Binomial with weights

The Binomial mass function is given by

$$P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}, \tag{1}$$

denoting the probability of obtaining k successes out of n trials.
The experimental conditions vary in each trial, affecting the probability of success. In particular, we focus on a situation where the either side of the coin we flip can be one of two values: *heavy* or *light* and that this varies by trial and is recorded by trial. Now the probability of k heads depends on the weight of the heads side and the weight of the tails side, notationally

$$P(X = k|w_h, w_t) = \binom{n}{k} p^k_{w_h,w_t} (1-p_{w_h,w_t})^{n-k}, \tag{2}$$

where w_h and w_t are the weight of the heads and tails sides, respectively. The probability of heads now depend on those weights.
Data for 8 hypothetical trials might look like

```
df0 <- expand.grid(heads.weight = c("light", "heavy"),
                   tails.weight = c("light", "heavy"))

df <- cbind(outcome = c("H","T","H","H","T","T","H", "T"),
            df0[rep(1:4, each = 2), 2:1])

df$outcome <- factor(as.character(df$outcome), levels = c("T", "H"))

rownames(df) <- NULL

print(df, row.names = FALSE)
```

```
##  outcome tails.weight heads.weight
##        H        light        light
##        T        light        light
##        H        light        heavy
##        H        light        heavy
##        T        heavy        light
##        T        heavy        light
##        H        heavy        heavy
##        T        heavy        heavy
```

The associated probabilities of heads under given conditions are

```
table(df[df$outcome == "H", c("tails.weight", "heads.weight")])/
table(df[, c("tails.weight", "heads.weight")])
```

```
##              heads.weight
## tails.weight light heavy
##        light   0.5   1.0
##        heavy   0.0   0.5
```

First way of analyzing the data using a glm, effectively a Bernoulli trial

```
glm0 <- glm(outcome ~ tails.weight * heads.weight, family = "binomial", data = df)

pred.df <- unique(df[, 2:3])

pred.df$p0 <- round(predict(glm0, newdata = pred.df, type = "response"), 6)

pred.df
```

```
##   tails.weight heads.weight  p0
## 1        light        light 0.5
## 3        light        heavy 1.0
## 5        heavy        light 0.0
## 7        heavy        heavy 0.5
```

Second method is to group the counts

```
library(reshape2)
(df.binom <- dcast(df, heads.weight + tails.weight ~ outcome,
                   fun.aggregate = length))
```

```
## Using heads.weight as value column:  use value.var to override.
```

```
##   heads.weight tails.weight T H
## 1        light        light 1 1
## 2        light        heavy 2 0
## 3        heavy        light 0 2
## 4        heavy        heavy 1 1
```

```
glm1 <- glm(cbind(H, T) ~ tails.weight * heads.weight, family = "binomial",
            data = df.binom)

pred.df$p1 <- round(predict(glm1, newdata = pred.df, type = "response"), 6)

pred.df
```

```
##   tails.weight heads.weight  p0  p1
## 1        light        light 0.5 0.5
## 3        light        heavy 1.0 1.0
## 5        heavy        light 0.0 0.0
## 7        heavy        heavy 0.5 0.5
```

Third method is incorrect but mimics the issue with how weights were treated in the multinomial fits to date

```
df$weight <- with(df, ifelse(outcome == "H", as.character(heads.weight),
                             as.character(tails.weight)))

df

##   outcome tails.weight heads.weight weight
## 1       H        light        light  light
## 2       T        light        light  light
## 3       H        light        heavy  heavy
## 4       H        light        heavy  heavy
## 5       T        heavy        light  heavy
## 6       T        heavy        light  heavy
## 7       H        heavy        heavy  heavy
## 8       T        heavy        heavy  heavy

glm2 <- glm(outcome ~ weight, family = "binomial", data = df)

coef(glm2)

##   (Intercept)   weightlight
## -3.955490e-17  1.692593e-16
```

Note that the weight effect is cancelled out. When there are component-specific influencing the probabilities of success, these should be included as separate variables, at least the way the data have been generated above. For the continuous weight data of the gear trials, we will include separate variables for the bulk-weight of each cod-end and the second order interactions as a first pass.