So, lets try to implement this in `R`. First we need to extract the Hessian matrix from our `optimx()` result object. Note, that you need to set the option `hessian = TRUE` in your `optimx()` call. This asks `optimx()` to estimate the Hessian matrix for the different optimization algorithms and allows us to obtain this information after the optimization is finished. In the example below, I only obtain the Hessian matrix for the optimization algorithm `Rcgmin`, since it showed the best fit compared to the results from the `glm()` model.

```
# 7. Estimate the standard error -----------------------------------------------
#Extract hessian matrix for Rcgmin optimisation
hessian_m <- attributes(opt)$details["Rcgmin", "nhatend"][[1]]
```

After we extracted the Hessian matrix, we can follow the procedure described above. Also note, that I used the Hessian matrix, instead of the negative Hessian matrix in my example. When I used the negative Hessian matrix, I got negative values for the diagonal values of the inverse. Hence, I was not able to obtain the squared root of these values. Also, I obtained the correct SEs using this approach.

```
# Estimate se based on hession matrix
fisher_info <- solve(hessian_m)
prop_se <- sqrt(diag(fisher_info))
```

Now were we obtained our estimates for the SEs, it would be interesting to compare them with the results of a `glm()` call, that tries to fit the same model as we do.

```
# Compare the estimated se from our model with the one from the glm
ses <- data.frame(se_Rcgmin = prop_se,
se_glm = tidy(glm_model)$std.error) %>%
print()
```

```
## se_Rcgmin se_glm
## 1 0.69888433 0.69884208
## 2 0.01065624 0.01065569
## 3 0.37177192 0.37176526
```

```
all.equal(ses[,"se_Rcgmin"], ses[, "se_glm"])
```

```
## [1] "Mean relative difference: 4.57513e-05"
```

The differences between the estimates of the SEs using the Hessian matrix and the `glm()` model are very small. It seems like our approach did a fairly good job. Hence, we can now use our SE estimates to compute the 95%CIs of our point estimates.

```
# 8. Estimate 95%CIs using estimation of SE ------------------------------------
# Extracting estimates from Rcgmin optimisaiton
coef_test <- coef(opt)["Rcgmin",]
# Compute 95%CIs
upper <- coef_test + 1.96 * prop_se
lower <- coef_test - 1.96 * prop_se
# Print 95%CIs
data.frame(coef_test, lower=lower, upper=upper, se = prop_se)
```

```
## coef_test lower upper se
## p1 -3.05669070 -4.426503993 -1.68687741 0.69888433
## p2 0.02758409 0.006697859 0.04847032 0.01065624
## p3 -0.01131098 -0.739983952 0.71736199 0.37177192
```