# R_gam_M1_diff_dfs_and_models_superlet20_prevmodels_logpower_

# Load data & libs

```
library(mgcv)
```

```
Loading required package: nlme

This is mgcv 1.9-0. For overview type 'help("mgcv-package")'.
```

```
library(nlme)
library(abind)
library(gratia)
library(ggeffects)
library(emmeans)
library(tidyverse)
```

```
── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
✔ dplyr     1.1.2     ✔ readr     2.1.4
✔ forcats   1.0.0     ✔ stringr   1.5.0
✔ ggplot2   3.4.2     ✔ tibble    3.2.1
✔ lubridate 1.9.2     ✔ tidyr     1.3.0
✔ purrr     1.0.1
── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::collapse() masks nlme::collapse()
✖ dplyr::filter()   masks stats::filter()
✖ dplyr::lag()      masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(itsadug)
```

```
Loading required package: plotfunctions

Attaching package: 'plotfunctions'

The following object is masked from 'package:ggplot2':

    alpha

Loaded package itsadug 2.4 (see 'help("itsadug")' ).
```

```
subjects = c('S01', 'S02', 'S03', 'S04', 'S05', 'S06', 'S07', 'S08', 'S09', 'S10', 'S11', 'S12', 'S13', 'S14', 'S15', 'S16')

print('Load data')
```

```
[1] "Load data"
```

```
df_subj <- data.frame()
for (s in 1:length(subjects)) {
  subj <- subjects[s]
  print(subj)
  df <-read.csv(paste('/run/media/okapi/TOSHIBA EXT/for_r/', subj, '_M1_superlet_20_alpha10_and_12_beta15_and_betavolume_rot_in
  df_subj <- rbind(df_subj,df)
}
```

```
[1] "S01"
[1] "S02"
[1] "S03"
[1] "S04"
[1] "S05"
[1] "S06"
[1] "S07"
[1] "S08"
```

```
[1] "S09"
[1] "S10"
[1] "S11"
[1] "S12"
[1] "S13"
[1] "S14"
[1] "S15"
[1] "S16"
```

```
df_log_centr <- df_subj
df_log_centr$Subject <- as.factor(df_subj$Subject)

df <- df_log_centr
```

# log-powers, unlog-beta volume + z-scored data

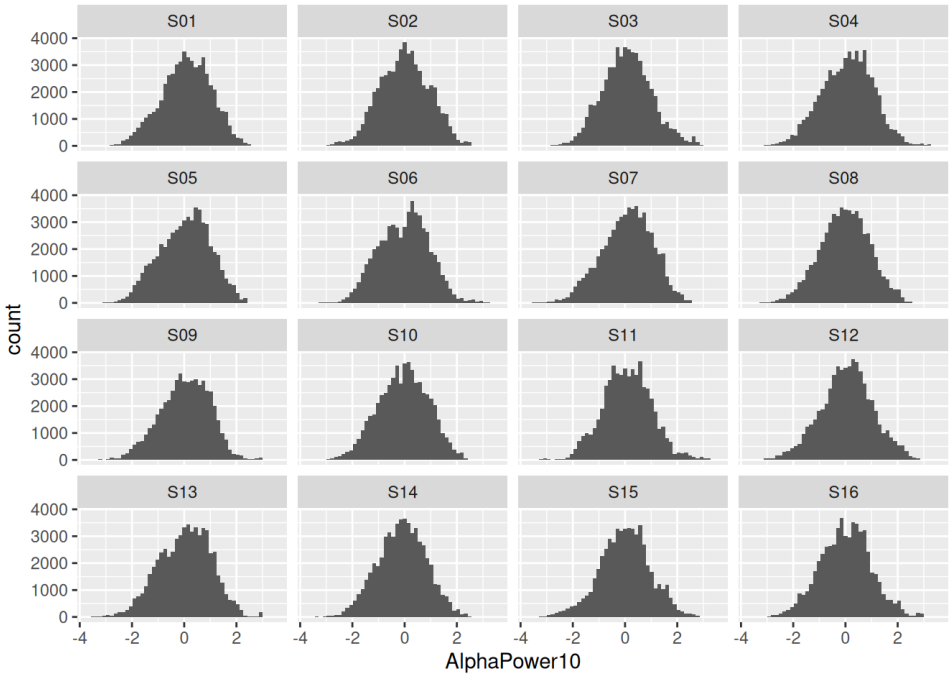
## Visualize data

### Elbow rotation distribution in subjects

```
df %>%
  filter(Rotation < .5) %>%
  distinct(Subject, Trial, Rotation) %>%
  ggplot(aes(x = Rotation, fill = (Rotation<=.05))) +
  facet_wrap(~ Subject, ncol = 4) +
  geom_histogram(breaks = seq(0, .5, .05))
```

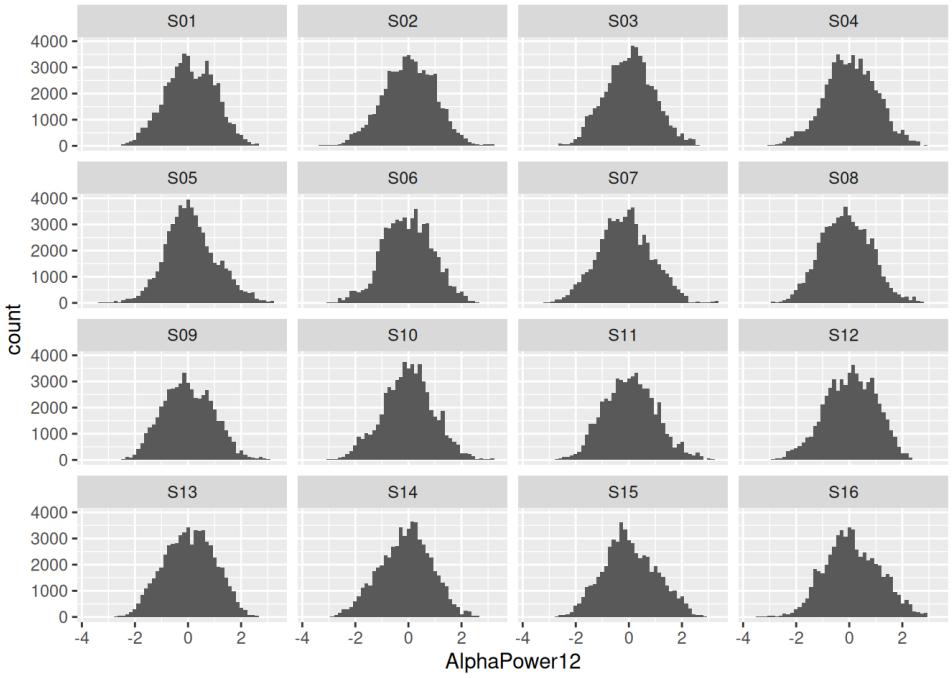

### Alpha Power 10 Hz distribution in subjects

```
df %>%
  filter(Rotation < .5) %>%
  ggplot(aes(x = AlphaPower10)) +
  facet_wrap(~ Subject, ncol = 4) +
  geom_histogram(bins=50)
```



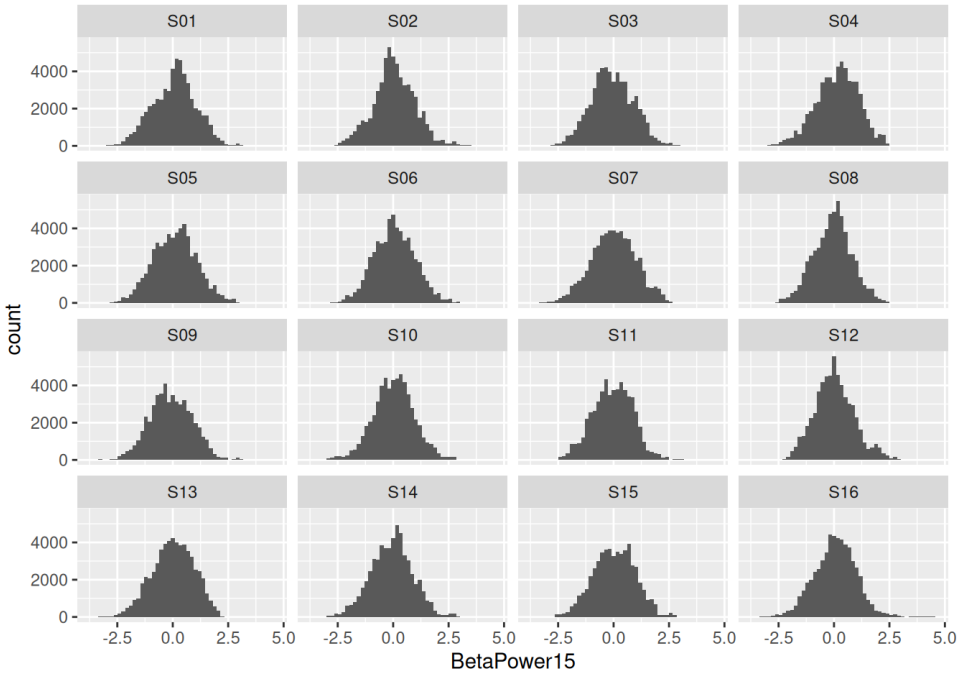

## Alpha Power 12 Hz distribution in subjects

```
df %>%
  filter(Rotation < .5) %>%
  ggplot(aes(x = AlphaPower12)) +
  facet_wrap(~ Subject, ncol = 4) +
  geom_histogram(bins=50)
```



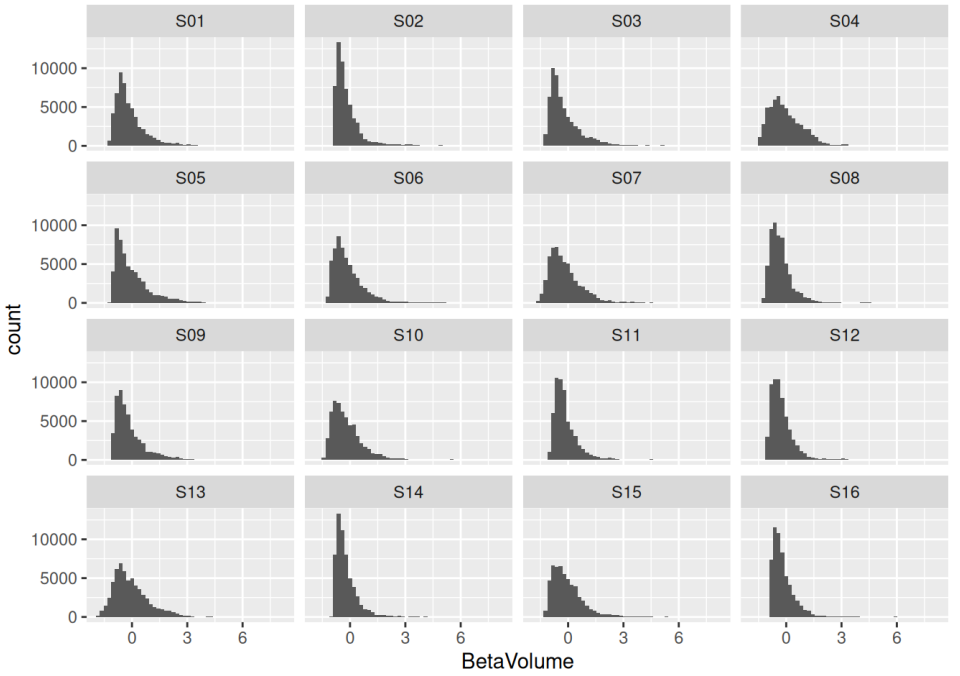

## Beta Power 15 Hz distribution in subjects

```
df %>%
  filter(Rotation < .5) %>%
  ggplot(aes(x = BetaPower15)) +
  facet_wrap(~ Subject, ncol = 4) +
  geom_histogram(bins=50)
```



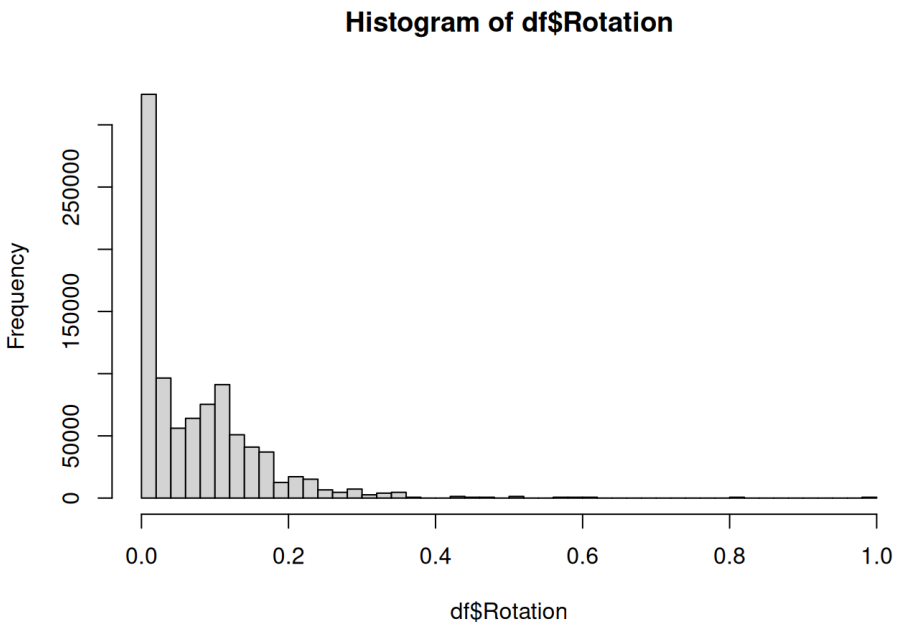

## Beta volume distribution in subjects

```
df %>%
  filter(Rotation < .5) %>%
  ggplot(aes(x = BetaVolume)) +
  facet_wrap(~ Subject, ncol = 4) +
  geom_histogram(bins=50)
```



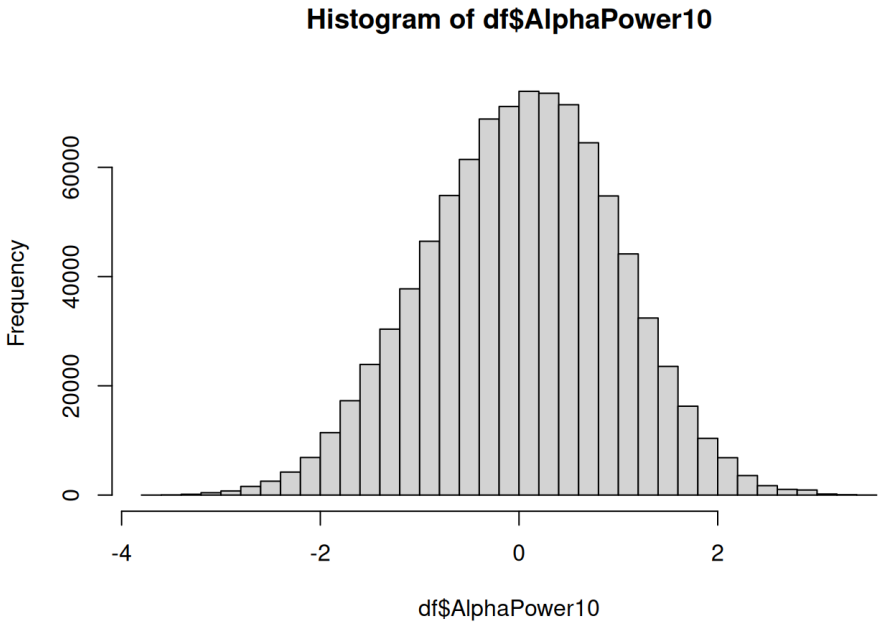

## Elbow rotation overall, histogram:

```
hist(df$Rotation, breaks = 50)
```



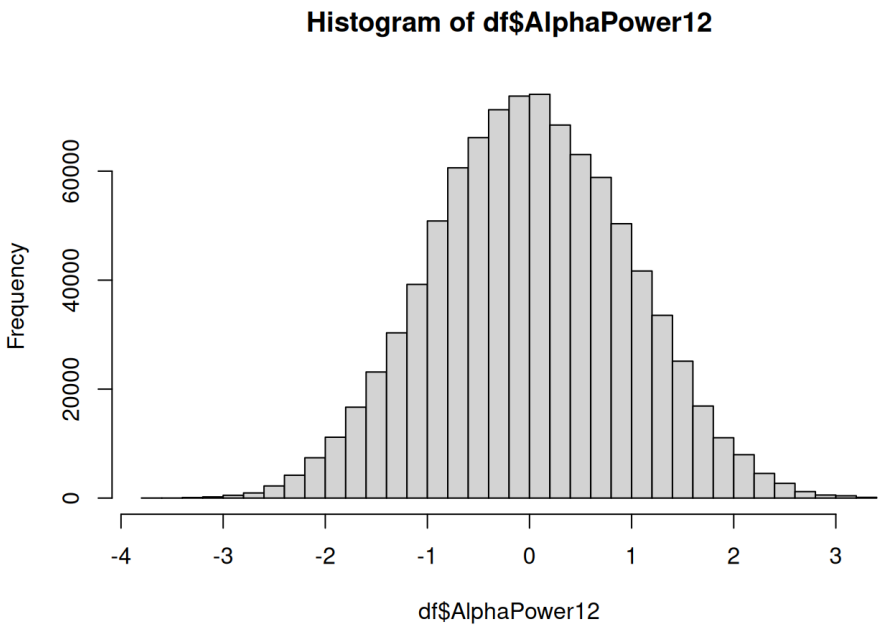

## Alpha Power 10Hz overall, histogram:

```
hist(df$AlphaPower10, breaks = 50)
```



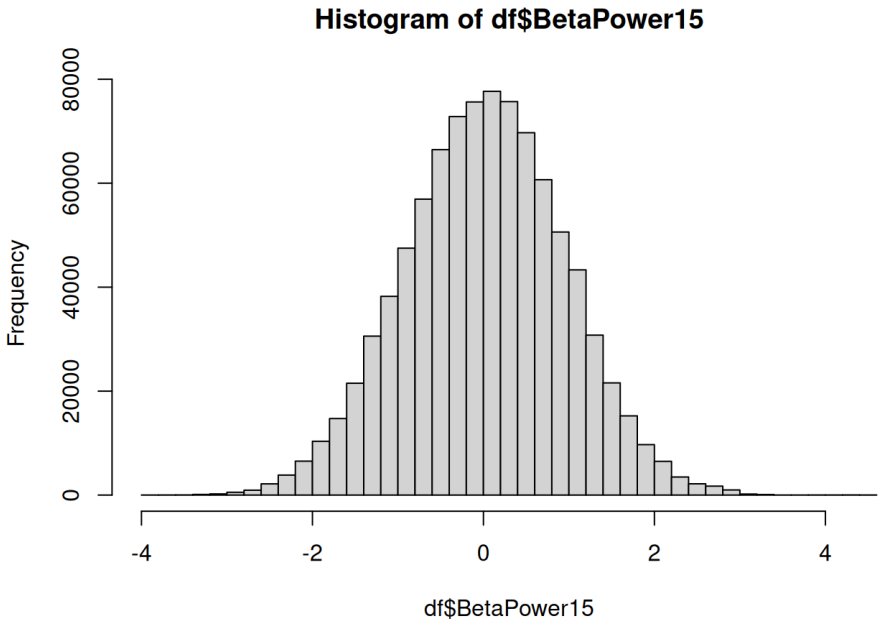

## Alpha Power 12Hz overall, histogram:

```
hist(df$AlphaPower12, breaks = 50)
```



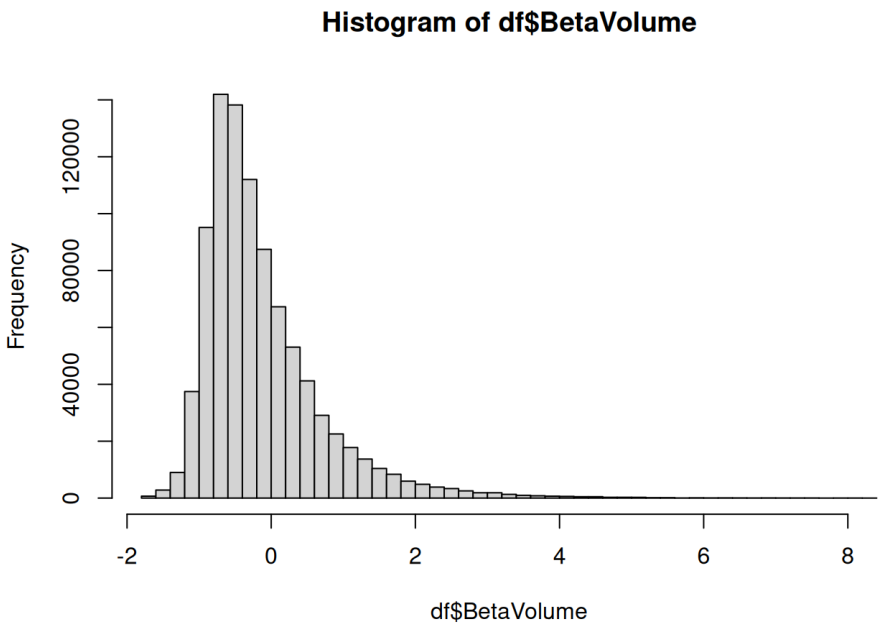

## Beta Power 15Hz overall, histogram:

```
hist(df$BetaPower15, breaks = 50)
```




## Beta Volume overall, histogram:

```
hist(df$BetaVolume, breaks=50)
```

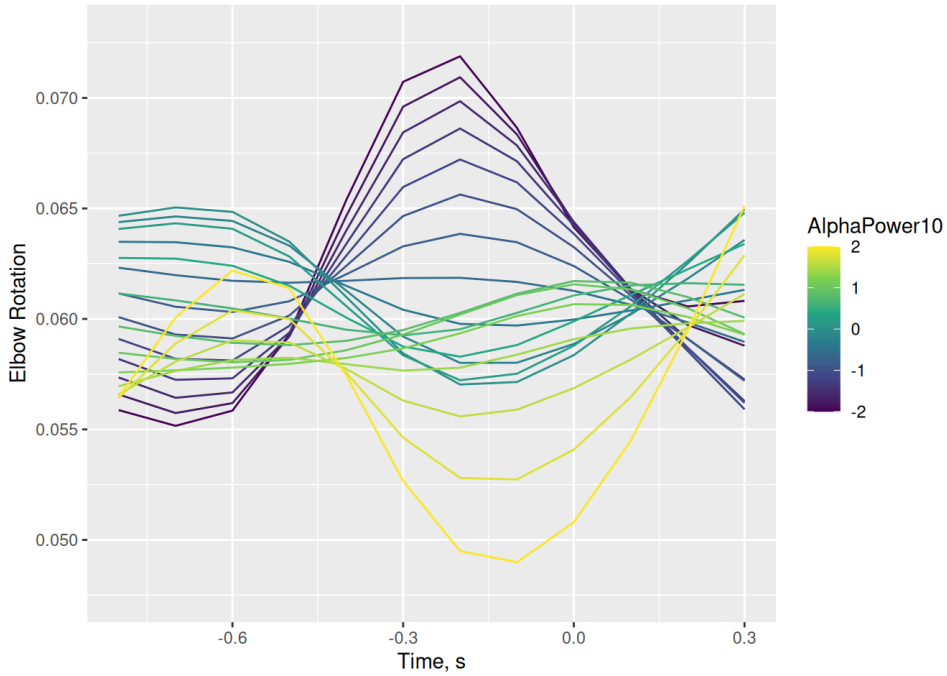



## Run models + visualize predictions

### Alpha 10Hz only + trials: predict Elbow Rotation

```
model_a10 = bam(Rotation ~ s(Trial, Subject, bs='re') +te(AlphaPower10,Time),
                data=filter(df, Rotation<0.5), method='REML')
```

```
(ggemmeans(model_a10, terms = c("Time [-0.8:0.3, by=0.1]", "AlphaPower10 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = predicted
```

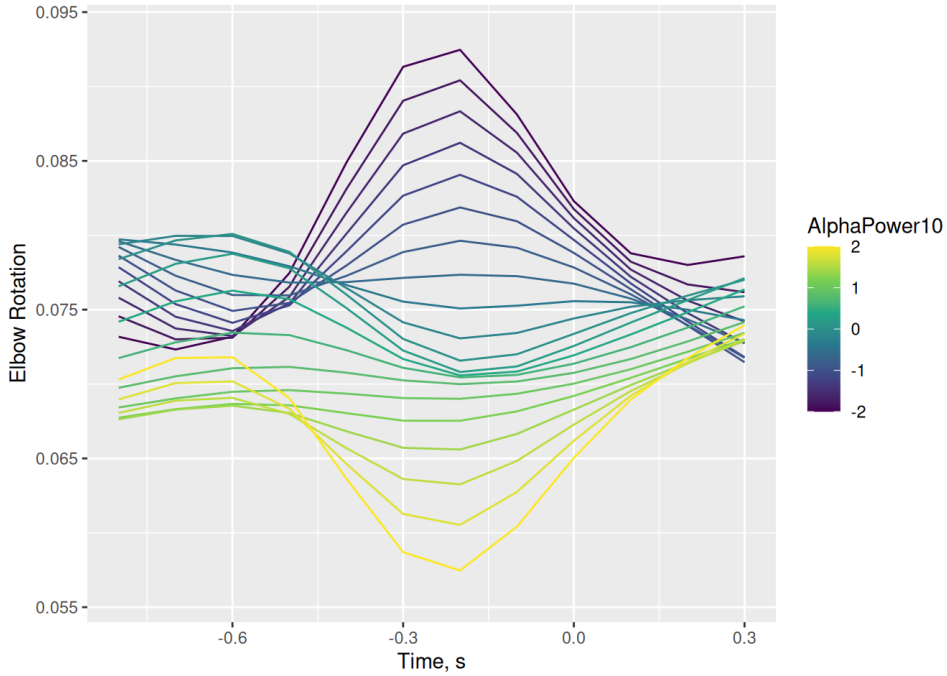



### Alpha 10Hz only + no trials: predict Elbow Rotation

```
model_a10_notri = bam(Rotation ~ te(AlphaPower10,Time), # + s(Time) +ti(AlphaPower,Time),
```

```
              data=filter(df, Rotation<0.5), method='REML')
```

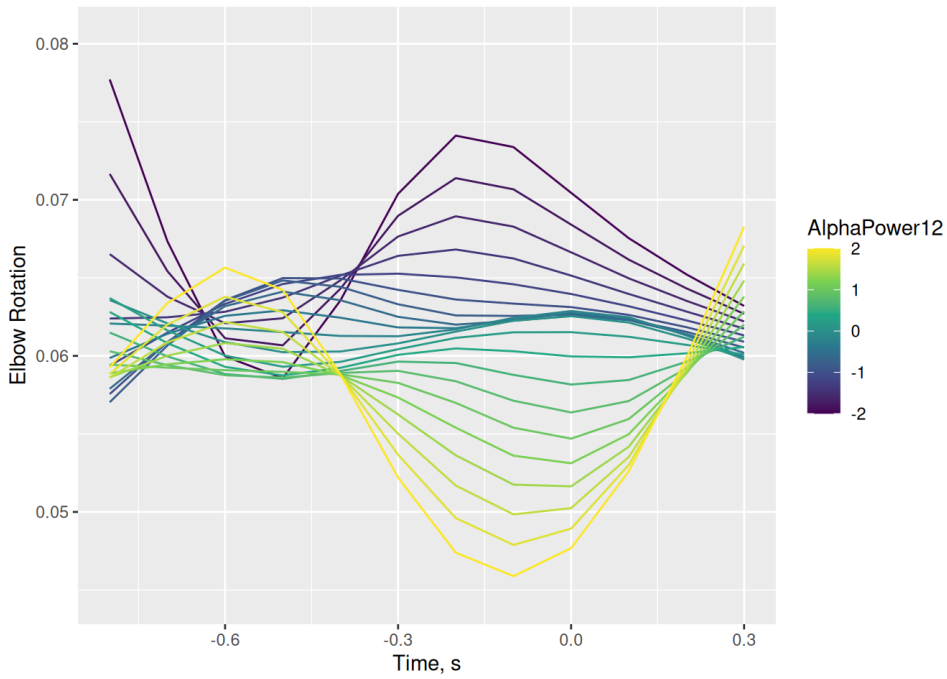
```
(ggemmeans(model_a10_notri, terms = c("Time [-0.8:0.3, by=0.1]", "AlphaPower10 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = pre
```




## Alpha 12Hz only + trials: predict Elbow Rotation

```
model_a12 = bam(Rotation ~ s(Trial, Subject, bs='re') +te(AlphaPower12,Time),
              data=filter(df, Rotation<0.5), method='REML')
```

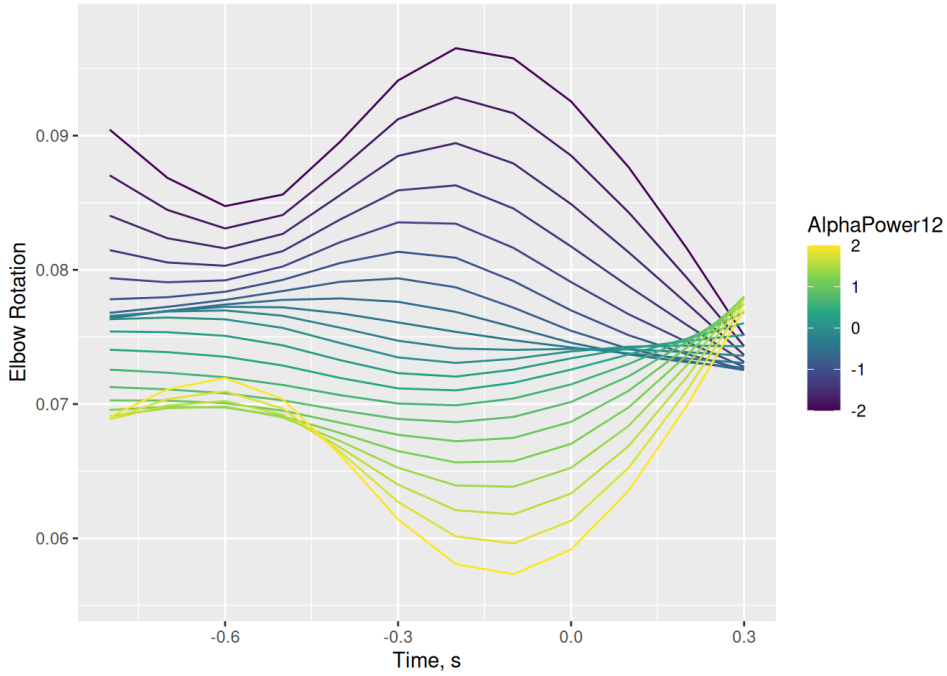
```
(ggemmeans(model_a12, terms = c("Time [-0.8:0.3, by=0.1]", "AlphaPower12 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = predicted
```




## Alpha 12Hz only + no trials: predict Elbow Rotation

```
model_a12_notri = bam(Rotation ~ te(AlphaPower12,Time),
                data=filter(df, Rotation<0.5), method='REML')
```

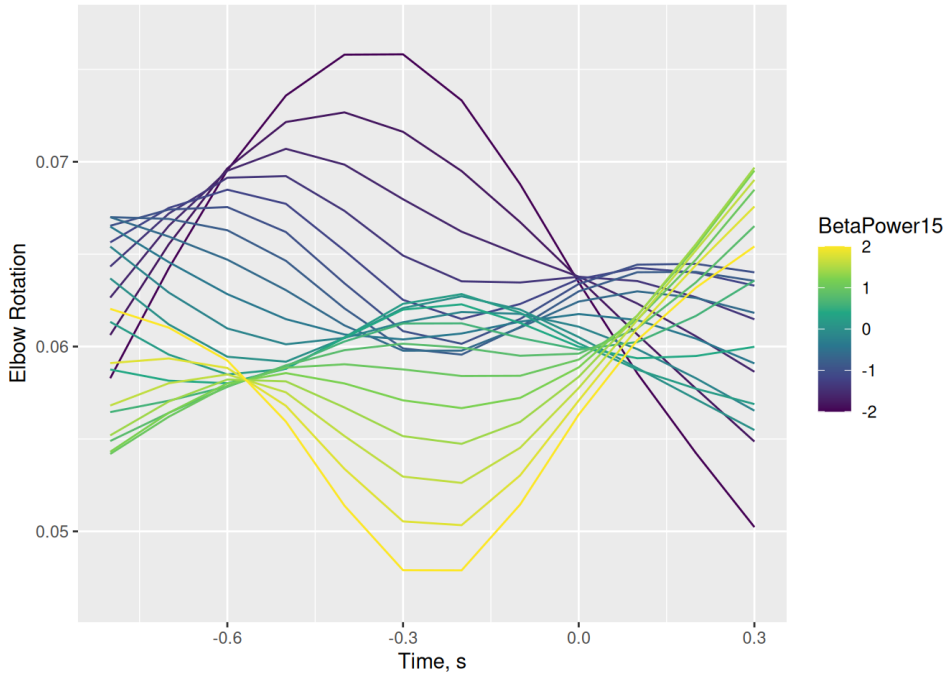
```
(ggemmeans(model_a12_notri, terms = c("Time [-0.8:0.3, by=0.1]", "AlphaPower12 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = pre
```




## Beta 15Hz only + trials: predict Elbow Rotation

```
model_b15 = bam(Rotation ~ s(Trial, Subject, bs='re') +te(BetaPower15,Time),
                data=filter(df, Rotation<0.5), method='REML')
```

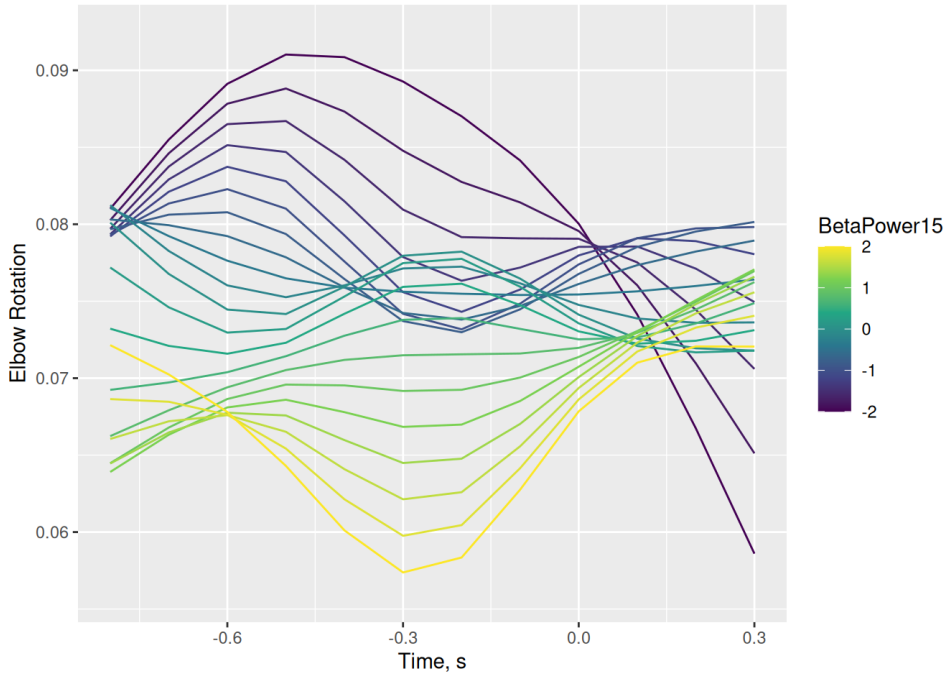
```
(ggemmeans(model_b15, terms = c("Time [-0.8:0.3, by=0.1]", "BetaPower15 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = predicted,
```



## Beta 15Hz only + no trials: predict Elbow Rotation

```
model_b15_notri = bam(Rotation ~ te(BetaPower15,Time),
                data=filter(df, Rotation<0.5), method='REML')
```

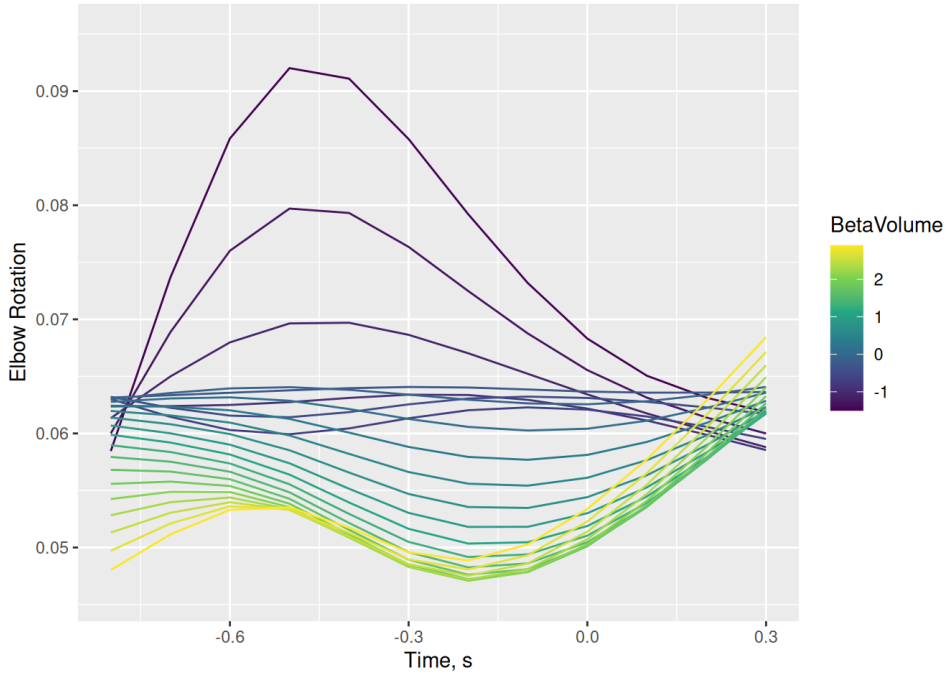
```
(ggemmeans(model_b15_notri, terms = c("Time [-0.8:0.3, by=0.1]", "BetaPower15 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = pred
```




## Beta volume only + trials: predict Elbow Rotation

```
model_b = bam(Rotation ~ s(Trial, Subject, bs='re') +te(BetaVolume,Time), #s(Time) +ti(BetaVolume,Time),
                data=filter(df, Rotation<0.5), method='REML')
```

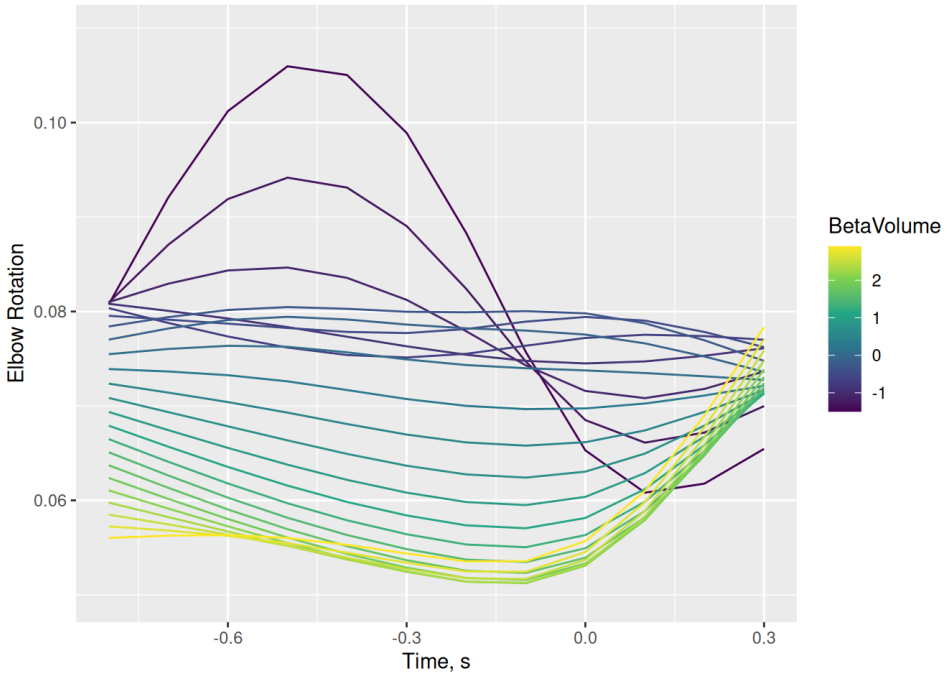
```
(ggemmeans(model_b, terms = c("Time [-0.8:0.3, by=0.1]", "BetaVolume [-1.5:3, by=0.2]"))) %>% ggplot(aes(x = x, y = predicted,
```



## Beta volume only + no trials: predict Elbow Rotation

```
model_b_notri = bam(Rotation ~ te(BetaVolume,Time),
                data=filter(df, Rotation<0.5), method='REML')
```

```
(ggemmeans(model_b_notri, terms = c("Time [-0.8:0.3, by=0.1]", "BetaVolume [-1.5:3, by=0.2]"))) %>% ggplot(aes(x = x, y = predi
```

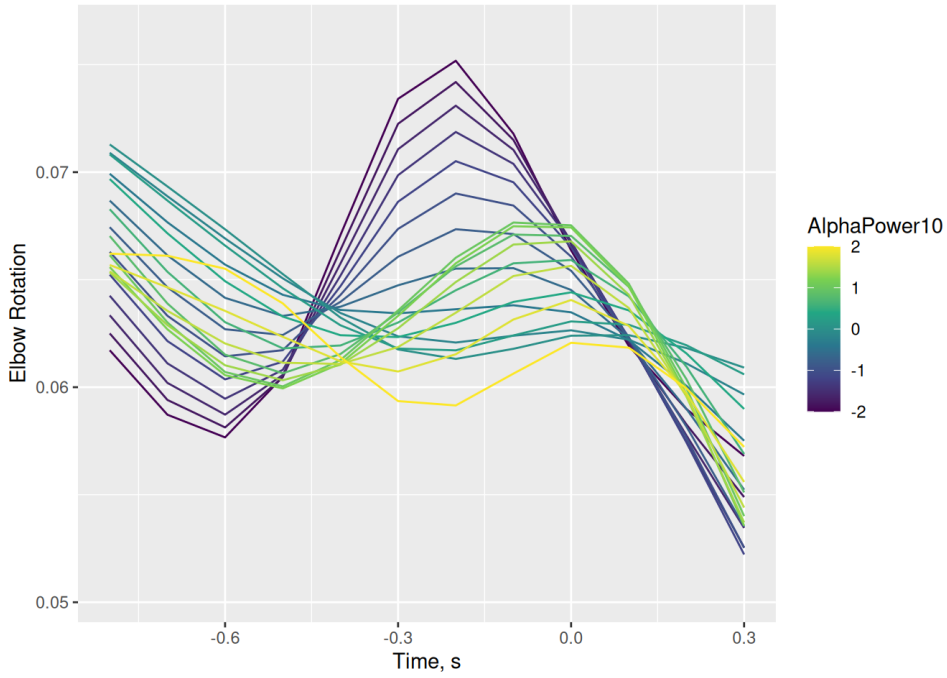



## Alpha & Beta (all factors: alpha 10Hz, alpha 12Hz, beta 15Hz, beta volume) + trials: predict Elbow Rotation

```
model_ab = bam(Rotation ~ s(Trial, Subject, bs='re') +te(AlphaPower10,Time) +te(AlphaPower12,Time) + te(BetaVolume,Time) +te(Be
                data=filter(df, Rotation<0.5), method='REML')
```

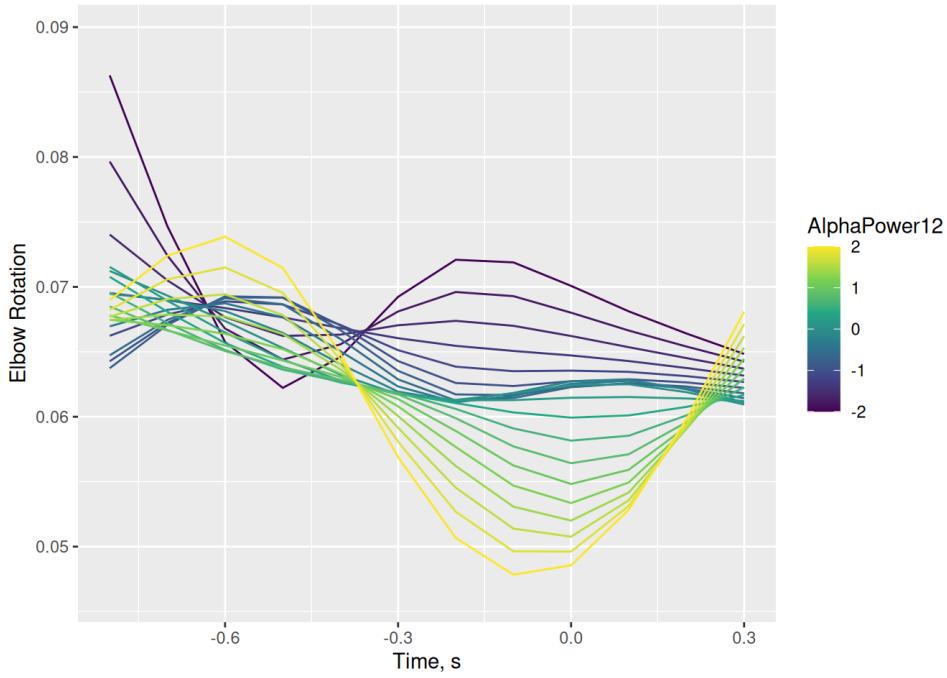
### Alpha Power 10Hz prediction + trials

```
(ggemmeans(model_ab, terms = c("Time [-0.8:0.3, by=0.1]", "AlphaPower10 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = predicted,
```



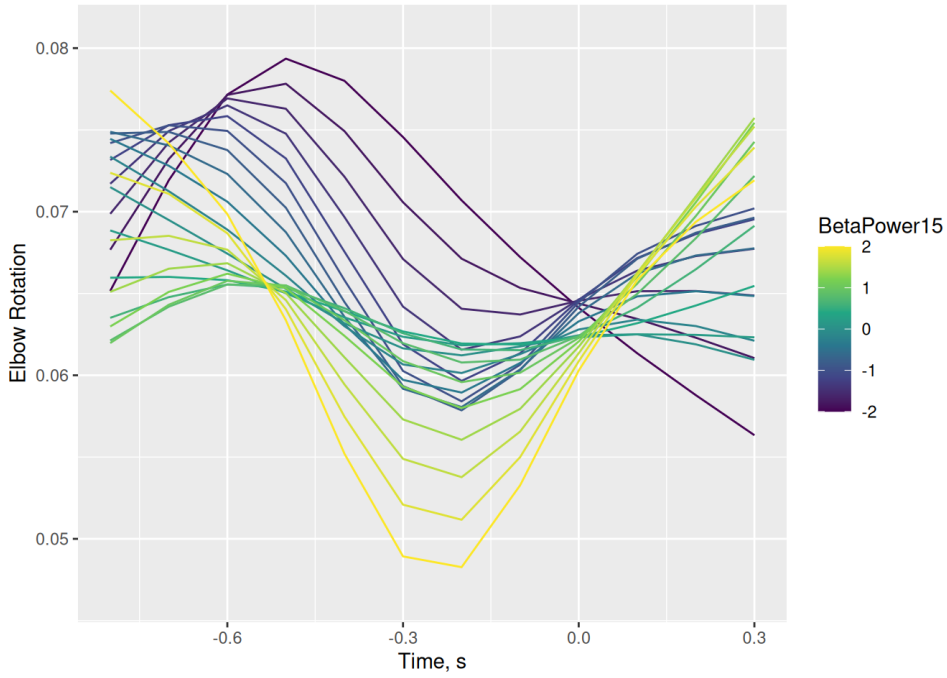

## Alpha Power 12Hz prediction + trials

```
(ggemmeans(model_ab, terms = c("Time [-0.8:0.3, by=0.1]", "AlphaPower12 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = predicted,
```



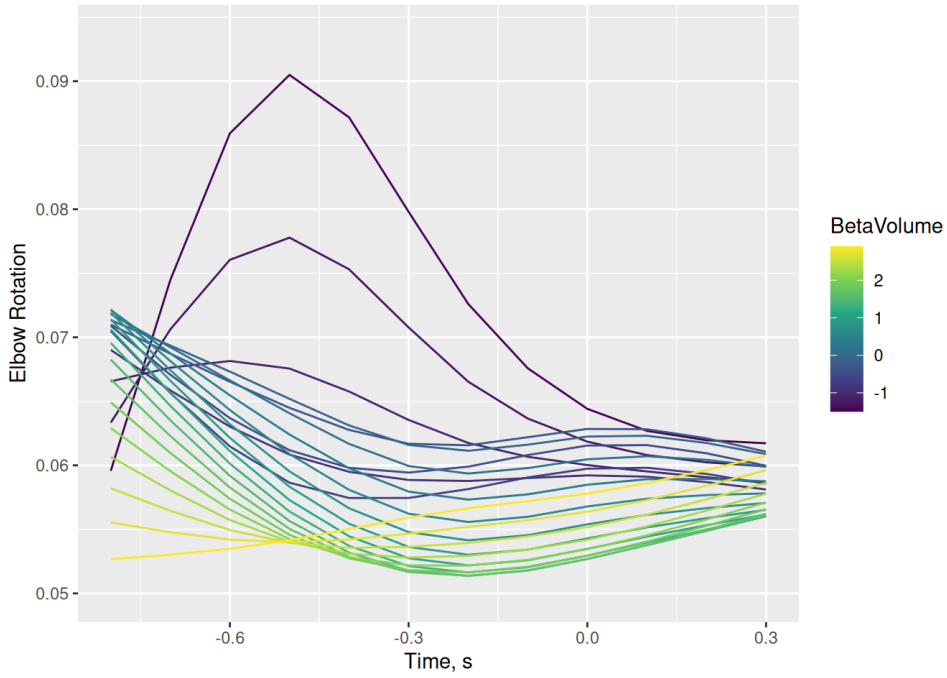

## Beta Power 15Hz prediction + trials

```
(ggemmeans(model_ab, terms = c("Time [-0.8:0.3, by=0.1]", "BetaPower15 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = predicted,
```




### Beta Volume prediction + trials

```
(ggemmeans(model_ab, terms = c("Time [-0.8:0.3, by=0.1]", "BetaVolume [-1.5:3, by=0.2]"))) %>% ggplot(aes(x = x, y = predicted,
```

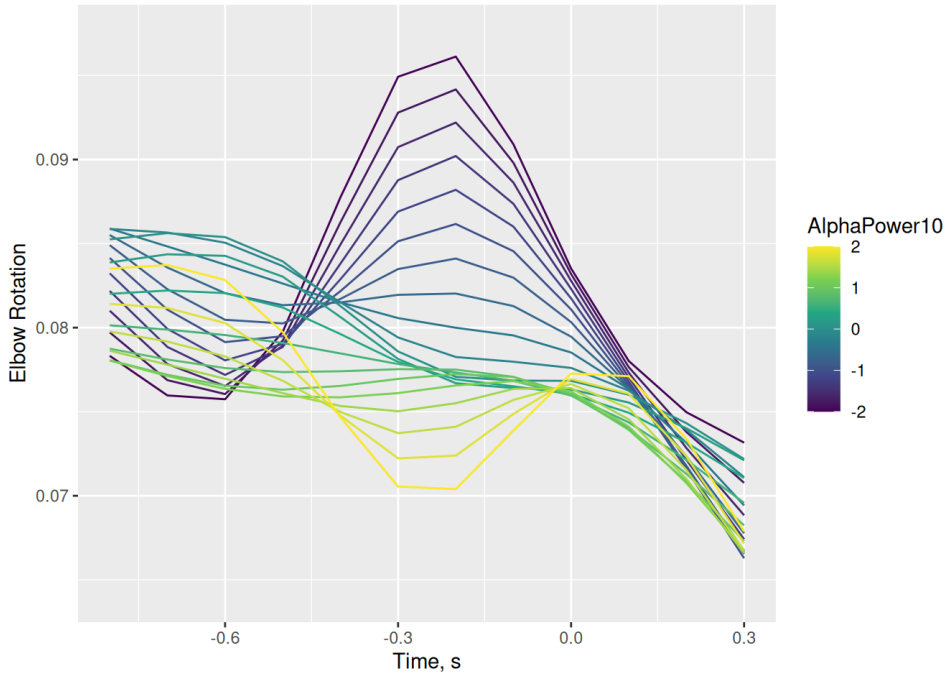



## Alpha & Beta (all factors: alpha 10Hz, alpha 12Hz, beta 15Hz, beta volume) + no trials: predict Elbow Rotation

```
model_ab_notri = bam(Rotation ~ te(AlphaPower10,Time) +te(AlphaPower12,Time) + te(BetaVolume,Time) +te(BetaPower15,Time),
                data=filter(df, Rotation<0.5), method='REML')
```

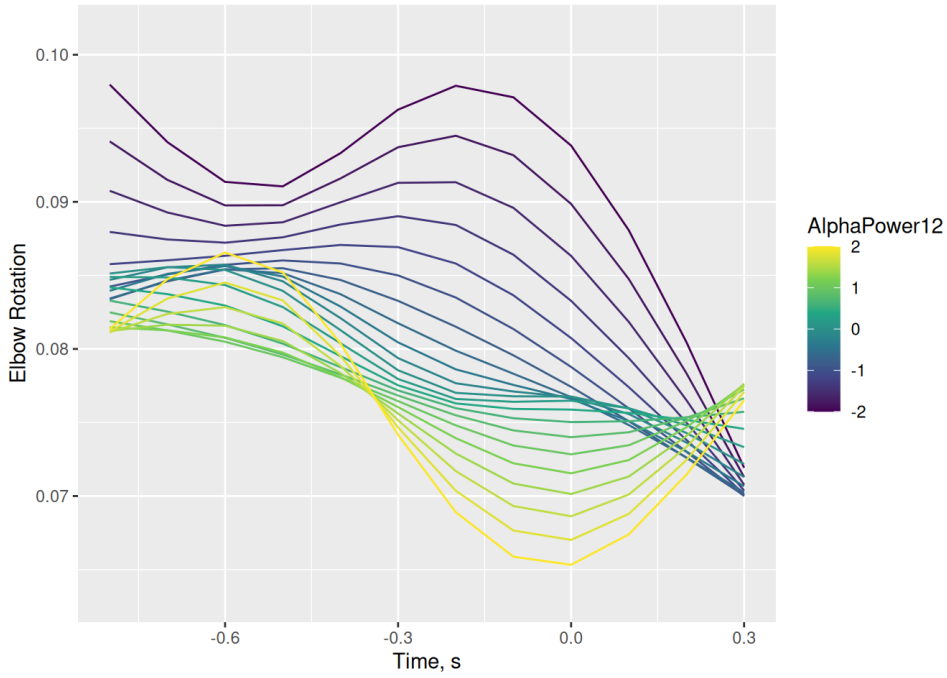
### Alpha Power 10Hz prediction + no trials

```
(ggemmeans(model_ab_notri, terms = c("Time [-0.8:0.3, by=0.1]", "AlphaPower10 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = pred
```



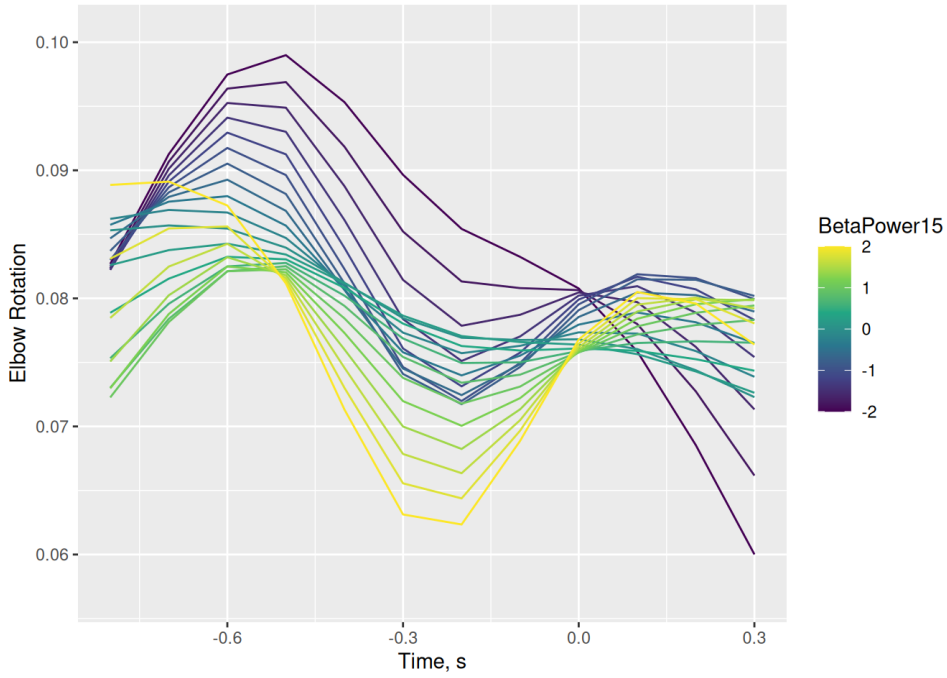

## Alpha Power 12Hz prediction + no trials

```
(ggemmeans(model_ab_notri, terms = c("Time [-0.8:0.3, by=0.1]", "AlphaPower12 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = pred
```



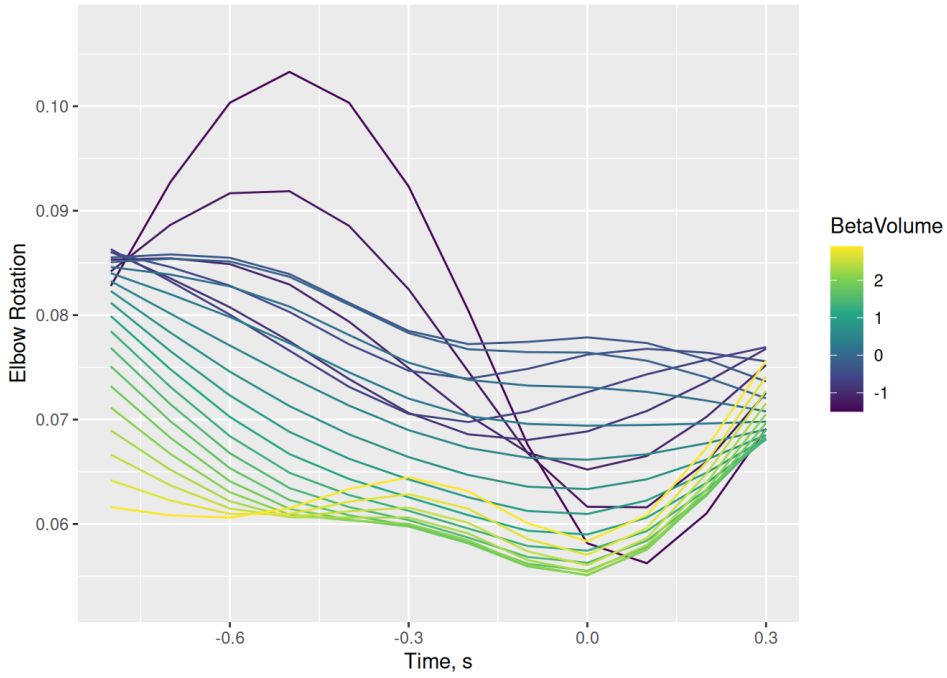

## Beta Power 15Hz prediction + no trials

```
(ggemmeans(model_ab_notri, terms = c("Time [-0.8:0.3, by=0.1]", "BetaPower15 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = predi
```




### Beta Volume prediction + no trials

```
(ggemmeans(model_ab_notri, terms = c("Time [-0.8:0.3, by=0.1]", "BetaVolume [-1.5:3, by=0.2]"))) %>% ggplot(aes(x = x, y = pred
```




## Summary of models

```
summary(model_a10)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ s(Trial, Subject, bs = "re") + te(AlphaPower10, Time)
```

```
Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.0607521  0.0001415   429.4   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                        edf Ref.df       F p-value
s(Trial,Subject)      16.00  16.00 15045.0  <2e-16 ***
te(AlphaPower10,Time) 23.57  23.97    68.7  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.211   Deviance explained = 21.1%
-REML = -1.1673e+06  Scale est. = 0.0045609  n = 914824
```

```
summary(model_a12)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ s(Trial, Subject, bs = "re") + te(AlphaPower12, Time)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.0611803  0.0001424   429.8   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                        edf Ref.df        F p-value
s(Trial,Subject)      16.00  16.00 14997.03  <2e-16 ***
te(AlphaPower12,Time) 23.67  23.98    94.44  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.212   Deviance explained = 21.2%
-REML = -1.1676e+06  Scale est. = 0.0045579  n = 914824
```

```
summary(model_a10_notri)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ te(AlphaPower10, Time)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 7.447e-02  7.944e-05   937.5   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                       edf Ref.df     F p-value
te(AlphaPower10,Time) 23.4  23.94 145.7  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.0038   Deviance explained = 0.382%
-REML = -1.0606e+06  Scale est. = 0.0057611  n = 914824
```

```
summary(model_a12_notri)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ te(AlphaPower12, Time)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 7.453e-02  7.934e-05   939.4   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                       edf Ref.df     F p-value
te(AlphaPower12,Time) 22.9  23.82 197.3  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.0051   Deviance explained = 0.513%
-REML = -1.0612e+06  Scale est. = 0.0057535  n = 914824
```

```
summary(model_b15)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ s(Trial, Subject, bs = "re") + te(BetaPower15, Time)

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.061066   0.000142     430   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                       edf Ref.df       F p-value
s(Trial,Subject)     16.00  16.00 15106.3  <2e-16 ***
te(BetaPower15,Time) 23.63  23.97   122.9  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.212   Deviance explained = 21.2%
-REML = -1.168e+06  Scale est. = 0.0045544  n = 914824
```

```
summary(model_b)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ s(Trial, Subject, bs = "re") + te(BetaVolume, Time)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.0613293  0.0001449   423.3   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                      edf Ref.df       F p-value
s(Trial,Subject)    16.00  16.00 14870.8  <2e-16 ***
te(BetaVolume,Time) 22.85  23.63   117.6  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
R-sq.(adj) =  0.212   Deviance explained = 21.2%
-REML = -1.1679e+06  Scale est. = 0.0045553  n = 914824
```

```
summary(model_b15_notri)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ te(BetaPower15, Time)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 7.451e-02  7.945e-05   937.9   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                       edf Ref.df   F p-value
te(BetaPower15,Time) 23.48  23.93 168  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.00437   Deviance explained = 0.44%
-REML = -1.0608e+06  Scale est. = 0.0057577  n = 914824
```

```
summary(model_b_notri)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ te(BetaVolume, Time)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 7.450e-02  7.929e-05   939.7   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                      edf Ref.df   F p-value
te(BetaVolume,Time) 22.92  23.66 290  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.00744   Deviance explained = 0.746%
-REML = -1.0622e+06  Scale est. = 0.00574   n = 914824
```

```
summary(model_ab)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ s(Trial, Subject, bs = "re") + te(AlphaPower10, Time) +
    te(AlphaPower12, Time) + te(BetaVolume, Time) + te(BetaPower15,
    Time)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.0618434  0.0001462   423.1   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
```

```
                        edf Ref.df        F p-value
s(Trial,Subject)      16.00  16.00 14719.31  <2e-16 ***
te(AlphaPower10,Time) 23.69  23.98    69.28  <2e-16 ***
te(AlphaPower12,Time) 19.74  19.99    85.01  <2e-16 ***
te(BetaVolume,Time)   18.78  20.00   180.02  <2e-16 ***
te(BetaPower15,Time)  19.76  20.00   175.55  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.217   Deviance explained = 21.7%
-REML = -1.1706e+06  Scale est. = 0.0045266  n = 914824
```

```
summary(model_ab_notri)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ te(AlphaPower10, Time) + te(AlphaPower12, Time) +
    te(BetaVolume, Time) + te(BetaPower15, Time)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 7.449e-02  7.917e-05   940.9   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                        edf Ref.df      F p-value
te(AlphaPower10,Time) 23.17  23.82  81.68  <2e-16 ***
te(AlphaPower12,Time) 19.11  20.00 128.18  <2e-16 ***
te(BetaVolume,Time)   19.46  20.00 233.16  <2e-16 ***
te(BetaPower15,Time)  19.79  20.00 117.60  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.0157   Deviance explained = 1.58%
-REML = -1.0659e+06  Scale est. = 0.0056919  n = 914824
```

## AIC model comparison

```
AIC(model_a10) - AIC(model_a10_notri)
```

```
[1] -213690.5
```

```
AIC(model_a12) - AIC(model_a12_notri)
```

```
[1] -213090.3
```

```
AIC(model_b15) - AIC(model_b15_notri)
```

```
[1] -214462.8
```

```
AIC(model_b) - AIC(model_b_notri)
```

```
[1] -211468.1
```

```
AIC(model_a12) - AIC(model_a10)
```

```
[1] -602.1293
```

```
AIC(model_b15) - AIC(model_a12)
```

```
[1] -700.1991
```

```
AIC(model_b15) - AIC(model_b)
```

```
[1] -172.9424
```

```
AIC(model_ab) - AIC(model_b15)
```

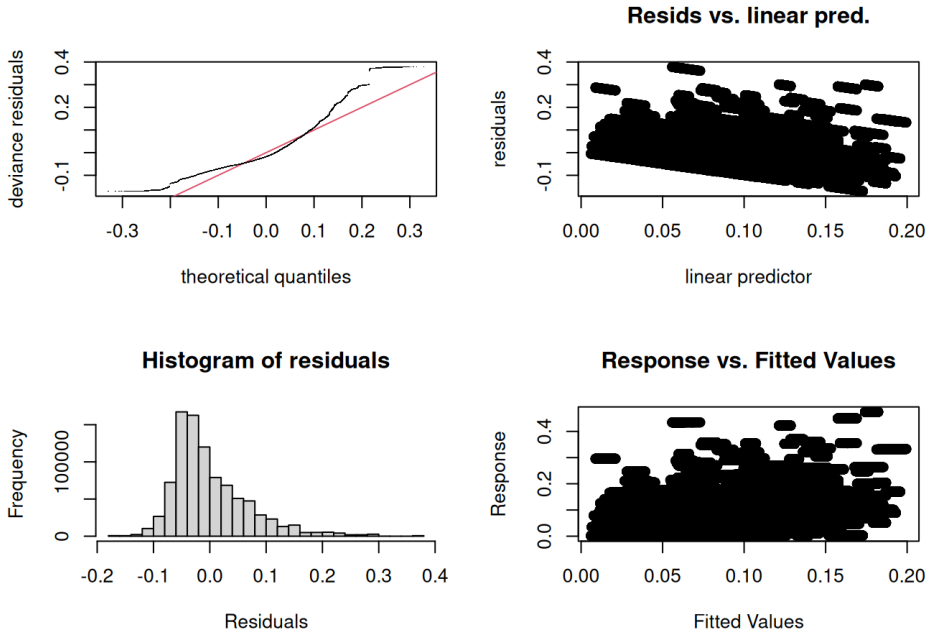
```
[1] -5544.361
```

```
AIC(model_ab) - AIC(model_ab_notri)
```

```
[1] -209556.5
```

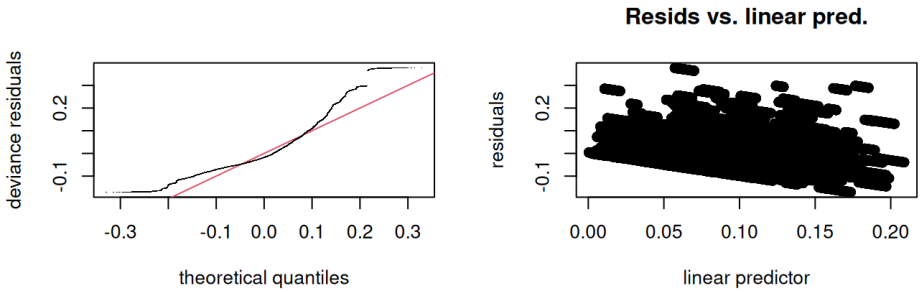
## Check models

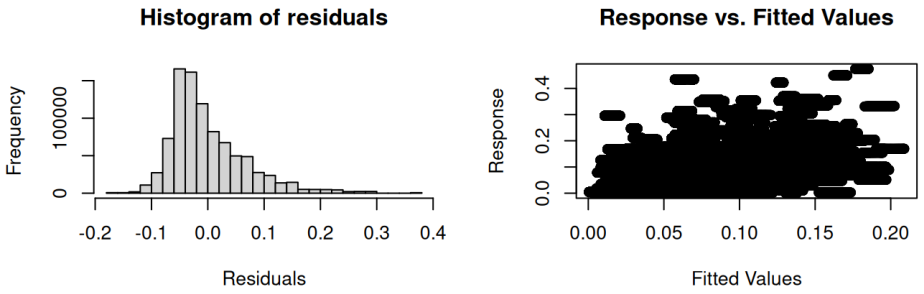
```
gam.check(model_a10)
```




```
Method: REML   Optimizer: outer newton
full convergence after 5 iterations.
Gradient range [-0.3156677,0.02457995]
(score -1167322 & scale 0.004560885).
Hessian positive definite, eigenvalue range [3.612177,457410.3].
Model rank =  41 / 41

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(Trial,Subject)      16.0 16.0      NA      NA
te(AlphaPower10,Time) 24.0 23.6    0.98   0.035 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

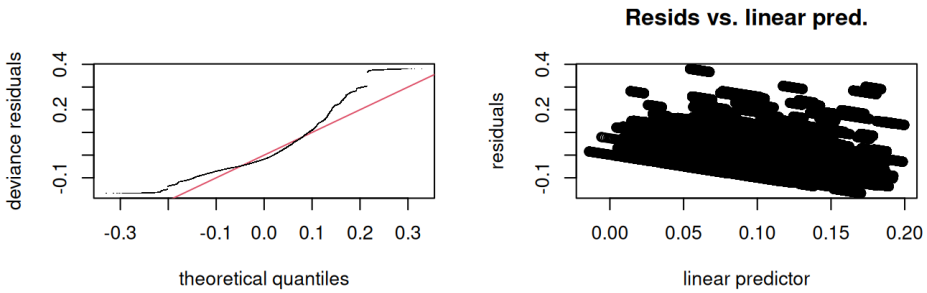
```
gam.check(model_a12)
```

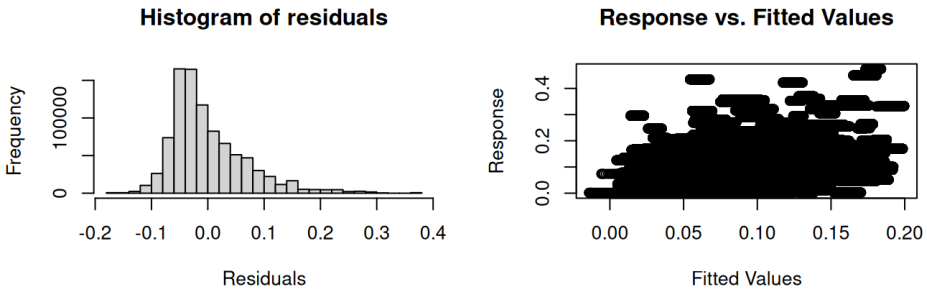



```
Method: REML   Optimizer: outer newton
full convergence after 8 iterations.
Gradient range [-0.1710393,0.1471913]
(score -1167620 & scale 0.004557884).
Hessian positive definite, eigenvalue range [3.412983,457410.2].
Model rank =  41 / 41

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(Trial,Subject)      16.0 16.0      NA      NA
te(AlphaPower12,Time) 24.0 23.7    0.98    0.02 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

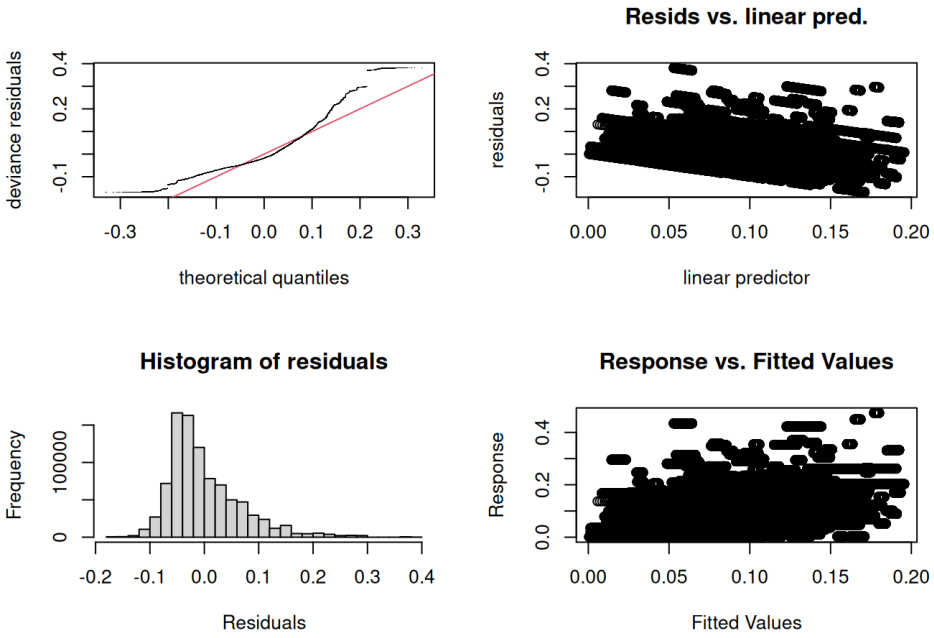
```
gam.check(model_b15)
```



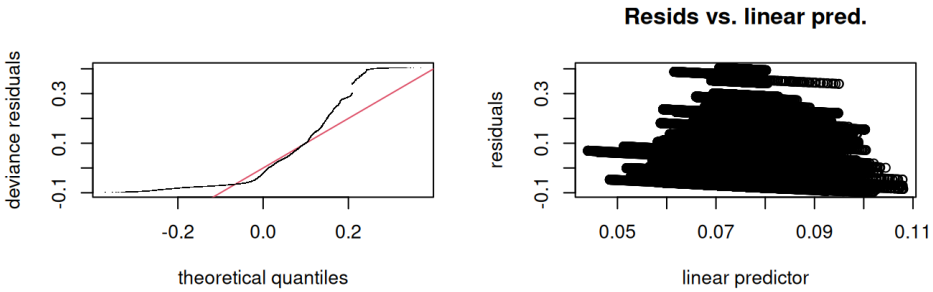
```
Method: REML   Optimizer: outer newton
full convergence after 7 iterations.
Gradient range [-0.0007460721,6.555254e-08]
(score -1167968 & scale 0.004554397).
Hessian positive definite, eigenvalue range [3.524015,457410].
Model rank =  41 / 41

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(Trial,Subject)     16.0 16.0      NA      NA
te(BetaPower15,Time) 24.0 23.6       1    0.61
```

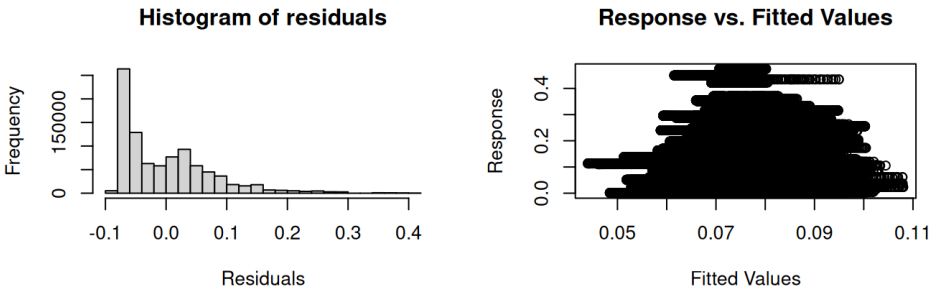
```
gam.check(model_b)
```




```
Method: REML   Optimizer: outer newton
full convergence after 7 iterations.
Gradient range [-0.1815775,1.00063e-07]
(score -1167888 & scale 0.00455526).
Hessian positive definite, eigenvalue range [3.148679,457410.2].
Model rank =  41 / 41

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(Trial,Subject)    16.0 16.0      NA      NA
te(BetaVolume,Time) 24.0 22.9    0.98     0.1
```

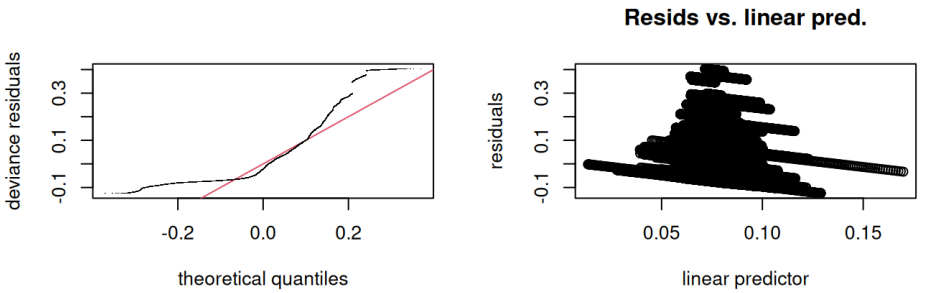
```
gam.check(model_a10_notri)
```

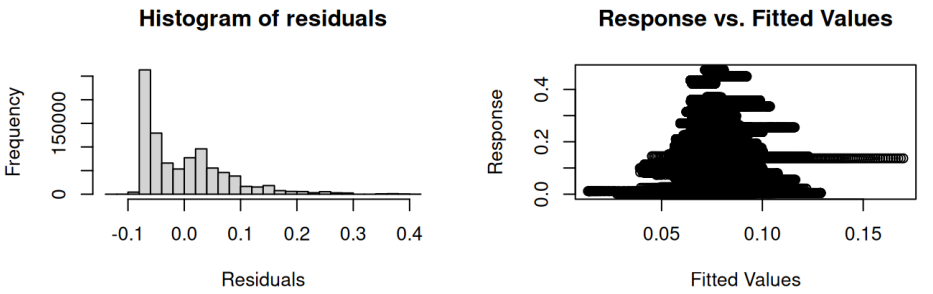



```
Method: REML   Optimizer: outer newton
full convergence after 6 iterations.
Gradient range [-0.0001424957,1.901265e-07]
(score -1060550 & scale 0.005761055).
Hessian positive definite, eigenvalue range [3.200347,457410].
Model rank =  25 / 25

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
te(AlphaPower10,Time) 24.0 23.4    0.98   0.055 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

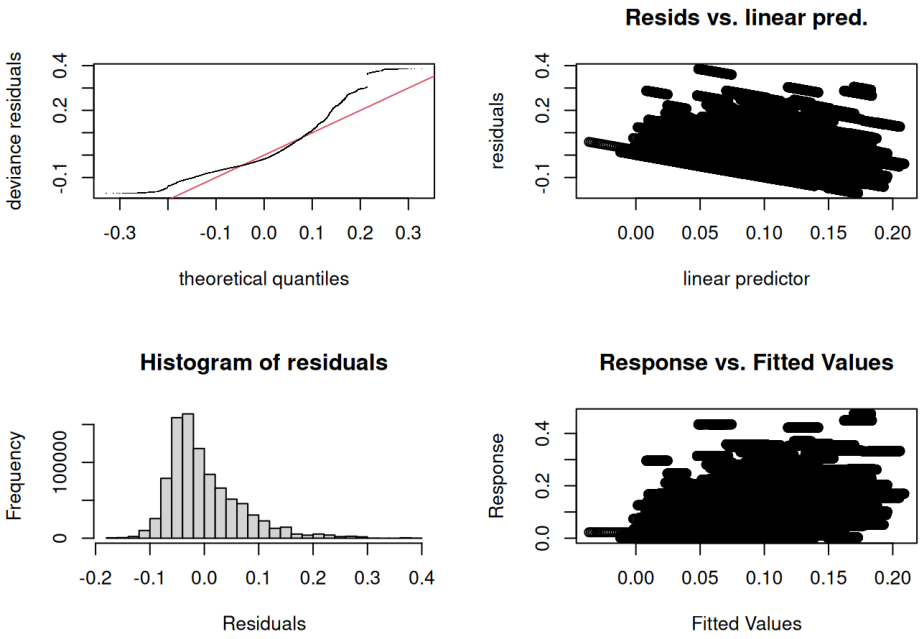
```
gam.check(model_b15_notri)
```



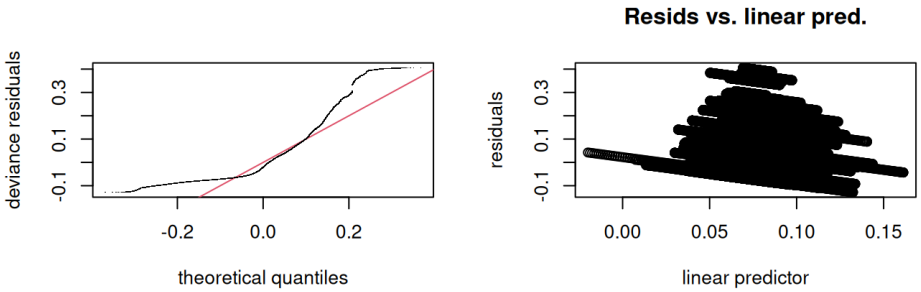
```
Method: REML   Optimizer: outer newton
full convergence after 7 iterations.
Gradient range [-0.0001434662,5.341796e-08]
(score -1060809 & scale 0.005757719).
Hessian positive definite, eigenvalue range [3.64952,457410].
Model rank =  25 / 25

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
te(BetaPower15,Time) 24.0 23.5    0.99    0.12
```

```
gam.check(model_ab)
```



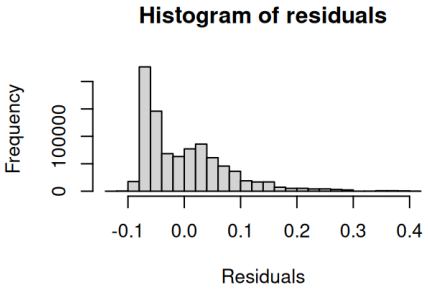

```
Method: REML   Optimizer: outer newton
full convergence after 12 iterations.
Gradient range [-0.7296576,0.7293553]
(score -1170586 & scale 0.004526585).
Hessian positive definite, eigenvalue range [1.998968,457410.2].
Model rank =  101 / 101

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(Trial,Subject)      16.0 16.0      NA      NA
te(AlphaPower10,Time) 24.0 23.7    0.99    0.22
te(AlphaPower12,Time) 20.0 19.7    0.99    0.16
te(BetaVolume,Time)   20.0 18.8    1.00    0.38
te(BetaPower15,Time)  20.0 19.8    1.01    0.73
```

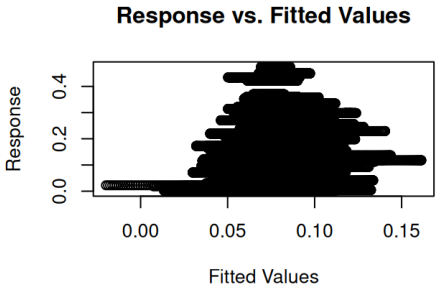
```
gam.check(model_ab_notri)
```










```
Method: REML   Optimizer: outer newton
full convergence after 8 iterations.
Gradient range [-0.2933753,0.06858606]
(score -1065898 & scale 0.005691945).
Hessian positive definite, eigenvalue range [1.52159,457410.3].
Model rank =  85 / 85

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
te(AlphaPower10,Time) 24.0 23.2    0.99    0.12
te(AlphaPower12,Time) 20.0 19.1    1.01    0.81
te(BetaVolume,Time)   20.0 19.5    1.01    0.78
te(BetaPower15,Time)  20.0 19.8    1.01    0.85
```