In [12]:

```
df = read.table("assign2.txt", header=TRUE)
str(df)
head(df)
```

```
'data.frame':   504 obs. of  4 variables:
 $ actor      : int  1 1 1 1 1 1 1 1 1 1 ...
 $ condition  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ prosoc_left: int  0 0 1 0 1 1 1 1 0 0 ...
 $ pulled_left: int  0 1 0 0 1 1 0 0 0 0 ...
```

actor	condition	prosoc_left	pulled_left
1	0	0	0
1	0	0	1
1	0	1	0
1	0	0	0
1	0	1	1
1	0	1	1

In [13]:

```
# Create attribute
df$actor = factor(df$actor)
df$condition = factor(df$condition)
df$prosoc_left = factor(df$prosoc_left)
df$pulled_left = factor(df$pulled_left)
df$is_prosoc = as.integer(df$prosoc_left == df$pulled_left)
```

In [14]:

```
# Also create only control and only experiment
single = df[which(df$condition == 0),]
pair = df[which(df$condition == 1),]
```

Binomial Regression

In [15]:

```
binom.model = glm(cbind(is_prosoc, 1 - is_prosoc) ~ ., df, family=binomial)
summary(binom.model)
```

```
Call:
glm(formula = cbind(is_prosoc, 1 - is_prosoc) ~ ., family = binomial,
    data = df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5697  -1.2269   0.8485   1.0813   1.4084

Coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)   6.268e-02  2.868e-01   0.219  0.82701
actor2       -4.958e-01  3.662e-01  -1.354  0.17584
actor3        6.409e-02  3.481e-01   0.184  0.85390
actor4        6.414e-02  3.481e-01   0.184  0.85379
actor5       -2.671e-05  3.465e-01   0.000  0.99994
actor6       -5.909e-01  3.463e-01  -1.707  0.08791 .
actor7       -3.755e-01  3.585e-01  -1.047  0.29492
condition1    1.036e-01  1.839e-01   0.563  0.57336
prosoc_left1  6.566e-01  1.861e-01   3.527  0.00042 ***
pulled_left1  5.299e-02  2.205e-01   0.240  0.81012
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 689.49  on 503  degrees of freedom
Residual deviance: 668.42  on 494  degrees of freedom
AIC: 688.42

Number of Fisher Scoring iterations: 4
```

In [16]:

```
bm.step = step(binom.model, trace=FALSE)
summary(bm.step)
```

```
Call:
glm(formula = cbind(is_prosoc, 1 - is_prosoc) ~ prosoc_left,
    family = binomial, data = df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.4428  -1.1572   0.9335   0.9335   1.1977

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.04763    0.12602  -0.378 0.705484
prosoc_left1  0.65274    0.18235   3.580 0.000344 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 689.49  on 503  degrees of freedom
Residual deviance: 676.50  on 502  degrees of freedom
AIC: 680.5

Number of Fisher Scoring iterations: 4
```

It seems there may be a preference as to where the prosocial option is. We can try see if the mean tells us anything.

In [17]:

```
pchisq(668.48, 1, lower.tail=FALSE)
```

2.13875460832085e-147

In [18]:

```
singlebm.step.mu = predict(bm.step, newdata=single, se.fit=TRUE, type="link")
singlebm.step.muFit = singlebm.step.mu$fit
singlebm.step.muError = singlebm.step.mu$se.fit
alpha = dnorm(0.975)

# Compute fitted CI
widthsingle = alpha * singlebm.step.muError

CIfitsingle = cbind(singlebm.step.muFit-widthsingle, singlebm.step.muFit+widthsingle)
CIsingle = data.frame(CIfitsingle[,1], singlebm.step.muFit, CIfitsingle[,2])

# Rename for clarity
names(CIsingle)[names(CIsingle)=="CIfitsingle...1."] = "2.5%"
names(CIsingle)[names(CIsingle)=="singlebm.step.muFit"] = "Fitted Value (single)"
names(CIsingle)[names(CIsingle)=="CIfitsingle...2."] = "97.5%"
```

In [19]:

```
head(CIsingle)
```

2.5%	Fitted Value (single)	97.5%
-0.07888433	-0.04762805	-0.01637177
-0.07888433	-0.04762805	-0.01637177
0.57242526	0.60511383	0.63780240
-0.07888433	-0.04762805	-0.01637177
0.57242526	0.60511383	0.63780240
0.57242526	0.60511383	0.63780240

In [20]:

```
pairbm.step.mu = predict(bm.step, newdata=pair, se.fit=TRUE, type="link")
pairbm.step.muFit = pairbm.step.mu$fit
pairbm.step.muError = pairbm.step.mu$se.fit
alpha = dnorm(0.975)

# Compute fitted CI
widthpair = alpha * pairbm.step.muError

CIfitpair = cbind(pairbm.step.muFit-widthpair, pairbm.step.muFit+widthpair)
CIpair = data.frame(CIfitpair[,1], pairbm.step.muFit, CIfitpair[,2])

# Rename for clarity
names(CIpair)[names(CIpair)=="CIfitpair...1."] = "2.5%"
names(CIpair)[names(CIpair)=="pairbm.step.muFit"] = "Fitted Value (pair)"
names(CIpair)[names(CIpair)=="CIfitpair...2."] = "97.5%"
```

In [21]:

```
head(CIpair)
```

	2.5%	Fitted Value (pair)	97.5%
37	-0.07888433	-0.04762805	-0.01637177
38	-0.07888433	-0.04762805	-0.01637177
39	0.57242526	0.60511383	0.63780240
40	-0.07888433	-0.04762805	-0.01637177
41	-0.07888433	-0.04762805	-0.01637177
42	-0.07888433	-0.04762805	-0.01637177

In [22]:

```
F = binom.model$family$linkinv
```

In [23]:

```
CI2logit = data.frame(F(CIsingle[,1]), F(CIsingle[,2]), F(CIsingle[,3]))
# Rename for clarity
names(CI2logit)[names(CI2logit)=="F.CIsingle...1.."] = "2.5%"
names(CI2logit)[names(CI2logit)=="F.CIsingle...2.."] = "Fitted Value (pair)"
names(CI2logit)[names(CI2logit)=="F.CIsingle...3.."] = "97.5%"

head(CI2logit)
```

2.5%	Fitted Value (pair)	97.5%
0.4802891	0.4880952	0.4959071
0.4802891	0.4880952	0.4959071
0.6393226	0.6468254	0.6542565
0.4802891	0.4880952	0.4959071
0.6393226	0.6468254	0.6542565
0.6393226	0.6468254	0.6542565

In [24]:

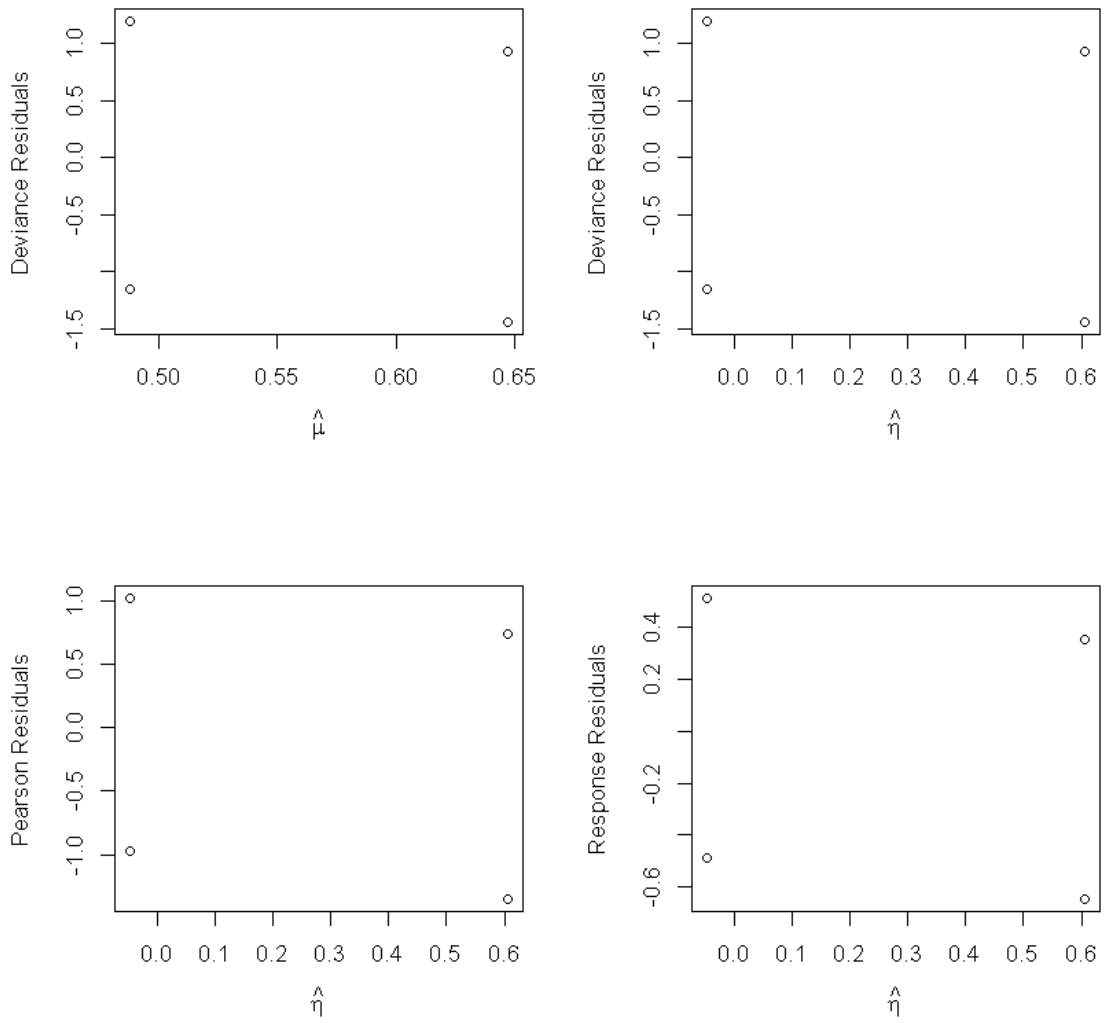
```
CI3logit = data.frame(F(CIpair[,1]), F(CIpair[,2]), F(CIpair[,3]))
# Rename for clarity
names(CI3logit)[names(CI3logit)=="F.CIpair...1.."] = "2.5%"
names(CI3logit)[names(CI3logit)=="F.CIpair...2.."] = "Fitted Value (pair)"
names(CI3logit)[names(CI3logit)=="F.CIpair...3.."] = "97.5%"

head(CI3logit)
```

2.5%	Fitted Value (pair)	97.5%
0.4802891	0.4880952	0.4959071
0.4802891	0.4880952	0.4959071
0.6393226	0.6468254	0.6542565
0.4802891	0.4880952	0.4959071
0.4802891	0.4880952	0.4959071
0.4802891	0.4880952	0.4959071

In [25]:

```
# Residual Plots
par(mfrow=c(2,2))
plot(residuals(bm.step) ~ predict(bm.step, type="response"), xlab=expression(hat(mu)),
ylab="Deviance Residuals")
plot(residuals(bm.step) ~ predict(bm.step, type="link"), xlab=expression(hat(eta)), yla
b="Deviance Residuals")
plot(residuals(bm.step, type="pearson") ~ predict(bm.step, type="link"),
     xlab=expression(hat(eta)), ylab="Pearson Residuals")
plot(residuals(bm.step, type="response") ~ predict(bm.step, type="link"),
     xlab=expression(hat(eta)), ylab="Response Residuals")
```



There seems to be no trend at all

Let's check for overdispersion - we've assumed $\hat{\phi} = 1$ so far...

In [26]:

```
n = dim(df)[1]
p = 2 # number of fitted params
phi.hat = sum(residuals(bm.step, type="pearson")^2) / (n - p)
phi.hat
```

1.00398406374501

Nice, our dispersion parameter is pretty much 1, so the model is good.

Contingency Table - Testing for Independence

In [27]:

```
# The first most interesting attribute to look at is the handedness, which is tested fo
r during control experiment
t1 = table(single$pulled_left, single$actor)
summary(t1)
```

```
Number of cases in table: 252
Number of factors: 2
Test for independence of all factors:
        Chisq = 49.14, df = 6, p-value = 6.99e-09
```

It is very significant, so there is a dependency between the handedness of each chimp. This is expected since it is natural to have a preferred handedness (humans are more right handed).

In [28]:

```
# We can also see if some chimps may be more prosocial by nature
t2 = table(df$is_prosoc, df$actor)
summary(t2)
```

```
Number of cases in table: 504
Number of factors: 2
Test for independence of all factors:
        Chisq = 7.518, df = 6, p-value = 0.2756
```

So during experiments, we observe that chimps are acting independently regardless of the prosocial option

In [29]:

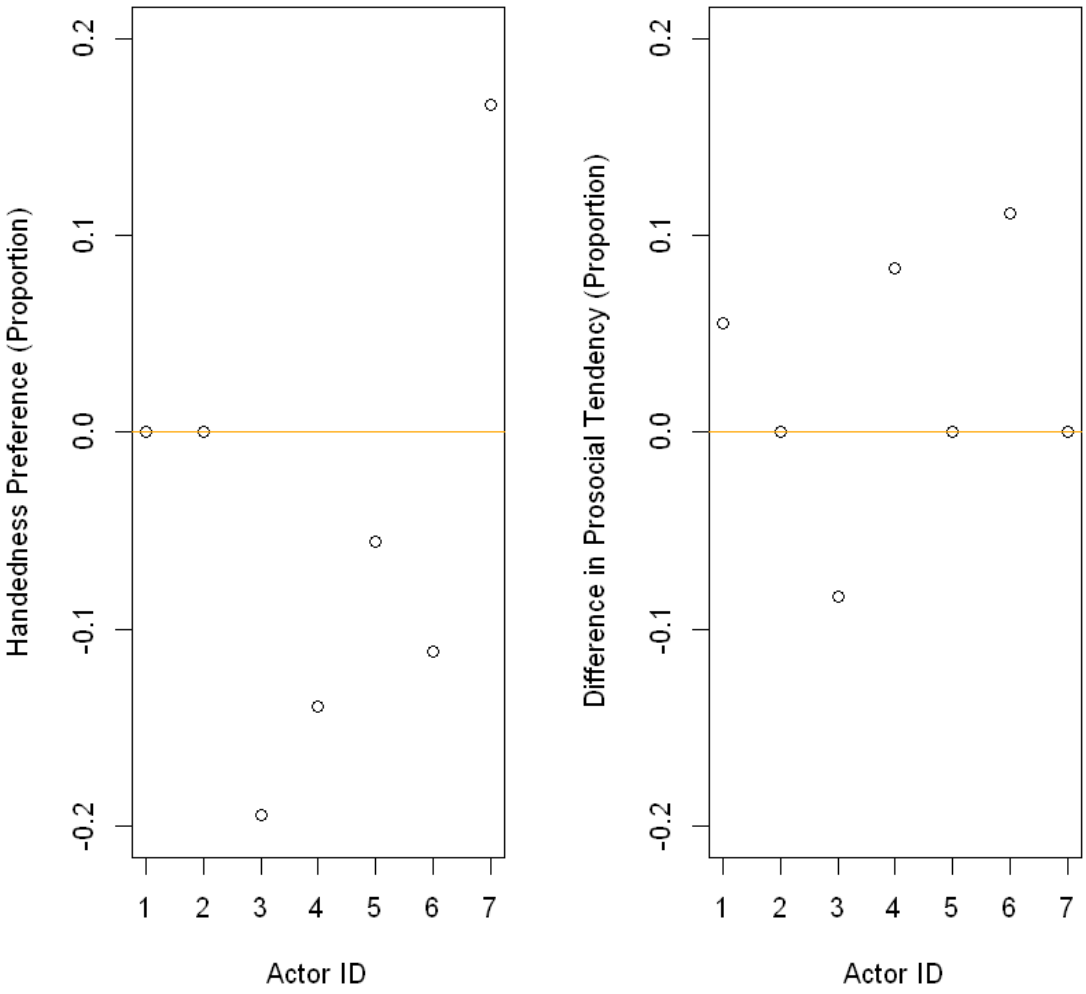
```
par(mfrow=c(1,2))

l = (prop.table(table(single$actor, single$pulled_left), 1) - prop.table(table(pair$act
or, pair$pulled_left), 1))[,1]
plot(l, ylim=c(-.2,.2), xlab="Actor ID",
     ylab="Handedness Preference (Proportion)")
lines(x=c(0,8), y=c(0,0), col="orange")

# If the value is 0, then there is either no preference (50/50) or complete bias (100/
0) to a side
# Negative difference indicates a tendency to pull the left lever compared to the contr
ol,
# and a positive indicates a tendency to pull the right lever compared to the control

diff = (prop.table(table(single$actor, single$is_prosoc), 1) - prop.table(table(pair$ac
tor, pair$is_prosoc), 1))[,1]
plot(diff, ylim=c(-.2,.2), xlab="Actor ID",
     ylab="Difference in Prosocial Tendency (Proportion)")
lines(x=c(0,8), y=c(0,0), col="orange")

# A 0 value inidcates no change in their original choice (i.e their actions are more in
diciative of just pulling the
# lever and not choosing the prosocial option). From observations, there is only a very
minor difference between
# choosing the prosocial option.
```



In [30]:

```
cbind(l, diff)
```

l	diff
0.00000000	0.05555556
0.00000000	0.00000000
-0.19444444	-0.08333333
-0.13888889	0.08333333
-0.05555556	0.00000000
-0.11111111	0.11111111
0.16666667	0.00000000

In [31]:

```
l = (prop.table(table(single$actor, single$pulled_left), 1) - prop.table(table(pair$act
or, pair$pulled_left), 1))[,1]
diff = (prop.table(table(single$actor, single$is_prosoc), 1) - prop.table(table(pair$ac
tor, pair$is_prosoc), 1))[,1]
plot(l - diff, xlab="Actor ID", ylab="Proportion Difference", main="Tendency to be more
prosocial given handedness")
lines(x=c(0,8), y=c(0,0), col="orange")
```