Calculating a weighted regression using matrix algebra

Below we are solving a very simple linear regression to illustrate how to compute a weighted regression using matrix algebra.
In this case, we know the weights already. However, in some more common instances, you may have to estimate the weights by perhaps using the standard deviations of the IV's in the model

$x_i :=$	$y_i :=$	$w_i :=$
0.5578196	18.63654	1
2.0217271	103.49646	0.90334913
2.5773252	150.35391	0.75988974
3.4140288	190.51031	0.42621714
4.3014084	208.70115	0.08686171
4.7448394	213.71135	0.01072308
5.1073781	228.49353	0

$$X\_MATRIX := \begin{pmatrix} 1 & x_0 \\ 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ 1 & x_4 \\ 1 & x_5 \\ 1 & x_6 \end{pmatrix} \qquad X\_MATRIX = \begin{pmatrix} 1 & 0.558 \\ 1 & 2.022 \\ 1 & 2.577 \\ 1 & 3.414 \\ 1 & 4.301 \\ 1 & 4.745 \\ 1 & 5.107 \end{pmatrix}$$

The matrix containing the weights will be a diagonal matrix, where the elements of the main diagonal have the weights to be used in the estimation

$$WEIGHT := \begin{pmatrix} w_0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & w_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & w_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & w_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & w_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & w_6 \end{pmatrix} \qquad WEIGHT = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.903 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.76 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.426 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.087 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.011 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

The equations used to estimate the coefficients are very similar to those used in solving the Ordinary Least Squares (OLS):

$$XTX\_1 := \left(X\_MATRIX^T \cdot WEIGHT \cdot X\_MATRIX\right)^{-1} \qquad XTX\_1 = \begin{pmatrix} 1.339 & -0.525 \\ -0.525 & 0.269 \end{pmatrix}$$

$$XTy := \left(X\_MATRIX^T \cdot WEIGHT \cdot y\right) \qquad XTy = \begin{pmatrix} 328.001 \\ 859.945 \end{pmatrix}$$

Notice that the only difference with the equations we use to solve the OLS is the fact that we add the weight matrix (WEIGHT) to both X^TX^{-1} and X^Ty

$$B := XTX\_1 \cdot XTy \qquad B = \begin{pmatrix} -12.337 \\ 59.033 \end{pmatrix}$$

Using weighted regression in Local Linear regression (LOESS/LOWESS)

When estimating a Local linear regression (or LOcally WEighted regreSSion, thus the name), the weights in the WEIGHT matrix represent the closeness to the value being estimated. At every point in the estimation, a low-level polynomial regression (usually degree 1 or 2) is fitted to a subset of the data that is close to the point being estimated. Usually, the weight of 1 is assigned to the observation being fitted. And for the purposes of fitting the new lines, it is also important to find what is the predicted value (the y-hat) for the value being fitted.

$b0 := B_0 \qquad b0 = -12.337 \qquad b1 := B_1 \qquad b1 = 59.033$

$y\_hat := b0 + b1 \cdot x$

$$x = \begin{pmatrix} 0.558 \\ 2.022 \\ 2.577 \\ 3.414 \\ 4.301 \\ 4.745 \\ 5.107 \end{pmatrix} \quad y = \begin{pmatrix} 18.637 \\ 103.496 \\ 150.354 \\ 190.51 \\ 208.701 \\ 213.711 \\ 228.494 \end{pmatrix} \quad w = \begin{pmatrix} 1 \\ 0.903 \\ 0.76 \\ 0.426 \\ 0.087 \\ 0.011 \\ 0 \end{pmatrix} \quad y\_hat = \begin{pmatrix} 20.593 \\ 107.012 \\ 139.811 \\ 189.204 \\ 241.589 \\ 267.766 \\ 289.167 \end{pmatrix}$$

Thus, for the original data, when x = 0.558, and y = 18.637, the predicted value is now: 20.593. Below, we illustrate how to compute the value for x = 2.022:
In order to compute the new value, we change the weight to reflect the new values:

$w_i :=$

0.712642
1
0.982589
0.748942
0.212501
0.030572
0

$$dif := \frac{0.5578196 - 2.021727}{3.085651} \qquad dif = -0.474$$

$$wi := \left[1 - \left(|dif|\right)^3\right]^3 \qquad wi = 0.713$$

Notice that the scaling procedure we use here matches what is called in the Excel spreadsheet "***based on normalization***". You must estimate the maximum difference and then you use that in the denominator.

$$dif := \frac{0.5578196 - 2.021727}{5} \qquad dif = -0.293$$

$$wi := \left[1 - \left(|dif|\right)^3\right]^3 \qquad wi = 0.927$$

The proposal suggested in Guo & Fraser is to use a percent of the cases (what they call XN. in the Excel spreadsheet I illustrated estimates using X= 0.25), and the equation to the left illustrates how to calculate that value

$$WEIGHT := \begin{pmatrix} w_0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & w_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & w_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & w_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & w_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & w_6 \end{pmatrix}$$

$$WEIGHT = \begin{pmatrix} 0.713 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.983 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.749 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.213 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.031 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$XTX\_1 := \left(X\_MATRIX^{T} \cdot WEIGHT \cdot X\_MATRIX\right)^{-1} \qquad XTX\_1 = \begin{pmatrix} 1.533 & -0.543 \\ -0.543 & 0.234 \end{pmatrix}$$

$$XTy := \left(X\_MATRIX^{T} \cdot WEIGHT \cdot y\right) \qquad XTy = \begin{pmatrix} 458.078 \\ 1.306 \times 10^{3} \end{pmatrix}$$

$$B := XTX\_1 \cdot XTy \qquad B = \begin{pmatrix} -7.176 \\ 56.554 \end{pmatrix}$$

$$b0 := B_0 \qquad b0 = -7.176 \qquad b1 := B_1 \qquad b1 = 56.554$$

$$y\_hat := b0 + b1 \cdot x$$

$$x = \begin{pmatrix} 0.558 \\ 2.022 \\ 2.577 \\ 3.414 \\ 4.301 \\ 4.745 \\ 5.107 \end{pmatrix} \quad y = \begin{pmatrix} 18.637 \\ 103.496 \\ 150.354 \\ 190.51 \\ 208.701 \\ 213.711 \\ 228.494 \end{pmatrix} \quad w = \begin{pmatrix} 0.713 \\ 1 \\ 0.983 \\ 0.749 \\ 0.213 \\ 0.031 \\ 0 \end{pmatrix} \quad y\_hat = \begin{pmatrix} 24.371 \\ 107.16 \\ 138.582 \\ 185.9 \\ 236.085 \\ 261.163 \\ 281.666 \end{pmatrix}$$

This time, when x = 2.022, and y = 103.496, the predicted value is: 107.16

Estimating weighted linear regression using R

```
> w <- c(1, 0.903349126, 0.75988972, 0.426217146, 0.086861708, 0.010723079, 0)
> y <- c(18.63654, 103.49646, 150.35391, 190.51031, 208.70115, 213.71135, 228.49353)
> x <- c(0.5578196, 2.0217271, 2.5773252, 3.4140288, 4.3014084, 4.7448394, 5.1073781)
> reg1 <- lm(y~x, weight=w)
> summary(reg1)

Call:
lm(formula = y ~ x, weights = w)

Weighted Residuals:
      1       2       3       4       5       6       7
-1.9565 -3.3413  9.1907  0.8529 -9.6927 -5.5975  0.0000

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -12.337      8.686   -1.42  0.22856
x             59.033      3.893   15.16  0.00011 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7.508 on 4 degrees of freedom
Multiple R-squared: 0.9829,    Adjusted R-squared: 0.9786
F-statistic:   230 on 1 and 4 DF,  p-value: 0.0001103

> pred1 <- fitted.values(reg1)
> pred1
        1         2         3         4         5         6         7
 20.59302 107.01200 139.81067 189.20386 241.58862 267.76572 289.16749

## calculating another value (notice that the weights are computed in excel

> w <- c(0.712642, 1, 0.982589, 0.748942, 0.212501, 0.030572, 0)
> y <- c(18.63654, 103.49646, 150.35391, 190.51031, 208.70115, 213.71135,
228.49353)
> x <- c(0.5578196, 2.0217271, 2.5773252, 3.4140288, 4.3014084, 4.7448394,
5.1073781)
> reg1 <- lm(y~x, weight=w)
> summary(reg1)

Call:
lm(formula = y ~ x, weights = w)

Weighted Residuals:
      1       2       3       4       5       6       7
 -4.841  -3.664  11.669   3.990 -12.623  -8.297   0.000

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   -7.176     12.646  -0.567 0.600728
x             56.554      4.938  11.454 0.000332 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 10.21 on 4 degrees of freedom
Multiple R-squared: 0.9704,    Adjusted R-squared: 0.963
F-statistic: 131.2 on 1 and 4 DF,  p-value: 0.0003316

> pred1 <- fitted.values(reg1)
> pred1
        1         2         3         4         5         6         7
 24.37072 107.16031 138.58151 185.90031 236.08503 261.16275 281.66571
```