## 3.11.4 → Local regression

→ overview → recall the weighted least squares algorithm:

$$\hat{\beta} = (X^T W X)^{-1} X^T W y$$

where $W$ is an $n \times n$ diagonal matrix w/ weights $w_1, \dots, w_n$ as its entries. Equivalently, the estimates minimize the expression

$$\boxed{\sum_{i=1}^{n} w_i (y_i - \hat{y}_i)^2}$$

→ The point here is that the weight $w_i$ indicates how important the $i$th observation is in determining the fit. For example, if $w_i$ is small, then the contribution of $(y_i - \hat{y}_i)^2$ to the whole expression is reduced, causing observation $i$ to be deemphasized when estimating the coefficients.

→ <u>local regression</u> implements this weighted least squares idea in an interesting way. This method is algorithmic in calculating the fitted values, & thus we cannot express the fitted curve w/ an equation. To predict the response for the covariate having a value of $x_*$, the procedure is:

→ 1) select the <u>span</u> — a # between 0 & 1 that signifies a proportion of the observations. Let $v$ denote the integer such that the span is $v/n$.

→ 2) Identify the $v$ observations whose $x$ values are the closest to $x_*$.

→ 3) assign weights $w_1, \dots, w_n$ based on each observation's closeness to $x_*$.

(a) For observations not among the $v$ closest, their weights are all 0.

(b) for observations among the $v$ closest, the weight is the largest for closest observation, & decreases until reaching 0 for the furthest observation. The weights are determined using a <u>weighting function</u>.

→ 4) Solve for the estimates $\hat{\beta}_{0*}$ & $\hat{\beta}_{1*}$ by minimizing the expression

$$\boxed{\sum_{i=1}^{n} w_i (y_i - (\hat{\beta}_{0*} + \hat{\beta}_{1*} x))^2}$$

→ 5) calculate the fitted value as $\hat{\beta}_{0*} + \hat{\beta}_{1*} x_*$.

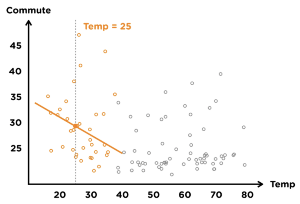
→ Therefore, local regression computes $\hat{y}$ for an input $x_*$ by fitting a line w/ only the observations that are close to the input, & also favors those that are closer. Realize that coefficient estimates depend on the weights, which in turn depend on the choice of $x_*$, so a different input produces a different fitted line. However, the fitted lines are not the end goal; it is only the fitted values that we desire.

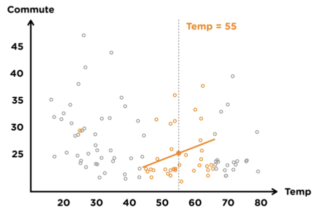
→ Although not found in the official reading, an example of an appropriate weighting function for the $v$ closest observations is

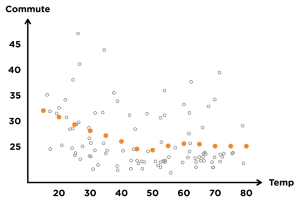
$$\left(1 - \left|\frac{x_* - x}{x_* - x_{(v)}}\right|^3\right)^3 \quad \text{(don't have to sum to 1} \Rightarrow \textstyle\sum w_i \neq 1 \text{ necessarily)}$$

where $x_{(v)}$ is the $x$ value of the furthest observation among the $v$ closest to the input. This ensures a weight of 0 for that furthest observation, i.e. when $x = x_{(v)}$

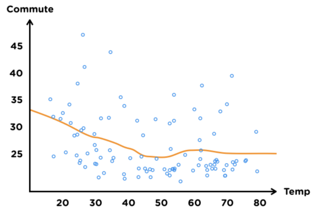
→ Example → span $= 0.35 \Rightarrow v = 35$ $(n = 100)$




→ By repeating the procedure for many different $x_*$ inputs, we obtain the fitted curve by connecting the individual fitted values




→ there are other essential points about local regression:

→ A small span means each fitted line depends on a few observations, resulting in fitted values that are highly sensitive to data patterns in narrow intervals. Conversely, a large span allows more observations to influence each fitted line, so the fitted values will be more stable. In other words, the span is inversely related to flexibility. Cross-validation can help to determine the proper flexibility level.

→ Instead of lines to subsets of observations, we could choose to do a constant or polynomial fits. This also impacts the regression's flexibility, but not as much as changing the span.

→ Higher dimensions

→ Although we mainly focus on using one covariate, local regression can be extended to multiple covariates. If we consider $p$ covariates, then the procedure identifies the closest observations to the input in $p$-dimensional space. However, local regression is sensitive to the curse of dimensionality, meaning it does not perform well in high dimensions.

---

## 3.11.5 → Generalized Additive Models

→ overview → we may extend all of the concepts discussed thus far for one covariate to $g$ recorded variables. A generalized additive model (GAM) does this by assuming a model equation of

$$\boxed{Y = \beta_0 + f_1(x_1) + \cdots + f_g(x_g) + \varepsilon}$$

→ For $m = 1, \dots, g$, the explanatory variable $x_m$ contributes to the mean response through a function $f_m$ doing so independently of the other explanatory variables. $f_m$ may have any functional form of regressions w/ one explanatory variable that we discussed the previous sections, & the functions are also free to vary across the $g$ explanatory variables. For example, given two covariates $(x_1, x_2)$ & one factor $(x_3)$, we can model

→ $f_1$ as a cubic spline function,
→ $f_2$ as a smoothing spline function, &
→ $f_3$ as a function w/ dummy variables

→ Incorporating these individual models in the larger generalized additive model gives a sense of <u>models within models</u>. From the perspective that the mean response $f(x_1, \dots, x_g)$ is a function of $f_1, \dots, f_g$, we can likewise say there is a <u>functions w/in functions structure</u>.

→ If $f_1, \dots, f_g$ are all based on regressions that use OLS (i.e. not smoothing spline or local regression), then the GAM is a MLR. In other words, each $f_m(x_m)$ is a sum of basis functions w/ their coefficients (no intercept since $\beta_0$ handles it for the entire model), with all regression coefficients estimated by OLS.

→ However, if at least one of $f_1, \dots, f_g$ employs a smoothing spline or local regression, then fitting the GAM requires a different approach such as <u>backfitting</u>. In summary, backfitting runs separate fits for each $x_m$ while fixing the other variables, & repeatedly uses a past fit to inform the next fit until the overall fit converges.

→ The additive of the model means the effect of each $x_m$ on the response can be investigated individually, assuming the other variables are held constant. This includes conducting appropriate statistical inference. On the other hand, its additivity is also a restriction since no interactions are considered. One solution is to manually include interaction terms to the model, but it would no longer constitute a GAM.