Computing “quasi-AIC” (QAIC), in R is a minor pain, because the R Core team (or at least the ones who wrote \texttt{glm}, \texttt{glmmPQL}, etc.) are purists and don’t believe that quasi- models should report a likelihood. As far as I know, there are three R packages that compute/handle QAIC: \texttt{bbmle}, \texttt{AICcmodavg} and \texttt{MuMIn}.

The basic problem is that quasi- model fits with \texttt{glm} return an \texttt{NA} for the log-likelihood, while the dispersion parameter (\(c\), \(\phi\), whatever you want to call it) is only reported for quasi- models. Various ways to get around this are:

- fit the model twice, once with a regular likelihood model (\texttt{family=binomial}, \texttt{poisson}, etc.) and once with the quasi- variant — extract the log-likelihood from the former and the dispersion parameter from the latter

- only fit the regular model; extract the overdispersion parameter manually with

\begin{verbatim}
dfun <- function(object) {
  with(object,sum((weights * residuals^2)[weights > 0])/df.residual)
}
\end{verbatim}

- use the fact that quasi- fits still contain a deviance, even if they set the log-likelihood to \texttt{NA}. The deviance is twice the negative log-likelihood (it’s offset by some constant which I haven’t figured out yet, but it should still work fine for model comparisons)

The whole problem is worse for \texttt{MASS::glmmPQL}, where (1) the authors have gone to greater efforts to make sure that the (quasi-)deviance is no longer preserved anywhere in the fitted model, and (2) they may have done it for good reason — it is not clear whether the number that would get left in the ‘deviance’ slot at the end of \texttt{glmmPQL}'s alternating \texttt{lme} and \texttt{glm} fits is even meaningful to the extent that regular QAICs are. (For discussion of a similar situation, see the \texttt{WARNING} section of \texttt{?gamm} in the \texttt{mgcv} package.)

Example: use the values from one of the examples in \texttt{?glm}:
counts <- c(18,17,15,20,10,20,25,13,12)
outcome <- gl(3,1,9)
treatment <- gl(3,3)

Fit Poisson and quasi-Poisson models with all combinations of predictors:

```
glmOT.D93 <- glm(counts ~ outcome + treatment, family=poisson)
glmO.D93  <- update(glmOT.D93, . ~ . - treatment)
glmT.D93  <- update(glmOT.D93, . ~ . - outcome)
glmX.D93  <- update(glmT.D93, . ~ . - treatment)
glmQOT.D93 <- update(glmOT.D93, family=quasipoisson)
glmQO.D93 <- update(glmO.D93, family=quasipoisson)
glmQT.D93 <- update(glmT.D93, family=quasipoisson)
glmQX.D93 <- update(glmX.D93, family=quasipoisson)
```

Extract log-likelihoods:

```
(sum(dpois(counts, 
    lambda=exp(predict(glmOT.D93)),log=TRUE))) ## by hand
```

```
[1] -23.38066
```

```
(logLik(glmOT.D93)) ## from Poisson fit
```

```
'log Lik.' -23.38066 (df=5)
```

The deviance (`deviance(glmOT.D93)`=5.129) is not the same as $-2L (-2 \log \text{Lik}(\text{glmOT.D93}) = 46.761)$, but the calculated differences in deviance are consistent, and are also extractable from the quasi- fit even though the log-likelihood is NA:

```
(-2*(logLik(glmT.D93)-logLik(glmOT.D93))) ## Poisson fit
```

```
'log Lik.' 5.452305 (df=3)
```

```
(deviance(glmT.D93)-deviance(glmOT.D93)) ## Poisson fit
```

```
[1] 5.452305
```

```
(deviance(glmQT.D93)-deviance(glmQOT.D93)) ## quasi-fit
```

```
[1] 5.452305
```

Compare hand-computed dispersion (in two ways) with the dispersion computed by `summary.glm()` on a quasi- fit:
## 

1.2933

\[
\frac{\text{sum}(\text{residuals}(\text{glmOT.D93}, \text{"pearson"})^2)}{\text{glmOT.D93}$\text{df.residual}}
\]

## 

1

\[
\text{summary}(\text{glmQOT.D93}$\text{dispersion})
\]

### Examples

bbmle

```r
library(bbmle)
(qAIC(glmOT.D93, dispersion=dfun(glmOT.D93)))
## [1] 46.15658

(qAICc(glmOT.D93, dispersion=dfun(glmOT.D93), nobs=length(counts)))
## [1] 90.15658

ICtab(glmOT.D93, glmT.D93, glmO.D93, glmX.D93, 
   dispersion=dfun(glmOT.D93), type="qAIC")
## dqAIC df
## glmO.D93 0.0 3
## glmX.D93 0.2 1
## glmOT.D93 4.0 5
## glmT.D93 4.2 3

ICtab(glmOT.D93, glmT.D93, glmO.D93, glmX.D93, 
   dispersion=dfun(glmOT.D93), nobs=length(counts), type="qAICc")
## dqAICc df
## glmX.D93 0.0 1
## glmO.D93 7.8 3
## glmT.D93 12.0 3
## glmOT.D93 43.8 5

detach("package:bbmle")
```
AICcmodavg

```r
library(AICcmodavg)
aictab(list(glmOT.D93,glmT.D93,glmO.D93,glmX.D93),
       modnames=c("OT","T","O","X"),
       c.hat=dfun(glmOT.D93))
```

```
##
## Model selection based on QAICc:
## (c-hat estimate = 1.2933)
##
## K QAICc Delta_QAICc QAICcWt Cum.Wt Quasi.LL
## X 2 46.37 0.00 0.98 0.98 -20.19
## O 4 54.16 7.78 0.02 1.00 -18.08
## T 4 58.37 12.00 0.00 1.00 -20.19
## OT 6 90.16 43.78 0.00 1.00 -18.08

detach("package:AICcmodavg")
```

MuMIn

```r
library(MuMIn); packageVersion("MuMIn")
```

```
## [1] '1.40.0'

## from ?QAIC
x.quasipoisson <- function(...) {
  res <- quasipoisson(...)
  res$aic <- poisson(...)$aic
  res
}
glmQOT2.D93 <- update(glmOT.D93,family="x.quasipoisson",
                       na.action=na.fail)
(gg <- dredge(glmQOT2.D93,rank="QAIC", chat=dfun(glmOT.D93)))
```

```
## Fixed term is "(Intercept)"

## Global model call: glm(formula = counts ~ outcome + treatment, family = "x.quasipoisson",
##                        na.action = na.fail)
## ---
## Model selection table
## (Intrc) outcm trtmn df logLik QAIC delta weight
## 2 3.045 + 3 -23.381 44.2 0.00 0.464
## 1 2.813 1 -26.107 44.4 0.22 0.417
```
## 4 3.045 + + 5 -23.381 48.2 4.00 0.063
## 3 2.813 + 3 -26.107 48.4 4.22 0.056

## Models ranked by QAIC(x, chat = 1.29330044006699)

(ggc <- dredge(glmQOT2.D93, rank="QAICc", chat=dfun(glmOT.D93)))

## Fixed term is "(Intercept)"

## Global model call: glm(formula = counts ~ outcome + treatment, family = "x.quasipoisson",
## na.action = na.fail)

## ---

## Model selection table

## (Intrc) outcm trtmn df logLik QAICc delta weight
## 1 2.813 1 -26.107 46.4 0.00 0.978
## 2 3.045 + 3 -23.381 54.2 7.78 0.020
## 3 2.813 + 3 -26.107 58.4 12.00 0.002
## 4 3.045 + + 5 -23.381 90.2 43.78 0.000

## Models ranked by QAICc(x, chat = 1.29330044006699)

detach("package:MuMIn")

Notes: ICtab only gives delta-IC, limited decimal places (on purpose, but how do you change these defaults if you want to?). Need to add 1 to parameters to account for scale parameter. When doing corrected-IC you need to get the absolute number of parameters right, not just the relative number . . . Not sure which classes of models each of these will handle (lm, glm, (n)lme, lme4, mle2 . . .). Remember need to use overdispersion parameter from most complex model. glmmPQL: needs to be hacked somewhat more severely (does not contain deviance element, logLik has been NA’d out).

<table>
<thead>
<tr>
<th>package</th>
<th>lm</th>
<th>glm</th>
<th>(n)lme</th>
<th>multinom</th>
<th>polr</th>
<th>lme4</th>
<th>mle2</th>
</tr>
</thead>
<tbody>
<tr>
<td>AICcmodavg</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>y</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>