Maximum likelihood estimation and analysis with the bbmle package

Ben Bolker

February 14, 2007

```
> library(bbmle)
```

Get data from Crowder 1978 [?], as incorporated in the aod package:

```
> library(aod)
> data(orob1)
```

Implement beta-binomial distribution:

```
> dbetabinom <- function(x, mu, size, theta, log = FALSE) {
+     v <- lchoose(size, x) - lbeta(theta * (1 - mu), theta * mu) +
+         lbeta(size - x + theta * (1 - mu), x + theta * mu)
+     if (log)
+         v
+     else exp(v)
+ }
> rbetabinom <- function(n, mu, size, theta) {
+     a <- theta * mu
+     b <- theta * (1 - mu)
+     rbinom(n, size = size, prob = rbeta(n, a, b))
+ }
```

Generate random deviates from a random beta-binomial:

```
> x1 = rbetabinom(n = 1000, mu = 0.1, size = 50, theta = 10)
```

Simple likelihood model:

```
> mtmp <- function(mu, size, theta) {
+     -sum(dbetabinom(x1, mu, size, theta, log = TRUE))
+ }

> m0 <- mle2(mtmp, start = list(mu = 0.2, theta = 9), data = list(size = 50))
> p0 <- profile(m0)
> confint(p0)
> confint(m0, method = "quad")
```

Works well. The curvature-based confidence limits are very close to the profile confidence limits.

Alternatively, using the formula interface:

```
> m0f <- mle2(x1 ~ dbetabinom(mu, size = 50, theta), start = list(mu = 0.2,
+     theta = 9))
```

Now construct a negative log-likelihood function that differentiates among groups:

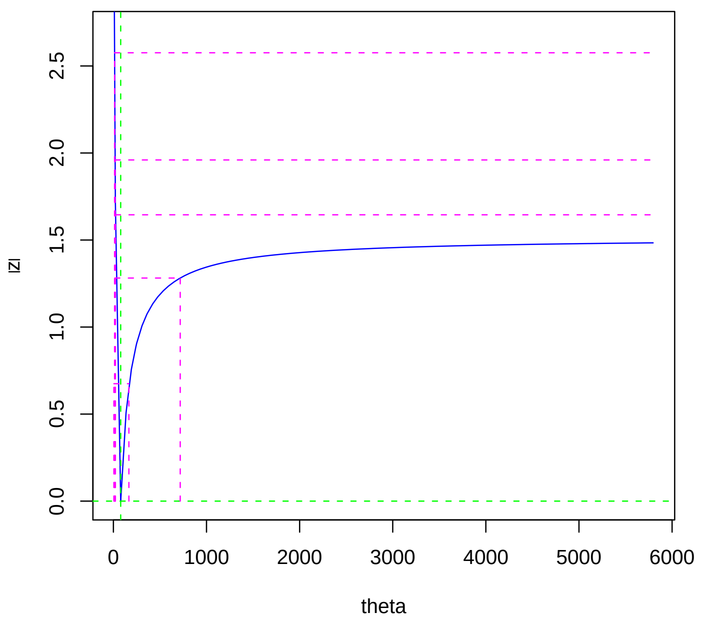
```
> ml1 <- function(mu1, mu2, mu3, theta, x) {
+     mu <- c(mu1, mu2, mu3)[as.numeric(x$dilution)]
+     size <- x$n
+     -sum(dbetabinom(x$y, mu, size, theta, log = TRUE))
+ }

> m1 <- mle2(ml1, start = list(mu1 = 0.5, mu2 = 0.5, mu3 = 0.5,
+     theta = 1), data = list(x = orob1))
```

Get convergence warning — also almost but not quite the results obtained in [?].

```
> m2 <- mle2(ml1, start = as.list(coef(m1)), control = list(parscale = coef(m1)),
+     data = list(x = orob1))
> rbind(coef(m1), coef(m2))
> p2 <- profile(m2)
> confint(p2)
> confint(m2, method = "quad")
> sqrt(diag(vcov(m2)))

> plot(p2, which = "theta")
```



```
> ml0 <- function(mu, theta, x) {
+     size <- x$n
+     -sum(dbetabinom(x$y, mu, size, theta, log = TRUE))
+ }
> m0 <- mle2(ml0, start = list(mu = 0.5, theta = 100), data = list(x = orob1))
```

```
> ml3 <- function(mu1, mu2, mu3, x) {
+     mu <- c(mu1, mu2, mu3)[as.numeric(x$dilution)]
+     size <- x$n
+     -sum(dbinom(x$y, prob = mu, size = size, log = TRUE))
+ }
```

```
> m3 <- mle2(ml3, start = as.list(coef(m1)[1:3]), data = list(x = orob1))
```

```
> anova(m2, m0)
```

```
Likelihood Ratio Tests
Model 1: m2,
Model 2: m0,
  Tot Df Deviance  Chisq Df Pr(>Chisq)
1      4   69.981
2      2  112.515 42.534  2  5.805e-10 ***
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1
```

```
> AICtab(m2, m0, weights = TRUE, delta = TRUE)

   AIC   df dAIC  weight
m2  78.0 4    0.0 1
m0 116.5 2   38.5 <0.001

> BICtab(m2, m0, delta = TRUE, nobs = nrow(orob1))

   BIC   df dBIC
m2  81.1 4    0.0
m0 118.1 2   37.0

> AICctab(m2, m0, nobs = nrow(orob1))

   AICc  df
m2  81.6 4
m0 117.4 2
```

- anova method
- warnings on convergence failure
- more robust to non-positive-definite Hessian
- when profiling fails because better value is found, report new values
- can take named vectors as well as lists as starting parameter vectors
- added optional arguments to AIC (corr, nobs, delta), BIC, confint (method=c(”spline”,”uniroot”,”quad”))
- more options for colors and line types in profile plots
- mle.options()
- data= argument
- handling of names in argument lists

Wish list:

- variable-length chunks in argument list
- limited automatic differentiation (add capability for common distributions)
- ability to use alternate optimizers (e.g. nlmin/b)