R_gam_M1_diff_dfs_and_models_superlet20_prevmodels_logpower_alle

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_inh_-08to03_
  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"
```

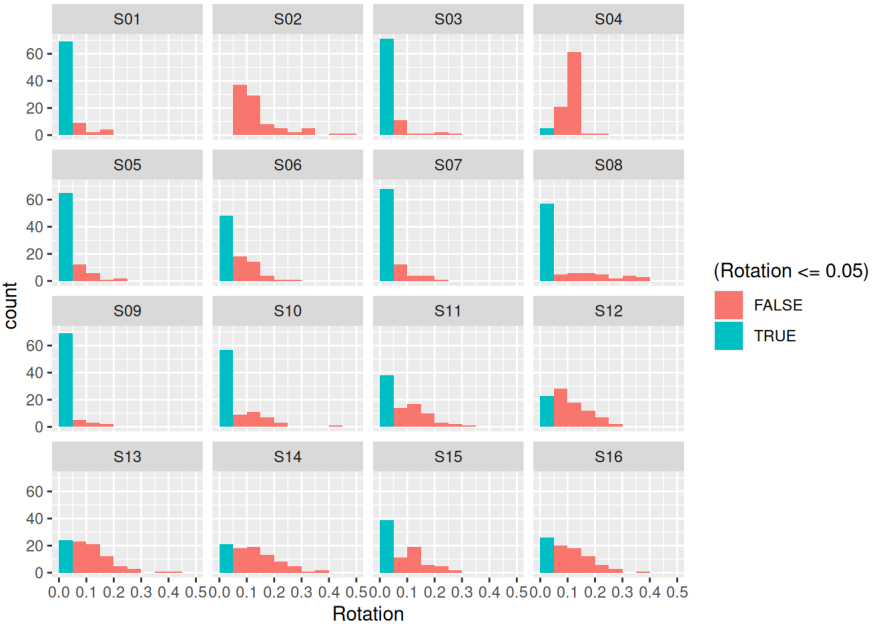
```
df <- df_subj
df$Subject <- as.factor(df$Subject)
```

log-powers, unlog-beta volume + z-scored data + without excluding large elbow rotations

Visualize data

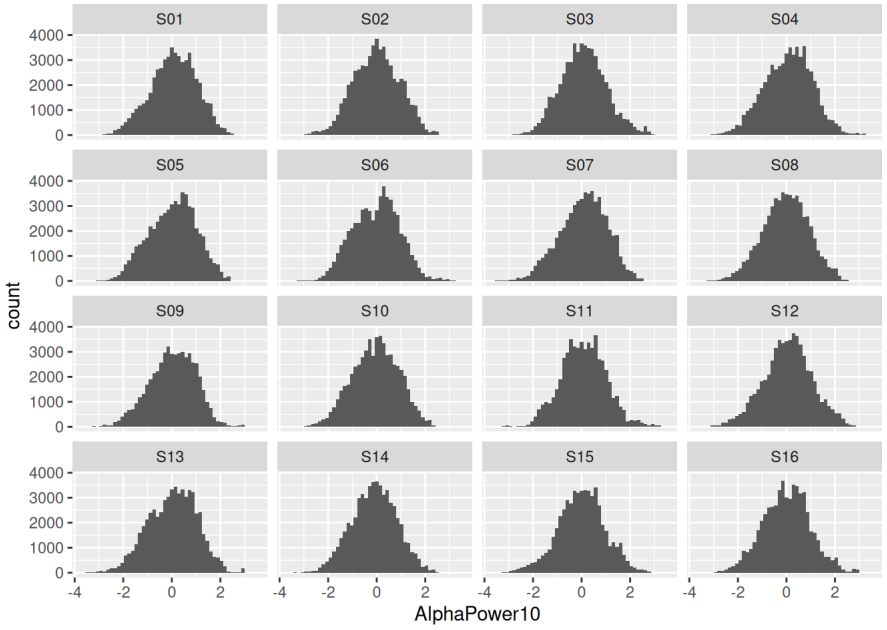
Elbow rotation distribution in subjects

```
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))
```



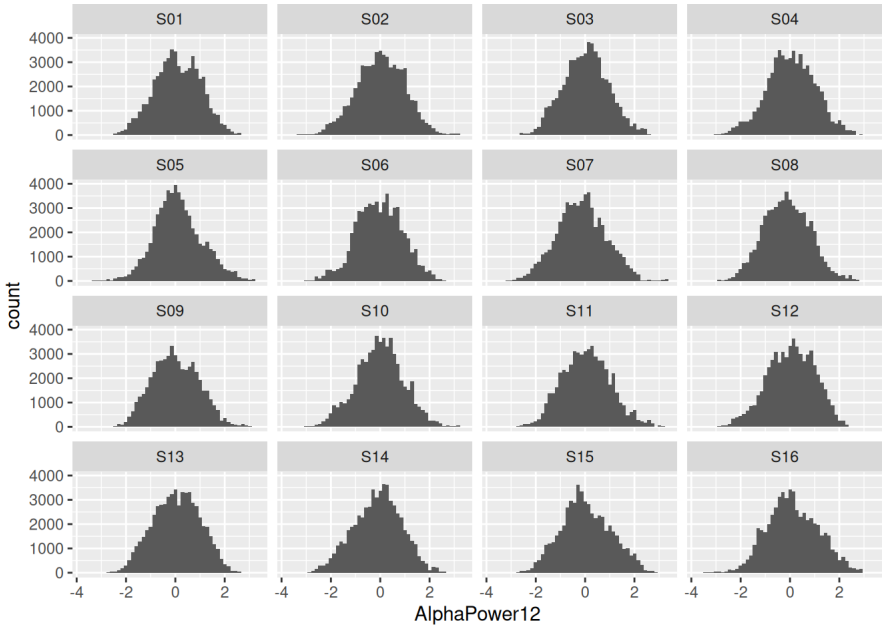
Alpha Power 10 Hz distribution in subjects

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



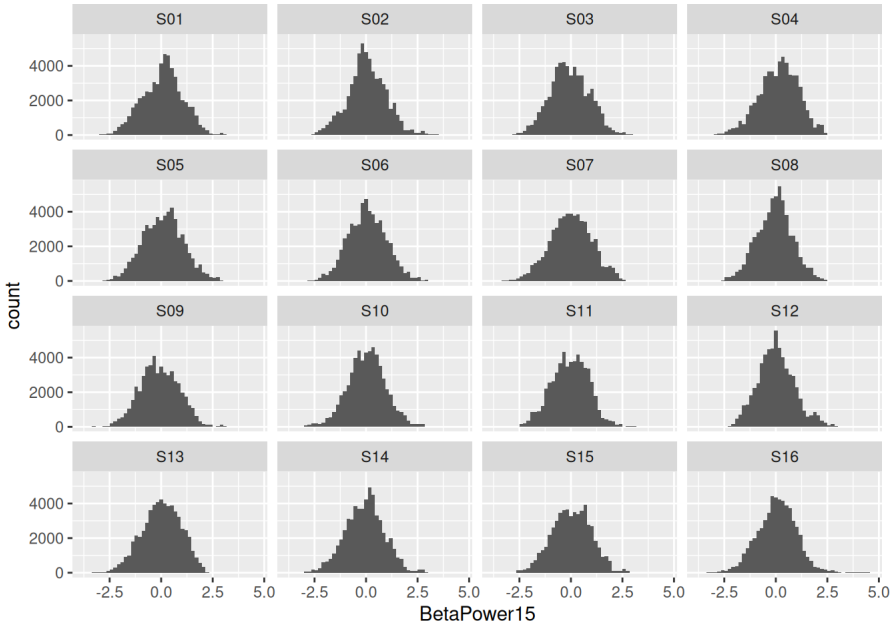
Alpha Power 12 Hz distribution in subjects

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



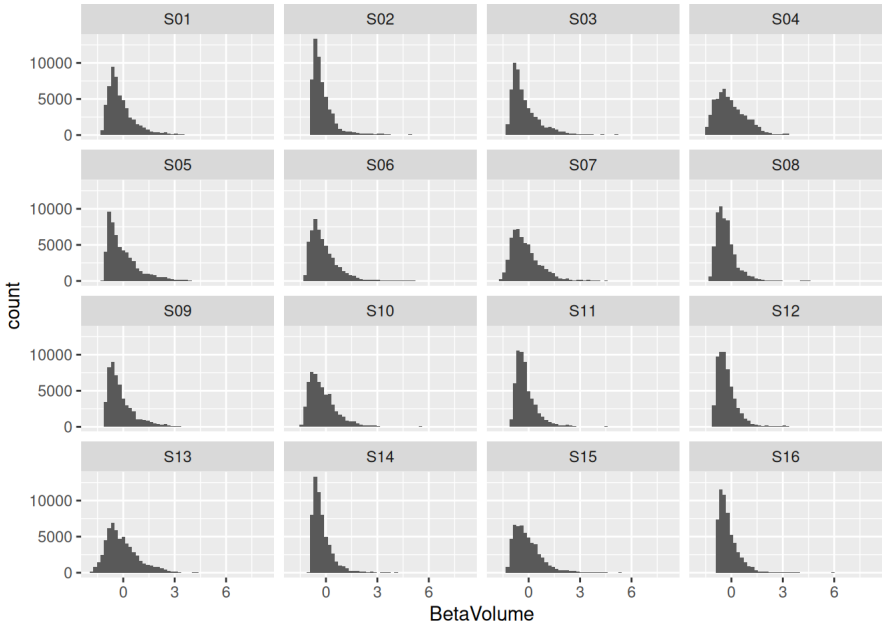
Beta Power 15 Hz distribution in subjects

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



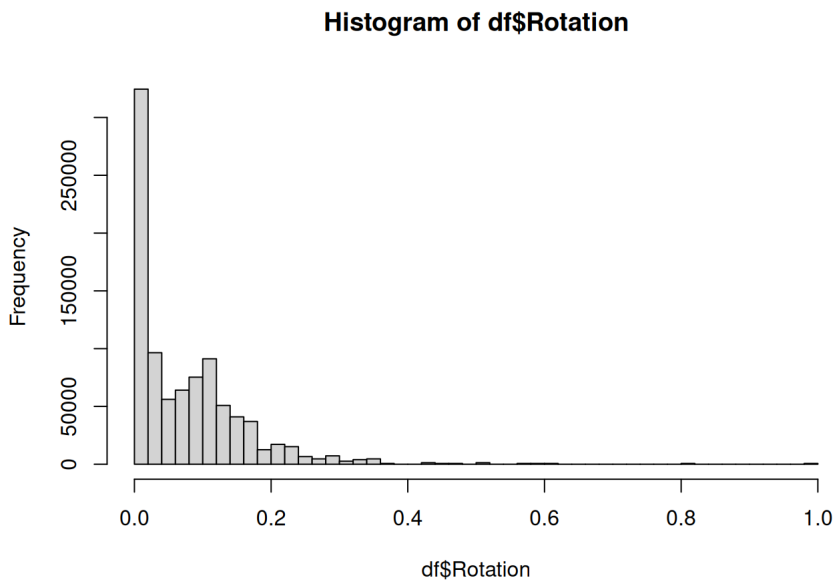
Beta volume distribution in subjects

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



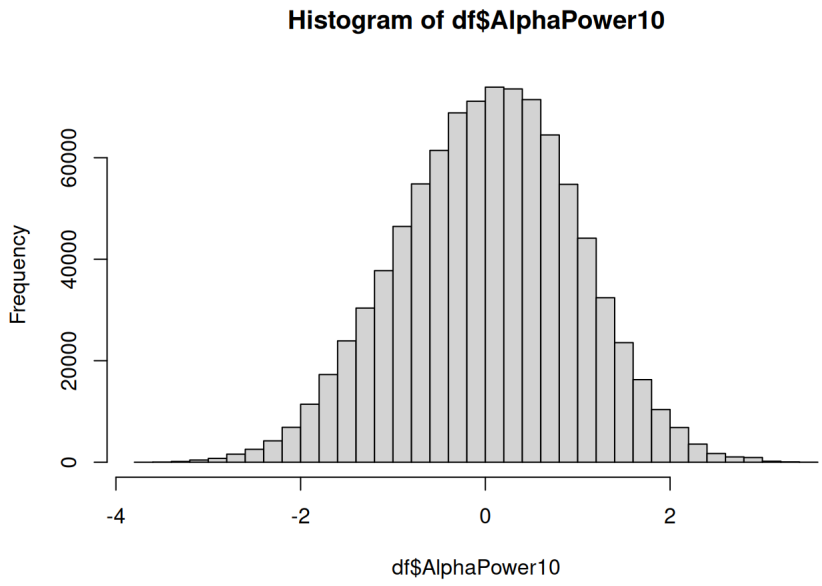
Elbow rotation overall, histogram:

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



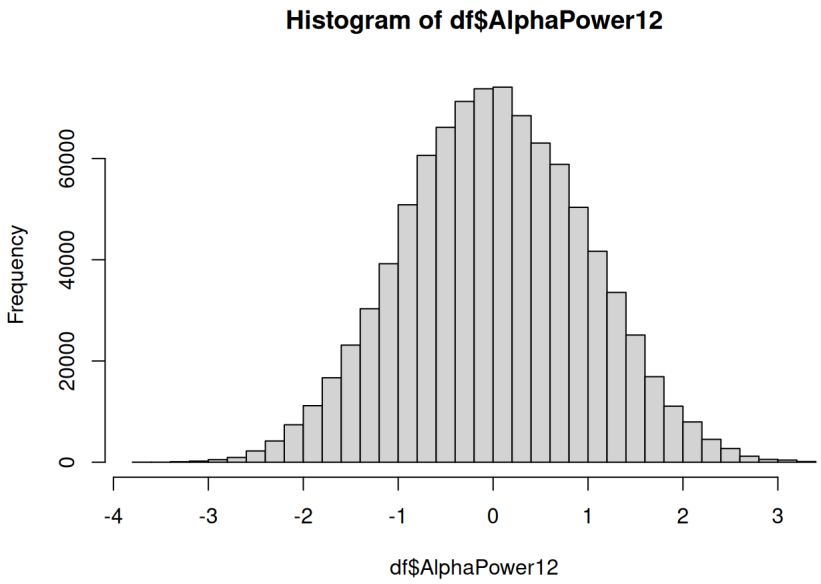
Alpha Power 10Hz overall, histogram:

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



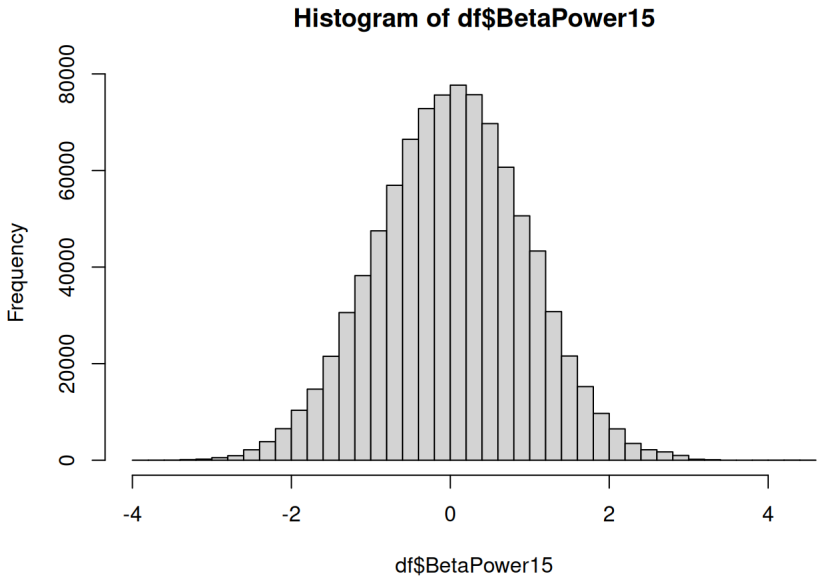
Alpha Power 12Hz overall, histogram:

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



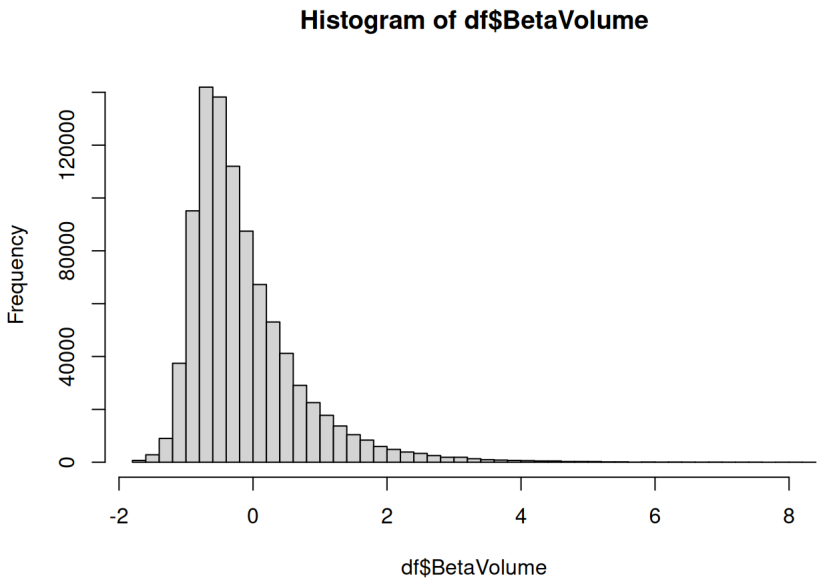
Beta Power 15Hz overall, histogram:

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



Beta Volume overall, histogram:

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

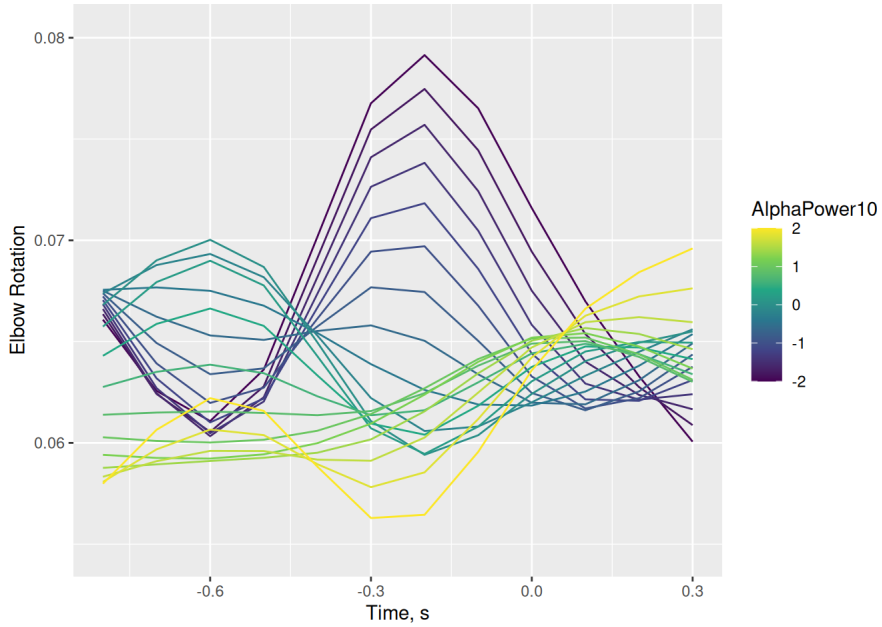


Run models + visualize predictions

Alpha 10Hz only + trials: predict Elbow Rotation

```
model_a10 = bam(Rotation ~ s(Trial, Subject, bs='re') +te(AlphaPower10,Time, k=5),
                data=df, method='REML')
```

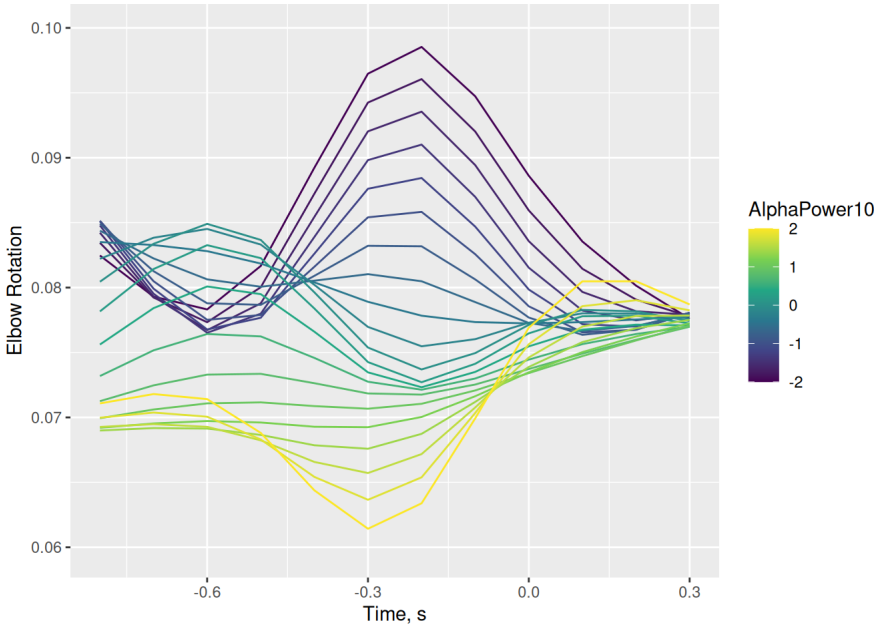
```
(ggemmeans(model_a10, terms = c("Time [-0.8:0.3, by=0.1]", "AlphaPower10 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = predicted, ymin=con
```



Alpha 10Hz only + no trials: predict Elbow Rotation

```
model_a10_notri = bam(Rotation ~ te(AlphaPower10,Time,k=5), # + s(Time) +ti(AlphaPower,Time),
                data=df, method='REML')
```

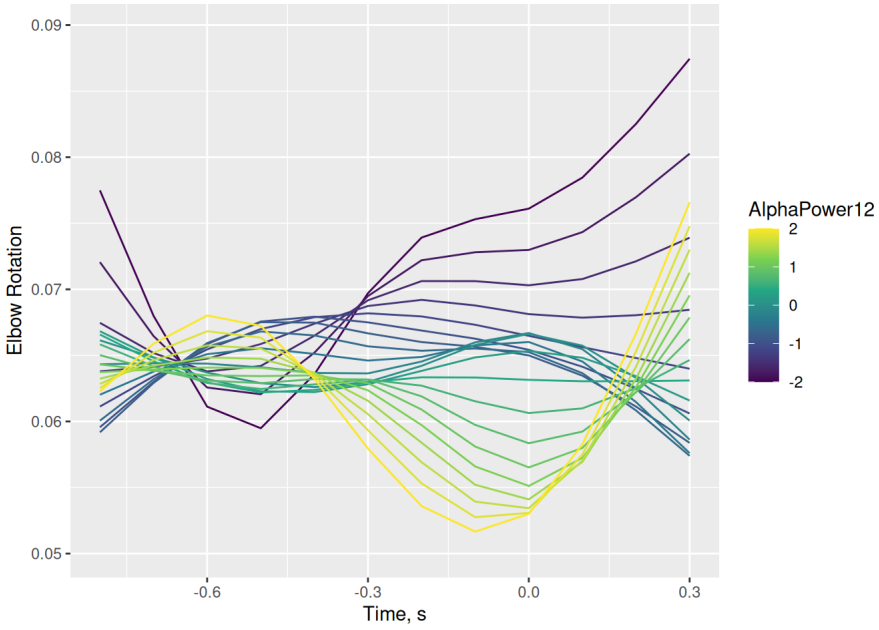
```
(ggemmeans(model_a10_notri, terms = c("Time [-0.8:0.3, by=0.1]", "AlphaPower10 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = predicted, ym
```



Alpha 12Hz only + trials: predict Elbow Rotation

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

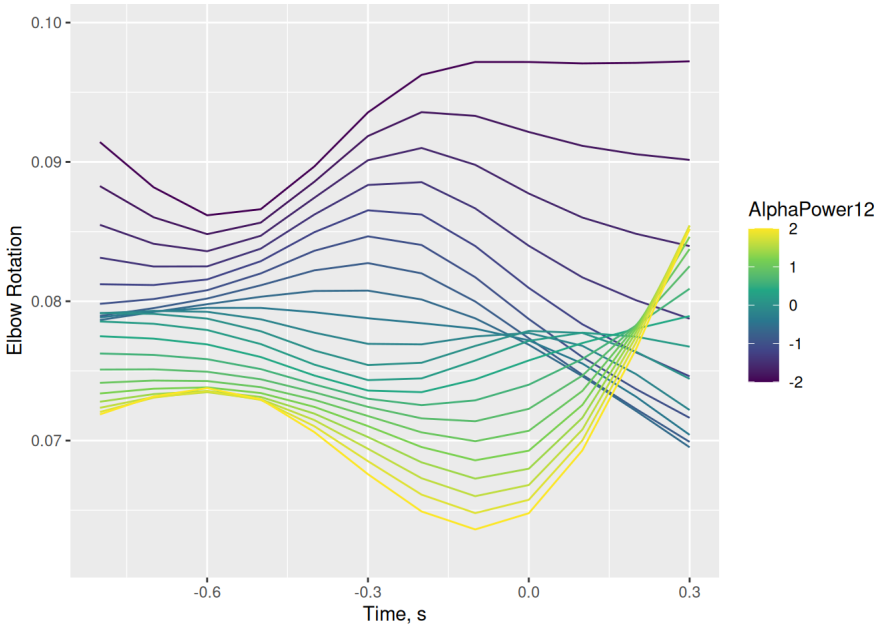
```
(ggemmeans(model_a12, terms = c("Time [-0.8:0.3, by=0.1]", "AlphaPower12 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = predicted, ymin=con
```



Alpha 12Hz only + no trials: predict Elbow Rotation

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

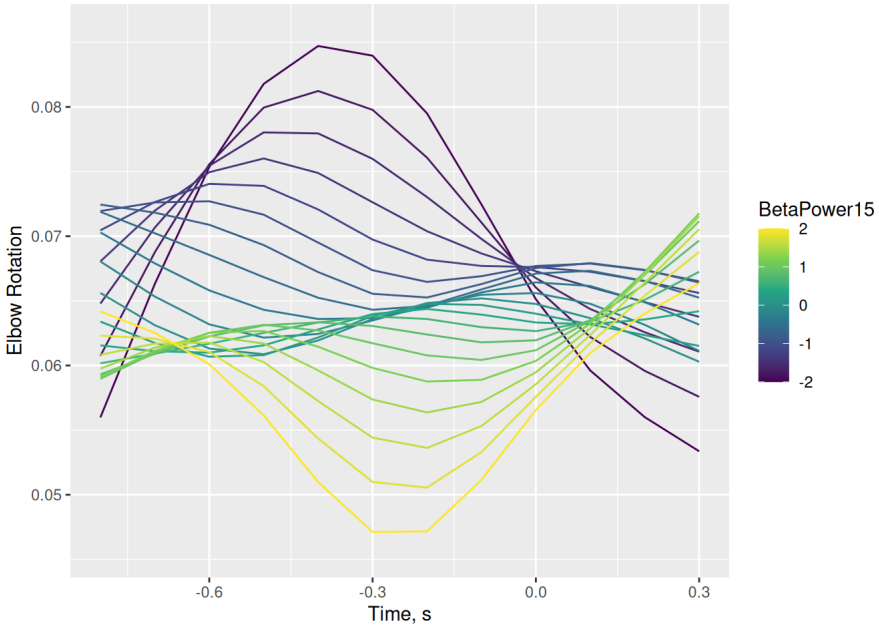
```
(ggemmeans(model_a12_notri, terms = c("Time [-0.8:0.3, by=0.1]", "AlphaPower12 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = predicted, ym
```



Beta 15Hz only + trials: predict Elbow Rotation

```
model_b15 = bam(Rotation ~ s(Trial, Subject, bs='re') +te(BetaPower15,Time,k=5),
                data=df, method='REML')
```

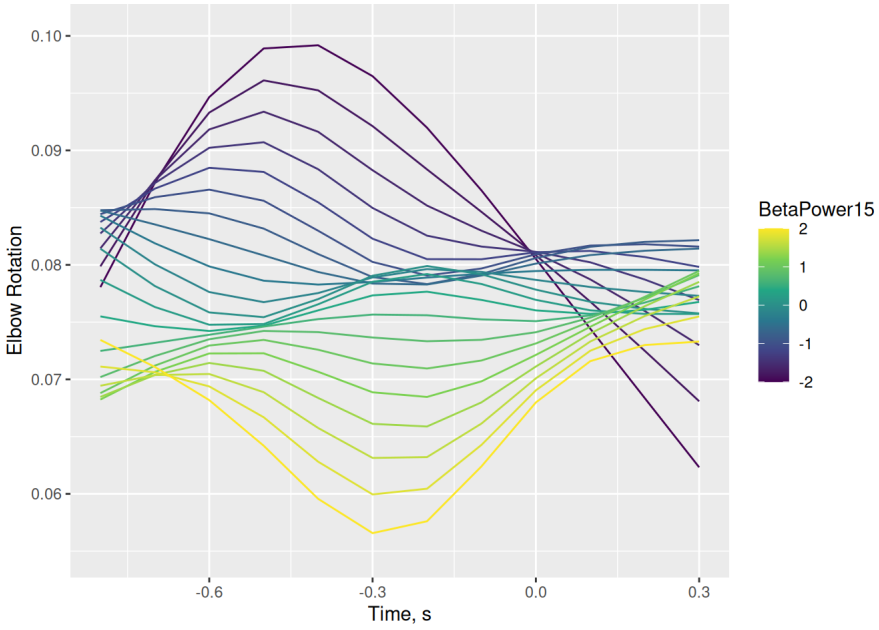
```
(ggemmeans(model_b15, terms = c("Time [-0.8:0.3, by=0.1]", "BetaPower15 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = predicted, ymin=conf
```



Beta 15Hz only + no trials: predict Elbow Rotation

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

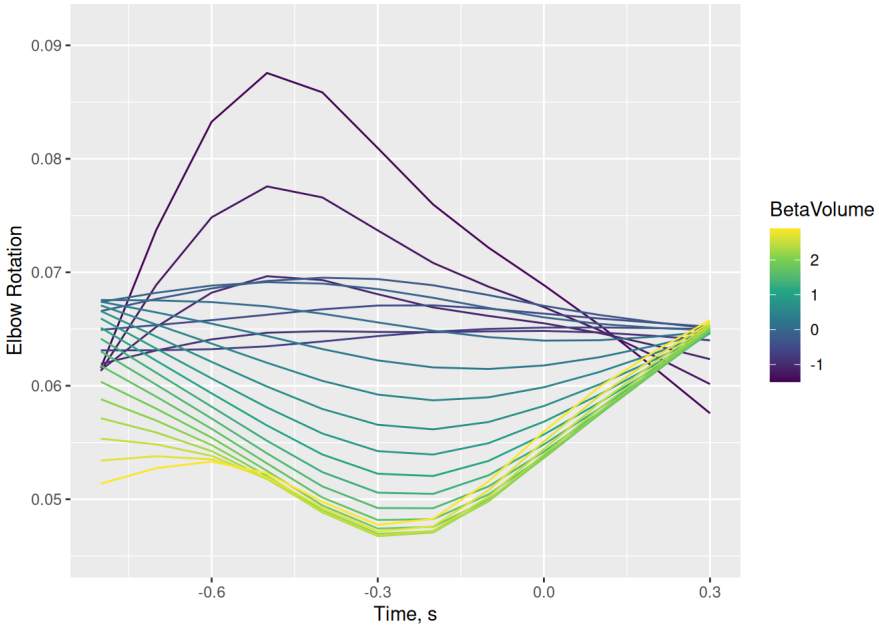
```
(ggemmeans(model_b15_notri, terms = c("Time [-0.8:0.3, by=0.1]", "BetaPower15 [-2:2, by=0.2]"))) %>% ggplot(aes(x = x, y = predicted, ymi
```



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=df, method='REML')
```

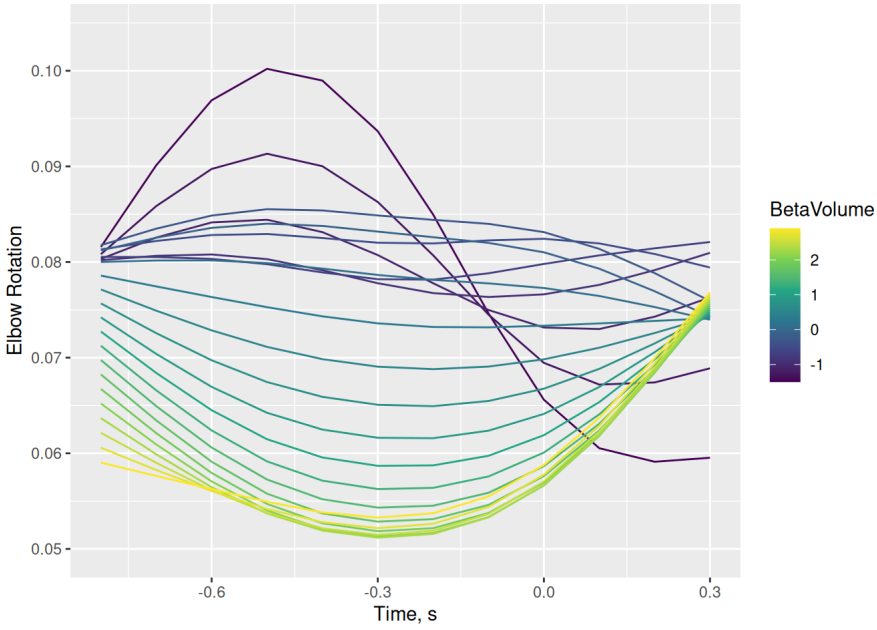
```
(ggemmeans(model_b, terms = c("Time [-0.8:0.3, by=0.1]", "BetaVolume [-1.5:3, by=0.2]"))) %>% ggplot(aes(x = x, y = predicted, ymin=conf.
```



Beta volume only + no trials: predict Elbow Rotation

```
model_b_notri = bam(Rotation ~ te(BetaVolume,Time),
              data=df, 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 = predicted, ymin
```

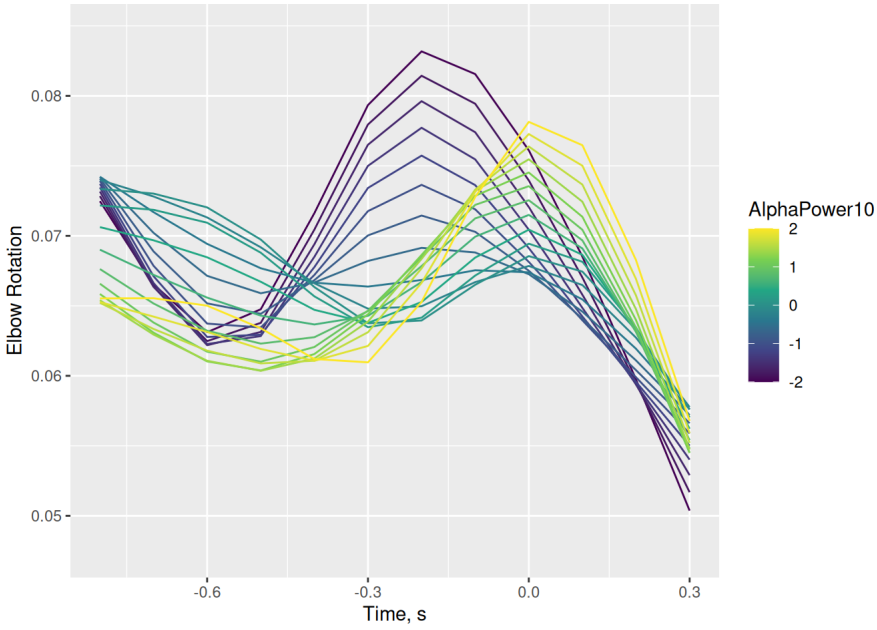


Alpha & Beta (all factors: alpha 10Hz, alpha 12Hz, beta 15Hz, beta volume) + trials: predict Elbow Rotation

```
set.seed(1)
model_ab = bam(Rotation ~ s(Trial, Subject, bs='re') +te(AlphaPower10,Time,k=5) +te(AlphaPower12,Time,k=5) + te(BetaVolume,Time,k=5) +te(
               data=df, method='REML')
```

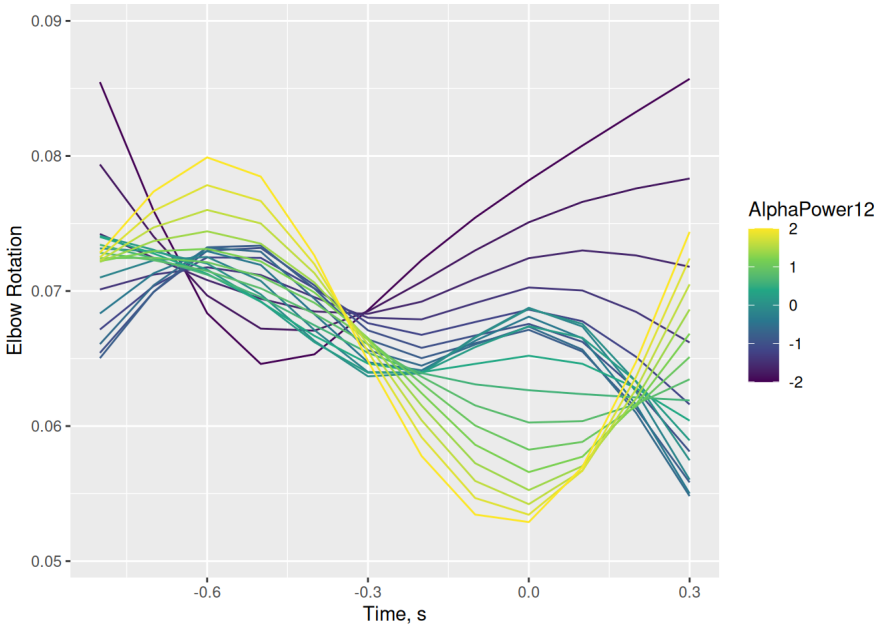
Alpha Power 10Hz prediction + trials

```
(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, ymin=conf
```



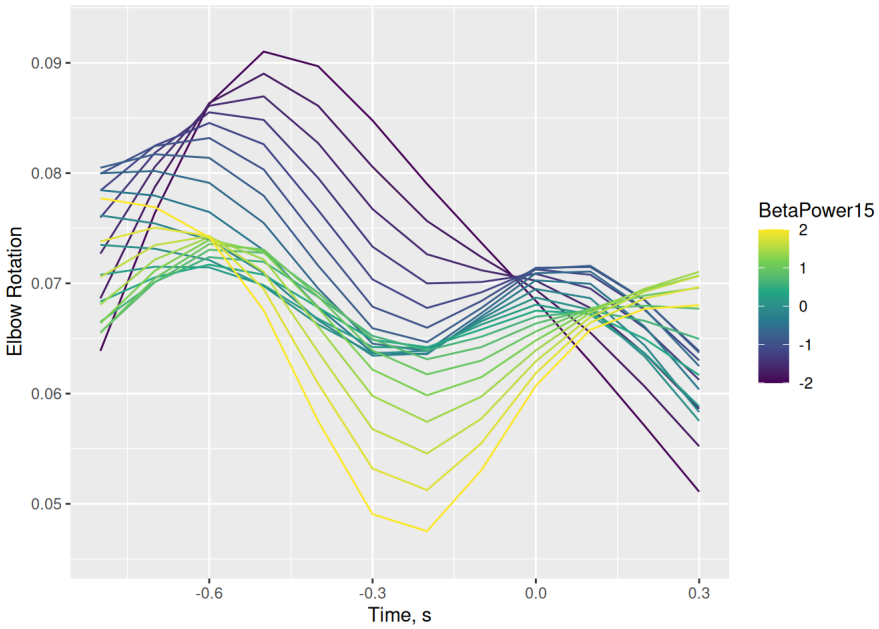
Alpha Power 12Hz prediction + trials

```
(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, ymin=conf
```



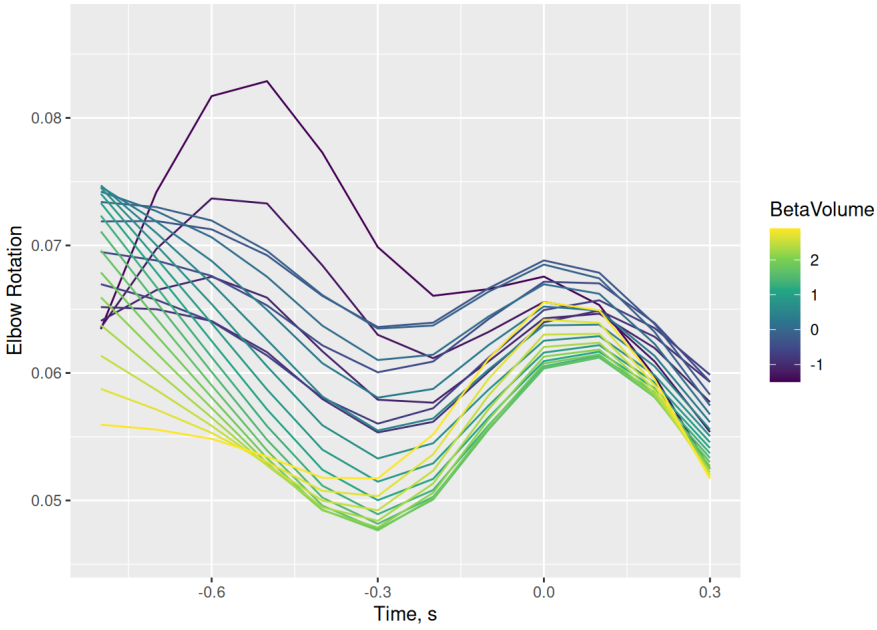
Beta Power 15Hz prediction + trials

```
(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, ymin=conf.
```



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, ymin=conf
```

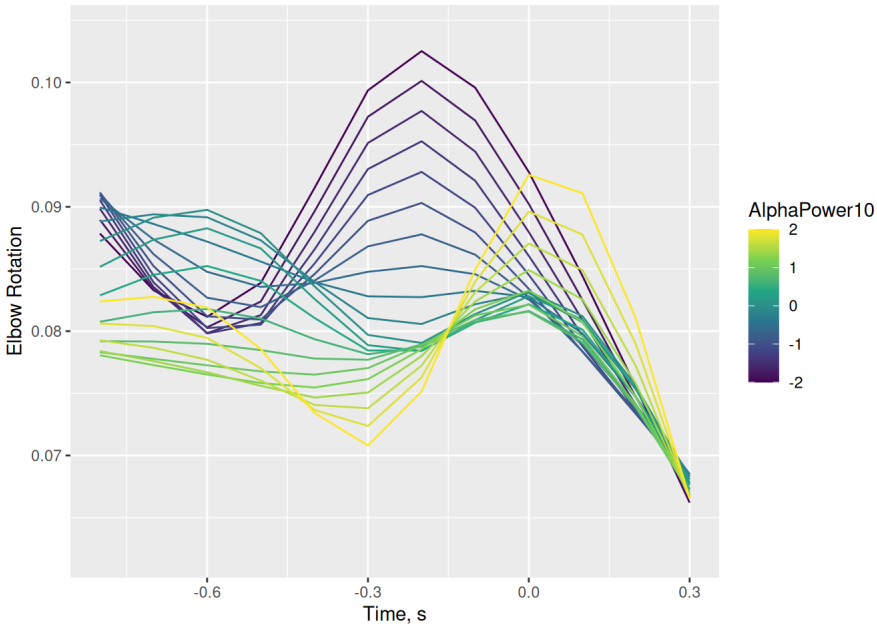


Alpha & Beta (all factors: alpha 10Hz, alpha 12Hz, beta 15Hz, beta volume) + no trials: predict Elbow Rotation

```
set.seed(1)
model_ab_notri = bam(Rotation ~ te(AlphaPower10,Time) +te(AlphaPower12,Time) + te(BetaVolume,Time,k=5) +te(BetaPower15,Time),
                data=df, method='REML')
```

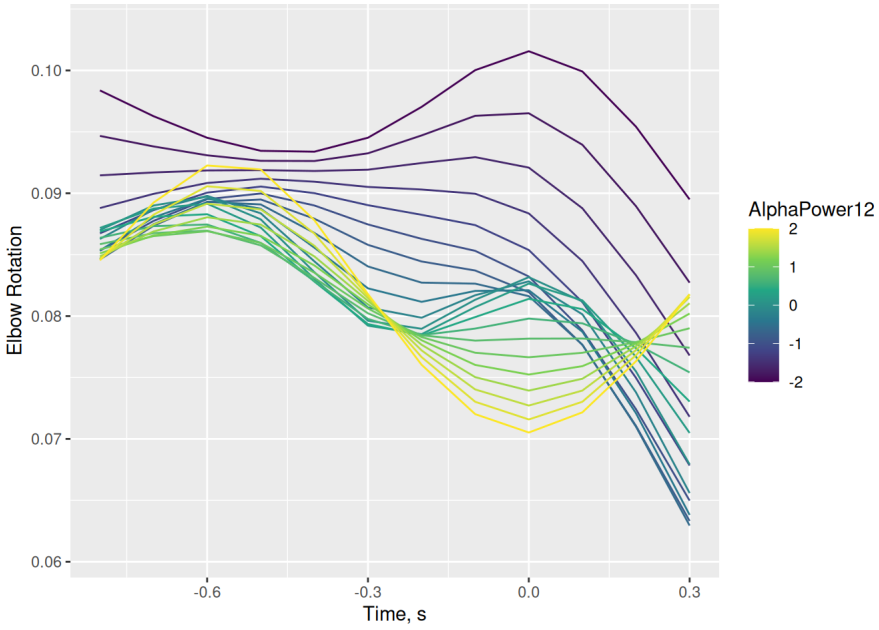
Alpha Power 10Hz prediction + no trials

```
(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 = predicted, ymi
```



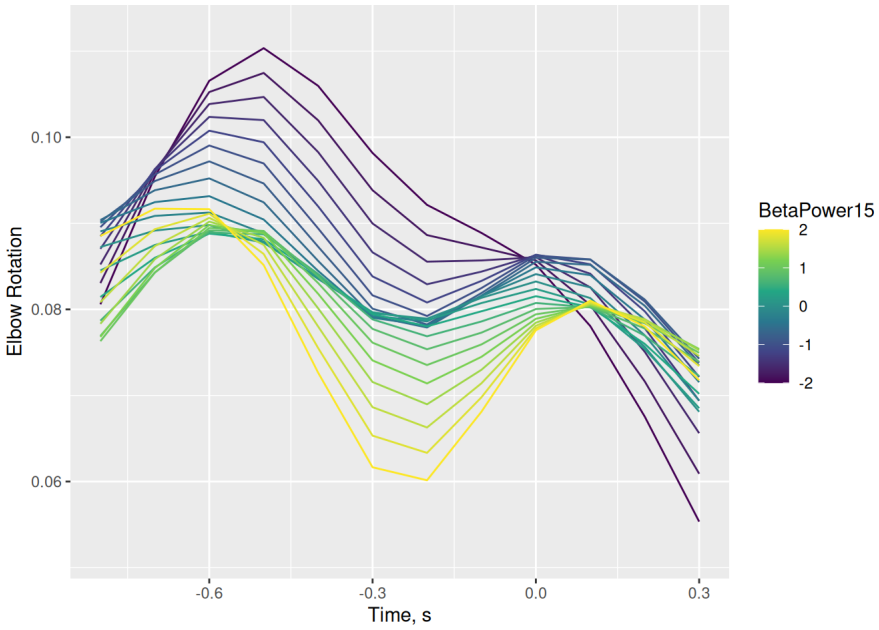
Alpha Power 12Hz prediction + no trials

```
(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 = predicted, ymi
```



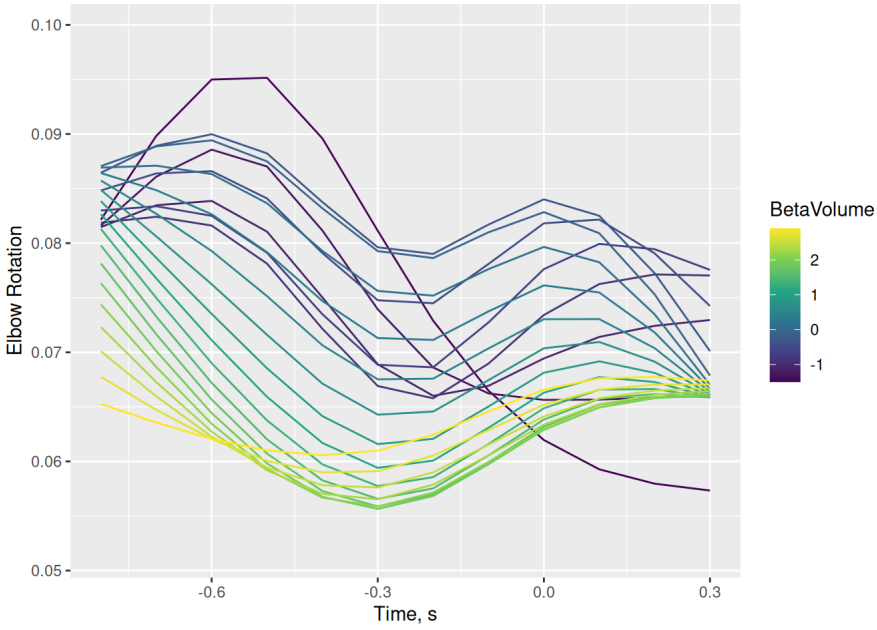
Beta Power 15Hz prediction + no trials

```
(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 = predicted, ymin
```



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 = predicted, ymi
```



Summary of models

```
summary(model_a10)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ s(Trial, Subject, bs = "re") + te(AlphaPower10, Time,
    k = 5)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.0641162  0.0001667   384.5   <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
s(Trial,Subject)     16.00   16.0 10769.82  <2e-16 ***
te(AlphaPower10,Time) 23.22   23.9    73.11  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.161   Deviance explained = 16.1%
-REML = -1.02e+06  Scale est. = 0.0063653  n = 919451
```

```
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) 0.0643548  0.0001677   383.7   <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
s(Trial,Subject)     16.00  16.00 10783.17  <2e-16 ***
te(AlphaPower12,Time) 23.63  23.98    98.33  <2e-16 ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.162   Deviance explained = 16.2%
-REML = -1.0203e+06  Scale est. = 0.0063611  n = 919451
```

```
summary(model_a10_notri)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ te(AlphaPower10, Time, k = 5)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 7.752e-02  9.078e-05   853.9   <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) 23.36  23.94 147.9  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.00384   Deviance explained = 0.386%
-REML = -9.411e+05  Scale est. = 0.0075581  n = 919451
```

```
summary(model_a12_notri)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ te(AlphaPower12, Time)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 7.750e-02  9.077e-05   853.7   <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(AlphaPower12,Time) 22.54  23.54 167.5  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.00427   Deviance explained = 0.429%
-REML = -9.413e+05  Scale est. = 0.0075549  n = 919451
```

```
summary(model_b15)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ s(Trial, Subject, bs = "re") + te(BetaPower15, Time,
    k = 5)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.0645443  0.0001672   386.1   <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 10847.2  <2e-16 ***
te(BetaPower15,Time) 23.57  23.96   131.3  <2e-16 ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.162   Deviance explained = 16.2%
-REML = -1.0207e+06  Scale est. = 0.0063556  n = 919451
```

```
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) 0.0645386  0.0001706   378.3   <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 10621.2  <2e-16 ***
te(BetaVolume,Time) 23.02  23.75   105.3  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.162   Deviance explained = 16.2%
-REML = -1.0204e+06  Scale est. = 0.0063601  n = 919451
```

```
summary(model_b15_notri)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ te(BetaPower15, Time)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 7.748e-02  9.072e-05   854.1   <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
te(BetaPower15,Time) 23.31   23.9 162.7  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.00421   Deviance explained = 0.424%
-REML = -9.4127e+05  Scale est. = 0.0075553  n = 919451
```

```
summary(model_b_notri)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ te(BetaVolume, Time)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 7.751e-02  9.064e-05   855.1   <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
te(BetaVolume,Time) 22.96  23.68 266.5  <2e-16 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.00682   Deviance explained = 0.684%
-REML = -9.4247e+05  Scale est. = 0.0075356  n = 919451
```

```
summary(model_ab)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ s(Trial, Subject, bs = "re") + te(AlphaPower10, Time,
    k = 5) + te(AlphaPower12, Time, k = 5) + te(BetaVolume, Time,
    k = 5) + te(BetaPower15, Time)

Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.065306   0.000172   379.7   <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
s(Trial,Subject)      16.00  16.00 10536.0  <2e-16 ***
te(AlphaPower10,Time) 23.47  23.95    75.4  <2e-16 ***
te(AlphaPower12,Time) 19.68  20.00   234.0  <2e-16 ***
te(BetaVolume,Time)   18.81  20.00   246.2  <2e-16 ***
te(BetaPower15,Time)  19.74  20.00   204.4  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.168   Deviance explained = 16.8%
-REML = -1.0235e+06  Scale est. = 0.0063138  n = 919451
```

```
summary(model_ab_notri)
```

```
Family: gaussian
Link function: identity

Formula:
Rotation ~ te(AlphaPower10, Time) + te(AlphaPower12, Time) +
    te(BetaVolume, Time, k = 5) + te(BetaPower15, Time)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 7.769e-02  9.051e-05   858.3   <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
te(AlphaPower10,Time) 23.32  23.89 116.3  <2e-16 ***
te(AlphaPower12,Time) 19.60  20.00 131.5  <2e-16 ***
te(BetaVolume,Time)   19.43  19.86 233.2  <2e-16 ***
te(BetaPower15,Time)  19.69  20.00 118.9  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.0153   Deviance explained = 1.54%
-REML = -9.4624e+05  Scale est. = 0.0074712  n = 919451
```

AIC model comparison

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

```
[1] -157919.8
```

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

```
[1] -158117.3
```

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

```
[1] -158974.7
```

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

```
[1] -155915.8
```

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

```
[1] -594.4015
```

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

```
[1] -803.4205
```

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

```
[1] -650.6106
```

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

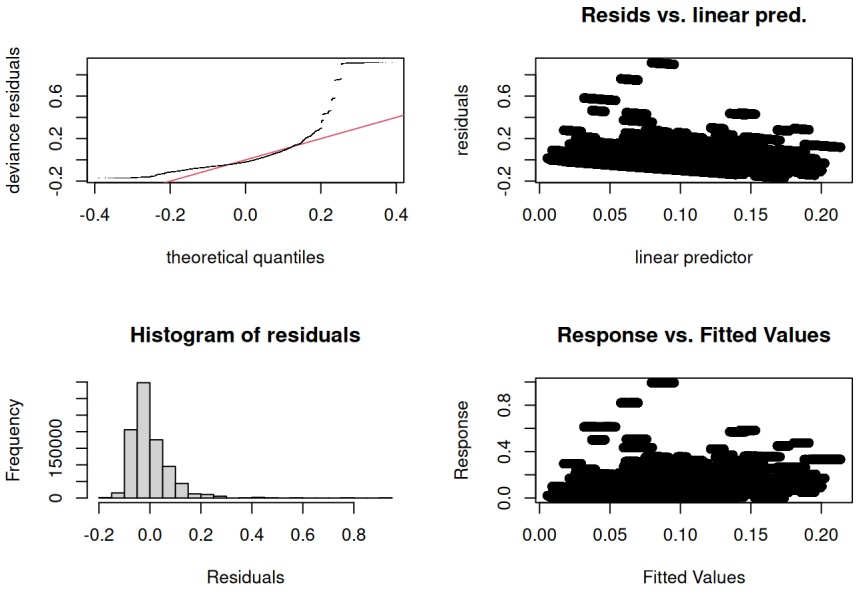
```
[1] -6009.178
```

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

```
[1] -154751.5
```

Check models

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



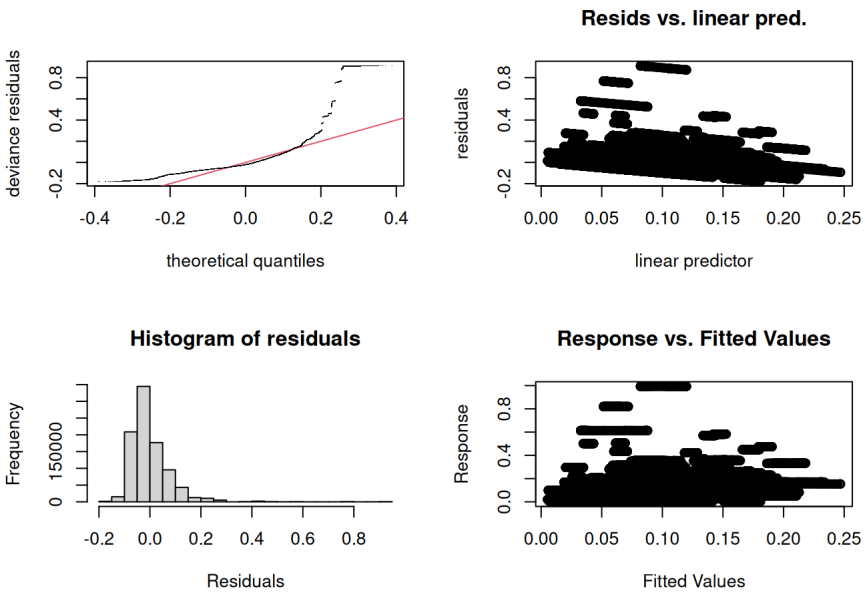

```
Method: REML   Optimizer: outer newton
full convergence after 6 iterations.
Gradient range [-7.066282e-05,8.155508e-09]
(score -1019993 & scale 0.006365255).
Hessian positive definite, eigenvalue range [3.033753,459723.5].
Model rank =  41 / 41

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 23.2    0.99    0.29
```

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

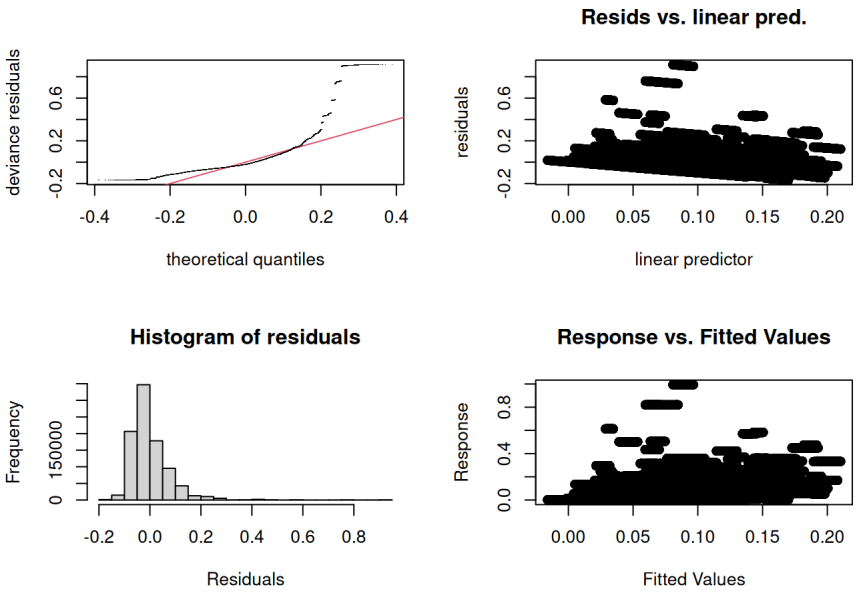



```
Method: REML   Optimizer: outer newton
full convergence after 8 iterations.
Gradient range [-0.02475434,0.01769504]
(score -1020282 & scale 0.00636114).
Hessian positive definite, eigenvalue range [3.346569,459723.5].
Model rank =  41 / 41

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 23.6    1.02    0.79
```

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

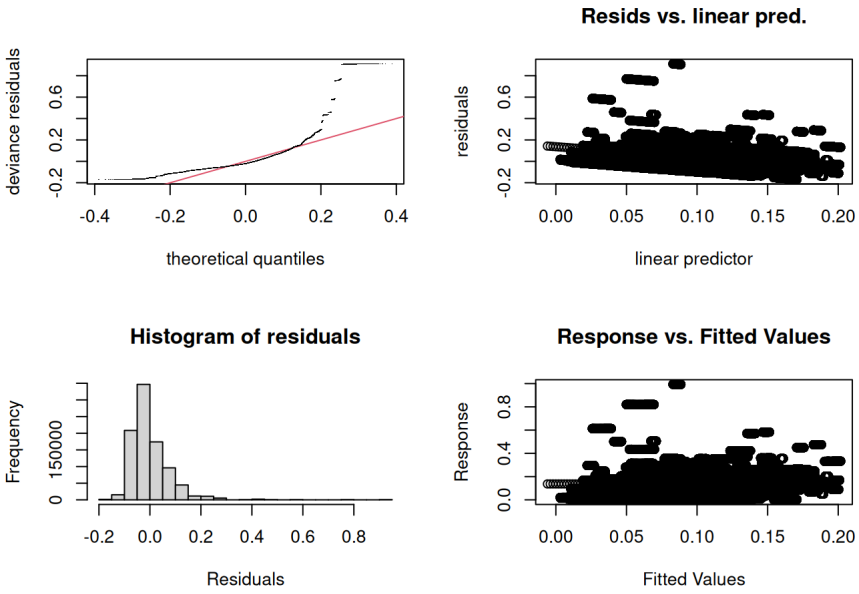


```
Method: REML   Optimizer: outer newton
full convergence after 7 iterations.
Gradient range [-0.0216611,0.01034977]
(score -1020683 & scale 0.006355585).
Hessian positive definite, eigenvalue range [3.439309,459723.5].
Model rank =  41 / 41

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    0.98    0.13
```

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

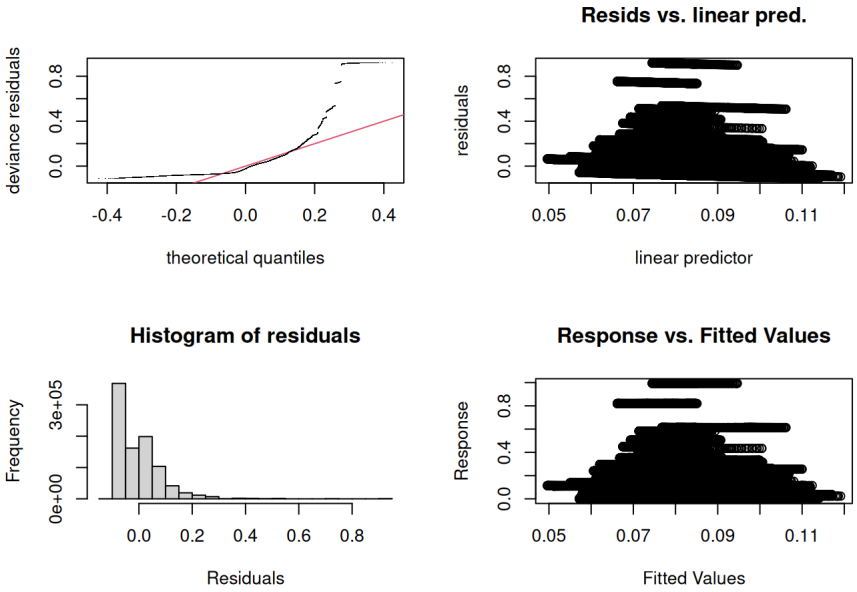



```
Method: REML   Optimizer: outer newton
full convergence after 6 iterations.
Gradient range [-0.0006681819,0.0001983555]
(score -1020362 & scale 0.006360082).
Hessian positive definite, eigenvalue range [2.341975,459723.5].
Model rank =  41 / 41

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  16      NA      NA
te(BetaVolume,Time) 24  23    0.99    0.36
```

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

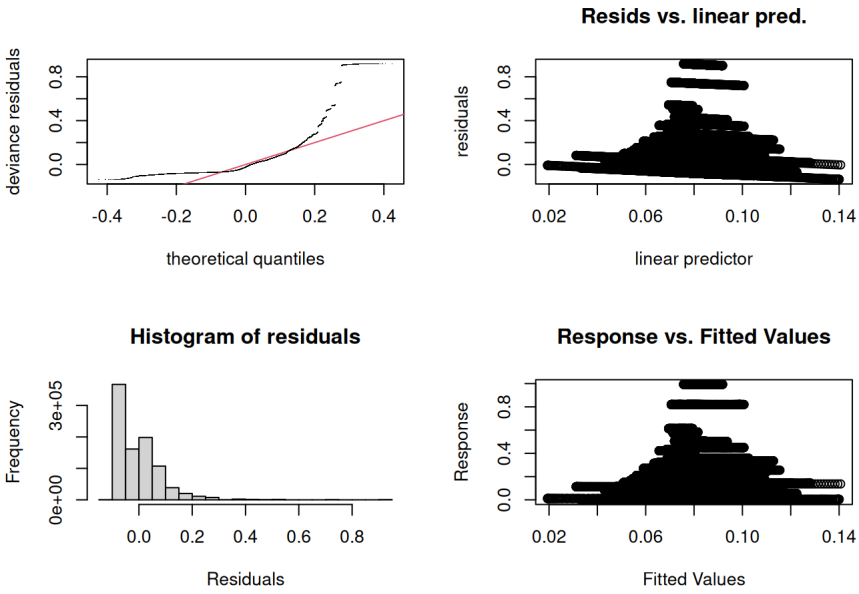



```
Method: REML   Optimizer: outer newton
full convergence after 6 iterations.
Gradient range [-0.004354881,3.044694e-08]
(score -941098.1 & scale 0.007558147).
Hessian positive definite, eigenvalue range [2.955954,459723.5].
Model rank =  25 / 25

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       1    0.54
```

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

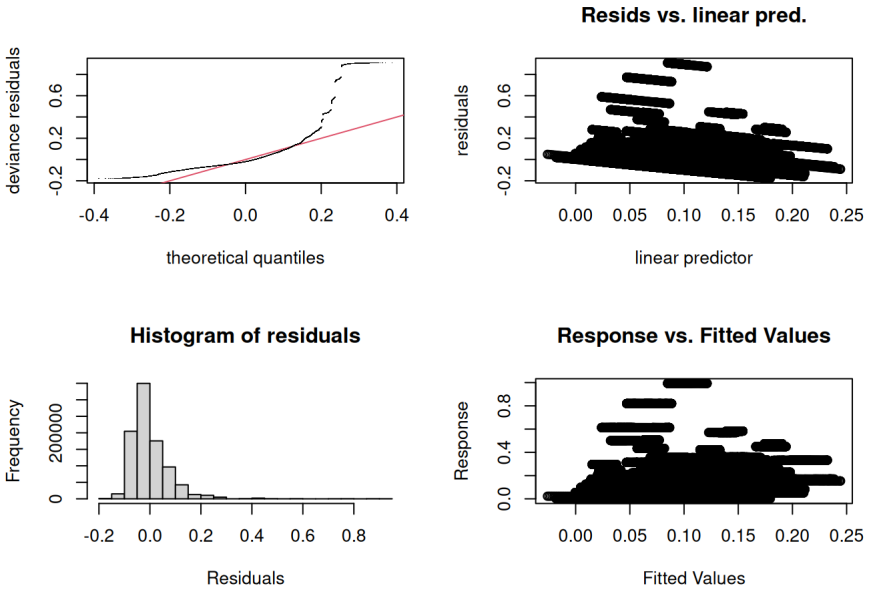



```
Method: REML   Optimizer: outer newton
full convergence after 8 iterations.
Gradient range [-0.0001494858,-1.611342e-10]
(score -941267.4 & scale 0.00755533).
Hessian positive definite, eigenvalue range [3.490632,459723.5].
Model rank =  25 / 25
```

```
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 23.3    1.02     0.9
```

```r
set.seed(1)
gam.check(model_ab)
```

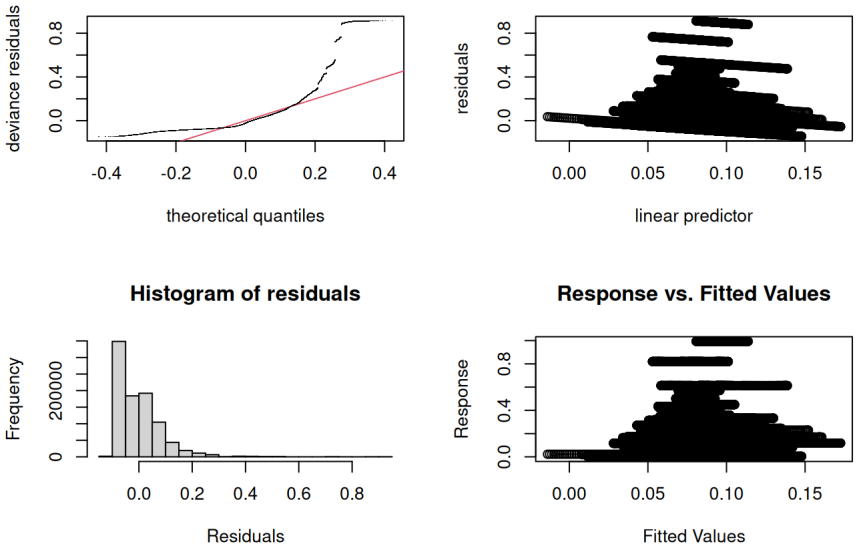



```
Method: REML   Optimizer: outer newton
full convergence after 13 iterations.
Gradient range [-0.6001023,0.5996927]
(score -1023549 & scale 0.006313776).
Hessian positive definite, eigenvalue range [2.388188,459724.1].
Model rank =  101 / 101

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 23.5    1.00    0.59
te(AlphaPower12,Time) 20.0 19.7    1.00    0.53
te(BetaVolume,Time)   20.0 18.8    1.00    0.47
te(BetaPower15,Time)  20.0 19.7    1.01    0.70
```

```r
set.seed(1)
gam.check(model_ab_notri)
```




```
Method: REML   Optimizer: outer newton
full convergence after 9 iterations.
Gradient range [-0.2041517,0.1720016]
(score -946236.5 & scale 0.007471233).
Hessian positive definite, eigenvalue range [2.240895,459722.7].
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 23.3    1.01    0.85
te(AlphaPower12,Time) 20.0 19.6    1.00    0.45
te(BetaVolume,Time)   20.0 19.4    1.00    0.54
te(BetaPower15,Time)  20.0 19.7    1.01    0.61
```