Distribuição Normal

Aula 2

Prof. André Luiz Cunha

21/05/2021

1 Transformação de escala

Um passo importante de qualquer análise de dados é a uniformização do intervalo de dados, de modo que todas as variáveis do banco de dados tenham o mesmo intervalo de variação. Na literatura temos dois tipos de transformação

```r
# Conjunto de valores aleatórios com média 50 e desvio 15.
(x <- rnorm(100, 50, 15))
```

```
##   [1] 81.95492 63.70145 59.84402 29.03614 36.83837 52.31954 47.48039 16.96690
##   [9] 79.02751 42.22281 49.93790 72.16353 32.75996 62.75825 62.04287 34.34612
##  [17] 56.74150 22.51613 38.40618 41.54669 46.13155 35.90177 62.68782 56.70538
##  [25] 64.97357 47.59002 51.65478 63.96517 39.68162 31.25627 88.35457 19.24039
##  [33] 58.89939 61.81595 34.87505 68.15630 34.21176 47.18986 63.18132 36.26722
##  [41] 47.71326 56.51160 66.98419 49.05309 64.77358 45.82890 62.42944 43.79950
##  [49] 26.60777 54.16811 76.10061 44.60954 60.52618 15.77870 65.35911 96.65488
##  [57] 53.61846 54.65751 38.53526 82.34462 52.68212 60.41166 69.38666 46.06130
##  [65] 47.40419 61.44111 40.32041 51.39102 78.02165 59.96898 50.18075 54.75996
##  [73] 81.92924 45.55012 42.18383 62.26823 62.06747 70.70899 29.05432 38.32979
##  [81] 59.46730 31.51399 43.59235 47.96043 40.52407 35.08822 43.51377 40.17733
##  [89] 43.41403 84.55618 45.04116 40.96168 59.59239 38.38171 49.40163 40.94027
##  [97] 59.01125 47.68123 50.32989 52.94093
```

```r
summary(x)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   15.78   40.84   50.06   51.60   62.05   96.65
```

1.1 Normalização [0,1]

```r
x_norm <- (x - min(x)) / (max(x) - min(x))
summary(x_norm)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.3098  0.4239  0.4429  0.5721  1.0000
```

1.2 Padronizar [-3,3]

```
x_pad <- (x - mean(x)) / sd(x)
summary(x_pad)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -2.26265 -0.67973 -0.09709  0.00000  0.66033  2.84644
```

A padronização dos dados nada mais é do que a transformação para a escala da curva normal padrão (z-padrão). Vide Figura 1 a tabela z-padrão.

```
## OLHANDO A TABELA
#z = 1,0 ----> p(z) = 0,1587
1 - 2 * 0.1587
```

```
## [1] 0.6826
```

```
#z = 2,0 ----> p(z) = 0,0228
1 - 2 * 0.0228
```

```
## [1] 0.9544
```

```
# p(z) = 95% -----> z(p = 0,025) = ?
1.96
```

```
## [1] 1.96
```

```
# p(z) = 99% -----> z =??
2.575
```

```
## [1] 2.575
```

2 Funções do R

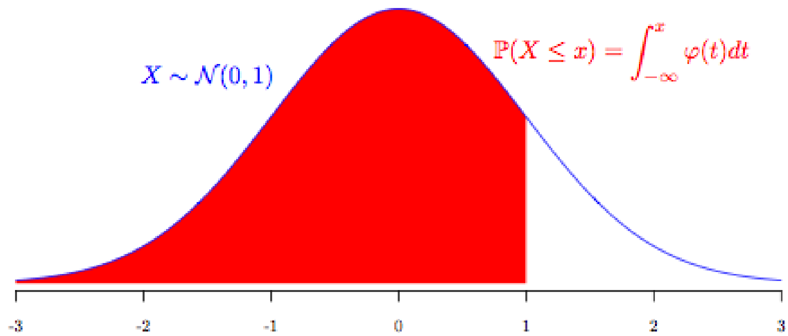
2.1 Números aleatórios

```
## Uniformemente distribuídos
```

Função: `runif(n, min, max)`

```
runif(10)
```

```
##  [1] 0.7446925 0.5346661 0.5412608 0.4029396 0.9155983 0.5955964 0.5216851
##  [8] 0.0957525 0.4979599 0.4498968
```



	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.7	0.7580	0.7611	0.7642	0.7673	0.7704	0.7734	0.7764	0.7794	0.7823	0.7852
0.8	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.9	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1.0	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621
1.1	0.8643	0.8665	0.8686	0.8708	0.8729	0.8749	0.8770	0.8790	0.8810	0.8830
1.2	0.8849	0.8869	0.8888	0.8907	0.8925	0.8944	0.8962	0.8980	0.8997	0.9015
1.3	0.9032	0.9049	0.9066	0.9082	0.9099	0.9115	0.9131	0.9147	0.9162	0.9177
1.4	0.9192	0.9207	0.9222	0.9236	0.9251	0.9265	0.9279	0.9292	0.9306	0.9319
1.5	0.9332	0.9345	0.9357	0.9370	0.9382	0.9394	0.9406	0.9418	0.9429	0.9441
1.6	0.9452	0.9463	0.9474	0.9484	0.9495	0.9505	0.9515	0.9525	0.9535	0.9545
1.7	0.9554	0.9564	0.9573	0.9582	0.9591	0.9599	0.9608	0.9616	0.9625	0.9633
1.8	0.9641	0.9649	0.9656	0.9664	0.9671	0.9678	0.9686	0.9693	0.9699	0.9706
1.9	0.9713	0.9719	0.9726	0.9732	0.9738	0.9744	0.9750	0.9756	0.9761	0.9767
2.0	0.9772	0.9778	0.9783	0.9788	0.9793	0.9798	0.9803	0.9808	0.9812	0.9817
2.1	0.9821	0.9826	0.9830	0.9834	0.9838	0.9842	0.9846	0.9850	0.9854	0.9857
2.2	0.9861	0.9864	0.9868	0.9871	0.9875	0.9878	0.9881	0.9884	0.9887	0.9890
2.3	0.9893	0.9896	0.9898	0.9901	0.9904	0.9906	0.9909	0.9911	0.9913	0.9916
2.4	0.9918	0.9920	0.9922	0.9925	0.9927	0.9929	0.9931	0.9932	0.9934	0.9936
2.5	0.9938	0.9940	0.9941	0.9943	0.9945	0.9946	0.9948	0.9949	0.9951	0.9952
2.6	0.9953	0.9955	0.9956	0.9957	0.9959	0.9960	0.9961	0.9962	0.9963	0.9964
2.7	0.9965	0.9966	0.9967	0.9968	0.9969	0.9970	0.9971	0.9972	0.9973	0.9974
2.8	0.9974	0.9975	0.9976	0.9977	0.9977	0.9978	0.9979	0.9979	0.9980	0.9981
2.9	0.9981	0.9982	0.9982	0.9983	0.9984	0.9984	0.9985	0.9985	0.9986	0.9986
3.0	0.9987	0.9987	0.9987	0.9988	0.9988	0.9989	0.9989	0.9989	0.9990	0.9990

Figure 1: Tabela Z-padrão

```r
runif(10, 100, 150)
```

```
##  [1] 108.4437 120.4169 133.6860 123.8815 128.2571 121.4598 101.1226 111.9532
##  [9] 128.4889 106.8365
```

```r
## Normalmente distribuídos
```

Função: `rnorm(n, mean, sd)`

```r
rnorm(10)
```

```
##  [1]  0.519541457 -0.637647229 -0.446732105 -0.428468526  0.009846958
##  [6]  1.320150538  0.015717242 -1.107154262 -0.846468565 -0.956322799
```

```r
rnorm(10, 100, 15)
```

```
##  [1]  87.81023 100.38945 100.87788 115.11102 116.57867 125.59136 102.83233
##  [8] 130.02110 100.60110 112.35421
```

2.2 Distribuição Normal

Encontrando o valor z-padrão com a função `qnorm(area da curva, mean=0, sd=1)`

- Unicaudal a esquerda: $z_\alpha = qnorm(\alpha)$
- Unicaudal a direita: $z_\alpha = qnorm(1 - \alpha)$
- Bicaudal: $z_frac\alpha 2 = qnorm(1 - \alpha/2)$

```r
qnorm(.90)
```

```
## [1] 1.281552
```

```r
qnorm(.5)
```

```
## [1] 0
```

Encontrando o p-valor com a função `pnorm(valor z, mean=0, sd=1)`

- Unicaudal a esquerda: $p - value = pnorm(z, lower.tail = TRUE)$
- Unicaudal a direita: $p - value = pnorm(z, lower.tail = FALSE)$
- Bicaudal: $p - value = 2 * pnorm(abs(z), lower.tail = FALSE)$

```r
pnorm(1.96)
```

```
## [1] 0.9750021
```

```r
pnorm(1.96, lower.tail = FALSE)
```

```
## [1] 0.0249979
```

```r
pnorm(0)
```

```
## [1] 0.5
```

Encontrando a densidade do valor com a função `dnorm(valor z, mean=0, sd=1)`

```r
dnorm(1.96)
```

```
## [1] 0.05844094
```

```r
dnorm(-1.96)
```

```
## [1] 0.05844094
```

```r
dnorm(0)
```

```
## [1] 0.3989423
```

EXEMPLO 1

```r
## P(z > 1,65)
pnorm(1.65, lower.tail = FALSE)
```

```
## [1] 0.04947147
```

```r
## P(z < 1,65)
pnorm(1.65)
```

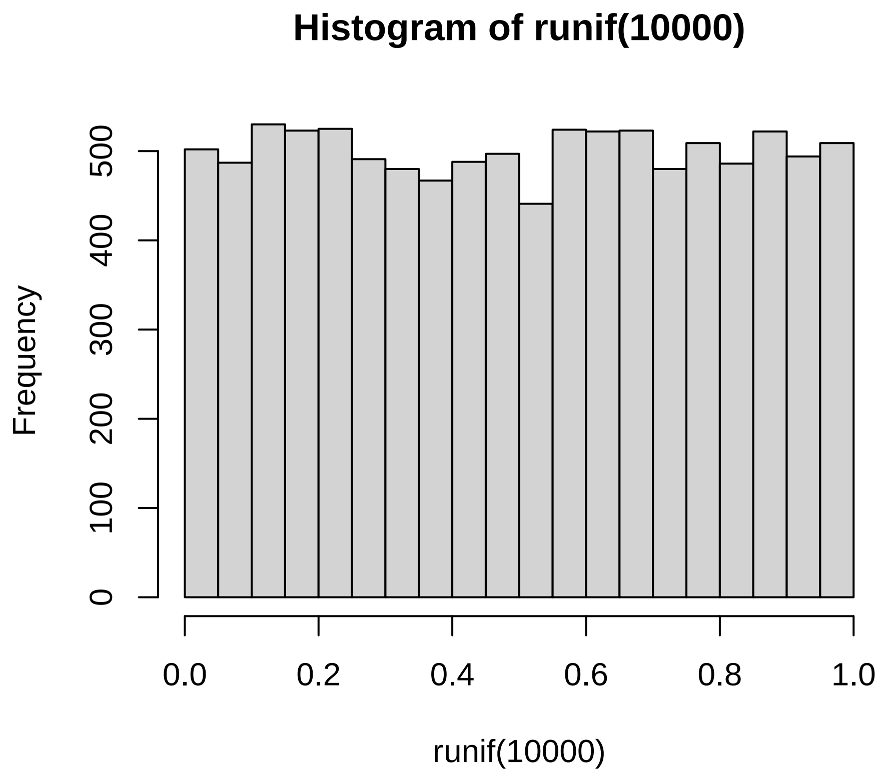
```
## [1] 0.9505285
```

```r
## P(1,40 < z < 1,70)
pnorm(1.7) - pnorm(1.4)
```

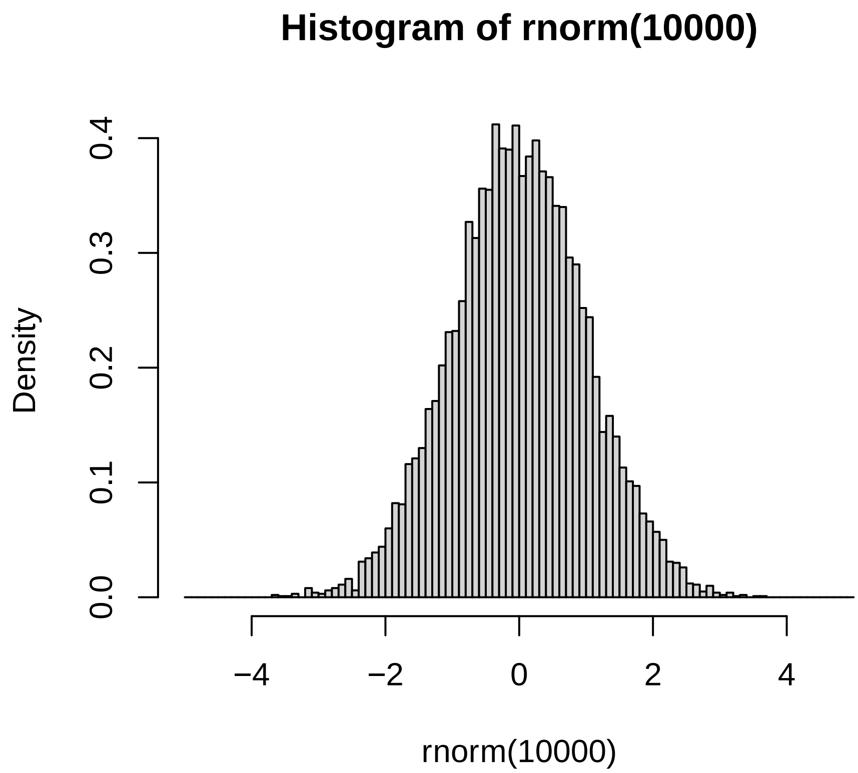
```
## [1] 0.0361912
```

2.3 Histogramas

```r
hist(runif(10000))
```




```
hist(rnorm(10000), breaks = seq(-5,5,.1),
     freq = FALSE)
```



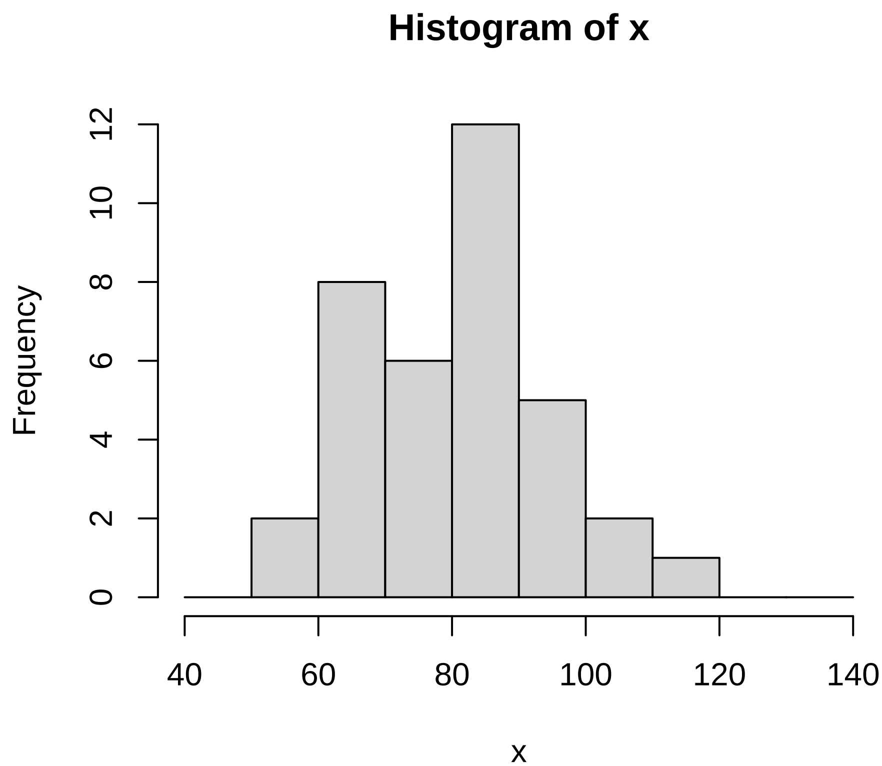
EXEMPLO 2

```
x = c(58,78,84,90,97,70,
      90,86,82,59,90,70,
      74,83,90,75,88,84,
      68,96,70,94,70,110,
      67,68,75,80,68,82,
      104,92,112,84,98,80)


## Análise descritiva
summary(x)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   58.00   70.00   82.50   82.39   90.00  112.00
```

```
hdados <- hist(x,
               breaks = seq(40,140,10))
```




```r
hdados$breaks
```

```
##  [1]  40  50  60  70  80  90 100 110 120 130 140
```

```r
hdados$counts
```

```
##  [1]  0  2  8  6 12  5  2  1  0  0
```

```r
hdados$density
```

```
##  [1] 0.000000000 0.005555556 0.022222222 0.016666667 0.033333333 0.013888889
##  [7] 0.005555556 0.002777778 0.000000000 0.000000000
```

O parâmetro `density` traz a razão entre a porcentagem de elementos e o intervalo de bins, tanto que a soma das porcentagens `density` é igual a
`0.1`

```r
#.

sum(hdados$density)
```

```
## [1] 0.1
```

Ao multiplicar cada densidade pelo intervalo do bin, a porcentagem total será de 100%.

```
sum(hdados$density) * 10
```

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

3 Testes do R

```
x_pad <- (x - mean(x))/sd(x)
```

3.1 Qui-quadrado

```
chisq.test(x, rnorm(36, mean(x), sd(x)) )
```

```
## Warning in chisq.test(x, rnorm(36, mean(x), sd(x))): Chi-squared approximation
## may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  x and rnorm(36, mean(x), sd(x))
## X-squared = 792, df = 770, p-value = 0.2836
```

3.2 Kolomogorv-Smirnof

```
ks.test(x, "pnorm", mean(x), sd(x))
```

```
## Warning in ks.test(x, "pnorm", mean(x), sd(x)): ties should not be present for
## the Kolmogorov-Smirnov test
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  x
## D = 0.10407, p-value = 0.8304
## alternative hypothesis: two-sided
```

```
ks.test(x_pad, "pnorm")
```

```
## Warning in ks.test(x_pad, "pnorm"): ties should not be present for the
## Kolmogorov-Smirnov test
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  x_pad
## D = 0.10407, p-value = 0.8304
## alternative hypothesis: two-sided
```

3.3 Shapiro

```
shapiro.test(x)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  x
## W = 0.97612, p-value = 0.6139
```