Problem Set 5

[Your name] and [student ID]
MAT1801-2020

Problem 1 [BHK 6.7] (10 points). Consider an algorithm that uses a random hash function and gives an estimate $\hat{x}$ of the true value x of some variable. Suppose that $\frac{x}{4} \leq \hat{x} \leq 4x$ with probability at least 0.6. The probability of the estimate is with respect to the choice of the hash function. How would you improve the probability that $\frac{x}{4} \leq \hat{x} \leq 4x$ to 0.8?

Hint: Since we do not know the variance, taking averages may not help. Consider other textbook measures of central tendency (don't overthink it!).

Problem 2 [BHK 6.16] (10 points). Suppose we want to pick a row of an $n \times n$ matrix at random where the probability of picking row i is proportional to the sum of squares of the entries of that row. How would we do this in the streaming model?

1. Do the problem when the matrix is given in row order. i.e. you receive a stream of the entries of the matrix one-by-one, but you get all the entries of row 1 first, and then all the entries of row 2, etc.
2. Do the problem when the matrix is given in column order. i.e. you receive a stream of the entries of the matrix one-by-one, but you get all the entries of column 1 first, and then all the entries of column 2, etc.
3. Do the problem when the matrix is represented in sparse notation: it is just presented as a list of triples (i, j, a_{ij}) in arbitrary order.

Suppose you are limited to storing at most $O(n)$ entries at a time (i.e. you can store several rows of the matrix if needed, but not the entire matrix). You may need multiple passes for some of the variants above, but try to give a solution in the minimum number of passes. Here, a pass means that you restart the stream a second, or third time. Hint: we didn't discuss the streaming sampling algorithm much in class, but this is covered in some depth in the book: sections 6.1 and 6.3.2.

Problem 3 (10 points). The MinHash sketch measures the Jaccard index (resemblance) of two sets by storing the minimum hash value of each set; the probability that the minimum hash values are the same is precisely the Jaccard index. Obviously, to get a reasonable error, you will want to repeat the process multiple times (roughly k times to get $O\sqrt{k}$ error.

MinHash has been applied to biological sequences to measure similarity by measuring the Jaccard index of the set of 'k-mers' (length-k substrings) of a sequence. For example, the string AACCGGTT has 4-mers AACC, ACCG, CCGG, CGGT, GGTT.

Write a Python function with call signature

```
def approximate_jaccard(A, B, k):
   '''A and B are Python strings
      k is an integer specifying the k-mer length
      ans is a float
   '''
   ...
   return ans
```

The Python function should take two strings A and B, and compute the Jaccard index of their k-mer sets to an error of 10% with 95% probability. I will be running your code on real bacterial sequences, so be sure your code is scalable. i.e. I'll be unhappy if it crashes my computer.