Statistical Inference Project

Chris McKelt

Part 1: Simulation Exercise Instructions

In this project you will investigate the exponential distribution in R and compare it with the Central Limit Theorem.

The exponential distribution can be simulated in R with rexp(n, lambda) where lambda is the rate parameter.

The mean of exponential distribution is 1/lambda and the standard deviation is also 1/lambda. Set lambda = 0.2 for all of the simulations.

You will investigate the distribution of averages of 40 exponentials. Note that you will need to do a thousand simulations.

```
##    dplyr    tidyr  ggplot2    knitr markdown  moments  nortest    e1071
##     TRUE     TRUE     TRUE     TRUE     TRUE     TRUE     TRUE     TRUE
```

Part 1 - Simulations

simulation inputs

```
lambda <- 0.2
size <- 40
simulations <- 1000
```

settings a seed will allow us to reproduce the results

```
set.seed(881) # prime
```

create a dataframe with column title 'individual_mean' for recording each sample distributions mean

```
samples <- data.frame(individual_mean = numeric(size))
replicate(simulations, mean(rexp(size, lambda)))
```

```
##    [1] 4.477747 5.822793 4.054247 4.873564 4.609780 4.585438 4.651006
##    [8] 4.595969 3.712426 4.149294 5.157933 6.458950 3.078232 4.094112
##   [15] 5.986086 5.472690 5.419147 5.531831 4.518665 4.869121 4.133342
##   [22] 5.370928 6.321934 5.059804 5.418288 4.286439 4.486671 4.319625
##   [29] 4.219707 5.810918 5.205436 5.650520 4.473421 4.982367 5.520775
##   [36] 4.965440 6.454077 4.808679 4.626727 5.466139 4.920505 4.649314
```

```
##   [43] 4.160529 5.025663 5.858350 5.117681 3.781178 3.525809 4.439272
##   [50] 5.287117 4.048858 6.141934 4.395114 5.259441 4.761815 6.137541
##   [57] 5.071056 5.588132 4.553956 5.814644 5.581375 4.024610 5.823562
##   [64] 4.997891 5.942808 5.522696 4.459563 6.178316 6.842838 5.671803
##   [71] 4.469114 4.801853 4.665882 4.445754 5.185193 4.125845 4.727188
##   [78] 5.067062 4.473093 5.528742 5.744555 4.635694 6.904941 4.193948
##   [85] 5.632864 4.178830 5.915544 4.502363 3.669769 4.329402 4.869422
##   [92] 4.685059 3.636841 4.480857 5.263643 4.469195 6.226405 4.741717
##   [99] 3.654736 4.509778 5.263810 5.635516 5.218723 4.096542 4.541604
##  [106] 4.534021 5.623070 4.931131 5.739006 5.122181 6.069965 4.566364
##  [113] 5.362757 3.719220 4.740278 3.962560 5.738909 5.337926 4.209345
##  [120] 4.819896 6.107608 4.827245 4.517337 4.867988 5.181967 4.549030
##  [127] 4.591865 4.430591 4.256263 4.283010 4.926699 5.304738 5.358154
##  [134] 4.795842 6.429036 4.669127 5.379337 5.436265 6.186132 5.295183
##  [141] 4.221416 5.669997 4.058217 4.649527 3.738601 4.742113 3.576317
##  [148] 6.330689 5.466280 4.131557 4.104211 4.131771 5.498004 5.178683
##  [155] 4.646368 5.689019 4.459512 4.115204 5.638382 3.870438 5.749795
##  [162] 3.855602 5.025425 6.389348 6.262231 4.758908 5.770511 5.148280
##  [169] 6.373413 4.043356 4.776938 4.573203 6.905300 5.429451 4.062134
##  [176] 6.138414 4.325707 4.142002 5.511726 5.941754 4.837658 5.515795
##  [183] 6.372840 3.553169 5.071859 3.876947 5.907754 4.925903 4.269575
##  [190] 3.066773 5.880013 6.792044 5.073425 4.867108 4.181081 5.724922
##  [197] 4.465301 4.643285 4.829955 3.171293 4.363128 5.140164 5.040522
##  [204] 7.061140 4.951941 4.512984 5.306142 5.044555 5.293588 4.525819
##  [211] 5.265364 4.719418 4.608964 6.138074 4.805210 5.350031 5.130589
##  [218] 4.140409 6.404625 4.949603 4.472438 4.161453 5.882515 4.725728
##  [225] 5.480705 4.810632 3.470458 4.684096 4.993168 5.061233 4.288307
##  [232] 4.882722 3.208555 5.239177 4.294336 3.822187 6.072706 4.725571
##  [239] 3.757056 4.893307 3.513157 3.734274 5.302279 5.551670 4.788951
##  [246] 4.930225 4.788817 4.400297 5.856831 4.613834 4.341910 4.410191
##  [253] 5.182609 4.291738 6.046860 4.864585 3.968699 4.683065 4.387079
##  [260] 6.429382 4.680842 4.588689 4.016393 4.720207 5.276618 5.784195
##  [267] 3.038445 4.637327 6.522068 5.175961 5.275567 4.987059 4.773204
##  [274] 6.263420 5.759631 5.794074 3.613251 5.450131 5.604221 4.221041
##  [281] 5.011036 4.836544 5.912570 3.899990 4.038655 3.702410 4.195832
##  [288] 4.932583 4.634887 3.991942 5.263496 5.797995 4.693850 4.254799
##  [295] 4.712065 3.533879 6.649724 3.952595 4.980208 4.994566 4.667310
##  [302] 3.633684 3.972221 6.197083 4.298184 4.741036 4.601320 4.813853
##  [309] 6.168023 4.754357 4.535436 5.064495 5.523006 5.361631 5.326654
##  [316] 6.220913 4.829723 5.004350 4.539446 4.824659 5.157739 4.860073
##  [323] 6.363575 5.043780 4.857918 5.169988 5.099879 5.616262 3.668756
##  [330] 5.595927 6.176703 5.248422 4.904656 6.116093 4.426638 4.698453
##  [337] 4.983272 5.714761 4.780819 5.111644 3.508864 4.943474 4.989637
##  [344] 4.882901 5.233404 5.894025 6.696226 3.763661 5.691752 4.360740
##  [351] 5.182130 4.464654 5.047331 5.362107 5.792441 4.938366 4.352097
##  [358] 4.985860 5.388543 4.159870 3.453201 4.432070 4.174557 3.694752
##  [365] 6.241594 5.100268 5.763008 4.397284 4.673909 6.084358 3.768166
##  [372] 3.562913 4.658418 5.364576 5.999158 5.426150 5.518400 5.127671
##  [379] 5.240800 5.871354 5.334936 5.242126 4.865186 6.206548 4.317952
##  [386] 4.901590 4.520377 5.044043 4.589646 4.706063 4.990059 4.956417
##  [393] 5.788356 4.489123 5.161494 3.893473 4.734369 5.170372 5.101158
##  [400] 5.496167 4.563370 5.000997 4.966623 5.227250 6.115488 5.362927
##  [407] 4.613273 5.511576 4.715607 5.380889 4.456061 4.502880 5.496374
##  [414] 4.798677 4.842253 4.455561 5.442711 5.728878 3.968698 4.836183
```

```
##  [421] 6.374506 6.092238 3.725423 4.749322 5.881284 4.645986 3.440139
##  [428] 5.527435 3.964469 4.853024 5.651641 6.148513 4.679216 5.175982
##  [435] 3.824224 4.242490 4.277092 4.561974 5.392557 6.083951 4.701939
##  [442] 5.501492 5.224103 5.061754 3.892543 3.520397 5.679169 4.991979
##  [449] 5.061649 3.554232 5.897533 6.061054 6.292199 4.897514 3.951472
##  [456] 5.072400 4.934352 4.739600 5.624493 4.984889 6.736274 5.320495
##  [463] 5.101484 4.798725 3.464178 6.136682 4.642252 6.309471 5.294252
##  [470] 4.251434 4.738161 4.411612 4.299202 4.658648 3.883301 3.554120
##  [477] 5.458336 3.921185 5.707558 4.516157 5.891252 4.747727 4.898291
##  [484] 5.572317 5.791519 5.239739 4.796976 4.311430 3.821789 4.138058
##  [491] 6.541622 4.283524 4.848950 4.794439 5.917761 4.202179 4.168637
##  [498] 4.890398 4.619871 6.019120 4.905601 4.746729 4.869200 4.527614
##  [505] 5.539640 4.065573 4.887316 3.752372 6.071505 6.061408 6.854201
##  [512] 6.326465 4.653098 4.236720 4.283393 5.321177 5.068602 5.138423
##  [519] 3.590435 7.504604 6.465339 4.414801 6.564753 3.893277 4.578836
##  [526] 5.242132 5.611620 3.354495 6.136239 5.382540 5.578820 5.493164
##  [533] 5.740477 6.504287 4.580708 3.873859 6.435901 3.954076 3.846098
##  [540] 4.276177 3.345978 5.092901 5.185938 4.490627 5.407109 7.011434
##  [547] 5.202242 4.454697 4.892080 4.367424 4.492683 5.212965 6.554220
##  [554] 4.577367 4.743780 5.080222 5.211727 5.444413 4.291443 4.853056
##  [561] 4.630321 5.539254 5.236185 4.828844 4.618248 5.530703 5.957565
##  [568] 5.578294 6.015347 4.537794 4.298652 5.580827 4.835162 5.759225
##  [575] 5.168847 5.266744 5.324653 4.387931 5.163712 4.125640 4.477525
##  [582] 3.747911 5.143150 5.003039 4.932372 6.094689 4.623277 5.807622
##  [589] 6.203238 5.869480 5.591491 4.662491 5.197290 5.043625 5.656121
##  [596] 4.182846 5.409931 5.072883 5.084881 6.159595 4.767353 5.732997
##  [603] 5.775053 4.286567 3.657520 5.405702 5.121611 5.069974 4.513671
##  [610] 6.936890 5.235590 5.348315 4.653865 5.614751 4.444038 5.532906
##  [617] 5.927265 5.429617 4.051481 3.806394 6.162959 4.866095 6.605226
##  [624] 4.981015 4.264633 5.318596 5.668851 4.463554 4.621976 4.153423
##  [631] 4.818440 4.476730 4.750898 6.577281 4.593291 3.395923 4.345696
##  [638] 4.887947 7.158041 4.827355 4.878246 5.996550 4.384831 3.728125
##  [645] 4.491725 5.735386 3.959715 6.161134 4.908355 3.883489 4.780960
##  [652] 3.694795 4.192615 5.437152 4.312696 4.273212 4.926011 5.762921
##  [659] 3.749239 5.171213 4.194477 4.583727 4.658810 5.572814 6.822411
##  [666] 5.638156 4.477625 4.506593 5.741876 4.510563 3.605504 4.644458
##  [673] 6.862343 6.652800 6.057681 4.450379 5.609990 5.423132 5.539745
##  [680] 4.610807 4.745065 4.912782 5.103121 4.340517 4.690703 4.994462
##  [687] 4.515350 4.638734 3.923666 3.711721 5.971237 5.896926 4.796163
##  [694] 4.543238 5.236295 5.005100 3.785963 3.969587 4.645283 5.037105
##  [701] 5.284682 3.783895 4.820856 5.497989 4.396230 6.406011 5.202607
##  [708] 4.976005 5.442225 5.509046 6.092741 6.031377 4.055405 5.923368
##  [715] 4.939050 4.023327 4.463944 4.855765 5.290758 4.504613 5.503977
##  [722] 6.316373 4.383938 5.377641 4.558542 4.904459 5.070459 4.366099
##  [729] 6.812654 5.670670 4.081834 4.156038 5.057307 4.068375 5.607176
##  [736] 5.682891 4.779732 4.852555 4.201290 5.214789 5.134449 4.901751
##  [743] 2.954696 4.519285 4.440303 5.847399 3.564119 5.014556 5.657918
##  [750] 5.106643 5.757282 4.518732 4.054912 5.902931 4.292346 3.632577
##  [757] 3.242516 4.105855 5.308046 5.001726 4.506080 5.118768 4.735058
##  [764] 5.349266 6.957045 5.935728 4.668147 3.895769 4.535237 5.102497
##  [771] 4.310592 3.938283 4.126013 3.852711 4.138734 5.606984 6.035594
##  [778] 3.567138 4.141862 4.821682 4.868641 5.307269 5.379678 4.882561
##  [785] 4.852300 4.710748 4.722812 4.216816 4.750527 6.061939 5.253589
##  [792] 5.044351 5.231250 4.752823 5.489209 4.942084 4.421773 5.349520
```

```
##  [799] 4.224268 5.360239 4.449251 4.884072 6.039860 4.084703 4.990998
##  [806] 3.146932 4.528028 4.432593 4.442897 6.174298 7.002486 5.508463
##  [813] 5.125758 4.677978 5.269065 6.025973 4.676071 6.506003 4.793703
##  [820] 5.375697 5.935360 4.409179 5.907521 5.025884 5.030626 4.669713
##  [827] 4.423117 5.238939 5.203819 6.012558 4.989242 5.899154 3.947181
##  [834] 4.977896 5.039147 4.847321 4.989667 5.841969 4.768444 4.604721
##  [841] 4.238668 3.417163 4.324409 3.604881 6.009194 5.156613 4.990983
##  [848] 6.725242 4.984345 4.951376 5.255504 4.296999 5.880931 4.063275
##  [855] 4.871584 5.262350 3.948572 4.214408 4.637993 5.117635 4.221434
##  [862] 5.444820 4.961525 5.751924 4.622110 6.191520 4.780517 4.822940
##  [869] 5.160174 6.296330 5.271327 3.306434 4.809731 5.253143 4.395803
##  [876] 3.868722 5.960459 4.070624 4.432135 6.238343 4.694527 4.774069
##  [883] 4.540402 3.858785 6.075365 5.345174 3.728951 5.435282 6.324202
##  [890] 4.010521 4.486403 4.565843 3.613974 6.790007 5.449002 4.358349
##  [897] 4.265322 3.934556 5.800707 4.472908 4.003084 6.659780 3.558281
##  [904] 4.735158 5.368375 3.848311 4.631142 6.201723 3.865159 4.727346
##  [911] 5.392299 4.604678 4.941461 5.356088 4.463160 5.447044 4.180540
##  [918] 5.730145 5.063693 4.701625 4.413578 5.359979 5.372060 4.651471
##  [925] 4.457331 3.776663 5.072917 3.698343 6.389875 3.550352 3.841411
##  [932] 4.475090 4.395080 3.964174 4.955963 4.066199 4.551732 3.380941
##  [939] 5.160395 4.324008 5.179249 3.652153 4.516456 5.977623 4.637730
##  [946] 5.270218 5.866953 4.916347 4.763201 4.886284 3.744452 4.756629
##  [953] 4.302548 5.346065 5.164238 4.716826 4.433024 4.742462 4.667845
##  [960] 6.417788 5.858835 4.246867 5.975394 4.746015 5.180167 4.086013
##  [967] 5.157983 5.079510 4.255219 4.710702 6.547870 5.525139 7.537365
##  [974] 5.092021 6.853127 5.201422 4.767741 3.870819 4.126253 4.784860
##  [981] 3.856164 6.803727 6.546472 5.605184 6.854520 6.595731 4.414702
##  [988] 4.153311 5.463051 4.459573 4.530181 6.155396 6.247589 4.499640
##  [995] 4.574150 5.863842 5.029003 4.885902 6.010222 4.312497
```

```
# create a new sample for each simulation and get its mean and added to samples data frame
# iteration over vectorisation
for (i in 1:simulations) {
    individual_sample <- rexp(size, lambda) # specific set simulation with 40 exponentials of lambda
    samples[i, 1] <- mean(individual_sample)
}
```

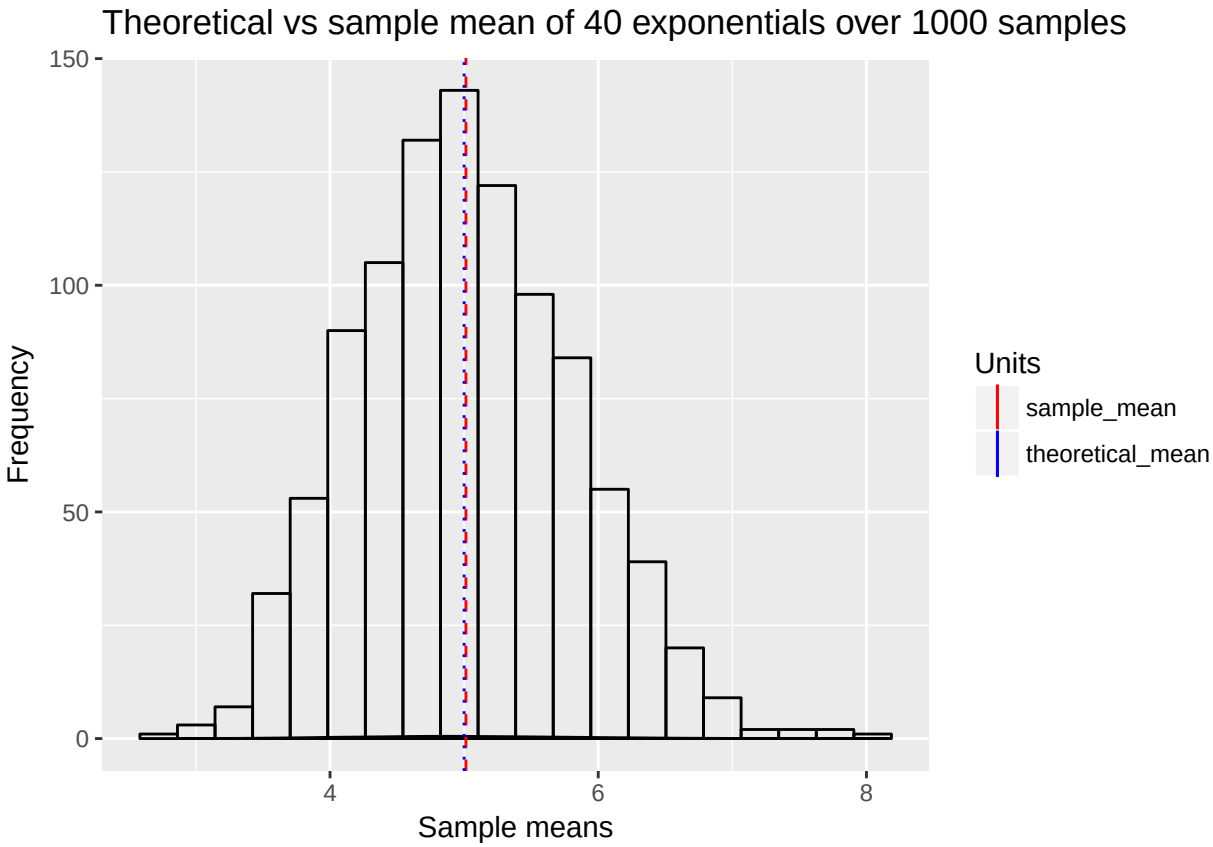
show the sample mean and compare it to the theoretical mean of the distribution.

```
    theoretical_mean <- 1 / lambda
    data <- samples[, 1]
    sample_mean <- mean(data)
```

plot samples with theoretical mean vs sample mean

```
ggplot(samples, aes(x = individual_mean)) +
    geom_histogram(bins = 20, boundary = -0.5, fill = NA, color = "black") +
    geom_density(alpha = .2, fill = "#FF6666", show.legend = FALSE) +
    geom_vline(aes(xintercept = sample_mean, color = "sample_mean", linetype = "sample_mean", show.lege
    geom_vline(aes(xintercept = theoretical_mean, color = "theoretical_mean", linetype = "theoretical_m
    scale_colour_manual(name = "Units", values = c(sample_mean = "red", theoretical_mean = "blue")) +
```

```
    scale_linetype_manual(name = "Units", values = c(sample_mean = "dashed", theoretical_mean = "dotted
    labs(title = "Theoretical vs sample mean of 40 exponentials over 1000 samples") +
    labs(x = "Sample means", y = "Frequency")
```



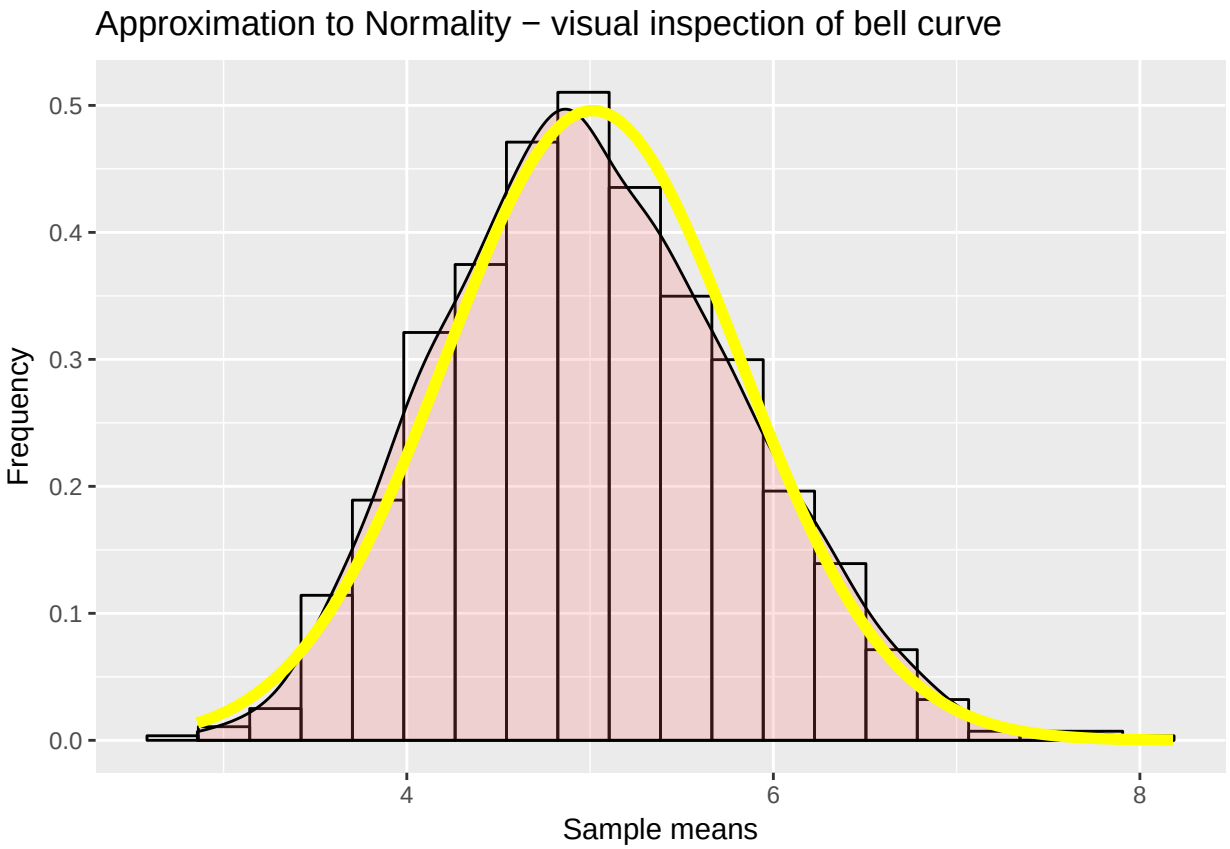
show how variable the sample is (via variance) and compare it to the theoretical variance of the distribution.

```
theoretical_sd <- (1 / lambda) / sqrt(size)
theoretical_variance <- theoretical_sd ^ 2
sample_variance <- var(data)
```

show that the distribution is approximately normal.

visually inspect bell curve

```
ggplot(samples, aes(x = individual_mean)) +
    geom_histogram(aes(y = ..density..), bins = 20, boundary = -0.5, fill = NA, color = "black") +
    geom_density(alpha = .2, fill = "#FF6666", show.legend = FALSE) +
    stat_function(fun = dnorm, args = list(mean = mean(data), sd = sqrt(sample_variance)), colour = "ye
    labs(title = "Approximation to Normality - visual inspection of bell curve") +
    labs(x = "Sample means", y = "Frequency")
```



Testing for normality

https://en.wikipedia.org/wiki/Normal_probability_plot

nortest package to the rescue

http://stats.stackexchange.com/questions/52293/r-qqplot-how-to-see-whether-data-are-normally-distribute 52295

Test 1 - skewness and kurtosis, they should be around (0,3)

```
skewness(data)
```

```
## [1] 0.2887447
```

```
kurtosis(data)
```

```
## [1] -0.0225037
```

Test 2 - Shapiro-Wilks test

```
shapiro.test(data)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data
## W = 0.99399, p-value = 0.0004827
```

Test 3 - Kolmogorov-Smirnov test

```
ks.test(data, "pnorm", mean(data), sqrt(var(data)))
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  data
## D = 0.033966, p-value = 0.1988
## alternative hypothesis: two-sided
```

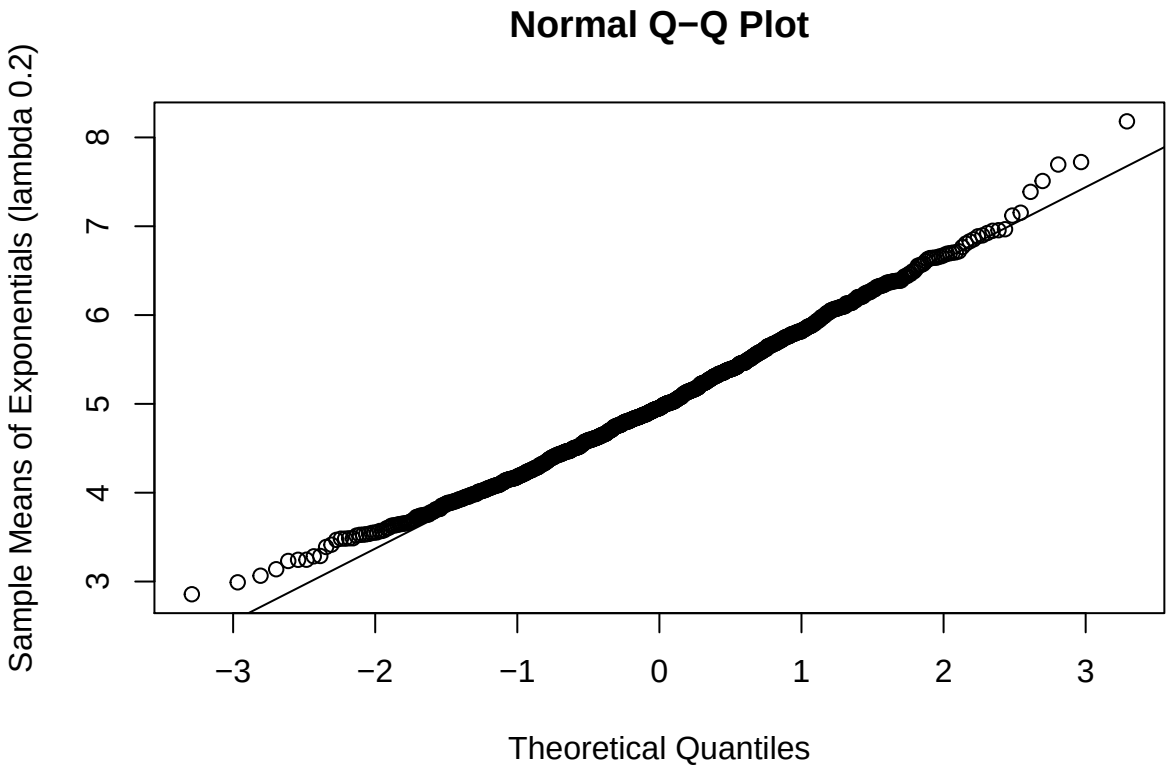
Test 4 - Anderson-Darling test

```
ad.test(data)
```

```
##
##  Anderson-Darling normality test
##
## data:  data
## A = 1.2078, p-value = 0.003775
```

Test 5 - qq-plot: you should observe a good fit of the straight line

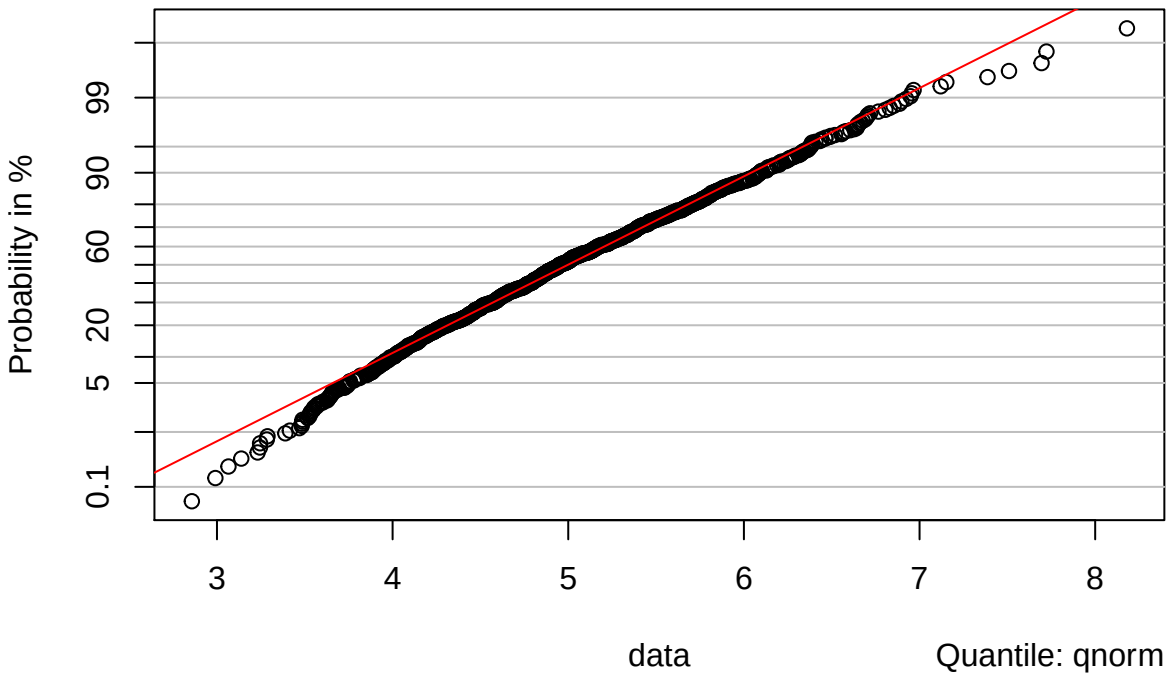
```
qqnorm(data, ylab = "Sample Means of Exponentials (lambda 0.2)")
qqline(data)
```



Observation: if the data follows the line the distribution is normal (as it does)

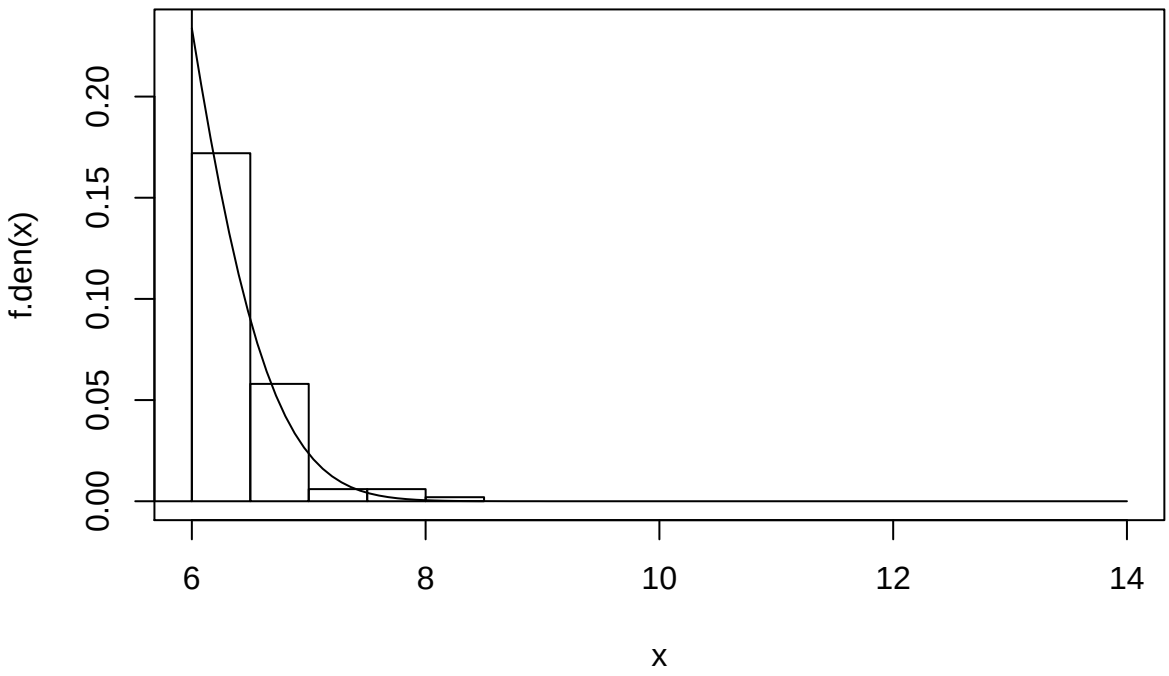
Test 6 - p-plot: you should observe a good fit of the straight line

```
probplot(data, qdist = qnorm)
```



Test 7 - fitted normal density

```
f.den <- function(t) dnorm(t, mean(data), sqrt(var(data)))
curve(f.den, xlim = c(6, 14))
hist(data, prob = T, add = T)
```



Conclusion: the data is normally distributed having passed 7 normality tests