STAT 153 & 248 - Time Series
Lecture Ten

Fall 2025, UC Berkeley

Aditya Guntuboyina

October 1, 2025

1 Background

We will now start our third main topic: High-dimensional linear regression. We shall motivate our basic high-dimensional linear regression model starting from the change of slope (or broken stick regression) model that we previously considered.

The change of slope (or broken stick regression) model is given by:

$$y_t = \beta_0 + \beta_1 t + \beta_2 (t - c)_+ + \epsilon_t. \tag{1}$$

Recall that $(t-c)_+$ equals 0 if $t \leq c$ and $t - c$ if $t \geq c$. We shall also sometimes write $(t-c)_+ = \mathrm{ReLU}(t-c)$. The model (1) says that for times $t \leq c$, the slope of the regression line is β_1 while for $t > c$, the slope changes to $\beta_1 + \beta_2$. Note β_2 represents the change in slopes (after $t = c$ and before $t = c$).

Generalizations of model (1) can be obtained by having k points of change of slope (instead of just one):

$$y_t = \beta_0 + \beta_1 t + \beta_2 \mathrm{ReLU}(t - c_1) + \beta_3 \mathrm{ReLU}(t - c_2) + \cdots + \beta_{k+1} \mathrm{ReLU}(t - c_k) + \epsilon_t. \tag{2}$$

We use least squares to estimate the unknown parameters $c_1, \ldots, c_k, \beta_0, \beta_1, \ldots, \beta_{k+1}$ in this model. In the previous lectures, we recommended the method where we first compute the function:

$$RSS(c_1, \ldots, c_k) := \min_{\beta_0, \ldots, \beta_{k+1}} \sum_{t=1}^{n} \left(y_t - \beta_0 - \beta_1 t - \beta_2 \mathrm{ReLU}(t - c_1) - \cdots - \beta_{k+1} \mathrm{ReLU}(t - c_k)\right)^2$$

and then attempt to minimize $RSS(c_1, \ldots, c_k)$ by some grid-based search jointly over $c_1, \ldots, c_k$ to estimate $c_1, \ldots, c_k$. Once $c_1, \ldots, c_k$ are estimated by $\hat{c}_1, \ldots, \hat{c}_k$, we estimate the remaining parameters $\beta_0, \ldots, \beta_{k+1}$ by least squares in the linear regression model obtained by fixing each c_j by $\hat{c}_j$.

This method is computationally inefficient especially when k is not small (even when $k \geq 3$). An alternative method of obtaining the least squares estimates is to directly minimize the least squares objective with respect to all parameters:

$$\min_{\beta_0, \ldots, \beta_{k+1}, c_1, \ldots, c_k} \sum_{t=1}^{n} \left(y_t - \beta_0 - \beta_1 t - \beta_2 \mathrm{ReLU}(t - c_1) - \cdots - \beta_{k+1} \mathrm{ReLU}(t - c_k)\right)^2. \tag{3}$$

This is a non-trivial optimization problem but several optimization libraries provide functions which can be employed for solving this. One option is the PyTorch library in Python which uses first order optimization algorithms (gradient descent and related methods). These methods require a good initialization for the parameters (for $c_1, \ldots, c_k$, a natural initialization is to take equally spaced quantiles of t; once c_j's are chosen, initial values for $\beta_0, \ldots \beta_{k+1}$ can be obtained by running a linear regression with those fixed c_j's).

When k is not small, the least squares estimates (3) can lead to fitted values which overfit the data. In such cases, we need to add a regularization term to the objective function to prevent overfitting. We shall see how to do this in the case of the high-dimensional model which uses all possible c_j's.

2 High-dimensional Version of (2)

We illustrate overfitting prevention via regularization in the high-dimensional model obtained by using all possible c_j's in (2). This model is described below. Note that because our time t runs from $1, \ldots, n$, we can restrict each c_j to the open interval $(1, n)$. Indeed, when $c_j \leq 1$, the ReLU term $\text{ReLU}(t - c_j)$ coincides with the linear term $t - c_j$ (which can be absorbed in $\beta_0 + \beta_1 t$) and when $c_j \geq n$, the ReLU term $\text{ReLU}(t - c_j)$ equals zero.

In fact, it turns out that it is enough to restrict c_j to be an integer in $(1, n)$ i.e., $c_j \in \{2, \ldots, n-1\}$. This is because $\text{ReLU}(t - c_j), t = 1, \ldots, n$ for any other c_j can be rewritten as a linear combination of $\text{ReLU}(t - c_j), t = 1, \ldots, n$ for integer c_j. As an example, note that

$$\text{ReLU}(t - 5.7) = 0.3\text{ReLU}(t - 5) + 0.7\text{ReLU}(t - 6) \qquad \text{for all } t \leq 5 \text{ and } t \geq 6.$$

Allowing all possible c_j's in $\{2, \ldots, n-1\}$ leads to the model:

$$y_t = \beta_0 + \beta_1(t-1) + \beta_2\text{ReLU}(t-2) + \cdots + \beta_{n-1}\text{ReLU}(t-(n-1)) + \epsilon_t \tag{4}$$

where, as always, $\epsilon_t \overset{\text{i.i.d}}{\sim} N(0, \sigma^2)$. Note that we are writing $\beta_1(t-1)$ instead of $\beta_1 t$ for aesthetical reasons. This does not change the model in any way.

The unknown parameters in this model are $\beta_0, \beta_1, \ldots, \beta_{n-1}$ as well as σ. Here are main differences between the two models (2) and (4):

1. The model (2) will be typically used with a small value of k (such as $1, 2, 3, 4$). This makes it a low-dimensional model. On the other hand, the number of unknown parameters in (4) equals $n+1$ which is quite large. So (4) is an example of a high-dimensional model.

2. (2) is a nonlinear model because of the presence of the parameters $c_1, \ldots, c_k$. On the other hand, there are no such nonlinear parameters in (4) which makes it a linear regression model.

To summarize, (2) is a low-dimensional nonlinear regression model, while (4) is a high-dimensional linear regression model.

2.1 Least Squares in (4) interpolates data

Since (4) is a linear regression model, we can estimate the coefficients in the usual way by the MLE, or equivalently, least squares by minimizing

$$\sum_{t=1}^{n} \left(y_t - \beta_0 - \beta_1(t-1) - \beta_2 \text{ReLU}(t-2) - \cdots - \beta_{n-1}\text{ReLU}(t-(n-1))\right)^2 \tag{5}$$

over all $\beta_0, \ldots, \beta_{n-1}$. The smallest value achievable in the above minimization will be the RSS.

Because the number of coefficients equals the number of data points, the optimization (5) will give a perfect fit to the data leading to $RSS = 0$. This is because, for any data $y_1, \ldots, y_n$, it is possible to choose $\beta_0, \ldots, \beta_{n-1}$ such that

$$y_t = \beta_0 + \beta_1(t-1) + \beta_2 \text{ReLU}(t-2) + \cdots + \beta_{n-1}\text{ReLU}(t-(n-1)) \tag{6}$$

for every $t = 1, \ldots, n$. To write $\beta_0, \ldots, \beta_{n-1}$ in terms of the observed data $y_1, \ldots, y_n$, first plug in $t = 1$ in (6) to get

$$\beta_0 = y_1$$

Then plug $t = 2$ in (6) to get $y_2 = \beta_0 + \beta_1 = y_1 + \beta_1$ so that

$$\beta_1 = y_2 - y_1.$$

Then plug $t = 3$ in (6) to get $y_3 = \beta_0 + 2\beta_1 + \beta_2$. Replacing $\beta_0 = y_1$ and $\beta_1 = y_2 - y_1$, we obtain

$$\beta_2 = (y_3 - y_2) - (y_2 - y_1)$$

Continuing this way plugging $t = 4, 5, \ldots, n$, we get

$$\beta_t = (y_{t+1} - y_t) - (y_t - y_{t-1}).$$

Therefore the least squares parameter estimates for the model (4) are given by:

$$\beta_0 = y_1 \quad \beta_1 = y_2 - y_1 \quad \beta_j = (y_{j+1} - y_j) - (y_j - y_{j-1})$$

for $j = 2, \ldots, n-1$. This will lead to fitted values which coincide exactly with the observed data. Also the MLE of σ will be zero. The unbiased estimate of σ (that we previously used in linear regression) will not exist because it will equal $\sqrt{RSS/(n-p)}$ with $p = n$.

To summarize, least squares in the model (4) will overfit the data, and will not produce any useful insight from the data.

To produce useful estimates in cases where the MLE overfits, one employs the idea of regularization. In the next lecture, we will discuss two ways of doing this: Ridge regularization and LASSO regularization. We will also understand regularization from the Bayesian perspective.