In this brief monograph I recall the Newton-Raphson algorithm for the derivation of MLE estimates in a logistic regression model, and describe its extension to a Bayesian model for maximum *a posteriori* (MAP) estimates of beta parameters.

The logistic model is generally constructed as follows. There is observed vector Y of size n with each component $Y_i \sim \text{Bern}(p_i)$, and for each Y_i there is an associated vector of observed covariates $x = \{1, x_1, ..., x_p\}$. The logistic regression model assumes that p_i has link function $g(\cdot)$ such that the link of the expectation of Y is a linear function of the predictors:

$$g(\mathbb{E}[Y_i|p_i]) \quad = \quad \text{logit}(p_i) \quad = \quad \log\left(\frac{p_i}{1-p_i}\right) \quad =$$

$$\beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip} \quad = \quad \boldsymbol{x}_i^T \boldsymbol{\beta},$$

Since the covariates are all known, observe that p_i is a function of $\boldsymbol{\beta}$. This model lends itself to a nice interpretation, as the model coefficients are interpreted as the log odds ratio between conditions, or for a one unit increase in a covariate. Thus, $\exp(\beta)$ is an odds ratio between conditions, controlling for the other covariates.

In general Newton's iterative root-finding method is

$$x_{t+1} = x_t - \frac{f(x_t)}{f'(x_t)}$$

The same derivation follows for multivariate functions (such as likelihoods), and for the derivative of the log likelihood of the logistic model, the iteration rule for the argument which maximizes the log likelihood becomes

$$\boldsymbol{\beta}_{t+1} \quad = \quad \boldsymbol{\beta}_t \; + \; \ell''(\boldsymbol{\beta})^{-1} \; \ell'(\boldsymbol{\beta}) \quad = \quad \boldsymbol{\beta}_t \; + \; \left(\boldsymbol{X}^T \boldsymbol{W}_t \boldsymbol{X}\right)^{-1} \left[\boldsymbol{X}^T(\boldsymbol{y} - p(\boldsymbol{\beta}_t))\right],$$

where $\ell''(\boldsymbol{\beta})^{-1}$ is the inverse of the Fisher Information matrix, and $\ell'(\boldsymbol{\beta})$ is the score function, i.e. the matrix of first partial derivatives. $\boldsymbol{W}$ is a diagonal matrix of terms $p_i(\boldsymbol{\beta}_t)(1 - p_i(\boldsymbol{\beta}_t))$.

To extend this framework to the maximum *a posteriori* estimates in a Bayesian model, I note that the posterior density is proportional to a product of the likelihood and the prior density. Thus, under differentiation the normalizing constant vanishes, and the logarithm of the product reduces to a sum:

$$\ell(P(\boldsymbol{\beta}|Y)) \propto \ell(P(Y|\boldsymbol{\beta}) \times P(\boldsymbol{\beta})) = \log P(Y|\boldsymbol{\beta}) + \log P(\boldsymbol{\beta})$$

Assuming a multivariate normal distribution for $\boldsymbol{\beta}$ with mean vector $\boldsymbol{\beta}_0$ and covariance matrix $\boldsymbol{\Sigma}_0$, differentiation of the log likelihood of the posterior by $\boldsymbol{\beta}$ yields

$$\frac{\partial}{\partial \boldsymbol{\beta}} \ell(P(Y|\boldsymbol{\beta}) \times P(\boldsymbol{\beta})) \; = \boldsymbol{X}^T(\boldsymbol{y} - p(\boldsymbol{\beta})) \; - \; \boldsymbol{\Sigma}_0^{-1}(\boldsymbol{\beta}_0 - \boldsymbol{\beta}),$$

as in general the derivative of a MVN density $\varphi(\cdot)$ with mean $\boldsymbol{m}$, covariance matrix $\boldsymbol{\Sigma}$ is $\partial\varphi(\boldsymbol{x})/\partial\boldsymbol{x} = -\varphi(\boldsymbol{x})\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{m})$. The second derivative is $\partial^2\varphi(\boldsymbol{x})/\partial^2\boldsymbol{x} = \varphi(\boldsymbol{x})\left(\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{m})(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{\Sigma}^{-1} - \boldsymbol{\Sigma}^{-1}\right)$.

Under the logarithm the second derivative of the MVN density is just $-\boldsymbol{\Sigma}^{-1}$. Thus, the Newton-Raphson algorithm for the MAP estimators of a logistic model is modified in each component with a term related to the prior:

$$\boldsymbol{\beta}_{t+1} \;=\; \boldsymbol{\beta}_t \;+\; \left(\boldsymbol{X}^T\boldsymbol{W}_t\boldsymbol{X} + \underbrace{\boldsymbol{\Sigma}^{-1}}\right)^{-1}\left[\boldsymbol{X}^T(\boldsymbol{y} - p(\boldsymbol{\beta}_t)) - \underbrace{\boldsymbol{\Sigma}^{-1}(\boldsymbol{\beta}_0 - \boldsymbol{\beta})}\right]$$

A sample `R` function for the method's implementation follows. It takes as arguments a vector of binary outcome values, a design matrix with intercept column, and a prior mean vector b0 and prior covariance matrix S.

```
NRmap <- function(y,X,b0,S,start=1,tol=1e-6) {
    B <-  matrix(NA,ncol=1,nrow=ncol(X)) #matrix to store betas
    B[,1] <- rep(start, ncol(X)) #starting values
    i<-2
     repeat {
        p <- plogis(X %*% B[,i-1])
        W <- diag(c(p*(1-p)))
        S.inv <-solve(S)
        score <- crossprod(X,(y - p))  - S.inv%*%(as.matrix(B[,i-1]-b0))
        increm <- solve(crossprod(X,W)%*%X + S.inv)
        B <- cbind(B,B[,i-1]+increm%*%score)
        if (all(abs(B[,i]-B[,i-1]) < tol)) break
        if (i>300) stop("Failure to find root after 300 iterations.
        Attempt different starting value.")
         i<-i+1
    }
   list(beta=t(B)[nrow(t(B)),], iterations=i-1)
}
```