Exercise: Sufficient statistics for online linear regression

(Source: Jaakkola.) Consider fitting the model $\hat{y} = w_0 + w_1 x$ using least squares. Unfortunately we did not keep the original data, x_i, y_i, but we do have the following functions (statistics) of the data:

$$\overline{x}^{(n)} = \frac{1}{n}\sum_{i=1}^{n} x_i, \quad \overline{y}^{(n)} = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{1}$$

$$C_{xx}^{(n)} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2, \quad C_{xy}^{(n)} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y}), \quad C_{yy}^{(n)} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \overline{y})^2 \tag{2}$$

1. What are the minimal set of statistics that we need to estimate w_1?

2. What are the minimal set of statistics that we need to estimate w_0?

3. Suppose a new data point, x_{n+1}, y_{n+1} arrives, and we want to update our sufficient statistics without looking at the old data, which we have not stored. (This is useful for online learning.) Show that we can do this for $\overline{x}$ as follows.

$$\overline{x}^{(n+1)} \triangleq \frac{1}{n+1}\sum_{i=1}^{n+1} x_i = \frac{1}{n+1}\left(n\overline{x}^{(n)} + x_{n+1}\right) \tag{3}$$

$$= \overline{x}^{(n)} + \frac{1}{n+1}(x_{n+1} - \overline{x}^{(n)}) \tag{4}$$

   This has the form: new estimate is old estimate plus correction. We see that the size of the correction diminishes over time (i.e., as we get more samples). Derive a similar expression to update $\overline{y}$.

4. Show that one can update $C_{xy}^{(n+1)}$ recursively using

$$C_{xy}^{(n+1)} = \frac{1}{n+1}\left[x_{n+1}y_{n+1} + nC_{xy}^{(n)} + n\overline{x}^{(n)}\overline{y}^{(n)} - (n+1)\overline{x}^{(n+1)}\overline{y}^{(n+1)}\right] \tag{5}$$

   Derive a similar expression to update C_{xx}.

5. Implement the online learning algorithm, i.e., write a function of the form `[w,ss] = linregUpdateSS(ss, x, y)`, where x and y are scalars and ss is a structure containing the sufficient statistics.

6. Plot the coefficients over "time", using the dataset in linregDemo1.m. Specifically, use

```
[x,y] = polyDataMake('sampling','thibaux')
```

   Check that they converge to the solution given by the batch (offline) learner (i.e, ordinary least squares). Your result should look like Figure 1.

Turn in your derivation, code and plot.

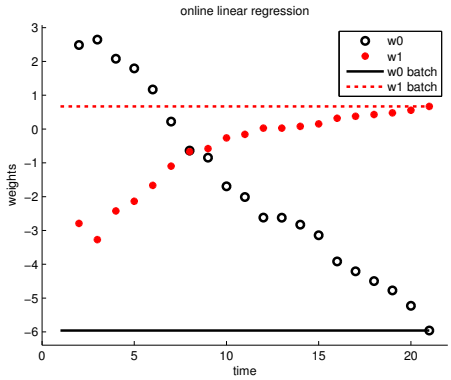

Figure 1: Regression coefficients over time.