Regressão linear

Vamos considerar o seguinte experimento: penduramos, de forma sucessiva, padrões de massa conhecida em uma mola de constante elástica desconhecida. O objetivo do experimento é determinar a constante elástica da mola.

Sabemos que, no regime elástico, o deslocamento da mola segue a Lei de Hooke, ou seja,

$$F_{el} = -kx$$

Como estamos tratando de um experimento, encontramos erro experimental.

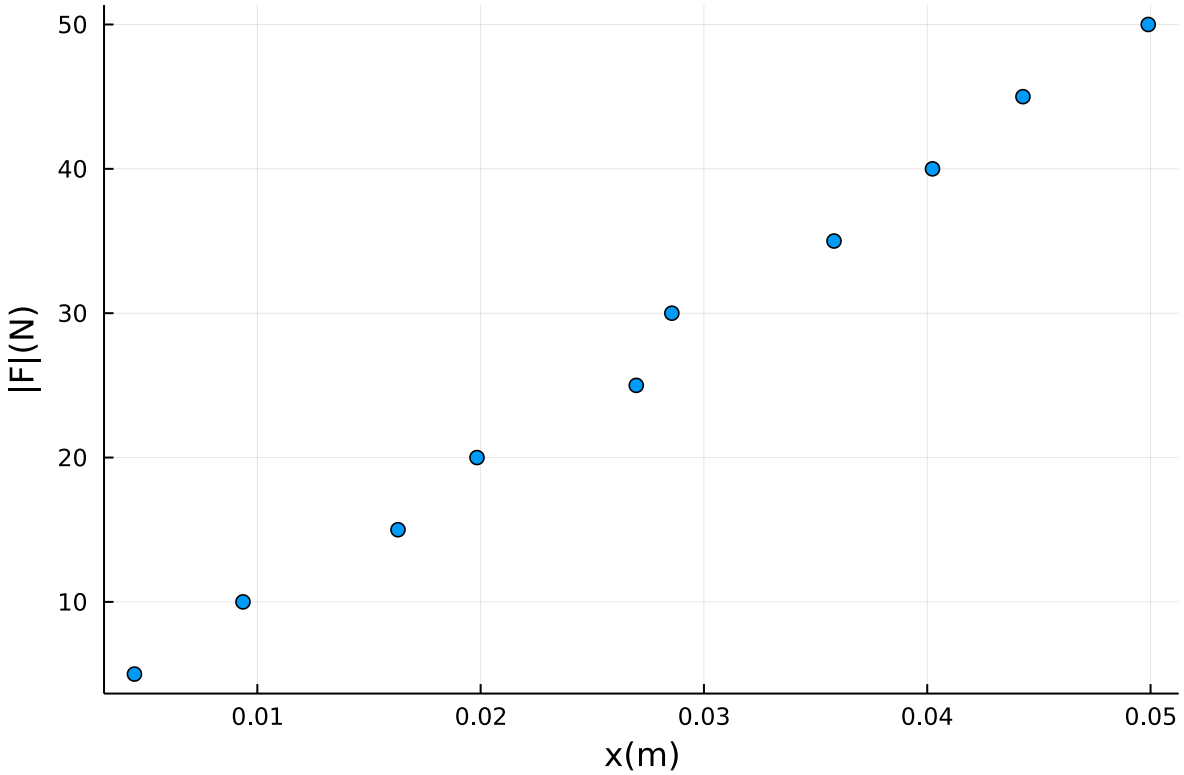
```
begin
    using Random
    using Distributions
    using Plots
end
```

```
simdisp (generic function with 1 method)
```

```
# Função para gerar dados fictícios de um experimento (com ruído)
function simdisp(k,σx)
    distx = Normal(0.0, σx)
    F = collect(5:5:50)
    errx = rand(distx, length(F))
    x = F ./ k + errx
    return x
end
```

```
medidas =
 [0.00449726, 0.00935762, 0.016299, 0.0198365, 0.0269695, 0.0285628, 0.035825, 0.0402346,
```

```
medidas = simdisp(1000,0.001) # x em metros
```




```
# Plotando dados experimentais (fictícios)
begin
    scatter(medidas, 5:5:50, label=false)
    xlabel!("x(m)")
    ylabel!("|F|(N)")
end
```

```
using LinearAlgebra
```

Resolvendo pela projeção

$$A^T Ac = A^T f$$

em que A é a matriz que contém os deslocamentos medidos, c é o vetor dos coeficientes (o que queremos encontrar) e f o vetor das forças medidas.

```
A = 10×2 Matrix{Float64}:
     1.0  0.00449726
     1.0  0.00935762
     1.0  0.016299
     1.0  0.0198365
     1.0  0.0269695
     1.0  0.0285628
     1.0  0.035825
     1.0  0.0402346
     1.0  0.0442859
     1.0  0.0498967
```

```
A = [fill(1,length(medidas)) medidas] # Matrix A
```

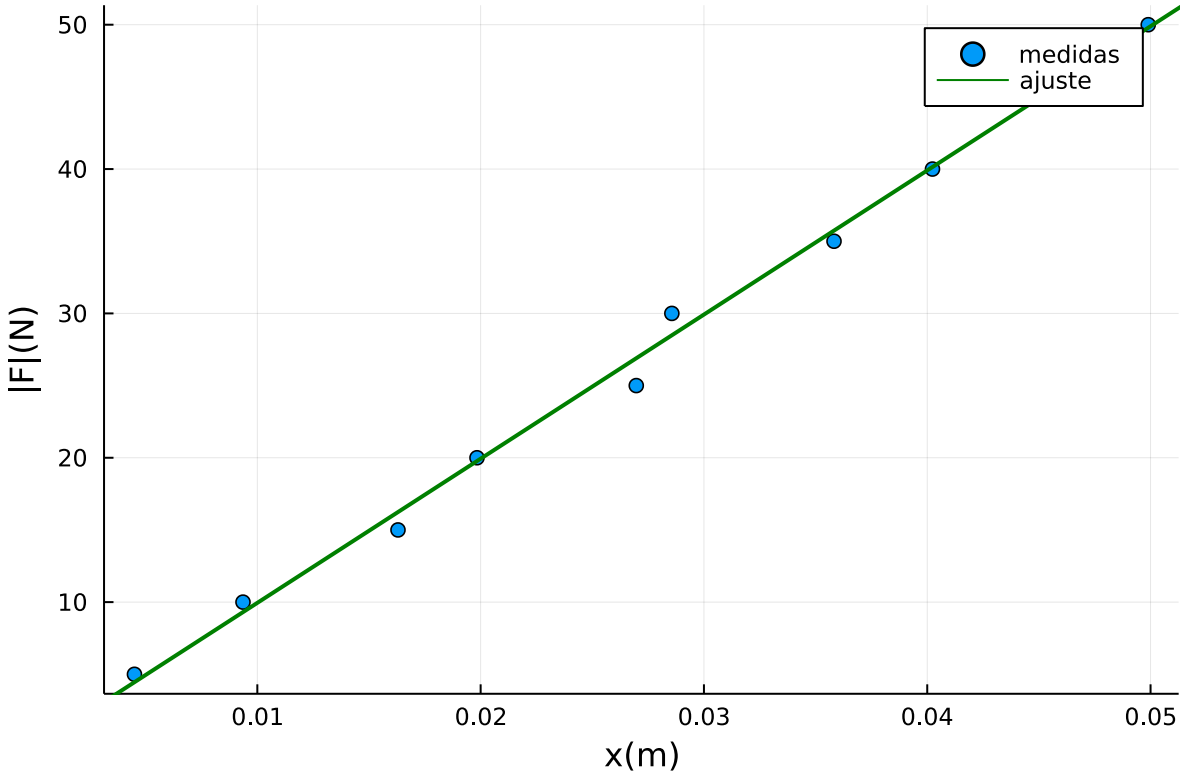
```
b =  [5, 10, 15, 20, 25, 30, 35, 40, 45, 50]
```

```
b = collect(5:5:50) # Dados de força medidos
```

```
coefs =  [-0.0383571, 998.617]
```

```
# Regressão
coefs = A'*A \ A'*b
```

Comparar o coeficiente angular com o valor da rigidez utilizada na simulação




```
# Plotando dados experimentais com ajuste
begin
    scatter(medidas, 5:5:50, label="medidas")
    xlabel!("x(m)")
    ylabel!("|F|(N)")
    Plots.abline!(coefs[2], coefs[1], linecolor="green", linewidth=2,
    label="ajuste")
end
```

Utilizando pacote LsqFit

Vamos utilizar o pacote LsqFit, escrito em Julia, encontrar o melhor ajuste.

```
using LsqFit
```

```
model (generic function with 1 method)
```

```
# modelo
@. model(x,p) = p[1]*x + p[2]
```

```
p0 =  [100.0, 0.0]
```

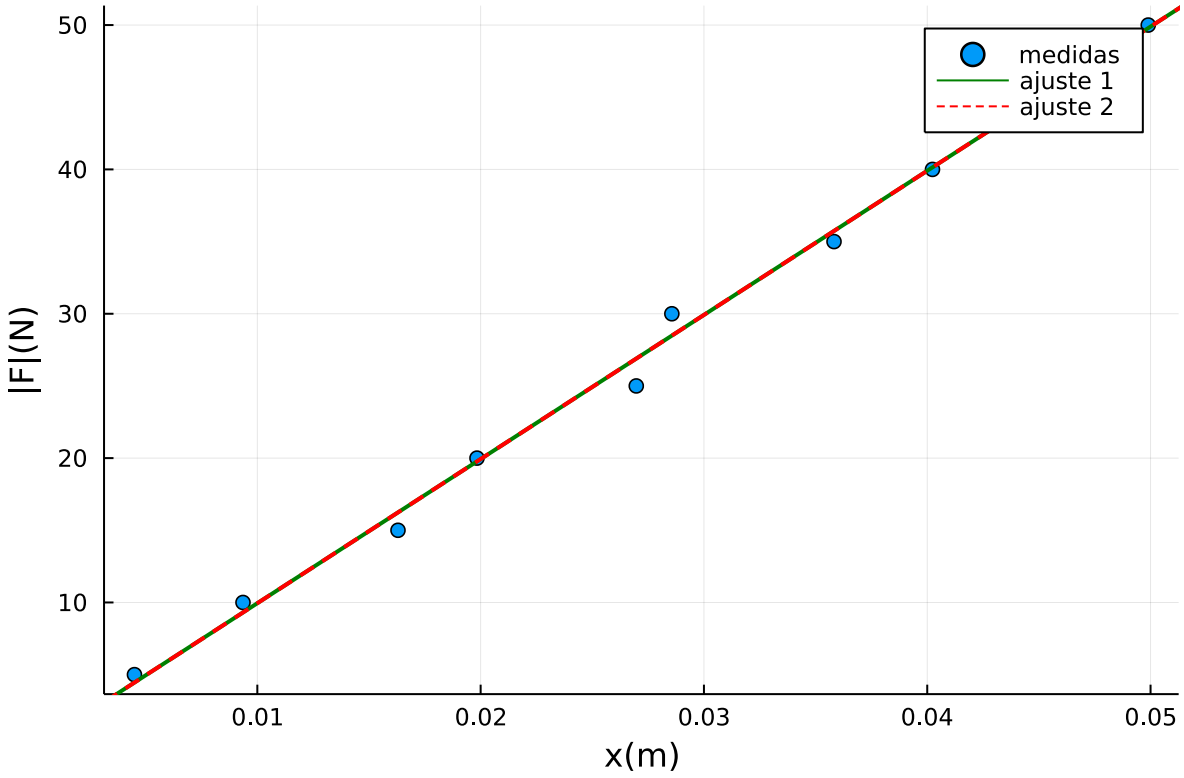
```
# Chute inicial para a rigidez
p0 = [100.0,0.0]
```

```
fit =
  LsqFitResult([998.617, -0.0383571], [-0.547318, -0.693671, 1.2381, -0.229323, 1.89387, -
```

```
# encontrando o melhor ajuste
fit = curve_fit(model, medidas, collect(5:5:50), p0)
```

```
[998.617, -0.0383571]
```

```
fit.param
```




```
# Plotando dados experimentais com ajustes
begin
    scatter(medidas, 5:5:50, label="medidas")
    xlabel!("x(m)")
    ylabel!("|F|(N)")
    Plots.abline!(coefs[2], coefs[1], linecolor="green", label="ajuste 1",
    linewidth=2)
    Plots.abline!(fit.param[1], fit.param[2], linecolor="red", linestyle=:dash,
    label="ajuste 2", linewidth=2)

end
```