Stat 230: Linear Models
Homework 4
Professor Ding
Lev Golod

QUESTION 5, PART (1)

What happens if you don't use a log-transformation?
Certain terms like n^n will go to infinity and mess up the calculation.

```
## A function to calculate log(f(k,n,p,phi)) for one value of k
logf <- function(k, n = 100, p = 0.3, phi = 0.5){

  terms = rep(NA, 7)
  terms[1] = lgamma(n+1)                          # does not depend on k
  terms[2] = -lgamma(k+1)
  terms[3] = -lgamma(n-k+1)

  # terms[4] = ((-phi+1) * k )     * log(k)
  # terms[5] = ((-phi+1) * (n-k))  * log(n-k)
  # terms[6] = ((-phi+1) * n )     * log(n)         # does not depend on k

  terms[4] = ((-phi+1) * k )     * log(k/n)
  terms[5] = ((-phi+1) * (n-k) ) * log((n-k)/n)

  terms[6] = (phi * k )          * log(p)
  terms[7] = (phi * (n-k))       * log(1-p)

  terms <- ifelse(is.nan(terms),0, terms)
  sum_terms <- sum(terms)

  return(sum_terms)

}


## Find the denominator for n = 1000
denom <- sum(sapply(seq.int(0,1000), function(k) exp(logf(k,n=1000))))
# denom


## Find the denominator for n = 100, 500, 1000, 2000
ns <- c(100, 500, 1000, 2000)
names(ns) <- paste0('n=',ns)
denoms <- sapply(ns, function(n) sum(sapply(seq.int(0,n),
                                            function(k) exp(logf(k,n)))))
print(denoms)
```

```
##  n=100  n=500 n=1000 n=2000
## 1.4189 1.4151 1.4147 1.4144
```

```
## determine average runtime for each n value
reps=100
times <- sapply(ns, function(n)
  system.time( replicate(reps, sum(sapply(seq.int(0,n),
```

```
                                          function(k) exp(logf(k,n)))))
  )[3]
)
print(times)
```

```
##  n=100.elapsed  n=500.elapsed n=1000.elapsed n=2000.elapsed
##          0.269          1.174          2.442          4.764
```

QUESTION 5, PART (2)

```
## vectorized fn to find the denominator
vecdenom <- function(n, p = 0.3, phi = 0.5){

  ks <- seq.int(0,n)
  terms <- rep(lgamma(n+1), n+1) +
   -lgamma(ks+1) +
   -lgamma(n-ks+1) +
   ifelse(ks == 0 | ks == 100, 0, ((-phi+1) * ks ) * log(ks/n) ) +
   ifelse(ks == 0 | ks == 100, 0, ((-phi+1) * (n-ks) ) * log((n-ks)/n) ) +
   (phi * ks ) * log(p) +
   (phi * (n-ks)) * log(1-p)
    # + phi * ( k * log(p/(1-p))  +  n * log(1-p) )
  sum(exp(terms))

}

denoms2 <- sapply(ns, vecdenom)
print(denoms)
```

```
##  n=100  n=500 n=1000 n=2000
## 1.4189 1.4151 1.4147 1.4144
```

```
times2 <- sapply(ns, function(n) system.time(replicate(reps, vecdenom(n)))[3] )
print(times2)
```

```
##  n=100.elapsed  n=500.elapsed n=1000.elapsed n=2000.elapsed
##          0.010          0.020          0.033          0.063
```

QUESTION 5, PART (3)

```
cat(sprintf('How much faster is the vectorized version when n = 2000?
%s times faster', floor(times[4]/times2[4])))
```

```
## How much faster is the vectorized version when n = 2000?
## 75 times faster
```