Math 55 - Spring 2004 - Lecture notes #21 - April 13 (Tuesday)

Goals for today: Application to politics
                          how close is an election before it is "too close to tell"
                    Variance and Standard deviation
                    Applications to computer science
                          searching a list

 EX: Suppose there is an election between two politicians, say G and B.
     Suppose that despite the best intentions (a big assumption!) there
     is still a small chance of miscounting a ballot for the other person.
     In fact, according to a front page NY Times article (11/17/2000) it
     is not unreasonable to assume an error rate of p=.001, that is the
     chance that a ballot will be counted for G instead of B, or for
     B instead of G, is .001. In fact an error rate of .01 is not
     unheard of.

     The question is: what is the probability that the wrong person wins,
     because of miscounting?

     The concern is that if the election is very close, that is if the
     actual number of votes cast for G and B is very close, then a
     few (randomly) miscounted ballots could swing the election the wrong
     way. If the election is not close, then we don't have to worry about
     this. So what we'd like to quantify is: Assuming the probability
     of miscounting a vote is some (tiny) p, how close does an election
     have to be before there is a large enough probability of the
     wrong outcome to want to do a recount? (The hope is that the
     recount is at least as accurate as the first count!)
     There are in fact laws in some jurisdictions that require recounts
     when the election is too close. We will try to give a mathematical
     basis for making such a decision, using data from the 2000 election.

     To model this, we use Bernoulli trials: each ballot is independently
     counted correctly with probability 1-p = .999, and incorrectly with
     probability p = .001. This does not model the real political process,
     (eg it does not account for ballots thrown out entirely)
     but is intended to say whether a 537 vote margin out of ~6 million
     votes cast is "statistically significant", i.e. whether there is
     a significant probability that B could win by 537 votes, including
     miscounts, even though more people voted for G.

1

```
To proceed, we assume there were T =
   vcB = 2,912,790 votes counted for B and
   vcG = 2,912,253 votes counted for G
for a total of vcB + vcG = 5,825,043 = T votes cast.
We will also let
   avB = votes actually cast for B and
   avG = votes actually cast for G
which of course we don't know
(though we know avB + avG = T, assuming no votes were lost!).
To see how likely it could be that the wrong person won,
we ask the following question: If avG = avB+1, that is
G should (barely) have won, then what is the probability that
the margin for B, vcB - vcG = 537, was as large as it was?
More precisely, we want

  P(vcB - vcG >= 537 when avG = avB+1)

(Note: you may think that what we really want is
  P(avG > avB        when vcB - vcG = 537)
 i.e. the probabilty that G won given the vote counts,
 but we don't know have enough information to compute this:
 The actual number of votes is not random, and there is
 no probability function for it. The most we can do is
 ask for different values of avG and avB, how likely are
 the vote counts. For example, suppose there is one vote,
 counted for B. The chance of this, given that the actual
 vote was cast for B, is 1-p. The chance given that the
 actual vote was cast for G, is p. This does not mean
 that the probability that G won is p, because that is
 not what is random.)

If avG = avB + 1 and avG + avB = T = 5,825,043, then
   avG = ceiling(5,825,043/2) = 2,912,522 votes, and
   avB =                       2,912,521 votes.
Each vote is counted correctly and independently with probability 1-p.
Now we compute the probability that with miscounts,
B wins by a margin of 2,912,790 - 2,912,253 = 537 or more.

We let the random variable
   mvB = margin of votes for B, i.e. the number of actual votes for B
         that were counted as votes for B, minus the number that
         were miscounted as votes for G.
```

Computing P(mvB=i) is identical to the problem of taking a coin marked
G and B, with probability P(B)=1-p and P(G)=p, flipping it avB times,
and computing the number of B's minus the number of G's. To get mvB=i
there need to be j of the avB votes counted for B and avB-j miscounted
for G, such that i = j-(avB-j) or j = (i+avB)/2. Thus

```
    P(mvB = i) = { C( avB, j ) * (1-p)^j * p^(avB-j) if j is an integer
                 { 0   if j is not an integer
```

Similarly, we let the random variable
   mvG = margin of votes for G, i.e. the number of actual votes for G
         that were counted as votes for G, minus the number that
         were miscounted as votes for B.
Again letting j = (i+avG)/2 we get

```
    P(mvG = i) = { C( avG, j ) * (1-p)^j * p^(avG-j) if j is an integer
                 { 0   if j is not an integer
```

Finally, we want the probability that B wins by a margin of 537 or
more votes: Let the random variable d = mvB - mvG. Then we want

```
    P(d >= 537)
        = P(mvB - mvG >= 537)
        = sum_{m = 537 to T} P(mvB - mvG = m)
              because mvB - mvG could equal 537, 538, ..., T
        = sum_{m = 537 to T} sum_{r = m to T} P(mvB=r and mvG=r-m)
            because we could have
                          mvB =  m  and mvG = 0   or
                          mvB = m+1 and mvG = 1   or
                          mvB = m+2 and mvG = 2   or
                                     ...
                          mvB =  T  and mvG = T - m
        = sum_{m = 537 to T} sum_{r = m to T} P(mvB=r) * P(mvG=r-m)
              by independence of mvG and mvB
```

Now you can plug in the formulas for P(mvB=r) and P(mvG=r-m)
and get something you can compute, which I invite you to try
(not required). If you write the simplest program,
it will cost O(T^3), too expensive! Also, many of the C(avB,j) will
overflow (be larger than 10^308, and so not be representable in double
precision floating point), and many p^(avB-j) will underflow (be less
than 10^(-308)) so care is needed

Later we will describe an easy approximation you can use for this
problem.

ASK&WAIT: When p = probability of error = .001, about what do you think
          the probability of a margin of 537 votes or more is?
          (just to the nearest power of 10)?

ASK&WAIT: When p = .01, what do you think the probability of margin of 537
          or more votes is? for p=.1? for p=.5?

 Mean and Standard deviation: many of you ask what the mean and standard
 deviation of the scores on the midterm are, so you already understand that
   1) the mean tells you the average score
   2) the standard deviation (std) tells you how many people were
      "close to the average"
      i.e expect many scores to lie in range from mean-x*std to mean+x*std
          where x = 1 or 2; if your score is < mean-3*std you know you're
          in trouble, and if your score is > mean+3*std you feel really good
 Now it is time to define this carefully:

DEF: Let S be a sample space, P a probability function, and f a random variable.
     E(f) the expectation of f.
     Then let g = (f - E(f))^2. Then the variance of f is defined as
     V(f) = E(g)
          = sum_{e in S} g(e)*P(e)
          = sum_{e in S} (f(e) - E(f))^2*P(e)
     The standard deviation of f is defined as
     sigma(f) = (var(f))^(1/2)

EX: Roll a fair die 100 times, getting value fi in set {1,2,3,4,5,6} with
    probability 1/6 at roll i.
    Then E(fi) = 1/6 + 2/6 + ... + 6/6 = 3.5 and
        V(fi) = (1-3.5)^2/6 + (2-3.5)^2/6 + ... + (6-3.5)^2/6 ~ 2.9
     sigma(fi) ~ sqrt(2.9) ~ 1.7
    Let g = f1 + f2 + ... + f100 and h = 100*f1.
    and E(g) = E(f1) + ... + E(f100) = 100*E(f1) = E(h) = 350.
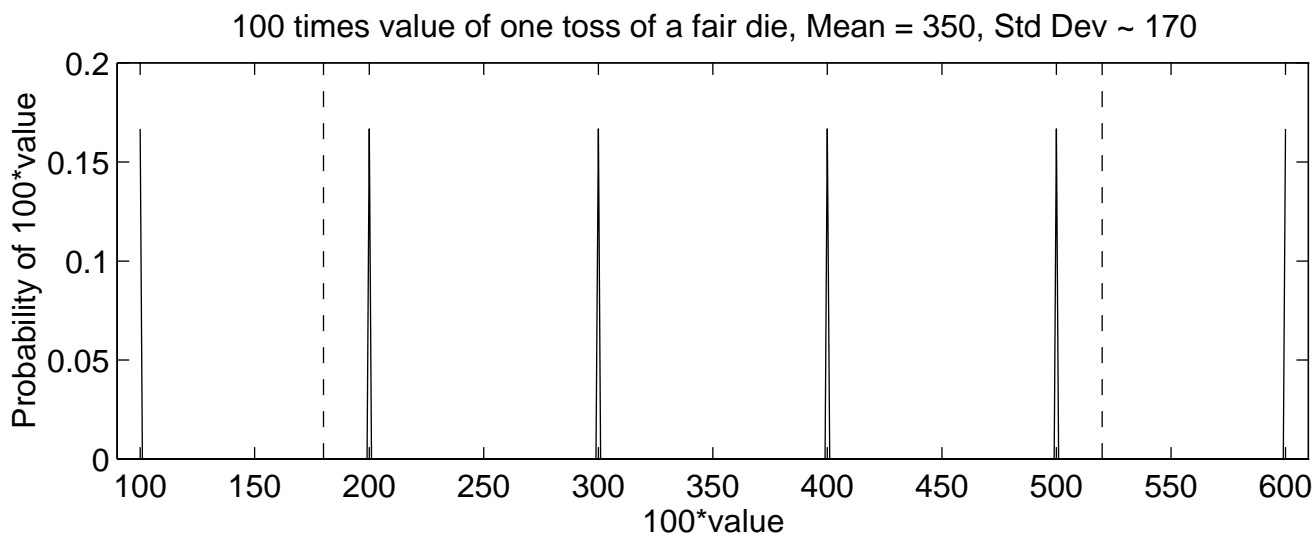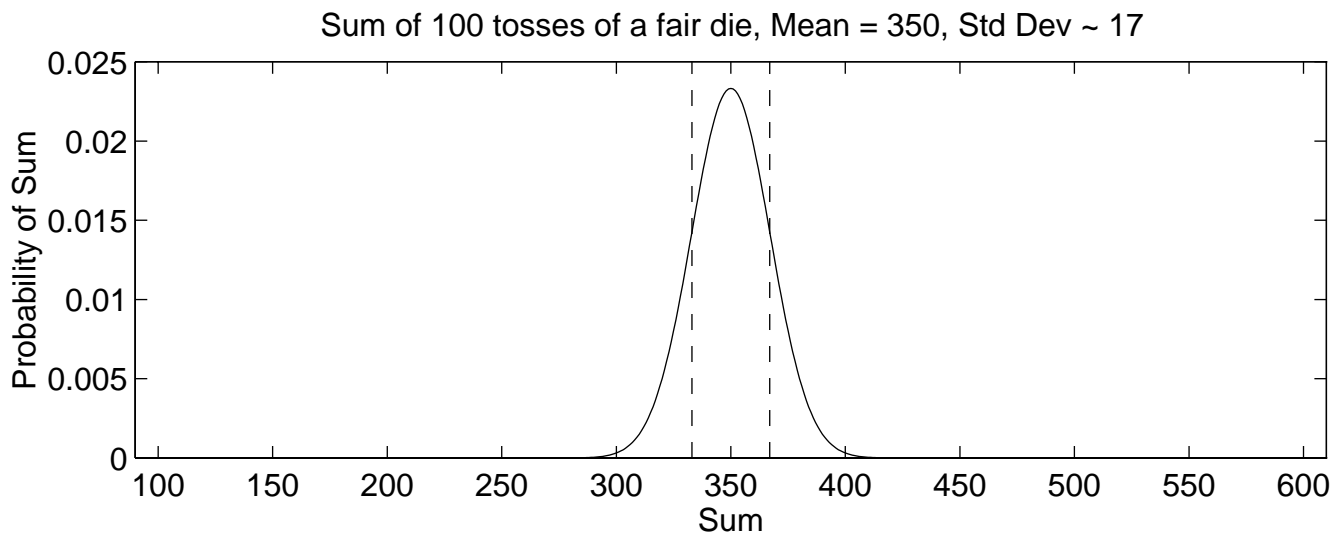   but V(g) = 100*V(f1) ~ 290   whereas V(h) = 100^2*V(f1) ~ 29000
     so sigma(g) ~ 17   and sigma(h) = 170
    (how to do this calculation later).
    The following figure illustrates this, where the vertical dashed lines
    are drawn at mean-sigma and mean+sigma. The point is that "a lot" of

the likely values are in this range, even more in the range
mean-2*sigma and mean+2*sigma, etc (we make this more precise later,
via Chebyshev's inequality)

Sum of 100 tosses of a fair die, Mean = 350, Std Dev ~ 17

100 times value of one toss of a fair die, Mean = 350, Std Dev ~ 170

Here are some other formulas for computing the variance:

```
Thm 1: V(f) = E(f^2) - (E(f))^2
     Proof: V(f) = E((f-E(f))^2)      by definition
                 = sum_x (f(x)-E(f))^2*P(x)    by definition
                 = sum_x f(x)^2*P(x) - sum_x 2*E(f)*f(x)*P(x) + sum_x E(f)^2*P(x)
                 = sum_x f(x)^2*P(x) - 2*E(f)*sum_x f(x)*P(x) + E(f)^2*sum_x P(x)
                 = E(f^2)            - 2*E(f)*E(f)             + E(f)^2*1
                 = E(f^2) - (E(f))^2
```

```
Thm 2: Suppose f and g are independent. Then V(f+g) = V(f) + V(g)
    Proof: V(f+g) = E((f+g)^2)) - E(f+g)^2        by Thm 1
              = E(f^2 + 2*f*g + g^2) - (E(f)+E(g))^2
              = E(f^2) + 2*E(f*g) + E(g^2) - (E(f)^2 - 2*E(f)*E(g) - (E(g)^2)
              = E(f^2) + 2*E(f)*E(g) + E(g^2) - (E(f))^2 - 2*E(f)*E(g) - (E(g))^2
                    by independence of f and g
              = E(f^2) - (E(f))^2 + E(g^2) - (E(g))^2
              = V(f)                + V(g)   by Thm 1

EX:  g = f1 + f2 + ... + f100 where fi is the result of rolling a fair die once.
    By Thm 2, V(g) = V(f1) + ... + V(f100) = 100*V(f1) ~ 100*2.9 = 290
    and so sigma(g) ~ sqrt(290) ~ 17
    But V(100*f1) = E((100*f1)^2) - E((100*f1))^2
                  = 100^2*(E(f1^2) - E(f1)^2)
                  = 100^2*V(f1)
                  = 10000*V(f1)
    and sigma(100*f1) ~ 100*sigma(f1) ~ 170 as claimed above


Now we quantify the notion that the value of a random variable f(x) is
unlikely to fall very far from E(f):

Thm (Chebyshev's Inequality): Let f be a random variable. Then
    P( |f(x) - E(f)| >= r*sigma(f) )   <= 1/r^2
In words, the probability that the value of a random variable f(x) is
farther than r times the standard deviation sigma(f) from its mean value
E(f) can be no larger than 1/r^2, which decreases rapidly as r increases.
Here is a table:
            r      P( |f(e) - E(f)| >= r*sigma(f))
           ---    ------------------------------
            1       <= 1  (trivial bound on a probability!)
            2       <= 1/4  = .25
            5       <= 1/25 = .04
           10       <= 1/100= .01
Proof: Let A be the event {e: |f(e) - E(f)| >= r*sigma(f)}. We want to
compute P(A). Now compute
            V(f) = sum_{e in S} (f(e)-E(f))^2 * P(e)
                 = sum_{e in A} (f(e)-E(f))^2 * P(e)
                       + sum_{e not in A} (f(e)-E(f))^2 * P(e)
The second sum is at least 0, and each term in the first sum is at
least (r*sigma(f))^2*P(e), by the definition of A. Thus
            V(f) >= sum_{e in A} (r*sigma(f))^2 * P(A)
```
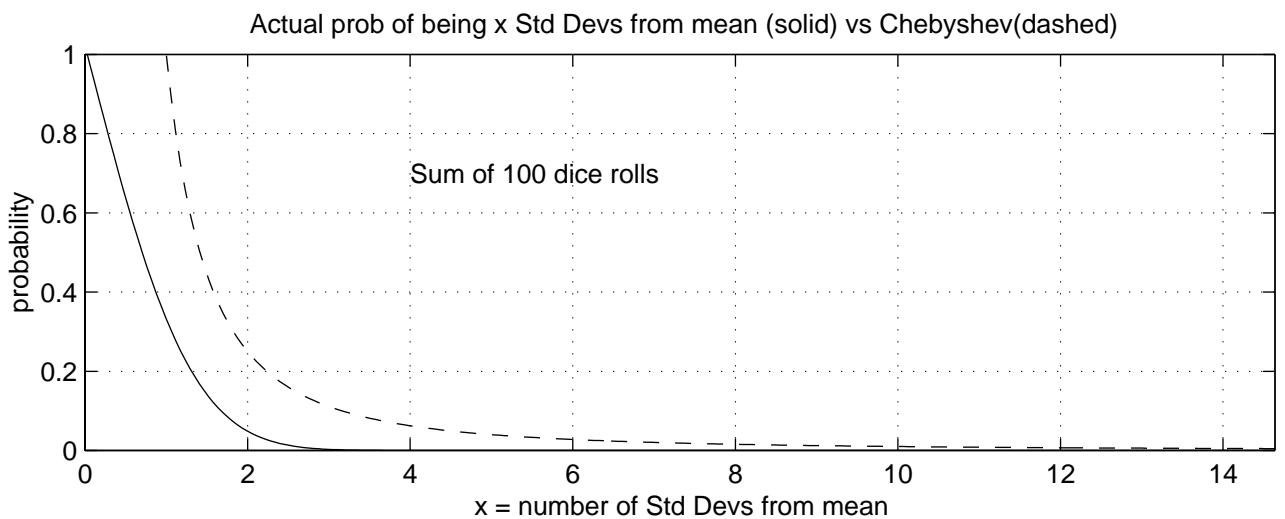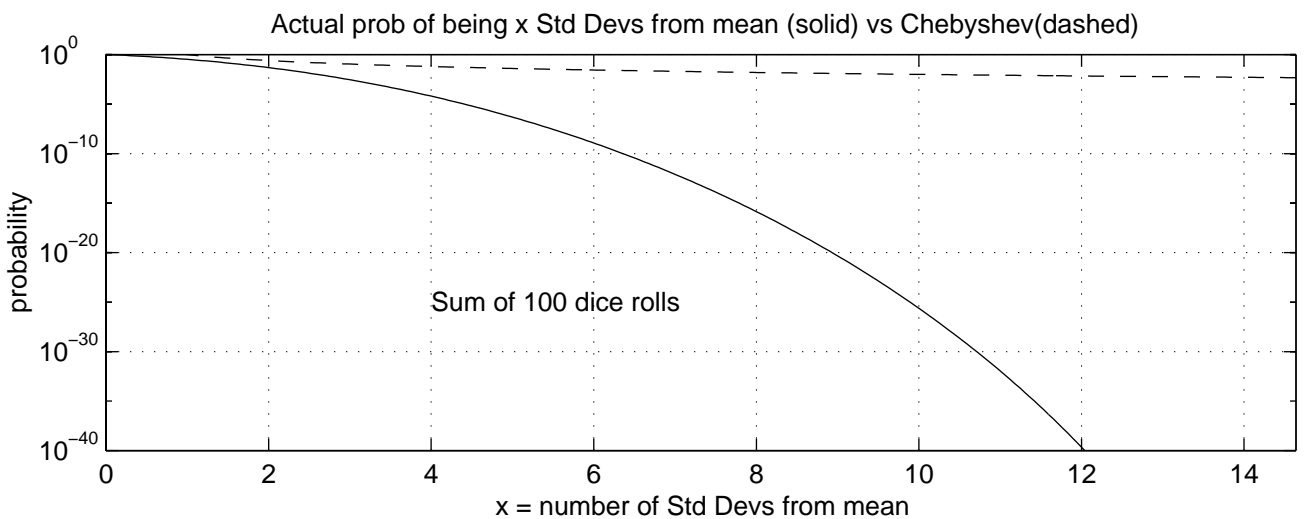
```
                        = (r*sigma(f))^2 * P(A)
  or            1/r^2 >= P(A) as desired.

   How good is Chebyshev's inequality, i.e. how close to
      P( |f(x) - E(f)| >= r*sigma(f) )  can 1/r^2 be?
   EX: Again consider rolling a die 100 times, and computing sum h.
        Then E(h) = 350 and sigma(h)~17.
        Comparing  actual P( |h-350| > r*17)  with 1/r^2 from Chebyshev
        yields plot below: Chebyshev much too large for large r:
```

Actual prob of being x Std Devs from mean (solid) vs Chebyshev(dashed)

x = number of Std Devs from mean

Actual prob of being x Std Devs from mean (solid) vs Chebyshev(dashed)

x = number of Std Devs from mean

```
   EX: Suppose we have a list of n distinct items L(1),...,L(n), and
        want an algorithm that takes an input x and
            (1) returns i if L(i)=x,
            (2) returns n+1 otherwise
```

An obvious algorithm is "linear search"
```
    i=0
    repeat
        i=i+1
    until L(i)=x or i=n+1
```

Suppose x is chosen at random from a sample space S
with probability function P. What is the expectation of the
operation count C of this algorithm, i.e. how many times
is the line "i=i+1" executed? If we run this algorithm
many times, E(C) tells us how long it will take "on average",
and sigma(C) tells us how variable the running time will be.

The answer depends on S and P:

EX 1: Suppose S = {L(1),...,L(n)} and P(L(i))=1/n; this means
        each input x must be in the list L, and is equally likely.
        Then E(C) = 1*(1/n) + 2*(1/n) + 3*(1/n) + ... + n*(1/n)
                  = (n+1)/2
        and V(C) = E(C^2) - (E(C))^2
                  = 1^2*(1/n) + 2^2*(1/n) + 3^2*(1/n) + ... + n^2*(1/n)
                    - ((n+1)/2)^2
                  = (n^3/3 + O(n^2))*(1/n) - (n^2/4 + O*(n^2))
                  = n^2/12 + O(n)
        so sigma(C) = n/sqrt(12) + O(1) ~ .29*n + O(1)

EX 2: Suppose S = {L(1),...,L(n),z}, and P(L(i))=p/n, P(z)=1-p
        i.e. the probability that x is not on the list is 1-p
ASK&WAIT: What is E(C)? How does it depend on p?
ASK&WAIT: What are V(C) and sigma(C)? How do they depend on p?

EX 3: Suppose P(L(i)) = p(i) are all different.
        In what order should we search the L(i) to minimize E(C)?
ASK&WAIT: Suppose n=2, and p(2) >> p(1); which item should we search first?

        Thm: Searching list in decreasing order of p(i) minimizes E(C).

        Proof: Suppose that we do not search list in decreasing order
        of p(i); then we will show that there is a different search
        order with smaller E(C). Let q(1),...,q(n) be the probabilities
        of the items in the order they are searched, so q(1),...,q(n)
        is a permutation of p(1),...,p(n). Suppose that q(1),...,q(n)

8

is not in decreasing order; this means that for some j
    q(j) < q(j+1).
Let r = 1 - p(1) - p(2) - ... - p(n) be the
probability that the search item is not in the list.

Then C1 = E(C) for search order 1,...,j,j+1,...,n
        = sum_{i=1 to n} i*q(i)   + (n+1)*r
and C2 = E(C) for search order 1,...,j+1,j,...,n
            (i.e. with j+1 searched before j)
        = sum_{i=1 to n except j and j+1} i*q(i)
            + j*q(j+1) + (j+1)*q(j) + (n+1)*r
Then C1 - C2 = j*q(j) + (j+1)*q(j+1) - j*q(j+1) - (j+1)*q(j)
            = q(j+1) - q(j)
            > 0
In other words, searching j+1 before j lowers the expected cost E(C)
So unless q(j)>q(j+1) for all j, i.e. unless the list is sorted,
there is a better order to search in.

EX 4:   Suppose P(L(i)) = c*q^i, q<1, where c is chosen so that
        sum_{i=1 to n} P(L(i)) = 1, i.e. x guaranteed to be in list
ASK&WAIT: What is c?
ASK&WAIT: What is E(C)?
ASK&WAIT: What happens to E(C) as n grows?

        Recall Binary Search:
            Assume L(1) < L(2) < ... < L(n), by sorting if necessary
            istart = 1; iend = n
            repeat
                middle = floor((istart + iend)/2)
                if x < L(middle) iend = middle-1
                if x > L(middle) istart = middle+1
            until x = L(middle) or iend < istart
        Cost = O(log_2 n) steps, because each time the list to search
        is at most half as long.

ASK&WAIT?   What is faster for large n, linear search or binary search?

        But in the worst case, linear search will take n steps,
        much slower than binary search.
ASK&WAIT: Can you think of an algorithm that takes
        O(1) steps on average, and O(log n) at worst, i.e. has the
        advantages of both kinds of search?

9