Welford’s method for computing variance

The standard deviation is a measure of how much a dataset differs from its mean; it tells us how dispersed the data are. A dataset that’s pretty much clumped around a single point would have a small standard deviation, while a dataset that’s all over the map would have a large standard deviation.

Given a sample $x_1, \ldots, x_N$, the standard deviation is defined as the square root of the *variance*:

$$s^2 = \frac{\sum_{i=1}^N (x_i - \bar{x})^2}{N-1}, s = \sqrt{s^2}$$

Here $\bar{x}$ is the mean of the sample: $\bar{x} = \frac{1}{N}\sum_{i=1}^N x_i$. The definition can be converted directly into an algorithm that computes the variance and standard deviation in two passes: compute the mean in one pass over the data, and then do a second pass to compute the squared differences from the mean. Doing two passes is not ideal though, and it can be impractical in some settings. For example, if the samples are generated by a random simulation it may be prohibitively expensive to store samples just so you can do a second pass over them.

A easy computation gives us the following identity that suggests a method for computing the variance in a single pass, by simply accumulating the sums of x_i and x_i^2:

$$s^2 = \frac{\sum_{i=1}^N (x_i^2 - 2\bar{x}x_i + \bar{x}^2)}{N-1} = \frac{\sum x_i^2 - 2N\bar{x}^2 + N\bar{x}^2}{N-1} = \frac{\sum x_i^2 - N\bar{x}^2}{N-1}$$

Pseudocode for a one-pass variance computation could then look like:

```
variance(samples):
  sum := 0
  sumsq := 0
  for x in samples:
    sum := sum + x
    sumsq := sumsq + x**2
  mean := sum/N
  return (sumsq - N*mean**2)/(N-1)
```

Now that you've seen it, *do not use this method to compute variance **ever***. This is one of those cases where a mathematically simple approach turns out to give wrong results for being numerically unstable. In simple cases the algorithm will seem to work fine, but eventually you will find a dataset that exposes the problem with the algorithm. If the variance is small compared to the square of the mean, and computing the difference leads *catastrophic cancellation* where significant leading digits are eliminated and the result has a large relative error. In fact, you may even compute a negative variance, which is mathematically impossible.

Welford's method is a usable single-pass method for computing the variance. It can be derived by looking at the differences between the sums of squared differences for N and N-1 samples. It's really surprising how simple the difference turns out to be:

$$
\begin{aligned}
&(N-1)s_N^2 - (N-2)s_{N-1}^2 \\
&= \sum_{i=1}^{N}(x_i-\bar{x}_N)^2 - \sum_{i=1}^{N-1}(x_i-\bar{x}_{N-1})^2 \\
&= (x_N-\bar{x}_N)^2 + \sum_{i=1}^{N-1}\left((x_i-\bar{x}_N)^2-(x_i-\bar{x}_{N-1})^2\right) \\
&= (x_N-\bar{x}_N)^2 + \sum_{i=1}^{N-1}(x_i-\bar{x}_N+x_i-\bar{x}_{N-1})(\bar{x}_{N-1}-\bar{x}_N) \\
&= (x_N-\bar{x}_N)^2 + (\bar{x}_N-x_N)(\bar{x}_{N-1}-\bar{x}_N) \\
&= (x_N-\bar{x}_N)(x_N-\bar{x}_N-\bar{x}_{N-1}+\bar{x}_N) \\
&= (x_N-\bar{x}_N)(x_N-\bar{x}_{N-1})
\end{aligned}
$$

This means we can compute the variance in a single pass using the following algorithm:

```
variance(samples):
  M := 0
  S := 0
  for k from 1 to N:
    x := samples[k]
    oldM := M
    M := M + (x-M)/k
    S := S + (x-M)*(x-oldM)
    return S/(N-1)
```

To see that this method does work better than the one derived earlier you can make an experiment, or analyze it theoretically. John D. Cook found the accuracy of this method comparable to the accuracy of the two-pass method derived directly from the definition, while the results from the previous one-pass algorithm were found to be useless as expected.

« How the main thread is created in Java

Let’s draw the Mandelbrot set! »