**Постановка задачи:**

В данной работе рассматривается разработка цифрового двойника механической системы с возможностью визуализации и симуляции в реальном времени. В качестве исследуемого объекта выбрана классическая задача сопротивления материалов – балка двутаврового сечения, закрепленная на двух шарнирных опорах по концам.

На балку могут прикладываться сосредоточенные силы, направленные перпендикулярно ее продольной оси, а также внешние моменты. Основная цель работы – разработка математической модели, реализующей статический и динамический анализ данной механической системы, с последующей визуализацией ее деформаций и реакции на внешние нагрузки.

**Теоретическая часть**:

**1. Режимы работы программы**

Разработанная программа цифрового двойника балки поддерживает три режима работы:

**1.1. Ручной режим**

В данном режиме пользователь самостоятельно задает расположение и величину приложенных к балке сосредоточенных сил и моментов. На основе введенных данных строятся эпюры:

• **Поперечных сил**,

• **Изгибающих моментов**,

• **Прогибов балки**.

Дополнительно реализована визуализация балки, где:

• Цвет балки зависит от напряжений:

• **Бордовый** – критическая нагрузка,

• **Фиолетовый** – отсутствие нагрузки.

• Черными линиями обозначены приложенные силы.

• Красными линиями обозначены моменты.

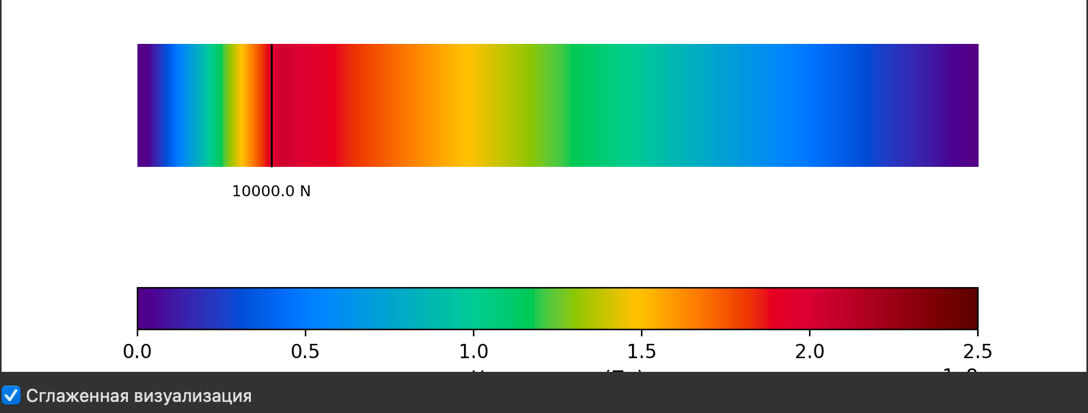
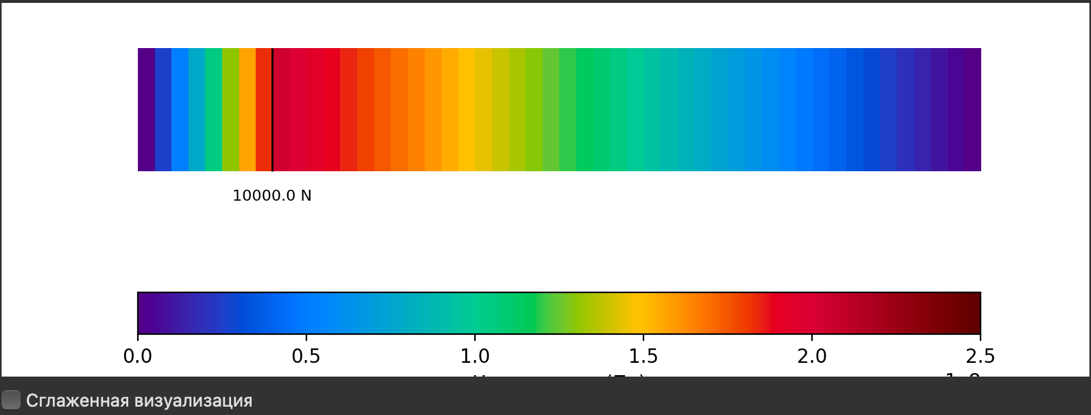
• Доступен **режим сглаженной визуализации**, который улучшает качество отображения балки при **малом количестве точек расчета**. Этот режим выполняет дополнительную интерполяцию между расчетными точками, визуально сглаживая график деформаций.

Ниже представлены скриншоты сравнения:

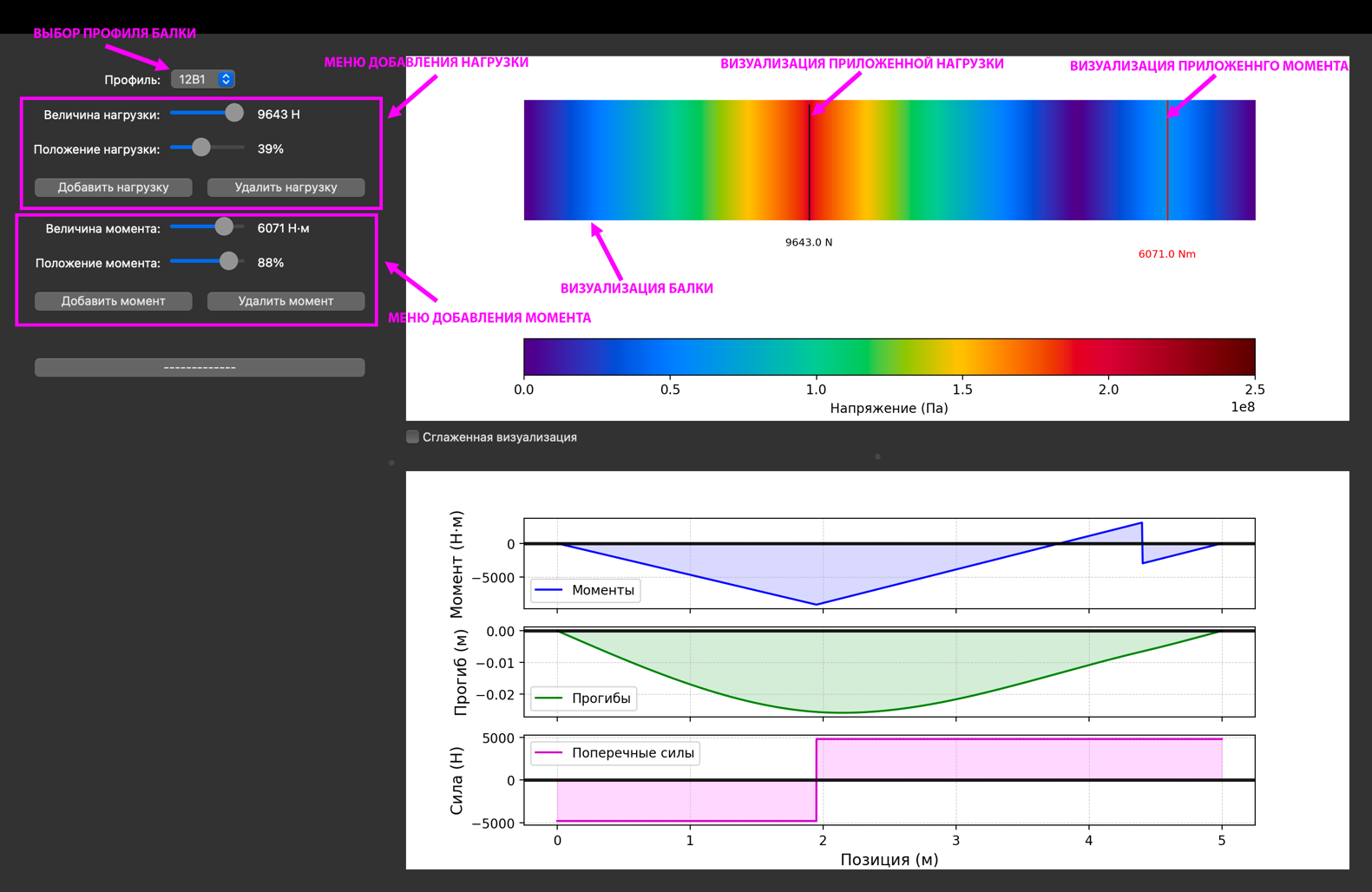
• **Сверху** – стандартная визуализация при n = 100 (до включения сглаживания).

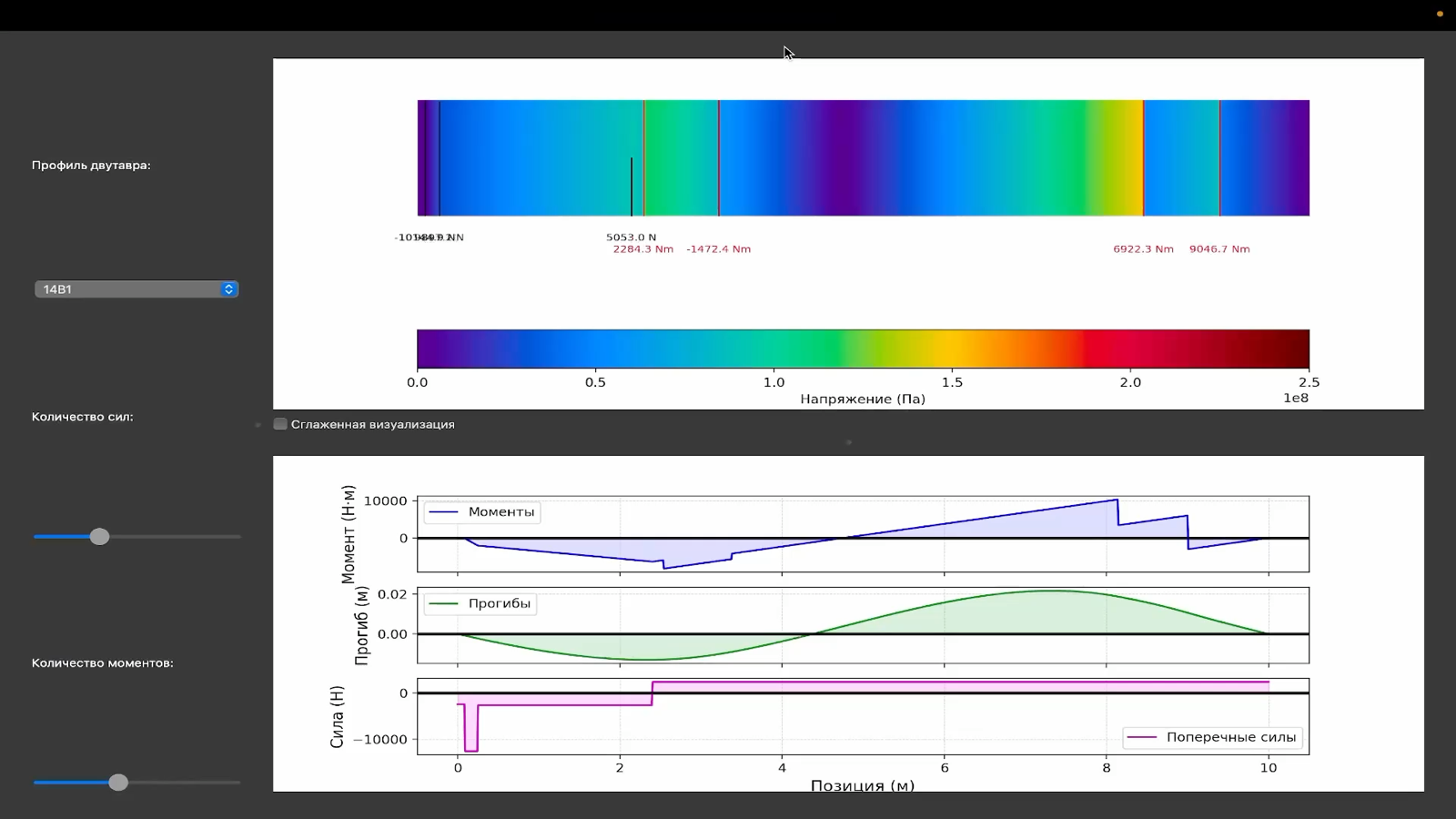
• **Снизу** – визуализация при n = 100 с включенным режимом сглаживания.

Для более точных результатов во всех остальных скриншотах использовано n = 10000, при этом режим сглаженной визуализации **отключен**.



• Пользователь может выбирать **профиль балки**.



**1.2. Автоматический режим**

В этом режиме пользователь выбирает:

• Количество приложенных к балке сил и моментов.

• Профиль двутавра.

После этого программа автоматически изменяет значения сил и моментов **каждый тик (заданная в программе единица времени)**, динамически обновляя эпюры и визуализацию балки, аналогично ручному режиму.

**1.3. Обратный режим**

В этом режиме программа выполняет расчет в обратном порядке:

1. Генерируется **случайный набор значений перемещений**.

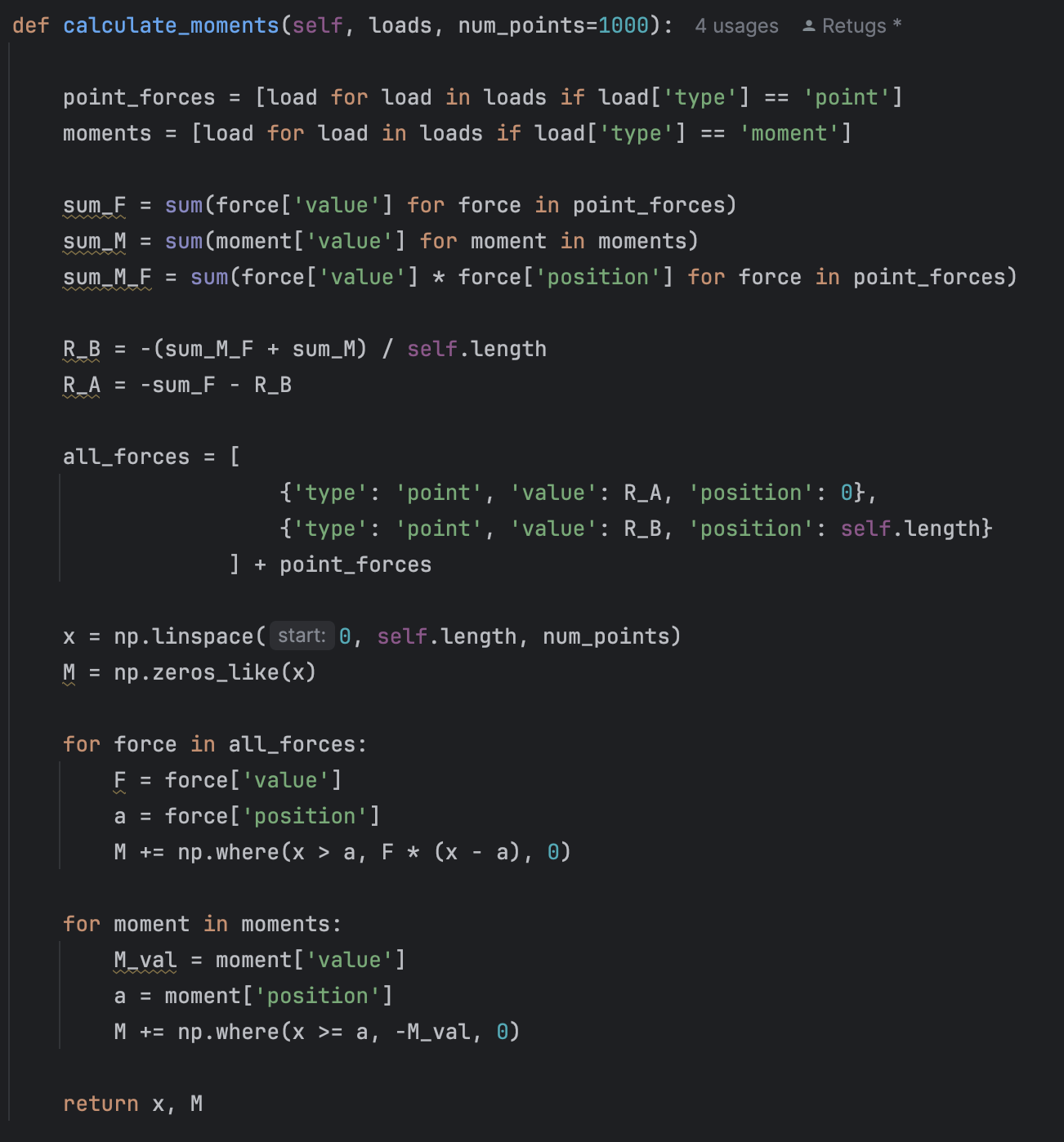
2. На основе этих данных строится **график исходных перемещений**.

3. Программа восстанавливает значения приложенных сил по функции перемещений.

4. Затем строится новый график перемещений **(график восстановленных перемещений)** для сравнения с исходным.

**2. Алгоритмы расчета:**

**2.1. Построение эпюр моментов (beam\_solver.py)**



**Шаг 1. Фильтрация нагрузок:**

Из общего списка **loads** выделяются:

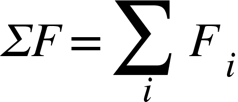
• Точечные силы {"mathml":"<math style=\"font-family:stix;font-size:16px;\" xmlns=\"http://www.w3.org/1998/Math/MathML\"><mstyle mathsize=\"16px\"><msub><mi>F</mi><mi>i</mi></msub></mstyle></math>","origin":"MathType for Microsoft Add-in"} с положениями {"mathml":"<math style=\"font-family:stix;font-size:16px;\" xmlns=\"http://www.w3.org/1998/Math/MathML\"><mstyle mathsize=\"16px\"><msub><mi>F</mi><mi>i</mi></msub></mstyle></math>","origin":"MathType for Microsoft Add-in"}

• Приложенные моменты {"mathml":"<math style=\"font-family:stix;font-size:16px;\" xmlns=\"http://www.w3.org/1998/Math/MathML\"><mstyle mathsize=\"16px\"><msub><mi>M</mi><mi>j</mi></msub></mstyle></math>","origin":"MathType for Microsoft Add-in"} с положениями {"mathml":"<math style=\"font-family:stix;font-size:16px;\" xmlns=\"http://www.w3.org/1998/Math/MathML\"><mstyle mathsize=\"16px\"><msub><mi>a</mi><mi>j</mi></msub></mstyle></math>","origin":"MathType for Microsoft Add-in"}

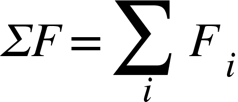
**Шаг 2. Расчёт суммарных нагрузок:**

Вычисляются:

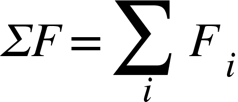
• Суммарная сила:



• Суммарный момент от сил:



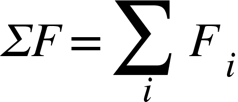
• Суммарный момент от моментов:



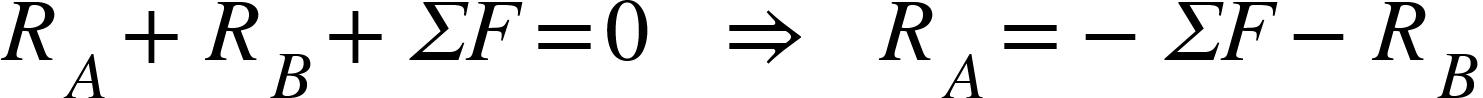
**Шаг 3. Определение реакций опор:**

Из условий равновесия получаем:

• Равновесие моментов относительно левой опоры:



• Равновесие сил:



**Шаг 4. Формирование списка нагрузок:**

К исходным точечным силам добавляются реакции {"mathml":"<math style=\"font-family:stix;font-size:16px;\" xmlns=\"http://www.w3.org/1998/Math/MathML\"><mstyle mathsize=\"16px\"><msub><mi>R</mi><mi>A</mi></msub></mstyle></math>","origin":"MathType for Microsoft Add-in"} (в точке x=0) и {"mathml":"<math style=\"font-family:stix;font-size:16px;\" xmlns=\"http://www.w3.org/1998/Math/MathML\"><mstyle mathsize=\"16px\"><msub><mi>R</mi><mi>B</mi></msub></mstyle></math>","origin":"MathType for Microsoft Add-in"} (в точке x=L).

**Шаг 5. Расчёт распределения момента:**

Создаётся дискретизация балки:

{"mathml":"<math style=\"font-family:stix;font-size:16px;\" xmlns=\"http://www.w3.org/1998/Math/MathML\"><mstyle mathsize=\"16px\"><mi>x</mi><mo>=</mo><mtext>linspace</mtext><mo>(</mo><mn>0</mn><mo>,</mo><mi>L</mi><mo>,</mo><mtext>num_points</mtext><mo>)</mo></mstyle></math>","origin":"MathType for Microsoft Add-in"}

Инициализируется массив моментов M(x) = 0. Далее:

• **Вклад точечных сил:**

Для каждой силы F в точке a добавляется:

\Delta M(x) = \begin{cases} F \cdot (x - a), & \text{если } x > a, \\ 0, & \text{иначе.} \end{cases}

\Delta M(x) = \begin{cases} F \cdot (x - a), & \text{если } x > a, \\ 0, & \text{иначе.} \end{cases}

• **Вклад моментов:**

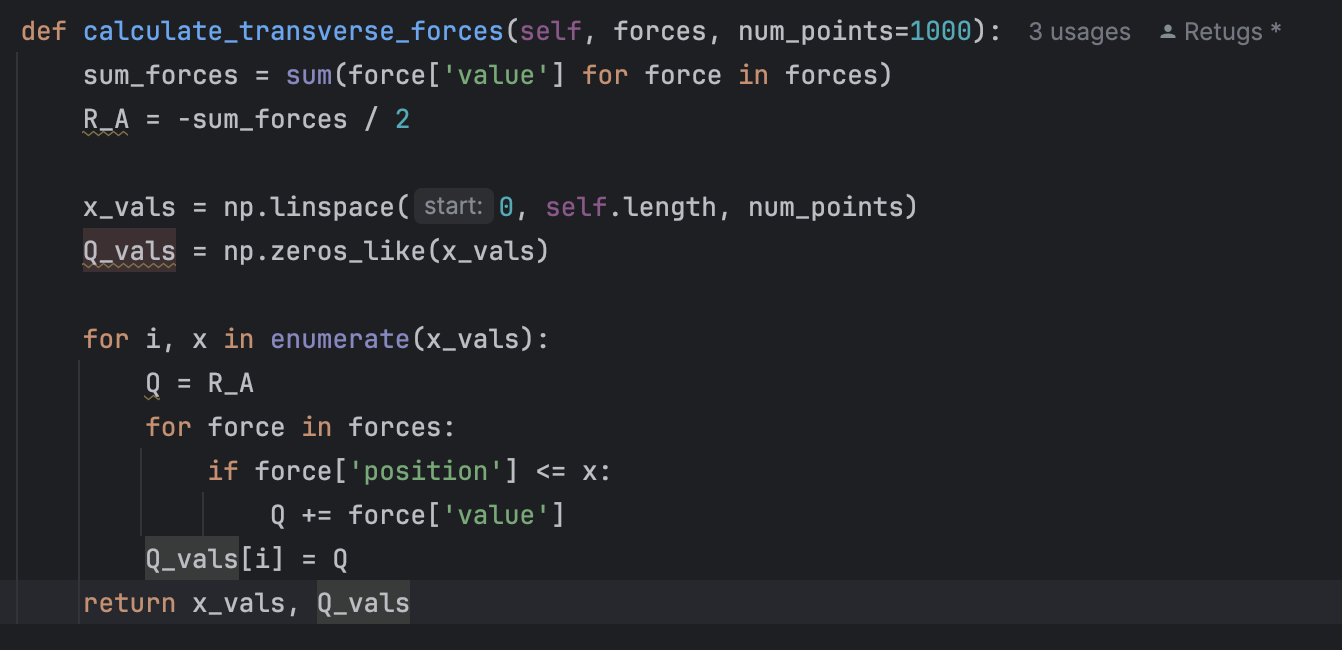
Для каждого момента M\_j в точке a добавляется:

\Delta M(x) = \begin{cases} • M\_j, & \text{если } x \geq a, \\ 0, & \text{иначе.} \end{cases}

**Шаг 6. Итог:**

Метод возвращает массивы x (координаты вдоль балки) и M(x) (распределение изгибающих моментов).

**2.2. Построение эпюр поперечных сил (beam\_solver.py)**



**Шаг 1. Расчёт реакции опоры**

Сначала вычисляется суммарная сила:

\Sigma F = \sum\_{i} F\_i

При условии симметричного распределения реакции опор, реакция в левой опоре определяется как:

R\_A = -\frac{\Sigma F}{2}

**Шаг 2. Дискретизация балки**

Создаётся массив координат вдоль балки:

x = \text{linspace}(0, L, \text{num\\_points})

**Шаг 3. Расчёт поперечной силы** Q(x)

Для каждой точки x рассчитывается значение поперечной силы, начиная с реакции R\_A. Затем к R\_A добавляются все силы, приложенные в точках, где x\_{\text{force}} \leq x:

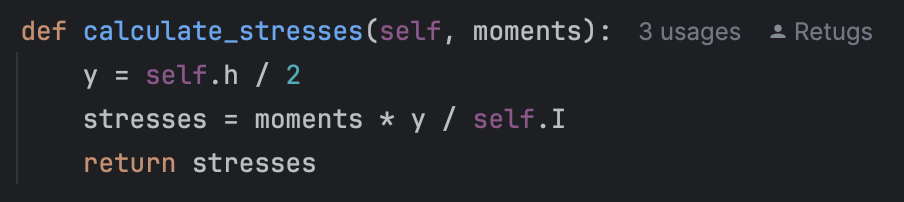
Q(x) = R\_A + \sum\_{x\_i \leq x} F\_i

Это значение сохраняется для каждой точки x.

**Шаг 4. Итог**

Метод возвращает массивы x (координаты вдоль балки) и Q(x) (распределение поперечных сил).

**2.3. Построение эпюр напряжений (beam\_solver.py)**



Метод принимает массив изгибающих моментов M(x) и вычисляет нормальные напряжения в сечении балки по следующей схеме:

1. **Определение расстояния от нейтральной оси:**

Расстояние до верхней (или нижней) точки сечения рассчитывается как:

y = \frac{h}{2}

где h – высота балки.

2. **Вычисление напряжений:**

Нормальные напряжения рассчитываются по формуле:

\sigma(x) = \frac{M(x) \cdot y}{I}

где I – момент инерции сечения.

3. **Возврат результата:**

Метод возвращает массив рассчитанных значений напряжений \sigma(x).

**2.4. Построение эпюр удлинений (beam\_solver.py)**



1. **Расчет моментов:**

Вызывается функция для получения распределения моментов M(x) вдоль балки.

2. **Интерполяция моментов:**

Создаётся кубическая интерполирующая функция M\_{\text{interp}}(x) для получения гладкой зависимости.

3. **Определение изгибающей жесткости:**

Вычисляется произведение модуля упругости и момента инерции:

EI = E \cdot I

4. **Расчет кривизны:**

По соотношению для гибких балок находим функцию кривизны:

f(x) = -\frac{M\_{\text{interp}}(x)}{EI}

5. **Двойное интегрирование:**

• Первое интегрирование даёт наклон балки:

\frac{dw}{dx}(x) = \int\_0^x f(s) \, ds

• Второе интегрирование – перемещения:

w(x) = \int\_0^x \frac{dw}{dx}(s) \, ds

6. **Коррекция по граничным условиям:**

Для соблюдения условий w(0)=w(L)=0 от полученной кривой вычитается линейная функция, равномерно меняющаяся от 0 до w(L):

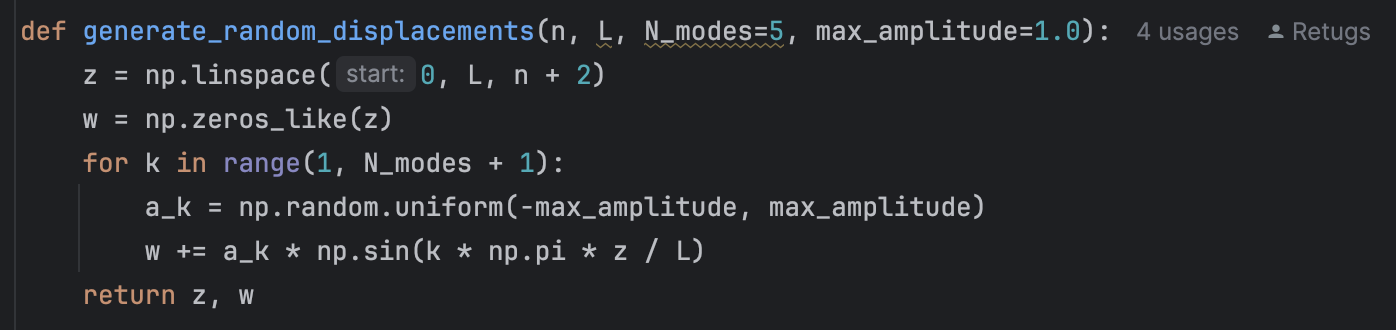
w\_{\text{corrected}}(x) = w(x) - \text{correction}(x)

где, например, \text{correction}(x) = \frac{w(L)}{L} x.

Метод возвращает массив координат x и откорректированные значения перемещений w\_{\text{corrected}}(x).

**3. Расчеты для обратной задачи (reverse\_loader.py)**

**3.1. Генерация случайных перемещений**



**Генерация случайных перемещений с использованием суперпозиции синусоидальных мод**

Функция генерирует случайные перемещения вдоль балки, применяя метод суперпозиции синусоидальных мод. Алгоритм работы функции выглядит следующим образом:

1. **Дискретизация координат:**

Создаётся равномерный массив координат z от 0 до L с количеством точек n + 2 (включая концы). Это обеспечивает достаточную детализацию профиля перемещений.

2. **Инициализация перемещений:**

Массив перемещений w(z) инициализируется нулями, что гарантирует выполнение граничных условий w(0)=0 и w(L)=0.

3. **Суперпозиция синусоидальных мод:**

Функция строит итоговое перемещение w(z) как сумму нескольких синусоидальных функций. Для каждого режима k (от 1 до N\_{\text{modes}}) генерируется случайный коэффициент a\_k из диапазона [-A\_{\text{max}}, A\_{\text{max}}], где A\_{\text{max}} – максимальная амплитуда. Каждая мода вносит вклад по формуле:

w(z) = w(z) + a\_k \cdot \sin\left(\frac{k \pi z}{L}\right)

В итоге функция имеет вид:

w(z) = \sum\_{k=1}^{N\_{\text{modes}}} a\_k \cdot \sin\left(\frac{k \pi z}{L}\right)

4. **Возврат результата:**

Функция возвращает массив координат z и вычисленные перемещения w(z).

**Объяснение принципа суперпозиции синусоидальных мод**

Суперпозиция синусоидальных мод – это представление сложного колебательного процесса в виде суммы нескольких простых синусоидальных функций, каждая из которых характеризуется своей частотой (модой) и амплитудой. Такой подход удобен в данной задаче по следующим причинам:

• **Автоматическое удовлетворение граничных условий:**

При выборе аргумента \frac{k \pi z}{L} синусоидальные функции \sin\left(\frac{k \pi z}{L}\right) естественным образом удовлетворяют условиям w(0)=0 и w(L)=0, что соответствует физическим ограничениям балки.

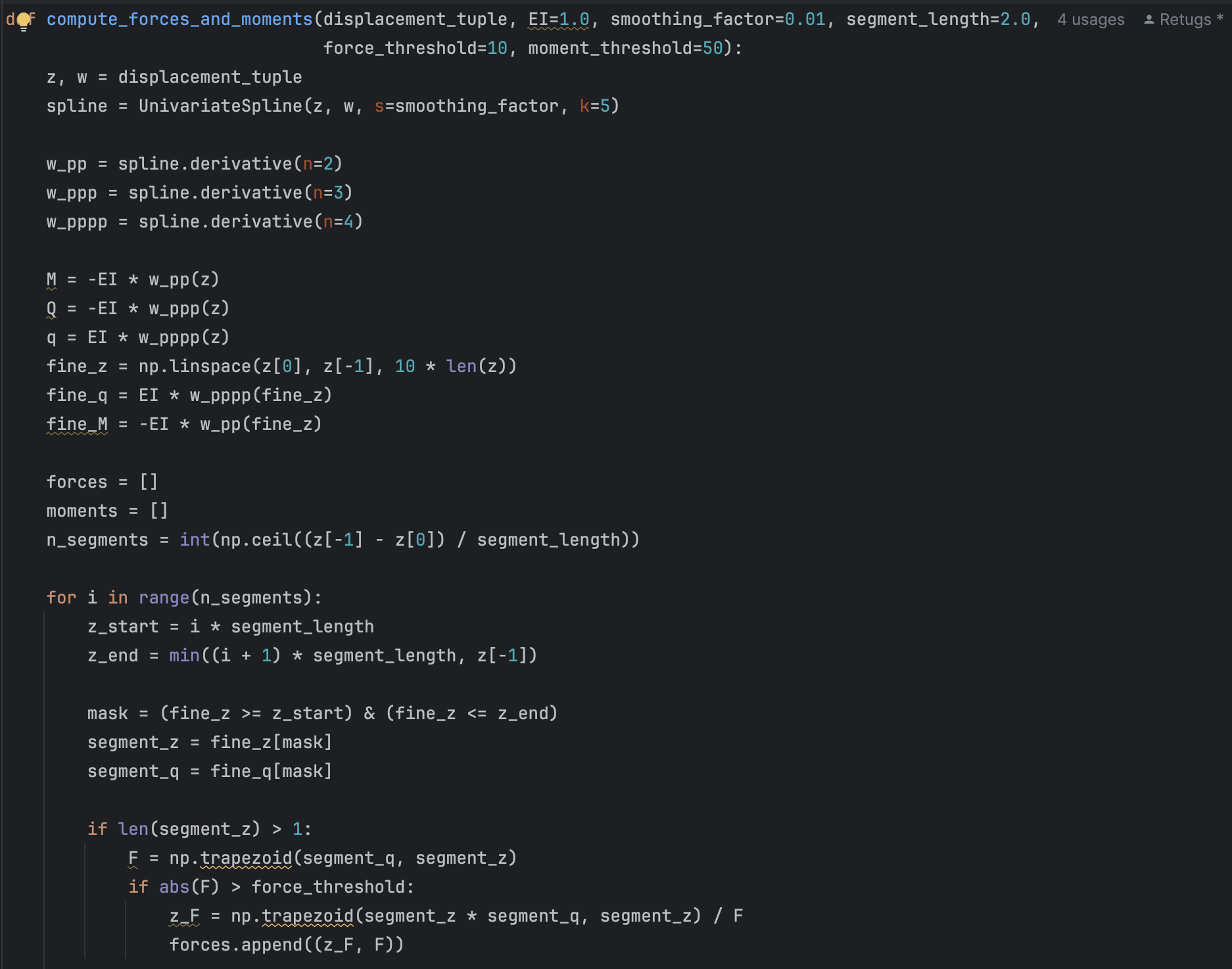
• **Линейность и простота расчётов:**

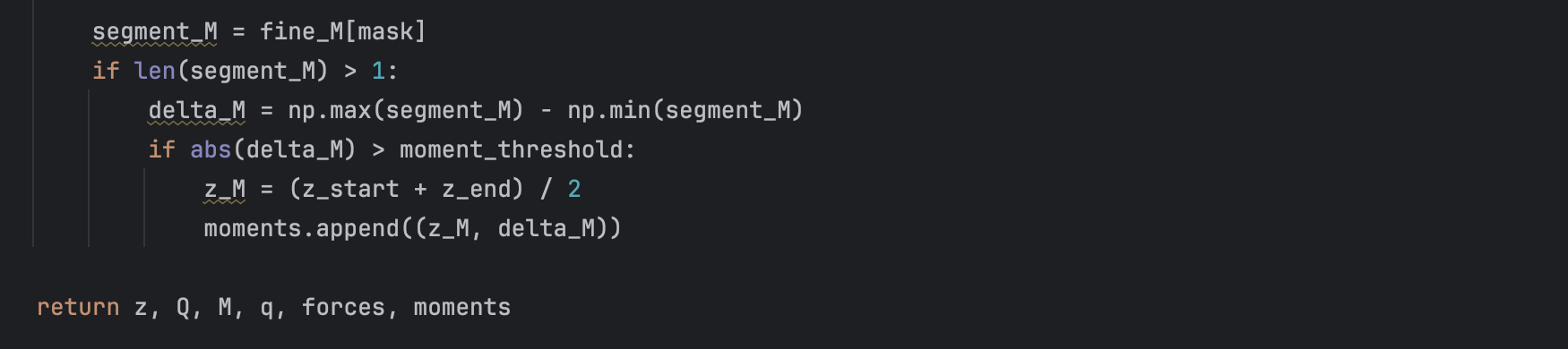
Благодаря линейности задачи результирующее перемещение можно получить как сумму отдельных мод:

w(z) = \sum\_{k=1}^{N\_{\text{modes}}} a\_k \cdot \sin\left(\frac{k \pi z}{L}\right)

Здесь случайные коэффициенты a\_k определяют вклад каждой моды, что позволяет легко варьировать форму перемещений.

**3.2. Расчет сил по заданным перемещениям**





Функция **compute\_forces\_and\_moments** вычисляет внутренние силы и моменты по заданной функции перемещений. Алгоритм работы следующий:

1. **Подготовка данных и аппроксимация перемещений:**

На вход подаётся кортеж (z, w), где z – координаты вдоль балки, а w(z) – перемещения. Для сглаживания и получения непрерывной функции применяется сплайн-приближение (UnivariateSpline) с заданным параметром сглаживания и степенью k=5.

2. **Вычисление производных:**

Рассчитываются вторая, третья и четвертая производные аппроксимирующей функции:

• w{\prime}{\prime}(z) – вторая производная (кривизна),

• w{\prime}{\prime}{\prime}(z) – третья производная,

• w{\prime}{\prime}{\prime}{\prime}(z) – четвертая производная.

3. **Расчет распределённых величин:**

С использованием модуля упругости E и момента инерции I (с учетом их произведения EI) вычисляются:

• **Изгибающий момент:**

M(z) = -EI \cdot w{\prime}{\prime}(z)

• **Поперечная сила:**

Q(z) = -EI \cdot w{\prime}{\prime}{\prime}(z)

• **Распределённая нагрузка:**

q(z) = EI \cdot w{\prime}{\prime}{\prime}{\prime}(z)

4. **Уточнение расчётов на более плотной сетке:**

Создается «тонкая» сетка \text{fine\\_z} для интегрирования и точного вычисления величин q(z) и M(z).

5. **Интегрирование по сегментам:**

Балка делится на сегменты длиной, заданной параметром segment\_length. Для каждого сегмента:

• **Расчет результирующей силы:**

Сила F на сегменте определяется по правилу трапеций:

F = \int\_{z\_{\text{start}}}^{z\_{\text{end}}} q(z) \, dz

Если абсолютное значение |F| превышает пороговое значение, вычисляется точка приложения силы:

z\_F = \frac{\int\_{z\_{\text{start}}}^{z\_{\text{end}}} z \, q(z) \, dz}{F}

• **Расчет разности моментов:**

Определяется разность между максимумом и минимумом M(z) на сегменте:

\Delta M = \max(M(z)) - \min(M(z))

Если |\Delta M| больше заданного порога, считается, что на сегменте присутствует значимый момент, и его значение закрепляется в средней точке сегмента:

z\_M = \frac{z\_{\text{start}} + z\_{\text{end}}}{2}

6. **Возврат результата:**

Функция возвращает:

• Исходный массив координат z,

• Поперечные силы Q(z),

• Изгибающие моменты M(z),

• Распределённую нагрузку q(z),

• Список найденных точек приложения сил (z\_F, F),

• Список найденных моментов (z\_M, \Delta M).