HW #2 - solutions

Problem 1 a) The likelihood function is:

$$P(r|\lambda) = e^{-\lambda}\frac{\lambda^r}{r!}$$

For convenience, let's work with the logarithm:

$$\log P(r|\lambda) = -\lambda + r\log\lambda - \log(r!)$$

If r is given, maximizing this with respect to λ (to find the Max Likelihood estimate for λ):

$$\left.\frac{d}{d\lambda}\right|_{\lambda=\lambda_*} \log P(r|\lambda) = 0 \quad \Rightarrow \quad -1 + \frac{r}{\lambda_*} = 0 \ , \ \boxed{\lambda_* = r}$$

b) For a Gaussian: $P(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$

$$\log P = -\frac{x^2}{2\sigma^2} - \frac{1}{2}\log(2\pi\sigma^2)$$

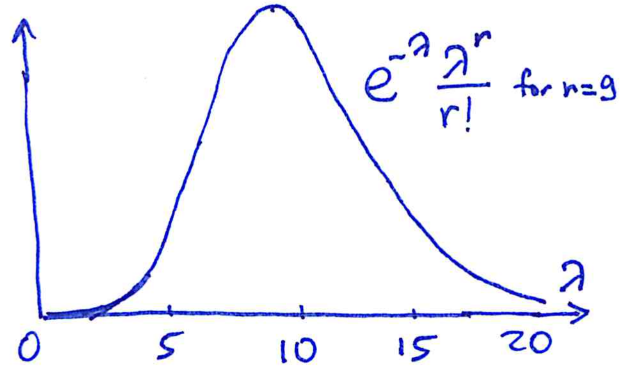
$$\left.\frac{d^2P}{dx^2}\right|_{x=0} = -\frac{1}{\sigma^2}$$

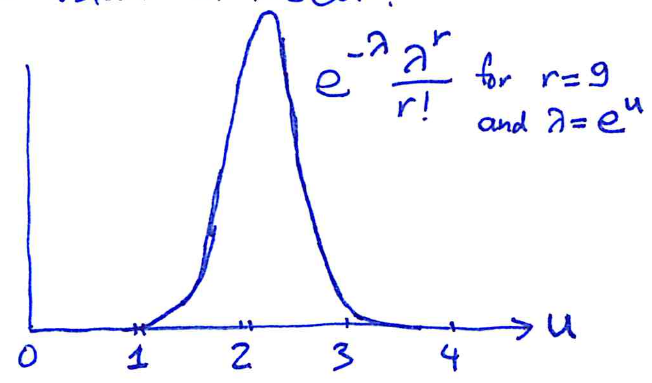
So <u>for a Gaussian</u>, its width σ is related to its curvature (second derivative) at its peak by a very simple relation:

$$\sigma = \sqrt{\frac{1}{-\left.\frac{d^2P}{dx^2}\right|_{\text{at peak}}}}$$

In our problem, the likelihood is not a Gaussian – but it's a probability distribution with a single hump, for which a Gaussian approximation ~~seems an~~ seems an acceptable way to estimate its width.

If you plot the distribution $P(r|\lambda)$, you will notice that it looks more Gaussian-like if plotted as a function of $u = \ln\lambda$ rather than λ itself:





Because of this, appoximating this distribution by a Gaussian ~~is b~~ will be more accurate if we do this in the u coordinate. Let's do both and compare.

Using λ directly: $\log P(r|\lambda) = -\lambda + r\log\lambda - \log(r!)$

$$\left.\frac{d^2}{d\lambda^2}\right|_{\lambda=\lambda_*} = \left.\frac{d}{d\lambda}\right|_{\lambda_*}\left(-1+\frac{r}{\lambda}\right) = -\frac{r}{\lambda_*^2} = -\frac{1}{r}$$

So fitting this with a Gaussian requires a Gaussian of width $\sigma_\lambda = \sqrt{\frac{1}{1/r}} = \sqrt{r}$. So our estimate for λ is $r \pm \sqrt{r}$
$= 9 \pm 3$.

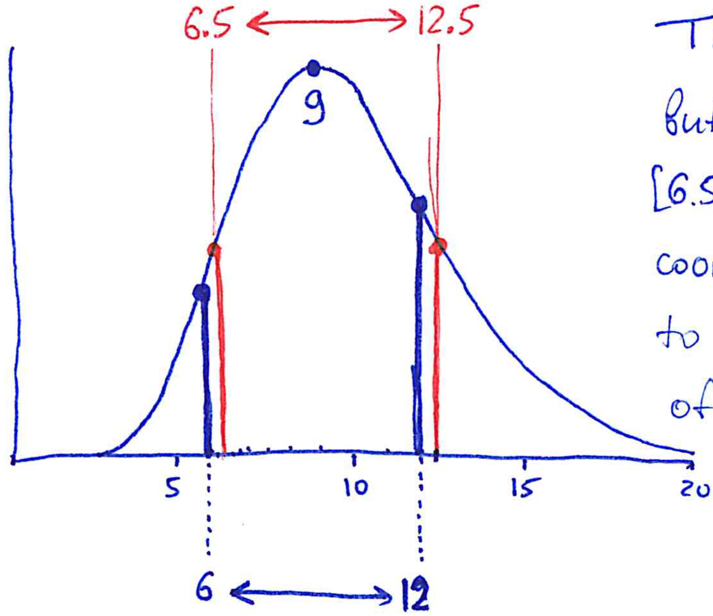
Using the u coordinate: $\log P(r|\lambda) = -\lambda + r\log\lambda - \log(r!)$
$u = \log\lambda$
$= -e^u + ru - \log(r!)$

$\left.\frac{d^2}{du^2}\right|_{u=u_*} \log P(r|\lambda) = -e^{u_*} = -\lambda_*$, and fitting this with a Gaussian

requires one of width $\sigma_u = \sqrt{\frac{1}{\lambda_*}} = \sqrt{\frac{1}{r}}$. So our estimate

for $u = \log\lambda$ is $u_* \pm \sigma_u = \log 9 \pm 0.33 = 2.20 \pm 0.33$.

The corresponding range for λ is $\left.\begin{array}{l} e^{2.20+0.33} = 12.5 \\ e^{2.20-0.33} \approx 6.5 \end{array}\right\}$ note that this captures the asymmetry of the Poisson distribution.

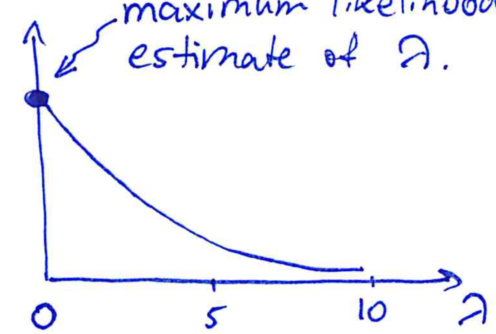


The two answers are similar, but the confidence interval [6.5 .. 12.5] obtained in the $u = \ln \lambda$ coordinates does in fact seem to capture better the slight skew of the Poisson distribution.

(c) Taking into account the extra photons from the background.

$$P(r|\lambda) = e^{-(\lambda+b)} \frac{(\lambda+b)^r}{r!}$$

valid only for $\lambda > 0$ $\Rightarrow$ The likelihood:



With this much background, all we can say is that the star is not very bright $\Rightarrow$ we can bound its expected intensity from above. The MLE estimate for λ is 0.

Problem 2 Gaussian in dimension $k \gg 1$.

(a) If ~~points~~ (coordinates) $x_1 .. x_k$ are distributed according to

$$P(x_1..x_k) = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^k e^{-\frac{x_1^2 + \dots + x_k^2}{2\sigma^2}},$$

how is the distance to origin distributed? $r = \sqrt{x_1^2 + x_2^2 + \dots + x_k^2}$

Polar coordinates in dimension $k=2$:

$$\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^2 e^{-\frac{x_1^2+x_2^2}{2\sigma^2}} dx_1\, dx_2 = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^2 e^{-\frac{r^2}{2\sigma^2}} r\, dr\, d\varphi$$

For any function $f(r)$, what is its average value?

$$\langle f(r)\rangle = \int_0^\infty\!\!\int_0^{2\pi} f(r)\, \frac{e^{-r^2/2\sigma^2}}{\left(\sqrt{2\pi\sigma^2}\right)^2}\, r\, dr\, d\varphi = \int_0^\infty \cancel{dr} \left[f(r) \cdot \underbrace{\frac{e^{-r^2/2\sigma^2}}{\left(\sqrt{2\pi\sigma^2}\right)^2}\, 2\pi r}_{\text{Probability density distribution of } r.} \right] dr$$

Comparing this with $\langle f(r)\rangle = \int_0^\infty f(r) \cdot \underbrace{P_r(r)}\, dr$

we read of $P_r(r)$ in dimension 2 : $P_r(r)\cancel{dr} = \underbrace{\frac{e^{-r^2/2\sigma^2}}{\left(\sqrt{2\pi\sigma^2}\right)^2}}_{\text{density of points}} \cdot \underbrace{2\pi r}_{\text{total area distance } r \text{ from } 0.} dr$

Coefficient 2π = length of unit circle.

Similarly, in dimension $k=3$:

$$\langle f(r)\rangle = \int_0^\infty f(r) \cdot \frac{e^{-\frac{r^2}{2\sigma^2}}}{\left(\sqrt{2\pi\sigma^2}\right)^3}\, 4\pi r^2 dr$$

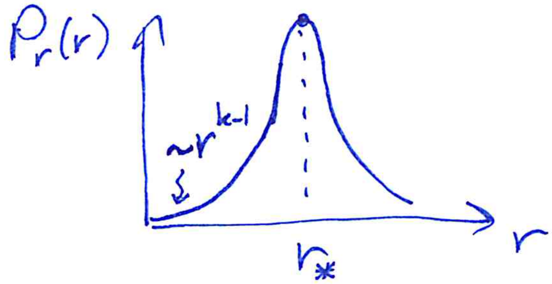
$\Rightarrow$ In dimension 3, $P_r(r)dr = \dfrac{e^{-r^2/2\sigma^2}}{\left(\sqrt{2\pi\sigma^2}\right)^3} \cdot \overbrace{4\pi}^{\text{area of unit circle}} r^2 dr$

In dimension k:

$$P_r(r)\, dr = \frac{e^{-r^2/2\sigma^2}}{\left(\sqrt{2\pi\sigma^2}\right)^k} \cdot \underbrace{C_k}_{\substack{\text{coefficient} \\ = \text{volume of} \\ \text{a unit } (k-1)\text{-sphere}}} \cdot \underbrace{r^{k-1} dr}_{\substack{\text{Must have units of} \\ \text{volume} \Rightarrow \text{fixes power of } r.}}$$

The exact value of C_k is irrelevant, as it just fixes the overall normalization (ensuring $P_r(r)$ integrates to 1).

Now that we know $P_r(r)$, we can see where it peaks and how narrow is it.



$$\frac{d}{dr}\bigg|_{r=r_*} \left(r^{k-1} e^{-\frac{r^2}{2\sigma^2}}\right) = 0$$

$$\Rightarrow (k-1)\, r_*^{k-2} e^{-\frac{r_*^2}{2\sigma^2}} = \frac{r_*^k}{\sigma^2} e^{-\frac{r_*^2}{2\sigma^2}}$$

$$\Rightarrow r_*^2 = (k-1)\sigma^2$$

$$r_* = \sigma\sqrt{k-1} \underset{\text{at large } k}{\approx} \sigma\sqrt{k}$$

And to find the thickness (width of $P_r(r)$), we can proceed exactly as in problem 1b.

$$\log P_r(r) = (k-1)\log r - \frac{r^2}{2\sigma^2}$$

$$\frac{d^2}{dr^2}\bigg|_{r=r_*} \log P_r(r) = -\frac{k-1}{r_*^2} - \frac{1}{\sigma^2} = -\frac{2}{\sigma^2} \Rightarrow \text{Width of } P_r(r)$$

can be estimated as $\sqrt{\frac{1}{2/\sigma^2}} = \boxed{\frac{\sigma}{\sqrt{2}}} = \frac{r_*}{\sqrt{2}\cdot\sqrt{k-1}} \underset{\text{at large } k}{\propto} \frac{r_*}{\sqrt{k}}$

(b) Probability density $P(x_1 \dots x_k)$ at origin $x_1 = x_2 = \dots = x_k = 0$

is $\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^k e^{-\frac{0}{2\sigma^2}} = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^k$

Whereas at $r = r_* \approx \sigma\sqrt{k}$ it is $\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^k \underbrace{e^{-\frac{r_*^2}{2\sigma^2}}}_{e^{-k/2}}$, i.e.

exponentially smaller.

(c) For $k=1000$, we find $r_* = \sigma\sqrt{1000} \approx 32\sigma$ (shell radius)

If $P_r(r)$ is approximately a Gaussian, then ± 2 standard deviations of that Gaussian contains 95% mass $\Rightarrow$ The shell containing 95% mass is $32\sigma \pm 2\cdot\boxed{\frac{\sigma}{\sqrt{2}}}$

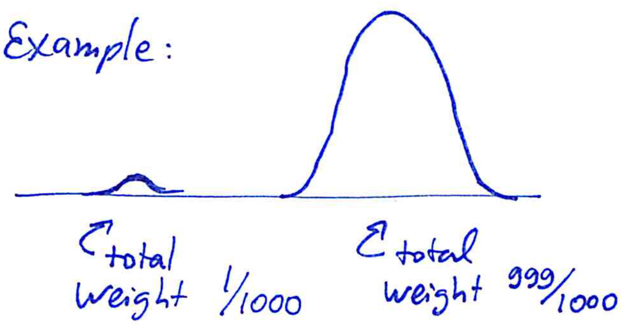
$\sim (32 \pm 1.4)\sigma$

But the probability density 32σ away from mean is astronomically small: $e^{-\frac{r_*^2}{2\sigma^2}} = e^{-k/2} = e^{-500} \cong 10^{-218}$

(d) Gaussian density at zero is $\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^k \propto \left(\frac{1}{\sigma}\right)^k$

Changing σ by 1% changes this by a factor of $(1.01)^k = e^{k \log(1.01)} \simeq e^{0.01k}$, so for $k=1000$ this is $e^{10} \approx 22000$ (the narrower Gaussian has higher density at zero)

(e) Example:

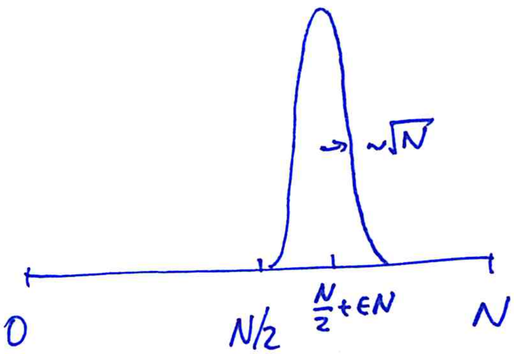


Let's say this ↖ Gaussian is 1% narrower. Then (d) tells us its maximal density is ~ 20000 times larger. So despite it carrying only $\frac{1}{1000}$ of total weight, the maximum likelihood estimator would still pick this Gaussian (by a factor of $\frac{20000}{1000} \sim 20$) over the more representative one (where most of the probability mass is contained).

Problem 3 (a,b,c) see printout.

(d) If $p_a \approx 0.5$: let's say $p_a = 0.5 + \epsilon$.

The expected number of "a" outcomes in N tosses: $\frac{N}{2} + \epsilon N$



But this distribution has a finite width $\sim \sqrt{N}$. So if N is too small, one cannot reliably distinguish this from the fair coin scenario.

To be able to make this call, we must have $\sqrt{N} \lesssim \epsilon N$

$$\Rightarrow N \sim \frac{1}{\epsilon^2}$$

To check this in simulations, we might run simulations for different ϵ and record the value of N when R crosses some threshold (e.g. $R=10$).

One would then plot $N \cdot (p - 1/2)^2$ versus p. If our prediction is correct, the curve should exhibit no obvious trend — neither up nor down. The code & the plot — see next page.

(e) What is the best case scenario for evidence for a biased coin?

If we only get "a"'s! $F_a = F$, $F_b = 0$: $R = \frac{2^F F_a! F_b!}{(F+1)!} = \frac{2^F}{F+1}$

$\Rightarrow \log R$ is linear in F.

But the best case for a fair coin: $F_a = F_b = F/2 \Rightarrow \log R = \text{const} - \frac{\log F}{2} + \dots$

$\Rightarrow$ Decrease only logarithmic.

Problem 3 - Simulating a bent coin

The expression derived in class for the evidence ratio R was:

$$R = \frac{2^F F_a! F_b!}{(F+1)!}$$

Here *F* is the total number of tosses, and *Fa*, *Fb* denote the number of times they came out as *a* or *b*. To calculate *R* as a function of time for a given sequence of tosses, note that its value follows a rather simple update rule. Specifically, if toss #F comes out as *a*:

$$R_F = \frac{2F_a}{F+1} R_{F-1}$$

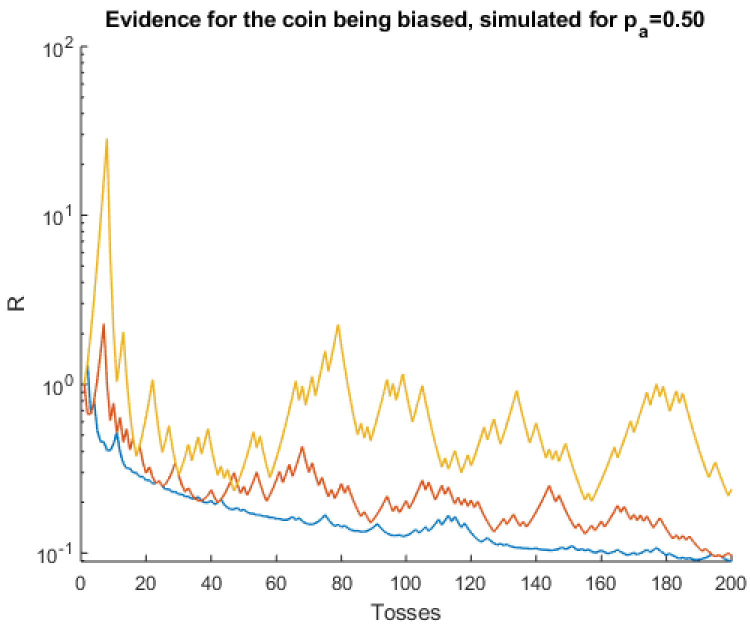
And if, instead, toss #F comes out as *b*:

$$R_F = \frac{2F_b}{F+1} R_{F-1}$$

Simulation code

```
clf;
hold all;
pa = 0.5;
N = 200;
replicates = 3;

s = rand(N,replicates)<pa;
Fa = cumsum(s);
Fb = cumsum(~s);

R = NaN(size(s));
R(1,:) = 1;

% This can be done much more efficiently, without nested for loops.
% But the code below aims for maximum readability:
for i=2:N
    for rep = 1:replicates
        if s(i,rep)
            R(i,rep) = R(i-1,rep)*2*Fa(i,rep)/(i+1);
        else
            R(i,rep) = R(i-1,rep)*2*Fb(i,rep)/(i+1);
        end
    end
end

plot(R,'LineWidth',1)
```

```
title(sprintf('Evidence for the coin being biased, simulated for p_a=%.2f',pa))
xlabel('Tosses'); ylabel('R'); set(gca,'YScale','log');
```



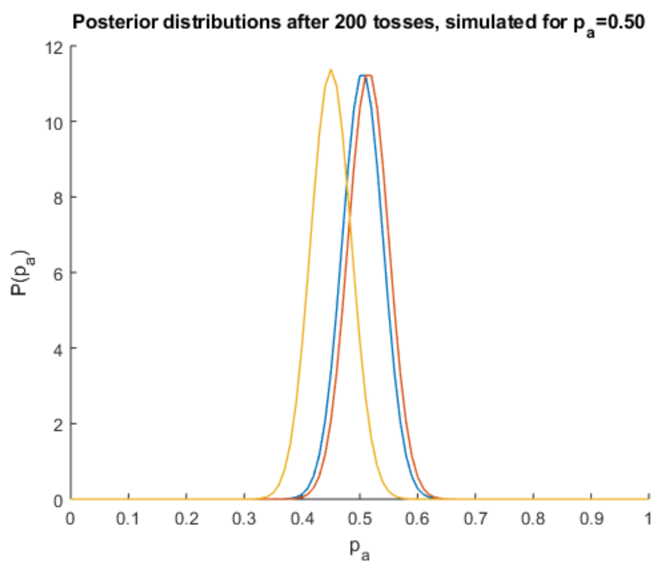
Posterior distributions

The formula for the posterior distribution:

$$P(p_a|s) \propto p_a^{F_a}(1-p_a)^{F_b}$$

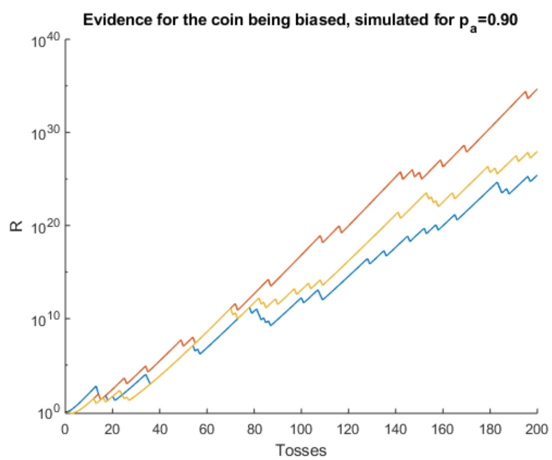
The normalization coefficient has an explicit formula, but for simplicity we can just normalize the distributions numerically.

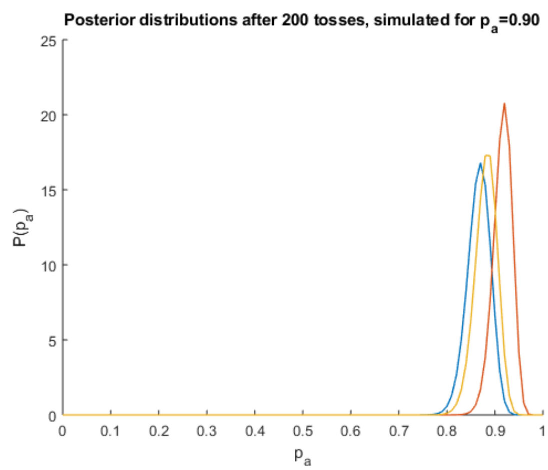
```
clf;
hold all;
step = 0.01;
xs = 0:step:1;
for rep=1:replicates
    final_Fa = Fa(end,rep);
    final_Fb = Fb(end,rep);
    ys = xs.^final_Fa .* (1-xs).^final_Fb;
    ys = ys/(sum(ys*step));
    plot(xs,ys,'-','LineWidth',1);
end
title(sprintf('Posterior distributions after %d tosses, simulated for p_a=%.2f',N,pa))
xlabel('p_a'); ylabel('P(p_a)');
```



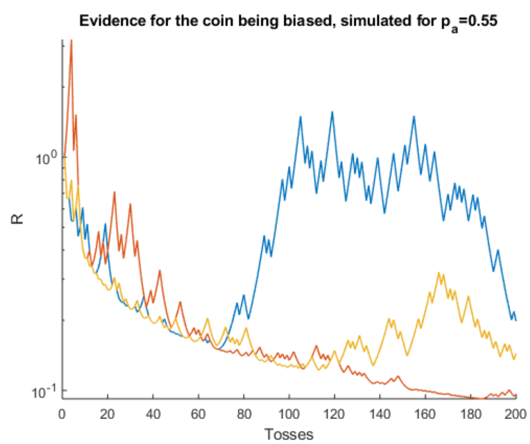
Part (c) – biased coin

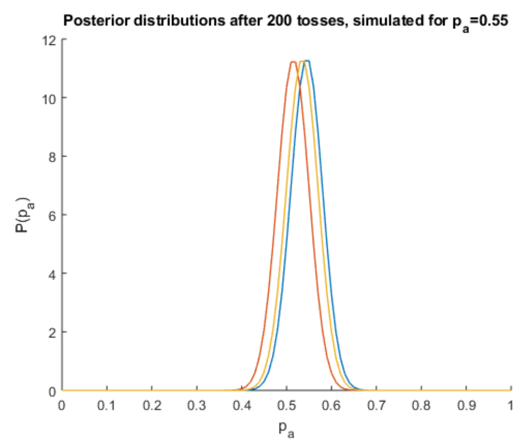
Repeating the same for pa = 0.9, we get the expected behavior:



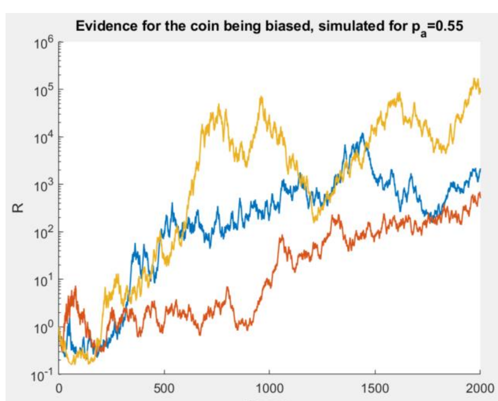


But for pa = 0.55:





Should we be worried that the math seems to favor the unbiased model when the coin is, in fact, biased? Not really, because when p_a is so close to 0.5, N=200 tosses are simply not enough tosses to reliably make the call (see next question). Increasing the number of simulated tosses ten-fold:



Simulations for part (d)

```
function HW2
%%
repN = 1;

paList = 0.53:0.01:0.6;
Ns = zeros(length(paList),repN);
parfor i=1:length(paList)
    for rep=1:repN
        Ns(i,rep) = simulateTrajectory(paList(i),10000,10);
    end
end
%%
mean_Ns = mean(Ns,2)';
std_Ns = std(Ns,[],2)';
clf
errorbar(paList, mean_Ns.*(paList-0.5).^2, std_Ns.*(paList-0.5).^2)
xlabel('p')
ylabel('N*(p-1/2)^2')
title('Check predicted scaling of the number of tosses')
axis([0.5 0.6 0 5 ])
%%
end
```