DES – An Inventory Model

Yuxin Kang

```r
set.seed(56)

orderCost <- function(y) y

G <- function() {

  ## Generate Binomial(10, 0.8)
  U <-runif(1)
  c <- 0.8/0.2
  i <- 0
  pr <-(0.2)^10
  F. <- pr
  while (U >= F.) {
    pr <- ((c * (10 - i)) / (i + 1)) * pr
    F. <- F. + pr
    i <- i + 1
  }
  i
}

updateCustomerArr <- function(r, h, lambda, L, s, S) {
  H <<- H + h*SS[1]*(EL[1] - cur.time)
  cur.time <<- EL[1]

  D <- G() + 1
  w <- min(D, SS[1])
  if(w < D) {
    ## update unmet demand here
    lost <<- lost + (D - w)

  }

  SS[1] <<- SS[1] - w
  R <<- R + w * r

  if(SS[1] < s & SS[2] == 0) {
    SS[2] <<- S - SS[1]
    EL[2] <<- cur.time  + L
  }

  EL[1] <<- cur.time - (1 / lambda) * log(runif(1))
}

updateOrderArr <- function(r, h, lambda, s, S) {
```

```
  H <<- H + h*SS[1]*(EL[2] - cur.time)
  cur.time <<- EL[2]
  Cost <<- Cost + orderCost(SS[2])
  SS[1] <<- SS[1] + SS[2]
  SS[2] <<- 0
  EL[2] <<- Inf
}

generateOneRun <- function(r, h, lambda, L, s, S, T) {
  cur.time <<- 0
  SS <<- c(S, 0)                # c(x,y)
  EL <<- c(0, Inf)               # c(t_0, t_1)
  EL[1] <<- EL[1] - (1 / lambda) * log(runif(1))
  R <<- 0; H <<- 0; Cost <<- 0
  lost <<- 0                     # unmet demand
  W <<- 0 #total number of unmet orders

  while(min(EL) <= T) {
    if(EL[1] < EL[2]) {
      updateCustomerArr(r, h, lambda, L, s, S)
    } else {
      updateOrderArr(r, h, lambda, s, S)
    }
  }

  H <<- H + h*SS[1]*(T - cur.time)

  profit <- R - H - Cost
  c(profit,lost)
}
```

Run 100 simulations with s = 75 and S = 100.

Let V represent the final profit, and W represent the total number of unmet orders.

```
output <- sapply(1:100, function(x) generateOneRun(2,0.25,2.5,3,75,100,100))

meanUpdate <- function(X.new, Xbar.j, j) {
  Xbar.j + (X.new - Xbar.j)/(j+1)
}
varUpdate <- function(Xbar.j, Xbar.j1, j, s2.j) {
  (1- 1/j)*s2.j + (j+1)*(Xbar.j1 - Xbar.j)^2
}
Xbar.old <- mean(output[1,])
Xbar.old2 <- mean(output[2,])
S2 <- var(output[1,])
n <- 100
while (2 * qnorm(0.975) * sqrt(S2/n) >= 5){
  update <-generateOneRun(2,0.25,2.5,3,75,100,100)
  Xbar.new <- meanUpdate(update[1], Xbar.old, n)
  Xbar.new2 <- meanUpdate(update[2], Xbar.old2, n)
  S2 <- varUpdate(Xbar.j = Xbar.old, Xbar.j1 = Xbar.new, j = n, s2.j=S2)
  Xbar.old <- Xbar.new
```

```
  Xbar.old2 <- Xbar.new2
  n <- n + 1
}
n; Xbar.new; Xbar.new2; sqrt(S2/n)
```

```
## [1] 5841
```

```
## [1] 1003.948
```

```
## [1] 679.6787
```

```
## [1] 1.275385
```

```
upper <- Xbar.new + qnorm(0.975) * sqrt(S2/n)
lower <- Xbar.new - qnorm(0.975) * sqrt(S2/n)
cat("The 95% confidence interval for E(V) is (",lower, ",",upper,").")
```

```
## The 95% confidence interval for E(V) is ( 1001.448 , 1006.447 ).
```

```
cat("The final point estimate for E(V) is",Xbar.new)
```

```
## The final point estimate for E(V) is 1003.948
```

```
cat("The final point estimate for E(W) is",Xbar.new2)
```

```
## The final point estimate for E(W) is 679.6787
```