FCT NOVA

Project Report

Optimization Methods and Generalized Linear Models

Leonard Storcks

Course by Dr. Regina Bispo

January 12, 2024

Abstract

At the example of binomial data on whether a face recognition algorithm can identify a person or not (depending on facial features), we employ a Generalized Linear Model (GLM) with (non-canonical) complementary log-log link and estimate the parameters using different numerical schemes.

Contents

1 Binomial GLM with complementary log-log link function for Predicting if a Face can be Recognized

1.1 Introduction of the Data Set and Task

Consider a situation, where our response variable Y is binary and indicates whether a face recognition algorithm was able to recognize a face or not. The predictor variables are

- measure of pixel intensity differences in the eye region (*eyediff*)
- measure of pixel intensity differences in the nose / cheek region (*nosecheekdiff*)
- measure of comparative pixel intensity variability between two images of the same person (*variabilityratio*)

so we have a dataset $\{y_i, \underline{x}_i\}_{i=1,\ldots,N}, \underline{x}_i \in \mathbb{R}^3, y_i \in \{0,1\}$ with $N = 1000$ in our case.

We want to fit a binomial Generalized Linear Model (GLM) with complementary log-log link function to this data.

Let us, however, take a first look at the data and what standard logistic regression with parameters related linearly to the log-odds does to it in figure 1. We already see that the face-recognition tends to fail for more extreme values of the predictor variables, but also that the groups might be hard to separate.

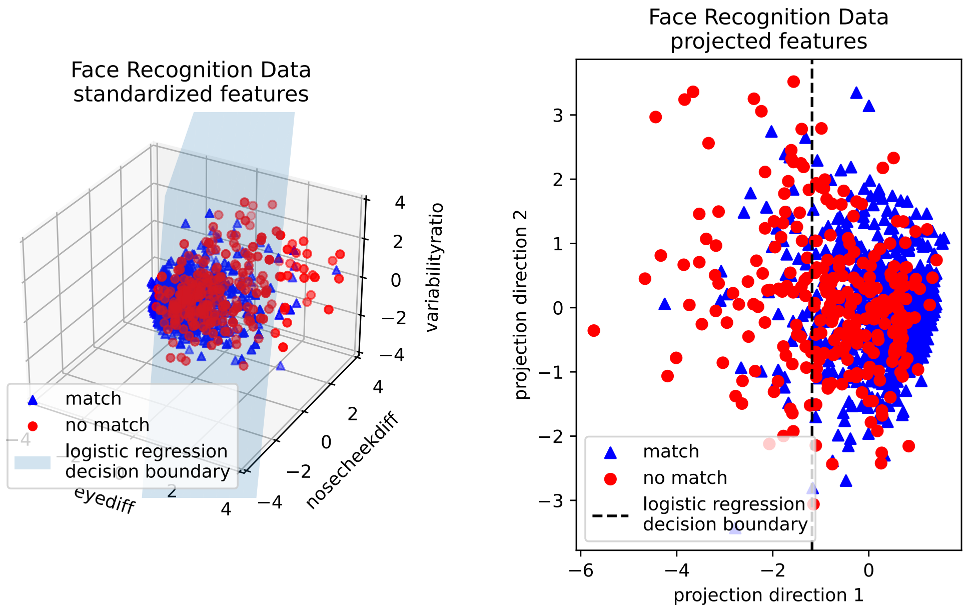

Figure 1: Standard logistic regression applied to the face recognition data. The projection on the right is to a hyperplane perpendicular to the decision boundary.

1.2 A short Introduction to GLMs

Repetition on Linear Regression

Remember that in a simple linear regression Model, we assume the underlying model to be

$$y(\underline{x}) = \underline{w}^T\underline{x} + \epsilon(\underline{x}), \quad \epsilon(\underline{x}) \sim \mathcal{N}(0, \sigma^2) \tag{1}$$

so linear in the parameters $\underline{w}$ and we assume a normal distribution of the noise $\epsilon(\underline{x})$. We can also write this as

$$p(y|\underline{x}, \underline{w}, \sigma^2) = \mathcal{N}(y|\underline{w}^T\underline{x}, \sigma^2), \quad E\left[y|\underline{x}, \underline{w}, \sigma^2\right] = \underline{w}^T\underline{x} \tag{2}$$

Naturally, such a model is not well suited for binary data or only positive data, ..., so we need to generalize this.

Generalized Linear Models

In the Generalized Linear Model, we assume a distribution which mean is modeled using

$$E\left[y|\underline{x}, \underline{w}, \sigma^2\right] = \mu = g^{-1}(\eta), \quad \eta = \underline{\beta}^T\underline{x} \tag{3}$$

where g is called the link function and g^{-1} the mean function. The distribution of which we model the mean is from the exponential family

$$\begin{gathered} p(y|\theta) = \exp\left(\frac{y\theta - b(\theta)}{a(\phi)} + c(y,\phi)\right), \\ \text{with real functions } a, b, c, \quad \text{and parameters } \theta, \phi \end{gathered} \tag{4}$$

for which holds

$$E[y] = \mu = \partial_\theta b(\theta) \tag{5}$$

connecting the parameter θ to the mean we model as

$$\mu = g^{-1}(\eta) = \partial_\theta b(\theta) \tag{6}$$

where for the canonical link function $\eta = \theta$.

Also, one can calculate the variance as

$$\mathrm{Var}[Y] = a(\phi)\partial_\theta^2 b(\theta) \tag{7}$$

Canonical Link Functions for a Binomial GLM

Consider the binomial distribution given by

$$\begin{aligned} f(y|p) &= p^y(1-p)^{1-y} \\ &= \exp\left(\underbrace{y\log\frac{p}{1-p}}_{\text{define }\theta} - [-\log(1-p)]\right) \\ &= \exp\left(y\theta - \log\left(1 + \frac{1}{1-p}\right)\right) \\ &= \exp\left(y\theta - \log(1+\exp\theta)\right) \\ &= \exp\left(y\theta - b(\theta)\right) \end{aligned} \tag{8}$$

with $a(\phi) = 1, c(y,\phi) = 0, b(\theta) = \log(1+\exp\theta)$ and $\theta = \log\frac{p}{1-p}$. From this we can find the well-known mean as

$$\mu = \partial_\theta b(\theta) = \frac{\exp\theta}{1+\exp\theta} = p \tag{9}$$

so for the canonical link ($\eta = \theta$) we have the mean function (a sigmoid)

$$g^{-1}(\eta) = \mu = \frac{\exp \eta}{1 + \exp \eta} \tag{10}$$

and thus the link function (log-odds)

$$g(\mu) = \eta = \log \frac{\mu}{1 - \mu} \tag{11}$$

Complementary Log-Log Link Function

Next to the canonical link for a binomial GLM, we can also use the complementary log-log link function (or e. g. the probit link function). The complementary log-log link function is given by

$$g(\mu) = \eta = \log\left(-\log\left(1 - \mu\right)\right) \tag{12}$$

and the mean function is

$$g^{-1}(\eta) = \mu = 1 - \exp\left(-\exp \eta\right) \tag{13}$$

which is common, when we observe and count events and observe either 0 (denoted by $y = 0$) or more events (denoted by $y = 1$). It exhibits asymmetry.

Maximum Likelihood Estimation for GLMs

Our aim is finding model parameters $\underline{\hat{\beta}}$ under which the data at hand, $\left\{y_i, \underline{x}_i\right\}_{i=1,\ldots,N}$ is most likely. We collect the response into a vector $\underline{Y} \in \mathbb{R}^N$ and the features into a feature matrix $\underline{\underline{X}} \in \mathbb{R}^{N \times p}$ (a column of 1s allows for an offset term in $\eta = \underline{\beta}^T \underline{x}$).

Finding $\underline{\hat{\beta}}$ amounts to maximizing the likelihood or equivalently maximizing the log-likelihood (as the logarithm is strictly monotonic) or minimizing the negative likelihood (a kind of loss). For this, iterative procedures are employed, consisting of a starting guess, a repeated update step and a condition on which to stop.

Below we present a simple scheme (there are more intricate possibilities for choosing the starting point and stop condition)

1. Start with $\underline{\beta}^{(0)} = \underline{0}$
2. Perform the update step with one of the following schemes

$$\underline{\beta}^{(k+1)}_{\text{gradient descent}} = \underline{\beta}^{(k)}_{\text{gradient descent}} - \gamma \underline{S}(\underline{\beta}^{(k)}_{\text{gradient descent}}) \tag{14}$$

$$\underline{\beta}^{(k+1)}_{\text{Newton-Rhapson}} = \underline{\beta}^{(k)}_{\text{Newton-Rhapson}} - \underline{\underline{H}}^{-1}(\underline{\beta}^{(k)}_{\text{Newton-Rhapson}}) \underline{S}(\underline{\beta}^{(k+1)}_{\text{Newton-Rhapson}}) \tag{15}$$

$$\underline{\beta}^{(k+1)}_{\text{Fisher Scoring}} = \underline{\beta}^{(k)}_{\text{Fisher Scoring}} + \underline{\underline{I}}^{-}1(\underline{\beta}^{(k)}_{\text{Fisher Scoring}})\underline{S}(\underline{\beta}^{(k)}_{\text{Fisher Scoring}}) \tag{16}$$

3. Repeat step 2., until $||\underline{\beta}^{(k+1)} - \underline{\beta}^{(k)}||_2^2 \leq \epsilon$ or better $|\mathbf{l}(\underline{\beta}^{(k+1)}) - \mathbf{l}(\underline{\beta}^{(k)})| \leq \epsilon$ and we break if a maximum number of steps is reached and set $\hat{\underline{\beta}}_{\text{MLE}} = \underline{\beta}^{(\text{last step})}$

where $\underline{S}(\underline{\beta})$ is the score-function with in the case of a GLM (for a derivation see e.g. the lecture slides)

$$\begin{aligned} \underline{S}(\underline{\beta}) &= \partial_{\underline{\beta}}\mathbf{l} = \underline{\underline{X}}^T\underline{\underline{A}}(\underline{Y} - \underline{\mu}) \\ \underline{\underline{A}} &= \text{diag}(A_1, \dots, A_N), \quad A_i = \frac{1}{\text{Var}[Y_i]}\partial_{\eta_i}\mu_i \\ \mu_i &= g^{-1}(\eta_i), \quad \eta_i = \underline{\beta}^T\underline{x}_i, \quad i = 1, \dots, N \end{aligned} \tag{17}$$

and $\underline{\underline{H}}(\underline{\beta})$ is the Hessian with

$$H_{jk}(\underline{\beta}) = \frac{\partial^2\mathbf{l}(\underline{\beta})}{\partial\beta_j\partial\beta_k} = \partial_{\beta_j}\sum_{i=1}^{N}\frac{y_i - \mu_i}{\text{Var}[y_i]}\frac{\partial\mu_i}{\partial\eta_i}x_{ki}, \quad j, k = 1, \dots, p \tag{18}$$

and the Fisher information matrix $\underline{\underline{I}}(\underline{\beta}) = -E[\underline{\underline{H}}(\underline{\beta})]$.

Maximum Likelihood Estimation for a Binomial GLM with Complementary Log-Log Link Function

To be able to perform the optimization schemes presented above and find $\hat{\underline{\beta}}_{\text{MLE}}$ we need to find expressions for $\underline{S}$, $\underline{\underline{H}}$ and $\underline{\underline{I}}$.

Log-Likelihood: In the case of the cloglog GLM (so the one with the complementary log-log link function), the log-likelihood is given by

$$\begin{aligned}
\mathbf{l}(\underline{\beta}) &= \log\left(\prod_{i=1}^{N} p(y_i|\underline{x}_i, \underline{\beta})\right) \\
&= \sum_{i=1}^{N} \log\big(p(y_i|\underline{x}_i, \underline{\beta})\big) \\
&= \sum_{i=1}^{N} y_i \log\big(g^{-1}(\eta_i)\big) + (1-y_i)\log\big(1-g^{-1}(\eta_i)\big) \\
&= \sum_{i=1}^{N} y_i \log\big(g^{-1}(\underline{\beta}^T\underline{x}_i)\big) + (1-y_i)\log\big(1-g^{-1}(\underline{\beta}^T\underline{x}_i)\big) \\
&= \sum_{i=1}^{N} y_i \log\big(1-\exp\big(-\exp\underline{\beta}^T\underline{x}_i\big)\big) + (1-y_i)\log\big(\exp\big(-\exp\underline{\beta}^T\underline{x}_i\big)\big) \\
&= \sum_{i=1}^{N} y_i \big(\log\big(1-\exp\big(-\exp\underline{\beta}^T\underline{x}_i\big)\big) - \log\big(\exp\big(-\exp\underline{\beta}^T\underline{x}_i\big)\big)\big) - \exp\underline{\beta}^T\underline{x}_i \\
&= \sum_{i=1}^{N} y_i \log\big(\exp\big(\exp\underline{\beta}^T\underline{x}_i\big)-1\big) - \exp\underline{\beta}^T\underline{x}_i
\end{aligned} \tag{19}$$

so the negative log-likelihood is

$$-\mathbf{l}(\underline{\beta}) = -\sum_{i=1}^{N} y_i \log\big(\exp\big(\exp\underline{\beta}^T\underline{x}_i\big)-1\big) - \exp\underline{\beta}^T\underline{x}_i \tag{20}$$

Note that using automatic differentiation, we can exactly (up to machine precision) calculate the gradient and Hessian of this function at given points without having to derive it by hand at very high efficiency (Baydin et al., 2018). We do the Newton-Raphson procedure in Julia as it lends itself well to automatic differentiation. We employ this to simple one-dimensional data, as shown in figure 2 and a plot of the loss function is given in figure 3.

Score: Let us start with the score $\underline{S}(\underline{\beta})$. The ingredients are

$$\begin{aligned}
&\eta_i = \underline{\beta}^T\underline{x}_i, \quad \mu_i = 1-\exp\left(-\exp\eta_i\right), \\
&\quad \mathrm{Var}[Y_i] = \mu_i(1-\mu_i), \quad \partial_{\eta_i}\mu_i = \exp\left(\eta_i - \exp\eta_i\right), \quad A_i = \frac{1}{\mathrm{Var}[Y_i]}\partial_{\eta_i}\mu_i
\end{aligned} \tag{21}$$

Based only on the score we can already perform gradient descent.

Hessian: The Hessian is given by (plug in and apply chain rule)

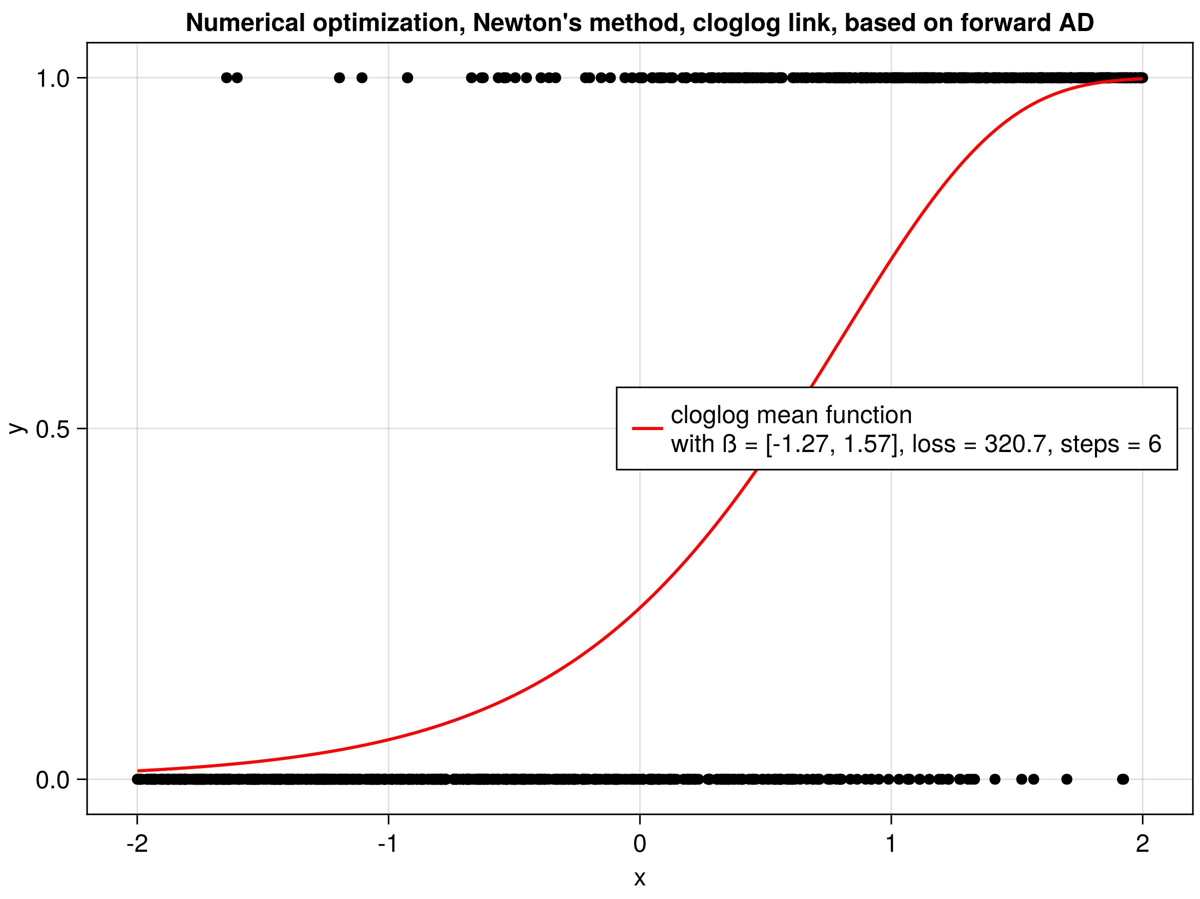

Figure 2: Newton-Raphson procedure for a cloglog link function on binomial responses and one feature.

$$
\begin{aligned}
H_{jk}(\underline{\beta}) &= \frac{\partial^2 \mathbf{l}(\underline{\beta})}{\partial \beta_j \partial \beta_k} = \partial_{\beta_j} \sum_{i=1}^{N} \frac{y_i - \mu_i}{\mathrm{Var}[y_i]} \frac{\partial \mu_i}{\partial \eta_i} x_{ki}, \quad j,k = 1,\dots,p \\
&= \partial_{\beta_j} \sum_{i=1}^{N} x_{ki} \frac{y_i - \left(1 - \exp\left(-\exp\left(\underline{\beta}^T \underline{x}_i\right)\right)\right)}{\left(1 - \exp\left(-\exp\left(\underline{\beta}^T \underline{x}_i\right)\right)\right)\left(\exp\left(-\exp\left(\underline{\beta}^T \underline{x}\right)\right)\right)} \exp\left(\underline{\beta}^T \underline{x} - \exp\left(\underline{\beta}^T \underline{x}\right)\right) \\
&= \partial_{\beta_j} \sum_{i=1}^{N} x_{ki} \left(\frac{y_i}{1 - \exp\left(-\exp\left(\underline{\beta}^T \underline{x}_i\right)\right)} - 1 \right) \exp\left(\underline{\beta}^T \underline{x}_i\right) \\
&= \sum_{i=1}^{N} x_{ki} \exp\left(\underline{\beta}^T \underline{x}_i\right) \left\{ \left(\frac{y_i}{1 - \exp\left(-\exp\left(\underline{\beta}^T \underline{x}_i\right)\right)} - 1 \right) x_{ji} + \right. \\
& \left. y_i \left(1 - \exp\left(-\exp\left(\underline{\beta}^T \underline{x}_i\right)\right)\right)^{-2} \exp\left(\underline{\beta}^T \underline{x}_i\right) x_{ij} \exp\left(-\exp\left(\underline{\beta}^T \underline{x}\right)\right) \right\}
\end{aligned} \tag{22}
$$

As of time constraints, we will not implement the expression derived here, but rather use

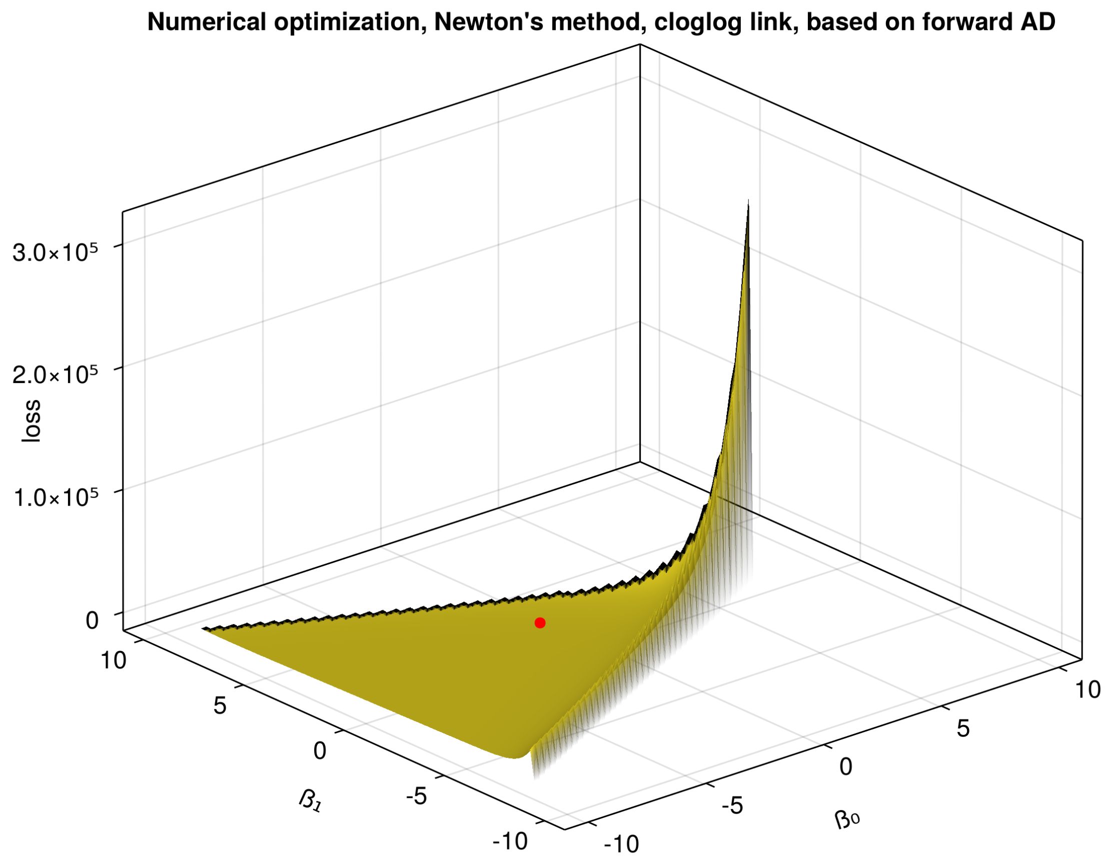

Figure 3: Loss function for a cloglog link function on binomial responses and one feature. Note that the mean function tends to have very high values, which can in more extreme cases lead to numerical problems.

automatic differentiation in Julia. We will also not implement the Fisher information matrix (which was announced to be Ok to leave away).

1.3 Finding the MLE for a Binomial GLM with Complementary Log-Log Link Function (task a, b, c, d)

Based on the negative log-likelihood found for the GLM with cloglog-link

$$-\mathbf{l}(\underline{\beta}) = -\sum_{i=1}^{N} y_i \log\left(\exp\left(\exp \underline{\beta}^T \underline{x}_i\right) - 1\right) - \exp \underline{\beta}^T \underline{x}_i \tag{23}$$

we employ the Newton-Raphson procedure as described (the update steps have also been described), which is implemented in Code-Snippet 1. Rather than implementing the found expression by hand (possible, but tedious), we use automatic differentiation.

The results compared to the ones of the standard GLM function in R (code-snippet 2) are given in table 1. The coefficients are very similar, while we yield a slightly lower loss but take 12 ms longer to execute, which seems to be a very good result. The convergence is very fast, as we can see in figure 4. We have also implemented the IRWLS (iteratively reweighted least squares) algorithm, as given on the lecture slides, which can be found in code-snippet 3. The results are also given in table 1.

	β_0	β_1	β_2	β_3	loss	execution-time
Our Newton-Raphson	0.5379	-6.0133	-7.7950	0.6695	469.72	$\sim$ 17 ms
Our IRWLS	0.2833	-5.4183	-8.0693	0.8854	470.16	$\sim$ 22 ms
R version	0.3863	-5.6838	-8.0727	0.7992	469.97	$\sim$ 5 ms

Table 1: Results of the our Newton-Raphson procedure, GLM in R and the IRWLS procedure for a cloglog link function on binomial responses and three features. Benchmarking was done using BenchmarkTools.jl in Julia and microbenchmark in R.

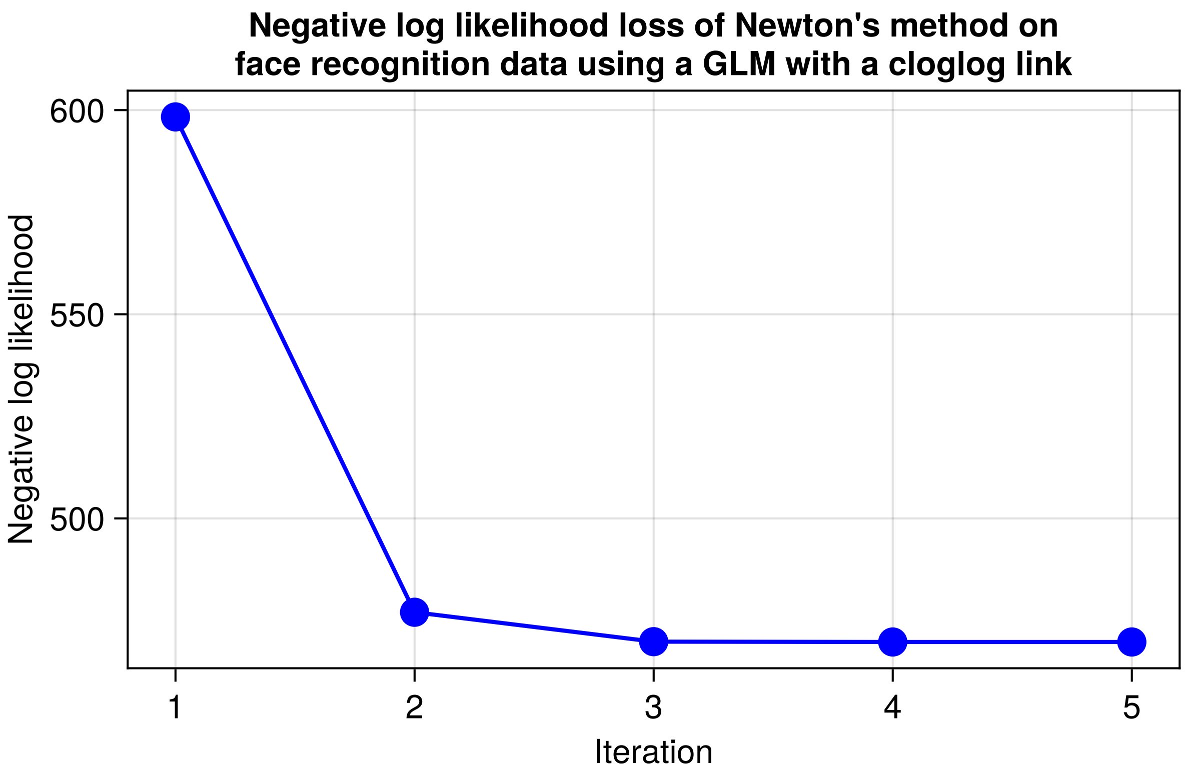

Figure 4: Loss function for the Newton-Raphson procedure for a GLM with cloglog link as a function of the iteration for data on a face recognition algorithm.

```julia
function loss(theta, X, Y)
    """Implementation of negative log likelihood loss for a cloglog
    ↪ link"""
    return -(sum(Y .* log.(exp.(exp.(X * theta)) .- 1) .- exp.(X *
    ↪ theta)))
end

function loss_gradient(theta, X, Y)
    """Gradient of the logistic loss"""
    return ForwardDiff.gradient(theta -> loss(theta, X, Y), theta)
end

function loss_hessian(theta, X, Y)
    """Hessian of the logistic loss"""
    return ForwardDiff.hessian(theta -> loss(theta, X, Y), theta)
end

function newton(theta, x, y; maxiter = 100, tol = 1e-4) # also gets
↪ stuck at saddle points and maxima
    """Newton's method for logistic regression"""
    # initialize the theta, we work on
    theta = copy(theta)
    # initialize the loss list (negative log likelihood)
    loss_list::Vector{Float64} = [loss(theta, x, y)]
    # initialize the number of iterations
    num_iterations = maxiter
    for i in 1:maxiter
        # Newton-Raphson update-step
        theta_new = theta - loss_hessian(theta, x, y) \
        ↪ loss_gradient(theta, x, y)
        # append the loss
        push!(loss_list, loss(theta_new, x, y))
        # check for convergence
        if abs(loss(theta_new, x, y) - loss(theta, x, y)) < tol
            theta = theta_new
            num_iterations = i
            break
        end
        theta = theta_new
    end
    return theta, num_iterations, loss_list
end

X, Y = get_face_rec_XY() # function that returns the design matrix
↪ (1000 x 4) and the response vector
theta_0 = [0.0, 0.0, 0.0, 0.0]
theta, num_iterations, loss_list = newton(theta_0, X, Y)
```

Code-Snippet 1: Newton-Raphson procedure in Julia for a cloglog link function.

```r
set.seed(1797)

# Load data
file <- readxl::read_excel("data/facerecognition.xlsx")
data <- file[sample(1:nrow(file), 1000),]

match <- data$match
eyediff <- data$eyediff
nosecheekdiff <- data$nosecheekdiff
variabilityratio <- data$variabilityratio

Y <- match
X <- cbind(eyediff, nosecheekdiff, variabilityratio)

# fit binomial GLM with cloglog link
mod <- glm(Y ~ X, family = binomial(link = "cloglog"))
print(summary(mod))

# benchmarking
library(microbenchmark)
print(microbenchmark(glm(Y ~ X, family = binomial(link =
 ↪  "cloglog"))))
```

Code-Snippet 2: R code for fitting a binomial GLM with cloglog link function to the face recognition data.

```r
# implementation based on slide 46 in lecture 7
irwls_cloglog <- function(X, Y, tol = 1e-6, maxit = 1000) {
    # allow for intercept
    X <- as.matrix(cbind(1, X))
    # initialize coefficients
    beta <- rep(0, ncol(X))  # initial coefficients (zeros)

    # IRWLS algorithm
    for (i in 1:maxit) {
        # linear predictors
        eta <- X %*% beta
        # mean function of cloglog link
        # inverse link
        mu <- 1 - exp(-exp(eta))
        # binomial variance
        var_mu <- mu * (1 - mu)
        # partial_eta mu
        partial_eta_mu <- exp(eta - exp(eta))
        # working weight
        W <- 1 / var_mu * partial_eta_mu^2
        # adjusted response variable
        z <- eta + (Y - mu) / W

        # Fitting weighted linear least squares model
        # 0 + as we handle the intercept manually
        fit <- lm(z ~ 0 + X, weights = W)
        beta_new <- coef(fit)

        # Check convergence
        if (sqrt(sum((beta - beta_new)^2)) < tol) {
            # update coefficients
            beta <- beta_new
            break
        }
        # update coefficients
        beta <- beta_new
    }
    return(beta)
}
```

Code-Snippet 3: R code for fitting a binomial GLM with cloglog link function to the face recognition data using IRWLS.

1.4 Wald-Test on the MLE coefficients (task e)

On a single parameter, the Wald statistic takes the form

$$W = \frac{\hat{\beta}_j - \beta_{j,0}}{\sqrt{\hat{\text{Var}}[\hat{\beta}_j]}} \tag{24}$$

which under the null hypothesis $H_0 : \beta_j = \beta_{j,0} = 0$ is distributed as ξ_1^2 (chi-squared distribution with one degree of freedom). We test this for all coefficients found by the R function `glm` at $\alpha = 0.05$. The results are given in table 2, the code can be found in code-snippet 4. We can see that the coefficients for the intercept and the variability are not significant, while the coefficients for the eye and nosecheek are significant.

	β_0 (*intercept*)	β_1 (*eye*)	β_2 (*nosecheek*)	β_3 (*variability*)
MLE estimates	0.3863	-5.6838	-8.0727	0.7992
Standard Errors	0.6834	1.0847	0.7994	0.6825
Wald statistic	0.3195	27.4576	101.9798	1.3714
p-value	57.1%	$1.6 \cdot 10^{-5}$%	0	24.16%
Reject H_0?	No	Yes	Yes	No

Table 2: Results of the Wald-Test on the MLE parameters.

```r
# fit binomial GLM with cloglog link
mod <- glm(Y ~ X, family = binomial(link = "cloglog"))
print(summary(mod))

# Function to perform Wald test
wald_test <- function(model, alpha = 0.05) {
    # Extracting coefficients and their standard errors
    coefs <- summary(model)$coefficients
    beta <- coefs[, "Estimate"]
    se_beta <- coefs[, "Std. Error"]

    # Calculating Wald statistic
    wald_stat <- (beta / se_beta)^2

    # Degrees of freedom (1 for each parameter)
    df <- 1

    # Calculating p-values
    p_values <- 1 - pchisq(wald_stat, df)

    # Results
    result <- data.frame(Estimate = beta, StdError = se_beta,
     ↪  WaldStatistic = wald_stat, PValue = p_values)
    return(result)
}

# Performing Wald test on the model
wald_test_results <- wald_test(mod)
print(wald_test_results)
```

Code-Snippet 4: R code for performing a Wald-Test on the MLE parameters.

References

Baydin, Atilim Gunes et al. (2018). »Automatic differentiation in machine learning: a survey«. In: *Journal of Marchine Learning Research* 18, pages 1–43.