МГТУ имени Н.Э. Баумана
Кафедра ИУ-1 «Системы автоматического управления»

Методы оптимизации

Методы линейного поиска: методы нулевого порядка

Завьялов Роман Александрович
zavialov@bmstu.ru

2023 г.

1

Релаксационная последовательность

Общая черта численных методов решения задачи оптимизации – построение такой последовательности $\{x_k\}$ точек в R^n, что значения целевой функции $f(x)$ в точках последовательности удовлетворяют неравенствам:

$$f(x_k) \le f(x_{k-1}),\ k \in \mathbb{N}$$

Итерационную последовательность $\{x_k\}$, обеспечивающую для целевой функции $f(x)$ выполнение условия

$$f(x_k) \le f(x_{k-1}),\ k \in \mathbb{N}$$

называют релаксационной последовательностью.

Условия останова построения релаксационной последовательности:

$$|x_k - x_{k-1}| < \varepsilon_1$$

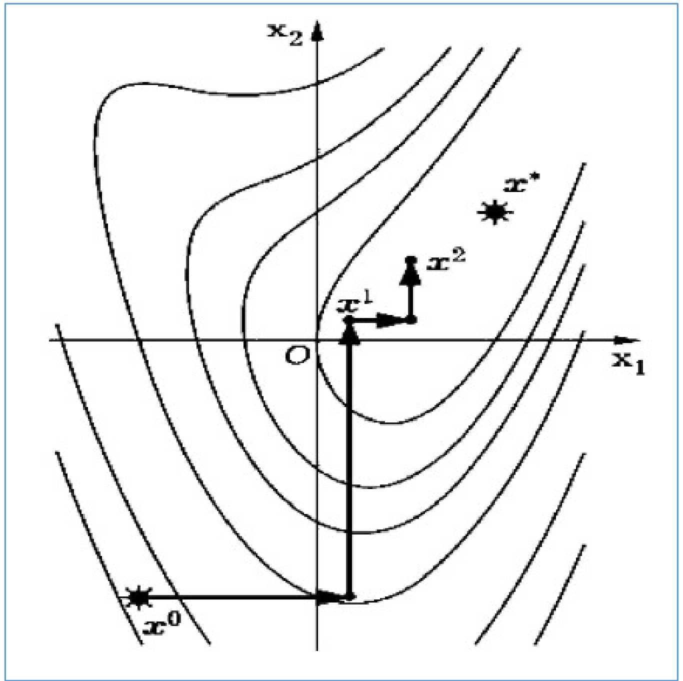
$$|f(x_k) - f(x_{k-1})| < \varepsilon_2$$

$$|\mathrm{grad}\, f(x_{k-1})| < \varepsilon_3$$

Метод покоординатного спуска

Coordinate Descent method

Идея подхода: *последовательно применяем известные одномерные методы оптимизации для каждой координаты по-отдельности, фиксируя значения остальных координат*



Подготовительный этап:

Зададим начальное приближение x_0^1 *и точность расчета* ε.

Шаг 1:

Рассчитаем значения на текущей итерации, решив задачу одномерной минимизации для каждой координаты:

$$\begin{cases} x_1^i = x_0^i + \lambda_1^i e_1 \\ \dots \\ x_n^i = x_{n-1}^i + \lambda_n^i e_n \end{cases}, \text{ где } \begin{cases} e_1 = (1, 0, 0, \dots, 0) \\ \dots \\ e_n = (0, 0, 0, \dots, 1) \end{cases} \text{ и } \begin{cases} \lambda_1^i = \min\limits_{\lambda} f\left(x_0^i + \lambda e_1\right) \\ \dots \\ \lambda_n^i = \min\limits_{\lambda} f\left(x_{n-1}^i + \lambda e_n\right) \end{cases}$$

Шаг 2:

Проверим выполнение условия останова:

Если $\left|x_n^i - x_0^i\right| > \varepsilon$ *, то* $x_0^{i+1} = x_n^i,\ i = i+1$

Иначе – завершаем поиск со значением $x^* = x_n^i$

3

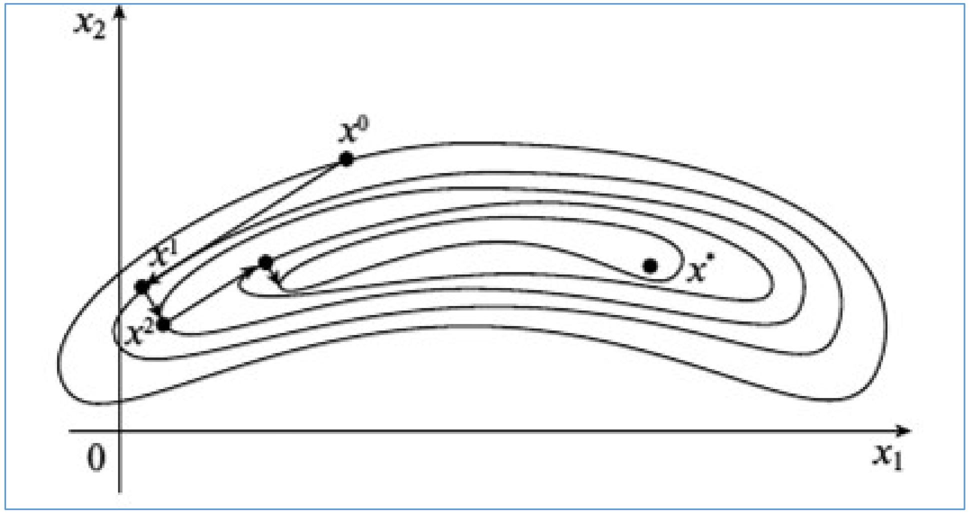
Метод покоординатного спуска

Coordinate Descent method

Недостаток метода: *медленная сходимость и/или неточное приближение к минимуму функции овражного типа. Овражные функции – это функции, поверхности уровня которых значительно вытянуты.*

Дважды непрерывно дифференцируемую функцию называют «овражной», если существует некоторая область $G \in R^n$*, где собственные числа матрицы Гессе, упорядоченные в любой точке* $x \in G$ *по убыванию модулей* $|\lambda_1| \geq |\lambda_2| \geq \ldots |\lambda_n|$*, удовлетворяют неравенствам* $|\lambda_1| \gg \left|\min\limits_{\lambda_i \neq 0} \lambda_i\right| > 0$.

Степень «овражности» характеризуется отношением $\frac{\lambda_1}{\min\limits_{\lambda_i \neq 0} \lambda_i}$.



4 Метод Хука-Дживса

Pattern Search

Идея подхода: *в окрестности начальной точки находим точку с наименьшим значением целевой функции (базовую точку). Затем решаем задачу одномерной минимизации – находим шаг оптимизации по направлению прямой, соединяющей начальную и базовую точку.*
Алгоритм предполагает два этапа – исследующий поиск и поиск по образцу:

Исследующий поиск:

На этапе исследующего поиска определяют базовую точку. Аналогично методу Гаусса-Зейделя, перебираем точки вдоль координатных направлений от начальной точки. После перебора всех n координат выбираем пробную точку с наименьшим значением целевой функции. Шаг поиска успешен, если значение целевой функции в найденной точке не превышает исходное. Полученную точку называют базовой.

Поиск по образцу:

Определяем величину шага спуска вдоль прямой, соединяющей начальную и базовую точки, решая задачу одномерной оптимизации (т.е. выполняем «поиск по образцу», установленному на первом этапе алгоритма). Результат решения – значение шага (ускоряющий множитель). Завершаем поиск по выполнении условия останова.

Метод Хука-Дживса

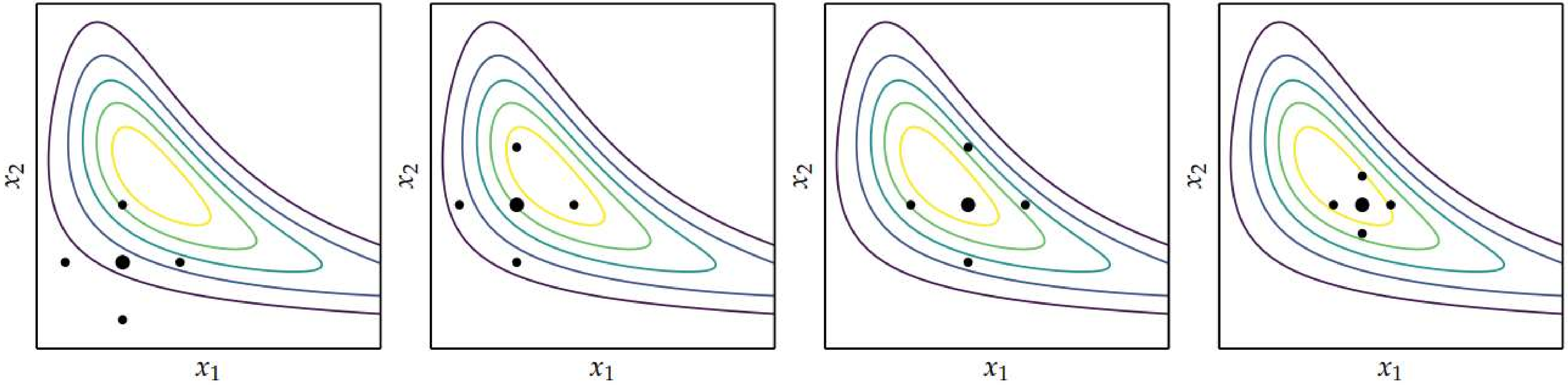
Пример

Исследующий поиск:

На этапе исследующего поиска определяют базовую точку. Аналогично методу Гаусса-Зейделя, перебираем точки вдоль координатных направлений от начальной точки. После перебора всех n координат выбираем пробную точку с наименьшим значением целевой функции. Шаг поиска успешен, если значение целевой функции в найденной точке не превышает исходное. Полученную точку называют базовой.

Поиск по образцу:

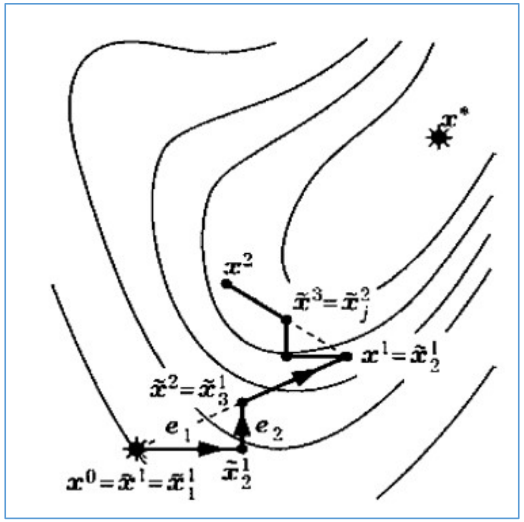
Определяем величину шага спуска вдоль прямой, соединяющей начальную и базовую точки, решая задачу одномерной оптимизации. Результат решения – значение шага (ускоряющий множитель). Завершаем поиск по выполнении условия останова.



Метод Хука-Дживса

Pattern Search

Идея подхода: *в окрестности начальной точки находим точку с наименьшим значением целевой функции (базовую точку). Затем решаем задачу одномерной минимизации – находим шаг оптимизации по направлению прямой, соединяющей начальную и базовую точку.*



Подготовительный этап:

Зададим начальное приближение x_0^1 *и точность расчета* ε.

Шаг 1:

Рассчитаем значения на текущей итерации:

$$\begin{cases} x_1^i = x_0^i + \lambda_1^i e_1 \\ \dots \\ x_n^i = x_{n-1}^i + \lambda_n^i e_n \end{cases}, \text{ где } \begin{cases} e_1 = (1,0,0,\dots,0) \\ \dots \\ e_n = (0,0,0,\dots,1) \end{cases} \text{ и } \begin{cases} \lambda_1^i = \min\limits_{\lambda} f\left(x_0^i + \lambda e_1\right) \\ \dots \\ \lambda_n^i = \min\limits_{\lambda} f\left(x_{n-1}^i + \lambda e_n\right) \end{cases}$$

Шаг 2:

Проверим выполнение условия останова:

Если $\left|x_n^i - x_0^i\right| \le \varepsilon$, *то* $x^* = x_n^i$.

Иначе – в качестве направления оптимизации выбираем прямую, соединяющую x_n^i *и* x_0^i, *и решаем задачу оптимизации:*

$$\lambda^i = \min\limits_{\lambda} f\left(x_n^i + \lambda\left(x_n^i - x_0^i\right)\right)$$

Положим $x_0^{i+1} = x_n^i + \lambda\left(x_n^i - x_0^i\right)$ *и перейдем к шагу 1.*

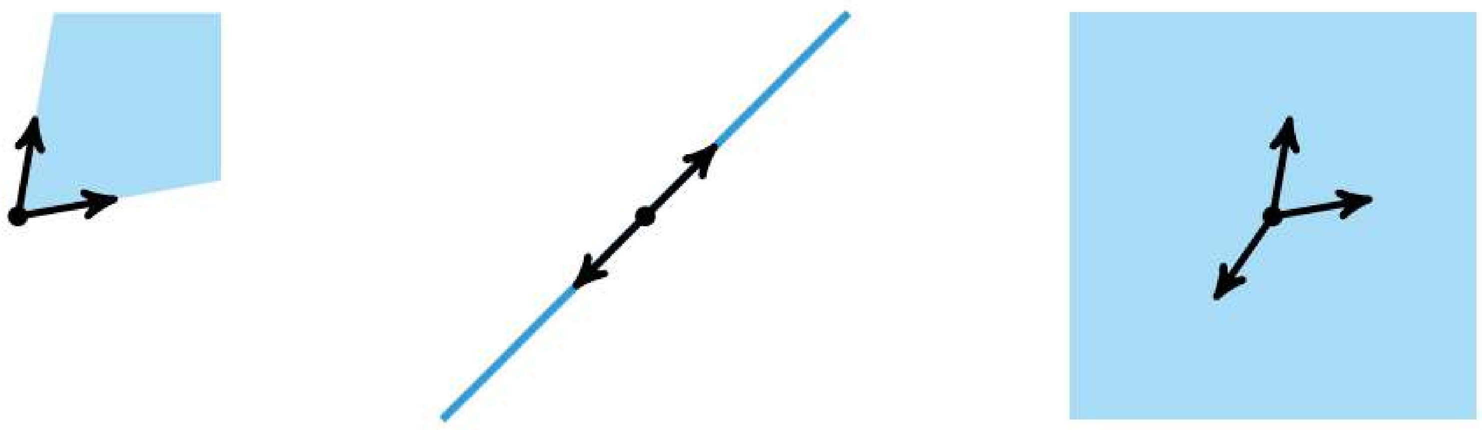
Обобщенный метод Хука-Дживса

Generalized Pattern Search

Идея подхода: *в отличие от метода Хука-Дживса, обобщенный метод предполагает выбор пробных точек исследующего поиска в произвольных направлениях* D *от начальной точки. Направление для поиска величины шага оптимизации будет выбрано из набора образцов:*

$$\mathrm{P} = \{x + \lambda d\} \quad \text{для} \quad \forall d \in D$$

При этом, выбранные направления $D = \{d^{(1)}, d^{(2)}, \dots, d^{(m)}\}$ *должны составлять базис в* $\mathbb{R}^n$*, чтобы любая точка могла быть представлена как линейная комбинация направлений.*

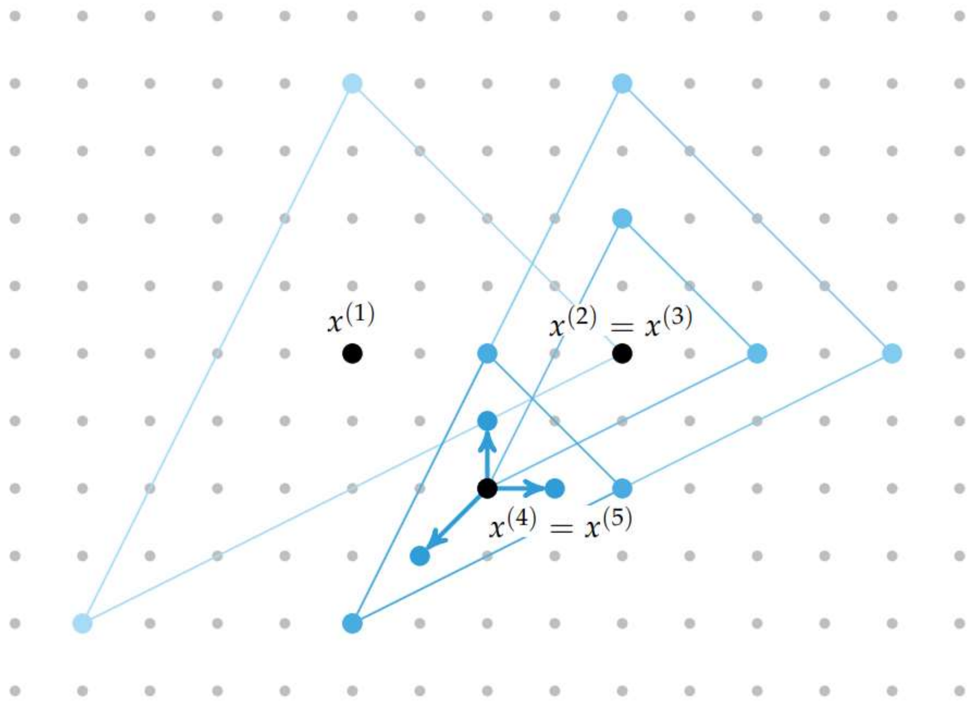


Обобщенный метод Хука-Дживса

Generalized Pattern Search

Идея подхода: *в отличие от метода Хука-Дживса, обобщенный метод предполагает выбор пробных точек исследующего поиска в произвольных направлениях* D *от начальной точки. Направление для поиска величины шага оптимизации будет выбрано из набора образцов:*

$$\mathrm{P} = \{x + \lambda d\} \quad \text{для} \quad \forall d \in D$$



Метод Хука-Дживса без одномерной оптимизации

Идея подхода: *выполняем метод Хука-Дживса, определив шаг спуска константой* λ. *Находим пробные точки покоординатно с шагом* λ *в положительном и отрицательном направлении. Если при заданном значении шага не удается обеспечить уменьшение целевой функции в пробных точках, уменьшаем шаг в* γ *раз. Выполняем алгоритм, пока выполнено условие* $\lambda > \varepsilon$.

Подготовительный этап:

Зададим начальное приближение x_0^1, *начальное значение целевой функции* $f^0 = f(x_0^1)$ *и точность расчета* ε.

Шаг 1:

Для $i = 1 \dots n$ *определяем пробные точки и значения целевой функции в них:*

$$\begin{cases} x_{1-}^i = x_0^i - \lambda e_1 \\ x_{1+}^i = x_0^i + \lambda e_1 \\ \dots \\ x_{n-}^i = x_0^i - \lambda e_n \\ x_{n+}^i = x_0^i + \lambda e_n \end{cases} \quad \text{и} \quad \begin{cases} f_{1-}^i = f\left(x_{1-}^i\right) \\ f_{1+}^i = f\left(x_{1+}^i\right) \\ \dots \\ f_{n-}^i = f\left(x_{n-}^i\right) \\ f_{n+}^i = f\left(x_{n+}^i\right) \end{cases}, \text{ где } \begin{cases} e_1 = (1, 0, 0, \dots, 0) \\ \dots \\ e_n = (0, 0, 0, \dots, 1) \end{cases}$$

Шаг 2:

Определим $f^i = \min\left(f_{1-}^i, f_{1+}^i, \dots, f_{n-}^i, f_{n+}^i\right)$ *и соответствующую точку* x^i.

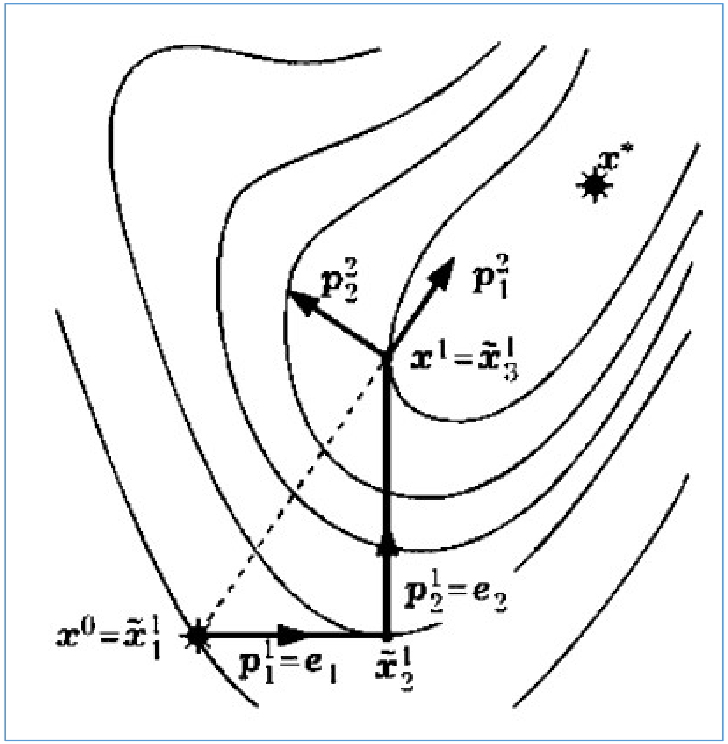
Если $f^i \geq f^{i-1}$, *уменьшаем* λ *в* γ *раз. Если выполнено* $\lambda < \varepsilon$, *получаем решение* $x^* = x^i$. *Иначе возвращаемся к шагу 1.*

Если $f^i < f^{i-1}$, *определяем* $x_0^{i+1} = x^i$ *и возвращаемся к шагу 1.*

Метод Розенброка

Rosenbrock's method

Идея подхода: *выбираем систему ортогональных направлений* $S^0 = \{s_1^0, s_2^0, \ldots, s_n^0\}$ *. По каждому из направлений находим минимальное значение и получаем следующую точку приближения к точке минимума (как в методе Гаусса-Зейделя). Затем поворачиваем систему ортогональных направлений так, чтобы одна из осей совпала с направлением полного перемещения (процесс Грама-Шмидта), и повторяем поиск по каждому из направлений.*



Метод Розенброка

Rosenbrock's method

Идея подхода: *выбираем систему ортогональных направлений* $S^0 = \{s_1^0, s_2^0, \dots, s_n^0\}$. *По каждому из направлений находим минимальное значение и получаем следующую точку приближения к точке минимума (как в методе Гаусса-Зейделя). Затем поворачиваем систему ортогональных направлений так, чтобы одна из осей совпала с направлением полного перемещения (процесс Грама-Шмидта), и повторяем поиск по каждому из направлений.*

Подготовительный этап:

Зададим начальное приближение x_0^1 *и точность расчета* ε.

Шаг 1:

Рассчитаем значения на текущей итерации:

$$\begin{cases} x_1^i = x_0^i + \lambda_1^i s_1^i \\ \dots \\ x_n^i = x_{n-1}^i + \lambda_n^i s_n^i \end{cases}, \text{ где } \begin{cases} \lambda_1^i = \min\limits_{\lambda} f\left(x_0^i + \lambda e_1\right) \\ \dots \\ \lambda_n^i = \min\limits_{\lambda} f\left(x_{n-1}^i + \lambda e_n\right) \end{cases}$$

Шаг 2:

Проверим выполнение условия останова:

Если $\left|x_n^i - x_0^i\right| \le \varepsilon$, *то* $x^* = x_n^i$.

Иначе – выбираем новую систему направлений по соотношениям:

$$s_i = \frac{b_i}{\left\|b_i\right\|}, \text{ где } a_i = \begin{cases} s_i, \lambda_i = 0 \\ \sum\limits_{j=i}^{n} \lambda_j s_j, \lambda_i \ne 0 \end{cases}, \quad b_i = \begin{cases} a_i, i = 1 \\ a_i - \sum\limits_{j=1}^{i-1} \left(a_j, s_j\right) s_j, i \ge 2 \end{cases}$$

Затем переходим к шагу 1.

Метод Пауэлла

Powell's method

Идея подхода: *поиск идет по обновляемому набору направлений. После нахождения промежуточного приближения к точке минимума, набор направлений обновляют. При обновлении удаляют наиболее старое направление и добавляют новое – соединяющее предыдущую (начальную) точку и найденную точку приближения к точке минимума.*

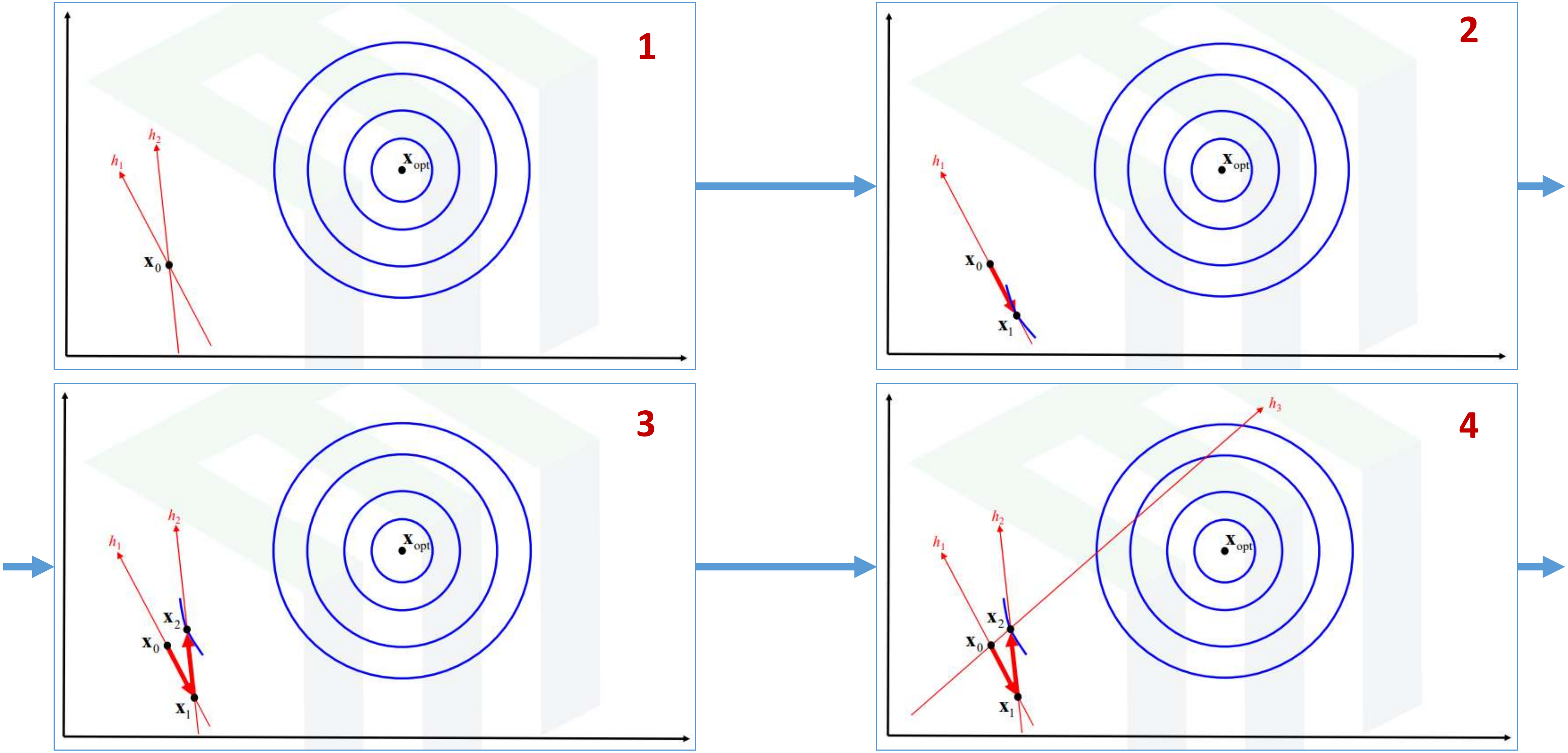
$$\boldsymbol{x}_i = line\_search\left(f, \boldsymbol{x}_{i-1}, \boldsymbol{e}_i\right), i = 1 \ldots n$$

$$\begin{cases} \boldsymbol{e}_i = \boldsymbol{e}_{i+1}, i = 1 \ldots (n-1) \\ \boldsymbol{e}_n = \boldsymbol{x}_n - \boldsymbol{x}_0 \end{cases}$$

Алгоритм выполняют до достижения условия останова. При этом можно показать, что для функций, имеющих квадратичный вид, метод Пауэлла обеспечивает сходимость к точке минимума за число шагов, не превышающее размерности задачи.

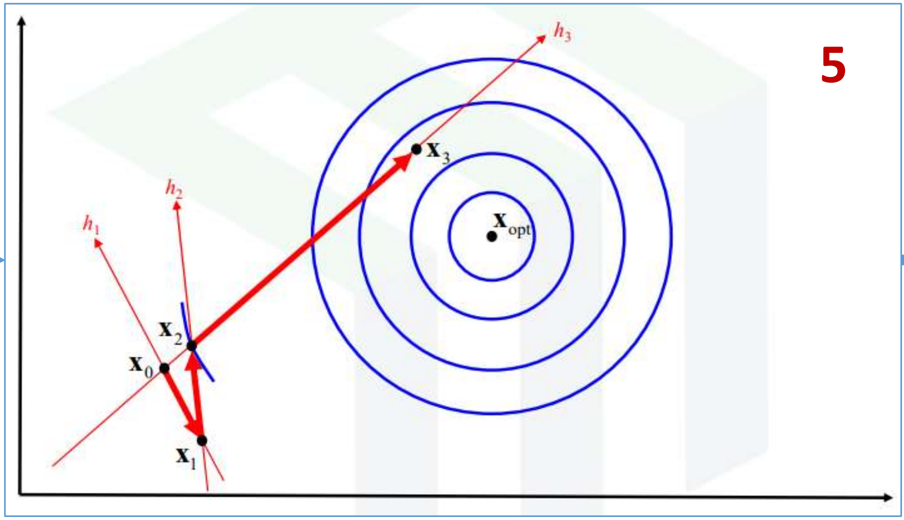
Метод Пауэлла

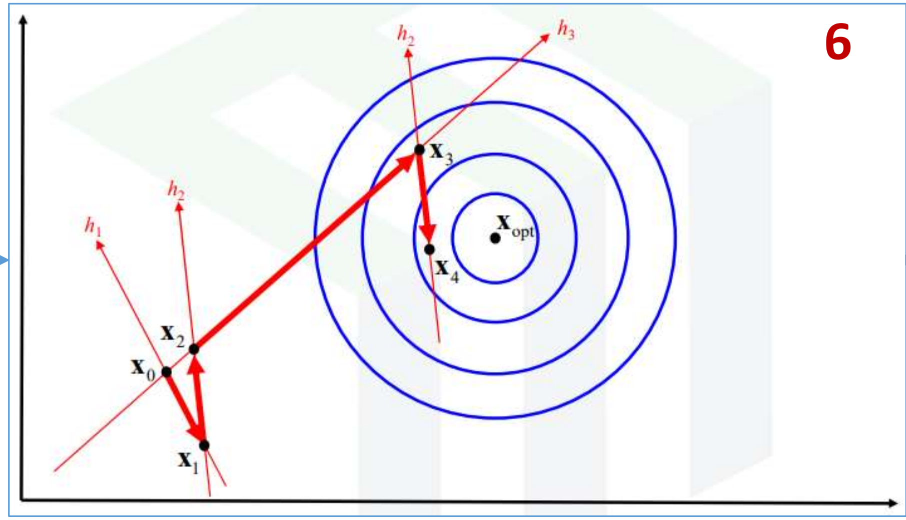
Пример

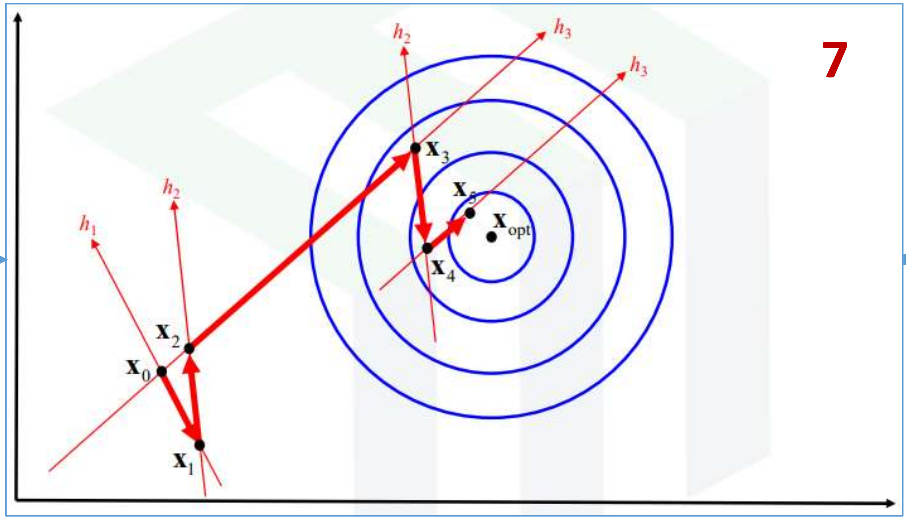


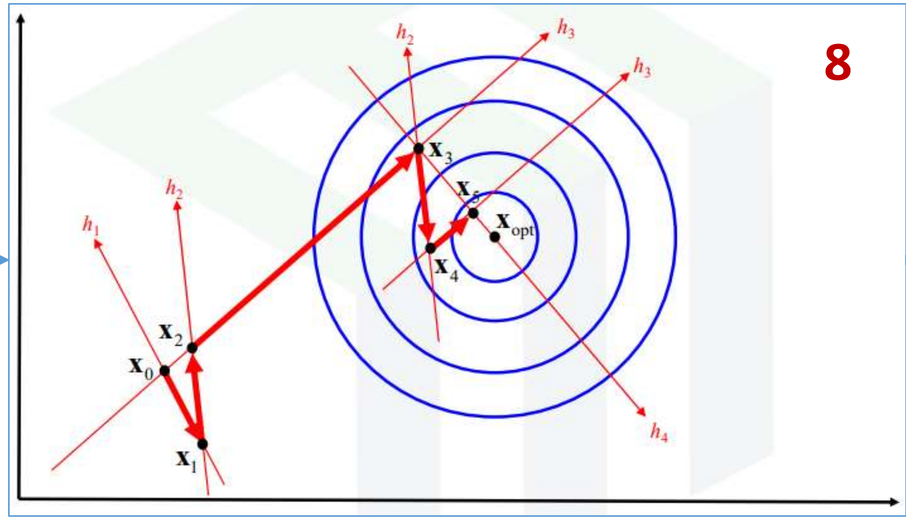
Метод Пауэлла

Пример









Метод Пауэлла

Пример

Идея подхода: *поиск идет по обновляемому набору направлений. После нахождения промежуточного приближения к точке минимума, набор направлений обновляют. При обновлении удаляют наиболее старое направление и добавляют новое – соединяющее предыдущую (начальную) точку и найденную точку приближения к точке минимума.*