When I learned and experimented a new model, I always like to start with its likelihood function in order to gain a better understanding about the statistical nature. That’s why I extensively used the SAS/NLMIXED procedure that gives me more flexibility. Today, I spent a couple hours playing the optim() function and its wrappers, e.g. mle() and mle2(), in case that I might need a replacement for my favorite NLMIXED in the model estimation. Overall, I feel that the optim() is more flexible. The named list required by the mle() or mle2() for initial values of parameters is somewhat cumbersome without additional benefits. As shown in the benchmark below, the optim() is the most efficient.

library(COUNT)

library(stats4)

library(bbmle)

data(rwm1984)

attach(rwm1984)

### OPTIM() ###

LogLike1 <- function(par) {

xb <- par[1] + par[2] \* outwork + par[3] \* age + par[4] \* female + par[5] \* married

mu <- exp(xb)

ll <- sum(log(exp(-mu) \* (mu ^ docvis) / factorial(docvis)))

return(-ll)

}

fit1 <- optim(rep(0, 5), LogLike1, hessian = TRUE, method = "BFGS")

std1 <- sqrt(diag(solve(fit1$hessian)))

est1 <- data.frame(beta = fit1$par, stder = stder, z\_values = fit1$par / stder)

# beta stder z\_values

#1 -0.06469676 0.0433207574 -1.493436

#2 0.27264177 0.0214085110 12.735205

#3 0.02283541 0.0008394589 27.202540

#4 0.27461355 0.0210597539 13.039732

#5 -0.11804504 0.0217745647 -5.421236

### MLE() ###

LogLike2 <- function(b0, b1, b2, b3, b4) {

mu <- exp(b0 + b1 \* outwork + b2 \* age + b3 \* female + b4 \* married)

-sum(log(exp(-mu) \* (mu ^ docvis) / factorial(docvis)))

}

inits <- list(b0 = 0, b1 = 0, b2 = 0, b3 = 0, b4 = 0)

fit2 <- mle(LogLike2, method = "BFGS", start = inits)

std2 <- sqrt(diag(vcov(fit2)))

est2 <- data.frame(beta = coef(fit2), stder = std2, z\_values = coef(fit2) / std2)

# beta stder z\_values

#b0 -0.06469676 0.0433417474 -1.492712

#b1 0.27264177 0.0214081592 12.735414

#b2 0.02283541 0.0008403589 27.173407

#b3 0.27461355 0.0210597350 13.039744

#b4 -0.11804504 0.0217746108 -5.421224

### BENCHMARKS ###

microbenchmark::microbenchmark(

"optim" = {optim(rep(0, 5), LogLike1, hessian = TRUE, method = "BFGS")},

"mle" = {mle(LogLike2, method = "BFGS", start = inits)},

"mle2" = {mle2(LogLike2, method = "BFGS", start = inits)},

times = 10

)

# expr min lq mean median uq max neval

# optim 280.4829 280.7902 296.9538 284.5886 318.6975 320.5094 10

# mle 283.6701 286.3797 302.9257 289.8849 327.1047 328.6255 10

# mle2 387.1912 390.8239 407.5090 392.8134 427.0569 467.0013 10