DOI 10.18720/SPBPU/2/id17-1_3

# 1. VIKOR METHOD: INTERVAL AND FUZZY MODIFICATIONS

# Метод VIKOR: интервальная и нечеткая модификации

**Annotation**. *We present the classical VIKOR multi-criteria decision-making method together with two important extensions: interval VIKOR and fuzzy VIKOR. We discuss their theoretical foundations, applicability, advantages and limitations, and provide comparative insight into decision stability under uncertainty. A computational example illustrates interval and fuzzy modifications of VIKOR evaluation and ranking.*

**Keywords**. *VIKOR, MCDM, interval analysis, fuzzy sets.*

**Аннотация**. *В работе рассматривается классический метод многокритериального принятия решений VIKOR, а также две его ключевые модификации: интервальный и нечеткий VIKOR. Излагаются теоретические основы методов, области применимости, особенности и ограничения, а также проводится сравнение решений в условиях неопределенности. На иллюстративном примере демонстрируется выполнение и результаты интервальной и нечеткой модификации VIKOR.*

**Ключевые слова**. *VIKOR, многокритериальное принятие решений, интервальный анализ, нечеткие множества.*

## 1.1. Введение

Метод VIKOR (*VIseKriterijumska Optimizacija I Kompromisno Resenje*) был предложен Серафимом Оприковичем как подход к многокритериальному принятию решений, ориентированный на нахождение *компромиссного решения*, максимально близкого к идеальной точке. В отличие от методов, стремящихся к парето-оптимальным решениям, VIKOR фокусируется на ситуациях, где компромисс между критериями является предпочтительным.

Однако классическая версия метода предполагает, что значения критериев заданы в виде точных чисел. В реальных задачах часто присутствуют различные формы неопределенности: интервальные оценки показателей, экспертные оценки в форме языковых термов, нечеткие треугольные числа и др. Это привело к необходимости расширить VIKOR.

Существуют две важные модификации:

– **интервальный VIKOR**, предложенный в работе [4], адаптирующий алгоритм к критериям, представленным через интервалы;
– **нечеткий VIKOR**, описанный, например, в работах [2; 5], позволяющий работать с лингвистическими и нечеткими данными.

Сравнительный анализ различных модификаций VIKOR выполнен в статье [1], где показано, что интервальная и нечеткая модификации позволяют повысить устойчивость решений при наличии неопределенности.

Цель данной работы — систематизировать основные идеи интервальной и нечеткой модификаций VIKOR, показать отличие от классического алгоритма и продемонстрировать работу интервального и нечеткого VIKOR на примере, написанном на языке программирования python.

## 1.2. Обзор метода VIKOR

Метод VIKOR относится к классу многокритериальных подходов принятия решений и был предложен Оприковичем как способ получения компромиссного решения в ситуациях, где необходимо учитывать несколько конфликтующих критериев. Первоначально метод создавался для инженерных задач, требующих выбора оптимального варианта среди множества альтернатив с разнородными характеристиками. Основная идея метода заключается в нахождении решения, минимизирующего расстояние до идеальной точки и одновременно обеспечивающего баланс между улучшением средних показателей и уменьшением максимального отклонения. Метод основан на компромиссном решении, но отличается тем, что явно использует процедуру поиска решения, приемлемого для большинства заинтересованных сторон, что делает его особенно полезным в задачах, предполагающих наличие противоречивых критериев.

VIKOR применяется в ситуациях, где требуется учитывать не только максимальную близость альтернативы к идеальному варианту, но и степень ее доминирования. Поэтому метод получил широкое применение в инженерии, инвестиционном анализе, экологии, управлении проектами и других областях, где невозможно достичь одновременной оптимизации всех целей. Преимуществом метода является способность сочетать стратегии минимизации наихудшего отклонения и максимизации общей пользы, что позволяет получать взвешенное компромиссное решение.

### *1.2.1. Математическая постановка задачи*

Пусть задано множество альтернатив $A = \{A_1, A_2, \dots, A_m\}$, каждая из которых оценивается по набору критериев $C = \{c_1, c_2, \dots, c_n\}$. Определена матрица оценок $f_{ij}$, где $f_{ij}$ — значение альтернативы $A_i$ по критерию $c_j$. Для каждого критерия задан тип: выгодный (benefit), для которого большие значения предпочтительны, или затратный (cost), для которого предпочтительны меньшие значения. Также каждому критерию назначен весовой коэффициент $w_j$, удовлетворяющий условию

$$\sum_{j=1}^{n} w_j = 1.$$

Для каждого критерия определяются лучшие и худшие значения среди всех альтернатив. Для выгодных критериев применяются выражения

$$f_j^* = \max_i f_{ij}, \qquad f_j^- = \min_i f_{ij},$$

а для затратных критериев

$$f_j^* = \min_i f_{ij}, \qquad f_j^- = \max_i f_{ij}.$$

Затем вычисляются две меры отклонения альтернативы от идеального решения. Первая мера представляет собой взвешенную сумму отклонений:

$$S_i = \sum_{j=1}^{n} w_j \, \frac{f_j^* - f_{ij}}{f_j^* - f_j^-}.$$

Вторая мера характеризует максимальное взвешенное отклонение по одному критерию:

$$R_i = \max_{1 \leqslant j \leqslant n} \left\{ w_j \, \frac{f_j^* - f_{ij}}{f_j^* - f_j^-} \right\}.$$

Для получения компромиссного решения определяется общая мера $Q_i$:

$$Q_i = v \cdot \frac{S_i - S^*}{S^- - S^*} + (1 - v) \cdot \frac{R_i - R^*}{R^- - R^*},$$

где

$$S^* = \min_i S_i, \quad S^- = \max_i S_i, \qquad R^* = \min_i R_i, \quad R^- = \max_i R_i,$$

а параметр $v \in [0,1]$ отражает предпочтения относительно стратегии компромисса. Значение $v = 0.5$ соответствует равному учету обеих мер, приближение $v \to 1$ подчеркивает ориентацию на минимизацию суммарных отклонений, а $v \to 0$ — на минимизацию наибольшего отклонения.

Итоговое решение определяется альтернативой с минимальным значением $Q_i$, однако процедура выбора включает дополнительные условия устойчивости. Если различие между лучшими альтернативами недостаточно велико или ранжирование по метрикам $S$ и $R$ противоречит индексу $Q$, формируется не одна альтернатива, а набор компромиссных решений.

## 1.3. Интервальная модификация метода VIKOR

Как уже упоминалось в 1.2, классический метод VIKOR предполагает, что оценки всех альтернатив по каждому критерию представлены точечными значениями. Однако в реальных задачах многокритериального анализа нередко встречаются ситуации, когда значения критериев заданы неточно, являются приближенными или варьируются в некотором допустимом диапазоне. Такие ситуации типичны для инженерных, экономических и экологических задач, в которых невозможно получить точные числа из-за экспертной неопределенности, вариативности данных и измерительных ошибок.

Для учета неопределенности авторы [4] предложили интервальную модификацию метода VIKOR, в которой вместо точечных значений используются интервальные оценки

$$f_{ij} = [\,\underline{f}_{ij}, \overline{f}_{ij}\,],$$

где $\underline{f}_{ij}$ и $\overline{f}_{ij}$ обозначают нижнюю и верхнюю границы возможного значения критерия. Такой подход позволяет работать с более гибкой моделью данных и принимать решения, устойчивые к колебаниям входной информации.

Для этого метода используется интервальная арифметика:

Сложение: $[a,b] + [c,d] = [a + c,\ b + d]$,

Вычитание: $[a,b] - [c,d] = [a - d,\ b - c]$, Умножение: $[a,b] \times [c,d] = [\min S,\ \max S], \quad S = \{ac, ad, bc, bd\}.$

### *1.3.1. Математическая постановка интервального VIKOR*

Пусть задано множество альтернатив $A = \{A_1, \ldots, A_m\}$, множество критериев $C = \{c_1, \ldots, c_n\}$ и интервальная матрица оценок

$$f_{ij} = [\, \underline{f}_{ij}, \overline{f}_{ij}\, ].$$

Каждый критерий относится к типу *выгодный* или *затратный*, а веса $w_j$ удовлетворяют условию $\sum_{j=1}^{n} w_j = 1$.

Для интервальных данных определяются *интервальные идеальные* и *надирные точки*. Для выгодного критерия:

$$f_j^* = \left[\max_i \underline{f}_{ij},\ \max_i \overline{f}_{ij}\right], \qquad f_j^- = \left[\min_i \underline{f}_{ij},\ \min_i \overline{f}_{ij}\right].$$

Для затратного критерия:

$$f_j^* = \left[\min_i \underline{f}_{ij},\ \min_i \overline{f}_{ij}\right], \qquad f_j^- = \left[\max_i \underline{f}_{ij},\ \max_i \overline{f}_{ij}\right].$$

Далее необходимо определить интервальные меры отклонения. Интервальный нормализованный разрыв альтернативы $A_i$ по критерию $c_j$ определяется как

$$D_{ij} = \left[ \frac{f_j^{*\,(\mathrm{low})} - \overline{f}_{ij}}{f_j^{*\,(\mathrm{low})} - f_j^{-\,(\mathrm{high})}}, \quad \frac{f_j^{*\,(\mathrm{high})} - \underline{f}_{ij}}{f_j^{*\,(\mathrm{high})} - f_j^{-\,(\mathrm{low})}} \right],$$

где использование противоположных концов интервалов обеспечивает *наиболее пессимистичную* и *наиболее оптимистичную* оценки, согласуясь с подходом Sayadi et al.

Интервальные показатели $S_i$ и $R_i$ вычисляются как:

$$S_i = \left[ \sum_{j=1}^{n} w_j\, D_{ij}^{(\mathrm{low})}, \quad \sum_{j=1}^{n} w_j\, D_{ij}^{(\mathrm{high})} \right],$$

$$R_i = \left[ \max_j w_j\, D_{ij}^{(\mathrm{low})}, \quad \max_j w_j\, D_{ij}^{(\mathrm{high})} \right].$$

### *1.3.2. Особенности метода*

Интервальный VIKOR сохраняет структуру исходного метода, но адаптирует все вычисления к интервальной арифметике. Основные особенности:

- вместо точек используются интервалы значений, что делает метод чувствительным к неопределенности;
- нормализация использует “наиболее широкое” сочетание концов интервалов, что гарантирует корректную оценку в условиях неполных данных;
- результаты ранжирования также представляются интервалами и требуют процедуры сравнения интервальных чисел;
- метод может порождать частичные или нестрогие порядки, когда интервалы $Q_i$ альтернатив пересекаются.

### *1.3.3. Недостатки и ограничения*

Несмотря на преимущества, интервальная модификация имеет ряд недостатков:

- усложнение вычислений из-за необходимости поддерживать интервальную арифметику;
- увеличение числа случаев, когда альтернативы оказываются несравнимыми из-за пересечения интервалов;
- зависимость качества результата от корректного задания интервалов экспертами.

Тем не менее модификация доказала свою эффективность в задачах с высокой неопределенностью и стала основой для дальнейших расширений метода, включая fuzzy модификацию [2; 5], о которой пойдет речь в 1.4.

### *1.3.4. Реализация интервальной модификации*

Для практической демонстрации интервальной модификации метода VIKOR был использован Python-код, реализующий шаги, описанные в работе [4].

Каждая альтернатива $A_i$ задается совокупностью интервальных оценок $f_{ij} = [\underline{f}_{ij}, \overline{f}_{ij}]$, веса критериев удовлетворяют $\sum_j w_j = 1$, а тип критерия $c_j$ определяется значением 1 (выгода) или $-1$ (затраты).

### *1.3.4.1. Исходные данные*

В качестве примера рассматриваются четыре альтернативы и три критерия. Интервальная матрица оценок имеет вид:

$$F = \begin{pmatrix} [0.7,0.9] & [0.6,0.8] & [0.8,1.0] \\ [0.5,0.7] & [0.8,1.0] & [0.6,0.8] \\ [0.8,1.0] & [0.5,0.7] & [0.7,0.9] \\ [0.6,0.8] & [0.7,0.9] & [0.5,0.7] \end{pmatrix}.$$

Веса критериев:

$$w = (0.3,\ 0.4,\ 0.3).$$

Типы критериев:

$$\text{types} = (1,\ 1,\ -1),$$

то есть первые два критерия — выгодные, третий — затратный.

### *1.3.4.2. Промежуточные вычисления*

**(a) Интервальные идеальные и надирные точки** Для каждого критерия вычисляются

$$f_j^* = [\max_i \underline{f}_{ij},\ \max_i \overline{f}_{ij}], \qquad f_j^- = [\min_i \underline{f}_{ij},\ \min_i \overline{f}_{ij}]$$

для выгодных критериев, и аналогично с инверсией минимума/максимума — для затратных.

*Таблица 1.1*

**Идеальная и надирная точки интервалов**

| Критерий | $f_j^*$ | $f_j^-$ |
|---|---|---|
| 1 | [0.800, 1.000] | [0.500, 0.700] |
| 2 | [0.800, 1.000] | [0.500, 0.700] |
| 3 | [0.500, 0.700] | [0.800, 1.000] |

**(b) Интервальные отклонения $D_{ij}$** Используются формулы интервальной нормализации:

$$D_{ij}^{(\text{low})} = \frac{f_j^{*(\text{low})} - \overline{f}_{ij}}{f_j^{*(\text{low})} - f_j^{-(\text{high})}}, \qquad D_{ij}^{(\text{high})} = \frac{f_j^{*(\text{high})} - \underline{f}_{ij}}{f_j^{*(\text{high})} - f_j^{-(\text{low})}}.$$

Полученные интервалы $D_{ij}$:

$$D_{\text{low}} = \begin{pmatrix} 0.0 & 0.0 & 1.0 \\ 1.0 & 0.0 & 0.6 \\ 0.0 & 1.0 & 0.8 \\ 0.0 & 0.0 & 0.4 \end{pmatrix}, \qquad D_{\text{high}} = \begin{pmatrix} 0.6 & 0.8 & 1.0 \\ 1.0 & 0.4 & 0.0 \\ 0.4 & 1.0 & 0.0 \\ 0.8 & 0.6 & 0.0 \end{pmatrix}.$$

**(c) Интервальные агрегированные показатели $S_i$ и $R_i$**

$$S_i = \Big[\sum_j w_j D_{ij}^{(\text{low})},\ \sum_j w_j D_{ij}^{(\text{high})}\Big], \qquad R_i = \Big[\max_j w_j D_{ij}^{(\text{low})},\ \max_j w_j D_{ij}^{(\text{high})}\Big].$$

$$S = \begin{pmatrix} [0.300, 0.800] \\ [0.480, 0.460] \\ [0.640, 0.520] \\ [0.120, 0.480] \end{pmatrix}, \qquad R = \begin{pmatrix} [0.300, 0.320] \\ [0.300, 0.300] \\ [0.400, 0.400] \\ [0.120, 0.240] \end{pmatrix}.$$

**(d) Интервальные значения $Q_i$**

$$Q_i = v\, Q_i^{(S)} + (1 - v)\, Q_i^{(R)}, \qquad v = 0.5.$$

Полученные интервалы:

$$Q = \begin{pmatrix} [0.000, 0.857] \\ [0.243, 0.571] \\ [1.000, 0.794] \\ [0.000, 0.479] \end{pmatrix}.$$

## *3. Результаты*

Поскольку в работе [4] не предлагается строгого правила упорядочивания интервальных значений $Q_i$, в настоящей реализации используется ранжирование по центрам интервалов $Q_i^{\text{center}} = \frac{1}{2}\big(Q_i^{(\text{low})} + Q_i^{(\text{high})}\big)$. Такой подход согласуется с классической версией метода VIKOR, где используются скалярные значения $Q_i$, и обеспечивает нейтральный выбор внутри интервала без предпочтения пессимистичного или оптимистичного сценария.

*Таблица 1.2*

**Интервальные метрики $S$, $R$ и $Q$ для альтернатив**

| Альтернатива | $S_i$ | $R_i$ | $Q_i$ | $Q_i^{\text{center}}$ |
|---|---|---|---|---|
| A1 | [0.300, 0.800] | [0.300, 0.320] | [0.000, 0.857] | 0.429 |
| A2 | [0.480, 0.460] | [0.300, 0.300] | [0.243, 0.571] | 0.407 |
| A3 | [0.640, 0.520] | [0.400, 0.400] | [1.000, 0.794] | 0.897 |
| A4 | [0.120, 0.480] | [0.120, 0.240] | [0.000, 0.479] | 0.239 |

*Таблица 1.3*

**Ранжирование альтернатив по метрике $Q$**

| Ранжирование | Лучшая альтернатива |
|---|---|
| A4 → A2 → A1 → A3 | A4 |

## 1.4. Нечеткая (fuzzy) модификация метода VIKOR

В классическом подходе оценочные матрицы и веса являются точными числами, интервальная работает с интервалами. Нечеткая (fuzzy) модификация VIKOR расширяет эти подходы и использует нечеткие числа (лингвистические оценки) для описания неясных, неточных или субъективных данных. Например, эксперт может оценивать критерий как Высокий, Средний или Низкий, эти термины аппроксимируются нечеткими числами.

Нечеткое число – это обобщение обычного (точного) числа, которое позволяет учитывать неопределенность или размытость в значении. Иначе говоря, нечеткое число – это число, у которого нет одной точной величины, но есть область возможных значений с разной степенью уверенности. Выделяют треугольные и трапециевидные нечеткие числа. Треугольные нечеткие числа $\tilde{f} = (l,m,h)$, где $l \leqslant m \leqslant h$, наиболее популярны на практике: они просты в использовании и часто задают лингвистические шкалы (см. табл. 1.4). Треугольное число задается тройкой $(l,m,h)$ с монотонным ростом функции принадлежности: $l$ – нижняя граница, $m$ – среднее пиковое значение (наиболее вероятное), $h$ – верхняя граница. Выделяют также трапециевидное число, которое задается аналогично, но имеет два средних значения, задающих пиковый промежуток.

*Таблица 1.4*

**Лингвистическая шкала.**

| Лингвистическая оценка | Нечеткое представление |
|---|---|
| Очень Плохо | (0, 0, 1) |
| Плохо | (0, 1, 3) |
| Средне | (1, 3, 5) |
| Хорошо | (3, 5, 7) |
| Очень хорошо | (5, 7, 9) |
| Отлично | (7, 9, 10) |

Благодаря нечеткости метод может учитывать разброс мнений, повышает устойчивость результатов при неопределенности и близок к реальным условиям принятия решений. Для удобства дальше будем работать с треугольными числами.

Для треугольных чисел выделяют следующие математические операции:

Сумма треугольных чисел: $\sum_{i=1\oplus}^{n} \tilde{N}_i = (\sum_{i=1}^{n} l_i, \sum_{i=1}^{n} m_i, \sum_{i=1}^{n} h_i)$.

Сумма треугольного числа и скаляра: $\tilde{N} \oplus K = (l + K, m + K, h + K)$.

Вычитание: $\tilde{N}_1 \ominus \tilde{N}_2 = (l_1 - r_2, m_1 - m_2, r_1 - l_2)$.

Вычитание скаляра: $\tilde{N} - K = (l - K, m - K, h - K)$.

Умножение на скаляр: $K \times \tilde{N} = (K \times l, K \times m, K \times h)$, для $K \geqslant 0$

Умножение: $\tilde{N}_1 \otimes \tilde{N}_2 = (l_1 \times l_2, m_1 \times m_2, h_1 \times h_2)$.

Деление на скаляр: $\frac{\tilde{N}}{K} = (\frac{l}{K}, \frac{m}{K}, \frac{h}{K})$.

Оператор MAX: $\underset{i}{MAX}\tilde{N}_i = (\underset{i}{max} l_i, \underset{i}{max} m_i, \underset{i}{max} h_i)$.

Оператор MIN: $\underset{i}{MIN}\tilde{N}_i = (\underset{i}{min} l_i, \underset{i}{min} m_i, \underset{i}{min} h_i)$.

### *1.4.1. Алгоритм Fuzzy VIKOR*

Алгоритм Fuzzy VIKOR расширяет классический, заменяя арифметику на нечеткую и вводя шаг дефаззификации (defuzzification). Основные этапы следующие:

A. Формирование нечеткой матрицы. Эксперты дают лингвистические оценки альтернатив по каждому критерию и/или важности критериев. Эти оценки переводятся в нечеткие числа $\tilde{f}_{ij} = (a_{ij}, b_{ij}, c_{ij})$ (например, через заранее установленную таблицу соответствия).

B. Определение весов критериев. Веса $w_j$ либо даны как четкие, либо также как нечеткие числа (например, через консенсус экспертов). Веса нормализуют или агрегируют (методом комплексного среднего, разрежение агрегации и т.п.) в одну нечеткую величину для каждого критерия.

C. Нечеткие идеальная и надирная точки. Для каждого критерия $j$ вычисляют нечеткую идеальную (наилучшую) $\tilde{f}_j^*$ и надирную (наихудшую) $\tilde{f}_j^\circ$.
$\tilde{f}_i^* = \underset{j}{MAX}\tilde{f}_{ij}$, $\tilde{f}_j^\circ = \underset{j}{MIN}\tilde{f}_{ij}$ при максимизации критерия.
$\tilde{f}_i^* = \underset{j}{MIN}\tilde{f}_{ij}$, $\tilde{f}_j^\circ = \underset{j}{MAX}\tilde{f}_{ij}$ при минимизации критерия.

D. Нормализация (нечеткие разности). Вычисляют разности для каждой альтернативы $i$ по каждому критерию $j$:
$\tilde{d}_{ij} = (\tilde{f}_i \ominus \tilde{f}_{ij})/(h_i^* - l_i^\circ)$ при максимизации,
$\tilde{d}_{ij} = (\tilde{f}_{ij} \ominus \tilde{f}_i)/(h_i^\circ - l_i^*)$ при минимизации,

E. Вычисление S и R (нечеткие показатели). Для каждой альтернативы $j$ вычисляются две агрегированные нечеткие величины:
$\tilde{S}_j = \sum_{i=1\oplus}^{n}(\tilde{w}_i \otimes \tilde{d}_{ij})$
$\tilde{R}_j = \underset{i}{MAX}(\tilde{w}_i \otimes \tilde{d}_{ij})$ где $\tilde{S}$ выражает групповую полезность, $\tilde{R}$ – индивидуальное сожаление.

F. Вычисление Q (нечеткий индекс). Аналогично классическому варианту вычисляют нечеткий индекс компромисса:
$\tilde{Q}_j = \mu(\tilde{S}_j \ominus \tilde{S}^*)/(S^{\circ h} - S^{*l}) \oplus (1-\mu)(\tilde{R}_j \ominus \tilde{R}^*)/(R^{\circ h} - R^{*l})$,
где $\tilde{S}^* = \underset{j}{MIN}\tilde{S}_j$, $\tilde{S^{\circ h}} = \underset{j}{max}\tilde{S}_j^h$, $\tilde{R}^* = \underset{j}{MIN}\tilde{R}_j$, $\tilde{R^{\circ h}} = \underset{j}{max}\tilde{R}_j^h$, $\mu$ задается пользователем VIKOR и выражает компромисс между стратегиями S и R.

G. Core ranking. На данном этапе можно произвести первое ранжирование (core ranking) по среднему значению $Q^b$.

H. Fuzzy ranking. Для нечеткого ранжирования (fuzzy ranking) нужно подтвердить ранжирование из предыдущего пункта. Для того, чтобы подтвердить ранг i-го объекта нужно, чтобы для любого j-го объекта с рангом меньше i-го выполнялось условие $\tilde{Q}_i^l \leqslant \tilde{Q}_j^l \& \tilde{Q}_i^m \leqslant \tilde{Q}_j^m \& \tilde{Q}_i^h \leqslant \tilde{Q}_j^h$.

I. Дефаззификация. Чтобы сравнивать альтернативы, преобразуют нечеткие $\tilde{S}_i$,$\tilde{R}_i$,$\tilde{Q}_i$ в обычные числовые значения. Обычно используют центроиду треугольного числа: для $\tilde{Q}_i = (l, m, h)$ задают $Q_i^{\text{cr}} = (l + m + h)/3$. Также может использоваться формула $Q_i^{\text{cr}} = (l + 2m + h)/4$. Аналогично получают $S_i^{\text{cr}}$,$R_i^{\text{cr}}$. Этот этап вводит дополнительную процедуру: разные методы дефаззификации могут дать чуть разные ранги, поэтому его выбор влияет на результат.

J. Ранжирование альтернатив. Альтернативы упорядочивают по возрастанию дефаззифицированных значений $S_i^{\text{cr}}$, $R_i^{\text{cr}}$, $Q_i^{\text{cr}}$. Мера $Q^{\text{cr}}$ считается основной: чем меньше $Q_i^{\text{cr}}$, тем ближе альтернатива к идеальному.

K. Выбор компромиссного решения. По тем же правилам, что и в классическом VIKOR: выбирают альтернативу с минимальным $Q_i^{\text{cr}}$, если выполняются условия достаточного преимущества и стабильности. То есть требуется, чтобы разница $Q_{(2)}^{\text{cr}} - Q_{(1)}^{\text{cr}}$ превысила порог $1/(m-1)$ ($m$ – число альтернатив) и лучшая альтернатива была хороша хотя бы по $S$ или $R$. В противном случае выводят несколько компромиссных решений аналогично классической схеме.

Дополнительно может вычисляться уступка (trade-off). Для этого вычисляется значение $tr_i = (D_k * w_i)/(D_i * w_k)$, где $i$ - индекс текущего критерия, $k$ - задаваемый индекс, $i \neq k$. $D_i = f_i^{*h} - f_i^{\circ l}$ для максимизируемого критерия, $D_i = f_i^{\circ h} - f_i^{*l}$ для минимизируемого критерия. Уступка может устанавливаться вручную. Далее вычисляются новые веса по формуле $w'_i = |(D_i * w^{\text{cr}} * tr_i)/D_k|$. После этого пункты D-K повторяются и определяется новый ранг в соответствии с уступкой.

### *1.4.2. Реализация fuzzy модификации на Python*

Для практической демонстрации нечеткой модификации метода VIKOR реализован класс `FuzzyVIKOR`, описывающий процедуру, предложенную в работе Opricovic (2011) [3]. Ниже перечислены ключевые шаги реализации и их соответствие формальному описанию метода.

A. Представление данных: каждая альтернатива $A_i$ задается набором треугольных нечетких чисел (TFN) по каждому критерию. Каждое число – массив длины 3.

Входные данные в коде представлены как трехмерный массив `matrix` формы $(m,n,3)$, где m – число альтернатив, n – число критериев.

B. Идеальная и надирная точки: для каждого критерия вычисляются покомпонентно:

```
ideal_point = np.zeros((n, 3))
nadir_point = np.zeros((n, 3))
for j in range(n):
for k in range(3):
if benefit_criteria[j]:
  ideal_point[j, k] = np.max(matrix[:, j, k])
  nadir_point[j, k] = np.min(matrix[:, j, k])
  else:
  ideal_point[j, k] = np.min(matrix[:, j, k])
  nadir_point[j, k] = np.max(matrix[:, j, k])
```

C. Нормализация нечетких расстояний: для каждой альтернативы и критерия формируется тройка компонент.

```
normalized = np.ndarray((m, n, 3))
for i in range(m):
  for j in range(n):
    for k in range(3):
    if benefit_criteria[j]:
      normalized[i, j, k] = (ideal_point[j, k] -
         matrix[i, j, 2 - k]) / (ideal_point[j, 2]
          - nadir_point[j, 0])
    else:
      normalized[i, j, k] = (matrix[i, j, k] -
         ideal_point[j, 2 - k]) / (nadir_point[j,
         2] - ideal_point[j, 0])
```

В коде нормализованные значения сохраняются в массиве `normalized`.

D. Вычисление нечетких $S$ и $R$: покомпонентное умножение нечетких весов $\tilde{w}_j$ и нормализованных расстояний:

```
S_values = np.zeros((m, 3))
R_values = np.zeros((m, 3))

for i in range(m):
```

```
    for k in range(3):
      s_val = 0
      r_val = -np.inf
      for j in range(n):
        cur_val = weights[j, k] * normalized[i, j, k
          ]
        s_val += cur_val
        if (cur_val > r_val):
          r_val = cur_val
      S_values[i, k] = s_val
      R_values[i, k] = r_val
```

В реализации это массивы S_values и R_values размерности $(m,3)$.

E. Вычисление $Q$:

```
S_star = np.min(S_values, axis=0)   # array [
   Sl_min, Sm_min, Sr_min]
S_bar_r = np.max(S_values[:, 2])    # scalar = max
    over Sr
R_star = np.min(R_values, axis=0)
R_bar_r = np.max(R_values[:, 2])

# 4. Расчет Q
print("4. Расчет значений Q...")
Q_values = np.zeros((m, 3))
for i in range(m):
for k in range(3):
S_part = (S_values[i, k] - S_star[2 - k]) / (
   S_bar_r - S_star[0])
R_part = (R_values[i, k] - R_star[2 - k]) / (
   R_bar_r - R_star[0])
Q_values[i, k] = v * S_part + (1 - v) * R_part
```

F. Дефаззификация: треугольные числа переводятся в скаляры:

```
crisp_Q = np.zeros(m)
crisp_R = np.zeros(m)
crisp_S = np.zeros(m)
for i in range(m):
  crisp_Q[i] = (Q_values[i, 0] + 2 * Q_values[i,
     1] + Q_values[i, 2]) / 4
  crisp_R[i] = (R_values[i, 0] + 2 * R_values[i,
     1] + R_values[i, 2]) / 4
  crisp_S[i] = (S_values[i, 0] + 2 * S_values[i,
     1] + S_values[i, 2]) / 4
```

G. В коде реализована проверка условий *acceptible advantage* и *stability* и механизм *trade-off* (уступки) с пересчетом весов и повторным выполнением шагов агрегации и ранжирования.

### *1.4.2.1. Замечания по параметрам реализации*

- `weights` в реализации ожидаются в виде TFN весов формы $(n,3)$. Если исходные веса заданы как скаляры, их можно расширить до TFN, положив все три компоненты равными.
- Параметр $v \in [0,1]$ задает стратегию компромисса (в коде — аргумент `v`).
- Механизм уступки (trade-off) в коде позволяет задать либо вектор `tr_custom_values`, либо вычислить его автоматически, после чего пересчитываются веса и алгоритм прогоняется повторно.

Алгоритм был запущен на тестовом наборе (6 альтернатив, 4 критерия, детали см. исходный `.ipynb`). Для данного прогона получены следующие истинные идеальные и надирные точки (покомпонентно, L/M/U):

*Таблица 1.5*

**Идеальная и надирная точки (TFN) для примера**

| Критерий $j$ | $\tilde{f}_j^*$ | $\tilde{f}_j^-$ |
|---|---|---|
| 1 (затратный) | (20.00, 21.06, 24.00) | (44.54, 46.89, 56.27) |
| 2 (выгодный) | (3.26, 4.08, 4.08) | (2.25, 2.50, 2.62) |
| 3 (затратный) | (6.00, 6.00, 6.00) | (60.00, 62.00, 68.00) |
| 4 (затратный) | (0.00, 0.00, 0.00) | (10.00, 10.00, 10.00) |

Из прогона выведены итоговые показатели (дефаззифицированные) и решение:

- Допустимое превосходство (acceptable_advantage): **True**.
- Допустимая стабильность (acceptable_stability): **True**.
- Компромиссное решение: **альтернатива A3**.

#### 1.4.2.2. Краткий анализ результатов

- Модификация корректно обрабатывает неопределенность оценок (TFN) — итоговое ранжирование учитывает и средние, и крайние оценки альтернатив.
- Возможность задания `tr_custom_values` и перерасчета весов позволяет моделировать различные сценарии trade-off и смотреть, как они влияют на итоговый компромисс.
- Важно учитывать метод дефаззификации: переход от TFN к скалярам существенно влияет на $S_i, R_i, Q_i$.

#### 1.4.2.3. Псевдокод реализованного алгоритма (кратко)

A. Ввести $\tilde{f}_{ij}$ и $\tilde{w}_j$ (TFN).
B. Для каждого $j$ вычислить $\tilde{f}_j^*$ и $\tilde{f}_j^-$.
C. Для всех $i,j$ получить $\tilde{d}_{ij}$ по формулам (L,M,U).
D. Вычислить $\tilde{S}_i$ и $\tilde{R}_i$, а потом и $Q_i$ покомпонентно.
E. Дефаззифицировать $\tilde{S}_i, \tilde{R}_i$ в скаляры $S_i, R_i$.
F. Посчитать $S^*$, $S^-$, $R^*$, $R^-$ и скалярный $Q_i$; ранжировать.
G. Проверить условия acceptable advantage, acceptable stability и, при необходимости, применить trade-off (пересчитать веса и повторить шаги с ранжированием).

### 1.4.3. Сравнение подходов

Для классического VIKOR, интервальной и fuzzy модификаций были вычислены ранги для похожих данных. В качестве исходных данных были взяты данные в виде нечетких чисел. Для интервальной модификации были взяты 1 и 3 значения из нечеткого числа. Для классического метода было взято 2 значение. Результаты представлены на таблице 1.6.

По таблице видно, что внесение неопределенности может влиять на ранжирование, а значит модификации VIKOR могут применяться в условиях, когда точное значение не известно.

*Таблица 1.6*

**Результаты ранжирования для разных методов VIKOR**

| Альтернатива | Классический | Интервальный | Fuzzy |
|---|---|---|---|
| A1 | 5 | 5 | 5 |
| A2 | 4 | 4 | 4 |
| A3 | 1 | 1 | 2 |
| A4 | 6 | 6 | 6 |
| A5 | 3 | 3 | 1 |
| A6 | 2 | 2 | 3 |

## 1.5. Выводы

В данной работе проведено систематическое исследование метода многокритериального принятия решений VIKOR и двух его ключевых модификаций, предназначенных для работы в условиях неопределенности: интервального VIKOR и нечеткого (fuzzy) VIKOR.

Основные результаты и положения, представленные в работе, могут быть сформулированы следующим образом:

Классический метод VIKOR, являясь эффективным инструментом для поиска компромиссного решения, оперирует точечными данными. В реальных условиях экспертные оценки, показатели эффективности и веса критериев часто носят неточный или качественный характер. Представленные модификации успешно решают эту проблему: интервальный VIKOR работает с диапазонами значений, а fuzzy VIKOR — с лингвистическими переменными, представленными треугольными нечеткими числами, что значительно повышает адекватность моделирования реальных задач.

В работе подробно изложены математические постановки, формулы и вычислительные процедуры для всех трех версий метода. Особое внимание уделено ключевым особенностям расширений: использованию интервальной арифметики, операциям с нечеткими числами и этапу дефаззификации, который необходим для финального ранжирования альтернатив в fuzzy VIKOR.

Для обеих модификаций разработаны и представлены программные реализации на языке Python, что подтверждает практическую применимость методов.

На конкретных примерах продемонстрирован полный вычислительный цикл — от задания интервальных и нечетких входных данных до получения итогового ранжирования и проверки условий приемлемости компромиссного решения (acceptable advantage и acceptable stability).

Сравнение итогов ранжирования, полученных классическим, интервальным и нечетким VIKOR на сопоставимых данных, выявило важный факт: учет неопределенности может изменять порядок предпочтения альтернатив. Как показано в таблице результатов, альтернатива A5, занимающая 3-е место в классической и интервальной версиях, выходит на 1-е место при использовании fuzzy VIKOR. Это наглядно иллюстрирует, что игнорирование размытости исходных данных может привести к выбору неоптимального с точки зрения реального контекста решения.

Рассмотренные модификации VIKOR особенно востребованы в областях с высокой степенью неопределенности: управление проектами, экологический менеджмент, инвестиционный анализ, оценка инновационных технологий. В качестве перспективных направлений дальнейших исследований можно выделить: развитие гибридных моделей, сочетающих интервальный и нечеткий подходы; интеграцию VIKOR с методами машинного обучения для обработки больших объемов неструктурированных данных; а также углубленное изучение влияния различных стратегий дефаззификации и методов сравнения интервалов на устойчивость итогового решения.

Таким образом, работа подтверждает, что интервальный и нечеткий VIKOR являются мощными и необходимыми расширениями классического метода, которые сохраняют его концептуальные преимущества — ориентацию на поиск взвешенного компромисса — и в то же время значительно расширяют сферу его применимости за счет возможности работы с неточной и качественной информацией. Это делает данные методы ценным инструментом для аналитиков и лиц, принимающих решения, в сложных и неопределенных условиях современного мира.

## Библиографический список


1. *Chatterjee P.*, *Chakraborty S.* A comparative analysis of VIKOR method and its variants // Decision Science Letters. — 2016. — T. 5, № 4. — C. 469—486. — DOI 10.5267/j.dsl.2016.5.004.
2. *Liu P.*, *Qin X.* An Extended VIKOR Method for Decision Making Problem with Interval-Valued Linguistic Intuitionistic Fuzzy Numbers Based on Entropy // Informatica. — 2017. — T. 28, № 4. — C. 665—685. — DOI 10.15388/Informatica.2017.151.
3. *Opricovic S.* Fuzzy VIKOR with an application to water resources planning // Expert Systems with Applications. — 2011. — T. 38, № 10. — C. 12983—12990. — DOI 10.1016/j.eswa.2011.04.097. — URL: https://www.sciencedirect.com/science/article/pii/S0957417411006245.
4. *Sayadi M. K.*, *Heydari M.*, *Shahanaghi K.* Extension of VIKOR method for decision making problem with interval numbers // Applied Mathematical Modelling. — 2009. — T. 33, № 5. — C. 2257—2262. — DOI 10.1016/j.apm.2008.06.002.
5. *Wan S.-P.* The extended VIKOR method for multi-attribute group decision making with triangular intuitionistic fuzzy numbers // Knowledge-Based Systems. — 2013. — T. 52. — C. 65—77. — DOI 10.1016/j.knosys.2013.06.019.