Поиск точки излома

часто можно столкнуться с задачей поиска точки, где кривая резко меняет характер зависимости. Здесь мы будем рассматривать переход от плато к линейной зависимости (см Рис. 1), обобщение на более высокие порядки зависимости не требует много усилий. Для решения будем использовать классический метод наименьших квадратов. Кроме этого мы покажем как учесть ошибки данных, и как оценить ошибку на саму точку излома.

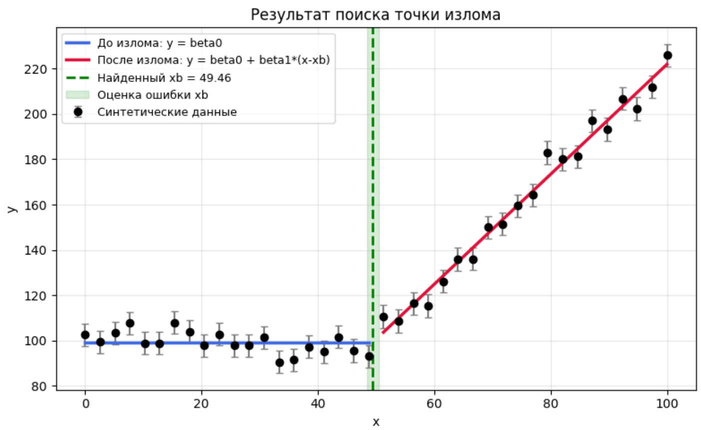

Рис. 1: Иллюстрация задачи поиска точки излома на синтезированных данных, истинная точка излома $x_b = 50$.

Пусть x_n, $n = 0, 1, 2, ..., N-1$ - данные, а σ_n - их ошибки. Используемая модель излома: слева от точки излома x_b модель является константой β_0, а справа - линейной с параметрами β_1 и β_0:

$$y(x) = \begin{cases} \beta_0, & x < x_b, \\ \beta_0 + \beta_1(x - x_b), & x \geq x_b. \end{cases}$$

Точка излома x_b находится минимизацией функции $\chi^2(x_b, \beta_0, \beta_1)$:

$$\chi^2(x_b, \beta_0, \beta_1) = \sum_{x_i < x_b} \frac{(y_i - \beta_0)^2}{\sigma_i^2} + \sum_{x_i \geq x_b} \frac{[y_i - \beta_0 - \beta_1(x_i - x_b)]^2}{\sigma_i^2}. \tag{1}$$

Ниже мы сделаем аналитически минимизацию по параметрам β_0, β_1, и найдём ошибку на точку излома; минимизация по параметру x_b выполняется численно.

Для решения нам понадобится, матрица вторых производных (матрица Гессе), которая имеет вид

$$H = \begin{pmatrix} \frac{\partial^2\chi^2}{\partial x_b^2} & \frac{\partial^2\chi^2}{\partial x_b \partial\beta_0} & \frac{\partial^2\chi^2}{\partial x_b \partial\beta_1} \\ \frac{\partial^2\chi^2}{\partial\beta_0 \partial x_b} & \frac{\partial^2\chi^2}{\partial\beta_0^2} & \frac{\partial^2\chi^2}{\partial\beta_0 \partial\beta_1} \\ \frac{\partial^2\chi^2}{\partial\beta_1 \partial x_b} & \frac{\partial^2\chi^2}{\partial\beta_1 \partial\beta_0} & \frac{\partial^2\chi^2}{\partial\beta_1^2} \end{pmatrix} = \begin{pmatrix} m_{00} & m_{01} & m_{02} \\ m_{10} & m_{11} & m_{12} \\ m_{20} & m_{21} & m_{22} \end{pmatrix}.$$

Оценка параметров β_0 и β_1.

Сперва мы можем выполнить минимизацию по параметрам β_0 и β_1, используя нормальные уравнения метода наименьших квадратов

$$(X^T W X)\hat{\beta} = X^T W y \tag{2}$$

где матрицы имеют следующий вид:

$$X = \begin{pmatrix} 1 & x_1 - x_b \\ 1 & x_2 - x_b \\ \vdots & \vdots \\ 1 & x_n - x_b \end{pmatrix},$$

матрица весов:

$$W = \mathrm{diag}\left(\frac{1}{\sigma_1^2}, \frac{1}{\sigma_2^2}, \dots, \frac{1}{\sigma_n^2}\right).$$

Если рассмотреть более подробнее, левые и правые части нормальных уравнений 2 имеют вид:

$$X^T W X = \begin{pmatrix} \sum w_i & \sum w_i(x_i - x_b) \\ \sum w_i(x_i - x_b) & \sum w_i(x_i - x_b)^2 \end{pmatrix} = \begin{pmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \end{pmatrix} \equiv M_{\beta\beta},$$

$$X^T W y = \begin{pmatrix} \sum w_i y_i \\ \sum w_i y_i (x_i - x_b) \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}.$$

Решение уравнений можно получить с помощью правила Крамера.
Сначала вычисляется определитель

$$\det(M_{\beta\beta}) = m_{11} m_{22} - m_{12}^2.$$

а затем находятся оценки параметров:

$$\hat{\beta}_0 = \frac{m_{22} b_1 - m_{12} b_2}{\det(M_{\beta\beta})}, \tag{3}$$

$$\hat{\beta}_1 = \frac{m_{11} b_2 - m_{12} b_1}{\det(M_{\beta\beta})}.$$

Ошибка на точку излома

Чтобы оценить ошибку на параметр x_b, рассматривается ковариационная матрица $V = H^{-1}$, дисперсия точки излома определяется элементом $\sigma_{x_b}^2 = [H^{-1}]_{00}$.

Используя дополнение Шура, можно получить явное выражение

$$\sigma_{x_b}^2 = [H^{-1}]_{00} = S^{-1} = \left(m_{00} - m_{0\beta}^T H_{\beta\beta}^{-1} m_{0\beta}\right)^{-1}$$

$$\sigma_{x_b}^2 = \left(m_{00} - \begin{bmatrix} m_{01} & m_{02} \end{bmatrix} \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \end{bmatrix}^{-1} \begin{bmatrix} m_{01} \\ m_{02} \end{bmatrix} \right)^{-1} \tag{4}$$

в развернутом виде

$$\sigma_{x_b}^2 = \left(m_{00} - \frac{m_{01}(m_{02} m_{12} - m_{01} m_{22}) + m_{02}(m_{01} m_{12} - m_{02} m_{11})}{\det(M_{\beta\beta})} \right)^{-1}.$$

На данном этапе мы сделали минимизацию по β_1, β_0, последний шаг - выполнить минимизацию по x_b и оценить её ошибку, что выполняется численно. Реализацию данного алгоритма на Python можно найти по ссылки на репозиторий GitHub: https://github.com/fedsvir/break_point.git