!/usr/bin/Rscript

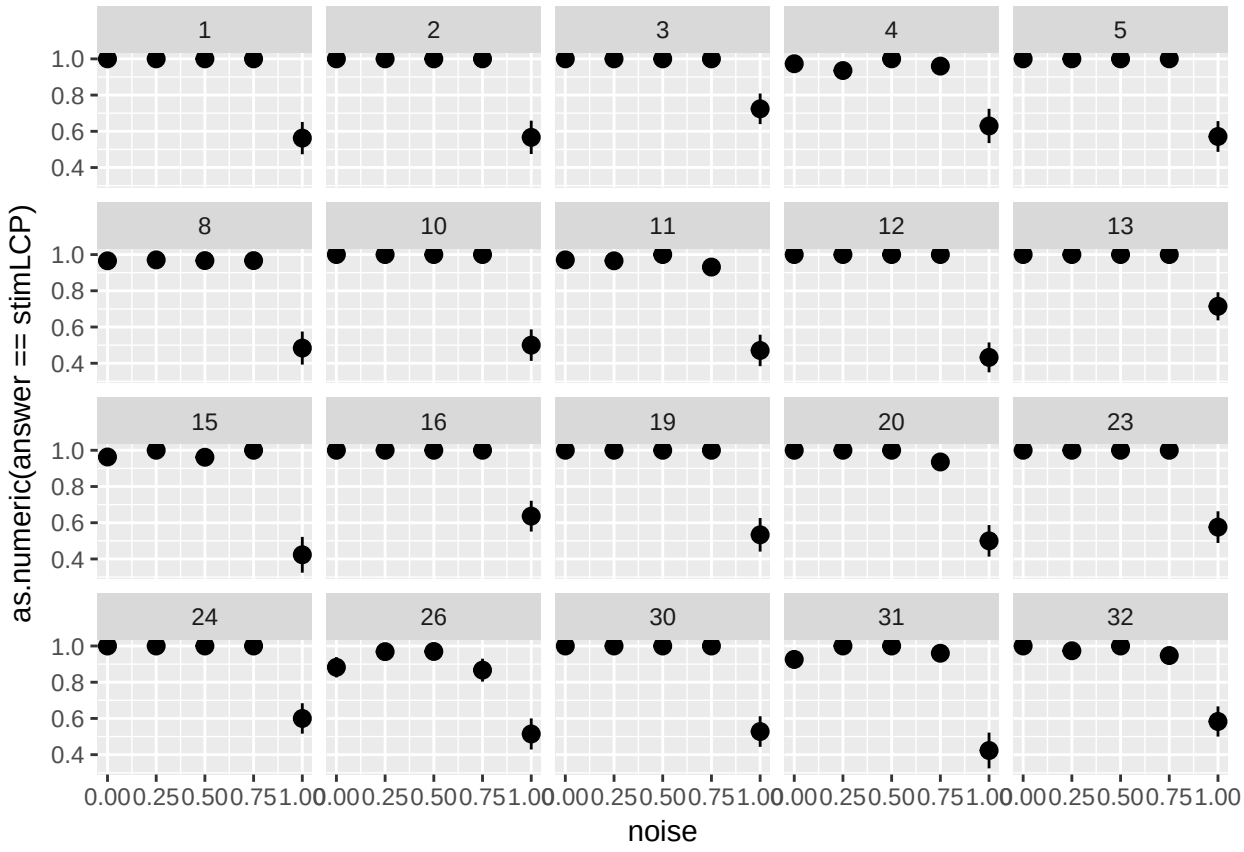
Some visualizations

Did they see the inset correctly?

```
ggplot(d_noise_raw%>%subset(stimPVisible==2),aes(x=noise,y=as.numeric(answer==stimLCP)))+stat_summary()
```

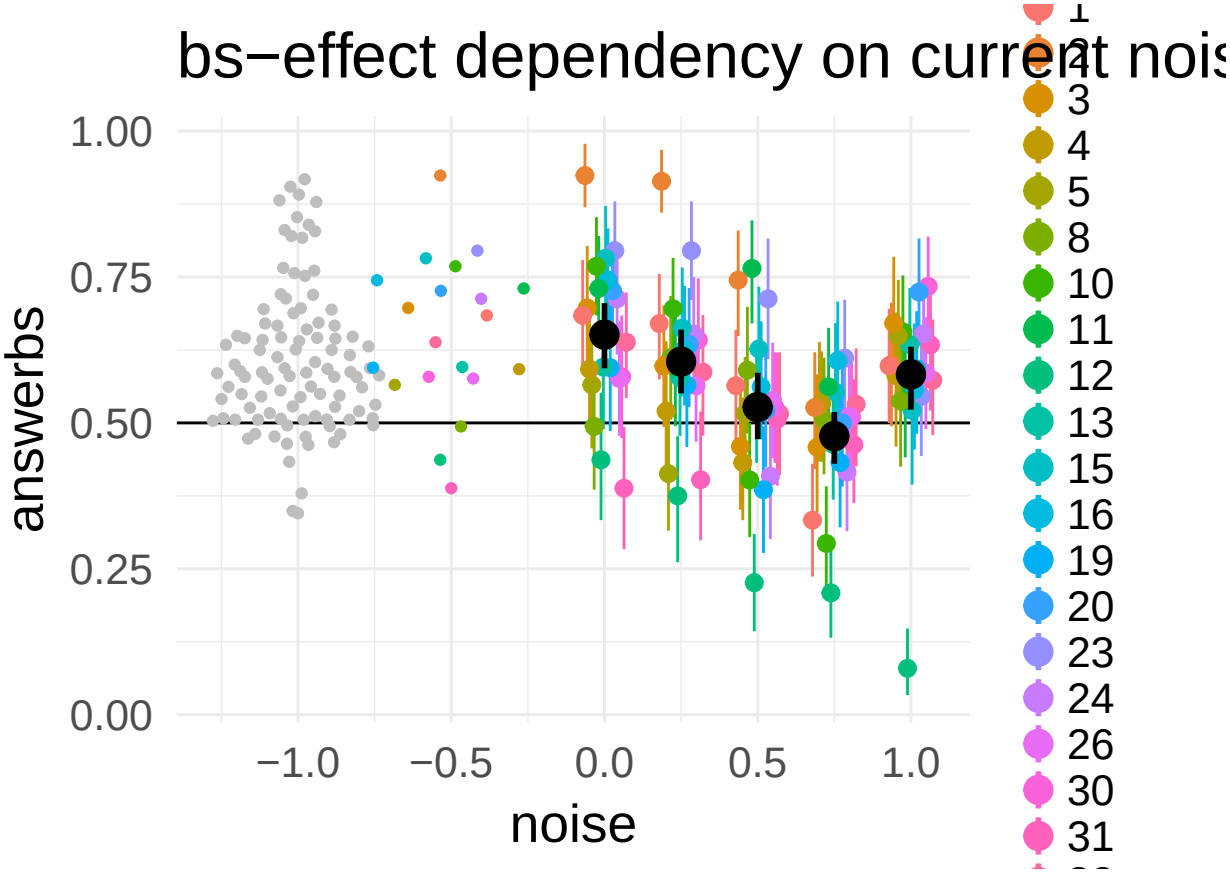
```
## Warning: Removed 255 rows containing non-finite values (stat_summary).
```

```
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
## No summary function supplied, defaulting to `mean_se()
```




```
# individual subject plot
p1 = ggplot(d_noise,aes(x=noise,y=answerbs,color=subject,group=subject))+
  geom_hline(yintercept = 0.5)+
  stat_summary(fun.data = 'mean_cl_boot',alpha=1,position=position_dodge(width=0.15))+
  stat_summary(data = d_noise%>%group_by(subject,noise)%>%summarise(answerbs=(mean(answerbs,na.rm=T))),
  ggbeeswarm::geom_quasirandom(inherit.aes = F,data = d_elife%>%group_by(subject)%>%summarize(bseffect=
  ggbeeswarm::geom_quasirandom(inherit.aes = F,data = d_noise%>%subset(noise==0)%>%group_by(subject)%>%
  ggtitle('bs-effect dependency on current noise')+theme_minimal(20)
p1
```

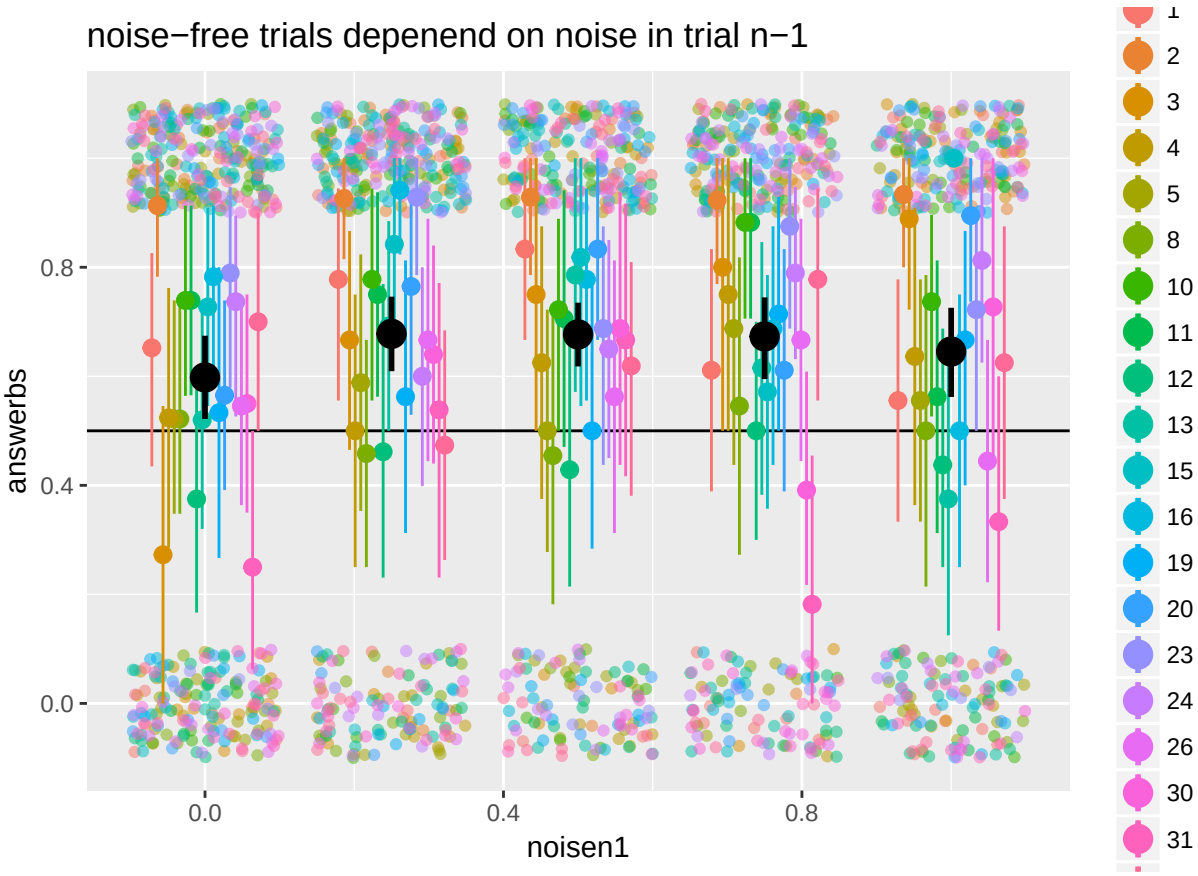
```
## Warning: Removed 783 rows containing non-finite values (stat_summary).
```




```
# noise n-1
p2 = ggplot(d_noise%>%subset(noise==0),aes(x=noisen1,y=answerbs,color=subject,group=subject))+
  geom_hline(yintercept = 0.5)+
  geom_jitter(height=0.1,width=0.1,alpha=0.5)+
  stat_summary(fun.data = 'mean_cl_boot',alpha=1,position=position_dodge(width=0.15))+
  stat_summary(data = d_noise%>%subset(noise==0)%>%group_by(subject,noisen1)%>%summarise(answerbs=mean(a
  ggtitle('noise-free trials depenend on noise in trial n-1')
p2
```

```
## Warning: Removed 170 rows containing non-finite values (stat_summary).
```

```
## Warning: Removed 170 rows containing missing values (geom_point).
```



Hypothesis 0

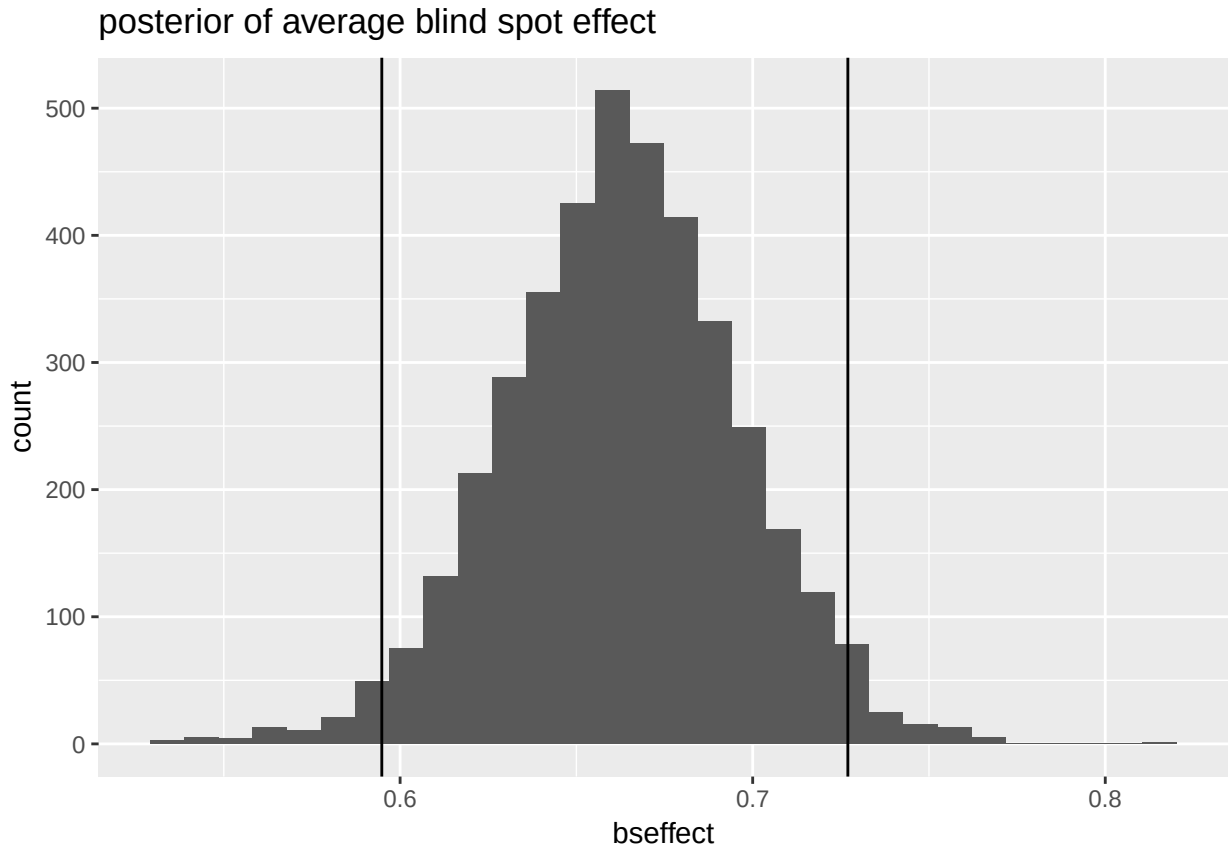
A blind spot effect for noise free stimuli

```
res.simple = stan_glmer(answerbs ~ 1+(1|subject),data=d_noise%>%subset(noise==0),family=binomial,prior_
```

```
## trying deprecated constructor; please alert package maintainer
```

```
# BS effect is coded with -1 / 1 => average effect is the result of bsFactor
posterior = as.data.frame(res.simple)
posterior$bseffect = 1/(1+exp(-posterior$'(Intercept)'))
ggplot(posterior,aes(x=bseffect))+geom_histogram()+geom_vline(xintercept = quantile(posterior$bseffect,
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Hypothesis 1 The effect should be larger than in the elife paper

```
d_combined = bind_rows(d_elife%>%mutate(experiment='elife',noise=0),d_noise)
```

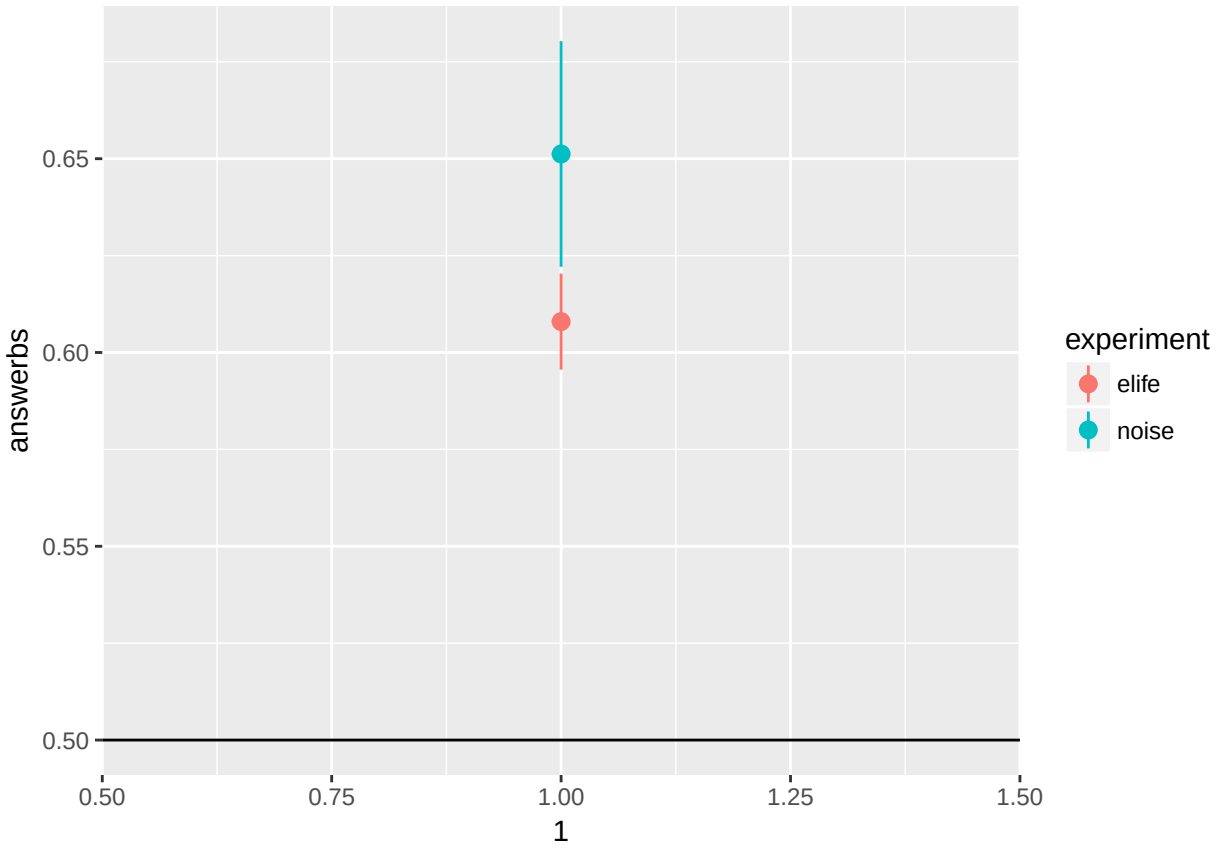
```
## Warning in bind_rows_(x, .id): Unequal factor levels: coercing to character
```

```
## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector
```

```
## Warning in bind_rows_(x, .id): binding character and factor vector,
## coercing into character vector
```

```
ggplot(d_combined%>%subset(noise==0)%>%group_by(experiment,subject)%>%summarise(answerbs = mean(answerbs
```

```
## No summary function supplied, defaulting to `mean_se()
```




```
d_summed = d_combined%>%subset(noise==0) %>% filter(!is.na(answerbs))%>%
  group_by(subject,experiment,noise)%>%
  summarise(totalanswer = length(answerbs),
            answerbs = sum(answerbs)
            )

#res.experiment2 = stan_glmer(answerbs ~ experiment+(1|experiment/subject),data=d_combined%>%subset(noi

# This is by a factor 100 faster:
res.experiment = stan_glmer(cbind(answerbs,totalanswer-answerbs) ~ experiment+(1|subject),data=d_summed
                            iter=4000)
```

```
## trying deprecated constructor; please alert package maintainer
```

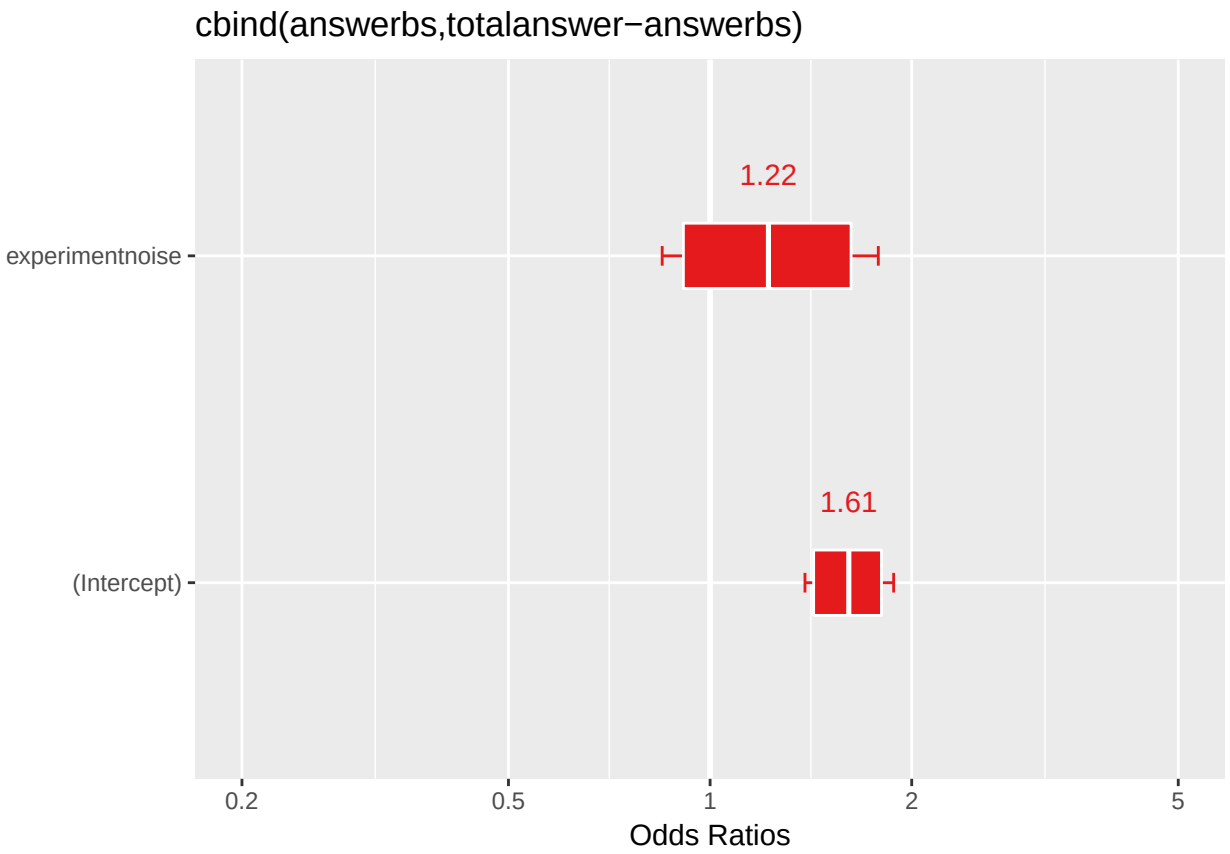
```
posterior = as.data.frame(res.experiment)
posterior$experimentnoise_prob = 1/(1+exp(-posterior$experimentnoise))

posterior$experimentnoise_prob = 1/(1+exp(-posterior$`(Intercept)` - posterior$experimentnoise))
posterior$experimentelife_prob = 1/(1+exp(-posterior$`(Intercept)`))

m = median(posterior$experimentnoise_prob)
m2 = median(posterior$experimentelife_prob)
lh  = coda::HPDinterval(coda::as.mcmc(posterior$experimentnoise_prob),0.95)
lh2 =coda::HPDinterval(coda::as.mcmc(posterior$experimentelife_prob),0.95)

sprintf('noise: med: %.1f%% (95HPD: [%.1f%%, %.1f%%])',m*100,lh[1]*100,lh[2]*100)
```

```
## [1] "noise: med: 66.3% (95HPD: [60.3%, 72.1%])"
```

```
sprintf('elife: med: %.1f%% (95HPD: [%.1f%%, %.1f%%])',m2*100,lh2[1]*100,lh2[2]*100)
```

```
## [1] "elife: med: 61.7% (95HPD: [58.8%, 64.3%])"
```

```
sjPlot::plot_model(res.experiment,show.intercept = T,show.values=T,prob.inner = 0.95,prob.outer = 0.99)
```



Hypothesis 2 We want to model the noise-dependency

```
res.noisespline = stan_gamm4(answerbs ~ 1 +  s(noise)         ,data=d_noise,random = ~(1+noise|subject)
res.noiselinear = stan_gamm4(answerbs ~ 1 +  noise            ,data=d_noise,random = ~(1+noise|subject)
res.noisebsl    = stan_glmer(answerbs ~ 1 + (1+noise|subject) ,data=d_noise,

#loo.noisespline = loo(res.noisespline)
#loo.noiselinear = loo(res.noiselinear)
#loo.noisebsl    = loo(res.noisebsl)

#compare_models(loo.noisespline,loo.noiselinear,loo.noisebsl)
```

Hypothesis 3a

```
res.noise_n1spline = stan_gamm4(answerbs ~ 1+s(noisen1),         data=d_noise%>%subset(noise==0),random
res.noise_n1linear = stan_glmer(answerbs ~ 1+noisen1 + (1+noisen1|subject),data=d_noise%>%subset(noise=

res.noise_n1bsl    = stan_glmer(answerbs ~ 1+(1+noisen1|subject),data=d_noise%>%subset(noise==0),family

#loo.noise_n1spline = loo(res.noise_n1spline)
#loo.noise_n1linear = loo(res.noise_n1linear)
```

```
#loo.noise_n1bsl    = loo(res.noise_n1bsl)
#compare_models(loo.noisespline,loo.noiselinear,loo.noise_n1bsl)
```

Hypothesis 3b

```
res.noise_n1_full   = stan_gamm4(answerbs ~ 1+s(noise,noisen1),   data=d_noise,random = ~(1+noise+noise
res.noise_n1_simple = stan_gamm4(answerbs ~ 1+s(noise)+s(noisen1),data=d_noise,random = ~(1+noise+noise

#loo.noise_n1_full = loo(res.noise_n1_ful)l
#loo.noise_n1_simple    = loo(res.noise_n1_simple)
#compare_models(loo.noisespline,loo.noise_n1_full,loo.noise_n1_simple)
```

```
save(paste0('/net/store/nbp/projects/EEG/blind_spot/',format(Sys.time(),"%Y-%m-%d_%H-%M-%S"),'_bs_noise_
```