Cvic 10

Příklad 1 – kostka

Vyřešíme dvěma způsoby (přímé dosazení do vzorece nebo použití funkce chisq.test). V obou případech samozřejmě vyjde totéž.

```
N=50
kostka = sample(6,N, replace=T)
#kostka = c(....)
obs = rep(0,6); obs
```

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

```
for(i in 1:6){
  obs[i] = sum(kostka==i)
}
obs
```

```
## [1] 13  5  7  5  9 11
```

```
sum(obs)
```

```
## [1] 50
```

```
T = sum((obs-N/6)^2/(N/6)); T
```

```
## [1] 6.4
```

```
1-pchisq(T,5)
```

```
## [1] 0.2692188
```

```
chisq.test(obs)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  obs
## X-squared = 6.4, df = 5, p-value = 0.2692
```

Totéž stručněji.

```
kostka = sample(6, 100, replace=T)
t=table(kostka)
chisq.test(t)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  t
## X-squared = 1.52, df = 5, p-value = 0.9108
```

Příklad 2 – emaily

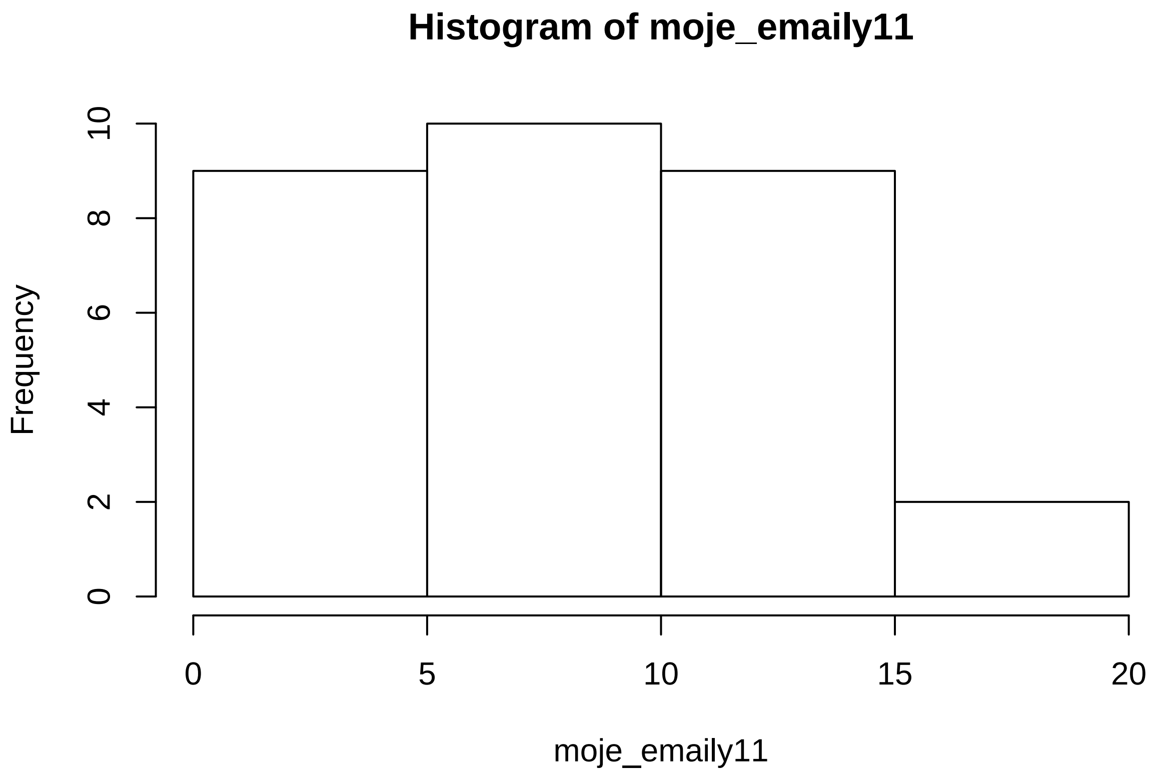
```
moje_emaily11 = c(0,6,14,8,8,9,3,3,12,12,15,7,15,2,5,13,5,17,15,11,9,2,16,8,9,11,6,2,2,9)
moje_emaily12 = c(13,14,3,8,5,4,12,22,8,4,5,3)
mean(moje_emaily11)
```

```
## [1] 8.466667
```

```
var(moje_emaily11)
```

```
## [1] 23.63678
```

```
hist(moje_emaily11)
```



```
day = 1:30
data = moje_emaily12[day %% 7!=5 & day %%7 != 6 & day <= length(moje_emaily12)]
data
```

```
## [1] 13 14  3  8 12 22  8  4  5
```

```
mean(data)
```

```
## [1] 9.888889
```

```
var(data)
```

```
## [1] 36.36111
```

```
n = 15
bins = 0:n
day = 1:30
data = moje_emaily11[day %% 7!=1 & day %%7 != 0 & day!=17]
lambda = mean(data); lambda
```

```
## [1] 11.05
```

```
var(data)
```

```
## [1] 12.05
```

```
p = dpois(bins,lambda)
p[n+1] = 1-ppois(n-1,lambda)
p
```

```
##  [1] 1.588715e-05 1.755530e-04 9.699303e-04 3.572577e-03 9.869243e-03
##  [6] 2.181103e-02 4.016864e-02 6.340907e-02 8.758378e-02 1.075334e-01
## [11] 1.188244e-01 1.193645e-01 1.099148e-01 9.342762e-02 7.374108e-02
## [16] 1.496184e-01
```

```
sum(p)
```

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

```
freq = bins*0
freq = rep(0,n+1)
for(i in bins){ freq[i+1] = sum(data==i) }
freq[n+1] = sum(data>=n)
freq
```

```
##  [1] 0 0 0 0 0 0 2 1 3 3 0 2 2 1 1 5
```

```
N = sum(freq); N
```

```
## [1] 20
```

```
length(data)
```

```
## [1] 20
```

```
T = sum((freq-p*N)^2/(p*N))
T
```

```
## [1] 8.153163
```

```
options(digits = 8)
pchisq(T,n)
```

```
## [1] 0.082504621
```

```
chisq.test(x=freq,p=p)
```

```
## Warning in chisq.test(x = freq, p = p): Chi-squared approximation may be
## incorrect
```

```
##
##  Chi-squared test for given probabilities
##
## data:  freq
## X-squared = 8.15316, df = 15, p-value = 0.9175
```

```
bin_ends = c(-Inf,5,8,10,12,Inf)

day = 1:30
data = moje_emaily11[day %% 7!=1 & day %%7 != 0 & day!=17]
lambda = mean(data); lambda
```

```
## [1] 11.05
```

```
var(data)
```

```
## [1] 12.05
```

```
p = diff(ppois(bin_ends,lambda))
sum(p)
```

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

```
freq = bin_ends*0
for(i in 1:length(bin_ends)){
  freq[i] = sum(data<=bin_ends[i])
}
freq = diff(freq)

N = sum(freq); N
```

```
## [1] 20
```

```
length(data)
```

```
## [1] 20
```

```
T = sum((freq-p*N)^2/(p*N))
T
```

```
## [1] 2.6272196
```

```
pchisq(T,n)
```

```
## [1] 0.00017470158
```

```
chisq.test(x=freq,p=p)
```

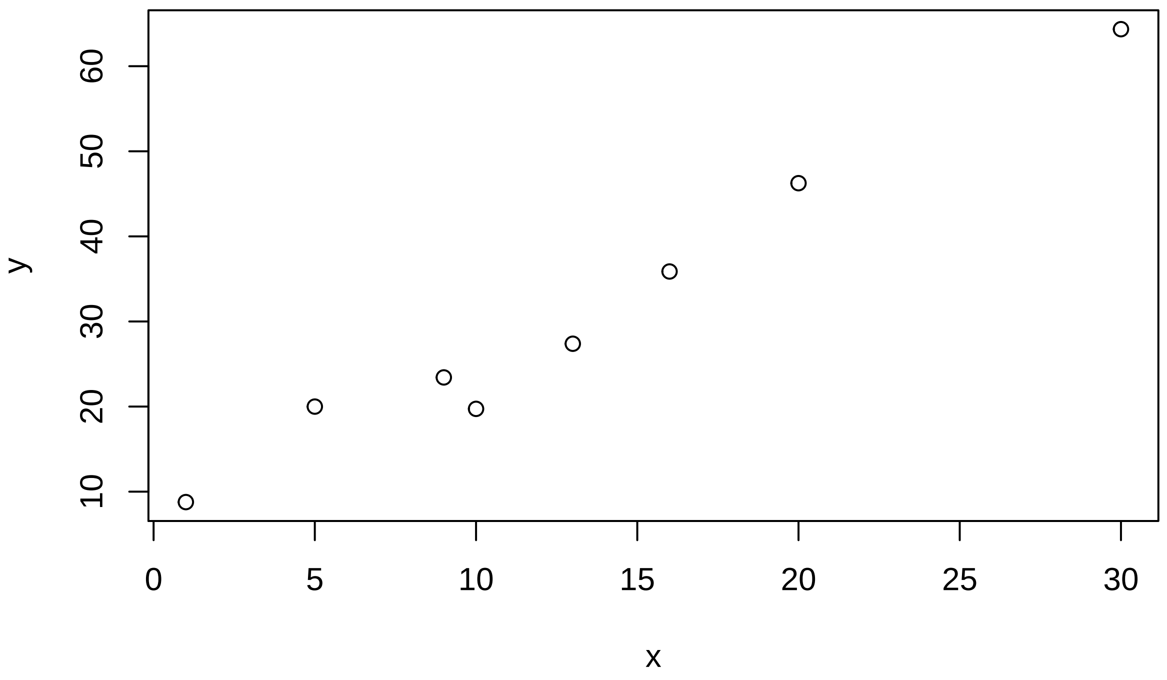
```
## Warning in chisq.test(x = freq, p = p): Chi-squared approximation may be
## incorrect
```

```
##
##  Chi-squared test for given probabilities
##
## data:  freq
## X-squared = 2.62722, df = 4, p-value = 0.62201
```

Příklad 3 – regrese

V zadání byla řečena data pro x a y. Zde vidíme (v zakomentované části) i jak byla data vyrobena: k ideálnímu vzorci přičteme náhodný šum. Můžeme pak dobře sledovat, jak se spočtené řešení bude lišit od "ideálu".

```
x = c(1,5,9,10,13,16,20,30)
y = c(6.1982, 12.9892, 23.8005, 23.8891, 30.0391, 35.7535, 49.0685, 63.1825)
y = 2*x+4 + rnorm(length(x),0,3)
#y = 2*x+4 + rnorm(1,0,3)
plot(x,y)
```



```r
xm = mean(x)
ym = mean(y)
a = sum((x-xm)*(y-ym))/sum((x-xm)^2); a
```

```
## [1] 1.9088427
```

```r
cov(x,y)/var(x)
```

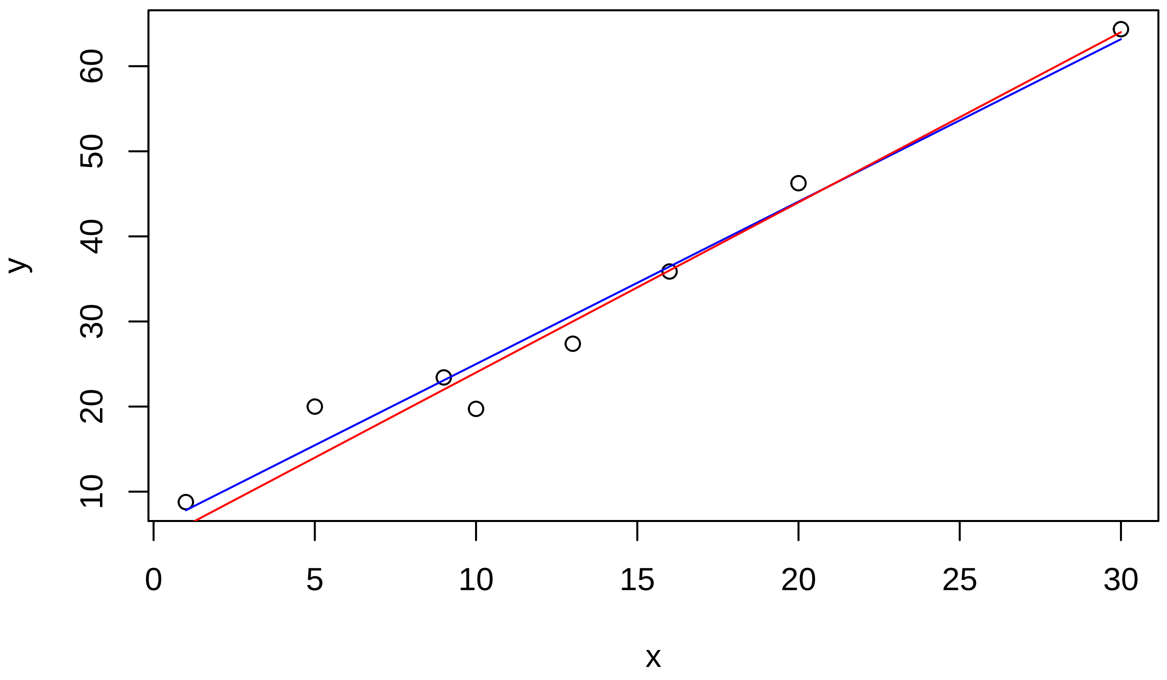
```
## [1] 1.9088427
```

```r
b = ym - a*xm; b
```

```
## [1] 5.9079694
```

Červená je ta původní přímka, před přičtením šumu.

```r
plot(x,y)
lines(x,a*x+b, col="blue")
lines(x,2*x+4, col="red")
```



Knihovní funkce na regresy ("linear model"). Umí např. i závislost na více proměnných (tak lze např. hledat aproximující polynom).

```
relation <- lm(y~x)

summary(relation)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.2662 -1.2668  0.6483  1.4228  4.5437
##
## Coefficients:
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept)  5.90797    2.15586  2.7404   0.03372 *
## x            1.90884    0.13873 13.7597 9.164e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.341 on 6 degrees of freedom
## Multiple R-squared:  0.96928,    Adjusted R-squared:  0.96416
## F-statistic: 189.33 on 1 and 6 DF,  p-value: 9.1637e-06
```

```
relation$coefficients
```

```
## (Intercept)           x
##   5.9079694   1.9088427
```

```
res = y-(a*x+b); res
```

```
## [1]  0.95929118  4.54373359  0.33730788 -5.26619658 -3.33990675 -0.57571911
## [7]  2.16661645  1.17487335
```