**Стохастические методы оптимизации и их приложения к задачам финансовой эконометрики**

Веселова Арина БЭК201

**Метод имитации отжига для ферзей**

Метод имитации отжига или просто метод отжига представляет собой простой и эффективный алгоритм для решения оптимизационных и поисковых задач, основанный̆ на стохастическом поиске. Метод отжига относится к так называемым эвристическим методам, поэтому с одной̆ стороны он может быстро находить решения экспоненциально сложных задач, а с другой̆ стороны этот метод не гарантирует сходимость к решению.

В качестве примера использования метода имитации отжига мы рассмотрим задачу-головоломку о размещении N ферзей на шахматной доске N × N таким образом, чтобы ни один из ферзей не атаковал других ферзей.

Пусть с0 – это начальное положение ферзей, а initialTemperature – изначальная температура, функция NumAttacks осуществляет подсчет количества ферзей под ударом в конкретном положении ферзей, функция GenerateCandidate осуществляет генерацию новой последовательности ферзей через перестановку двух случайных ферзей, функция MakeTransit определяет, будет ли происходить переход от одной последовательности к другой.

Таким образом, алгоритм выглядит следующим образом:

1. Генерируем новую последовательность ферзей и уменьшаем температуру.
2. Затем считаем разницу ферзей под ударом между новым состоянием и старым состоянием ферзей.
   * Если количество ферзей под ударом для нового положения меньше, чем для старого положения, то новое состояние принимаем за наиболее оптимальное для данной итерации и переходим к шагу 1.
   * Если количество ферзей под ударом для нового состояния больше, чем для старого, то считаем вероятность по формуле , где - разница в количестве ферзей под ударом для нового и старого состояний ферзей. Если данная вероятность больше порога, то принимаем новое состояние за наиболее оптимальное для данной итерации, иначе наиболее оптимальным остается старое положение ферзей и переходим к шагу 1.
3. Повторяем предыдущие шаги пока температура не приблизится к 0.

Главная функция выглядит следующим образом:

Изображение выглядит как текст

Автоматически созданное описание

Главная функция, реализующая метод имитации отжига, состоит из ряда других функций.

Составные функции:

1. Функция, осуществляющая подсчет ферзей под ударом

Изображение выглядит как текст

Автоматически созданное описание

2. Функция, осуществляющая генерацию новой последовательности

Изображение выглядит как текст

Автоматически созданное описание

3. Функция, осуществляющая проверку соответствия вероятности некоторому уровню (чтобы определить, принимается ли новая последовательность ферзей или нет)

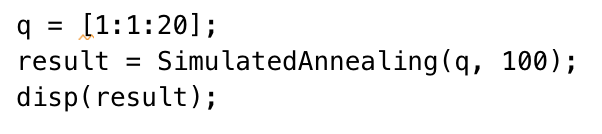
Изображение выглядит как текст

Автоматически созданное описание

Примеры реализации:

**Пример 1.**

Реализуем **поиск для 20 ферзей** с изначальной температуры 100.



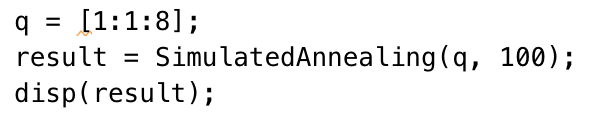
Вывод функции:



Где [7, 14, 4, 8, 13, 19, 12, 1, 5, 17, 9, 11, 15, 3, 6, 10, 20, 18, 16, 2] – это итоговое положение ферзей, а 0 – это количество ферзей под ударом для такой последовательности.

**Пример 2.**

Реализуем **поиск для 8 ферзей** с температуры 100.



Вывод функции:

Изображение выглядит как текст, часы, датчик

Автоматически созданное описание

Где [4, 2, 7, 3, 6, 8, 1, 5] – это итоговое положение ферзей, а 0 – это количество ферзей под ударом для такой последовательности.

**Метод имитации отжига для непрерывной функции**

Другим примером применения метода имитации отжига является **минимизация функции**

Пусть с0 – это начальная точка, а initialTemperature – изначальная температура, функция Value осуществляет подсчет значения функции в текущей точке, функция NewDot осуществляет генерацию новой точки по формуле , где – случайная величина равномерно распределённая на (-1,1), функция MakeTransit определяет, будет ли происходить переход от одной точки к другой.

Главная функция для данной задачи:

Изображение выглядит как текст

Автоматически созданное описание

Главная функция, реализующая метод имитации отжига, состоит из ряда других функций.

Составные функции:

1. Функция, осуществляющая вычисление значения функции в текущей точке

Изображение выглядит как текст

Автоматически созданное описание

1. Функция, осуществляющая генерацию новой точки

Изображение выглядит как текст

Автоматически созданное описание

1. Функция, осуществляющая проверку соответствия вероятности некоторому уровню (чтобы определить, принимается ли новая точка или нет)

Изображение выглядит как текст

Автоматически созданное описание

Примеры реализации:

**Пример 1.**

Реализуем **поиск точки минимума функции с температуры 100 до 50.**

Изображение выглядит как текст

Автоматически созданное описание

Вывод функции:

Изображение выглядит как текст

Автоматически созданное описание

Где верхнее число – это значение функции в найденном минимуме, а нижнее – найденная точка минимума.

**Пример 2.**

Реализуем **поиск точки минимума с температуры 100 до 10.**

Изображение выглядит как текст

Автоматически созданное описание

Вывод функции:

Изображение выглядит как текст

Автоматически созданное описание

Где верхнее число – это значение функции в найденном минимуме, а нижнее – найденная точка минимума.

**Пример 3.**

Реализуем **поиск точки минимума с температуры 100 до 0.0000001.**

Изображение выглядит как текст

Автоматически созданное описание

Результат:

Изображение выглядит как текст

Автоматически созданное описание

Где верхнее число – это значение функции в найденном минимуме, а нижнее – найденная точка минимума.

**Метод роения частиц**

Для решения оптимизационных задач большую роль играют коллективные методы поиска решений, когда одно временно ищут не одна приближение к оптимальному решению, а сразу большое количество приближений, которые объединены в сообщество. При этом на каждом шаге каждое приближение уточняется в зависимости от всех приближений. Наиболее известными коллективными методами решения оптимизационных задач являются методы роения частиц и генетические алгоритмы.

Алгоритм выглядит следующим образом:

1. Пусть в начальный момент времени мы имеем сообщество точек с начальной скоростью
2. В следующий момент времени t + 1 эти точки вычисляются по следующей формуле , где vm(t) ∈ Rn суть скорости точек в момент времени t. Эти скорости вычисляются по формуле , где – это параметры методы, а - независимые случайные величины, равномерно распределенные на (0, 1).
3. Далее для каждой точки в момент времени t рассмотрим точку такую, что . Эта точка является наилучшей точкой для m-ой точки за все ее историю.
4. Также определим еще точку , которая определяется соотношением . Эта точка является наилучшей для всего сообщества.

Идея метода роения частиц состоит в том, что каждая частица перемещается со скоростью, которая направлена на лучшую точку в истории частицы и одновременно на лучшую точку для всего сообщества. При этом компоненты скорости будут содержать случайные компоненты.

**Пример 1**

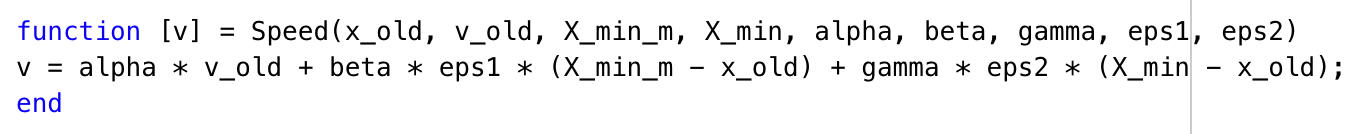
Найдем **минимум для функции Розенброка**

Главная функция, реализующая метод роения частиц:

Изображение выглядит как текст

Автоматически созданное описание

* Внутри главной функции вложена функции по вычислению скорости на каждой новой итерации:



* Конкретно для примера функции Розенброка реализована также следующая функция подсчета значения в точке:

Изображение выглядит как текст

Автоматически созданное описание

Тогда после реализации алгоритма получаем результат:

Изображение выглядит как текст

Автоматически созданное описание

То есть минимальное значение близкое к 0 было найдено в точке (1, 1).

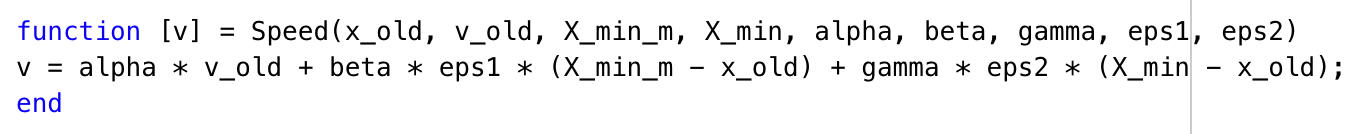
**Пример 2**

Главная функция, реализующая метод роения частиц:

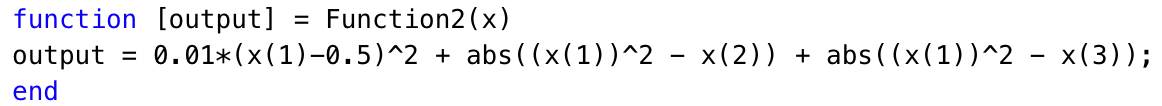
Изображение выглядит как текст

Автоматически созданное описание

* Внутри главной функции вложена функции по вычислению скорости на каждой новой итерации:



* Реализована также следующая функция вычисления значения заданной функции:



Тогда после реализации алгоритма получаем результат:

Изображение выглядит как диаграмма

Автоматически созданное описание

То есть минимальное значение было найдено в точке (0.4911, 0.2412, 0.2412).

**Генетические алгоритмы**

Идея генетических алгоритмов состоит в том, что мы будем одновременно оптимизировать не один ген, а целую популяцию генов. При этом в процессе оптимизации используются следующие основные процедуры:

1. Порождение новых генов случайным образом.
2. Скрещивание генов.
3. Мутация генов.

Операция скрещивания генов является основной в генетических алгоритмах. Под скрещиванием мы будем понимать операцию создания нового гена, у которого часть хромосом от одного гена, а часть хромосом от другого гена. При этом те гены, которые являются лучшими относительно оптимизируемой функции, чаще участвуют в скрещивании.

Под мутацией мы будем понимать операцию, когда у гена случайным образом изменяются одна или несколько хромосом.

В генетическом алгоритме нужно будет ранжировать особи (гены) по их качеству, т. е. близости к оптимальному решению. В генетических алгоритмах для оценки этой близости используют термин — приспособленность особи. При этом приспособленность оценивается с помощью функции приспособленности . Тогда особь x′ более приспособленная или лучше, чем особь x′′, если .

Генетический алгоритм предполагает, что что мы одновременно имеем пул различных особей, которые будут эволюционировать по приведенным выше правилам.

Алгоритм метода выглядит следующим образом:

1. Создаем пул из M случайных особей, которые будем обозначать где – это случайные величины, равномерно распределенные на отрезке [ai, bi].

2. Для каждой особи рассчитываем ее функцию приспособленности.

3. Ранжируем особи по значениям их функции приспособленности таким образом, чтобы

4. Особи с номерами , где – число «убиваемых» наихудших особей, и заменяем на результаты скрещивания особи xm и случайной особью в пуле. Процесс скрещивания: пусть мы имеем две особи x, y ∈ D. Вычислим вероятности

В результате их скрещивания получаем новую особь , где

и — равномерно распределенная случайная величина.

5. Особей с номерами заменяем на новые случайные особи из случайной мутации. То есть выбираем случайный номер 1 ≤ j ≤ n. Во-вторых, вычисляем случайную величину , такую что После этого особь состоит из компонент

6. Переходим к шагу 2, пока не будет выполнено все число итераций.

**Пример 1**

Осуществим метод для **минимизации функции .**

Главная функция, реализующая метод роения частиц:

Изображение выглядит как текст

Автоматически созданное описание

Внутри главной функции вложены следующие составляющие функции:

* функция подсчета приспособленности особи:

Изображение выглядит как текст

Автоматически созданное описание

* функция осуществления скрещивания

Изображение выглядит как текст

Автоматически созданное описание

Внутри которой происходит подсчет вероятности замены того или иного элемента особи на элемент другой особи:

Изображение выглядит как текст

Автоматически созданное описание

* функция осуществления мутации

Изображение выглядит как диаграмма

Автоматически созданное описание

И для конкретного примера функции реализован подсчет значений этой функции:

Изображение выглядит как текст

Автоматически созданное описание

Тогда после реализации алгоритма получаем результат:

Изображение выглядит как текст

Автоматически созданное описание

То есть минимальное значение было найдено в точке (0.0370, -0.0122, -0.0042, -0.0544, 0.0080).

**Пример 2**

Рассмотрим другую задачу. Пусть у нас есть конечное множество натуральных чисел A. **Необходимо разбить данное множество на два подмножества** , таким образом, чтобы достигала минимального значения.

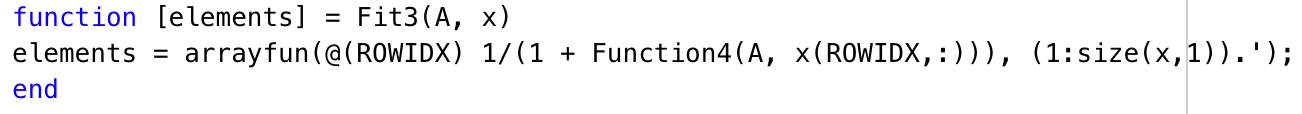
Главная функция, реализующая метод роения частиц:

Изображение выглядит как текст

Автоматически созданное описание

Внутри главной функции вложены следующие составляющие функции:

* функция подсчета приспособленности особи:



* функция осуществления скрещивания

Изображение выглядит как текст

Автоматически созданное описание

Внутри которой происходит подсчет вероятности замены того или иного элемента особи на элемент другой особи:

Изображение выглядит как текст

Автоматически созданное описание

* функция осуществления мутации

Изображение выглядит как текст

Автоматически созданное описание

И для конкретного примера функции реализован подсчет значений этой функции **H**:

Изображение выглядит как текст

Автоматически созданное описание

Тогда после реализации алгоритма получаем результат:

Изображение выглядит как текст, Шрифт, белый, типография

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, белый, дизайн

Автоматически созданное описание

То есть было найдено такое разбиение, которое дает значение равное 0 целевой функции.