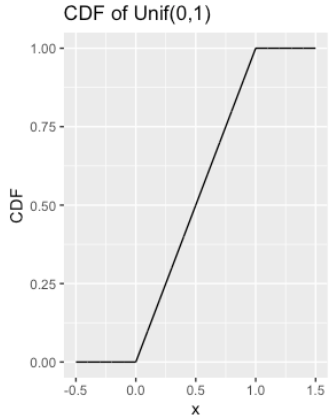
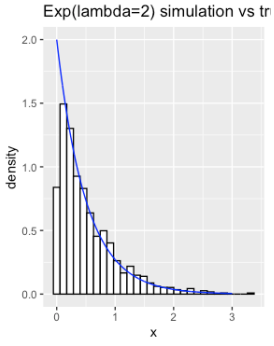
Jimin Lee jxl529

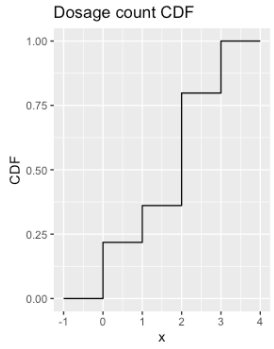
```
> punif(c(-2, 0, 0.27, 0.64, 1, 2), min=0, max=1)
[1] 0.00 0.00 0.27 0.64 1.00 1.00
> dunif(c(-2, 0, 0.27, 0.64, 1, 2), min=0, max=1)
[1] 0 1 1 1 1 0
> qunif(c(-2, 0, 0.27, 0.64, 1, 2), min=0, max=1)
[1]  NaN 0.00 0.27 0.64 1.00  NaN
Warning message:
In qunif(c(-2, 0, 0.27, 0.64, 1, 2), min = 0, max = 1) : NaNs produced
> library(ggplot2)
> # plot CDF of Unif(0, 1) distribution
> xset <- seq(-.5, 1.5, .001)
> udat <- data.frame(x=xset,y=punif(xset, min=0, max=1))
> ggplot(data=udat,mapping=aes(x=x, y=y)) +geom_line() + ylab("CDF") + ggtitle("CDF of
Unif(0,1)")
```




```
> # set PRNG seed
> set.seed(123)
> # generate 3 pseudo-random numbers from Unif(0, 1)
> runif(3, min=0, max=1)
[1] 0.2875775 0.7883051 0.4089769
> set.seed(456)
> runif(3, min=0, max=1)
[1] 0.0895516 0.2105123 0.7329553
> # Step 1: generate uniform samples
> unif_samples <- runif(1000, min=0, max=1)
> # Step 2: apply inverse transformation
> exp_samples <- -log(1 - unif_samples) / 2
> # plot histogram of samples along with true density
> ggplot(data=data.frame(x=exp_samples)) +
+   # create histogram of samples
+   geom_histogram(mapping=aes(x=x, y=..density..),fill="white", color="black", bins=30) +
+   # add true density using'dexp'function
+   geom_line(data=data.frame(x=seq(0, 3, .001),y=dexp(seq(0, 3, .001),
                                  rate=2)),mapping=aes(x=x, y=y),color="blue") +ggtitle("Exp(lambda=2)
                                   simulation vs truth")
```



```
> dosage_probs = c(.218, .143, .437, .202)
> cumsum(dosage_probs)
[1] 0.218 0.361 0.798 1.000
> # plot the true CDF
> ggplot(data=data.frame(x=c(-1, 0, 1, 2, 3, 4),y=cumsum(c(0, dosage_probs,
0))),mapping=aes(x=x,y=y)) +geom_step(direction="hv") +ggtitle("Dosage count CDF")
+ylab(“CDF")
```




```
> dosage_cdf = function(x) {
+   #'CDF for dosage count distribution
+   #'@param x input value
+   if (x < 0) {
+     0
+     } else if (x < 1) {
+       .218
+       } else if (x < 2) {
+         .361
+         } else if (x < 3) {
+           .798
+           } else {
+             1
+             }
+           }
> dosage_inv_cdf = function(v) {
+   #'Inverse CDF for dosage count distribution
+   #'@param v input value
+   if (v < .218) {
+   0} else if (v < .361) {
+     1} else if (v < .798) {
+       2} else {
+         3}}
> dosage_inv_cdf_alt = function(v) {
+   #'Inverse CDF for dosage count distribution without cases
+   #'@param v input value
+   which(v < cumsum(dosage_probs))[1] - 1}
> # step 1: generate uniform samples
> dosage_unif_sams = runif(1000, min=0, max=1)
> # step 2: apply inverse transformation
> dosage_dist_sams = sapply(dosage_unif_sams,dosage_inv_cdf)
> # summarize results in a table
> table(dosage_dist_sams)/1000
dosage_dist_sams
    0     1     2     3
0.224 0.124 0.446 0.206
```