Tail correction in the 'occupancy' family

H. Rue

October 4, 2024

This is a short note to document the tail correction implemented in the *occupancy* family. The likelihood has the following form

$$f(y) = \phi \prod_{i=1}^{m} p_i^{y_i} (1-p_i)^{1-y_i} + (1-\phi) 1_{[y_i = 0, \forall i]}$$

where $\{p_i\}$ are determined by covariates (hence we can treat it as constants in this discussion), and ϕ is determined by the linear predictor η

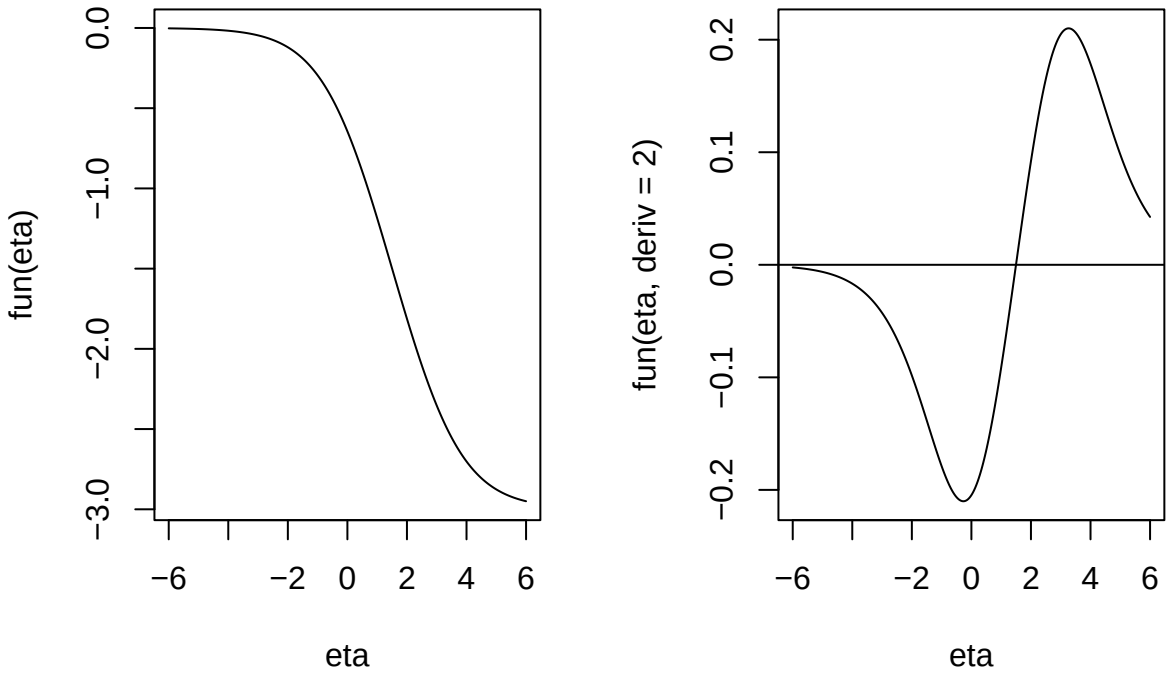
$$\text{logit}(\phi) = \eta.$$

The case "$y_i = 0, \forall i$'' cause an issue in the right tail, as a function of η, which is the purpose of this note. We can simplify the log-likelihood in this case as

$$l(\eta) = \log\left(a\phi + (1-\phi)\right)$$

for $a = \prod_{i=1}^{m}(1-p_i)$. The issue displays itself when displaying $l(\eta)$ and its Hessian/second derivative, $l''(\eta)$

```
ll <- function(eta, a=0.5) {
    phi <- inla.link.invlogit(eta)
    return (log(a * phi + (1-phi)))
}

eta <- seq(-6, 6, by=0.005)
a <- 0.05
par(mfrow=c(1,2))
fun <- splinefun(eta, ll(eta,a))
plot(eta, fun(eta), type="l", lwd=1)
plot(eta, fun(eta, deriv=2), type="l", lwd=1)
abline(h=0)
```



The main issue is the positive Hessian which indicate a negative local variance/precision in the Taylor expansion. This happens in the right tail of the log-likelihood, which is also the low-likelihood part hence not of particular interest. The *Hessian has the wrong sign*-issue would normally require some modifications like truncating it at zero, or requesting support from higher forces that these Hessians would not destroy the inference.

We can work out the point where $l''(\eta^*) = 0$, which is

$$\eta^* = -\frac{1}{2}\log(a)$$

and 1.498 in our example.

A trunctation of the Hessian at 0 will cause a an abrupt change that is no good for the INLA numerics:

- We loose curvature information which is critical for the inner-optimisation using Newthon-Raphson
- We introduce a singular point in the Hessian
- This also raise the question about what to do with the local gradient information, if we should keep it (the default choice `control.inla=list(b.strategy="keep")`) or should simply ignore the likelihood contribution (`control.inla=list(b.strategy="skip")`).

Whatever we do, we will likely loose the quadratic convergence property of the inner optimisation, and the linear quadratic one will be much slower in the sense of requiring many more function evaluations. (This turn out to be a pretty serious issue in terms of running-time.)

However, we can contruct a smooth truncation as follows. Define the interval as a function of the critical value η^*,

$$\eta_L = 0.9\eta^*, \qquad \text{and} \qquad \eta_H = 0.999\eta^*$$

We can express the 2nd order expansion around η'=x0 as

$$l(\eta) \approx c_0(\eta') + c_1(\eta')(\eta - \eta') - \frac{1}{2}c_2(\eta')(\eta - \eta')^2$$

where

$$c_0(\eta') = \log\left(\frac{a\,\mathrm{e}^{x0} + 1}{\mathrm{e}^{x0} + 1}\right)$$

$$c_1(\eta') = \frac{\mathrm{e}^{x0}\,(a-1)}{(\mathrm{e}^{x0}+1)\,(a\,\mathrm{e}^{x0}+1)}$$

$$c_2(\eta') = \frac{\left(-a\,\mathrm{e}^{2\,x0} + \mathrm{e}^{2\,x0}a^2 - a + 1\right)\mathrm{e}^{x0}}{(\mathrm{e}^{x0}+1)^2\,(a\,\mathrm{e}^{x0}+1)^2}$$

The idea is to replace, locally, the log-likelihood with this expansion, *evaluated* within the interval $[\eta_L, \eta_H]$. Hence we need to map the interval $[\eta_L, \infty]$ into $[\eta_L, \eta_H]$, which we do with $(\eta \geq \eta_L)$

$$\tilde{\eta} = \eta_L + (\eta_H - \eta_L)\frac{z}{1+z}, \qquad z = \frac{\eta - \eta_L}{\eta_H - \eta_L}$$

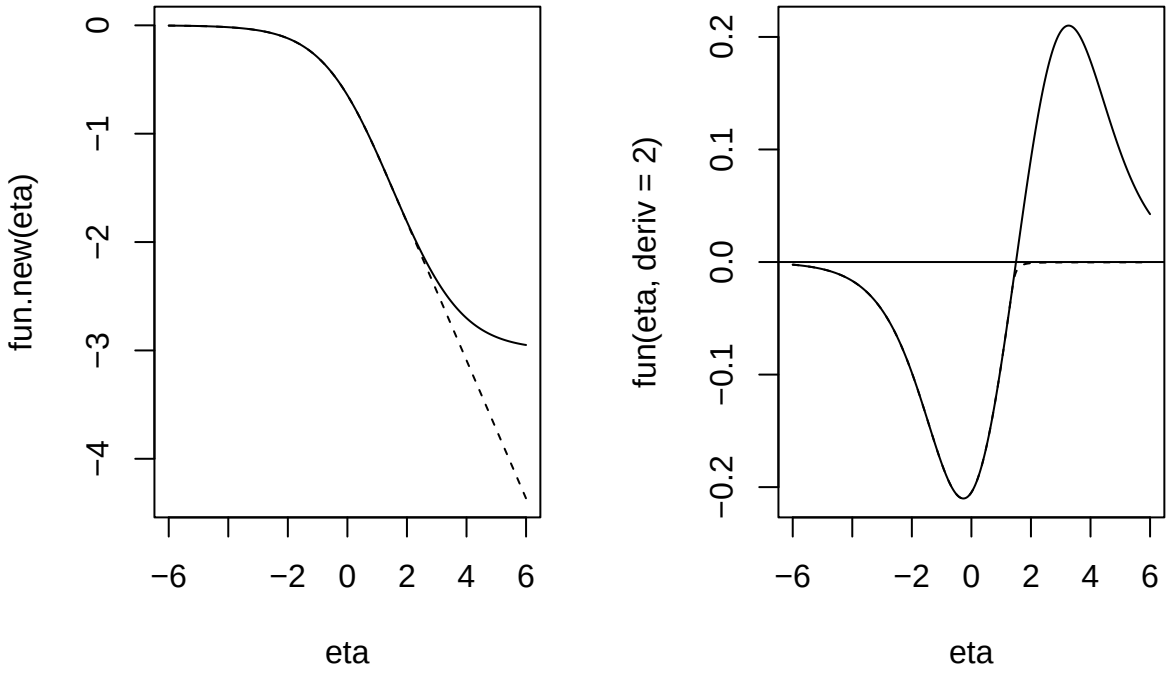
This is implemented as

```
ll.new <- function(eta, a=0.5) {

    c0 <- function(a, x0) {
        ex0 <- exp(x0)
        return (log((a*ex0+1) / (ex0 + 1)))
    }
    c1 <- function(a, x0) {
        ex0 <- exp(x0)
        return ((ex0 * (a-1)) / ((ex0+1)*(a*ex0+1)))
    }
    c2 <- function(a, x0) {
        ex0 <- exp(x0)
        return (((-a*ex0^2 + ex0^2 * a^2 - a + 1) * ex0) /
                ((ex0+1)^2 * (a*ex0+1)^2))
    }

    eta.c <- -0.5 * log(a)
    eta.L <- 0.90 * eta.c
    eta.H <- 0.999 * eta.c

    phi <- inla.link.invlogit(eta)
    ll <- log(a * phi + (1-phi))

    ## correct
    idx <- which(eta >= eta.L)
    if (length(idx) > 0) {
        x <- eta[idx]
        ##      xx <- eta.L + (eta.H -eta.L) * (1-exp(-sqrt(x-eta.L)))
        ee <- (x-eta.L)/(eta.H - eta.L)
        xx <- eta.L + (eta.H -eta.L) * ee/(1+ee)
        ll.fix <- (c0(a,xx) + c1(a,xx) * (x-xx)
            - 0.5 * c2(a,xx) * (x-xx)^2)
        ll[idx] <- ll.fix
    }
```

```
    return (ll)
}
```

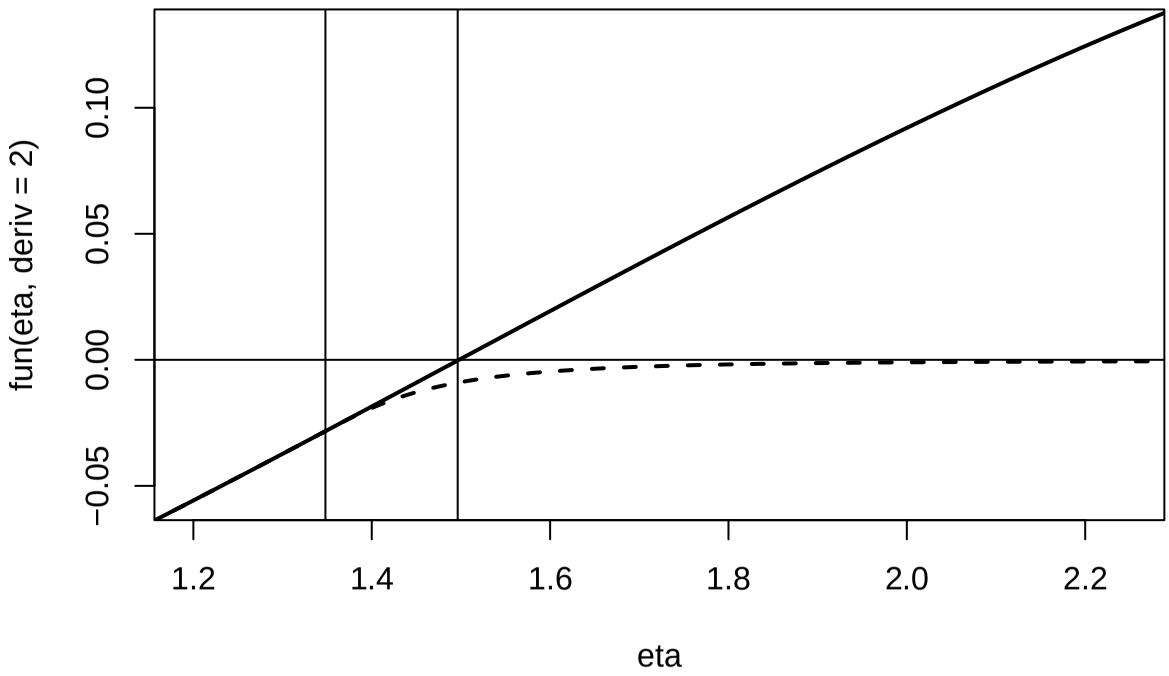
and we can display the changes reusing the previous example, where the *corrected* log-likelihood and its Hessian, are shown with dashed lines

```
par(mfrow=c(1,2))
fun.new <- splinefun(eta, ll.new(eta,a))
plot(eta, fun.new(eta), type="l", lwd=1,lty=2)
lines(eta, fun(eta), type="l", lwd=1,lty=1)
plot(eta, fun(eta, deriv=2), type="l", lwd=1)
lines(eta, fun.new(eta, deriv=2), type="l", lwd=1, lty=2)
abline(h=0)
```



We can magnify the critical region to see better the Hessian of the corrected log-likelihood

```
xc <- -0.5 * log(a)
plot(eta, fun(eta, deriv=2), type="l", lwd=2,
        xlim=c(0.8*xc,1.5*xc),
        ylim=c(fun(0.8*xc,deriv=2), fun(1.5*xc,deriv=2)))
lines(eta, fun.new(eta, deriv=2), type="l", lwd=2, lty=2)
abline(h=0)
abline(v=0.90 * xc)
abline(v=0.999 * xc)
```


fun(eta, deriv = 2)
−0.05
0.00
0.05
0.10
1.2
1.4
1.6
1.8
2.0
2.2
eta