправить текущее положение и продолжить исследования в этой области, адаптировав один из известных традиционных стохастических методов глобальной оптимизации к использованию интервальных техник. В качестве такового были выбраны алгоритмы глобальной оптимизации на основе искусственных иммунных систем.

Глава 1

Интервальный анализ

1.1 Основы интервального анализа

Определение 1.1 *Замкнутый отрезок* $\boldsymbol{x} = [\,\underline{\boldsymbol{x}}, \overline{\boldsymbol{x}}\,]$ *вещественной оси* $\mathbb{R}$ *будем называть одномерным интервалом. При этом* $\underline{\boldsymbol{x}}$ *— левый (нижний),* а $\overline{\boldsymbol{x}}$ *— правый (верхний) концы интервала* $\boldsymbol{x}$*, или формально:*

$$\boldsymbol{x} = [\,\underline{\boldsymbol{x}}, \overline{\boldsymbol{x}}\,] = \{x \in \mathbb{R} \,|\, \underline{\boldsymbol{x}} \leqslant x \leqslant \overline{\boldsymbol{x}}\}.$$

Будем говорить, что интервал $\boldsymbol{x}$ *вырожден*, если $\underline{\boldsymbol{x}} = \overline{\boldsymbol{x}}$, иначе *невырожден*, вещественные числа $x \in \mathbb{R}$ будем отождествлять с вырожденным интервалом $[\,x, x\,]$. Любой интервал полностью задаётся двумя числами — своим нижним и верхним концами, но часто используются и другие характеристики интервала среди которых одними из самых важных являются *середина*

$$\operatorname{mid} \boldsymbol{x} = \tfrac{1}{2}(\underline{\boldsymbol{x}} + \overline{\boldsymbol{x}})$$

и *радиус*

$$\operatorname{rad} \boldsymbol{x} = \tfrac{1}{2}(\overline{\boldsymbol{x}} - \underline{\boldsymbol{x}}).$$

Равносильным радиусу признаком интервала будет $\operatorname{wid} \boldsymbol{x} = (\overline{\boldsymbol{x}} - \underline{\boldsymbol{x}})$, что то же самое, что и $2 \cdot \operatorname{rad} \boldsymbol{x}$, который назовём *шириной*. Некоторые определения и свойства можно переписать в терминах этих характеристик, например: $\boldsymbol{x}$ — вырожденный интервал $\iff \operatorname{wid} \boldsymbol{x} = 0$.

Помимо середины и радиуса интервал обладает множеством иных атрибутов, подробно описанных в [9]. Едва ли имеет смысл в этой работе вводить их все, поэтому рассмотрим только те, которые будем непосредственно использовать в ходе дальнейших исследований.

Определение 1.2 *Абсолютной величиной интервала $\boldsymbol{x}$ называется величина*

$$|\boldsymbol{x}| \stackrel{\text{def}}{:=} \max\{|x|, x \in \boldsymbol{x}\} = \max\{|\underline{\boldsymbol{x}}|, |\overline{\boldsymbol{x}}|\}.$$

Определение 1.3 *Вектор $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$, компонента $\boldsymbol{x}_i$ которого для любого номера $i = 1, \ldots, n$ суть одномерный интервал, назовём многомерным интервалом или брусом.*

Прежде чем продолжать, введём несколько обозначений: множество всех одномерных вещественных интервалов $\boldsymbol{x}$ будем обозначать как $\mathbb{IR}$, а всех многомерных интервалов — $\mathbb{IR}^n$.

При работе с многомерными интервалами их характеристики определяются покомпонентно, т. е. пусть $\boldsymbol{x} \in \mathbb{IR}^n$, тогда $\operatorname{rad} \boldsymbol{x}$ определяется как вектор размерности n, составленный из радиусов каждой компоненты $\boldsymbol{x}$.

Для того, чтобы формализовать операции над интервалами необходимо ввести некоторую алгебраическую систему — интервальную арифметику. Существуют несколько таких систем: классическая интервальная арифметика, арифметика Каухера, арифметика Кахана, комплексные интервальные арифметики и некоторые другие (более подробно см., например, в [9]).

Однако, наиболее распространённой является классическая интервальная арифметика.

Определение 1.4 *Алгебраическую систему* $\langle\, \mathbb{IR}, +, -, \cdot, / \,\rangle$ *у которой носителем является множество всех вещественных интервалов, а операции определяются следующим образом:*

$$\boldsymbol{x} * \boldsymbol{y} = \{x * y \,|\, x \in \boldsymbol{x},\ y \in \boldsymbol{y}\},\ \textit{где} * \in \{+, -, \cdot, /\} \tag{1.1}$$

назовём классической интервальной арифметикой.

Формула (1.1) из Определения 1.4 есть так называемый **основной принцип интервальной арифметики**, суть которого заключается в том, что все операции между интервалами проводятся «по представителям».

Выпишем теперь явные формулы для сложения, вычитания, умножения и деления интервалов в классической интервальной арифметике (вывод этих результатов в данной работе мы опустим, но заинтересованный читатель может обратиться к [9] и изучить самостоятельно):

$$\boldsymbol{x} + \boldsymbol{y} = [\,\underline{\boldsymbol{x}} + \underline{\boldsymbol{y}}, \overline{\boldsymbol{x}} + \overline{\boldsymbol{y}}\,], \tag{1.2}$$

$$\boldsymbol{x} - \boldsymbol{y} = [\,\underline{\boldsymbol{x}} - \overline{\boldsymbol{y}}, \overline{\boldsymbol{x}} - \underline{\boldsymbol{y}}\,], \tag{1.3}$$

$$\boldsymbol{x} \cdot \boldsymbol{y} = \left[\, \min\{\underline{\boldsymbol{x}}\underline{\boldsymbol{y}}, \underline{\boldsymbol{x}}\overline{\boldsymbol{y}}, \overline{\boldsymbol{x}}\underline{\boldsymbol{y}}, \overline{\boldsymbol{x}}\overline{\boldsymbol{y}}\}, \max\{\underline{\boldsymbol{x}}\underline{\boldsymbol{y}}, \underline{\boldsymbol{x}}\overline{\boldsymbol{y}}, \overline{\boldsymbol{x}}\underline{\boldsymbol{y}}, \overline{\boldsymbol{x}}\overline{\boldsymbol{y}}\} \,\right], \tag{1.4}$$

$$\boldsymbol{x}/\boldsymbol{y} = x \cdot [\,1/\overline{\boldsymbol{y}}, 1/\underline{\boldsymbol{y}}\,],\ \text{если } 0 \notin \boldsymbol{y}. \tag{1.5}$$

Следует отметить, что в классической интервальной арифметике операции вычитания и деления вводятся самостоятельно, в отличии от поля вещественных чисел $\mathbb{R}$, где они определены как обратные к сложению и умножению соответственно. В интервальной арифметике такой приём не сработает. Дело в том, что в общем случае сложение и вычитание, умножение и

деление не обратны:

$$(\boldsymbol{x} + \boldsymbol{y}) - \boldsymbol{y} \neq \boldsymbol{x}, \quad (\boldsymbol{x} \cdot \boldsymbol{y})/\boldsymbol{y} \neq \boldsymbol{x}.$$

Связано это с тем, что интервальные операции с невырожденными интервалами не сужают радиус интервала-результата. Из этого также следует, что

$$\boldsymbol{x} - \boldsymbol{x} \neq 0, \quad \boldsymbol{x}/\boldsymbol{x} \neq 1. \tag{1.6}$$

Алгебраические свойства $\mathbb{IR}$ несколько беднее чем, скажем, у поля вещественных чисел $\mathbb{R}$. Нейтральным по сложению элементом будет вырожденный интервал $[0, 0]$, по умножению — $[1, 1]$. Из (1.6) следует, что не существует обратных элементов по сложению и умножению. В общем случае не выполняется свойство дистрибутивности:

$$(\boldsymbol{x} + \boldsymbol{y})\boldsymbol{z} \neq \boldsymbol{x}\boldsymbol{z} + \boldsymbol{y}\boldsymbol{z}.$$

Таким образом алгебраическая система $\langle\, \mathbb{IR}, +, -, \cdot, / \,\rangle$ не является полем, т. к. выполняются только две аксиомы, но ряд «хороших» свойств есть:

$$\begin{aligned}
(\boldsymbol{x} + \boldsymbol{y}) + \boldsymbol{z} = \boldsymbol{x} + (\boldsymbol{y} + \boldsymbol{z}) &\text{ — ассоциативность сложения,} \\
(\boldsymbol{x}\boldsymbol{y})\boldsymbol{z} = \boldsymbol{x}(\boldsymbol{y}\boldsymbol{z}) &\text{ — ассоциативность умножения,} \\
\boldsymbol{x} + \boldsymbol{y} = \boldsymbol{y} + \boldsymbol{x} &\text{ — коммутативность сложения,} \\
\boldsymbol{x}\boldsymbol{y} = \boldsymbol{y}\boldsymbol{x} &\text{ — коммутативность умножения,} \\
(\boldsymbol{x} + \boldsymbol{y})\boldsymbol{z} \subseteq \boldsymbol{x}\boldsymbol{z} + \boldsymbol{y}\boldsymbol{z} &\text{ — субдистрибутивность.}
\end{aligned}$$

Нужно отметить, что в ряде случаев дистрибутивность тем не менее вы-

полняется:

$$(\boldsymbol{x}+\boldsymbol{y})z = \boldsymbol{x}z + \boldsymbol{y}z, \text{ если } z \text{ — вещественное число,} \tag{1.7}$$

$$(\boldsymbol{x}+\boldsymbol{y})\boldsymbol{z} = \boldsymbol{x}\boldsymbol{z} + \boldsymbol{y}\boldsymbol{z}, \text{ если } \boldsymbol{x}, \boldsymbol{y} \leqslant 0 \text{ или } \boldsymbol{x}, \boldsymbol{y} \geqslant 0. \tag{1.8}$$

$\mathbb{IR}$ частично упорядоченное по включению множество, частичный порядок на котором определяется следующим образом:

$$\boldsymbol{x} \subseteq \boldsymbol{y} \iff \underline{\boldsymbol{x}} \geqslant \underline{\boldsymbol{y}} \text{ и } \overline{\boldsymbol{x}} \leqslant \overline{\boldsymbol{y}}. \tag{1.9}$$

С порядком, заданным в (1.9) связано одно из важнейших свойств операций (1.2)–(1.5) классической интервальной арифметики — *монотонность по включению*: для любых интервалов $\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{y}_0, \boldsymbol{y}_1 \in \mathbb{IR}$ таких, что $\boldsymbol{x}_0 \subseteq \boldsymbol{x}_1$ и $\boldsymbol{y}_0 \subseteq \boldsymbol{y}_1 \Rightarrow \boldsymbol{x}_0 * \boldsymbol{y}_0 \subseteq \boldsymbol{x}_1 * \boldsymbol{y}_1$ для любой $* \in \{+, -, \cdot, /\}$.

Теперь рассмотрим значимый результат интервального анализа на котором основывается один из приёмов оценивания значений функции на брусе, и который в дальнейшем мы будем активно использовать.

Теорема 1.1 (основная теорема интервальной арифметики) *Пусть $f(x_1, \ldots, x_n)$ — рациональная функция вещественных аргументов $x_1, \ldots, x_n$ и для неё определён результат $\boldsymbol{f}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ подстановки вместо аргументов интервалов их изменения $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{IR}$ и выполнения всех операций над ними в соответствии с правилами интервальной арифметики. Тогда*

$$\{f(x_1, \ldots, x_n) \,|\, x_1 \in \boldsymbol{x}_1, \ldots, x_n \in \boldsymbol{x}_n\} \subseteq \boldsymbol{f}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n), \tag{1.10}$$

т. е. $\boldsymbol{f}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ содержит множество значений функции $f(x_1, \ldots, x_n)$ на брусе $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$.

Если выражение $f(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ содержит не больше одного вхождения каждой переменной в первой степени, то в (1.10) *вместо включения выполняется точное равенство.*

С **доказательством** теоремы и **примерами** её применения можно ознакомиться в [9].

Из основной теоремы интервальной арифметики также следует, что при интервальном оценивании имеет смысл говорить не в терминах функций, а в терминах задающих их выражений из-за их неэквивалентности. Пусть задана функция f, тогда выражение для неё будем обозначать f.

1.2 Интервальное оценивание функций

Приложение интервального анализа к задачам оптимизации базируется на возможности оценивать значения функций на брусах некоторым внешним интервалом, гарантируя, что ни одно значение функции не будет меньше нижнего конца и больше верхнего конца этого интервала. Т.е. зная интервальную оценивающую функцию $\boldsymbol{f}$ для f и некоторый брус $\boldsymbol{X}$ из области определения можем оценить область значений функции f на данном брусе и получить включение:

$$\operatorname{ran}(f, \boldsymbol{X}) \subseteq \boldsymbol{f}(\boldsymbol{X}) = \left[\underline{\boldsymbol{f}(\boldsymbol{X})}, \overline{\boldsymbol{f}(\boldsymbol{X})} \right].$$

Воспользуемся тем фактом, что для непрерывной функции $f\colon \boldsymbol{X} \to \mathbb{R}$ справедливо следующее равенство:

$$\operatorname{ran}(f, \boldsymbol{X}) = \left[\min_{x \in \boldsymbol{X}} f(x), \max_{x \in \boldsymbol{X}} f(x) \right]$$

и в результате получим включение

$$\Big[\min_{x\in \boldsymbol{X}} f(x), \max_{x\in \boldsymbol{X}} f(x)\Big] \subseteq \Big[\underline{\boldsymbol{f}(\boldsymbol{X})}, \overline{\boldsymbol{f}(\boldsymbol{X})}\Big].$$

Далее, по Определению (1.9)

$$\underline{\boldsymbol{f}(\boldsymbol{X})} \leqslant \min_{x\in \boldsymbol{X}} f(x) \text{ и } \max_{x\in \boldsymbol{X}} f(x) \leqslant \overline{\boldsymbol{f}(\boldsymbol{X})}.$$

При этом равенства достижимы при условии выполнения второй части Теоремы 1.1.

Таким образом, получив оценку области значений функции на некотором брусе $\boldsymbol{X}$, можно утверждать что мы решим и задачу оптимизации постановка которой является аналогичной тем, что рассматриваются в теории оптимизации и математическом программировании:

$$\text{найти } \min_{x\in D} f(x), \tag{1.11}$$

где $D \subseteq \mathbb{R}^n$ — множество допустимых решений, задаваемое системой ограничений (равенств или неравенств). Также будем считать, что функция f опередлена всюду на D.

В случае безусловной оптимизации, когда $D = \mathbb{R}^n$, в интервальном анализе задача (1.11) решается в следующей постановке:

$$\text{найти } \min_{x\in \boldsymbol{X}} f(x), \quad \boldsymbol{X} \text{ — прямоугольный брус из } \mathbb{IR}^n.$$

При этом наличие некоторого бруса в задаче безусловной оптимизации не влечёт за собой противоречий. Присутствие обычных двухсторонних ограничений для каждой переменной не будет вызывать никаких принципи-

альных трудностей, которые обычно возникают в задачах условной оптимизации в классической постановке. Наличие бруса допустимых решений обычно служит для ограничения области поиска (могут быть нужны положительные оптимумы и т. д.). Если же задача (1.11) не подразумевает никакого исходного бруса, то мы можем всегда взять его настолько большим, чтобы он наверняка содержал все точки глобального минимума.

В случае условной оптимизации будем считать, что задан некоторый исходный брус $\boldsymbol{X} \subseteq \mathbb{IR}^n$, и на нём при дополнительных ограничениях ищется глобальный оптимум целевой функции $f(x)$. Если ограничения, определяющие D, заданы в виде неравенства $a - x_i \leqslant 0$ или $x_i - b \leqslant 0$, то они просто определяют стороны исходного бруса $\boldsymbol{X}$. Если сторона бруса не указана явно как ограничение в виде неравенства, то станем считать её условием, ограничивающим область поиска. Подход к ограничению аналогичен задаче безусловной оптимизации. Затем, в ходе работы подобных алгоритмов (см. [8]) подбрусы, не принадлежащие области D, убираются из рассмотрения.

Мы показали, что с помощью интервального анализа можно решить любую из задач, рассматриваемых теорией оптимизации, но для этого необходимо научиться получать интервальные оценки функций на брусах.

Сейчас мы можем оценивать только рациональные выражения. Нельзя ли как-нибудь расширить наши возможности? Ведь целевые функции, возникающие в прикладных задачах, в большинстве своём выходят из класса рациональных функций. Ответ положительный.

Определение 1.5 *Интервальная функция* $\boldsymbol{f}\colon \mathbb{IR}^n \to \mathbb{IR}^m$ *называется интервальным продолжением функции* $f\colon \mathbb{R}^n \to \mathbb{R}^m$, *если* $\boldsymbol{f}(x) = f(x)$.

Определение 1.6 *Интервальная функция* $\boldsymbol{f} : \mathbb{IR}^n \to \mathbb{IR}^m$ *называется*

интервальным расширением точечной функции $f\colon \mathbb{R}^n \to \mathbb{R}^m$ *на* $D \subseteq \mathbb{R}^n$, *если*

- $\boldsymbol{f}$ *является интервальным продолжением* f,
- $\boldsymbol{f}$ *монотонна по включению:* $\boldsymbol{x} \subseteq \boldsymbol{y} \Rightarrow \boldsymbol{f}(\boldsymbol{x}) \subseteq \boldsymbol{f}(\boldsymbol{y})$.

Если $\boldsymbol{f}$ — интервальное расширение для f, то:

$$\operatorname{ran}(f, \boldsymbol{X}) \subseteq \boldsymbol{f}(\boldsymbol{X})$$

В самом деле, пусть $x \in \boldsymbol{X}$, тогда $f(x) = \boldsymbol{f}(x) \subseteq \boldsymbol{f}(\boldsymbol{X})$ в силу монотонности по включению.

Определение 1.7 *Внешняя оценивающая функция — это интервальная функция* $\boldsymbol{f} : \mathbb{IR}^n \to \mathbb{IR}^m$, *такая что*

$$\boldsymbol{f}(\boldsymbol{X}) \supseteq \operatorname{ran}(f, \boldsymbol{X}).$$

Дальнейшая цель — расширить класс выражений, которые оцениваются интервальными методами, включив в них выражения, содержащие элементарные функции.

Напомним, что *элементарыми* называются функции, которые можно получить с помощью конечного числа арифметических операций и композиций из основных элементарных функций.

Основными элементарными функциями называется некоторый фиксированный исторически сложившийся набор вещественных функций (обо-

значим как $\mathcal{EF}$), а именно

абсолютная величина или модуль: $|x|$;

степенная функция: x^α;

показательная функция: α^x;

логарифмическая функция: $\log_\alpha x$;

тригонометрические функции: $\sin, \cos$;

обратные тригонометрические функции: $\arcsin, \arccos$.

Для получения области значений функций из $\mathcal{EF}$ обычно пользуются соображениями о монотонности, а для тригонометрических функций — информацией об участках монотонности.

С тем как получаются эти оценки можно ознакомиться в [9]. Мы же просто воспользуемся фактом их существования и тем, что они являются оптимальными.

Определение 1.8 *Аналитические выражения, образованные из символов переменных, констант, четырёх арифметических операций — сложения, вычитания, умножения и деления — и элементарных функций, будем называть элементарными функциональными выражениями.*

Имеет смысл различать понятия элементарной функции и элементарного выражения из-за уже упомянутой неэквивалентности функций и функциональных выражений для интервального оценивания.

Определение 1.9 *Естественным интервальным расширением называется интервальное расширение элементарного функционального выражения, полученное в результате замены его аргументов на интервалы их из-*

менения, а арифметических операций и элементарных функций на их интервальные аналоги и расширения.

Определением выше в самом деле задаётся интервальное расширение, т. к. выполнены оба условия его определения — совпадение с функцией на точечных аргументах и монотонность по включению.

Будем обозначать естественное интервальное расширение элементарного выражения f через $\mathbf{f}_\natural$, а в случае когда вид выражения не играет роли будем говорить о естественном интервальном расширении $\boldsymbol{f}_\natural$ для точечной функции f.

В [9] строго доказано, что естественное интервальное расширение имеет первый порядок точности

$$\operatorname{dist}\left(\boldsymbol{f}_\natural(\boldsymbol{x}), \operatorname{ran}(f, \boldsymbol{x})\right) \leqslant C\|\operatorname{wid}\boldsymbol{x}\|. \tag{1.12}$$

Основной трудностью при работе с естественным интервальным расширением является «эффект зависимости» который неявно описан в основной теореме интервальной арифметики. Чтобы подробнее описать смысл этого эффекта введём несколько определений.

Определение 1.10 *Интервальной величиной называется пара* $\lfloor x, \boldsymbol{x}\rceil$, *где* x *— переменная,* $\boldsymbol{x}$ *— интервал её возможных значений. В тех случаях, где это не приводит к недоразумениям, договоримся обозначать интервальную величину* $\lfloor x, \boldsymbol{x}\rceil$ *как «*$x \in \boldsymbol{x}$*».*

Определение 1.11 *Интервальные величины* $x_1 \in \boldsymbol{x}_1, \ldots, x_n \in \boldsymbol{x}_n$ *назовём независимыми, если упорядоченный набор соответсвующих переменных* $(x_1, \ldots, x_n)$, *принимает любое значение из декартова произведения* $\boldsymbol{x}_1 \times \boldsymbol{x}_2 \times \ldots \times \boldsymbol{x}_n$. *В противном случае интервальные величины называются зависимыми.*

Пусть $x_1 \in [0,1]$, $x_2 \in [0,1]$ — независимые интервальные величины и $x_3 = x_1 * x_2$, где $* \in \{+,-,\cdot,/\}$. Тогда $x_3 \in [0,2]$ будет зависима как с $x_1 \in [0,1]$, так и с $x_2 \in [0,1]$.

Если интервальные величины $x \in \boldsymbol{x}$ и $y \in \boldsymbol{y}$ не являются независимыми, то при их сложении, вычитании и умножении формулы для интервальных арифметических операций могут давать лишь огрублённую внешнюю оценку истинной области значений результата операции [9].

Появление зависимости промежуточных интервальных величин, которые получаются при расчёте тех или иных выражений, как следствие их общего происхождения от исходных переменных и будем называть *эффектом зависимости.*

Чтобы избавиться от данного эффекта или, как минимум, минимизировать степень возрастания неопределённости, нужно построить для элементарной функции такое элементарное функциональное выражение, которое содержало бы как можно меньшее число вхождений одной и той же переменной.

Определение 1.12 *Интервальная оценивающая функция* $\boldsymbol{f}_c(\boldsymbol{X})$ *для вещественнозначной функции* $f(x)$ *имеет центрированную форму с центром* z*, если она представима как*

$$\boldsymbol{f}_c(\boldsymbol{X}) = f(z) + \sum_{i=1}^{n} \boldsymbol{g}_i(z, \boldsymbol{X})(\boldsymbol{X}_i - z_i), \tag{1.13}$$

где $\boldsymbol{g}_i(z, \boldsymbol{X})$ *— какие-то интервалы, зависящие от* z *и* $\boldsymbol{X}$*.*

Центрированные формы интервальных оценивающих функций относятся к другому подходу в получении интервальных оценок. Суть заключается в том, что мы заменяем исходное выражение которое требуется оценить на эквивалентное, но которое будет оцениваться проще, точнее. После

этого применяем идею, использованную при построении естественного интервального расширения.

Выбирать $\boldsymbol{g}_i(z, \boldsymbol{X})$ из (1.13) можно по-разному. Например, имеет место следующий вариант.

Пусть $f : \mathbb{R} \supseteq [a, b] \to \mathbb{R}$ — непрерывно-дифференцируемая функция, $\boldsymbol{X}$ — интервал из $[a, b]$, $x, x_c \in \boldsymbol{X}$. По теореме Лагранжа о конечном приращении имеем

$$f(x) - f(x_c) = f'(\xi)(x - x_c), \text{ где } \xi \in \square\{x, x_c\}.$$

Преобразуем полученное выражение

$$f(x) = f(x_c) + f'(\xi)(x - x_c)$$

и возьмём интервальное расширение правой части по $\xi \in \boldsymbol{X}$ и $x \in \boldsymbol{X}$. В итоге получим, что

$$f(x) \in f(x_c) + \boldsymbol{f}'(\boldsymbol{X})(\boldsymbol{X} - x_c),$$

где под $\boldsymbol{f}'(\boldsymbol{X})$ понимается внешняя интервальная оценка для производной функции f на $\boldsymbol{X}$.

В результате мы получили внешнюю оценивающую функцию

$$\boldsymbol{f}_{mv}(\boldsymbol{X}, x_c) := f(x_c) + \boldsymbol{f}'(\boldsymbol{X})(\boldsymbol{X} - x_c), \quad x_c \in \boldsymbol{X} \tag{1.14}$$

для области значений f на $\boldsymbol{X}$, которую назовём *дифференциальной центрированной формой*. Оказывается, что (1.14) легко обобщается на случай

функции многих переменных

$$\boldsymbol{f}_{mv}(\boldsymbol{X}, x_c) = f(x_c) + \boldsymbol{f}'(\boldsymbol{X}) \cdot (\boldsymbol{X} - x_c), \quad x_c \in \boldsymbol{X}, \tag{1.15}$$

где $\boldsymbol{X} \in \mathbb{IR}^n$, $\boldsymbol{f}'(\boldsymbol{X})$ — вектор-строка интервальных оценок частных производных $\frac{\partial f}{\partial x_i}(\boldsymbol{X})$ функции f на брусе $\boldsymbol{X}$.

Теорема 1.2 (Капрани-Мадсена) *Если $x_c = \operatorname{mid} \boldsymbol{X}$, то дифференциальная центрированная форма*

$$\boldsymbol{f}_{mv}(\boldsymbol{X}) := \boldsymbol{f}_{mv}(\boldsymbol{X}, \operatorname{mid} \boldsymbol{X}) = f(\operatorname{mid} \boldsymbol{X}) + \boldsymbol{f}'(\boldsymbol{X}) \cdot (\boldsymbol{X} - \operatorname{mid} \boldsymbol{X})$$

монотонна по включению.

С **доказательством** можно ознакомиться в [9].

Очевидно, что $\boldsymbol{f}_{mv}(x) = f(x)$. Тогда $\boldsymbol{f}_{mv}(x)$ — интервальное расширение для f.

Теорема 1.3 (Кравчика-Ноймайера) *Пусть $f : \mathbb{R}^n \supseteq D \to \mathbb{R}$, $\boldsymbol{X}$ — брус $\subseteq D$, $x_c \in D$. Предположим, что $\boldsymbol{g}$ — интервальная n-строка, такая что для любого $x \in \boldsymbol{X}$*

$$f(x) = f(x_c) + g(x - x_c)$$

с некоторой $g \in \boldsymbol{g}$. Тогда интервал $f(x_c) + \boldsymbol{g}(\boldsymbol{X} - x_c)$ содержит область значений $f(x)$ на $\boldsymbol{X}$ и верна оценка

$$\operatorname{dist}\left(f(x_c) + \boldsymbol{g}(\boldsymbol{X} - x_c), \operatorname{ran}\left(f, \boldsymbol{X}\right)\right) \leqslant 2(\operatorname{rad} \boldsymbol{g})|\boldsymbol{X} - x_c|.$$

Выводом из Теоремы 1.3 является то, что если вектор-строка $\boldsymbol{g}$ в центрированной форме вычисляется так, что $\|\operatorname{wid} \boldsymbol{g}\| \leqslant C\|\operatorname{wid} \boldsymbol{X}\|$, то центриро-

ванная форма обеспечивает второй порядок точности оценивания области значений функции.

1.3 Алгоритмическое дифференцирование

В предыдущем разделе мы построили несколько интервальных оценивающих функций для вещественнозначных функций $f : \mathbb{R}^n \to \mathbb{R}$. Одной из них была центрированная дифференциальная форма $\boldsymbol{f}_{mv} : \mathbb{IR}^n \to \mathbb{IR}$. Она обладает более высоким порядком точности. Для нас это является полезным свойством, т. к. во время работы интервальных алгоритмов оптимизации ширина брусов стремится к нулю. Однако, чтобы построить $\boldsymbol{f}_{mv}$ нужны оценки частных производных функции f на брусе. Обсудим как же можно их вычислять.

Использование *символьного* и *численного* по тем или иным причинам вызывает либо трудности в расчётах, либо даёт неудовлетворительный результат (см. [7]). Поэтому мы воспользуемся технологией *алгоритмического* дифференцирования, которая основана на разборе дерева Канторовича оцениваемого выражения.

Пусть $u = u(x)$ и $v = v(x)$ — функции переменной $x \in \mathbb{R}$. Тогда,

$$(u + v)' = u' + v',$$

$$(u - v)' = u' - v',$$

$$(uv)' = u'v + v'u,$$

$$\left(\frac{u}{v}\right)' = \frac{u'v - v'u}{v^2}.$$

Видно, что численное значение производной от суммы, разности, произведения или частного может быть вычислено на основе числовых значений

u и v и их производных.

Идея алгоритмического дифференцирования: ввести арифметику пар, которую назовём *дифференциальной арифметикой*, вида (u, u'), где u — значение выражения, а u' — значение производной, в соответствии со следующими правилами:

$$(u, u') + (v, v') = (u + v, u' + v'), \tag{1.16}$$

$$(u, u') - (v, v') = (u + v, u' - v'), \tag{1.17}$$

$$(u, u')(v, v') = (uv, u'v + v'u), \tag{1.18}$$

$$\frac{(u, u')}{(v, v')} = \left(\frac{u}{v}, \frac{u'v - v'u}{v^2}\right). \tag{1.19}$$

Затем в рациональном выражении переопределяем арифметические операции на (1.16)–(1.19), подставляем вместо переменных x и констант c пары $(x, 1)$ и $(c, 0)$ соответственно, проводим вычисления по новым правилам, получаем некоторую пару (ω, ω'), где ω — значение выражения, ω' — численное значение производной.

Чтобы распространить алгоритмическое дифференцирование на $\mathcal{EF}$ к выражениям (1.16)–(1.19) добавим ещё ряд правил:

$$(u, u')^{\alpha} = \left(u^{\alpha}, \alpha u^{\alpha-1} u'\right), \tag{1.20}$$

$$\ln(u, u') = \left(\ln u, \frac{u'}{u}\right), \tag{1.21}$$

$$\sin(u, u') = (\sin u, u' \cos u), \tag{1.22}$$

$$\cos(u, u') = (\cos u, -u' \sin u), \tag{1.23}$$

Аналогично рациональным функциям распространим технологию алгоритмического дифференцирования на выражения, которые содержат вхожде-

ния элементарных функций.

Чтобы использовать идею алгоритмического дифференцирования для интервального оценивания производных нужно взять интервальное расширение правил дифференциальной арифметики, а вычисление выражений начать с пар $(\boldsymbol{x}, 1)$, где $\boldsymbol{x}$ — интервал значений аргумента. Тогда, в результате вычислений получим пару $(\boldsymbol{\omega}, \boldsymbol{\omega}')$, где $\boldsymbol{\omega}$ — естественная интервальная оценка выражения, $\boldsymbol{\omega}'$ — интервальная оценка производной выражения на интервале $\boldsymbol{x}$.

В случае, когда оцениваемое выражение — функция многих переменных

$$u = u(x_1, \ldots, x_n), \quad v = v(x_1, \ldots, x_n),$$

то будем оперировать записями длины $(n+1)$ вида

$$(u, u_{x_1}, \ldots, u_{x_n}).$$

Начинать вычисления будем с упорядоченных кортежей

$$(x_i, 0, 0, \ldots, 0, 1, 0, \ldots, 0), \text{ 1 стоит на } (i+1)\text{-ом месте},$$

а операции будут проводиться покомпонентно в соответствии с правилами для частных производных. Например, для суммы и разности получим следующий результат:

$$(u, u_{x_1}, \ldots, u_{x_n}) \pm (v, v_{x_1}, \ldots, v_{x_n}) = \\ (u \pm v, u_{x_1} \pm v_{x_1}, u_{x_2} \pm v_{x_2}, \ldots, u_{x_n} \pm v_{x_n}),$$

Аналогично на многомерный случай обобщаются остальные операции дифференциальной арифметики.

Глава 2

Анализ методов оптимизации

2.1 Точечные методы

Напомним как формулируется задача оптимизации в традиционной постановке: пусть $D \subseteq \mathbb{R}^n$ — множество допустимых решений, определяемое системой ограничений (равенств или неравенств), f — заданная на D *целевая функция*. Требуется найти

$$f^* = \min_{x \in D} f(x). \tag{2.1}$$

Суть работы точечных методов оптимизации (как локальных, так и глобальных) заключается в построении последовательности *точек испытаний* [3], лежащих в допустимой области, так, чтобы при ограниченном числе испытаний достичь наибоее точного решения. И если набор разработанных алгоритмов этого класса позволяет достаточно быстро находить локальные минимумы целевых функций, то в случае глобальной оптими-

зации многоэкстремальных функций точечный подход сдаёт свои позиции.

Дело тут не в неполноценности точечных методов, а в сути работы подобных алгоритмов и их неспособности оценивать функцию «глобально», на целом континууме значений. Вместо этого, пользуясь какой-либо информацией о f и D, область допустимых решений заменяют на дискретный, а главное, конечный или счётный «представительный» набор точек из D. Таким образом мы сможем получить лишь оценку сверху на минимум. Для того, чтобы найти гарантированную оценку нужна будет дополнительная информация о функции (константа Липшица и т. д.).

Большая часть классических алгоритмов глобальной оптимизации основана на последовательных запусках алгоритмов спуска и всех его вариаций: методов случайного поиска, локальных методов поиска, методов переменной метрики. С их помощью из заданной начальной точки можно спуститься в точку локального минимума, в которой, по определению, значение целевой функции меньше, чем её значения в соседних точках. Однако, локальный минимум совпадает с глобальным далеко не всегда. Такое совпадение гарантировано для унимодальных функций, но проверить, является ли функция унимодальной можно лишь в исключительных случаях. В практических задачах нет никаких гарантий унимодальности целевой функции [3]. Поэтому её следует рассматривать как многоэкстремальную. Кроме того, проверка условий глобальной оптимальности [11] является трудоёмкой задачей.

Преимущество интервального анализа заключается в возможности получать двухсторонние оценки функций. В результате чего не требуется вычисления дополнительных констант и проведения вспомогательных процедур которые иногда могут оказаться сравнимыми по сложности с исходной задачей.

2.2 Интервальные методы глобальной оптимизации

Будем далее рассматривать задачу глобальной оптимизации вещественнозначной функции $f : \mathbb{R}^n \supseteq \boldsymbol{X} \to \mathbb{R}$, где $\boldsymbol{X}$ – прямоугольный брус, сформулированную следующим образом:

$$\text{найти } \min_{x \in \boldsymbol{X}} f(x). \tag{2.2}$$

Интервальные методы глобальной оптимизации опираются на знание любой из рассмотренных ранее интервальных оценивающих функций $\boldsymbol{f}$ для f и следующий факт:

$$\operatorname{dist}\big(\boldsymbol{f}(\boldsymbol{X}), \operatorname{ran}(f, \boldsymbol{X})\big) \to 0, \quad \text{при } \operatorname{wid} \boldsymbol{X} \to 0. \tag{2.3}$$

который очевидно следует из (1.12) и теоремы (1.3) и который можно использовать для построения алгоритма решения задачи (2.2).

Разобьём исходный брус $\boldsymbol{X}$ на $\boldsymbol{X}'$ и $\boldsymbol{X}''$, такие что $\boldsymbol{X} = \boldsymbol{X}' \cup \boldsymbol{X}''$. Сделать это можно, например, так

$$\boldsymbol{X}'_i = \left[\underline{\boldsymbol{X}}_i, \operatorname{mid} \boldsymbol{X}_i\right], \quad \boldsymbol{X}''_i = \left[\operatorname{mid} \boldsymbol{X}_i, \overline{\boldsymbol{X}_i}\right], \quad i = 1, \ldots, n.$$

Далее, вычисляя интервальные расширения $\boldsymbol{f}(\boldsymbol{X}')$ и $\boldsymbol{f}(\boldsymbol{X}'')$, а затем выбирая наименьший из нижних концов, получим новую оценку снизу для минимума целевой функции которая, вообще говоря, будет не хуже чем $\underline{\boldsymbol{f}(\boldsymbol{X})}$, т. к. размеры каждого из полученных подбрусов меньше чем у исходного в два раза. Каждый из полученных подбрусов по аналогичной технологии можно раздробить ещё раз и получить новую оценку для минимума, а потом ещё раз и так далее до момента, когда ширина бруса, на котором

достигается минимальная нижняя граница интервального расширения и ширина интервальной оценки станет меньше некоторой заданной величины ε (Рис. 2.1).

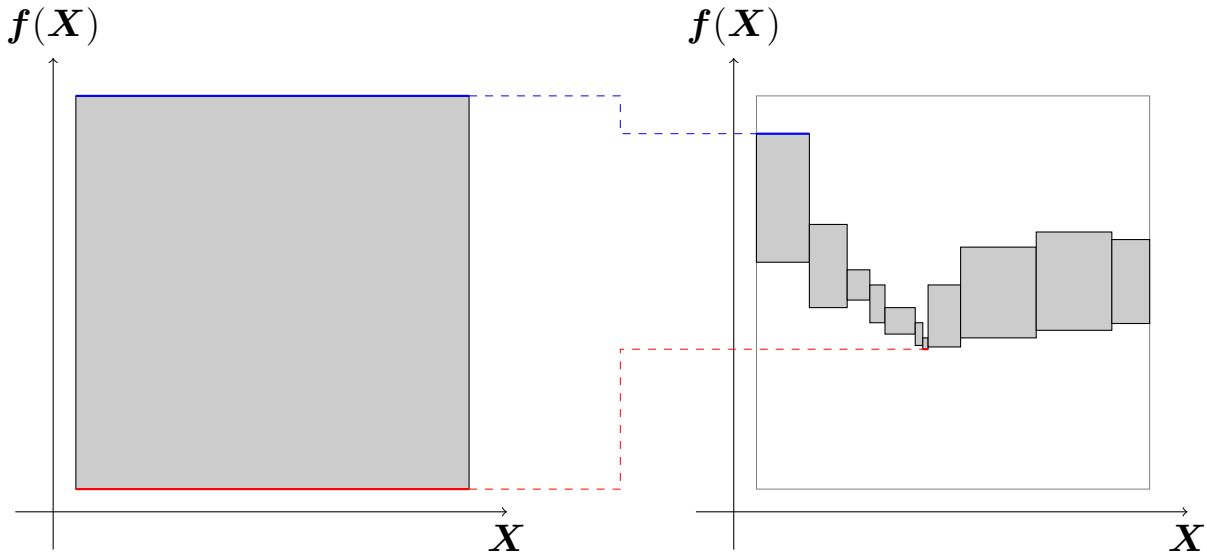


Рис. 2.1: Поиск оптимума целевой функции посредством дробления исходного бруса

Описанный приём уточнения оценки области значений функции более содержателен теоретически, нежели практически, т. к. его трудоёмкость пропорциональна 2^n и в случае реальных задач (обычно высокой размерности, [3]) потребует довольно больших затрат машинного времени.

Поэтому разумным шагом было бы ввести в алгоритм механизм адаптации, на основе которого на каждом следующем шаге будут выбираться брус и его компоненты по которым будет производиться дробление.

По большому счёту все интервальные алгоритмы глобальной оптимизации имеют общую структру и различаются лишь стратегиями дробления и выбора дробимых брусов. Ниже приведён псевдокод подобного алгоритма.

Исходные параметры:
Брус $\boldsymbol{X} \in \mathbb{R}^n$;
Интервальная оценивающая функция $\boldsymbol{f}$ для целевой функции f;
Точность $\varepsilon > 0$.
Результат:
Оценка глобального минимума $f(x)$ с точностью ε.

$\boldsymbol{Y} \leftarrow \boldsymbol{X}$;
вычислим интервальную оценку $\boldsymbol{f}(\boldsymbol{Y})$;
$y \leftarrow \underline{\boldsymbol{f}(\boldsymbol{Y})}$;
инициализируем *рабочий список* $\mathcal{L} \leftarrow \{(y, \boldsymbol{Y})\}$;
до тех пор, пока $(\operatorname{wid} \boldsymbol{f}(\boldsymbol{Y}) \geqslant \varepsilon)$ **выполнять**
    дробим брус $\boldsymbol{Y}$ на потомки $\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_m$;
    вычисляем $\boldsymbol{f}(\boldsymbol{Y}_1), \ldots, \boldsymbol{f}(\boldsymbol{Y}_m)$;
    $U_1 \leftarrow \underline{\boldsymbol{f}(\boldsymbol{Y}_1)}, \ldots, U_m \leftarrow \underline{\boldsymbol{f}(\boldsymbol{Y}_m)}$;
    удаляем из списка L ведущую запись $(y, \boldsymbol{Y})$;
    помещаем записи $(U_1, \boldsymbol{Y}_1), \ldots, (U_m, \boldsymbol{Y}_m)$ в список $\mathcal{L}$
      в порядке возрастания первого поля;
    обозначаем новую ведущую пару через $(y, \boldsymbol{Y})$;
конец
оптимальное значаение принимается за y.

Алгоритм 1: Простейший интервальный алгоритм

В Алгоритме 1 встречается понятие *рабочего списка*. Обусудим что же он из себя представляет. Рабочий список $\mathcal{L}$ — некоторая структура, в которой данные хранятся уже описанными парами. Её реализация может быть разной и сильно зависит от языка программирования и того, как хранятся данные. Договоримся, что рабочий список всегда поддерживается упорядоченным по возрастанию второго поля и вставка в список тоже упорядоченная. В MATLAB это может быть реализованно как обычный массив массивов, в C++ как связный список, элементами которого являются структуры, описывающие пары, а в Java как связный список, хранящий экземпляры класса с двумя требуемыми полями.

2.3 Рандомизированные интервальные методы глобальной оптимизации

Обычные интервальные алгоритмы глобальной оптимизации, рассмотренные в предыдущем разделе, уже гораздо лучше приспособлены к решению задач глобальной оптимизации. Тем не менее, и эти методы сталкиваются с некоторыми трудностями.

Первой из них являются целевые функции с *большим числом локальных минимумов*, слабо различающихся между собой. Глобальный минимум в свою очередь слабо отличается от локальных. В таких случаях приходится интенсивно дробить брусы, содержащие локальные минимумы [7] и получать на них интервальные оценки. Вычисление даже естественного интервального расширения занимает примерно в два раза больше времени, чем вычисление значения этого выражения в точке (интервальные операции проводятся по концам). Для более сложных интервальных оценивающих функций потребуется ещё больше времени.

Вторая проблема — *задачи большой размерности.* Мы уже говорили, что чем меньше размеры бруса, тем точнее получается на нём интервальная оценка значений функции. Однако, эффективность дробления брусов пополам с увеличением размерности снижается. Так, при дроблении двухмерного бруса по самой широкой компоненте на два подбруса его норма уменьшается на 21%, десятимерного — на 3.8%. Соответственно, оценка области значений будет улучшатся медленнее.

И последнее — *застаивание интервальной оценки* (Рис. 2.2). Когда вследствие зависимости интервальных величин или особенностей функции получаемые интервальные оценки оказываются грубыми и долгое время практически не уточняются при дроблении бруса.

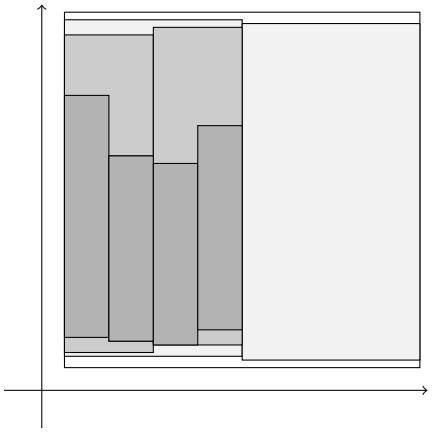


Рис. 2.2: Застаивание интервальной оценки

Возможным решением вышеперечисленных проблем могло бы стать введение рандомизации в интервальные методы. Это утверждение не является предположением. С. П. Шарый в 2005 г. предложил идею введения стохастики в интервальные алгоритмы глобальной оптимизации, и в работах [10], [6], [7] были проведены исследования и эксперименты в этом направлении.

Результаты были положительными. Удалось создать интервальные алгоритмы имитации отжига [10] и интервальный генетический алгоритм [6], которые успешно справлялись с набором тестовых целевых функций не хуже детерменированного интервального алгоритма глобальной оптимизации, а в некоторых случаях даже быстрее.

Последние достижения в этой области были получены Пановым Н. В. в работе [7], где был реализован параллельный мультиметодный интервальный алгоритм для глобальной оптимизации.

Автору данной работы было предложено продолжить исследования в этом направлении и попытаться построить другой интервальный алгоритм, взяв в качестве базового один из известных точечных рандомизированных алгоритмов, чему и будет посвящена следующая глава.

Глава 3

Интервальный алгоритм глобальной оптимизации на основе ИИС

Иммунной системой любого живого организма называется подсистема, объединяющая органы и ткани, которые защищают организм от различных заболеваний. Назначение иммунной системы живого организма заключается в том, что она идентифицирует и уничтожает чужеродные тела, попавшие в организм, и совершенствуется, накапливая опыт борьбы с ними.

Иммунная система представляет большой интерес для специалистов в области информатики, так как она является примером мощных и гибких возможностей децентрализованной обработки информации. По сути, она выполняет большой объем сложных параллельных вычислений. В настоящее время иммунные алгоритмы разрабатываются для решения множества задач в широкой области приложений, таких как задачи оптимизации, задачи распознавания образов и т.д.

3.1 Обзор искусственных иммунных систем

Определение 3.1 *Искусственная иммунная система (ИИС) — информационная технология, использующая понятия, аппарат и некоторые достижения теоретической иммунологии для решения прикладных задач.*

Модели, основанные на принципах функционирования системы иммунитета, применяются в различных областях науки и техники:

- анализ данных;
- распознавание образов;
- компьютерная безопасность;
- диагностика неисправностей технических объектов;
- оптимизация.

Методы ИИС, ориентированные на решение задачи глобальной оптимизации, вдохновлены некоторыми аспектами поведения иммунной системы человека в процессе защиты ею организма от внешних факторов (патогенов и антигенов). Защитные клетки иммунной системы (антитела) при этом претерпевают множество изменений, целью которых является создание клеток, обеспечивающих наилучшую защиту от данного фактора. В результате создается большое число различных антител, и в борьбе с патогеном побеждают те из них, которые оказались наилучшим образом приспособлены для защиты. Иммунная система человека сохраняет в своей «памяти» победившие антитела, на основании чего именно такие антитела производятся при повторном проникновении в организм схожего патогена.

При разработке методов оптимизации на основе ИИС не ставится цель в точности воспроизвести детали функционирования живых иммунных систем. Авторы таких методов ограничиваются применением лишь некоторых принципов функционирования ИИС для конструирования механизмов

оптимизации. С различными алгоритми оптимизации на основе ИИС можно ознакомиться в [2] и [4].

3.2 Интервальный алгоритм ИИС

Далее, будем использовать следующие понятия:

- антитело — брус из $\mathbb{IR}^n$;
- аффинность — величина, характеризующая полезность бруса;
- популяция — набор антител;
- клон — антитело, образованное из другого антитела полным его копированием;
- мутация — случайное дробление бруса.

Определение *аффинности* требует некоторых уточнений. Разумно полагать, что полезность должна определяться некоторым неотрицательным числом и чем она больше, тем полезнее объект для которого она определяется. В интервальных алгоритмах глобальной оптимизации больший приоритет на дробление получают брусы у которых ниже левый конец интервальной оценки. Определив аффинность $\boldsymbol{x}$ как $-\underline{\boldsymbol{x}}$ эта величина не всегда остаётся больше или равной нулю (например, для интервала $[5, 10]$). Поэтому, учитывая то, что аффинность понадобится нам только в рамках алгоритма оптимизации, и т. к. у нас заведомо имеется некоторый брус $\boldsymbol{X}$, на котором начинают работу все алгоритмы подобного сорта, то для того, чтобы сохранить неотрицательность полезности опередим её как:

$$\varphi = |\boldsymbol{f}(\boldsymbol{X})| - \underline{\boldsymbol{f}(\boldsymbol{x})}.$$

При этом справедливо, что $\varphi \geqslant 0$ для любых $\boldsymbol{X}$ и $\boldsymbol{x}$.

Приступим теперь к описанию разработанного алгоритма. В качестве базового был выбран самый простой и в то же время наиболее подходящий метод `CLONALG` [4], изначально разработанный для решения задач машинного обучения и распознавания образов. Позднее метод адаптировали для задач оптимизации.

Для алгоритмов оптимизации, использующих ИИС характерно наличие клеток памяти, которые хранят наилучшие решения на протяжении всей работы. Адаптируя `CLONALG` к интервальным методом оптимизации мы не будем использовать эту структуру и ограничимся лишь наличием рабочего списка $\mathcal{L}$.

Исходные параметры:
Брус $\boldsymbol{X} \in \mathbb{R}^n$;
Интервальная оценивающая функция $\boldsymbol{f}$ для целевой функции f;
Точность $\varepsilon > 0$.
Результат:
f^* — оценка глобального минимума $f(x)$ с точностью ε.

на исходном брусе $\boldsymbol{X}$ определяется начальная популяция;
вычисляем полезность φ_i каждого антитела $\boldsymbol{X}_i$ в популяции;
инициализируем рабочий список $\mathcal{L} \leftarrow \{(\boldsymbol{X}_i, \varphi_i)\}$;
до тех пор, пока $\operatorname{wid} \boldsymbol{f}$(ведущий брус) $\geqslant \varepsilon$ **выполнять**
    берём k антител из $\mathcal{L}$;
    создаём l клонов каждого из них;
    мутируем каждый клон соответствующего антитела;
    выбираем самый перспективный клон и заменяем им родителя в рабочем списке $\mathcal{L}$;
конец
$f^* \leftarrow \underline{\boldsymbol{f}(\text{ведущий брус})}$.

Алгоритм 2: псевдокод ИИИС

`Начальная популяция`. Выбор начальной популяции непринципиален и слабо влияет на скорость сходимости алгоритма. Поэтому имеет смысл пройти этот этап как можно быстрее. Выполнение сложных схем дроб-

ления занимает относительно много времени, поэтому ограничимся либо простым равномерным дроблением, когда брус разбивается на несколько одинаковых подбрусов меньших размеров, либо примем за начальную популяцию единственный брус $\boldsymbol{X}$.

`Параметры` k `и` l. Являются натуральными числами, изменяя которые можно управлять скоростью сходимости ИИИС и подстраивать его к различным целевым функциям. Первый параметр определяет сколько антител будут улучшаться на одном шаге алгоритма, а второй — сколько попыток будет тратиться на улучшение каждого из выбранных брусов. Представленный алгоритм сохраняет одно из важнейших свойств методов ИИС — его легко распараллелить. После выбора брусов из списка $\mathcal{L}$ действия с каждым из них (клонирование, мутация клонов, выбор самого перспективного клона) происходит независимо от других. Поэтому работу над разными брусами можно поручить разным потокам, при этом каждый из этих потоков будет проводить много сложных вычислений и наверняка затраты на запуск потока будут значительно меньше, чем нагрузка на него во время работы алгоритма.

В случае последовательного выполнения программы слишком большие величины k и l могут сильно увеличить время работы, однако общее число шагов метода наоборот сильно уменьшиться. Поэтому в таких случаях, чтобы алгоритм на захлёбывался в большом количестве сложных вычислений, следует ограничить выбор параметров относительно малыми числами. Когда есть возможность выполнять высокопроизводительные параллельные вычисления, наоборот, k и l выбираются большими. Тогда, вследствие снижения числа шагов алгоритма и одновременного выполнения большого количества вычислений на каждой итерации скорость работы алгоритма будет расти.

`Мутация`. Как мы уже говорили, когда антитела представляют собой интервальные сущности то под мутацией понимается дробление антитела. Слишком грубая интервальная оценка и эффект её застаивания не всегда позволяют обычному детерминированному дроблению быть оптимальным [7]. В данной работе было предложено и реализовано случайное квазиравномерное дробление на основе распределния Гаусса. Далее, под мутацией будем понимать именно такое дробление.

`Перспективность клонов`. Уменьшение размеров брусов списка $\mathcal{L}$ во время работы алгоритма позволяет нам улучшать оценку минимума целевой функции, всё сильнее «прижимая» её к истинному минимуму снизу. Поэтому чем больше наименьший из левых концов интервальных оценок функции на подбрусах, получающихся при мутации клонированного бруса, тем перспективнее этот клон. Формально, для выбора самого перспективного клона решается следующая задача

$$\text{найти} \max_{c\in\{1,\ldots,l\}} \min_{i\in\{1,\ldots,m\}} \underline{\boldsymbol{f}(\boldsymbol{X}^{c,i})},$$

где $\boldsymbol{X}^{c,i}$ — i-ый подбрус результата мутации клона c, m — количество подбрусов, порождаемых c-м клоном.

Представленный Алгоритм 2 включает в себя лишь некоторые базовые шаги, выполнение которых рано или поздно даст решение поставленной задачи. Тем не менее имеет смысл для ускорения данного процесса добавить дополнительные улучшения [8].

3.3 Квазиравномерное дробление

В работах [10], [6], [7] рандомизация алгоритмов заключалась, в основном, в случайном выборе элементов из рабочего списка $\mathcal{L}$. В Алгоритме 2 такой приём не подходит по построению. Поэтому пойдём другим путём и введём рандомизацию не в процедуру выбора бруса для дробления, а в процесс самого дробления.

Напомним как выглядит *распределение Гаусса* (или нормальное распределение):

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{(x-\alpha)^2}{2\sigma^2}},$$

где параметр α — среднее значение, σ — среднеквадратичное отклонение.

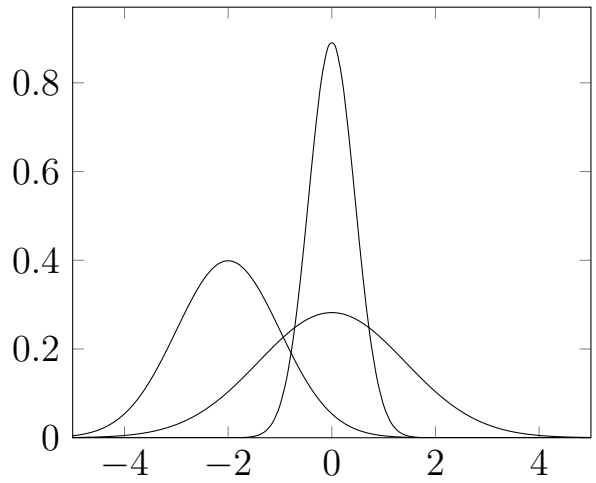


Рис. 3.1: Плотности нормального распределения

Теорема 3.1 (правило трёх σ)

Приблизительно с вероятностью 0.9973 *значение нормально распределённой случайной величины лежит в интервале* $[\,\alpha - 3\sigma, \alpha + 3\sigma\,]$.

Пусть имеется одномерный интервал $\boldsymbol{x} = [\,\underline{\boldsymbol{x}}, \overline{\boldsymbol{x}}\,]$. Мы хотим раздробить его на два подинтервала меньших размеров. Разумный способ — дробить на одномерные подбрусы одинакового размера, т. к. в этом случае во время

работы интервальных алгоритмов глобальной оптимизации размеры подбрусов стремятся к нулю (док-во см. в [9]). Была предложена следющая технология. Введём рандомизацию в процедуру дробления так, чтобы точка, относительно которой происходит дробление была расположена близко к центру с небольшим отклонением, которое является некоторой случайной величиной.

Для этой цели подходит, например, нормальное распределение, графики плотностей которого изображены на Рис. 3.1, а на Рис. 3.2 показано, как должна выглядеть плотность случайной величины, чтобы её моделируемые значения с большей вероятностью попадали в окрестность $\operatorname{mid} \boldsymbol{x}$ и не выходили за пределы самого интервала.

Нетрудно видеть, что эта картина соответствует нормальному распределению с некоторыми параметрами, которые легко определяются с использованием Теоремы 3.1. Положим $\alpha = \operatorname{mid} \boldsymbol{x}$. Воспользуемся правилом трёх сигм. Получим, что

$$3\sigma = \operatorname{rad} \boldsymbol{x} \quad \Rightarrow \quad \sigma = \frac{\operatorname{rad} \boldsymbol{x}}{3}.$$

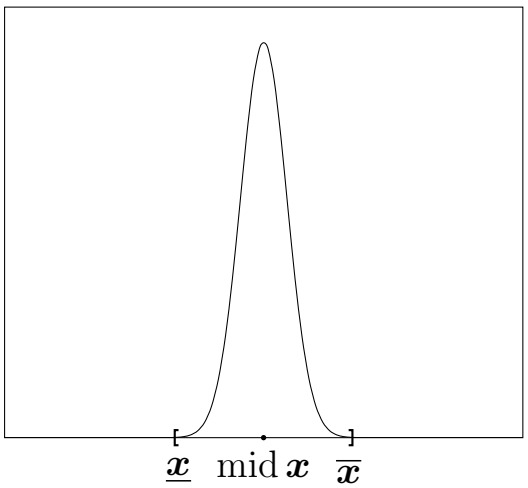


Рис. 3.2: Плотность распределения случайной величины на интервале

Теперь остаётся непосредственно провести моделирование случайной величины, реализации которой удовлетворяют требуемым вероятностным распределениям. Для этого воспользуемся теоретическими фактами из [1] и [5].

Теорема 3.2 *Пусть a, b — независимые случайные величины, равномерно распределённые на интервале $[\,0, 1\,]$. Тогда $z_0 = \cos(2\pi a)\sqrt{-2\ln b}$ и $z_1 = \sin(2\pi a)\sqrt{-2\ln b}$ — независимы и распределены нормально с математическим ожиданием $\alpha = 0$ и дисперсией $\sigma^2 = 1$.*

Теорема 3.3 *Если X имеет стандартное нормальное распределение, то величина $Y = \sigma X + \alpha$ распределена нормально с параметрами (α, σ^2).*

Таким образом, чтобы выполнить дробление интервала $\boldsymbol{x}$ «почти пополам», необходимо выполнить следующую последовательность действий:

1. Находим α и σ.
2. Моделируем число q со стандартным нормальным распределением.
3. С помощью правил преобразования случайных величин переводим в число q' с распределением с требуемыми параметрами.
4. Дробим интервал $\boldsymbol{x}$ на два потомка: $\boldsymbol{x}' = [\,\underline{\boldsymbol{x}}, q'\,]$ и $\boldsymbol{x}'' = [\,q', \overline{\boldsymbol{x}}]$.

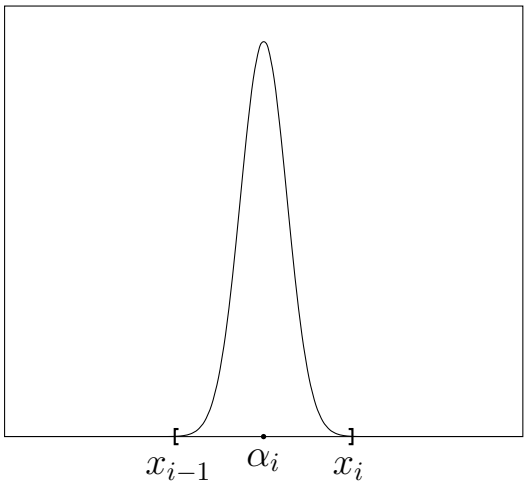
Вообще говоря, вероятность выйти за пределы $\boldsymbol{x}$ всегда останется ненулевой. Поэтому в случае розыгрыша случайной величины $a < \underline{\boldsymbol{x}}$ ($a > \overline{\boldsymbol{x}}$) примем её значение за $\underline{\boldsymbol{x}}$ ($\overline{\boldsymbol{x}}$).

Чтобы обобщить этот алгоритм на произвольное количество подинтервалов, построим на интервале $\boldsymbol{x}$ равномерную сетку $\underline{\boldsymbol{x}} = x_0 \leqslant x_1 \leqslant \ldots \leqslant$

$x_n = \overline{\boldsymbol{x}}$. Далее, для $i = 1, \ldots, n$ найдём параметры нормального распределения для каждого из интервалов $[\, x_{i-1}, x_i \,]$:

$$\alpha_i = \frac{1}{2}(x_i + x_{i-1}), \tag{3.1}$$

$$\sigma_i = \frac{x_i - \alpha_i}{3}. \tag{3.2}$$



Аналогично случаю одного интервала для каждой пары параметров (α_i, σ_i) моделируются псевдослучайные величины с нормальным распределением. Затем исходный интервал дробится относительно этих точек.

Исходные параметры:
Одномерный интервал $\boldsymbol{x}$,
Целое число n — количество подинтервалов.
Результат:
Разбиение $\boldsymbol{x}$ на n интервалов меньшего размера.

строим на $\boldsymbol{x}$ равномерную сетку $x_0 \leqslant x_1 \leqslant \ldots \leqslant x_n$;
цикл $i = 1$ **до** n **выполнять**
  вычисляем α_i и σ_i;
  моделируем соответствующую нормальную величину q_i;
конец
дробим исходный интервал относительно q_i.

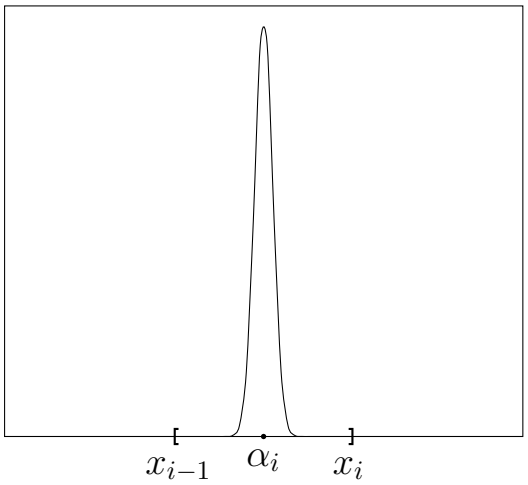
Алгоритм 3: Алгоритм квазиравномерного дробления

Такую процедуру дробления интервалов будем называть *квазиравномерным дроблением.*

В случаях, когда требуется меньший разброс точек, относительно которых происходит дробление, уменьшим среднеквадратичное отклонение точек от центра интервала $\boldsymbol{x}$. В условиях Теоремы 3.1 возьмём более широкий интервал, например $[\,\alpha - 6\sigma, \alpha + 6\sigma\,]$. Тогда (3.2) запишется как

$$\sigma_i = \frac{x_i - \alpha_i}{6}.$$

При этом получим следующую картину



Получаем выражение для σ_i, которое зависит от некотрого параметра ω, изменяя который, сможем управлять положением точек для дробления на интервале:

$$\sigma_i(\omega) = \frac{x_i - \alpha_i}{\omega}.$$

Так, например, взяв в качестве ω достаточно большое число, то реализации случайной величины будут с любым наперёд заданным ε попадать в интервал $\left[\,\mathrm{mid}\,[\,x_{i-1}, x_i\,] - \varepsilon,\ \mathrm{mid}\,[\,x_{i-1}, x_i\,] + \varepsilon\,\right]$. Дополнительно, установив параметр, отвечающий за количество брусов которые берутся из рабочего списка $\mathcal{L}$ на каждом шаге Алгоритма 2 и количество клонов, равными

единице, то получим самый простой детерменированный интервальный алгоритм глобальной оптимизации.

3.4 Реализация алгоритма ИИИС

При использовании интервального анализа и интервальных вычислений основной вклад в выбор языка программирования вносит наличие или отсутствие поддержки интервальных вычислений. На данный момент имеется ряд подобных продуктов: Intlab для MATLAB или Octave, библиотека `Boost` для C++. Начинает развиваться стандартное расширение в языке Ruby для поддержки интервальных вычислений.

По тем или иным причинам они кажутся автору неудовлетворительными. Наиболее перспективной и подходящей представляется библиотека `JInterval` для языка Java, разработанная в России группой энтузиастов Д. Ю. Надежиным, С. И. Жилиным и их учениками. Преимуществами данной библиотеки перед другими аналогами являются открытость проекта, обеспечение гибкости в выборе приоритетов вычислений (высокая производительность или высокая точность).

Программа написана на языке Java в стиле объектно-ориентированного программирования и состоит из семи Java-классов — `Main`, `Functions`, `Distribution`, `Algorithm`, `RGauss`, `Gradient`, `IntervalEvaluator`. Каждый из которых выполняет определённые задачи. Такой подход позволяет заменять части программы, не затрагивая её логику и не переписывая больших объёмов кода.

`Functions` хранит некоторый набор целевых функций, которые используются для верификации алгоритма ИИИС.

В `Distribution` и `RGauss` реализованы методы для генерации случай-

ных нормальных величин и работы с ними. Необходимость в этих классах возникла вследствие того, что стандартные библиотеки для генерации случайных величин в Java позволяют получать только значения равномерно распределённых величин. В том числе стандартных.

`Gradient` и `IntervalEvaluator` реализуют описанное ранее алгоритмическое дифференцирование.

Класс `Algorithm` хранит поля и методы, необходимые непосредственно для реализации алгоритма ИИИС.

`Main` — класс, с которого начинается выполнение любого кода на Java. Содержит только входные параметры алгоритма — исходный брус $\boldsymbol{X}$ и требуемую точность ε которые затем передаются в методы других классов, вызываемые функцией `main` из соответсвующего класса.

Листинг 3.1: Класс `Main`

```
import ...

public class Main {
  public static void main(String[] args) {
    /* set accuracy */
    ExtendedRational eps = ExtendedRational.valueOf(1.0E-9);
    /* set interval context */
    SetIntervalContext ic = SetIntervalContexts.getAccur64();
    /* set starting box */
    SetInterval[] box = ic.numsToInterval(-512,512);
    /* create object from class Algorithm */
    Algorithm alg = new Algorithm();
    /* init function for optimization */
    Function.initFunc();
    /* init distribution */
    Distribution.initFunc();
    /* start algorithm */
    ExtendedRational min = alg.start(ic,eps,box);
    /* display minimum */
    System.out.println(min.doubleValue());
  }
}
```

Глава 4

Численные эксперименты

Заключительным этапом проделанной работы будет проверка корректности результатов, которые даёт разработанный метод глобальной оптимизации и их сравнение с простейшим детерминированным интервальным алгоритмом глобальной оптимизации. Для этого будем использовать некоторый набор целевых функций, которые можно найти, например, в [12].

Алгоритм ИИИС является стохастическим и некорректно говорить о результатах только по одному его запуску. Поэтому представленные данные — это усреднённое значение сорока запусков метода с одними и теми же входными параметрами.

Целевая функция «Шестигорбый верблюд» (Рис. 4.1)

$$f(x, y) = 4x^2 - 2.1x^4 + \frac{1}{3}x^6 + xy - 4y^2 + 4y^4$$

на брусе $\boldsymbol{X} = \big([\,-5, 5\,], [\,-5, 5\,]\big)$ имеет два совпадающих глобальных минимума $f^* = -1.0316$ в двух различных точках. На Рис. 4.2 можно увидеть поведение функции вблизи глобальных минимумов.

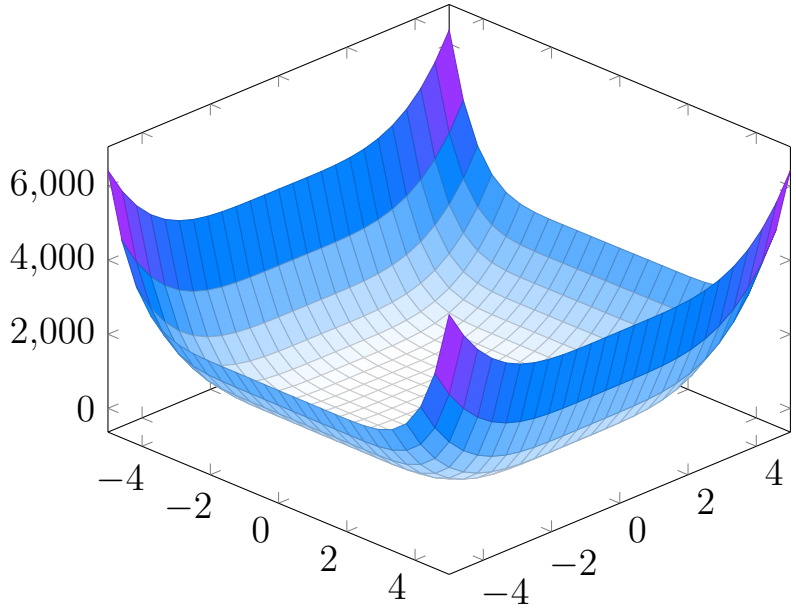


Рис. 4.1: «Шестигорбый верблюд»

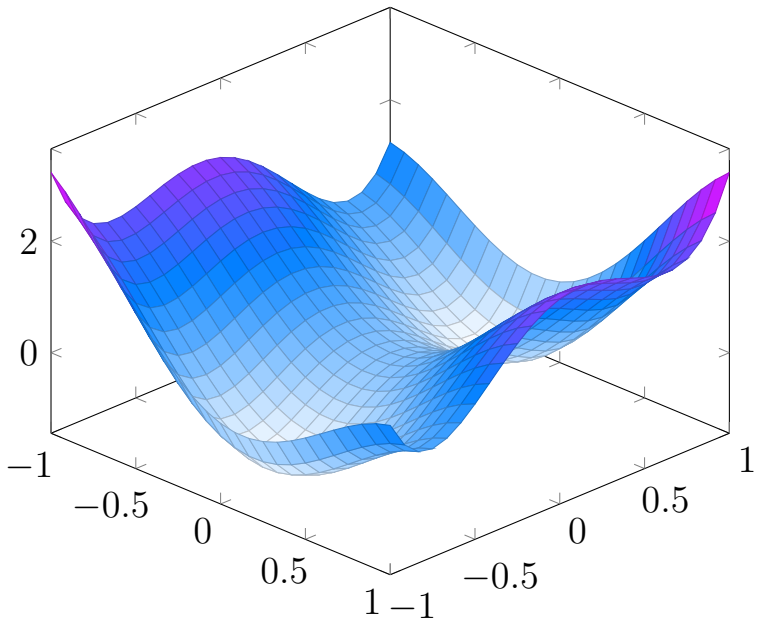


Рис. 4.2: «Шестигорбый верблюд» вблизи минимумов

Метод	Шаги	Время	Минимум
Станд. алг.	1443	0.43	-1.0316284538
ИИИС	1383	0.37	-1.0316284538

Целевая функция `Треккани` (Рис. 4.3)

$$f(x,y) = x^4 + 4x^3 + 4x^2 + y^2$$

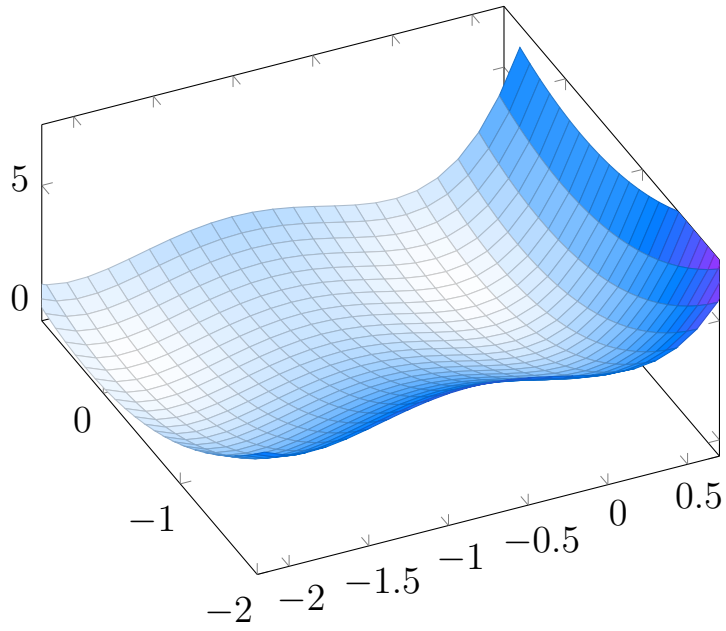


Рис. 4.3: Функция Треккани

на брусе $\boldsymbol{X} = \big([\,-5,5\,],[\,-5,5\,]\big)$ имеет локальный минимум $f^* = 0$.

Метод	Шаги	Время	Минимум
Станд. алг.	487	0.12	-9.094E-10
ИИИС	1383	0.287	-9.414E-10

Заметим, что выражение $f(x,y)$ функции Треккани является неудачным для интервального оценивания, т. к. первые три слагаемых являются зависимыми интервальными величинами. Преобразуем $f(x,y)$.

$$\mathsf{f}(x,y) = x^2(x^2+2)^2 + y^2$$

Метод	Шаги	Время	Минимум
Станд. алг.	2	0.05	0.0
ИИИС	1	0.06	0.0

Целевая функция `«Решётка для яиц»` (Рис. 4.4)

$$f(x,y) = -(y+47)\sin(\sqrt{|\frac{x}{2}+(y+47)|}) - x\sin(\sqrt{|x-(y+47)|})$$

на брусе $\boldsymbol{X} = \big([-512,512],[-512,512]\big)$ имеет глобальный минимум $f^* = -959.6407$.

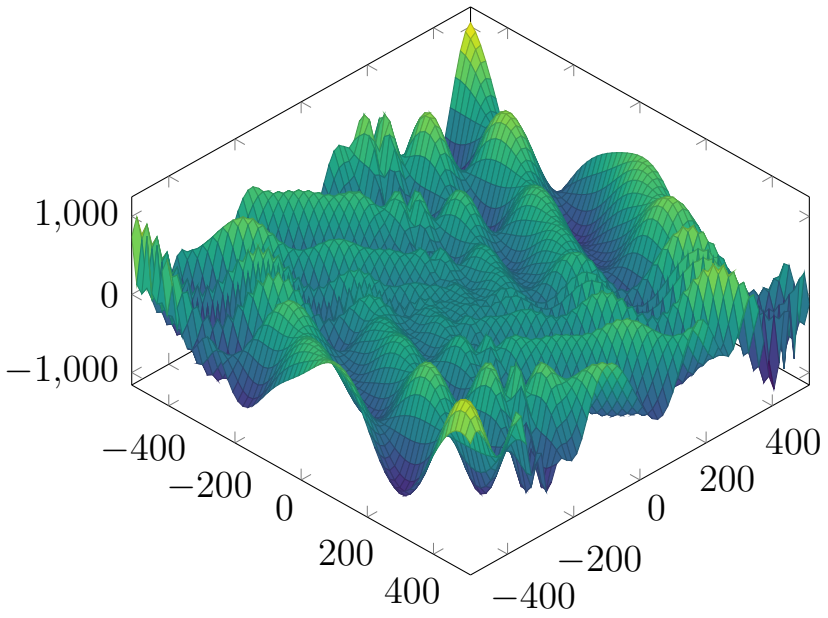


Рис. 4.4: «Решётка для яиц»

Метод	Шаги	Время	Минимум
Станд. алг.	1435483	497.0	-959.6406627212988
ИИИС	1435099	386.44	-959.6406627213928

Целевая функция `Голдстайн-Прайс` (Рис. 4.5)

$$f(x,y) = \Big(1 + (x+y+1)^2(19 - 14x + 3x^2 - 14y + 6xy + 3y^2)\Big) \times \\ \times \Big(30 + (2x-3y)^2(18 - 32x + 12x^2 + 48y - 36xy + 27y^2)\Big) \quad (4.1)$$

на брусе $\boldsymbol{X} = \big([\,-5,5\,],[\,-5,5\,]\big)$ имеет глобальный минимум $f^* = 3$.

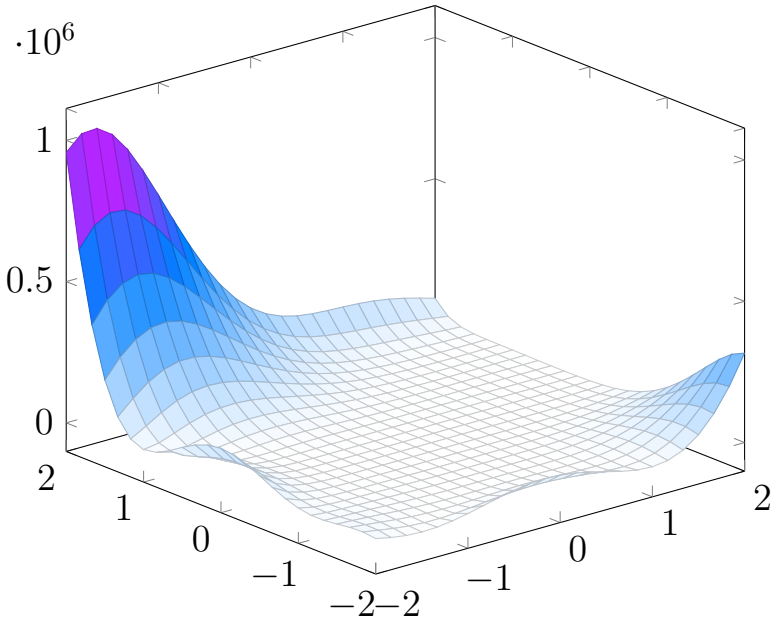


Рис. 4.5: Голдстайн-Прайс

Метод	Шаги	Время	Минимум
Станд. алг.	26005	2.19	2.999999999451421
ИИИС	25967	2.16	2.999999999166645

Тестовая функция `Розенброка` (двухмерный случай на Рис. 4.6)

$$f(x) = \sum_{i=1}^{N-1} \Big((0.8 - x_i)^2 + 100\left(x_{i+1} - x_i^2\right)^2\Big)$$

на брусе $\boldsymbol{X} = \big([\,-10,10\,],[\,-10,10\,],\ldots,[\,-10,10\,]\big) \in \mathbb{IR}^N$. Является популярной тестовой функцией для традиционных методов оптимизации, ввиду

сложности. Глобальный минимум находится в вытянутом овраге. Известно, что трудность задачи оптимизации не ограничена полиномом от длины входа, поэтому с ростом размерности сильно усложняется. Удалось посчитать значение оптимума только до $N = 6$ (результаты в таблице).

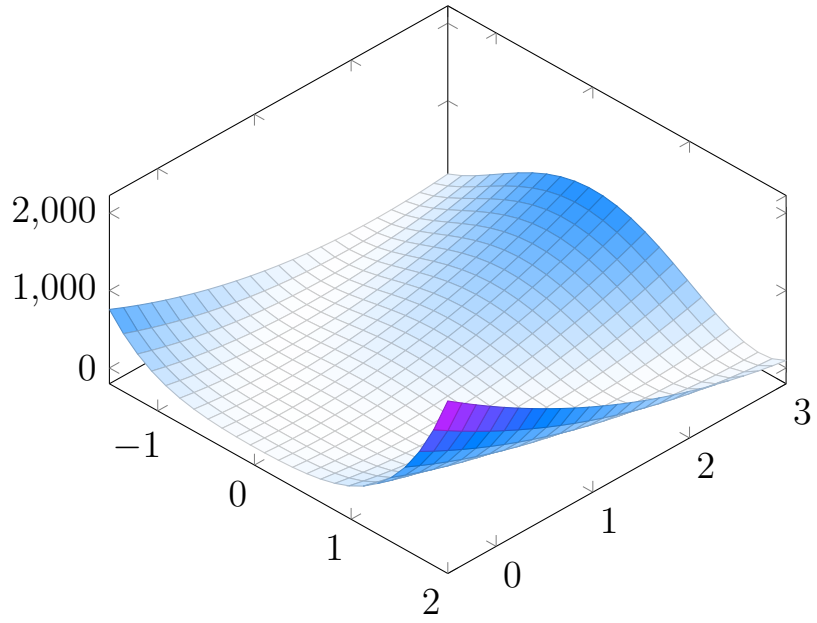


Рис. 4.6: Функция Розенброка

Метод	Шаги	Время	Минимум
Станд. алг.	4918863	156.59	0.08009663861456291
ИИИС	4533843	149.08	0.08009663789080479

Заключение

На сегодняшний день задачи глобальной оптимизации являются актуальными и появляются во многих областях науки, техники и в различных приложениях. Однако, несмотря на разнообразие традиционных методов решения этой задачи, зачастую они либо вообще не могут дать ответ (например, вследствие большой размерности задачи и большого числа вычислений), либо дают ответ плохого качества (из-за сложного рельефа функции). Использование интервального подхода в проблемах такого типа позволяет решить вторую проблему. Поэтому актуальной является разработка интервальных стохастических методов глобальной оптимизации, для которых была показана возможность более быстрой сходимости к ответу даже в задачах большой размерности.

В дипломной работе исследовалась возможность интервализации алгоритмов глобальной оптимизации на основе искусственных иммунных систем. Работа оказалась сложной, т. к. является поисковой, вследствие малого количества исследований в этой области. В результате, был разработан и представлен простейший интервальный алгоритм оптимизации на основе ИИС. Проводилось качественное сравнение простейшего интервального алгоритма ИИИС и показано, что разработанный метод успешно справляется с набором тестовых целевых функций не хуже первого. Однако, эта работа всё же ещё находится в начале своего пути. Перспективным улучшением представляются параллельные вычисления, которые легко применимы к ИИИС, вследствие особенностей алгоритмов ИИС.

В работе предложена идея случайного дробления брусов и построена процедура квазиравномерного дробления, которое в некоторых случаях помогала уменьшать время работы алгоритма.

Симбиоз инструментов интервального анализа и стохастических методов глобальной оптимизации на основе ИИС позволил создать качественно новый алгоритм для решения задачи глобальной оптимизации. Его улучшение, возможно, ускорит нахождение решения одной из востребованных проблем современной вычислительной и прикладной математики — глобальной оптимизации.

Литература

[1] Войтишек А. В. *Основы метода Монте-Карло*: Учеб. пособие / Новосиб. гос. ун-т. Новосибирск, 2010. 108с.

[2] Дасгупта Д. *Искусственные иммунные системы и их применение* / Пер. с англ. под ред. А. А. Романюхи. — М.: ФИЗМАТЛИТ, 2006. — 344с.

[3] Жилинскас А. А., Шалтянис В. Р. *Поиск оптимума: компьютер расширяет возможности.* — М.: Наука, 1989.

[4] Карпенко А. П. *Гибридный метод глобальной оптимизации на основе искусственной иммунной системы* / А. П. Карпенко, Д. Л. Шуров // Наука и образование.

[5] Лотов В. И. *Лекции по теории вероятностей*: Учеб. пособие / Новосиб. гос. ун-т. Новосибирск, 2015. 113с.

[6] Панов Н. В., Шарый С. П. *Интервальный эволюционный алгоритм поиска глобального оптимума.*

[7] Панов Н. В. *Разработка рандомизированных алгоритмов в интервальной глобальной оптимизации*: дис. канд. ф.-м. наук. НГУ, Новосибирск, 2012.

[8] Хансен Э., Уолстер Дж. У. *Глобальная оптимизация с помощью методов интервального анализа.* М.-Ижевск. НИЦ «Регулярная и хаотическая динамика», Институт компьютерных исследований , 2012.

[9] Шарый С. П. *Конечномерный интервальный анализ.*
– Электронная книга, доступная по адресу
`http://www.nsc.ru/interval/Library/InteBooks/SharyBook.pdf`

[10] Шарый С. П. *Рандомизированные алгоритмы в интервальной глобальной оптимизации* // Сиб. журн. вычисл. математики / РАН. Сиб. отд-ние. — Новосибирск, 2008.—Т.11, №4.—С. 457–474.

[11] Neumaier A. *Second-order sufficient optimality conditions for local and global nonlinear programming.*

[12] Jansen–Knuppel