# Notebook: Races vs Tournament

Last updated: 06 June, 2017

## Contents

## 1 Descriptive statistics

Table XXX shows summary statistics. It also provides p-values from Kruskal-Wallis non-parametric test, showing no evidence of systematic differences across the three treatment groups (the lowest p-value was xxxx). Hence, our randomization appears successful.

```r
# Library
require(xtable)
require(races)

# Load data
data(races)
attach(races)

# Define help functions
balance.test <- function(x) {
    l <- split(x, treatment)
    kruskal.test(l)$p.val
}
nomiss <- function(x) sum(!is.na(x))

# New variables
year <- as.numeric(format(member_date, "%Y"))
month <- as.numeric(format(member_date, "%m"))
hours <- week1 + week2 + week3 + week4
rating <- mm_rating
submissions <- mm_events
registrations <- mm_reg
lpaid <- log(paid)

# Dataset
dat <- data.frame(year, month, rating, submissions, registrations,
    lpaid, nwins, ntop10, risk, hours, timezone)

# Compute summary statistics
mu <- sapply(dat, mean, na.rm = TRUE)
q50 <- sapply(dat, median, na.rm = TRUE)
std <- sapply(dat, sd, na.rm = TRUE)
pval <- sapply(dat, balance.test)
n <- sapply(dat, nomiss)

# Render table
tab <- cbind(mu, q50, std, n, pval)
colnames(tab) <- c("Mean", "Median", "St.Dev.", "Obs.", "P-value")
xtab <- xtable(tab)
digits(xtab) <- c(rep(0, ncol(tab)), 3)
align(xtab) <- c("@{}l", rep("r", ncol(tab)))
caption(xtab) <- "Summary statistics"
label(xtab) <- "summary"

detach(races)
```

```r
render.xtable(xtab)
```

## 2 Analysis of entry decision (June 2, 2017)

We test differences in participation rates across treatment groups. We find no significant differences using a Fisher's exact test. We further test differences in participation rates between large and small rooms. But we find no significant differences.

**Table 1:** Summary statistics

| | Mean | Median | St.Dev. | Obs. | P-value |
|---|---|---|---|---|---|
| year | 2010 | 2010 | 4 | 299 | 0.596 |
| month | 6 | 6 | 4 | 299 | 0.793 |
| rating | 1322 | 1278 | 425 | 205 | 0.989 |
| submissions | 7 | 2 | 12 | 299 | 0.867 |
| registrations | 18 | 9 | 23 | 299 | 0.626 |
| lpaid | 8 | 8 | 3 | 139 | 0.791 |
| nwins | 0 | 0 | 2 | 299 | 0.370 |
| ntop10 | 2 | 0 | 5 | 299 | 0.273 |
| risk | 6 | 7 | 2 | 279 | 0.958 |
| hours | 31 | 24 | 25 | 277 | 0.995 |
| timezone | 2 | 2 | 5 | 277 | 0.389 |

We model binary responses of entry as logistic:

$$y = 1 \iff \text{ability} \geq \theta.$$

We find that "ratings" and "hours" are strong predictors of entry. Yet, precision of the predictions seems pretty low using GLM. Fit does not seem change much if we use more flexible non-parametric models, such as GAMs.

```r
rm(list = ls())

# Load libraries
require(races)
require(stargazer)
# require(mgcv) require(arm)

# Load Data
data(races)
attach(races)

# Odds between treatments
tab <- table(submit, treatment)
fisher.test(tab)
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  tab
## p-value = 0.5255
## alternative hypothesis: two.sided
```

```r
# Odds between large & small rooms
tab <- table(submit, room_size)
fisher.test(tab)
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  tab
## p-value = 1
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##  0.5689643 1.6912635
## sample estimates:
## odds ratio
##  0.9846648
```

```
# New variables
year <- as.numeric(format(member_date, "%Y"))
month <- as.numeric(format(member_date, "%m"))
hours <- week1 + week2 + week3 + week4
rating <- (mm_rating - median(mm_rating, na.rm = TRUE))/100

# Impute missing values
hours.imp <- impute(hours, "random")
lhours.imp <- log(hours.imp)
risk.imp <- impute(risk, "random")
timezone.imp <- impute(timezone, "random")


# New dataset
dat <- data.frame(submit, treatment, rating, nwins, ntop10, year,
    month, lhours.imp, risk.imp, timezone.imp)


# Binomial [odds(y) = exp(alpha_i + beta * rating)]
fit <- rep()
fit$m1 <- glm(submit ~ treatment, binomial(logit), data = dat)
fit$m2 <- update(fit$m1, "~ . + rating")
fit$m3 <- update(fit$m1, "~ . + rating + lhours.imp")
fit$m4 <- glm(submit ~ ., binomial(logit), data = dat)
fit$m5 <- glm(submit ~ . * treatment, binomial(logit), data = dat)

# Model selection
fit$m5 <- step(fit$m4)
```

```
## Start:  AIC=272.64
## submit ~ treatment + rating + nwins + ntop10 + year + month +
##     lhours.imp + risk.imp + timezone.imp
##
##                Df Deviance    AIC
## - treatment     2   251.07 269.07
## - risk.imp      1   250.64 270.64
## - year          1   250.65 270.64
## - timezone.imp  1   250.84 270.84
## - ntop10        1   251.23 271.23
## - nwins         1   251.44 271.44
## - month         1   251.49 271.49
## <none>              250.64 272.64
## - rating        1   254.18 274.18
## - lhours.imp    1   255.48 275.48
##
## Step:  AIC=269.07
## submit ~ rating + nwins + ntop10 + year + month + lhours.imp +
##     risk.imp + timezone.imp
##
##                Df Deviance    AIC
## - risk.imp      1   251.07 267.07
## - year          1   251.08 267.08
## - timezone.imp  1   251.28 267.28
## - ntop10        1   251.63 267.63
## - nwins         1   251.88 267.88
## - month         1   252.02 268.02
## <none>              251.07 269.07
## - rating        1   254.71 270.71
## - lhours.imp    1   255.95 271.95
##
## Step:  AIC=267.08
## submit ~ rating + nwins + ntop10 + year + month + lhours.imp +
##     timezone.imp
##
##                Df Deviance    AIC
## - year          1   251.09 265.09
## - timezone.imp  1   251.28 265.28
## - ntop10        1   251.64 265.64
## - nwins         1   251.88 265.88
## - month         1   252.03 266.04
## <none>              251.07 267.07
## - rating        1   254.75 268.75
## - lhours.imp    1   256.26 270.26
##
## Step:  AIC=265.09
## submit ~ rating + nwins + ntop10 + month + lhours.imp + timezone.imp
##
##                Df Deviance    AIC
## - timezone.imp  1   251.29 263.29
## - ntop10        1   251.64 263.64
## - nwins         1   251.88 263.88
## - month         1   252.04 264.04
## <none>              251.09 265.09
## - rating        1   254.76 266.76
## - lhours.imp    1   256.61 268.61
##
## Step:  AIC=263.29
## submit ~ rating + nwins + ntop10 + month + lhours.imp
##
##              Df Deviance    AIC
## - ntop10      1   251.84 261.84
## - nwins       1   252.09 262.09
## - month       1   252.29 262.29
## <none>            251.29 263.29
## - rating      1   254.82 264.82
## - lhours.imp  1   256.64 266.64
##
## Step:  AIC=261.84
## submit ~ rating + nwins + month + lhours.imp
##
##              Df Deviance    AIC
## - nwins       1   252.11 260.11
## - month        1   252.76 260.76
## <none>             251.84 261.84
## - lhours.imp  1   257.10 265.10
## - rating      1   259.45 267.45
##
## Step:  AIC=260.11
## submit ~ rating + month + lhours.imp
##
##              Df Deviance    AIC
## - month        1   252.96 258.96
## <none>             252.11 260.11
## - lhours.imp  1   257.76 263.76
## - rating      1   260.27 266.27
##
## Step:  AIC=258.96
## submit ~ rating + lhours.imp
##
##              Df Deviance    AIC
## <none>             252.96 258.96
## - lhours.imp  1   259.15 263.15
## - rating      1   262.67 266.67
```

```
# Coefficients
stargazer(fit, type = "text", digits = 2, keep.stat = "n")
```

```
##
## ===============================================================
##                                 Dependent variable:
##                     -------------------------------------------
##                                       submit
##                       (1)      (2)      (3)      (4)     (5)
## ---------------------------------------------------------------
## treatmenttournament   0.32     0.32     0.27     0.24
##                      (0.31)   (0.37)   (0.37)   (0.38)
##
## treatmentreserve      0.04     0.20     0.18     0.17
##                      (0.32)   (0.37)   (0.37)   (0.38)
##
## rating                       0.10***  0.11***   0.09*  0.11***
##                               (0.04)   (0.04)   (0.05)  (0.04)
##
## nwins                                           -0.16
##                                                 (0.18)
##
## ntop10                                           0.05
##                                                 (0.07)
##
## year                                            0.004
##                                                 (0.05)
##
## month                                            0.04
##                                                 (0.05)
```

```
##
## lhours.imp                                 0.57**  0.55**   0.58**
##                                            (0.24)  (0.26)   (0.24)
##
## risk.imp                                            0.002
##                                                    (0.07)
##
## timezone.imp                                        -0.01
##                                                    (0.03)
##
## Constant             -1.03*** -0.81*** -2.62*** -10.90  -2.49***
##                       (0.23)   (0.27)   (0.82)  (94.56)  (0.79)
##
## ----------------------------------------------------------------
## Observations           299      205      205      205     205
## ================================================================
## Note:                                *p<0.1; **p<0.05; ***p<0.01
```

```
# Non-symmetric distributions
fit$cauchit <- update(fit$m4, family = binomial(cauchit))
fit$cloglog <- update(fit$m4, family = binomial(cloglog))

# What model to choose?
cbind(deviance = sapply(fit, deviance), n = sapply(fit, function(x) length(x$y)))
```

```
##         deviance   n
## m1      357.4931 299
## m2      258.3834 205
## m3      252.4151 205
## m4      250.6376 205
## m5      252.9567 205
## cauchit 251.9220 205
## cloglog 251.5145 205
```

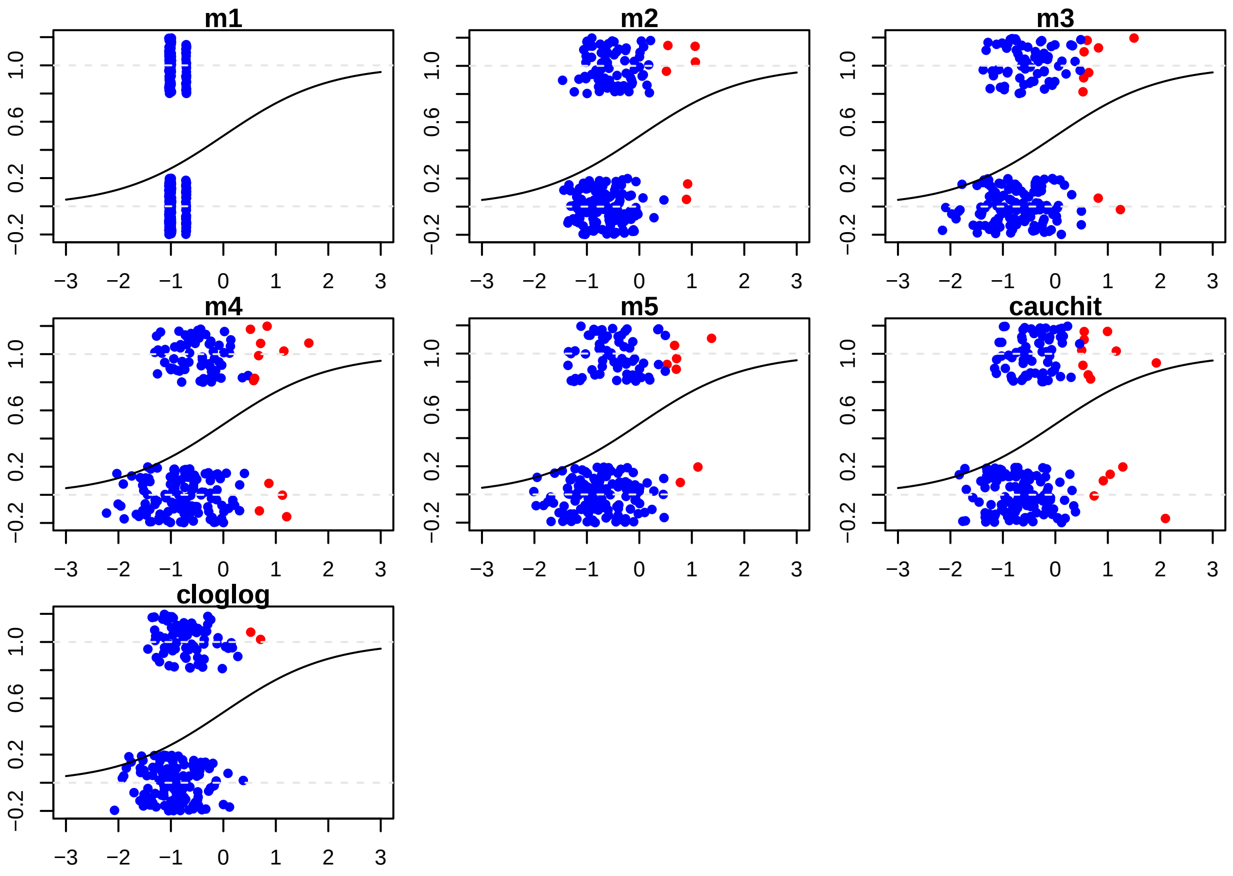
```
# FIGURE
plot.fit <- function(x, ...) {
    yhat <- predict(x)
    plot(jitter(yhat), jitter(as.numeric(x$y)), pch = 16, col = ifelse(yhat >
        0.5, "red", "blue"), ...)
    curve(ilogit, add = TRUE)
    abline(h = c(0, 1), lty = 2, col = gray(0.9))
}
par(mfrow = c(3, 3), mar = c(2, 2, 1, 1))
sapply(1:length(fit), function(i) {
    plot.fit(fit[[i]], xlim = c(-3, 3))
    title(names(fit)[i])
})
```

```
## [[1]]
## NULL
##
## [[2]]
## NULL
##
## [[3]]
## NULL
##
## [[4]]
## NULL
##
## [[5]]
## NULL
##
## [[6]]
## NULL
##
## [[7]]
## NULL
```

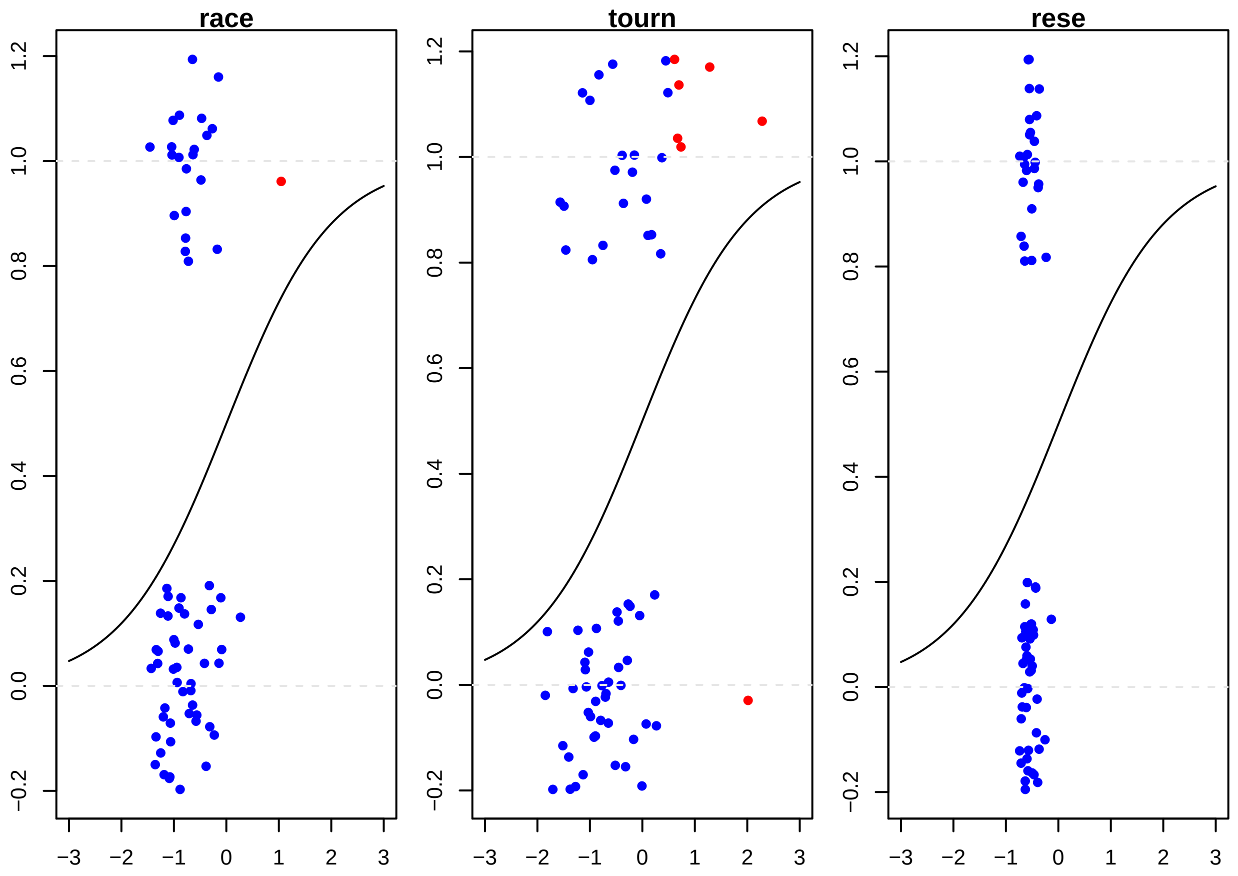
```
# Regression by treatment
fit2 <- rep()
fit2$race <- glm(submit ~ rating, data = dat, binomial(logit),
    subset = treatment == "race")
fit2$tourn <- glm(submit ~ rating, data = dat, binomial(logit),
    subset = treatment == "tournament")
fit2$rese <- glm(submit ~ rating, data = dat, binomial(logit),
    subset = treatment == "reserve")

stargazer(fit2, type = "text", column.labels = names(fit2))
```

```
##
## ==============================================
##                         Dependent variable:
##                     ----------------------------
##                                submit
##                        race      tourn     rese
##                        (1)        (2)       (3)
## ----------------------------------------------
## rating                0.104    0.189***   0.029
##                      (0.063)    (0.070)  (0.060)
##
## Constant            -0.813***  -0.551**  -0.558**
##                      (0.271)    (0.268)  (0.254)
##
## ----------------------------------------------
## Observations            68        69        68
## Log Likelihood       -41.376   -41.632  -44.604
## Akaike Inf. Crit.     86.752    87.264   93.209
## ==============================================
## Note:                *p<0.1; **p<0.05; ***p<0.01
```

```
# Plot
par(mfrow = c(1, 3), mar = c(2, 2, 1, 1))
```

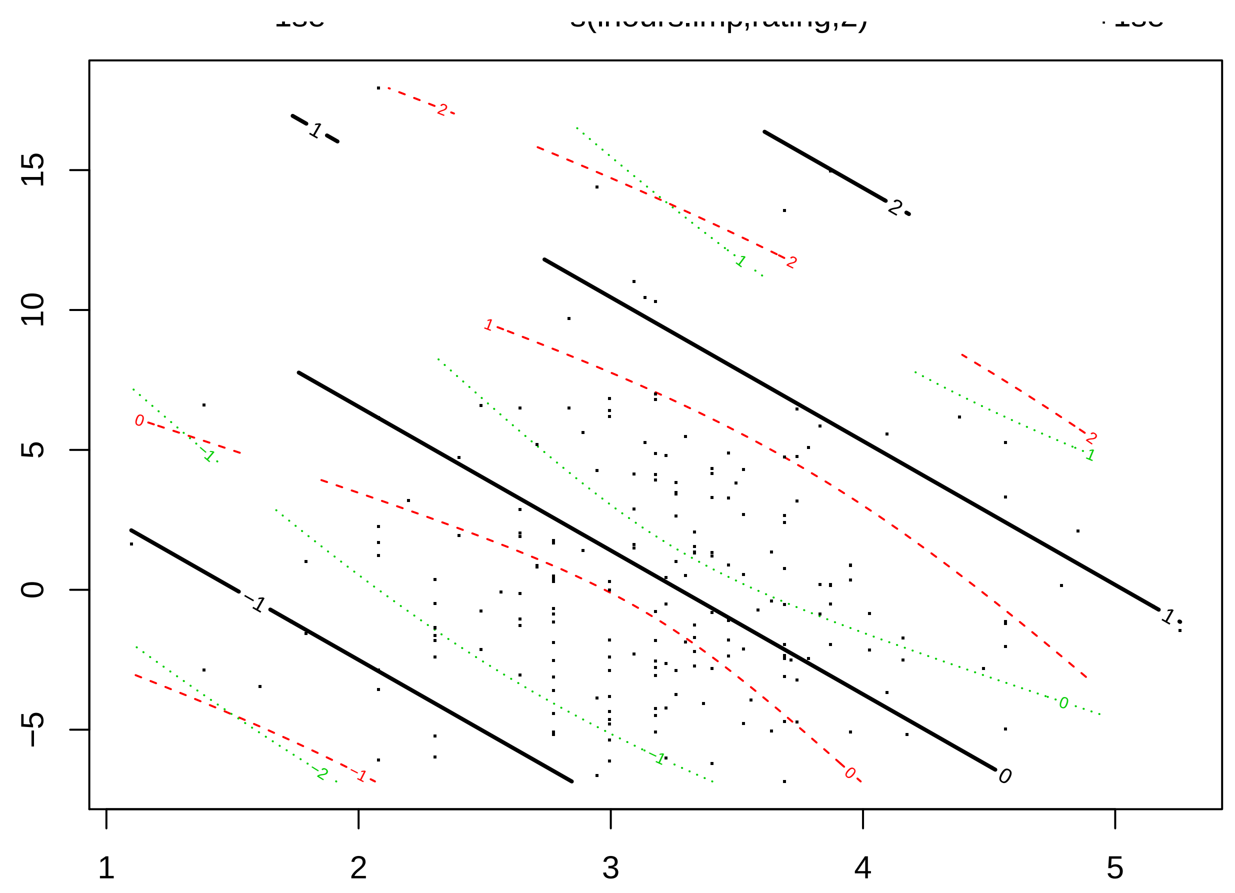


```
sapply(1:3, function(i) {
    plot.fit(fit2[[i]], xlim = c(-3, 3))
    title(names(fit2)[i])
})
```

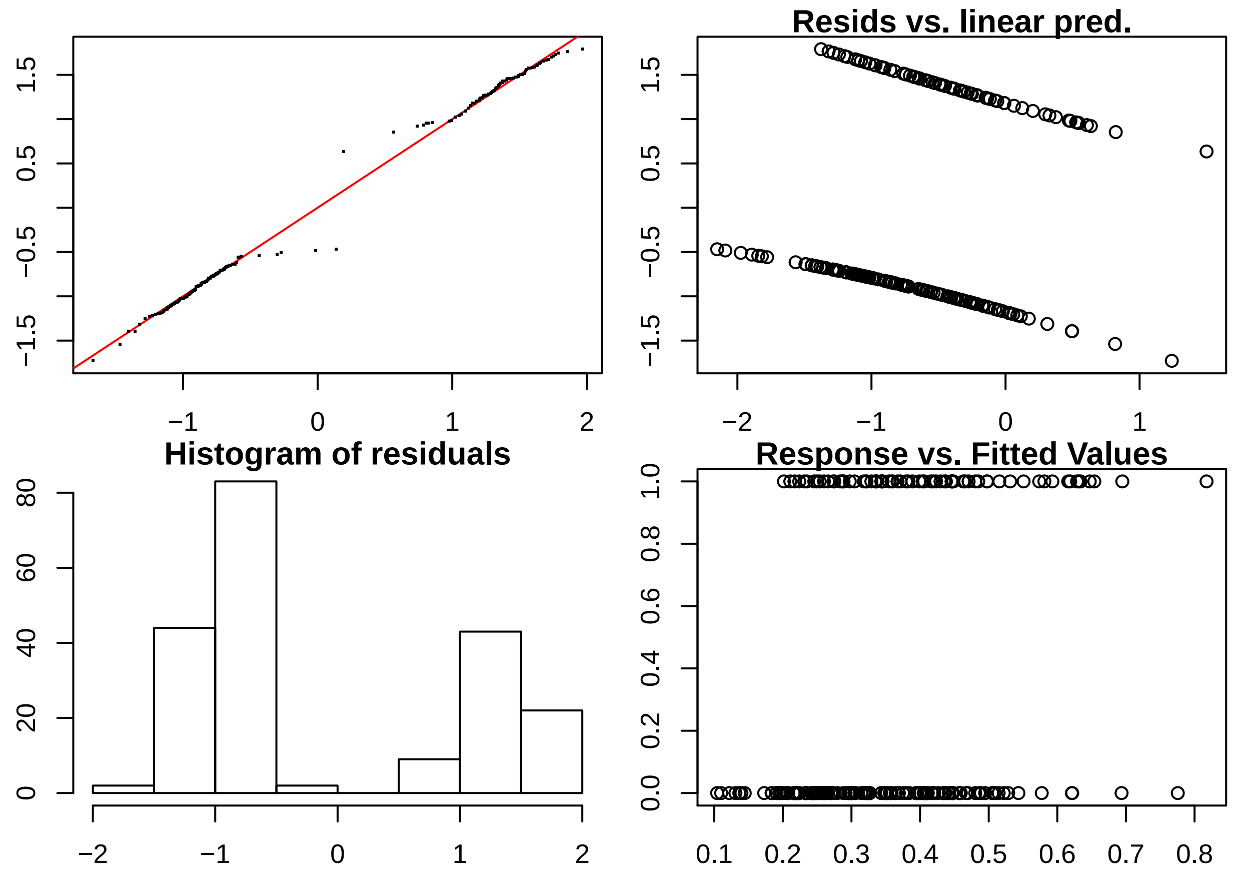


```
## [[1]]
## NULL
##
## [[2]]
## NULL
##
## [[3]]
## NULL
```

```
# Bayesian models
model <- "submit ~ treatment + tothours.imp + mm_rating.100 + educ+gender+timezone"
# bayes <- arm::bayesglm(model, family=binomial)
# summary(bayes)

# Generalized additive models
b <- mgcv::gam(submit ~ treatment + s(lhours.imp, rating), data = dat,
    family = binomial)
plot(b, pages = 1, seWithMean = TRUE)  ## 'with intercept' CIs
```

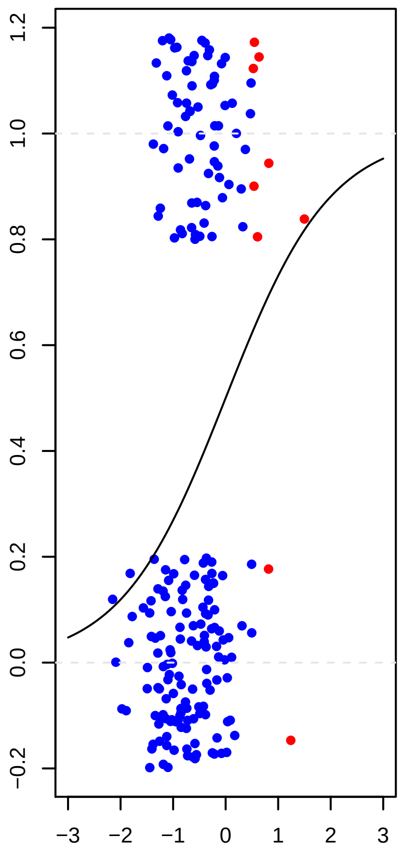


```r
mgcv::gam.check(b)
```



```
##
## Method: UBRE   Optimizer: outer newton
## full convergence after 8 iterations.
## Gradient range [-6.41759e-07,-6.41759e-07]
## (score 0.2800743 & scale 1).
## Hessian positive definite, eigenvalue range [6.417315e-07,6.417315e-07].
## Model rank =  32 / 32
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                            k'    edf k-index p-value
## s(lhours.imp,rating) 29.000  2.000   0.941    0.16
```

```
plot.fit(b, xlim = c(-3, 3))
```



## 3 Timing of entry decision using panel data

We transform data in a panel with the date of the first submission. The probability of entry in a given date is modeled with a conditional logit. We find that the probability of entry in a given date decreases over time. (if it was at random 1/8 chances of having a submission in a given date). The estimated drop is about 13 percent from one date to the next (a reduction of above 100% from the first to the last date). Using interactions with treatment dummies, we find that the decrease in the probability of entry is negative in all treatments but it is about 25 percent in the race (with pvalue<0.05) it is only 8 (pval) and 6 (pval) percent for tournaments and reserve, respectively.

```r
rm(list = ls())
require(magrittr)
require(survival)
require(races)
data(scores)

# Create variable
dmin <- function(x) x - min(x)
scores$date <- scores$timestamp %>% as.Date %>% as.numeric %>%
    dmin

create.panel <- function(data) {
    data$submit_first <- 1
    g <- with(data, expand.grid(date = seq(min(date), max(date)),
        coder_id = unique(coder_id), submit_first = 0))
    g2 <- rbind(g, data[, c("date", "coder_id", "submit_first")])
    aggregate(submit_first ~ date + coder_id, g2, FUN = sum)
}

scores.panel <- create.panel(subset(scores, submission == 1))
z <- merge(scores.panel, races, by = "coder_id")


# Unconditional semi-parametric [Odds(first) = exp(alpha +
# beta * t )]
fit <- rep()
fit$clog <- clogit(submit_first ~ date, data = z)
summary(fit$clog)
```

```
## Call:
## coxph(formula = Surv(rep(1, 774L), submit_first) ~ date, data = z,
##     method = "exact")
##
##   n= 774, number of events= 86
##
##          coef exp(coef) se(coef)      z Pr(>|z|)
## date -0.1510    0.8598   0.0461 -3.275  0.00106 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##      exp(coef) exp(-coef) lower .95 upper .95
## date    0.8598      1.163    0.7855    0.9412
##
## Rsquare= 0.014   (max possible= 0.498 )
## Likelihood ratio test= 11.17  on 1 df,   p=0.0008303
## Wald test            = 10.73  on 1 df,   p=0.001055
## Score (logrank) test = 11.02  on 1 df,   p=0.0008998
```

```
# Unconditional parametric [Odds(first) = exp(alpha + beta *
# t )]
fit$logi <- glm(submit_first ~ date, data = z, binomial(logit))
summary(fit$logi)
```

```
##
## Call:
## glm(formula = submit_first ~ date, family = binomial(logit),
##     data = z)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.6252  -0.5441  -0.4394  -0.3801   2.3687
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.53318    0.18853  -8.132 4.21e-16 ***
## date        -0.15121    0.04614  -3.277  0.00105 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 539.99  on 773  degrees of freedom
## Residual deviance: 528.81  on 772  degrees of freedom
## AIC: 532.81
##
## Number of Fisher Scoring iterations: 5
```

```
# Unconditional with interaction
fit$logi2 <- glm(submit_first ~ date + date:treatment, data = z,
    binomial(logit))
summary(fit$logi2)
```

```
##
## Call:
## glm(formula = submit_first ~ date + date:treatment, family = binomial(logit),
##     data = z)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.6267  -0.5273  -0.4601  -0.3801   2.5737
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -1.52808    0.18858  -8.103 5.36e-16 ***
## date                     -0.21835    0.07201  -3.032  0.00243 **
## date:treatmenttournament  0.08546    0.07472   1.144  0.25273
## date:treatmentreserve     0.09344    0.07689   1.215  0.22423
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 539.99  on 773  degrees of freedom
## Residual deviance: 526.96  on 770  degrees of freedom
## AIC: 534.96
##
## Number of Fisher Scoring iterations: 5
```

```
# Note the difference when the intercept is not there

# Conditional [Odds(first) = exp(alpha_i + beta * t )]
fit$coclog <- clogit(submit_first ~ date + strata(coder_id),
    data = z)
summary(fit$coclog)
```

```
## Call:
## coxph(formula = Surv(rep(1, 774L), submit_first) ~ date + strata(coder_id),
##     data = z, method = "exact")
##
##   n= 774, number of events= 86
##
##         coef exp(coef) se(coef)      z Pr(>|z|)
## date -0.1340    0.8746   0.0433 -3.095  0.00197 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
##      exp(coef) exp(-coef) lower .95 upper .95
## date    0.8746      1.143    0.8034    0.9521
##
## Rsquare= 0.013   (max possible= 0.386 )
## Likelihood ratio test= 9.93  on 1 df,   p=0.001627
## Wald test            = 9.58  on 1 df,   p=0.00197
## Score (logrank) test = 9.81  on 1 df,   p=0.001735
```

```
# Interactions with treatment dummies [Odds(first) =
# exp(alpha_i + beta_i * t )]
fit$coclog2 <- clogit(submit_first ~ date:treatment + strata(coder_id),
    data = z)
summary(fit$coclog2)
```

```
## Call:
## coxph(formula = Surv(rep(1, 774L), submit_first) ~ date:treatment +
##     strata(coder_id), data = z, method = "exact")
##
##   n= 774, number of events= 86
##
##                                coef exp(coef) se(coef)      z Pr(>|z|)
## date:treatmentrace       -0.27991   0.75585  0.08801 -3.180  0.00147 **
## date:treatmenttournament -0.08726   0.91644  0.06847 -1.274  0.20253
## date:treatmentreserve    -0.06708   0.93512  0.07522 -0.892  0.37256
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##                          exp(coef) exp(-coef) lower .95 upper .95
## date:treatmentrace          0.7559      1.323    0.6361    0.8982
## date:treatmenttournament    0.9164      1.091    0.8013    1.0481
## date:treatmentreserve       0.9351      1.069    0.8069    1.0837
##
## Rsquare= 0.018   (max possible= 0.386 )
## Likelihood ratio test= 14.16  on 3 df,   p=0.002689
## Wald test            = 12.53  on 3 df,   p=0.005761
## Score (logrank) test = 13.61  on 3 df,   p=0.003487
```

```
# Note: the above model is the same as clogit(submit_first ~
# date + date:treatment + strata(coder_id), data=z)

# Coefficients
sapply(fit, coef)
```

```
## $clog
##       date
## -0.1510078
##
## $logi
## (Intercept)        date
##  -1.5331826  -0.1512076
##
## $logi2
##              (Intercept)                      date date:treatmenttournament
##              -1.52808481               -0.21835013               0.08545552
##    date:treatmentreserve
##               0.09344282
##
## $coclog
##       date
## -0.1339918
##
## $coclog2
##       date:treatmentrace date:treatmenttournament    date:treatmentreserve
##              -0.27990753              -0.08725850              -0.06707555
```

```
# Days as factors


# # Compute first submission time time.max <- '2015-03-16
# 11:59:59 EDT' time.min <- '2015-03-08 12:00:00 EDT'
# time.censored <- 192 fsutime <-
# as.numeric(difftime(races.sub£timestamp, time.min ,
# unit='hours')) dat <- with(races.sub, aggregate(fsutime ~
# handle, FUN=min)) dat£fsu <- 1 dat2 <- merge(dat, races,
# by='handle', all=TRUE) dat2£fsu[is.na(dat2£fsu)] <- 0
# dat2£fsutime[is.na(dat2£fsutime)] <- time.censored surv <-
# rep() surv£m1 <- survreg(Surv(fsutime, fsu) ~ 1, dat2,
# dist='weibull', scale=1) surv£m2 <- survreg(Surv(fsutime,
# fsu) ~ treatment + mmevents, dat2 , dist='weibull',
# scale=1) surv£m3 <- survreg(Surv(fsutime, fsu) ~ treatment
# + mmevents, dat2, dist='exponential') surv£tobin <-
# survreg(Surv(fsutime, fsu, type='left') ~ treatment +
# mmevents, data=dat2, dist='gaussian') stargazer(surv,
# type='text') # A model with different baseline survival
# shapes for two groups, i.e., # two different scale
# parameters survreg(Surv(time, status) ~ ph.ecog + age +
# strata(sex), lung) # There are multiple ways to
# parameterize a Weibull distribution. The survreg # function
# embeds it in a general location-scale family, which is a #
# different parameterization than the rweibull function, and
# often leads # to confusion.  # survreg's scale =
# 1/(rweibull shape) # survreg's intercept = log(rweibull
# scale) # For the log-likelihood all parameterizations lead
# to the same value.  y <- rweibull(1000, shape=2, scale=5)
# survreg(Surv(y)~1, dist='weibull') # Economists fit a model
# called 'tobit regression', which is a standard # linear
# regression with Gaussian errors, and left censored data.
# tobinfit <- survreg(Surv(durable, durable>0, type='left') ~
# age + quant, data=tobin, dist='gaussian')
```

## 4 Analysis of scores

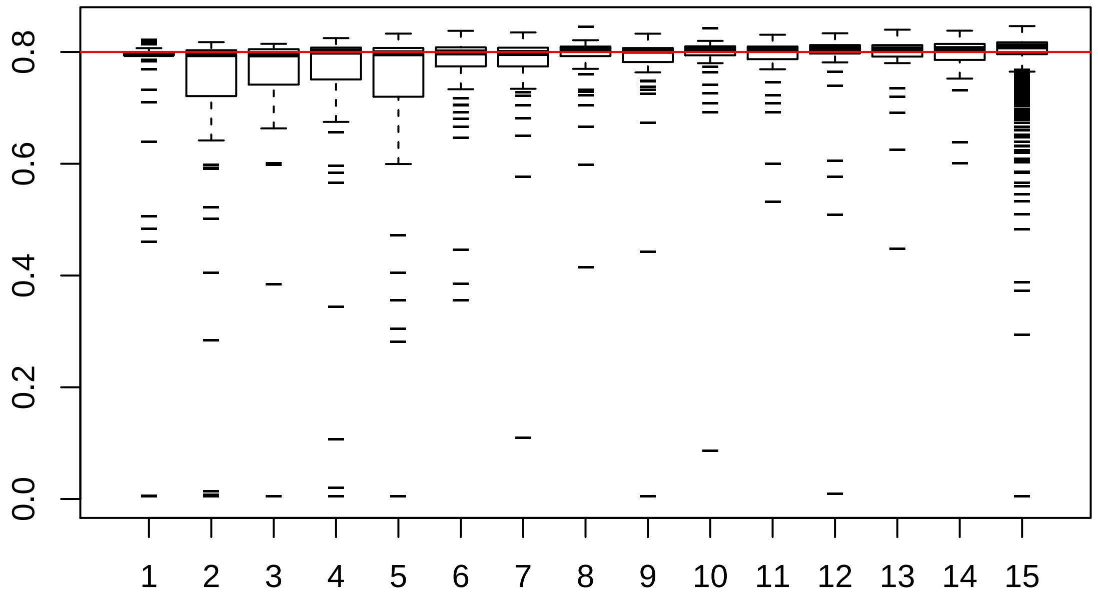
Each submission was scored on 300 abstracts. The score was the F-measure

$$F = 2 * (prediction * recall)/(prediction + recall).$$

This measure ranges from 0 to 1 (perfect score). Abstracts had about 5 mentions to annotate each, for a total of about 1500 entites to "predict" but the number of entities in an abstract was very large.

The best score achieved by Banner was $F = 0.817866$. The winning code achieved a score of $F = 0.843962$. This 0.026096 percentage points difference corrsponded to an improvement of more than 3 percent of the baseline, which was considered excellent by researchers at NIH.

```r
rm(list = ls())

# Load libraries
require(races)
require(stargazer)
# require(mgcv) require(arm)

# Load Data
data(races)
data(scores)

# Merge
dat <- merge(scores, races[, c("coder_id", "treatment", "room")],
    by = "coder_id")
attach(dat)

# New variable
last_submission <- ave(submission, coder_id, FUN = max)


submission.cap <- ifelse(submission > 15, 15, submission)
boxplot(provisional/1e+06 ~ submission.cap, pch = "-")
abline(h = 0.8, col = 2)
```

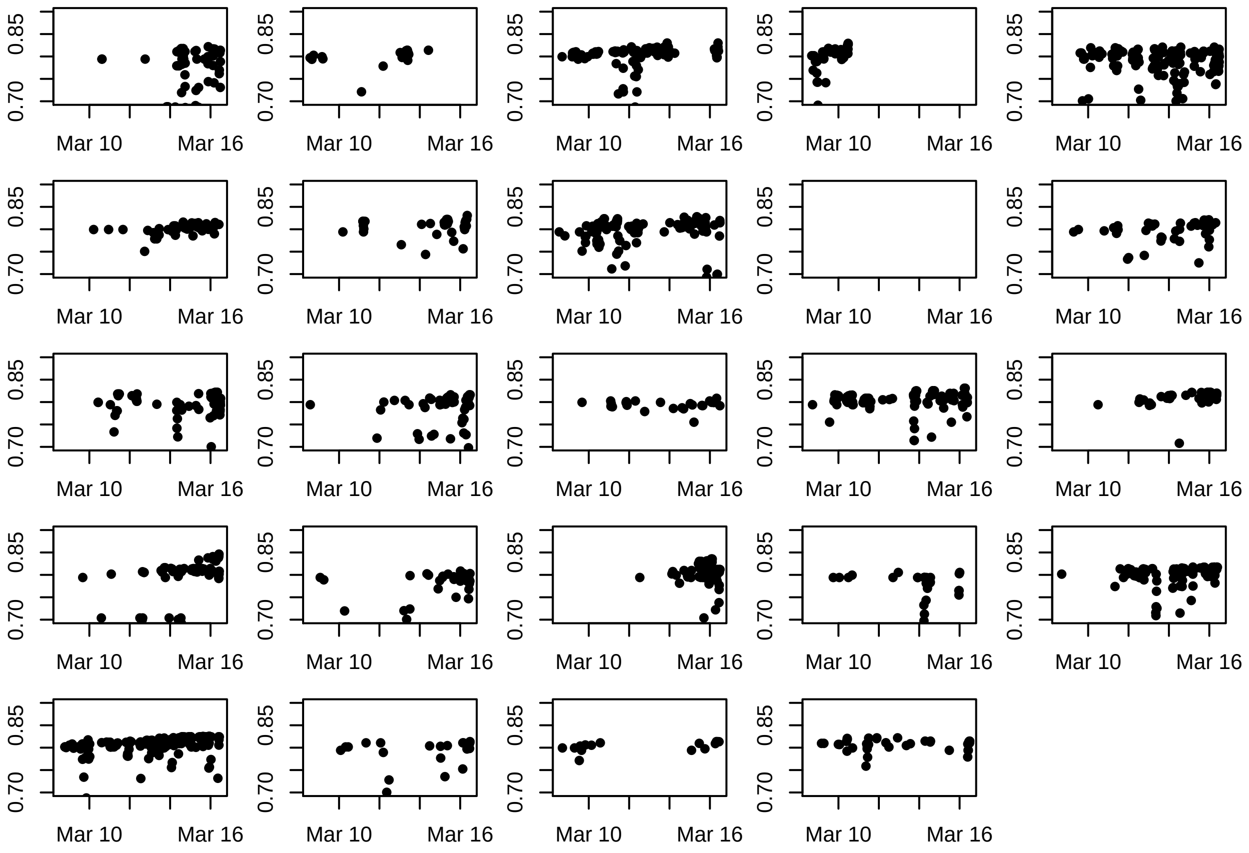


```r
# Time series
l <- split(dat, paste(treatment, room))
xlim <- range(timestamp)
ylim <- c(0.7, 0.9)
par(mfrow = c(5, 5), mar = c(2, 2, 1, 1))
sapply(l, function(x) plot(x$timestamp, x$provisional/1e+06,
    xlim = xlim, ylim = ylim, pch = 16))
```

```
## $`race 1`
## NULL
##
## $`race 2`
## NULL
##
```

```
## $`race 3`
## NULL
##
## $`race 4`
## NULL
##
## $`race 5`
## NULL
##
## $`race 6`
## NULL
##
## $`race 7`
## NULL
##
## $`race 8`
## NULL
##
## $`reserve 1`
## NULL
##
## $`reserve 2`
## NULL
##
## $`reserve 3`
## NULL
##
## $`reserve 4`
## NULL
##
## $`reserve 5`
## NULL
##
## $`reserve 6`
## NULL
##
## $`reserve 7`
## NULL
##
## $`reserve 8`
## NULL
##
## $`tournament 1`
## NULL
##
## $`tournament 2`
## NULL
##
## $`tournament 3`
## NULL
##
```

```
## $`tournament 4`
## NULL
##
## $`tournament 5`
## NULL
##
## $`tournament 6`
## NULL
##
## $`tournament 7`
## NULL
##
## $`tournament 8`
## NULL
```



## 5 Identification

### 5.1 The model

For each room ($l = 1, ..., L$), we observe the count of participants $y_l$ among the $n_l$ registered competitors who have been assigned to some treatment $w_l \in \{0, 1, 2\}$. The count of participating competitors is the realization of a random variable $Y_l$ that depends on a vector of room characteristics $z_l = (x_l, w_l)$ that include the assignment $w_l$ and additional room characteristics $x_l$. We assume that $Y_l$ is distributed as a binomial variable with parameters $n_l$ and $p(x_l, \theta)$ where $p(x, \theta)$ describes the probability for a competitor to participate in a room described by the vector $x$. The model is then the following:

$$Y_l \sim \text{Binomial}(N_l, p(x_l, \theta)) \quad \text{for each } l = 1, ..., L. \tag{1}$$

In binomial linear models, the probability function $p(x, \theta)$ is related to a linear prediction through a one-to-one transformation called the link function (see xxxx). Following our theoretical approach, participation is one for those with abilities above a given value:

$$Y_l = 1 \iff \text{ability} \geq m_l$$

where $m_l$ is the marginal type for the room. Given $F_l(\cdot)$ the distribution of individual abilities in each room, the probability of entry is:

$$p(m_l, \theta) = 1 - F_l(m_l) \tag{2}$$

The marginal type is determined by the zero profit condition:

$$R_l(m_l) = m_l^\alpha y_0^\beta t_0^\gamma \tag{3}$$

where the $R_l(m_l)$ is the expected revenue of the marginal type in a given room and it must be equal to the costs evaluated at the deadline $t_0$ and the target $y_0$.

If we assume one single prize $q = 1$ (and we normalize the cost of meeting the deadline to one $t_0^\gamma = 1$) the zero profit condition becomes:

$$F_l(m_l)^{n_l - 1} = m_l^\alpha y_0^\beta \tag{4}$$

By raising to the power $1/(n_l - 1)$ both sides of the equation, we obtain a structural relationship between the probability of entry (eq. XXX) and the marginal type, the number of registered competitors, and the cost function.

$$p(x_l, \theta) = 1 - \left[m_l^\alpha y_0^\beta\right]^{1/(n_l - 1)} \qquad \text{with } \alpha \leq -1, \beta \geq 1.$$

As a result, one could use a "method of moments" to estimate the parameters $\theta = (\alpha, \beta, m_l)$ ($y_0$ and $n_l$ are observed).

## 5.2 Estimation

Let $Y_1, Y_2, ..., Y_L$ be the count of participants generated by our binomial-distribution model. Let denote the corresponding $j$-th moment by $\mu_j$ (about the origin) of the distribution and use a hat to denote the sample moments.

$$\mu_j = \frac{1}{n} \sum x^j$$

Equate the first sample moment to the theoretical mean:

$$\hat{\mu}_1 = np.$$

And equate the second sample moment to:

$$\hat{\mu}_2 = np(1 - p) + (np)^2$$

Let further impose $\alpha = -1$. Then, we have a system of two equations with two unknowns $(\beta, m_l)$.

```
# Compute sample moments
moments <- function(x, k) {
    mean(x^k)
}

mom <- function(x) {
    m1 <- mean(x^1)
}
```

```
# Simulations
prob <- function(alpha, beta, n, m, y0) {
    1 - (m^alpha * y0^beta)^(1/(n - 1))
}
nrooms <- 100

submit.10 <- rbinom(nrooms, size = 10, prob = prob(1, 2, n = 10,
    0.75, 0.1))
submit.15 <- rbinom(nrooms, size = 15, prob = prob(1, 2, n = 15,
    0.75, 0.1))

size <- c(rep(10, nrooms), rep(15, nrooms))
dat <- data.frame(y = c(submit.10, submit.15), size)
chisq.test(table(dat$y, dat$size))
```

```
##
##  Pearson's Chi-squared test
##
## data:  table(dat$y, dat$size)
## X-squared = 6.8286, df = 8, p-value = 0.5552
```

```
# Difference in prob
rbind(tapply(dat$y/dat$size, dat$size, FUN = mean), c(prob(1,
    2, n = 10, 0.75, 0.1), prob(1, 2, n = 15, 0.75, 0.1)))
```

```
##            10        15
## [1,] 0.419000 0.2973333
## [2,] 0.419375 0.2949520
```

## 6 Identification (old)

### 6.1 Example with the Uniform distribution

Imagine that the distribution $F_\theta$ is uniform on $(0, \theta)$ and the cost ability $c_a(x) = 1/x$. Then the zero profit condition (3) becomes

$$\left(\frac{m}{\theta}\right)^{n-1} m = \beta x_l \tag{5}$$

Solving for the marginal type gives:

$$m = \left(\beta x_l \theta^{n-1}\right)^{1/n} \tag{6}$$

So, the probability of participation (**??**) becomes

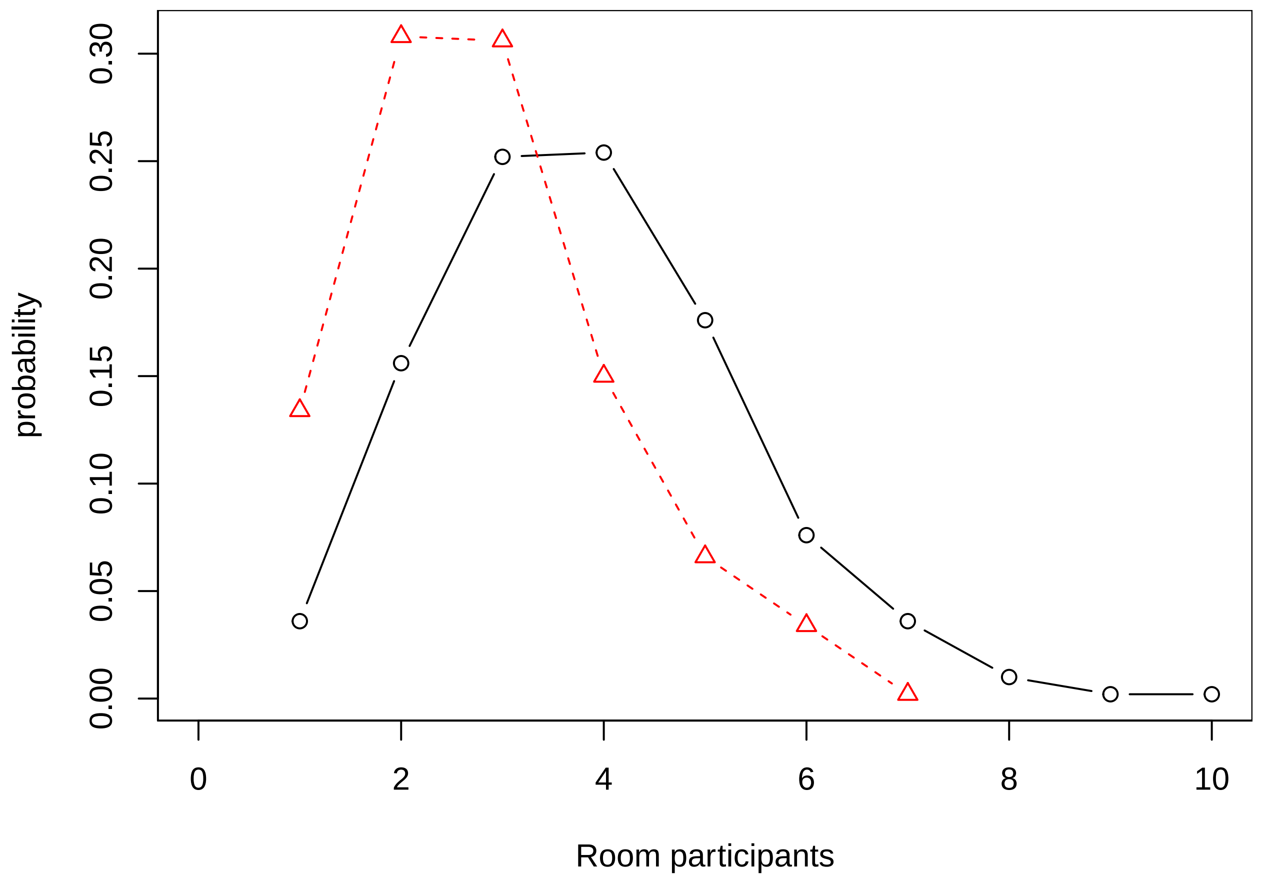


**Figure 1:** Simulated distribution of participants with uniform distribution. Parameters are $\eta = 0.1$ and $n = 15$. Data simulated 500 times. Dashed curve had higher costs ($x_l = 1.5$) than the solid curve ($x_l = 0.5$).

$$
\begin{aligned}
p(x_l, \theta) &= 1 - F_\theta(m) && (7)\\
&= 1 - \frac{\left(\beta x_l \theta^{n-1}\right)^{1/n}}{\theta} && (8)\\
&= 1 - \left(\frac{\beta x_l}{\theta}\right)^{1/n}. && (9)
\end{aligned}
$$

Because it is non-linear in its parameters, the model is not identifiable [need proof]. However, we can re-parametrize the model with $\beta/\theta \equiv \eta$ to make the parameter $\eta$ identifiable.

### 6.2 Example with Beta distribution

In this example, we assume abilities are drawn from a lognormal distribution with parameters $\mu$ and $\sigma$.

### 6.3 Estimation

Our aim is to estimate the parameters $\theta$ and $\beta$ and to evaluate whether costs are different between the three competition modes under study: race, tournament, tournament + requirement.

It is natural to estimate the parameters $\theta$ by maximum likelihood because the ability distribution (and therefore the distribution of the outcomes) is known. The estimation criterion used here is the maximization of the *deviance*.

The deviance is:

$$D(\theta) = -2 \sum Y \log(\frac{Np(x,\theta)}{Y}) + (N - Y) \log(\frac{N - Np}{N - Y}). \tag{10}$$

Note that the deviance is a function of the likelihood (see xxx pg. xxx).

```r
# Define likelihood
structreg <- function(x, y, wt = rep(1, length(y)), intercept = TRUE,
    start = rep(0, p), ...) {
    fmin <- function(beta, X, y, w) {
        p <- 1 - (beta %*% X)^(1/n)   # Function of parameters
        -sum(2 * w * ifelse(y, log(p), log(1 - p)))
    }
    # gmin <- function(beta, X, y, w) { # Gradient here!  }
    if (is.null(dim(x)))
        dim(x) <- c(length(x), 1)
    dn <- dimnames(x)[[2]]
    if (!length(dn))
        dn <- paste("Var", 1:ncol(x), sep = "")
    p <- ncol(x) + intercept
    if (intercept) {
        x <- cbind(1, x)
        dn <- c("(Intercept)", dn)
    }
    if (is.factor(y))
        y <- (unclass(y) != 1)
    # fit <- optim(start, fmin, gmin, X = x, y = y, w = wt,
    # method = 'BFGS', ...)
    fit <- optim(0.5, fmin, X = x, y = y, w = wt, method = "BFGS",
        ...)
    names(fit$par) <- dn
    cat("\nCoefficients:\n")
    print(fit$par)
    cat("\nResidual Deviance:", format(fit$value), "\n")
    cat("\nConvergence message:", fit$convergence, "\n")
    invisible(fit)
}

# Create data
ilogit <- function(x) exp(x)/(1 + exp(x))
n <- 50000
competitors <- 5
x1 <- rchisq(competitors, df = 3)
x2 <- runif(n)
x3 <- rnorm(n)
X <- cbind(rep(1, length(x1)), x1, x2, x3)
betas <- c(-2.5, 0.1, 0.25, 0.5)
fc <- ilogit(X %*% betas)  ## This the fixed cost
p <- 1 - fc^(1/competitors)
y <- rbinom(n = n, size = competitors, prob = p)

# Objective function
fmin <- function(beta, X, y, w, competitors) {
    p <- 1 - ilogit(X %*% beta)^(1/competitors)  # Function of parameters
    -sum(2 * w * ifelse(y, log(p), log(1 - p)))
}

b.start <- runif(length(betas))
(out <- optim(b.start, fmin, X = X, y = y, competitors = competitors,
    w = rep(1, length(y)))$par)
```

```
## [1] -12.8125170   0.4427241   0.8308194   2.1121563
```

```
rbind(out/competitors, betas)
```

```
##             [,1]       [,2]      [,3]      [,4]
##        -2.562503 0.08854481 0.1661639 0.4224313
## betas -2.500000 0.10000000 0.2500000 0.5000000
```

## 6.4 More examples

Suppose F is lognormal$(\mu, \sigma)$, then the zero profit condition is:

$$\left[\frac{1}{2} + \frac{1}{2}\left(\frac{\log m - \mu}{\sqrt{2}\sigma}\right)\right] = a^{-1/(n-1)} K_l \quad (11)$$

```
# xxx
zeroprofit <- function(x, mu, sigma, n) {
    x * plnorm(x, meanlog=mu, sdlog=sigma)^(n-1)
}
for (i in 3:10)
    curve(zeroprofit(x, i, 10, 10), add=TRUE, lty=i)
```

```
# Clear
rm(list = ls())

# Libraries
require(xtable)
require(moments)
require(races)

# Data
data(races)
attach(races)

# TABLE 1.
size <- ave(submit, paste(treatment, room), FUN = length)
(m <- aggregate(submit ~ room + size + treatment, FUN = sum))
```

```
##    room size  treatment submit
## 1     1    9       race      2
## 2     2   10       race      2
## 3     3   10       race      5
## 4     4   10       race      1
## 5     5   15       race      4
## 6     6   15       race      3
## 7     7   15       race      4
## 8     8   15       race      5
## 9     1   10 tournament      3
## 10    2   10 tournament      5
## 11    3   10 tournament      5
## 12    4   10 tournament      2
## 13    5   15 tournament      5
## 14    6   15 tournament      4
## 15    7   15 tournament      4
## 16    8   15 tournament      5
## 17    1   10    reserve      1
## 18    2   10    reserve      3
## 19    3   10    reserve      3
## 20    4   10    reserve      2
## 21    5   15    reserve      4
## 22    6   15    reserve      6
## 23    7   15    reserve      3
## 24    8   15    reserve      5
```