Generate Poisson distribution by different ways

2022-08-01

In this work, we will generate numbers following Poisson distribution by two different methods. The first is the Inverse Transform Method, in this we need obtain the cumulative distribution of Poisson and so find the inverse of this function. Done this, we calculate the values of the f in the points generated by standard uniform distribution.

```
#function that returns the cumulative poisson
acumulada_poisson = function(t,lambda){

  #starts sum
  soma = 0

  #loop for sum
  for (i in 0:t){
    #increment sum
    soma <- soma + dpois(x = i,lambda = lambda)
  }

  #return accumulated sum
  return(soma)
}

#inverse of cumulative function
inversa_acumulada = function(x,lambda){

  #begins a vector that represents the domain
  lista_dominio <- vector("list", 1000)

  #loop for fill the list
  for (i in 0:1000){
    lista_dominio[i+1] <- acumulada_poisson(i,lambda)
  }

  #result of inverse
  retorno = 0

  #if is greater that the maximum number in domain return 1000
  if(x > lista_dominio[1000]){retorno <- 1000}


  #sweeps the domain
  for (i in 0:1000){

    #in the first time that x is greatest that a point in domain return this point
    if(x<lista_dominio[i+1]){
```

```
      #its the return
      retorno = i

      #break the for
      break
    }
  }

  #returns the point
  return(retorno)
}
```

Now, we will testing the function with parameter $\lambda = 3$, we hope that the most values been next of 3.

```
for (i in 0:10){
  print(inversa_acumulada(runif(1,0,1),3))
}
```

```
## [1] 4
## [1] 1
## [1] 2
## [1] 0
## [1] 3
## [1] 2
## [1] 1
## [1] 3
## [1] 3
## [1] 3
## [1] 4
```

Now, we will generate de Poisson distribution by Acceptance-Rejection Method, where a exponential distribution is used with an auxiliary. The mathematical development can be consulted in https://www.eg.bucknell.edu/~xmeng/Course/CS6337/Note/master/node59.html. Here, we just implement the algorithm provided in this link.

```
#function that generates Poisson by acceptance-rejection
estima_poisson = function(alpha){

  #initialize n,P
  P <- 1
  n <- 0

  #iterations until P < exp(-alpha)
  while (P >= exp(-alpha)){

    #increment P
    P <- P*runif(1, min=0,max=1)

    #increment n
    n <- n+1
  }

  #return the n
  return(n)
}
```

Now, we will testing the function with parameter $\lambda = 3$, we hope that the most values been next of 3.

```
for (i in 0:10){
  print(estima_poisson(3))
}
```

```
## [1] 3
## [1] 2
## [1] 6
## [1] 4
## [1] 2
## [1] 7
## [1] 2
## [1] 4
## [1] 5
## [1] 5
## [1] 6
```

The results came out as expected, so we can considered the implementation satisfactory.