Exe2

2024-04-16

Exercise 1 - Discrete random variable

- The probability distribution function of a discrete variable k is given by the *zero-truncated* Poisson distribution:
$$P(k) = \frac{\lambda^k e^{-\lambda}}{k!(1 - e^{-\lambda})}$$

1) Write the R functions for the probability density and cumulative distribution functions, using the R naming convention.

- Assuming $\lambda = 1.4$,

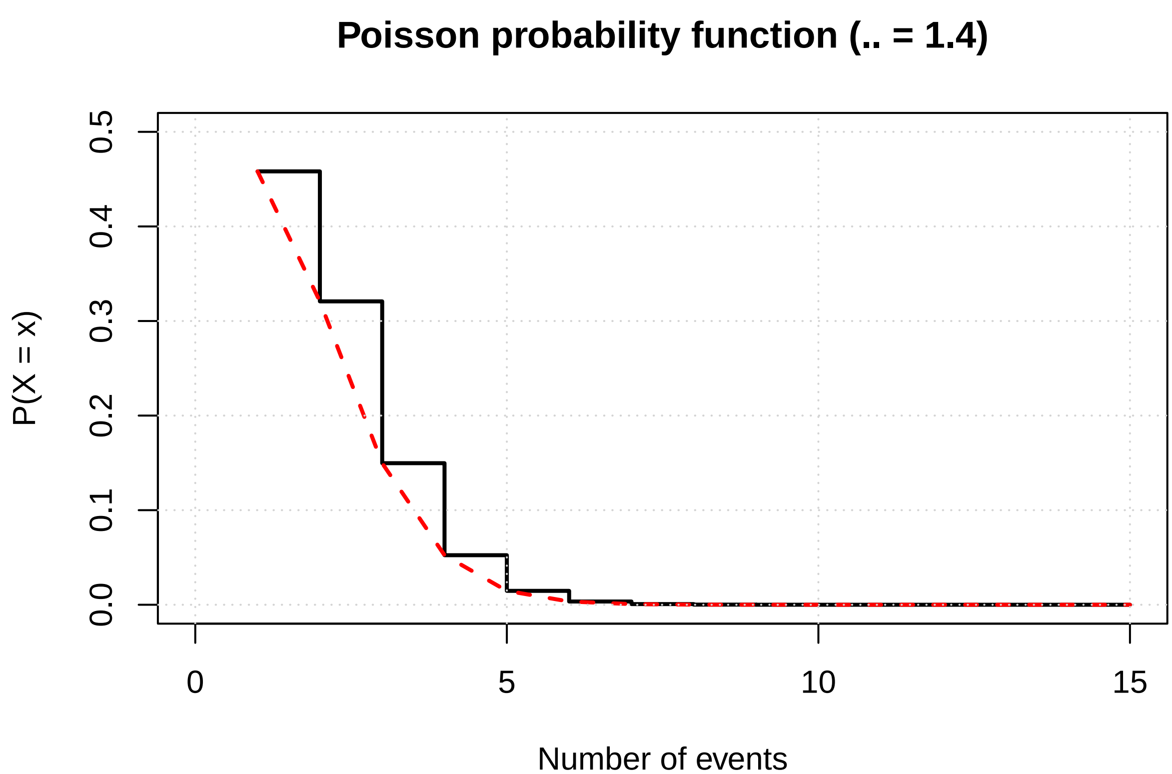
2) Produce two plots showing the pdf and cdf, separately.

```
## First let's define the zero truncated poisson function
## Remember that x is indeed a vector of n components

dpoisz <- function(x, lambda) {
    pmf <- (lambda^x * exp(-lambda) / (factorial(x) * (1 - exp(-lambda))))
  return(pmf)
}

## Let's now define the cumulative distribution function

ppoisz <- function(k, lambda) {

  cdf <- rep(0, length(k))

  for (i in 1:length(k)) {
    cdf[i] <- sum(dpoisz(1:k[i], lambda))

  }
  return(cdf)
}


N_samples <- 100

x <- 1:15
lambda <- 1.4


plot(dpoisz(x, lambda), type = 's', lwd = 2,
```

```r
     main = 'Poisson probability function (  = 1.4)',
     ylab = 'P(X = x)', xlab = 'Number of events',
     ylim = c(0, 0.5),
     xlim = c(0, 15)
     )
```

```
## Warning in title(...): conversione fallita da 'Poisson probability function (
## = 1.4)' in 'mbcsToSbcs': punto sostituito per <ce>

## Warning in title(...): conversione fallita da 'Poisson probability function (
## = 1.4)' in 'mbcsToSbcs': punto sostituito per <bb>
```

```r
lines(dpoisz(x, lambda), type = 'l', lty = 2, lwd = 2, col = 'red')
grid()
```



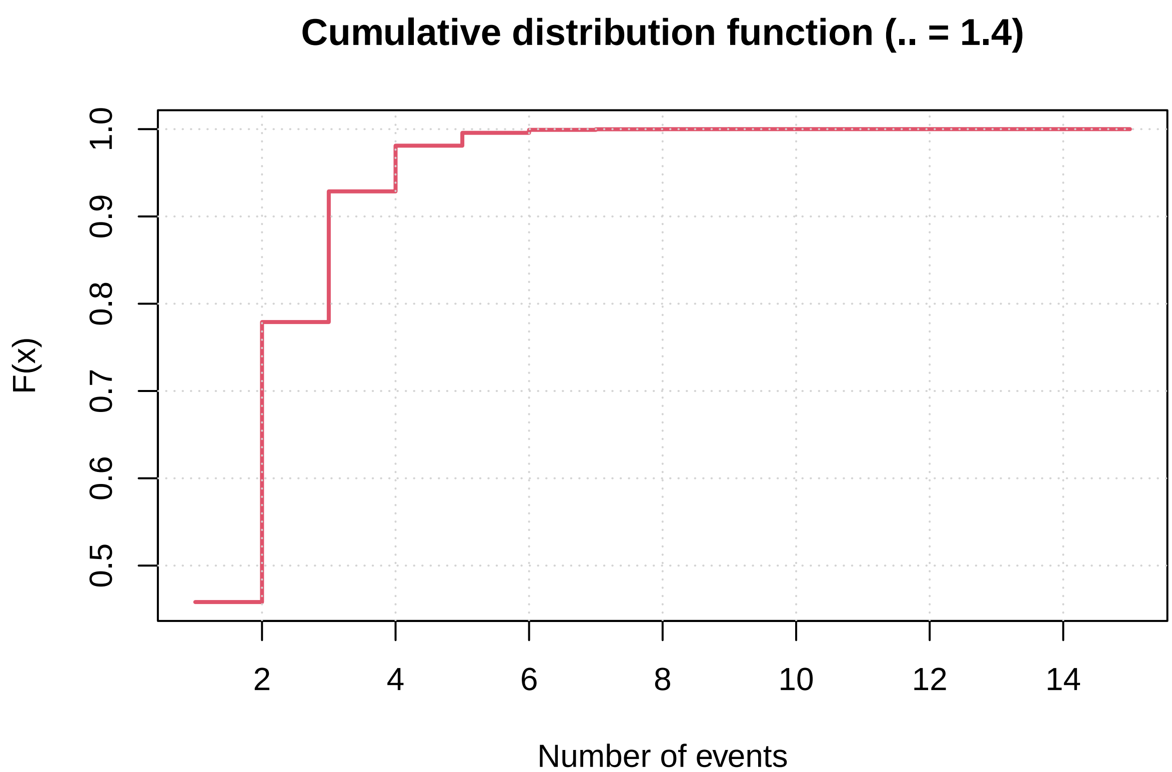

```r
#legend('topright', legend = '  = 1.4', col = 'black', lwd = 1, bty = 'n')
```

```r
#-----------
# lambda: 1.4
#-----------
lambda <- 1.4
k <- 1:15
```

```r
plot(ppoisz(k, lambda), type = "s", lwd = 2,
     main = "Cumulative distribution function (  = 1.4)",
     xlab = "Number of events", ylab = "F(x)", col = 2,
     ylim = c(min(ppoisz(k, lambda)), max(ppoisz(k, lambda)))
     )
```

```
## Warning in title(...): conversione fallita da 'Cumulative distribution function
## (  = 1.4)' in 'mbcsToSbcs': punto sostituito per <ce>

## Warning in title(...): conversione fallita da 'Cumulative distribution function
## (  = 1.4)' in 'mbcsToSbcs': punto sostituito per <bb>
```

```r
grid()
```



3) Compute the mean value and variance of the probability distribution using R.

```r
k <- 0:100
pdf <- dpoisz(k, lambda)
pdf.2 <- dpoisz(k^2, lambda)

mean_value <- sum(k * dpoisz(k, lambda))


print(sprintf('Mean: %.3f', mean_value))
```

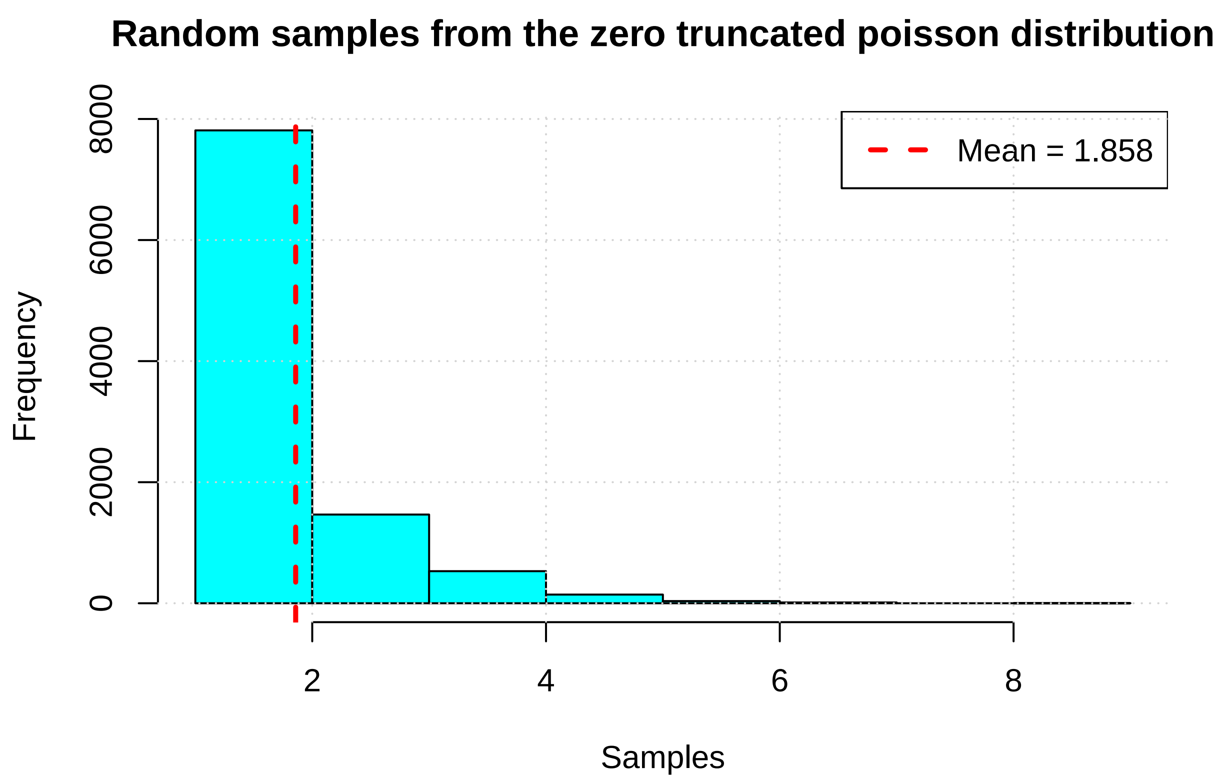
```
## [1] "Mean: 1.858"
```

```r
std_value <- sum(k^2 * dpoisz(k, lambda)) - mean_value^2


print(sprintf('Variance: %.3f', std_value))
```

```
## [1] "Variance: 1.007"
```

4) Generating a sample of random numbers from the given distribution

```r
k <- 1:1000
samples <- sample(k, size = 10000, replace = TRUE, prob = dpoisz(k, lambda))


hist(samples,
     breaks = seq(min(samples), max(samples), by = 1),
     main = 'Random samples from the zero truncated poisson distribution',
     xlab = 'Samples', ylab = 'Frequency',
     col = 'cyan')


abline(v = mean_value, col = 'red', lty = 2, lw = 2.5)
#lines(dpoisz(k, lambda), type = 'l', lty = 2, lwd = 2, col = 'black')
legend('topright', legend = sprintf('Mean = %.3f', mean_value), col = 'red', lty = 2, lw = 2.5)
grid()
```



Exercise 2 - Continuous random variable

- The energy distribution of CR muons at sea level can be approximated as follows:

$$\begin{cases} p(E) = N & for \;\; E < E_0 \\ p(E) = (E - E_0 + 1)^{-\gamma} & for \;\; E \; \geq \; E_0 \end{cases}$$

where $E_0 \;=\; 7.25 \; GeV$ and $\gamma \;=\; 2.7$.

a) Compute the normalization factor N using R.

b) Plot the probability density function in R.

c) Plot the cumulative density function in R.

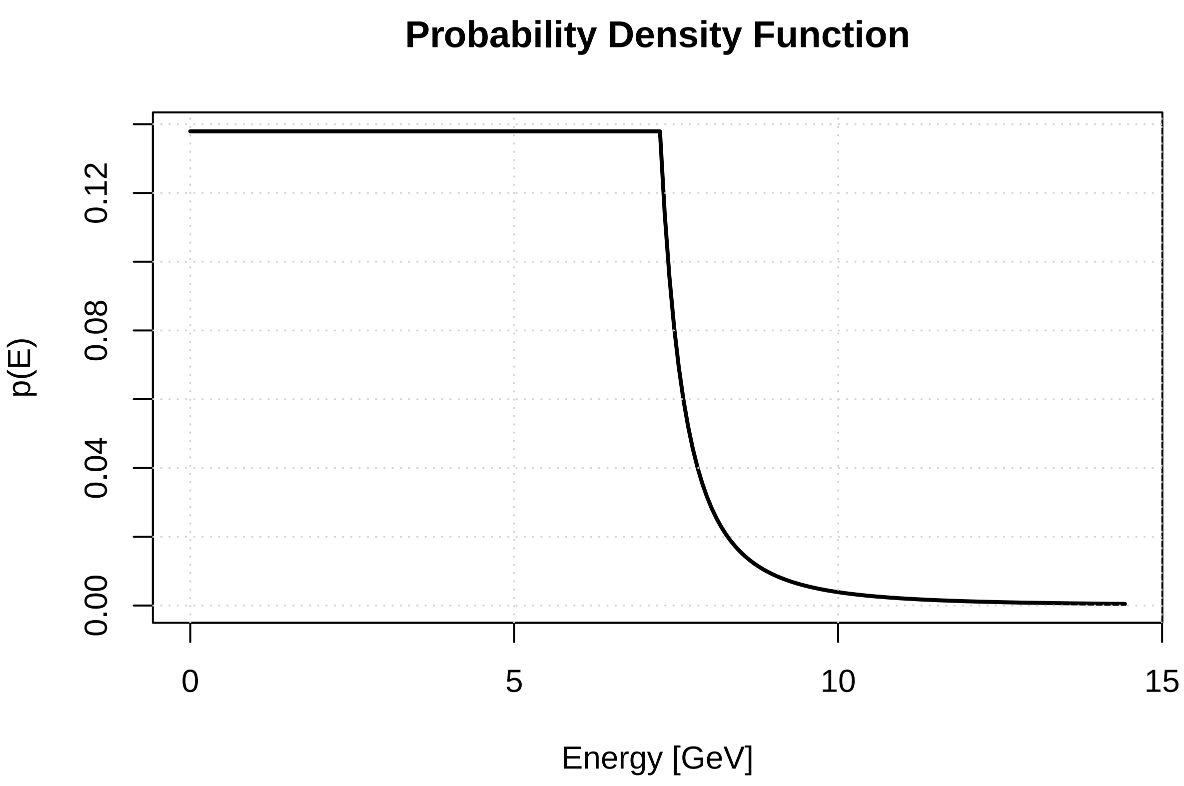
d) Compute the mean value using R.

e) [**Optional**] Generate 10^6 random numbers from this distribution, show them in an histogram and superimpose the pdf (with a line or with a sufficient number of points).

```
## R Markdown
# To compute the normalization factor we need the integral
# Let's first define the probability distribution

p <- function(E, N, E_0, gamma) {
  if (E < E_0) {
    return(N)
  } else {
    return(N * (E - E_0 + 1) ^ (-gamma))
  }
}

E_0 <- 7.25
gamma <- 2.7

n_steps <- 100
dE <- E_0 / n_steps

## First we need to compute the integral of the distribution
summ <- 0
for (i in n_steps) {
  summ <- summ + p(dE, N = 1, E_0, gamma)
}

N <- summ / E_0

print(sprintf('Normalization factor: %.3f ', N))
```

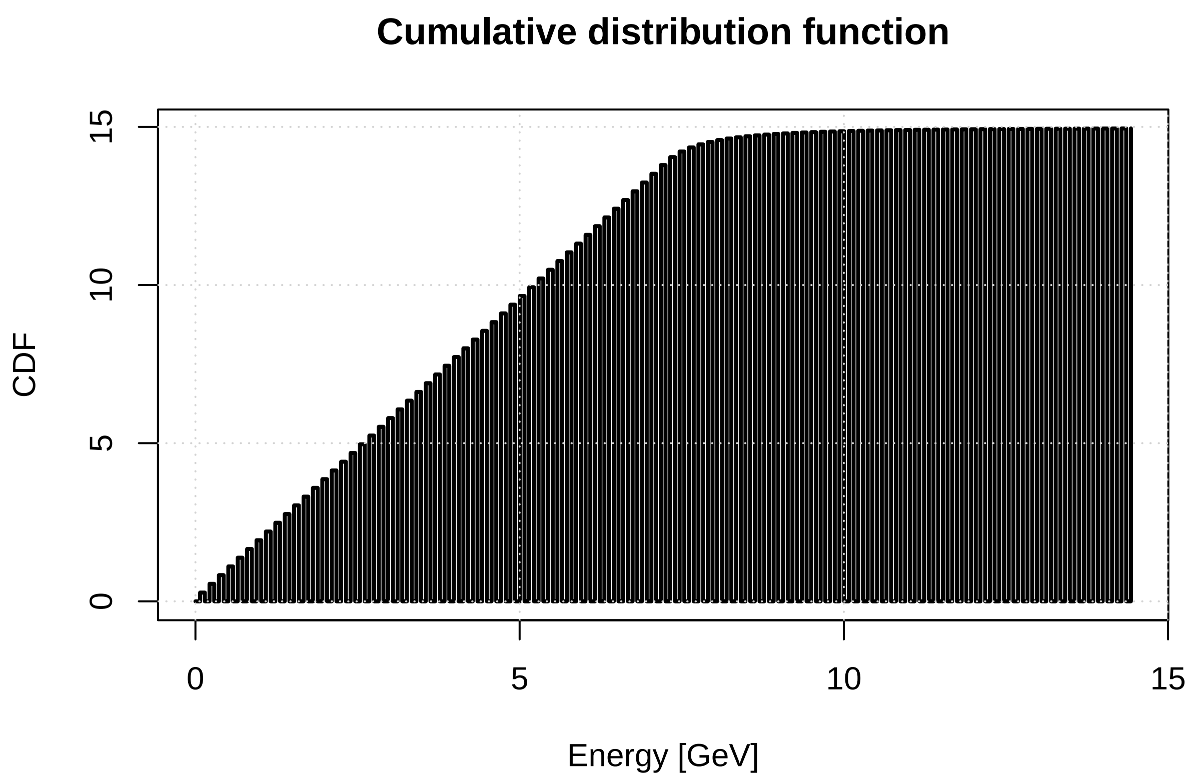
```
## [1] "Normalization factor: 0.138 "
```

```r
## Let's plot the probability density function in R
N_steps <- 100

E <- 0
dE <- E_0 / N_steps
E_stored <- numeric(N_steps * 2)
pdf <- numeric(N_steps * 2)

for (i in 1:(N_steps * 2)) {
  E_stored[i] <- E
  pdf[i] <-  p(E, N, E_0, gamma)
  E <- E + dE
}

plot(E_stored, pdf,
     xlab = 'Energy [GeV]',
     ylab = 'p(E)',
     type  ='l', lwd = 2,
     main = 'Probability Density Function')
grid()
```




```r
CDF <- numeric(N_steps * 2)

for (i in (1:N_steps * 2)) {
  CDF[i] <- sum(pdf[0:i])
```

```
}

plot(E_stored, CDF, type = 's',
     xlab = 'Energy [GeV]', ylab = 'CDF',
     main = 'Cumulative distribution function',
     col = 'black', lwd = 2)
grid()
```




```
## Compute the mean

E.x.1 <- integrate(function(E) {N * E}, lower = 0, upper = E_0)$value

E.x.2 <- integrate(function(E) {N * E * (E - E_0 + 1)^(-gamma)}, lower = E_0, upper = Inf)$value

integral <- E.x.1 + E.x.2
print(paste(sprintf('Mean of the distribution: %.3f', E.x.1 + E.x.2), 'GeV'))
```

```
## [1] "Mean of the distribution: 4.329 GeV"
```

Exercise 3 - Suppose that the average number of accidents at an intersection is two per day.

a) Using Markov's inequality, find a bound for the probability that at least five accidents will occur tomorrow.