Transformations Notes

Lt Col Ken Horton   Lt Col Kris Pruitt   Professor Bradley Warner

08 October, 2020

Objectives

1) Given a discrete random variable, determine the distribution of a transformation of that random variable.
2) Given a continuous random variable, use the cdf method to determine the distribution of a transformation of that random variable.
3) Use simulation methods to find the distribution of a transform of single or multivariate random variables.

Transformations

Throughout our coverage of random variables, we have mentioned transformations of random variables. These have been in the context of linear transformations. We have discussed expected value and variance of linear transformations. Recall that $\mathrm{E}(aX+b)=a\mathrm{E}(X)+b$ and $\mathrm{Var}(aX+b)=a^2\mathrm{Var}(X)$.

In this lesson, we will discuss transformations of random variables in general, beyond the linear case.

Transformations of discrete random variables

Let X be a discrete random variable and let g be a function. The variable $Y=g(X)$ is a discrete random variable with pmf:

$$f_Y(y)=\mathrm{P}(Y=y)=\sum_{g(x)=y}\mathrm{P}(X=x)=\sum_{g(x)=y}f_X(x)$$

An example would help since the notation can be confusing.

> *Example*:
> Suppose X is a discrete random variable with pmf:
>
> $$f_X(x)=\begin{cases} 0.05, & x=-2 \\ 0.10, & x=-1 \\ 0.35, & x=0 \\ 0.30, & x=1 \\ 0.20, & x=2 \\ 0, & \text{otherwise} \end{cases}$$

Find the pmf for $Y=X^2$.

It helps to identify the domain of Y. Since the domain of X is $S_X = \{-2, -1, 0, 1, 2\}$, the domain of Y is $S_Y = \{0, 1, 4\}$.

$$f_Y(0) = \sum_{x^2=0} f_X(x) = f_X(0) = 0.35$$

$$f_Y(1) = \sum_{x^2=1} f_X(x) = f_X(-1) + f_X(1) = 0.1 + 0.3 = 0.4$$

$$f_Y(4) = \sum_{x^2=4} f_X(x) = f_X(-2) + f_X(2) = 0.05 + 0.2 = 0.25$$

So,

$$f_Y(y) = \begin{cases} 0.35, & y = 0 \\ 0.4, & y = 1 \\ 0.25, & y = 4 \\ 0, & \text{otherwise} \end{cases}$$

It also helps to confirm that these probabilities add to one, which they do. This is the pmf of $Y = X^2$.

The key idea is to find the domain of the new random variable and then go back to the original random variable and sum all the probabilities that get mapped into that new domain element.

Transformations of continuous random variables

The methodology above will not work directly in the case of continuous random variables. This is because in the continuous case, the pdf, $f_X(x)$, represents **density** and not **probability**.

The cdf method

The **cdf method** can be used for transformations of continuous random variables. The idea is to find the cdf of the new random variable and then by way of the fundamental theorem of calculus.

Suppose X is a continuous random variable with cdf $F_X(x)$. Let $Y = g(X)$. We can find the cdf of Y as:

$$F_Y(y) = \mathrm{P}(Y \le y) = \mathrm{P}(g(X) \le y) = \mathrm{P}(X \le g^{-1}(y)) = F_X(g^{-1}(y))$$

To get the pdf of Y, we would need to take the derivative of the cdf.

This method requires the transformation function to have an inverse. Sometimes, we break the domain of the original random variables into regions where an inverse of the transformation function exists.

> *Example*: Let $X \sim \mathsf{Unif}(0, 1)$ and let $Y = X^2$. Find the pdf of Y.

Before we start, let's think about this. We are randomly taking numbers between 0 and 1 and then squaring them. Squaring a positive number less than 1 makes it even smaller. We thus suspect the pdf of Y will have larger density near 0 than 1. The shape is hard to determine so let's do some math.

Since X has the uniform distribution, we know that $F_X(x) = x$ for $0 \le x \le 1$. So,

$$F_Y(y) = \mathrm{P}(Y \le y) = \mathrm{P}(X^2 \le y) = \mathrm{P}(X \le \sqrt{y}) = F_X(\sqrt{y}) = \sqrt{y}$$

Taking the derivative of this yields:

$$f_Y(y) = \frac{1}{2\sqrt{y}}$$

for $0 < y \leq 1$ and 0 otherwise. Notice we can't have $y = 0$ since we would be dividing by zero. This is not a problem since we have a continuous distribution. We could verify this a proper pdf by determining if the pdf integrates to 1 over the domain:

$$\int_0^1 \frac{1}{2\sqrt{y}}\,\mathrm{d}y = \sqrt{y}\bigg|_0^1 = 1$$

We can also do this using `R` but we first have to create a function that can take vector input.

```
y_pdf <- function(y) {
  1/(2*sqrt(y))
}
```

```
y_pdf<- Vectorize(y_pdf)
```

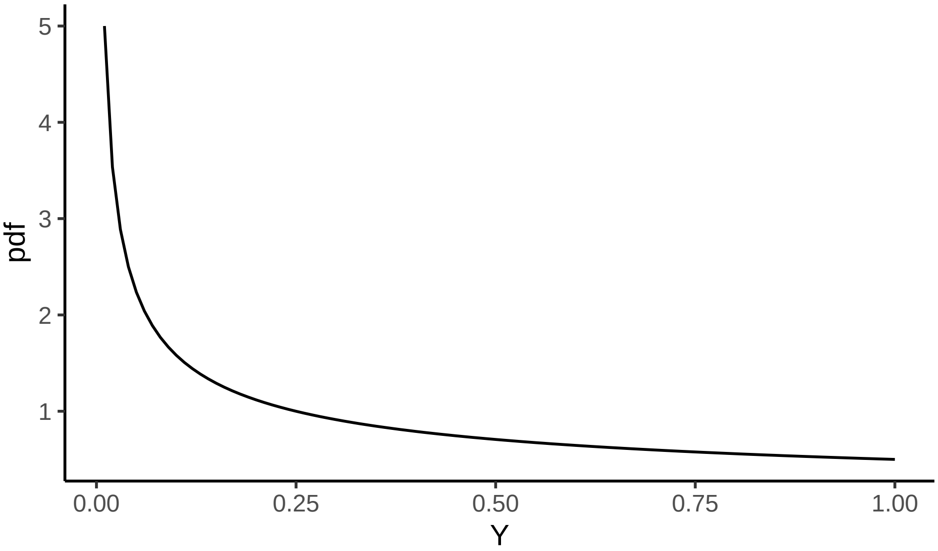
```
integrate(y_pdf,0,1)
```

```
## 1 with absolute error < 2.9e-15
```

Notice that since the domain of the original random variable was non-negative, the squared function had an inverse.

The pdf looks like:

```
gf_line(y_pdf(seq(0.01,1,.01))~seq(0.01,1,.01),xlab="Y",ylab="pdf") %>%
  gf_theme(theme_classic())
```



We can see that the density is much larger at we approach 0.

The pdf method - Optional

The cdf method of transforming continuous random variables also yields to another method called the **pdf method**. Recall that the cdf method tells us that if X is a continuous random variable with cdf F_X, and $Y = g(X)$, then

$$F_Y(y) = F_X(g^{-1}(y))$$

We can find the pdf of Y by differentiating the cdf:

$$f_Y(y) = \frac{\mathrm{d}}{\mathrm{d}y} F_Y(y) = \frac{\mathrm{d}}{\mathrm{d}y} F_X(g^{-1}(y)) = f_X(g^{-1}(y)) \left| \frac{\mathrm{d}}{\mathrm{d}y} g^{-1}(y) \right|$$

So, as long as g^{-1} is differentiable, we can use this method to directly obtain the pdf of Y.

Note that in some texts, the portion of this expression $\frac{\mathrm{d}}{\mathrm{d}y} g^{-1}(y)$ is sometimes referred to as the *Jacobian*. We need to take the absolute value of the transformation function $g(x)$ because if it is a decreasing function, we have

$$F_Y(y) = \mathrm{P}(Y \le y) = \mathrm{P}(g(X) \le y) = \mathrm{P}(X \ge g^{-1}(y)) = 1 - F_X(g^{-1}(y))$$

> **Exercise**:
> Repeat the previous example using the pdf method.

Since X has the uniform distribution, we know that $f_X(x) = 1$ for $0 \le x \le 1$. Also, $g(x) = x^2$ and $g^{-1}(y) = \sqrt{y}$, which is differentiable. So,

$$f_Y(y) = f_X(\sqrt{y}) \left| \frac{\mathrm{d}}{\mathrm{d}y} \sqrt{y} \right| = \frac{1}{2\sqrt{y}}$$

Simulation

We can also get an estimate of the distribution by simulating the random variable. If we have the cdf and can find its inverse, then just like we did in an earlier lesson, we sample from a uniform distribution and apply the inverse to get the distribution.

In an earlier lesson we had

> Let X be a continuous random variable with $f_X(x) = 2x$ where $0 \le x \le 1$.

Now let's find the distribution of $Y = \ln X$.

The cdf of X is $F_X(x) = x^2$ where $0 \le x \le 1$. We will draw a uniform random and then take the square root. We will replicate this 10,000 times. In `R` our code, which we have done before, is:

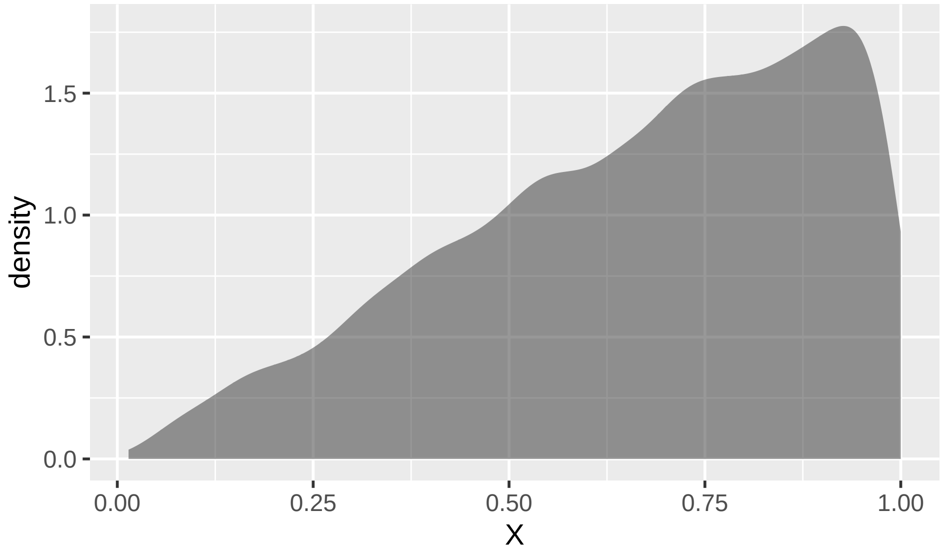
```
results <- do(10000)*sqrt(runif(1))
```

Remember, we are using the square root because we want the inverse of the cdf and not, for this method, the inverse of the transformation function as when we were using the mathematical method. This can be a point of confusion.

```
inspect(results)
```

```
## 
## quantitative variables:  
##      name   class        min        Q1    median        Q3       max      mean
## ...1 sqrt numeric 0.01434807 0.4981034 0.7029968 0.8604899 0.9999993 0.6628707
##             sd     n missing
## ...1 0.2344587 10000       0
```

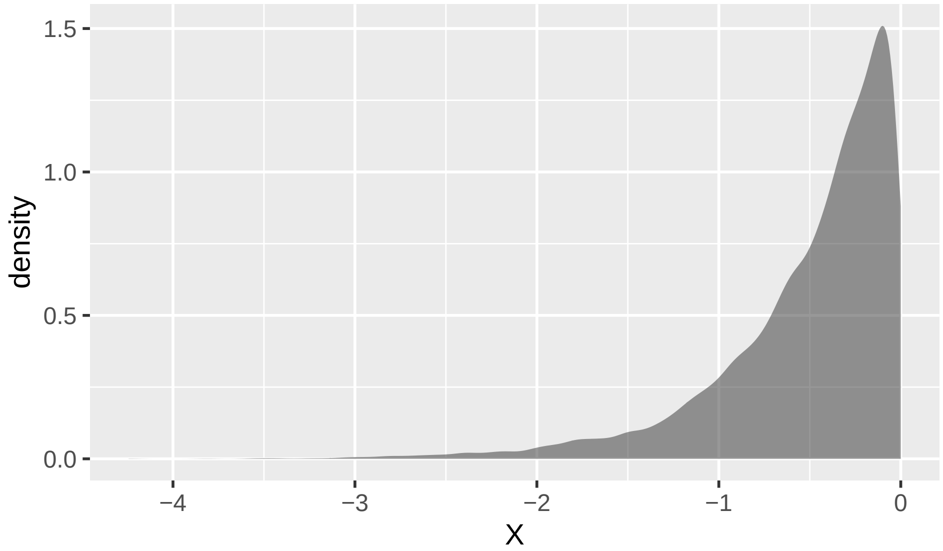
```
results %>%
  gf_density(~sqrt,xlab="X")
```



Now to find the distribution of Y we just apply the transformation.

```
y_results <- results %>%
  transmute(y=log(sqrt))
```

```
y_results %>%
  gf_density(~y,xlab="X")
```



```
inspect(y_results)
```

```
##
```

```
## quantitative variables:
##      name   class      min         Q1     median         Q3           max
## ...1    y numeric -4.24414 -0.6969476 -0.3524029 -0.1502534 -6.852211e-07
##            mean        sd     n missing
## ...1 -0.5047572 0.4944043 10000       0
```

Multivariate Transformations

Here's the scenario. Suppose X and Y are independent random variables, both uniformly distributed on $[5,6]$.

$$X \sim \textsf{Unif}(5,6) \qquad\qquad Y \sim \textsf{Unif}(5,6)$$

Let X be your arrival time for dinner and Y your friends arrival time. We picked 5 to 6 because this is the time in the evening we want to meet. Assume you travel independently.

Define Z as a transformation of X and Y such that $Z = |X - Y|$. Thus Z is the absolute value of the difference between your arrival times. The units for Z are hours. We would like to explore the distribution of Z. We could do this via Calc III methods but we will simulate instead.

We can use `R` to obtain simulated values from X and Y (and thus find Z).

First, simulate 100,000 observations from the uniform distribution with parameters 5 and 6. Assign those random observations to a variable. Next, repeat that process, assigning those to a different variable. These two vectors represent your simulated values from X and Y. Finally, obtain your simulated values of Z by taking the absolute value of the difference.

Exercise:

Complete the code on your own before looking at the code below.

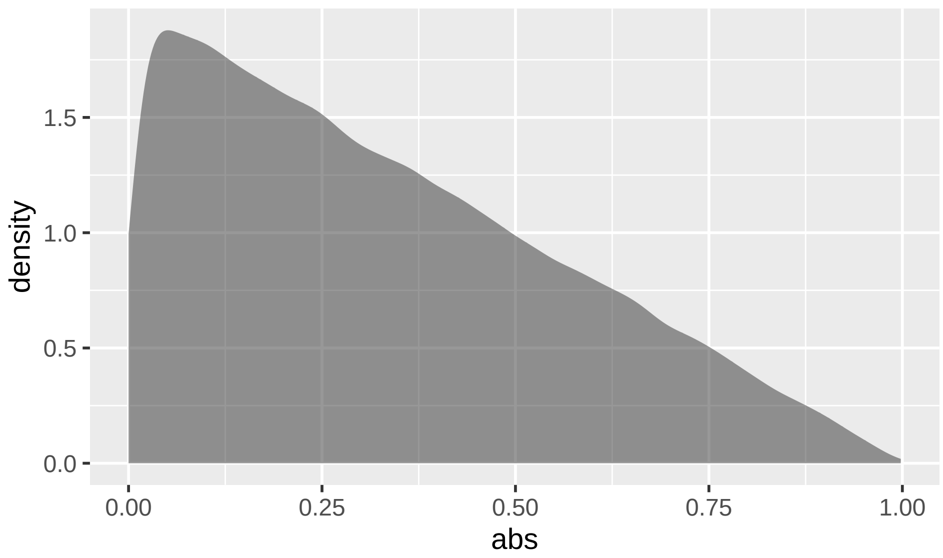
```
set.seed(354)
results <- do(100000)*abs(diff(runif(2,5,6)))
```

```
head(results)
```

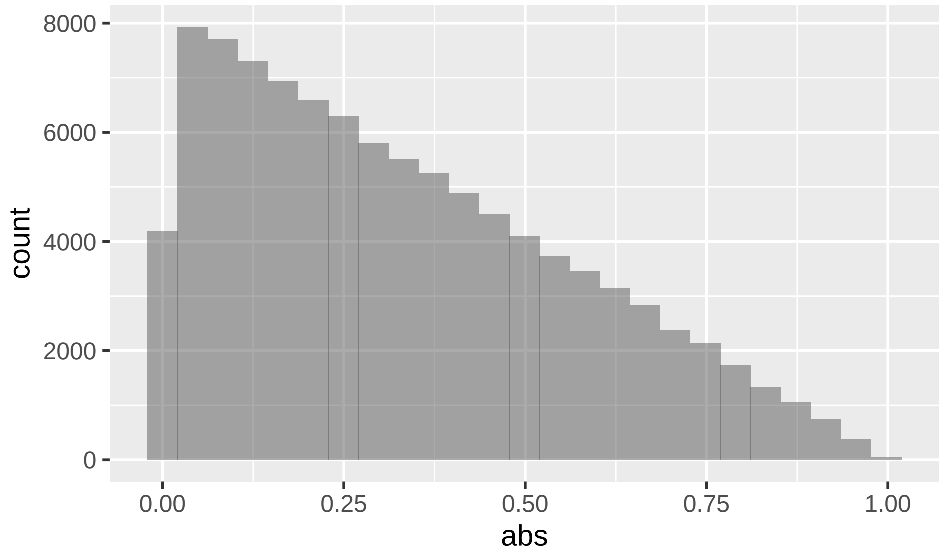
```
##          abs
## 1 0.03171229
## 2 0.77846706
## 3 0.29111599
## 4 0.06700434
## 5 0.08663187
## 6 0.40622840
```

Now, plot the estimated distribution.

```
results %>%
  gf_density(~abs)
```



```r
results %>%
  gf_histogram(~abs)
```



```r
inspect(results)
```

```
##
## quantitative variables:
##      name   class          min       Q1    median        Q3       max     mean
## ...1  abs numeric 1.265667e-06 0.133499 0.2916012 0.4990543 0.9979459 0.332799
##              sd      n missing
## ...1 0.2358863 100000       0
```

Exercise:
Now suppose whomever arrives first will only wait 5 minutes and then leave. What is the probability you eat together?

```
data.frame(results) %>%
  summarise(mean(abs<=5/60))
```

```
##   mean(abs <= 5/60)
## 1           0.15966
```

> **Exercise**:
> How long should the first person wait so that there is at least a 50% probability of you eating together?

Let's write a function to find the cdf.

```
z_cdf <- function(x) {
  mean(results$abs<=x)
}
```

```
z_cdf<- Vectorize(z_cdf)
```

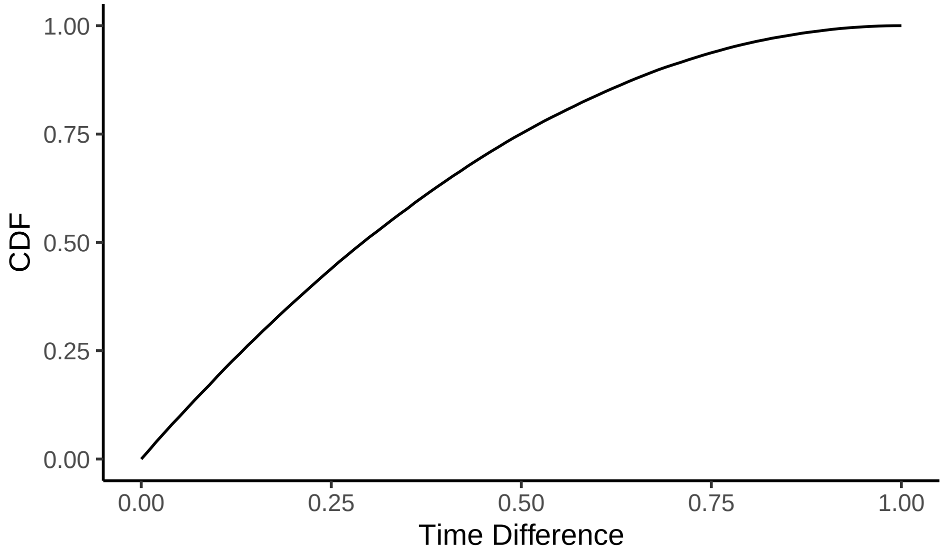
Now test for 5 minutes to make sure our function is correct since we determined above that this value should be 0.15966.

```
z_cdf(5/60)
```

```
## [1] 0.15966
```

Let's plot to see what the cdg looks like.

```
gf_line(z_cdf(seq(0,1,.01))~seq(0,1,.01),xlab="Time Difference",ylab="CDF") %>%
  gf_theme(theme_classic())
```



It looks like some where around 15 minutes, a quarter of an hour. But we will find a better answer by finding the root. In the code that follows we want to find where the cdf equals 0.5. The function `uniroot()` solves the given equations for roots so we want to put in the cdf minus 0.5. In other words, `uniroot()` want to solve $f(x) = 0$ for x.

```
uniroot(function(x)z_cdf(x)-.5,c(.25,35))$root
```

```
## [1] 0.2916077
```

So it is actually 0.292 hours, 17.5 minutes. So round up and wait 18 minutes.

File Creation Information

- File creation date: 2020-10-08
- Windows version: Windows 10 x64 (build 18362)
- R version 3.6.3 (2020-02-29)
- `mosaic` package version: 1.7.0
- `tidyverse` package version: 1.3.0