**HOMEWORK2**

**Welford Recursion**

Welford's method is an efficient way to compute the mean and variance of a data set in a single pass. It's especially useful for large datasets or streaming data because it avoids the need to store the entire dataset in memory.

In this method, we update the mean and the sum of squared deviations incrementally as new data points are observed. This reduces memory usage and avoids numerical issues that can arise with other methods, like catastrophic cancellation when subtracting two nearly equal numbers.

The recursion works as follows:

* **Mean Update**: After each new data point is observed, the mean is updated using the formula:

meann=meann−1+xn−meann−1n\text{mean}\_{n} = \text{mean}\_{n-1} + \frac{x\_n - \text{mean}\_{n-1}}{n}meann​=meann−1​+nxn​−meann−1​​

where xnx\_nxn​ is the current data point and nnn is the number of data points observed so far.

* **Variance Update**: To calculate the variance incrementally, we maintain a running total of the sum of squared deviations from the mean, M2M2M2, updated using:

M2n=M2n−1+(xn−meann−1)×(xn−meann)M2\_{n} = M2\_{n-1} + (x\_n - \text{mean}\_{n-1}) \times (x\_n - \text{mean}\_n)M2n​=M2n−1​+(xn​−meann−1​)×(xn​−meann​)

The variance can then be computed as:

variance=M2n−1\text{variance} = \frac{M2}{n-1}variance=n−1M2​

This method is advantageous because it:

* Avoids recalculating the mean and variance from scratch every time a new data point is added.
* Has good numerical stability for large datasets.

**2. Research**

Let’s explore what happens in terms of **mean and variance** in different scenarios related to random walks:

1. **Random Walks with -1 and +1 jumps**:
   * The random walk involves steps of size +1 or -1, chosen randomly with a probability ppp. Over time, as more steps are taken, the mean remains close to zero (as positive and negative steps tend to balance out). However, the variance grows linearly with time. The variance is proportional to the number of steps taken.
2. **Absolute and Relative Frequencies**:
   * **Absolute frequency** tracks the number of successes (steps of +1) and failures (steps of -1). Over time, we might observe that the absolute number of successes or failures tends to increase with the number of steps, but there will be fluctuations.
   * **Relative frequency**, on the other hand, converges to the probability of success (ppp) as the number of steps increases. This is an important difference: the relative frequency stabilizes over time, while the absolute frequency continues to fluctuate.
3. **Distributions**:
   * **Final Distribution**: After many steps, the distribution of the walkers will resemble a normal (Gaussian) distribution. This is because the random walk is an example of a stochastic process that follows the Central Limit Theorem (given enough steps).
   * **Intermediate Distributions**: At intermediate steps, the distribution is not as wide, but over time the variance grows, and the distribution spreads out.

In your simulations:

* **Mean**: Should hover around 0 because steps of +1 and -1 are equally probable (if p=0.5p = 0.5p=0.5).
* **Variance**: Will grow with time because as more steps are taken, the sum of the squared deviations increases.

**3. Explanations**

**A. Jumps -1 and +1 with Probability ppp [Random Walk]**

In this part, we model a random walk where each step can either be +1 (move forward) or -1 (move backward), with a probability ppp of moving forward. This is a standard random walk:

* If p=0.5p = 0.5p=0.5, the walk is symmetric, and on average, the walker will stay near the origin.
* If p>0.5p > 0.5p>0.5, the walker will tend to move more in the positive direction.
* If p<0.5p < 0.5p<0.5, the walker will tend to move in the negative direction.

The random walk is generated by iterating through the steps, and at each step, a random number is generated to decide whether the walker takes a +1 or -1 step. The sequence of positions of the walker is then plotted.

**B. Absolute and Relative Frequency Trajectories**

In this part, we track two quantities over time:

1. **Absolute Frequency**: This counts how many times the walker has moved forward (+1) or backward (-1). The total number of steps is split between the two directions, and this can be visualized as a cumulative count.
2. **Relative Frequency**: This is the ratio of the number of forward moves to the total number of steps. Over time, the relative frequency should converge to the probability ppp (for forward steps). If the walk is symmetric (p=0.5p = 0.5p=0.5), the relative frequency will hover around 0.5.

**C. Final and Intermediate Distributions (Mean and Variance)**

In this section, you compute the distribution of the walker’s position at a particular time step:

* **Final Distribution**: After all steps have been taken, the distribution of the walker's final positions is plotted. This gives a sense of where the walker might end up after NNN steps.
* **Intermediate Distribution**: This allows you to take a snapshot of the walker’s position at any given time step and observe the distribution at that point in time.

For both distributions, the mean and variance are important:

* The **mean** gives the average position of the walker.
* The **variance** measures how spread out the walker's positions are from the mean. Over time, the variance increases because the walker moves farther away from the origin with more steps.

A graph of a random walk simulation

Description automatically generated

This graph represents a **Random Walk Simulation** where 20 hackers (represented by different colored lines) attempt to breach servers, with each step representing a +1 or -1 jump. In this particular case:

* **Number of Steps**: 100
* **Probability of +1**: 0.3, which means there's a 30% chance the hackers make a successful breach (moving up), and a 70% chance they fail (moving down).
* **Time Steps**: 100 (the x-axis represents the progression of time).
* **Hackers**: There are 20 hackers represented by different lines, some of which use solid lines, while others use dashed lines for better visualization.

**Key Insights from the Graph:**

* The **mean** value is negative at -20.11, indicating that, on average, the hackers are mostly unsuccessful in breaching the servers (because p=0.3p = 0.3p=0.3, which is less than 0.5).
* The **variance** is 160.31, which signifies that there is a significant spread among the hackers' performance, i.e., while some hackers are doing worse than others, a few hackers have had occasional success (as shown by a few upward trends in some lines).
* The **general trend** is downward, which is expected because the probability of success is less than 50%. Most hackers struggle to break even or stay in positive territory.

This simulation provides a good visual demonstration of how a low probability of success results in mostly negative outcomes, with some variance and deviation between the individual hackers' performance over time.