We all hear about Maximum Likelihood Estimation (MLE) and we often see hints of it in our model output. As usual, doing things manually can give a better grasp on how to better understand how our models work. Here's a very short example implementing MLE based on the explanation from Gelman and Hill (2007), page 404-405.

The likelihood is literally how much our outcome variable **Y** is compatible with our predictor **X**. We compute this measure of compatibility with the probability density function for the normal distribution. In R, `dnorm` returns this likelihood. The plot on this website gives a very clear intuition on what `dnorm` returns: it is literally the *height* of the distribution, or in other words, the likelihood. We of course, want the highest likelihood, as it indicates greater compatibility.

For example, assuming `parameters` is a vector with the intercept `a`, the coefficient `b` and an error term `sigma`, we can compute the likelihood for any random value of these coefficients:

```
loglikelihood <- function(parameters, predictor, outcome) {
  # intercept
  a <- parameters[1]
  # beta coef
  b <- parameters[2]
  # error term
  sigma <- parameters[3]

  # Calculate the likelihood of `y` given `a + b * x`
  ll.vec <- dnorm(outcome, a + b * predictor, sigma, log = TRUE)

  # sum that likelihood over all the values in the data
  sum(ll.vec)
}

# Generate three random values for intercept, beta and error term
inits <- runif(3)

# Calculate the likelihood given these three values
loglikelihood(
  inits,
  predictor = mtcars$disp,
  outcome = mtcars$mpg
)
## [1] -11687.41
```

That's the likelihood given the random values for the intercept, the coefficient and sigma. How does a typical linear model estimate the **maximum** of these likelihoods? It performs an optimization search trying out a sliding set of values for these unknowns and searches for the combination that returns the maximum:

```
mle <-
  optim(
    inits, # The three random values for intercept, beta and sigma
    loglikelihood, # The loglik function
    lower = c(-Inf, -Inf, 1.e-5), # The lower bound for the three
values (all can be negative except sigma, which is 1.e-5)
```

```
    method = "L-BFGS-B",
    control = list(fnscale = -1), # This signals to search for the
maximum rather than the minimum
    predictor = mtcars$disp,
    outcome = mtcars$mpg
  )

mle$par[1:2]
## [1] 29.59985346 -0.04121511
```

Let’s compare that to the result of `lm`:

```
coef(lm(mpg ~ disp, data = mtcars))
## (Intercept)        disp
## 29.59985476 -0.04121512
```

In layman terms, MLE really just checks how compatible a given data point is with the outcome with the respect to a coefficient. It repeats that step many times until it finds the combination of coefficients that maximizes the outcome.