Bayes: Homework 1
Leslie Gains-Germain

1. The `dhyper(x, m, n, k, log = FALSE)` function gives the probability of drawing $X = x$ white balls without replacement from k draws out of an urn with m white balls and n black balls.

2. My function is below. The default value for x is 1.

```
hyper.fun <- function(theta, x=1) {
  choose(theta, x) * choose((12-theta), (5-x)) / choose(12, 5)
}
hyper.fun(0.1)

## [1] 0.0601
```

3.
```
theta.vec <- matrix(0:12)
nice.table <- matrix(NA, nrow=13, ncol=6)
for(i in 1:6){
  nice.table[,i] <- apply(theta.vec, 1, hyper.fun, x=i-1)
}
```

4.
```
col.sums <- apply(nice.table, 2, sum)
col.sums

## [1] 2.17 2.17 2.17 2.17 2.17 2.17

row.sums <- apply(nice.table, 1, sum)
row.sums

##  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1
```

The rows sum to one because for a given a value of theta, each row describes the probability of drawing x gold marbles. When we sum over the entire support, 0 to 5, we get 1 because the probability that some event in the sample space occurs is 1. When we sum across the columns, we are summing over the likelihood of different values of theta given x is fixed. These do not sum to 1 because the likelihood function is not a probability mass function.

5. Given a fixed x, θ is the independent variable, and the likelihood function, $L(\theta|x)$ is simply a function of θ. It is not a probability mass function because it does not describe the probabilities for a set of possible outcomes of a random event. Instead, it is a function that describes the likelihood for possible values of θ given the random event has occurred and x has been observed. We usually use the notation $f(x|\theta)$ for a probability mass function and $L(\theta|x)$ for a likelihood function.

6. We often use the maximum likelihood principle to estimate θ. We look at the values of the likelihood function across the possible values of θ, and we choose the θ that maximizes the likelihood function (given $X = x$ has been observed). The value of θ that maximizes the likelihood function is our best guess for the true value of θ.

7. An estimator is a random variable defined prior to data collection. For example, $\bar{X}$ is a common estimator, it is a function of the data before the data have been observed. An estimate is a realization of the estimator and is calculated after the data are observed. For example, $\bar{x}$ is a point estimate for a population mean, and it is calculated with observed data $x_1, x_2, .., x_n$.

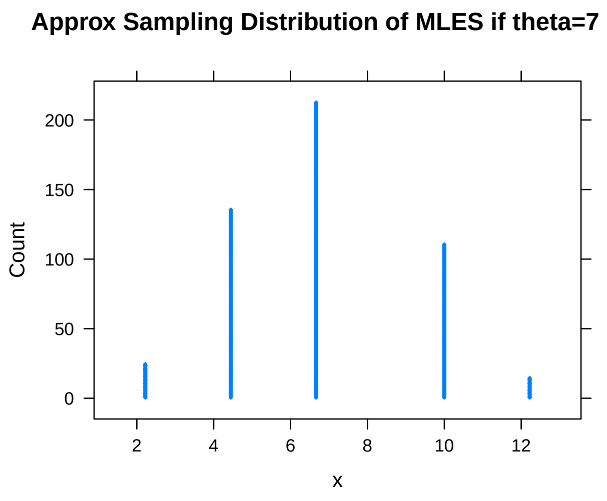
8. The maximum likelihood estimator could be $0, 2, 5, 7, 10$, or 12. I found these by looking at the above table, and they correspond to the 6 possible values of X as in the following table. For example, if we draw $X = 1$, then the likelihood function is maximized at $\hat{\theta} = 2$.

X	$\hat{\theta}$
0	0
1	2
2	5
3	7
4	10
5	12

9. Since we are drawing 5 marbles out of the bucket (with a total of 12 marbles), there are 792 possible samples that could be drawn. I'm going to suppose the true value of θ is 7, and simulate 500 samples. For each sample, I will find the maximum likelihood estimate and build a histogram to give you an idea of what the sampling distribution of ML estimates would look like.

   Here's an approximate sampling distribution of maximum likelihood *estimates* constructed with 500 simulations. It is centered at the true number of gold balls, 7, and the MLE's farther from 7 are less likely to occur.

```
set.seed(5)
sims <- rhyper(500, 7, 5, 5)
mles <- ifelse(sims==0, 0,
               ifelse(sims==1, 2,
                      ifelse(sims==2, 5,
                             ifelse(sims==3, 7,
                                    ifelse(sims==4, 10,
                                           ifelse(sims==5, 12, 0))))))
require(mosaic)
dotPlot(mles, main="Approx Sampling Distribution of MLES if theta=7", cex=2, pch=5)
```



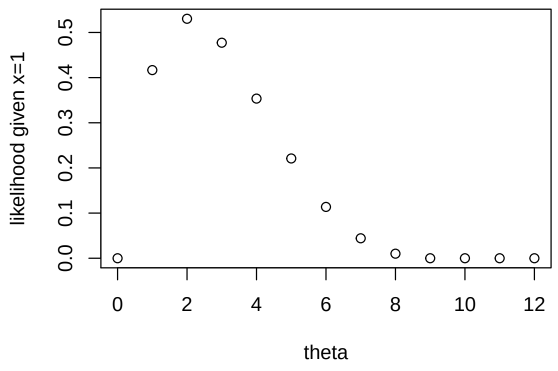
I think that the best way to display the sampling distribution of the maximum likelihood *estimator* is through a table. We would essentially just take the table that we used to describe the likelihood function previously, and list the possible values for the ML estimator at the top. For each true value of θ, the long run relative frequencies of each possible value of the ML estimator are shown in the rows of the table.

	0	2	5	7	10	12
0	1.00	0.00	0.00	0.00	0.00	0.00
1	0.58	0.42	0.00	0.00	0.00	0.00
2	0.32	0.53	0.15	0.00	0.00	0.00
3	0.16	0.48	0.32	0.05	0.00	0.00
4	0.07	0.35	0.42	0.14	0.01	0.00
5	0.03	0.22	0.44	0.27	0.04	0.00
6	0.01	0.11	0.38	0.38	0.11	0.01
7	0.00	0.04	0.27	0.44	0.22	0.03
8	0.00	0.01	0.14	0.42	0.35	0.07
9	0.00	0.00	0.05	0.32	0.48	0.16
10	0.00	0.00	0.00	0.15	0.53	0.32
11	0.00	0.00	0.00	0.00	0.42	0.58
12	0.00	0.00	0.00	0.00	0.00	1.00

10. (a) The maximum likelihood estimate is 2.

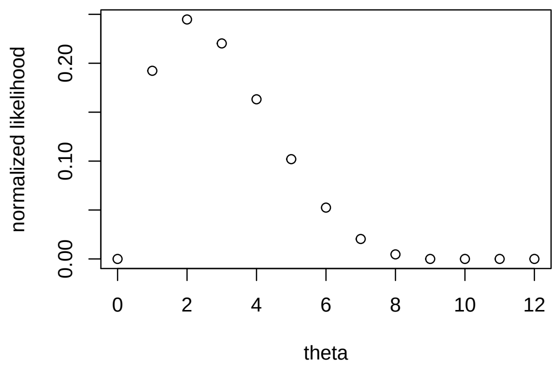
(b) Given x is 1, this plot shows the values of the likelihood function over the range of possible values of θ. You can clearly see that the maximum likelihood estimate is 2. A histogram wouldn't make sense here. I *think* it would show you how many likelihoods were in each 'bin', and the largest bin would be the 0 to 0.1 bin because there are so many zeros. This is not useful information at all.

```r
plot(0:12, nice.table[,2], xlab="theta", ylab="likelihood given x=1")
```



(c) The normalized likelihood function is just a rescaled version of the likelihood function so that the likelihood sums to 1 over all possible θ values.

```r
plot(0:12, nice.table[,2]/sum(nice.table[,2]), xlab="theta", ylab="normalized likelihood")
```

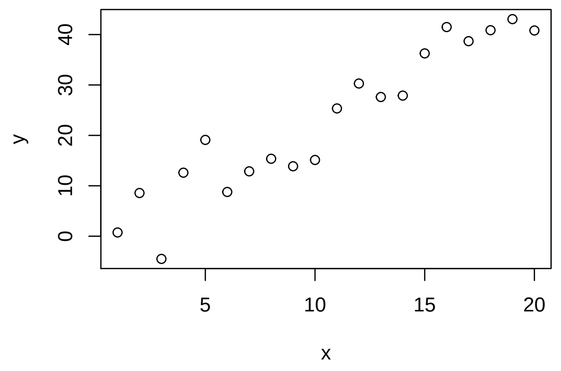


11. (a) Consider the simple linear regression model:

$$y_i = \beta_0 + \beta_1 x_i + e_i$$

I simulated data to investigate this problem.

```r
x <- 1:20
y <- 3 + 2*x + rnorm(20,0,5)
data.sim <- data.frame(cbind(x,y))
plot(x,y)
```



```r
lm.fit <- lm(y ~ x, data = data.sim)
coef(lm.fit)

## (Intercept)           x
##       -1.63        2.32

glm.fit <- glm(y ~ x, data = data.sim, family = "gaussian")
coef(lm.fit)

## (Intercept)           x
##       -1.63        2.32
```

The `lm()` procedure finds the ordinary least squares estimates for β_0 and β_1 by minimizing the sum of squared residuals. The distributional assumption is that $\epsilon_i \sim (0, \sigma^2)$. Note that the assumption for normality is not actually needed to calculate the OLS estimates - the assumption of normality is only made in order to calculate p-values and confidence intervals.

In the `glm()` procedure, we specify `family="gaussian"` and an identity link function, so the distributional assumption is: $\epsilon_i \sim N(0, \sigma^2)$. The default procedure used to find the estimates in `glm()` is maximum likelihood estimation, and it does this using iteratively reweighted least squares. Because we are assuming normally distributed errors, the maximum likelihood estimate is the same as the ordinary least squares estimate from `lm`.

(b) The `lm` procedure gives t-based confidence intervals when the `confint` function is used. It calculates these intervals by adding and subtracting from the estimates the standard error multiplied by the t multiplier.

The `glm` procedure gives likelihood based confidence intervals. It finds these by plotting the likelihood over the entire parameter space, and then it finds the 2.5th and 97.5th percentiles from the likelihood function. I think it looks at the marginal likelihood function separately for each parameter to do this?

```
confint(lm.fit)

##             2.5 % 97.5 %
## (Intercept) -6.32   3.06
## x            1.93   2.71

confint(glm.fit)

## Waiting for profiling to be done...

##             2.5 % 97.5 %
## (Intercept) -6.00   2.74
## x            1.96   2.69
```

12. This is how I think of Bayes Theorem:

$$f(\theta|x) = \frac{f(x|\theta) * f(\theta)}{f(x)}$$

A lot of books use p‘s for probability mass functions. In the bucket problem, both X and θ are discrete random variables, so the notation I would use for this problem is as follows.

$$p(\theta|x) = \frac{p(x|\theta) * p(\theta)}{p(x)}$$

13. In the bucket problem, we replace $p(x|\theta)$ with the likelihood function, $L(\theta|x)$, which we know to be a hypergeometric distribution. The marginal distribution in the denominator is found by finding the joint distribution of x and θ and then summing over all values of θ.

$$\begin{aligned} p(\theta|x) &= \frac{p(x|\theta) * p(\theta)}{p(x)} \\ &= \frac{L(\theta|x) * p(\theta)}{\sum_{\theta \in \Theta} L(\theta|x) * p(\theta)} = \frac{\frac{\binom{\theta}{x}\binom{12-\theta}{5-x}}{\binom{12}{5}} * p(\theta)}{\sum_{\theta=0}^{\theta=12} \frac{\binom{\theta}{x}\binom{1-\theta}{5-x}}{\binom{12}{5}} * p(\theta)} \end{aligned}$$

14. (a)

```
theta <- 0:12
priors <- c(0, 0, 1/36, 2/36, 3/36, 4/36, 5/36, 6/36, 5/36, 4/36, 3/36, 2/36, 1/36)
likelihood <- apply(theta.vec, 1, hyper.fun, x = 1)
product <- likelihood*priors
posterior <- likelihood*priors/sum(product)
dice.sum <- cbind(theta, priors, likelihood, product, posterior)
dice.sum

##       theta priors likelihood product posterior
##  [1,]     0 0.0000     0.0000 0.00000    0.0000
##  [2,]     1 0.0000     0.4167 0.00000    0.0000
##  [3,]     2 0.0278     0.5303 0.01473    0.1230
##  [4,]     3 0.0556     0.4773 0.02652    0.2213
##  [5,]     4 0.0833     0.3535 0.02946    0.2459
## [6,]     5 0.1111     0.2210 0.02455    0.2049
## [7,]     6 0.1389     0.1136 0.01578    0.1317
## [8,]     7 0.1667     0.0442 0.00737    0.0615
## [9,]     8 0.1389     0.0101 0.00140    0.0117
## [10,]    9 0.1111     0.0000 0.00000    0.0000
## [11,]   10 0.0833     0.0000 0.00000    0.0000
## [12,]   11 0.0556     0.0000 0.00000    0.0000
## [13,]   12 0.0278     0.0000 0.00000    0.0000
```

(b)
```
priors2 <- c(0, 1/6, 1/6, 1/6, 1/6, 1/6, 1/6, 0, 0, 0, 0, 0, 0)
product2 <- likelihood*priors2
posterior2 <- likelihood*priors2/sum(product2)
dice.one <- cbind(theta, priors2, likelihood, product2, posterior2)
dice.one
```

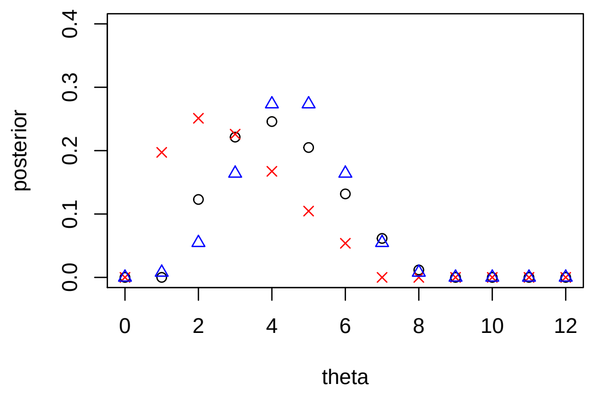
```
##       theta priors2 likelihood product2 posterior2
##  [1,]     0   0.000     0.0000   0.0000     0.0000
##  [2,]     1   0.167     0.4167   0.0694     0.1973
##  [3,]     2   0.167     0.5303   0.0884     0.2510
##  [4,]     3   0.167     0.4773   0.0795     0.2259
##  [5,]     4   0.167     0.3535   0.0589     0.1674
##  [6,]     5   0.167     0.2210   0.0368     0.1046
##  [7,]     6   0.167     0.1136   0.0189     0.0538
##  [8,]     7   0.000     0.0442   0.0000     0.0000
##  [9,]     8   0.000     0.0101   0.0000     0.0000
## [10,]     9   0.000     0.0000   0.0000     0.0000
## [11,]    10   0.000     0.0000   0.0000     0.0000
## [12,]    11   0.000     0.0000   0.0000     0.0000
## [13,]    12   0.000     0.0000   0.0000     0.0000
```

(c)
```
priors3 <- dbinom(0:12, 12, 1/2)
product3 <- likelihood*priors3
posterior3 <- likelihood*priors3/sum(product3)
```

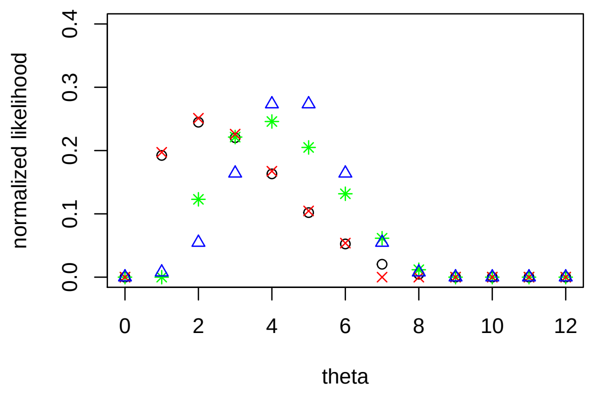
```
coin.flip <- cbind(theta, priors3, likelihood, product3, posterior3)
coin.flip

##       theta  priors3 likelihood product3 posterior3
##  [1,]     0 0.000244     0.0000  0.00000    0.00000
##  [2,]     1 0.002930     0.4167  0.00122    0.00781
##  [3,]     2 0.016113     0.5303  0.00854    0.05469
##  [4,]     3 0.053711     0.4773  0.02563    0.16406
##  [5,]     4 0.120850     0.3535  0.04272    0.27344
##  [6,]     5 0.193359     0.2210  0.04272    0.27344
##  [7,]     6 0.225586     0.1136  0.02563    0.16406
##  [8,]     7 0.193359     0.0442  0.00854    0.05469
##  [9,]     8 0.120850     0.0101  0.00122    0.00781
## [10,]     9 0.053711     0.0000  0.00000    0.00000
## [11,]    10 0.016113     0.0000  0.00000    0.00000
## [12,]    11 0.002930     0.0000  0.00000    0.00000
## [13,]    12 0.000244     0.0000  0.00000    0.00000
```

15. Yes, they do make sense. We can see that across all posterior functions, if either the likelihood or the prior probability was 0 for a given value of θ, then the posterior probability is 0 for that value of θ as well. It also makes sense that the one die prior gives a higher posterior probability for the lower values of θ (1 and 2) compared to the other two priors that give higher posteriors probabilities for the higher values of θ (5, and 6). I think the one die prior is the least informative because it yields a prior that looks closest to the likelihood function. The posterior distribution differs from the likelihood much more when the other two priors are used. It is surprising to me how much the choice of prior can affect the posterior distribution.



16. This confirms what I speculated on in the previous problem. The one die prior matches up closest to the normalized likelihood function. This is because this prior is the least informative - it just puts a discrete uniform distribution values of θ between 1 and 6. This allows the likelihood to take the primary role in shaping the posterior distribution.



17. See attached.