DATA605 Homework 9

Keith Colella

2023-10-29

```
library(tidyverse)
```

Question 1

Chapter 9.3 Q#11

The price of one share of stock in the Pilsdorff Beer Company is given by Y_n on the nth day of the year. Finn observes that the differences $X_n = Y_{n+1} - Y_n$ appear to be independent random variables with a common distribution having mean $\mu = 0$ and variance $\sigma^2 = 1/4$. If $Y_1 = 100$, estimate the probability that $Y_3 65$ is
(a) ≥ 100
(b) ≥ 110
(c) ≥ 120

Response

The first thing we need to establish is how we can formulate Y_{365}. We are given Y_1, and the problem tells us that the daily differences in price $Y_{n+1} - Y_n$ are a series of random variables denoted X_n. If we think about a stock price on a given day, we can express it as the sum of the previous day's price and the change price. For example:

$$Y_2 = Y_1 + (Y_2 - Y_1)$$

If we sub in our random variable representing the daily change in price ($X_1 = Y_2 - Y_1$), we get the following:

$$Y_2 = Y_1 + X_1$$

The next day (Y_3) will be the same, and we can sub in the previous values to things back to our known value of Y_1:

$$Y_3 = Y_2 + X_2$$
$$Y_3 = Y_1 + X_1 + X_2$$

If we continue to Y_4, we see a pattern emerge.

$$Y_4 = Y_3 + X_3$$
$$Y_4 = Y_1 + X_1 + X_2 + X_3$$

So, it is now clear that Y_n can be constructed as $Y_1 + \sum_{i=1}^{n-1} X_n$, or in our case, $Y_{365} = Y_1 + \sum_{i=1}^{364} X_n$

This sum of independent random variables (i.e. the X_n's) gives us a chance to apply the CLT. The CLT tells us that the sum of a large number of independent random variables will be approximately normally distributed. We also know that this distribution will have a mean of $n\mu$ and variance $n\sigma^2$, where μ and σ^2 are the mean of variance of the random variables themselves (again, the X_n's), which we know to be 0 and 1/4.

So, let's lay out the key values we have so far.

```
# Mean and standard deviation of X_n random variables
mean_Xn = 0
variance_Xn = 1/4

# Number of independent variables
n = 364

# Mean and sd of sum of independent variables
mean_CLT = n * mean_Xn
variance_CLT = n * variance_Xn
```

Going back to our formulation of Y_{365}, its expected value will equal the sum of Y_1 and the expected value of $\sum_{i=1}^{n-1} X_n$, which the same variance as our distribution of X_n's.

```
mean_Y365 = 100 + mean_CLT
```

Because the distribution of X_n's is approximately normal, we can use the `pnorm` function to estimate the probabilities. This function gives us the CDF of a normal distribution for a given quantile `q`, i.e. $P(Y_{365} \leq q)$. We're looking for the opposite, i.e. $P(Y_{365} \geq q)$, so we'll take the complement of the `pnorm` result.

Note that this function leverages standard deviation rather than variance, so we'll need to take the square root of the variance calculated above.

```
p_100 = 1 - pnorm(100, mean = mean_Y365, sd = sqrt(variance_CLT))
p_110 = 1 - pnorm(110, mean = mean_Y365, sd = sqrt(variance_CLT))
p_120 = 1 - pnorm(120, mean = mean_Y365, sd = sqrt(variance_CLT))

cat(
  'P(Y_365 >= 100) = ',p_100,'\n',
  'P(Y_365 >= 110) = ',p_110,'\n',
  'P(Y_365 >= 120) = ',p_120,
  sep = ''
)
```

```
## P(Y_365 >= 100) = 0.5
## P(Y_365 >= 110) = 0.1472537
## P(Y_365 >= 120) = 0.01801584
```

Question 2

Calculate the expected value and variance of the binomial distribution using the moment generating function.

Response

The moment generating function (MGF) for a binomial distribution is given as follows:

$$M_X(t) = (1 - p + pe^t)^n$$

The first moment is our expected value $E(X)$, which we can find by taking the derivative of the MGF and evaluating it at zero.

$$E(X) = M'_X(0) = \frac{d}{dt} M_X(t)|_{t=0}$$

The variance is defined as the difference between the second moment and the square of the first moment, i.e. $V(X) = E(X^2) - E(X)^2$. While we can simply square the value obtained above for our first moment, finding the second moment will require taking the *second* derivative of the MGF and evaluating it at zero.

$$E(X^2) = M''_X(0) = \frac{d^2}{dt^2} M_X(t)|_{t=0}$$

I'll rely on the `expression` and `D` functions to calculate the derivatives of these functions in R. We can tidy things up a bit by using q to denote the complement of p, i.e. $1 - p$.

```
mgf <- expression( (q + p * exp(t))^n )
first_moment <- D(mgf, 't')
second_moment <- D(first_moment, 't')

print(first_moment)
```

```
## (q + p * exp(t))^(n - 1) * (n * (p * exp(t)))
```

```
print(second_moment)
```

```
## (q + p * exp(t))^((n - 1) - 1) * ((n - 1) * (p * exp(t))) * (n *
##     (p * exp(t))) + (q + p * exp(t))^(n - 1) * (n * (p * exp(t)))
```

Pretty hairy! Luckily, we can use the `eval` function to evaluate these directly. Let's take an example with $n = 100$ and $p = 0.8$.

```
expected_val_binom <- function(n,p,q,t) eval(first_moment)
variance_binom <- function(n,p,q,t) eval(second_moment) - eval(first_moment)^2

n = 100
p = 0.8
q = 1 - p

cat(
  'Expected Value: ',expected_val_binom(n,p,q,0),'\n',
  'Variance: ',variance_binom(n,p,q,0),
  sep = ''
)
```

```
## Expected Value: 80
## Variance: 16
```

We can check this using our knowledge of the binomial distribution, for which the mean is np and variance is npq.

```
cat(
  'Expected Value: ',n*p,'\n',
  'Variance: ',n*p*q,
  sep = ''
)
```

```
## Expected Value: 80
## Variance: 16
```

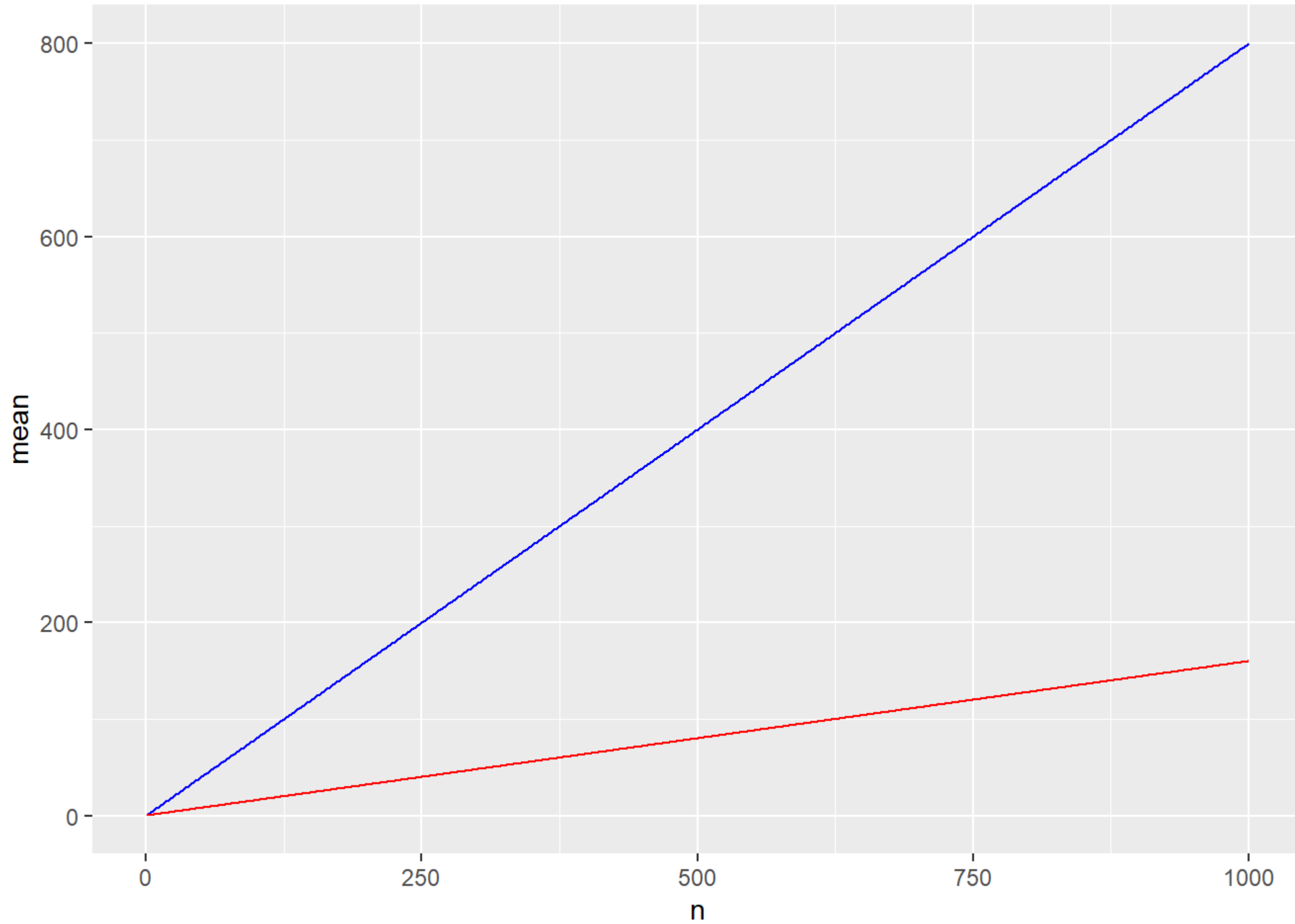
They match!

Since we have these functions, let's create a few plots to see how the mean and variance respond to varied levels of n.

```
results <- data.frame()
p <- 0.8
q <- 1 - p

for (n in 1:1000) {
  mean <- expected_val_binom(n,p,q,0)
  variance <- variance_binom(n,p,q,0)
  result <- list(n = n, mean = mean, variance = variance)
  results <- rbind (results, result)
}

results %>%
  ggplot() +
  geom_line(aes(x = n, y = mean), color = 'blue') +
  geom_line(aes(x = n, y = variance), color = 'red')
```

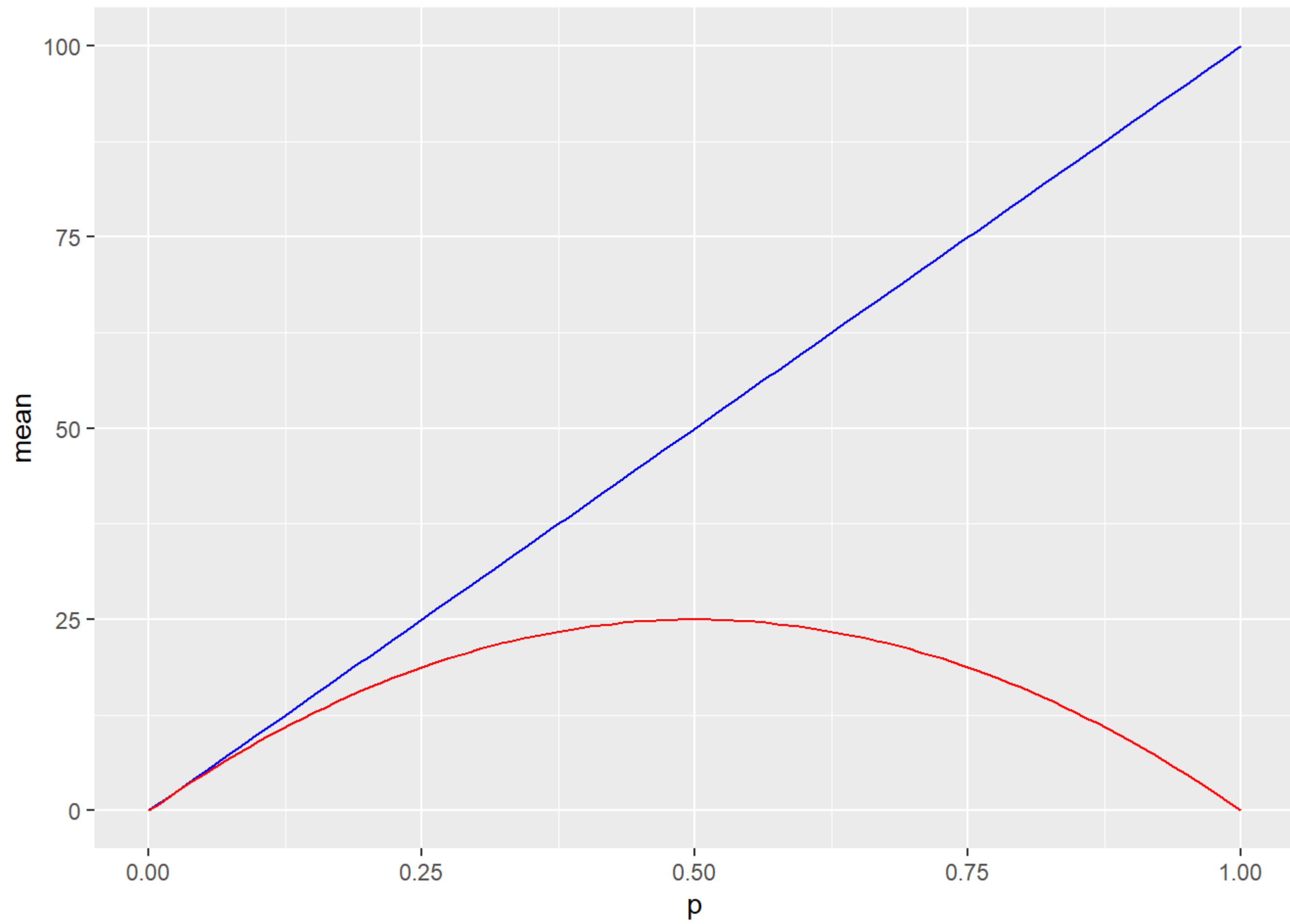


Fun! And now if we vary p.

```
results <- data.frame()
n = 100

for (p in seq(0, 1, length.out = 100)) {
  q <- 1 - p
  mean <- expected_val_binom(n,p,q,0)
  variance <- variance_binom(n,p,q,0)
  result <- list(p = p, mean = mean, variance = variance)
  results <- rbind (results, result)
}

results %>%
  ggplot() +
  geom_line(aes(x = p, y = mean), color = 'blue') +
  geom_line(aes(x = p, y = variance), color = 'red')
```



Even more fun!

Question 3

Calculate the expected value and variance of the exponential distribution using the moment generating function.

Response

We can apply the same framework as we did in problem #2. But now, the starting MGF will be different, defined for an exponential distribution as follows:

$$(1 - t\lambda^{-1})^{-1}, t < \lambda$$

We can express this a bit more cleanly as follows:

$$\frac{\lambda}{(\lambda - t)}$$

As with before, the first moment will be the first derivative of this function evaluated at zero, and the second moment will be the second derivative evaluated at zero.

```
mgf <- expression( lambda / (lambda - t) ) # (1 - t * lambda^-1)^-1 )
first_moment <- D(mgf, 't')
second_moment <- D(first_moment, 't')

print(first_moment)
```

```
## lambda/(lambda - t)^2
```

```
print(second_moment)
```

```
## lambda * (2 * (lambda - t))/((lambda - t)^2)^2
```

I'll use the `eval` function again to test these out. We can try it with $\lambda = 1/10$.

```
expected_val_expo <- function(lambda,t) eval(first_moment)
variance_expo <- function(lambda,t) eval(second_moment) - eval(first_moment)^2

lambda = 1/10

cat(
  'Expected Value: ',expected_val_expo(lambda,0),'\n',
  'Variance: ',variance_expo(lambda,0),
  sep = ''
)
```

```
## Expected Value: 10
## Variance: 100
```

And we'll check that with the known definition of mean and variance for an exponential distribution, i.e. $\mu = 1/\lambda$ and $\sigma^2 = 1/\lambda^2$.

```
cat(
  'Expected Value: ', 1 / lambda,'\n',
  'Variance: ', 1 / lambda^2,
  sep = ''
)
```

```
## Expected Value: 10
## Variance: 100
```

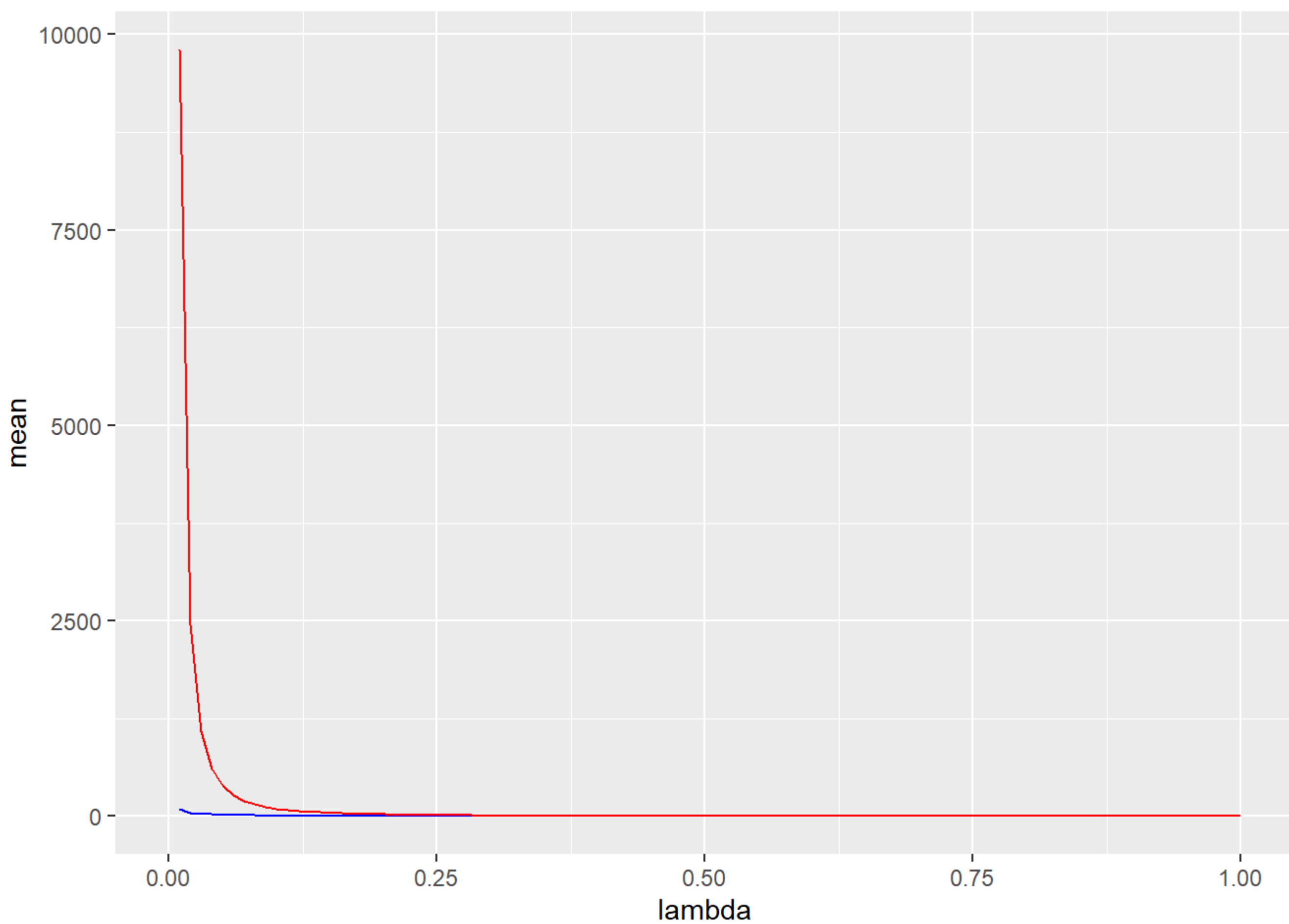
Another match! Let's do some plotting.

```
results <- data.frame()

for (lambda in seq(0, 1, length.out = 100)) {
  mean <- expected_val_expo(lambda,0)
  variance <- variance_expo(lambda,0)
  result <- list(lambda = lambda, mean = mean, variance = variance)
  results <- rbind (results, result)
}

results %>%
  ggplot() +
  geom_line(aes(x = lambda, y = mean), color = 'blue') +
  geom_line(aes(x = lambda, y = variance), color = 'red')
```

```
## Warning: Removed 1 row containing missing values (`geom_line()`).
## Removed 1 row containing missing values (`geom_line()`).
```



Fun!