1.2 Practical: Maximum likelihood

Benjamin Rosenbaum

November 2, 2021

Setup

```
rm(list=ls())
library(manipulate)
set.seed(123) # initiate random number generator for reproducibility
```

Generate data

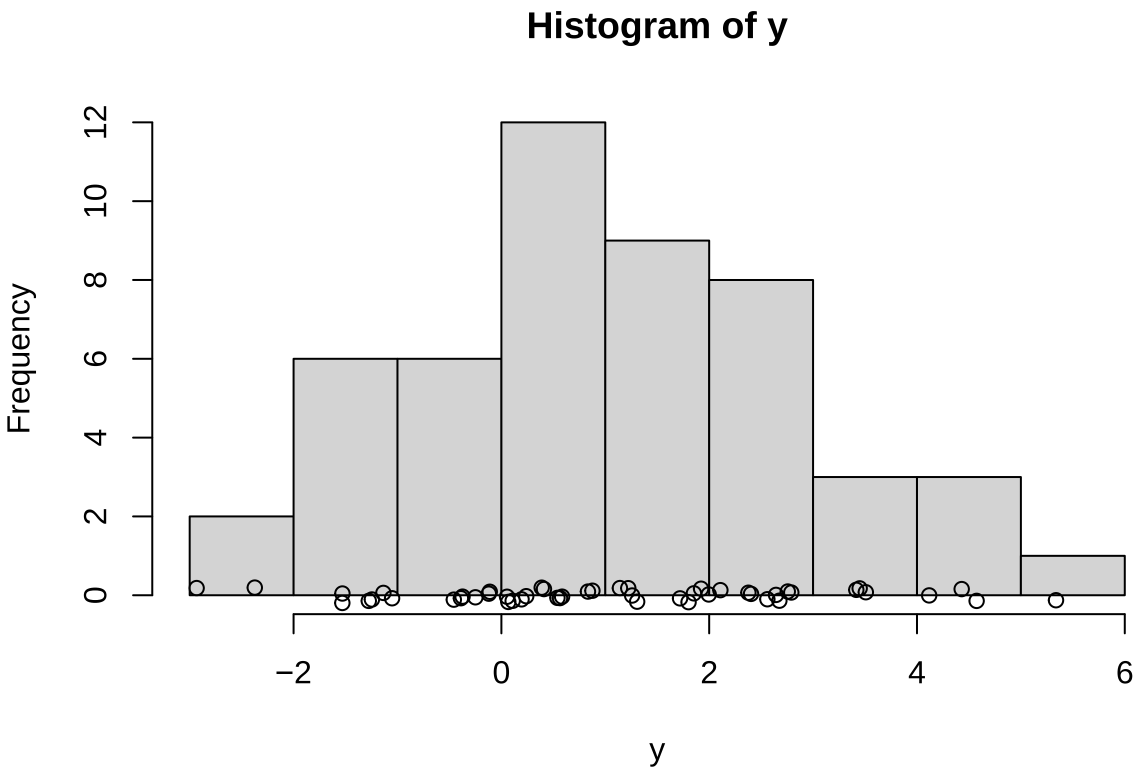
We draw sample data from a normal distribution.

```
n=50
y = rnorm(n=n, mean=1.0, sd=2.0)

y
```

```
##  [1] -0.12095129  0.53964502  4.11741663  1.14101678  1.25857547  4.43012997
##  [7]  1.92183241 -1.53012247 -0.37370570  0.10867606  3.44816359  1.71962765
## [13]  1.80154290  1.22136543 -0.11168227  4.57382627  1.99570096 -2.93323431
## [19]  2.40271180  0.05441718 -1.13564741  0.56405017 -1.05200890 -0.45778246
## [25] -0.25007854 -2.37338662  2.67557409  1.30674624 -1.27627387  3.50762984
## [31]  1.85292844  0.40985703  2.79025132  2.75626698  2.64316216  2.37728051
## [37]  2.10783531  0.87617658  0.38807467  0.23905800 -0.38941396  0.58416544
## [43] -1.53079270  5.33791193  3.41592400 -1.24621717  0.19423033  0.06668929
## [49]  2.55993024  0.83326187
```

```
hist(y)
points(y, jitter(rep(0, times=n), factor=10))
```



Statistical model, deterministic and stochastic part

statistical model: estimate mean and standard deviation

$$y_i \sim \text{normal}(\mu, \sigma)$$

The likelihood function

The likelihood function for a single datapoint is the probability density function

$$p(y_i|\mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left\{-\frac{(y_i - \mu)^2}{2\sigma^2}\right\}$$

For a single data point, the likelihood function `L` can be computed with the `dnorm()` function (for a given parameter combination μ, σ).

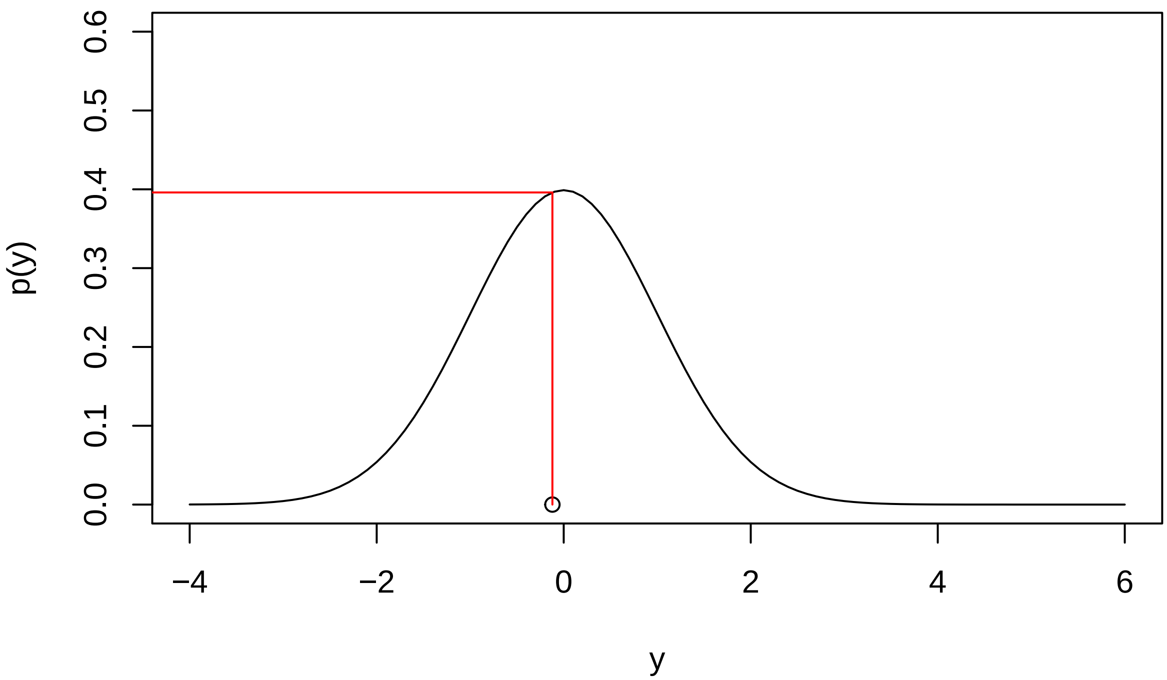
```
# likelihood of first data point:
i = 1
L = dnorm(x=y[i], mean=0, sd=1)
y[i]
```

```
## [1] -0.1209513
```

```
L
```

```
## [1] 0.3960348
```

```
# plot
curve(dnorm(x, mean=0, sd=1), from=-4, to=6, xlab="y", ylab="p(y)", ylim=c(0, 0.6))
points(y[i],0)
lines(c(y[i],y[i]), c(0, L), col="red")
lines(c(-50,y[i]), c(L, L), col="red")
```



For all datapoints, `dnorm()` can calculate all $p(y_i|\mu, \sigma)$ for given parameter combination μ, σ at once. `L` is a vector now.

```
L = dnorm(x=y, mean=0, sd=1)
L
```

```
##  [1] 3.960348e-01 3.448841e-01 8.309730e-05 2.080664e-01 1.806950e-01
##  [6] 2.183570e-05 6.293465e-02 1.237396e-01 3.720353e-01 3.965934e-01
## [11] 1.044879e-03 9.094520e-02 7.873111e-02 1.892275e-01 3.964620e-01
## [16] 1.143443e-05 5.445668e-02 5.402625e-03 2.224917e-02 3.983520e-01
## [21] 2.093420e-01 3.402703e-01 2.293973e-01 3.592557e-01 3.866605e-01
## [26] 2.386313e-02 1.112804e-02 1.698684e-01 1.766869e-01 8.496619e-04
## [31] 7.167520e-02 3.668032e-01 8.134103e-03 8.938010e-03 1.212978e-02
## [36] 2.364343e-02 4.326448e-02 2.717749e-01 3.700047e-01 3.877041e-01
## [41] 3.698121e-01 3.363634e-01 1.236127e-01 2.592196e-07 1.167131e-03
## [46] 1.835135e-01 3.914877e-01 3.980561e-01 1.506231e-02 2.819287e-01
```

The likelihood function of all datapoints for a given parameter combination μ, σ is the product of all single values

$$p(y_1, \ldots, y_n|\mu, \sigma) = p(y_1|\mu, \sigma) \cdot \ldots \cdot p(y_n|\mu, \sigma)$$

This holds because observations are independent

```
prod(L)
```

```
## [1] 1.43861e-69
```

There is a problem. Each $p(y_i|\mu, \sigma) < 1$.

So multiplying them all results in an extremely small number.

Better use

$$\begin{aligned} \log p(y_1, \ldots, y_n|\mu, \sigma) &= \log \{p(y_1|\mu, \sigma) \cdot \ldots \cdot p(y_n|\mu, \sigma)\} \\ &= \log p(y_1|\mu, \sigma) + \ldots + \log p(y_n|\mu, \sigma) \end{aligned}$$

log of a product is equal to sum of logs.

We minimize the negative log likelihood (NLL) to find model parameters, which is equivalent to maximum likelihood (mathematical convention is minimization instead of maximization)

```
-sum(log(L))
```

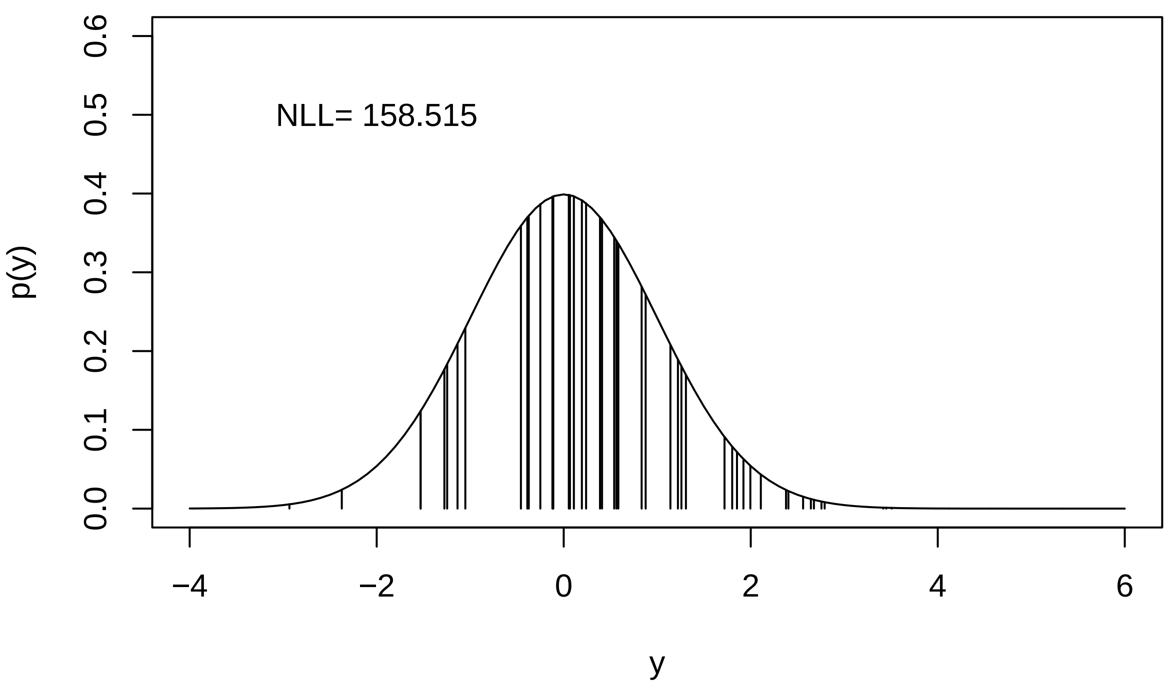
```
## [1] 158.5147
```

We can visualize the likelihood function of all datapoints for given parameters μ and σ.

You can play around with μ and σ.

Which combination maximizes likelihood of all datapoints at once?

```
curve.data <- function(mean, sd, y)
{
  # plot curve
  curve(dnorm(x, mean=mean, sd=sd), from=-4, to=6,
        xlab="y", ylab="p(y)", ylim=c(0, 0.6))
  # plot lines for all datapoints
  for(i in 1:n){
    lines(c(y[i],y[i]),c(0,dnorm(x=y[i], mean=mean, sd=sd)))
  }
  # plot NLL value as text
  L = dnorm(x=y, mean=mean, sd=sd)
  NLL = -sum(log(L))
  text(-2,0.5, paste("NLL=",round(NLL,3)) )
}
manipulate(curve.data(mean, sd, y),
           mean=slider(-3, 3, step=0.1, initial=0),
           sd=slider(0.1,4, step=0.1, initial=1) )
```



Maximum likelihood with `optim()`

We can use mathematical algorithms to search for the best parameter combination automatically.

First, we define a function that directly calculates the NLL for the data and a given parameter combination

```r
nll.function = function(data, par){
  LL = dnorm(x=data, mean=par[1], sd=par[2], log=TRUE) # LL: log-likelihood
  NLL = -sum(LL) # nll: negative log likelihood
  return(NLL)
}
```

Example: for mu=0, sd=1

```r
nll.function(data=y, par=c(0.0, 1.0))
```

```
## [1] 158.5147
```

`optim()` function automatically searches for parameters that minimize the NLL

```r
optim(par=c(0.0, 1.0), # initial guess mu=0, sd=1
      fn=nll.function,
      data=y)
```

```
## $par
## [1] 1.068710 1.833387
##
## $value
## [1] 101.2481
##
## $counts
## function gradient
##       61       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

The maximum likelihood estimates are $\mu = 1.068710$, $\sigma = 1.833387$