HOMEWORK 3 SOLUTIONS

JC

(1) $|\vec{a}+\vec{b}|^2 = (\vec{a}+\vec{b}).(\vec{a}+\vec{b})$ by the definition of length. Using distributivity of inner product over vector addition, commutativity of inner product and commutativity of vector addition we get $(\vec{a}+\vec{b}).(\vec{a}+\vec{b}) = |a|^2 + 2\vec{a}.\vec{b} + |b|^2$. Since as we saw in class $\vec{a}.\vec{b} \leq |a| \times |b|$, we conclude $|\vec{a}+\vec{b}|^2 \leq (|\vec{a}|+|\vec{b}|)^2$ and thus $|\vec{a}+\vec{b}| \leq (|\vec{a}|+|\vec{b}|)$.

(2) (a) There are $\binom{M}{N}$ sequences with M heads, and each has probability $p^M q^{N-M}$, so the probability is $\binom{M}{N} \times p^M q^{N-M}$.

(b) For $N = 2$ the expected value is by definition $p^2 \times 2 + 2pq \times 1 + q^2 0 = 2p(p+q) = 2p$, as $p+q = 1$.
For $N = 3$ we get $p^3 \times 3 + 3p^2q \times 2 + 3pq^2 \times 1 + q^3 \times 0$. This is $3p \times (p^2 + 2pq + q^2)$, and since $1^2 = (p+q)^2 = p^2 + 2pq + q^2$ this simplifies to $3p$.
In general the expectation value will be Np. This is easy to prove using the additivity of expectation, or by a direct generalisation of the arguments above. Of course it is also intuitively obvious since each trial gives "p heads".

(c) Here is the Maple output

```
> with(plots);
```

```
Warning, the name changecoords has been redefined
```

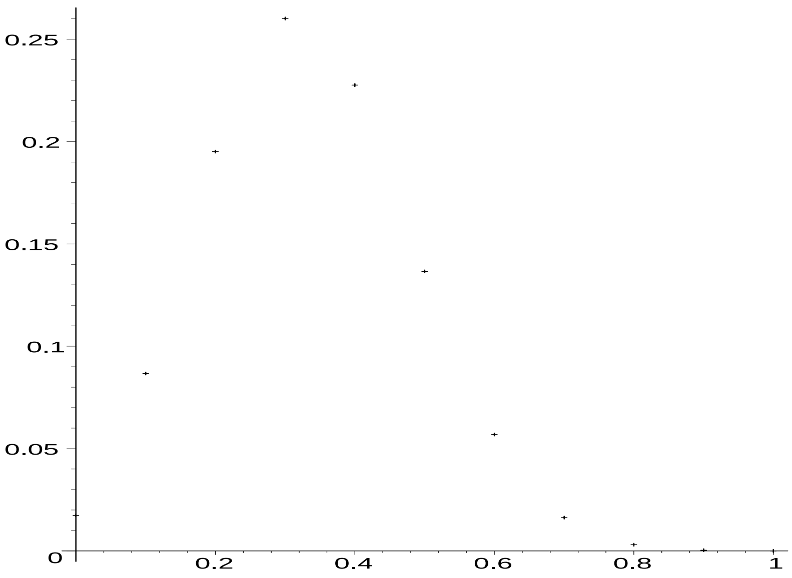
[*animate*, *animate3d*, *animatecurve*, *arrow*, *changecoords*, *complexplot*, *complexplot3d*, *conformal*, *conformal3d*, *contourplot*, *contourplot3d*, *coordplot*, *coordplot3d*, *cylinderplot*, *densityplot*, *display*, *display3d*, *fieldplot*, *fieldplot3d*, *gradplot*, *gradplot3d*, *graphplot3d*, *implicitplot*, *implicitplot3d*, *inequal*, *interactive*, *listcontplot*, *listcontplot3d*, *listdensityplot*, *listplot*, *listplot3d*, *loglogplot*, *logplot*, *matrixplot*, *odeplot*, *pareto*, *plotcompare*, *pointplot*, *pointplot3d*, *polarplot*, *polygonplot*, *polygonplot3d*, *polyhedra_supported*, *polyhedraplot*, *replot*, *rootlocus*, *semilogplot*, *setoptions*, *setoptions3d*, *spacecurve*, *sparsematrixplot*, *sphereplot*, *surfdata*, *textplot*, *textplot3d*, *tubeplot*]

```
> ?pointplot
```

```
> N:=10;
```

$$N := 10$$

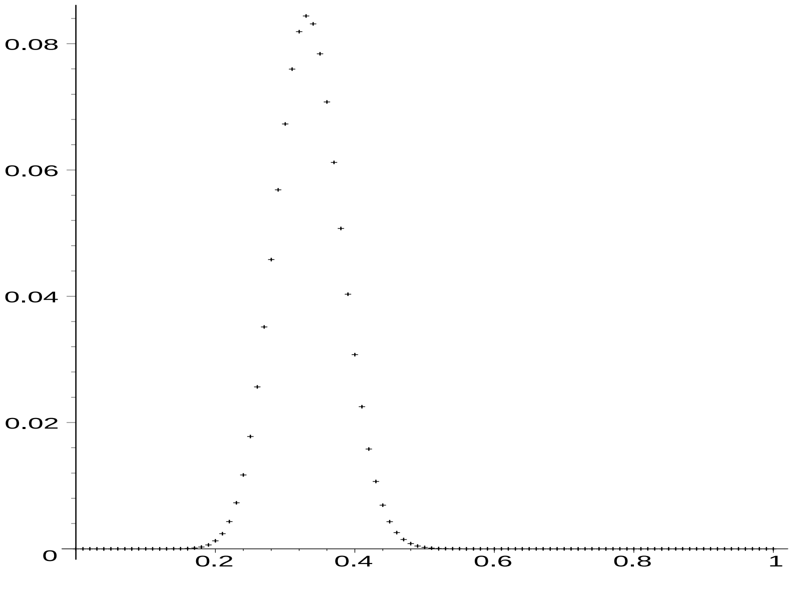
```
> pointplot( { seq( [M/N, N !/(M ! * (N -M)!) * (1/3)^M *
> (2/3)^(N-M)], M=0..N) } );
```




```
> N:=100;
```

$$N := 100$$

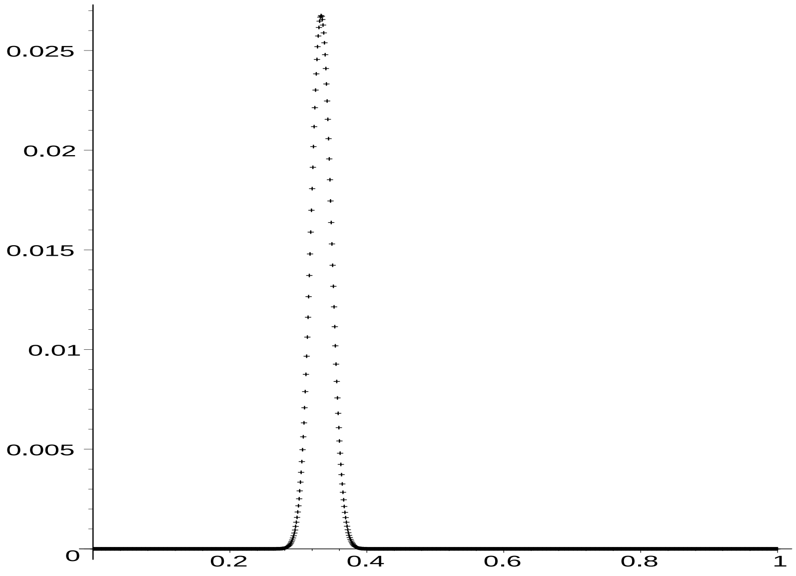
```
> pointplot( { seq( [M/N, N !/(M ! * (N -M)!) * (1/3)^M *
> (2/3)^(N-M)], M=0..N) } );
```




```
> N:=1000;
```

$$N := 1000$$

```
> pointplot( { seq( [M/N, N !/(M ! * (N -M)!) * (1/3)^M *
> (2/3)^(N-M)], M=0..N) } );
```



We see a bell-shaped distribution of probabilities, clustered more and more closely around the value 1/3 as N increases.
Cultural note: this is an example of the important Central Limit Theorem in statistics. With appropriate scaling of the x and y axes we would see convergence to a "normal distribution" as N increases.
As you probably guessed, the point of this question is to illustrate why we observe values of the IOC that are close to the expected values.

(3) (a) The gcd of a and b is 1, so there exist P and Q with $Pa + Qb = 1$. Multiply by X to get $(PX)a + (QX)b = X$.

(b) $Pa + Qb = Ma + Nb$, so $(P - M)a = (N - Q)b$. Since a and b have no common factors, a divides $N - Q$ and b divides $P - M$. Suppose $P - M = Yb$, then $Yba = (N - Q)b$ so $N - Q = Ya$.

(4) There are of course infinitely many ways of doing this. For example

```
R_1 = 0
R_4 = 0
R_1 = R_1 + R_2
R_3 = R_3 - 1
IF(R_4 < R_3) THEN GOTO 3
STOP
```

(5) It is completely insecure.

Note first that if someone suspects that we are reusing a one-time pad, they can confirm their suspicion by using the IOC method (exactly like determining the length of the key in the Vigenere cipher).

So suppose that someone intercepts our message and determines that we have used a key of length one million twice. Let c_i be character i in the plaintext and note that by subtracting character $i + 10^6$ of the ciphertext from character i in the ciphertext, the interceptor can determine $c_i - c_{i+10^6}$ (all of this modulo 26 of course)

To put it another way, if the interceptor has a guess at c_i this tells them what c_{i+10^6} will be. This turns out to be surprisingly bad, because of what we know about the statistics of English.

Notice that among the 18000 or so possible trigrams that could (in principle) occur in English there is a very wide variation in frequency. For example "the" is very common (about one and a half percent) but most trigrams (for example "waq") have a frequency very close to zero. Suppose that the interceptor has available a table giving the frequency of every trigram (such a table is not difficult to compile).

Now for each $i < 10^6 - 2$ the interceptor can do the following; let A run through the set of all trigrams, guess that $c_i c_{i+1} c_{i+2} = A$, and let B be the resulting guess at $c_{i+10^6} c_{i+10^6+1} c_{i+10^6+2}$. Compute the frequency of A times the frequency of B for each A. Since most trigrams have negligible frequency this product will generally be very small, but there will be a small number of "plausible pairs". The interceptor records the set of plausible pairs. (This involves in total about 1.8×10^{10} guesses at A, computations of B, lookups of the frequencies and computations of the products. This is a lot but not unfeasible)

Now by comparing adjacent plausible pairs and picking out the ones that match, the interceptor can recover the plaintext with high probability.