Dresden analysis

Shravan Vasishth

4/30/2018

Load and preprocess data

In the preprocessing (not printed out in this pdf), we separate the data from the two cell types and set up a sliding contrast coding to compare the successive effects of

- Dot(arem) vs Control
- Mag(nevist) vs Dot
- Gad(ovist) vs Mag
- GdC13 vs Gad

A positive coefficient for a Y vs X comparison implies that the value for Y is higher than for X.

Area as dependent measure

Set up priors

We will use two sets of priors: Normal(0,1) or Cauchy(0,10). The latter allows extreme values.

```
priors_normal <- c(set_prior("normal(0, 10)", class = "Intercept"),
            set_prior("normal(0, 1)", class = "b"),
            set_prior("normal(0, 1)", class = "sd"),
            set_prior("normal(0, 1)", class = "sigma"),set_prior("lkj(2)", class = "cor")
            )

priors_cauchy <- c(set_prior("cauchy(0, 10)", class = "Intercept"),
            set_prior("cauchy(0, 1)", class = "b"),
            set_prior("cauchy(0, 1)", class = "sd"),
            set_prior("cauchy(0, 1)", class = "sigma"),set_prior("lkj(2)", class = "cor")
            )
```

Using Cauchy priors

As there are more conservative, I used Cauchy priors in preference to Normal priors, but the outcome is not dependent on the priors.

Area

Posterior probabilities of the parameter being positive:

```
(Dot_CMonoA<-round(mean(mMonoApost[,2]>0),2))
```

```
## [1] 0.81
```

```
(Mag_DotMonoA<-round(mean(mMonoApost[,3]>0),2))
```

```
## [1] 0.78
```

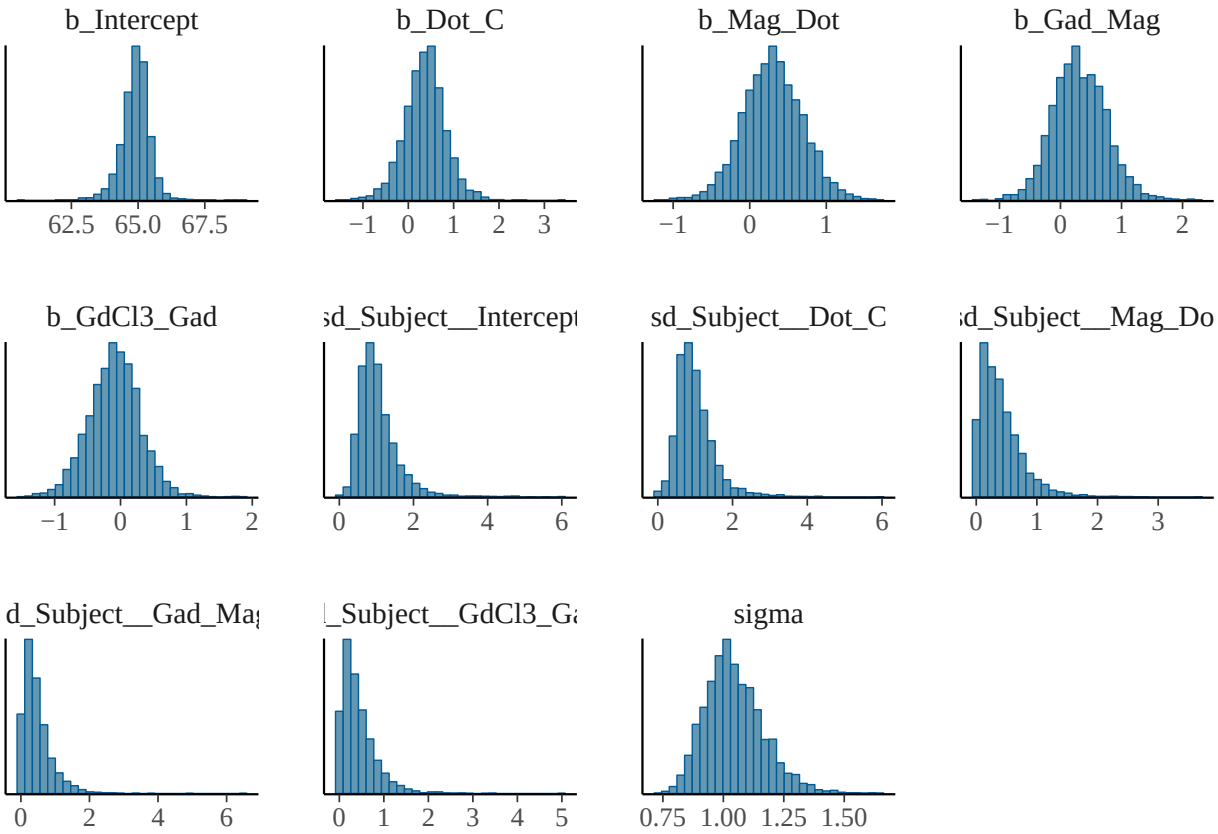
```
(Gad_MagMonoA<-round(mean(mMonoApost[,4]>0),2))
```

```
## [1] 0.76
```

```
(GdCl3_GadMonoA<-round(mean(mMonoApost[,5]>0),2))
```

```
## [1] 0.41
```

```
stanplot(mMonoA, type="hist",pars=c("^b","^sd","sigma"))
```



We see some weak evidence for Dot having higher values than Control, and Mag having higher values than Dot, and Gad being higher than Mag. But GdCl3 seems to be *lower* than Gad.

Posterior probabilities of the parameter being positive:

```
(Dot_CNeutroA<-round(mean(mNeutroApost[,2]>0),2))
```

```
## [1] 0.75
```

```
(Mag_DotNeutroA<-round(mean(mNeutroApost[,3]>0),2))
```

```
## [1] 0.91
```

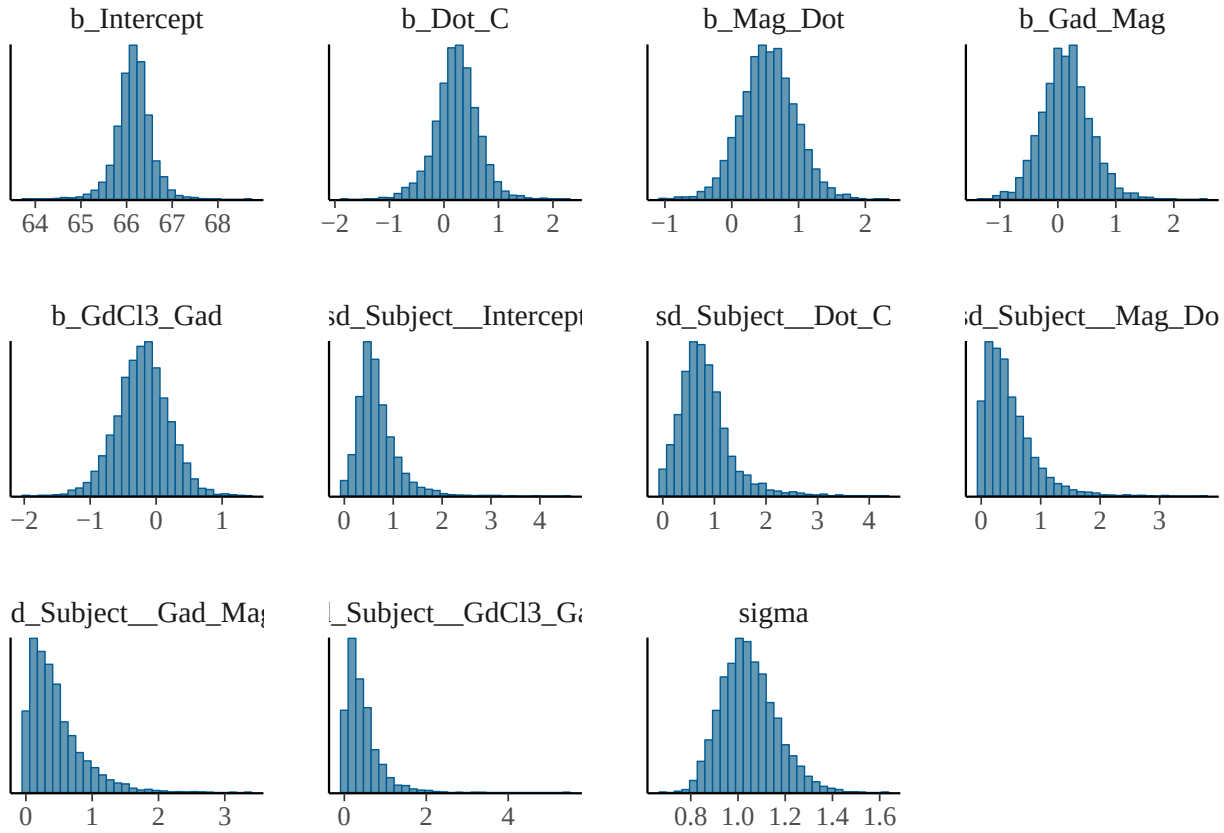
```
(Gad_MagNeutroA<-round(mean(mNeutroApost[,4]>0),2))
```

```
## [1] 0.63
```

```
(GdCl3_GadNeutroA<-round(mean(mNeutroApost[,5]>0),2))
```

```
## [1] 0.29
```

```
stanplot(mNeutroA, type="hist",pars=c("^b","^sd","sigma"))
```



Very similar results from Neurophils.

Deformation

Posterior probabilities of the parameter being positive:

```
(Dot_CMonoD<-round(mean(mMonoDpost[,2]>0),2))
```

```
## [1] 0.65
```

```
(Mag_DotMonoD<-round(mean(mMonoDpost[,3]>0),2))
```

```
## [1] 0.61
```

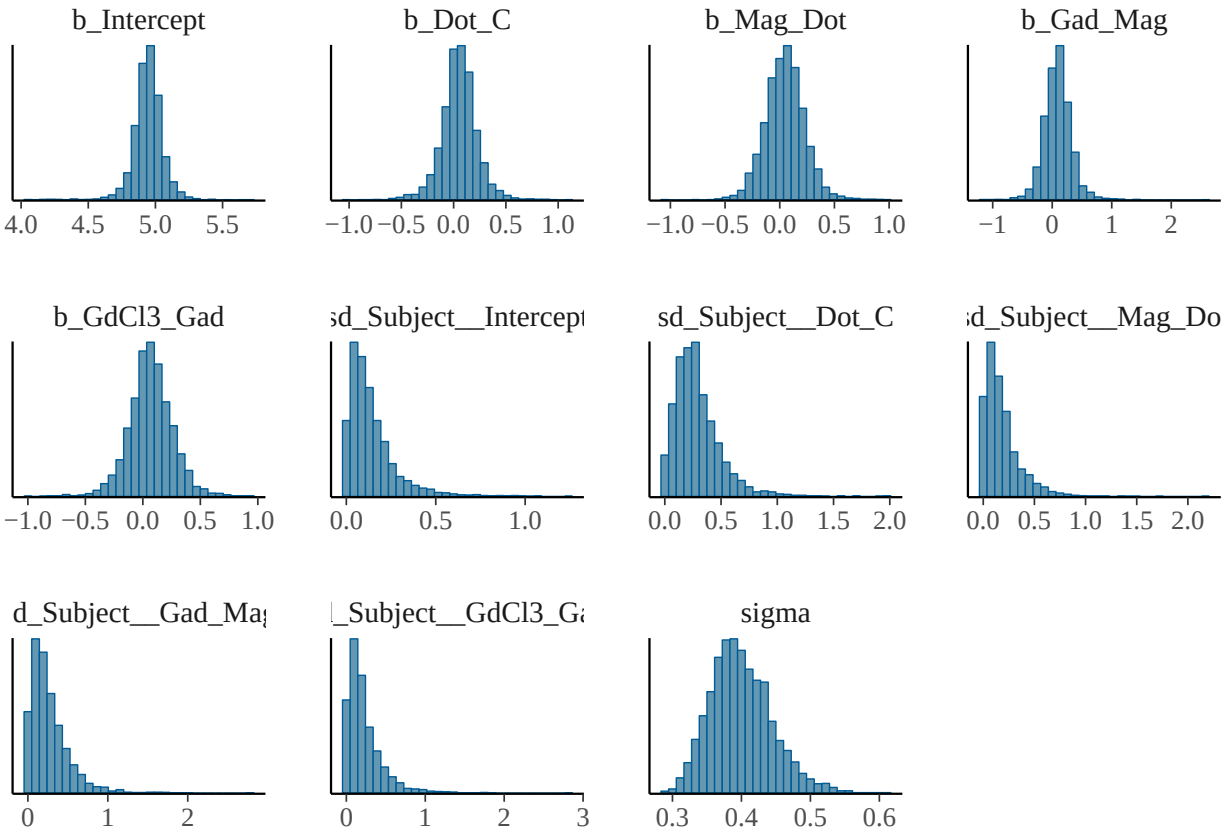
```
(Gad_MagMonoD<-round(mean(mMonoDpost[,4]>0),2))
```

```
## [1] 0.67
```

```
(GdCl3_GadMonoD<-round(mean(mMonoDpost[,5]>0),2))
```

```
## [1] 0.66
```

```
stanplot(mMonoD, type="hist",pars=c("^b","^sd","sigma"))
```



[Analysis of deformation to come]

Posterior probabilities of the parameter being positive:

```
(Dot_CNeutroD<-round(mean(mNeutroDpost[,2]>0),2))
```

```
## [1] 0.58
```

```
(Mag_DotNeutroD<-round(mean(mNeutroDpost[,3]>0),2))
```

```
## [1] 0.39
```

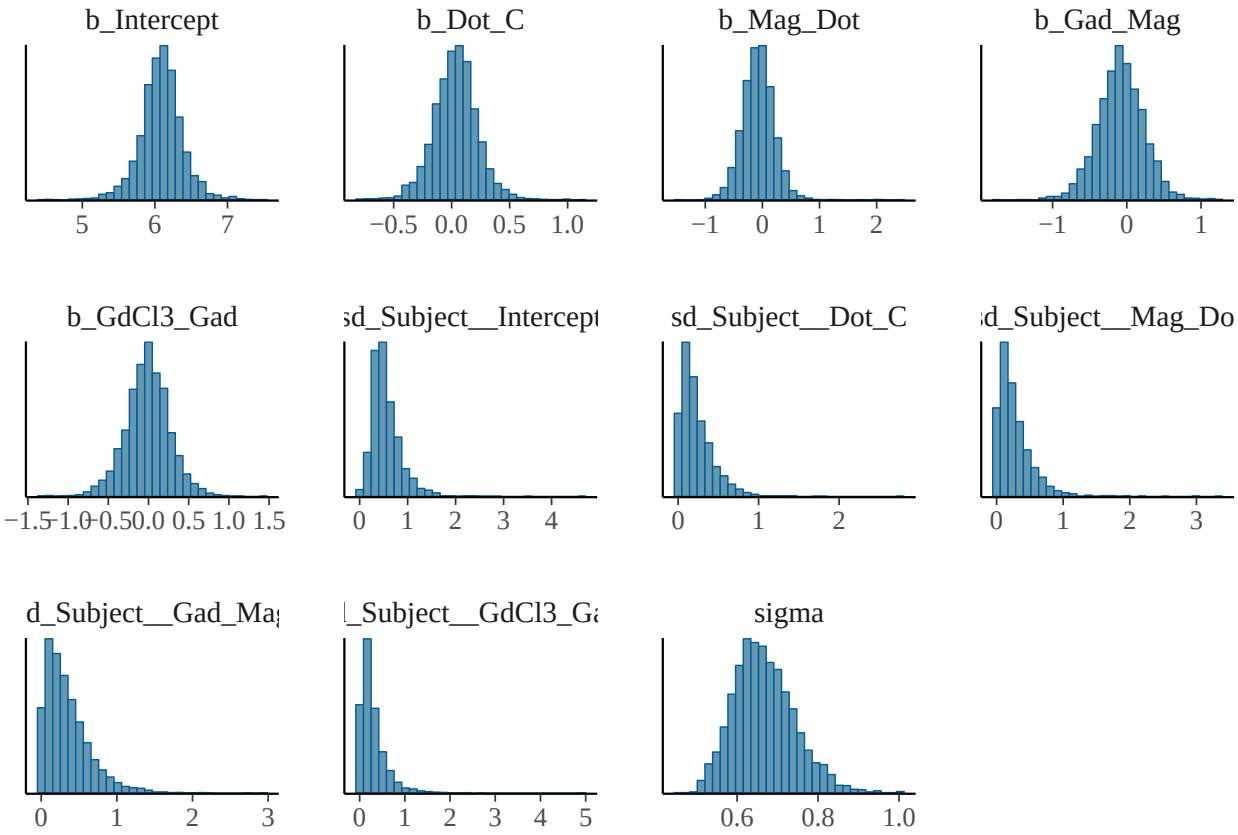
```
(Gad_MagNeutroD<-round(mean(mNeutroDpost[,4]>0),2))
```

```
## [1] 0.39
```

```
(GdCl3_GadNeutroD<-round(mean(mNeutroDpost[,5]>0),2))
```

```
## [1] 0.49
```

```
stanplot(mNeutroD, type="hist",pars=c("^b","^sd","sigma"))
```



Young's Modulus

Posterior probabilities of the parameter being positive:

```
(Dot_CMonoE<-round(mean(mMonoEpost[,2]>0),2))
```

```
## [1] 0.35
```

```
(Mag_DotMonoE<-round(mean(mMonoEpost[,3]>0),2))
```

```
## [1] 0.37
```

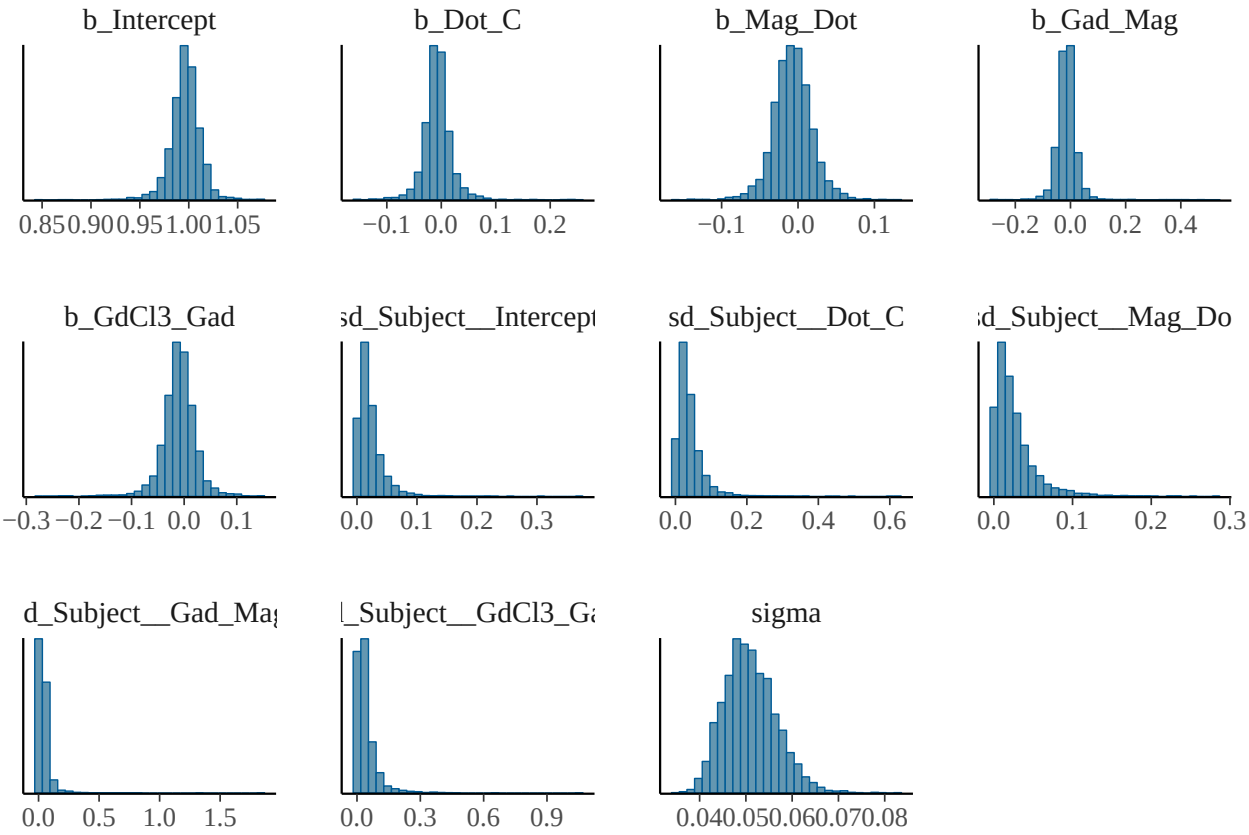
```
(Gad_MagMonoE<-round(mean(mMonoEpost[,4]>0),2))
```

```
## [1] 0.3
```

```
(GdCl3_GadMonoE<-round(mean(mMonoEpost[,5]>0),2))
```

```
## [1] 0.36
```

```
stanplot(mMonoE, type="hist",pars=c("^b","^sd","sigma"))
```



[Analysis of deformation to come]

Posterior probabilities of the parameter being positive:

```
(Dot_CNeutroE<-round(mean(mNeutroEpost[,2]>0),2))
```

```
## [1] 0.29
```

```
(Mag_DotNeutroE<-round(mean(mNeutroEpost[,3]>0),2))
```

```
## [1] 0.58
```

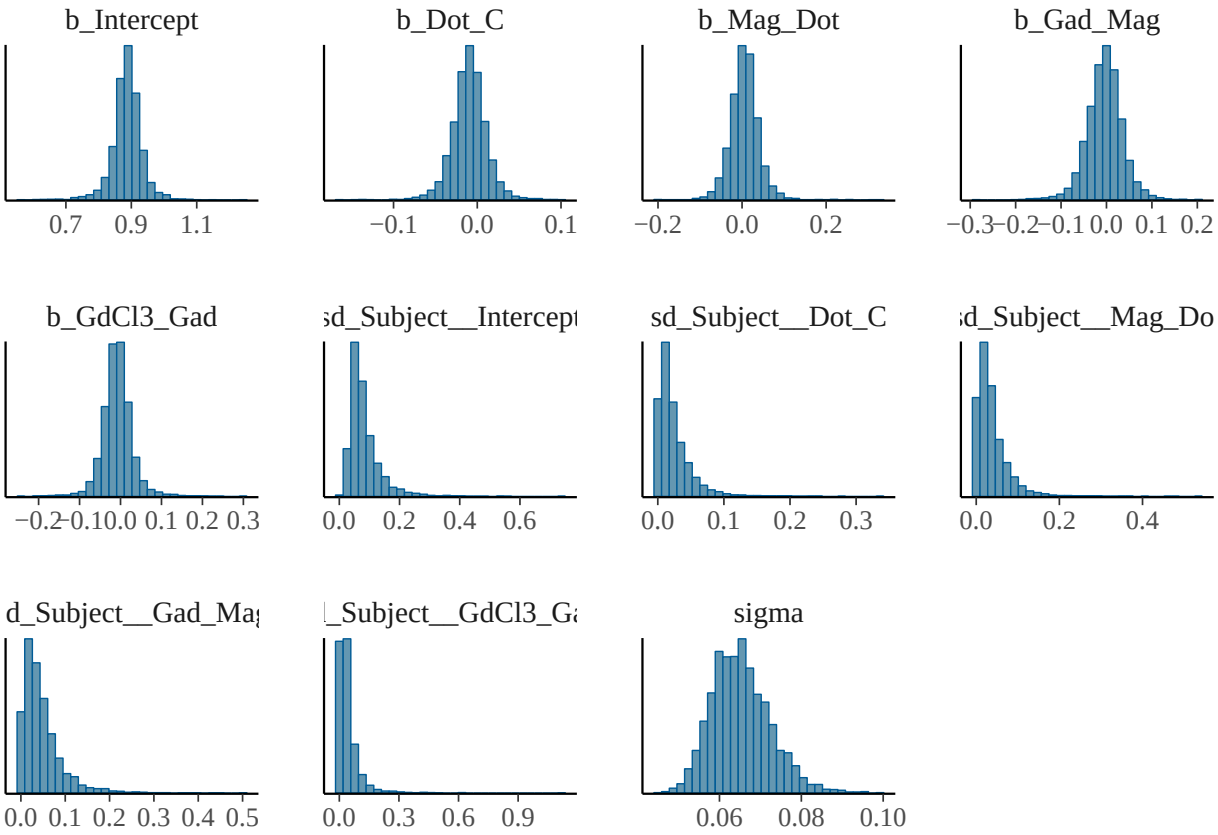
```
(Gad_MagNeutroE<-round(mean(mNeutroEpost[,4]>0),2))
```

```
## [1] 0.46
```

```
(GdCl3_GadNeutroE<-round(mean(mNeutroEpost[,5]>0),2))
```

```
## [1] 0.38
```

```
stanplot(mNeutroE, type="hist",pars=c("^b","^sd","sigma"))
```



Conclusion

Area

Monocytes: The probability (given data and model) that

- Dot has a higher value than control: 0.81
- Mag has a higher value than Dot: 0.78
- Gad has a higher value than Mag: 0.76
- GdCl3 has a higher value than Gad: 0.41

Neutrophils: The probability (given data and model) that

- Dot has a higher value than control: 0.75
- Mag has a higher value than Dot: 0.91
- Gad has a higher value than Mag: 0.63
- GdCl3 has a higher value than Gad: 0.29

Deformation

Monocytes: The probability (given data and model) that

- Dot has a higher value than control: 0.65
- Mag has a higher value than Dot: 0.61
- Gad has a higher value than Mag: 0.67
- GdCl3 has a higher value than Gad: 0.66

Neutrophils: The probability (given data and model) that

- Dot has a higher value than control: 0.58
- Mag has a higher value than Dot: 0.39
- Gad has a higher value than Mag: 0.39
- GdCl3 has a higher value than Gad: 0.49

Young's Modulus

Monocytes: The probability (given data and model) that

- Dot has a higher value than control: 0.35
- Mag has a higher value than Dot: 0.37
- Gad has a higher value than Mag: 0.3
- GdCl3 has a higher value than Gad: 0.36

Neutrophils: The probability (given data and model) that

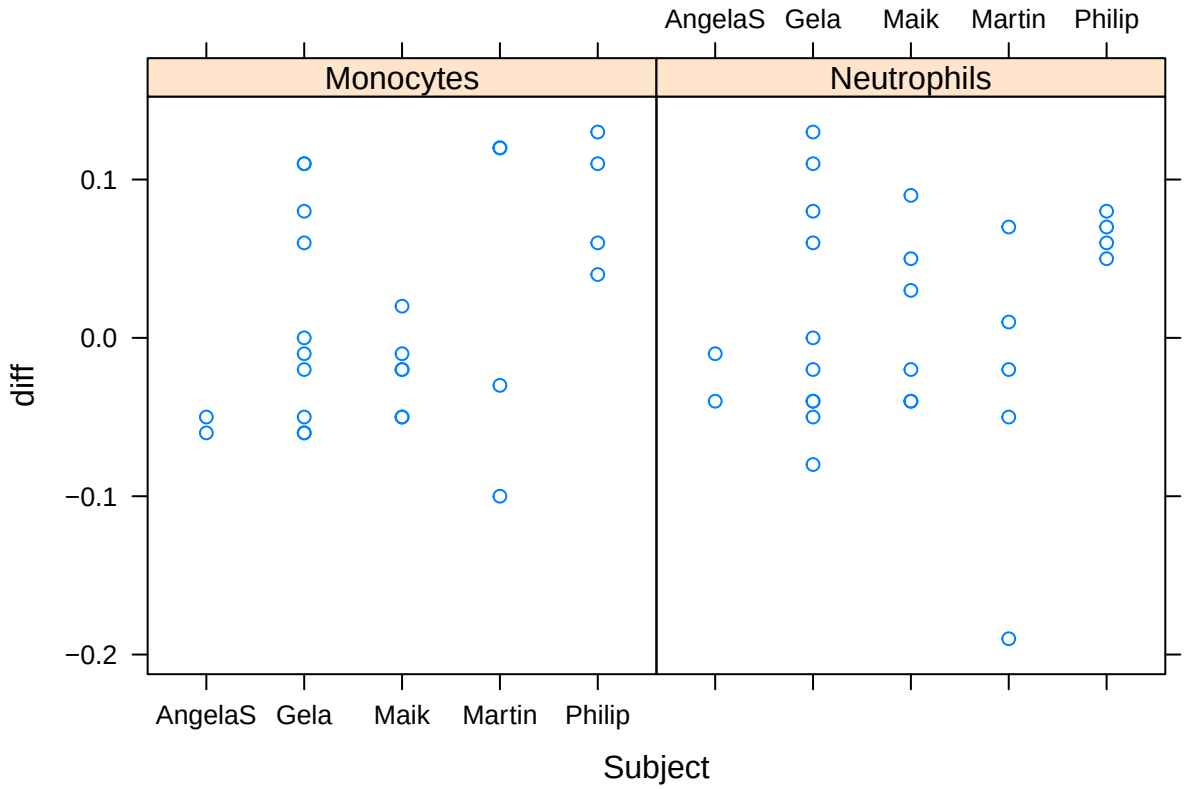
- Dot has a higher value than control: 0.29
- Mag has a higher value than Dot: 0.58
- Gad has a higher value than Mag: 0.46
- GdCl3 has a higher value than Gad: 0.38

So, I would conclude that given that we have only 5 people's data, there might be grounds for concluding that we can use different contrast agents on Monocytes and Neutrophils to study the effect of the agents on the cells. Only the dependent measure Area shows the expected increase by contrast agent (ordered from least toxic to most toxic). The only thing that may be odd is that the most toxic control, GdCl3, does not have higher values than Gad. If anything, Gad seems to have higher values; so Gad seems to be even more toxic than GdC13? Not sure what to make of that.

Analysis of Control vs Treatment

```
load("df_differences.RData")
#head(rtdc_diff)
diffdat<-subset(rtdc_diff,Parameter=="E" & Statistic=="Mean")
diffMono<-subset(diffdat,Cell_Type=="Monocytes")
diffNeutro<-subset(diffdat,Cell_Type=="Neutrophils")
#summary(diffMono)
#summary(diffNeutro)
```

```
library(lattice)
xyplot(diff~Subject|Cell_Type,diffdat)
```



Set priors (we will use cauchy priors):

```
priors_normal_diff <- c(set_prior("normal(0, 10)", class = "Intercept"),
            set_prior("normal(0, 1)", class = "sd"),
            set_prior("normal(0, 1)", class = "sigma"))

priors_cauchy_diff <- c(set_prior("cauchy(0, 10)", class = "Intercept"),
            set_prior("cauchy(0, 10)", class = "sd"),
            set_prior("cauchy(0, 10)", class = "sigma"))
```

```
mdiffMonoE <- brm(formula = diff ~ 1+(1 | Subject),
              data = diffMono, family = gaussian(), prior = priors_cauchy_diff,
              iter = 2000, chains = 4, control = list(adapt_delta = 0.999))
```

```
## Compiling the C++ model
## Start sampling
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 1).
##
## Gradient evaluation took 9e-06 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.09 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
```

```
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.414807 seconds (Warm-up)
##                0.19837 seconds (Sampling)
##                0.613177 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 2).
##
## Gradient evaluation took 5e-06 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.05 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.30519 seconds (Warm-up)
##                0.325047 seconds (Sampling)
##                0.630237 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 3).
##
## Gradient evaluation took 5e-06 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.05 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
```

```
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.311467 seconds (Warm-up)
##                0.347361 seconds (Sampling)
##                0.658828 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 4).
##
## Gradient evaluation took 6e-06 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.06 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.439849 seconds (Warm-up)
##                0.467323 seconds (Sampling)
##                0.907172 seconds (Total)
```

```
## Warning: There were 1 divergent transitions after warmup. Increasing adapt_delta above 0.999 may hel
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

```
mdiffNeutroE <- brm(formula = diff ~ 1+(1 | Subject),
              data = diffMono, family = gaussian(), prior = priors_cauchy_diff,
              iter = 2000, chains = 4, control = list(adapt_delta = 0.999))
```

```
## Compiling the C++ model
## Start sampling
```

```
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 1).
##
## Gradient evaluation took 1.1e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.11 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
```

```
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.3883 seconds (Warm-up)
##                0.599185 seconds (Sampling)
##                0.987485 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 2).
##
## Gradient evaluation took 5e-06 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.05 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.359972 seconds (Warm-up)
##                0.247607 seconds (Sampling)
##                0.607579 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 3).
##
## Gradient evaluation took 5e-06 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.05 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
```

```
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.296762 seconds (Warm-up)
##                0.508546 seconds (Sampling)
##                0.805308 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 4).
##
## Gradient evaluation took 5e-06 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.05 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.279155 seconds (Warm-up)
##                0.166743 seconds (Sampling)
##                0.445898 seconds (Total)
```

The posteriors don't show convincing evidence for any difference in either cell:

```
## posteriors
mdiffMonoEpost<-posterior_samples(mdiffMonoE, "Intercept")
mean(mdiffMonoEpost>0)
```

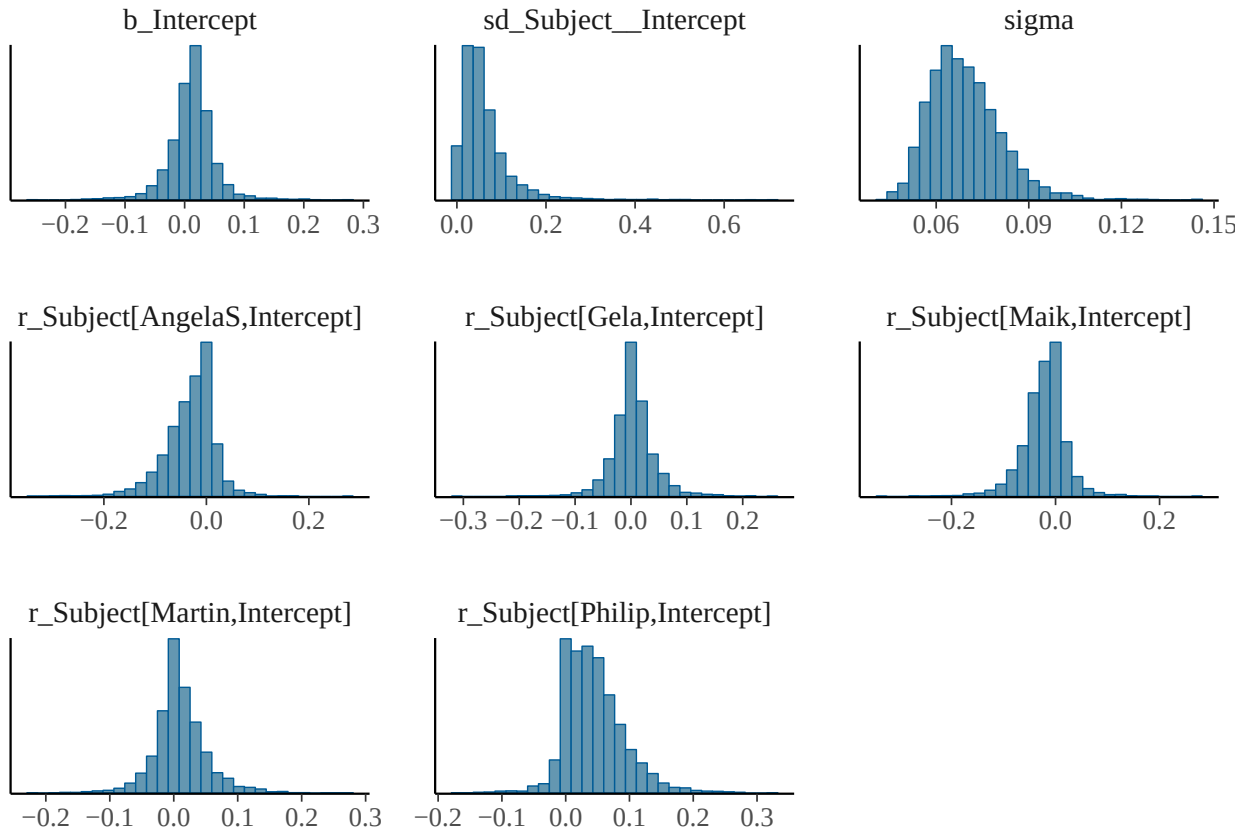
```
## [1] 0.6001071
```

```
## posteriors
mdiffNeutroEpost<-posterior_samples(mdiffNeutroE, "Intercept")
mean(mdiffNeutroEpost>0)
```

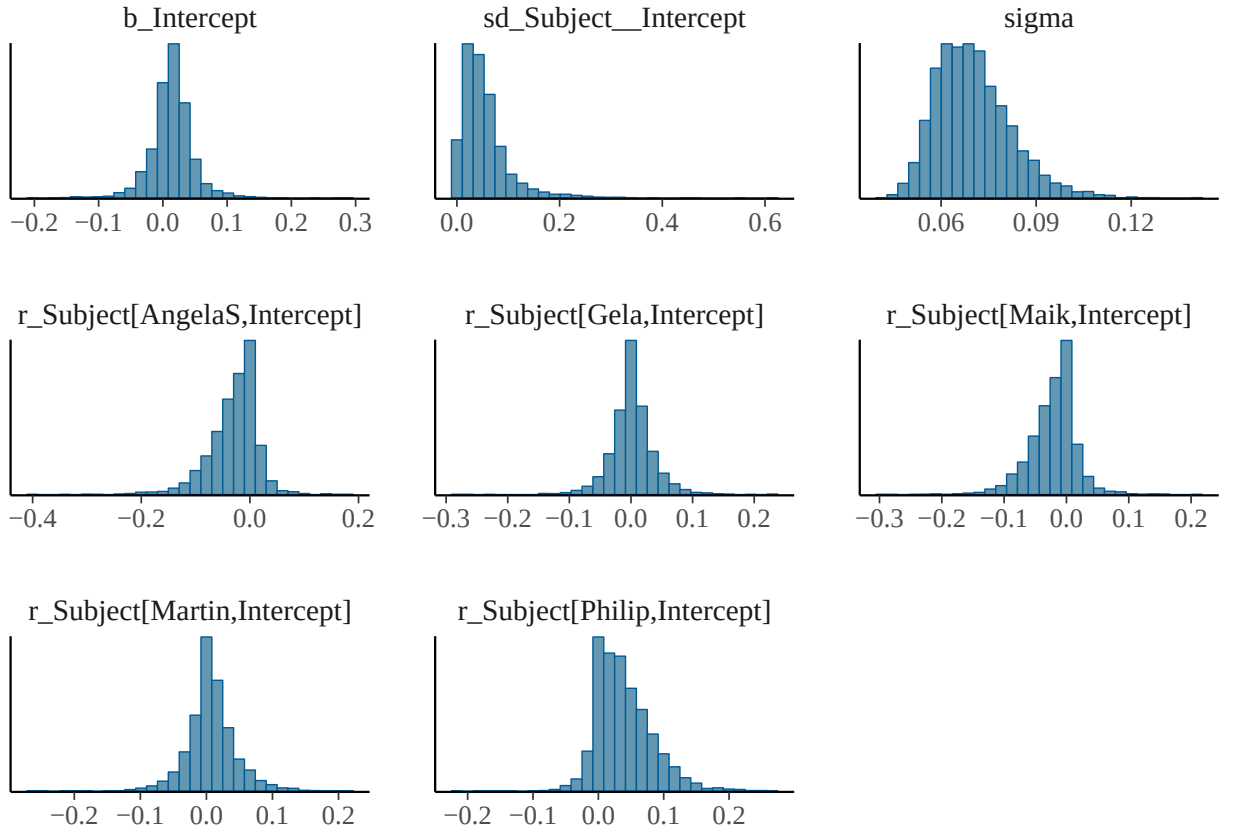
```
## [1] 0.5971786
```

Graphical summaries:

```
stanplot(mdiffMonoE, type="hist",pars=c("Intercept","^sd","sigma"))
```




```
stanplot(mdiffNeutroE, type="hist",pars=c("Intercept","^sd","sigma"))
```



Combined cells

Maybe it makes sense to have both cells' data combined and just control for Cell differences:

```
priors_normal_diff_cell <- c(set_prior("normal(0, 10)", class = "Intercept"),
            set_prior("normal(0, 1)", class = "b"),
            set_prior("normal(0, 1)", class = "sd"),
            set_prior("normal(0, 1)", class = "sigma"),set_prior("lkj(2)", class = "cor")
            )

priors_cauchy_diff_cell <- c(set_prior("cauchy(0, 10)", class = "Intercept"),
            set_prior("cauchy(0, 10)", class = "b"),
            set_prior("cauchy(0, 10)", class = "sd"),
            set_prior("cauchy(0, 10)", class = "sigma"),set_prior("lkj(2)", class = "cor")
            )
```

```
mdiffE <- brm(formula = diff ~ 1+Cell_Type+(1+Cell_Type | Subject),
              data = diffdat, family = gaussian(), prior = priors_cauchy_diff_cell,
              iter = 2000, chains = 4, control = list(adapt_delta = 0.999))
```

```
## Compiling the C++ model
```

```
## Start sampling
```

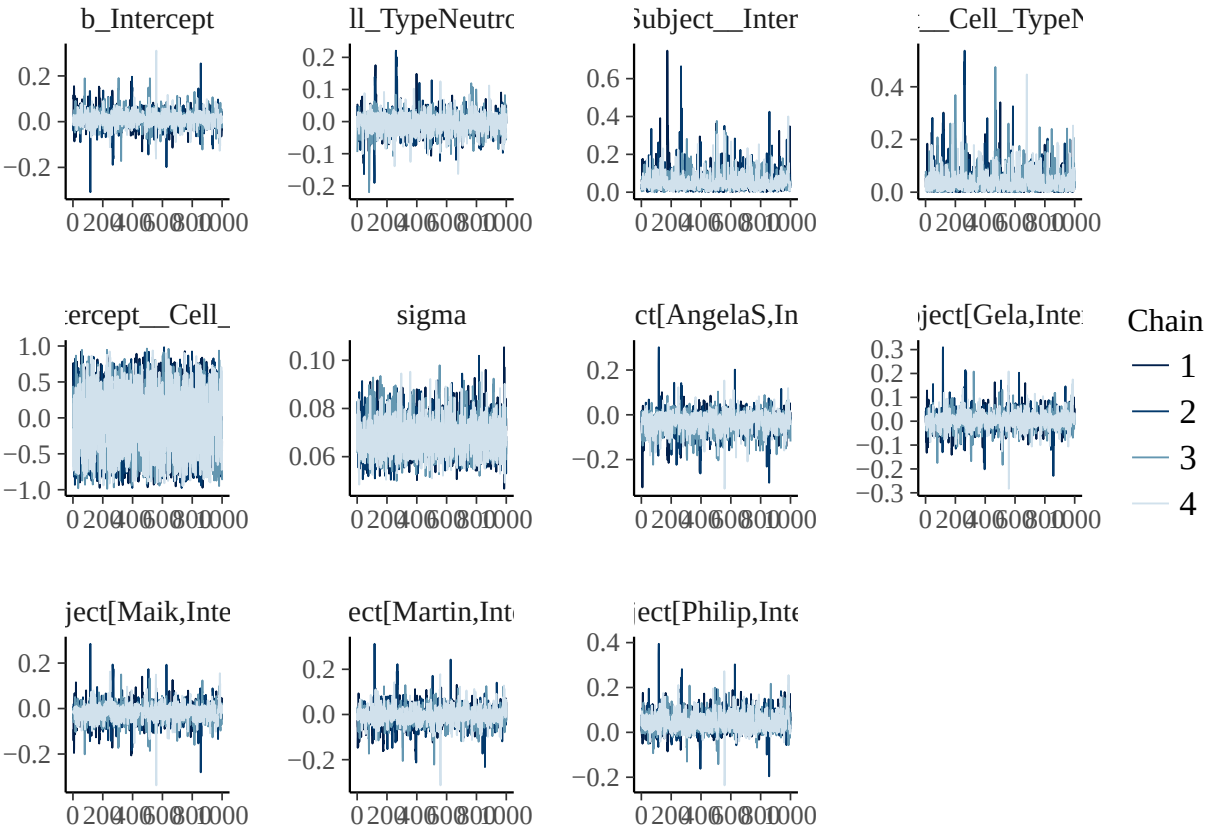
```
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 1).
##
## Gradient evaluation took 2.9e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.29 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 1.58128 seconds (Warm-up)
##                0.646591 seconds (Sampling)
##                2.22787 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 2).
##
## Gradient evaluation took 1.4e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.14 seconds.
## Adjust your expectations accordingly!
##
```

```
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 1.60241 seconds (Warm-up)
##                1.5552 seconds (Sampling)
##                3.15761 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 3).
##
## Gradient evaluation took 1.3e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.13 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 1.33207 seconds (Warm-up)
##                0.961433 seconds (Sampling)
##                2.2935 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 4).
##
## Gradient evaluation took 1.5e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.15 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
```

```
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 1.54733 seconds (Warm-up)
##                0.79818 seconds (Sampling)
##                2.34551 seconds (Total)
```

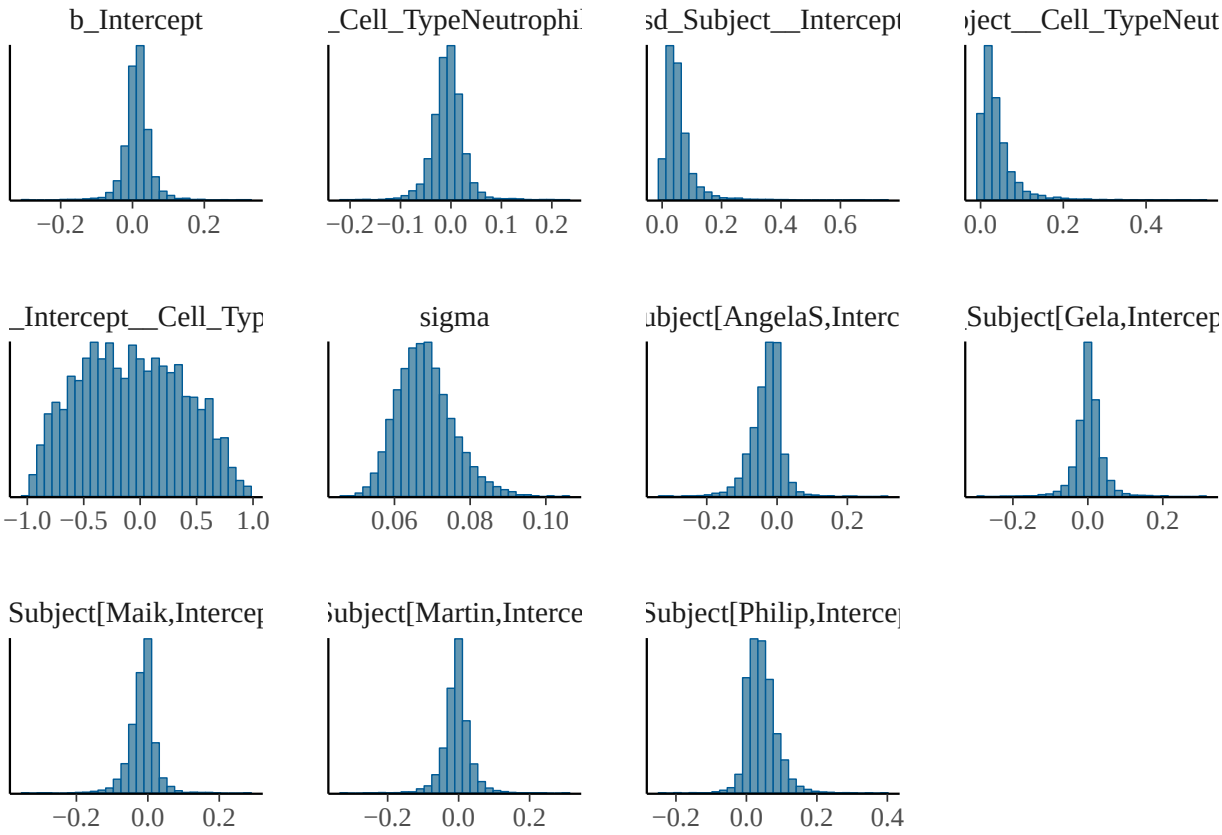
```
stanplot(mdiffE, type="trace",pars=c("Intercept","^b","^sd","sigma"))
```

```
## No divergences to plot.
```




```
stanplot(mdiffE, type="hist",
         pars=c("Intercept","^b","^sd","sigma"))
```

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



In Young's modulus, I don't see any convincing evidence for an effect in either the overall effect (the intercept), or in individual subjects' effects, except Philip, whose intercept seems to be positive; but there isn't a lot of data from Philip; Gela has the most data and he/she shows no effects.

Looking at differences by Contrast Agent

Angela wrote:

- The main variable to be compared is the Young´s modulus (E).

Eric also wrote:

- we were interested in a further analysis that had no control group by itself, but rather four groups, one for each contrast agent (Gad, Dot, Mag, GdCl3), whose measurement points are the difference between each subject's control and treatment measurements for that contrast agent. So, the measurement point for Angela in the Gad group is (Angela-Gad-treatment minus Angela-Gad-control) and the measurement point for Markus in Dotarem is (Markus-Dot-treatment minus Markus-Dot-control).

I implement this analysis next:

```
diffdat<-subset(rtdc_diff,Parameter=="E" & Statistic=="Mean")

## Do the contrast coding manually:
diffdat$Mag_Dot<-ifelse(diffdat$Contrast_Agent=="Dotarem",-1,
                        ifelse(diffdat$Contrast_Agent=="Magnevist",1,0))
diffdat$Gad_Mag<-ifelse(diffdat$Contrast_Agent=="Magnevist",-1,
                        ifelse(diffdat$Contrast_Agent=="Gadovist",1,0))
diffdat$GdCl3_Gad<-ifelse(diffdat$Contrast_Agent=="Gadovist",-1,
```

```
                          ifelse(diffdat$Contrast_Agent=="GdCl3",1,0))

diffMono<-subset(diffdat,Cell_Type=="Monocytes")
diffNeutro<-subset(diffdat,Cell_Type=="Neutrophils")
```

Monocytes

```
priors_cauchy_diff2 <- c(set_prior("cauchy(0, 10)", class = "Intercept"),
            set_prior("cauchy(0, 10)", class = "b"),
            set_prior("cauchy(0, 10)", class = "sd"),
            set_prior("cauchy(0, 10)", class = "sigma")
            )


mdiffMonoContrAgt_E <- brm(formula = diff ~ 1+Mag_Dot+Gad_Mag+GdCl3_Gad+
                (1| Subject),
              data = diffMono, family = gaussian(), prior = priors_cauchy_diff2,
              iter = 2000, chains = 4, control = list(adapt_delta = 0.999))
```

```
## Compiling the C++ model
```

```
## Start sampling
```

```
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 1).
##
## Gradient evaluation took 1.6e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.16 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.397809 seconds (Warm-up)
##                0.253257 seconds (Sampling)
##                0.651066 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 2).
##
## Gradient evaluation took 1e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.1 seconds.
```

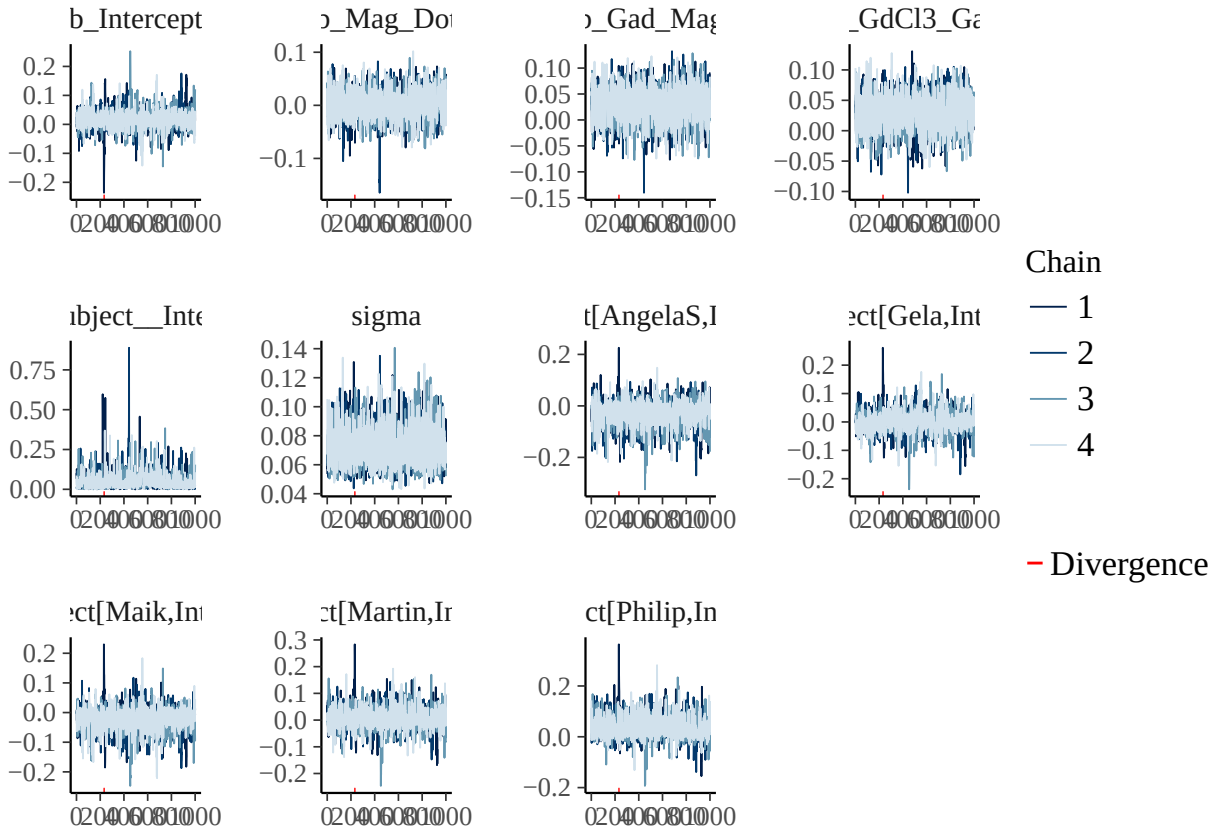
```
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.608445 seconds (Warm-up)
##                0.374875 seconds (Sampling)
##                0.98332 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 3).
##
## Gradient evaluation took 1e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.1 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.466676 seconds (Warm-up)
##                0.251383 seconds (Sampling)
##                0.718059 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 4).
##
## Gradient evaluation took 9e-06 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.09 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
```

```
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.461424 seconds (Warm-up)
##                0.20523 seconds (Sampling)
##                0.666654 seconds (Total)
```

```
## Warning: There were 1 divergent transitions after warmup. Increasing adapt_delta above 0.999 may hel
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

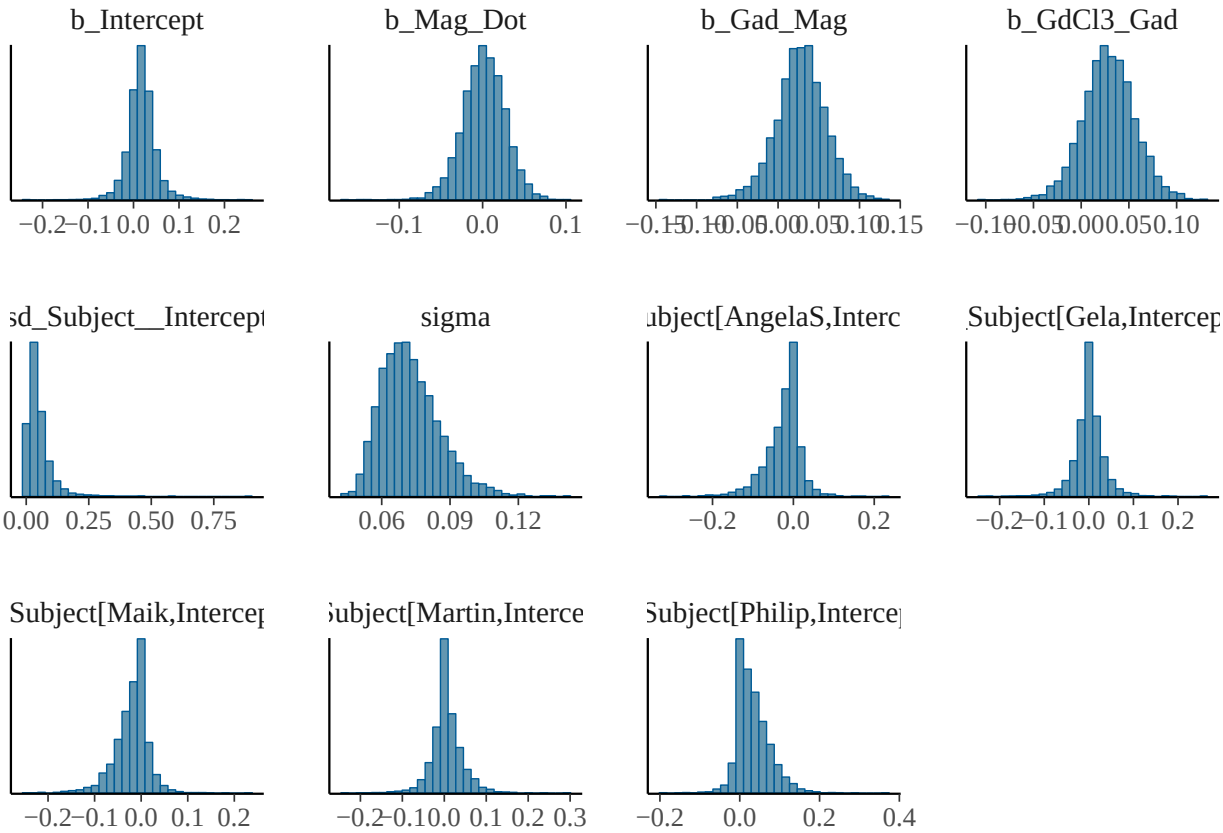
```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

```
stanplot(mdiffMonoContrAgt_E, type="trace",pars=c("Intercept","^b","^sd","sigma"))
```




```
stanplot(mdiffMonoContrAgt_E, type="hist",pars=c("Intercept","^b","^sd","sigma"))
```

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




```
mMonoContrAgtdiffEpost<-posterior_samples(mdiffMonoContrAgt_E, "^b")
```

For Monocytes, one could argue that there is some *weak* evidence for increasing differences by Contrast Agent:

```
mean(mMonoContrAgtdiffEpost$b_Mag_Dot>0)
```

```
## [1] 0.54025
```

```
mean(mMonoContrAgtdiffEpost$b_Gad_Mag>0)
```

```
## [1] 0.83675
```

```
mean(mMonoContrAgtdiffEpost$b_GdCl3_Gad>0)
```

```
## [1] 0.86175
```

Neutrophils

```
mdiffNeutroContrAgt_E <- brm(formula = diff ~ 1+Mag_Dot+Gad_Mag+GdCl3_Gad+
                (1| Subject),
              data = diffNeutro, family = gaussian(), prior = priors_cauchy_diff2,
              iter = 2000, chains = 4, control = list(adapt_delta = 0.999))
```

```
## Compiling the C++ model
```

```
## Start sampling
```

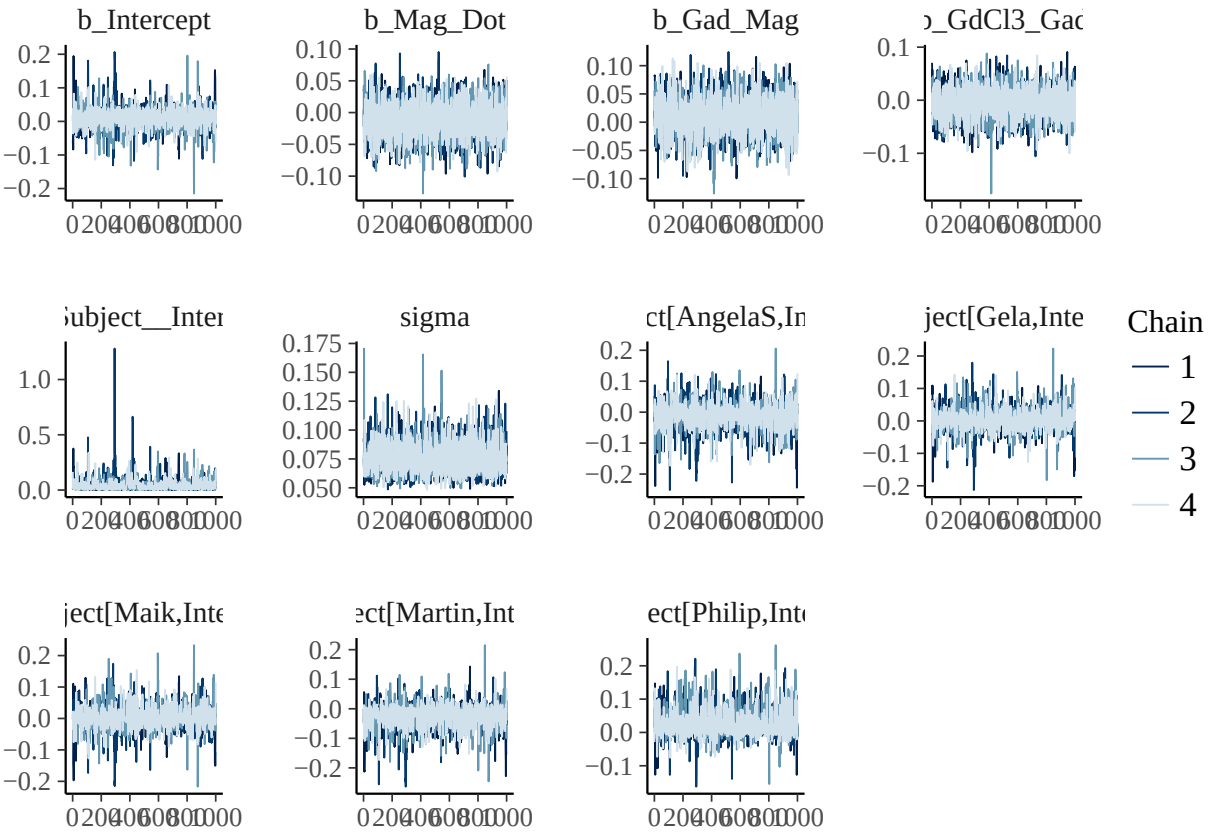
```
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 1).
##
```

```
## Gradient evaluation took 1.7e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.17 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.443451 seconds (Warm-up)
##                0.541905 seconds (Sampling)
##                0.985356 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 2).
##
## Gradient evaluation took 9e-06 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.09 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.478775 seconds (Warm-up)
##                0.50718 seconds (Sampling)
##                0.985955 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 3).
##
## Gradient evaluation took 9e-06 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.09 seconds.
## Adjust your expectations accordingly!
##
```

```
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.467046 seconds (Warm-up)
##                0.290476 seconds (Sampling)
##                0.757522 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 4).
##
## Gradient evaluation took 9e-06 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.09 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  200 / 2000 [ 10%]  (Warmup)
## Iteration:  400 / 2000 [ 20%]  (Warmup)
## Iteration:  600 / 2000 [ 30%]  (Warmup)
## Iteration:  800 / 2000 [ 40%]  (Warmup)
## Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Iteration: 2000 / 2000 [100%]  (Sampling)
##
##  Elapsed Time: 0.473512 seconds (Warm-up)
##                0.282206 seconds (Sampling)
##                0.755718 seconds (Total)
```

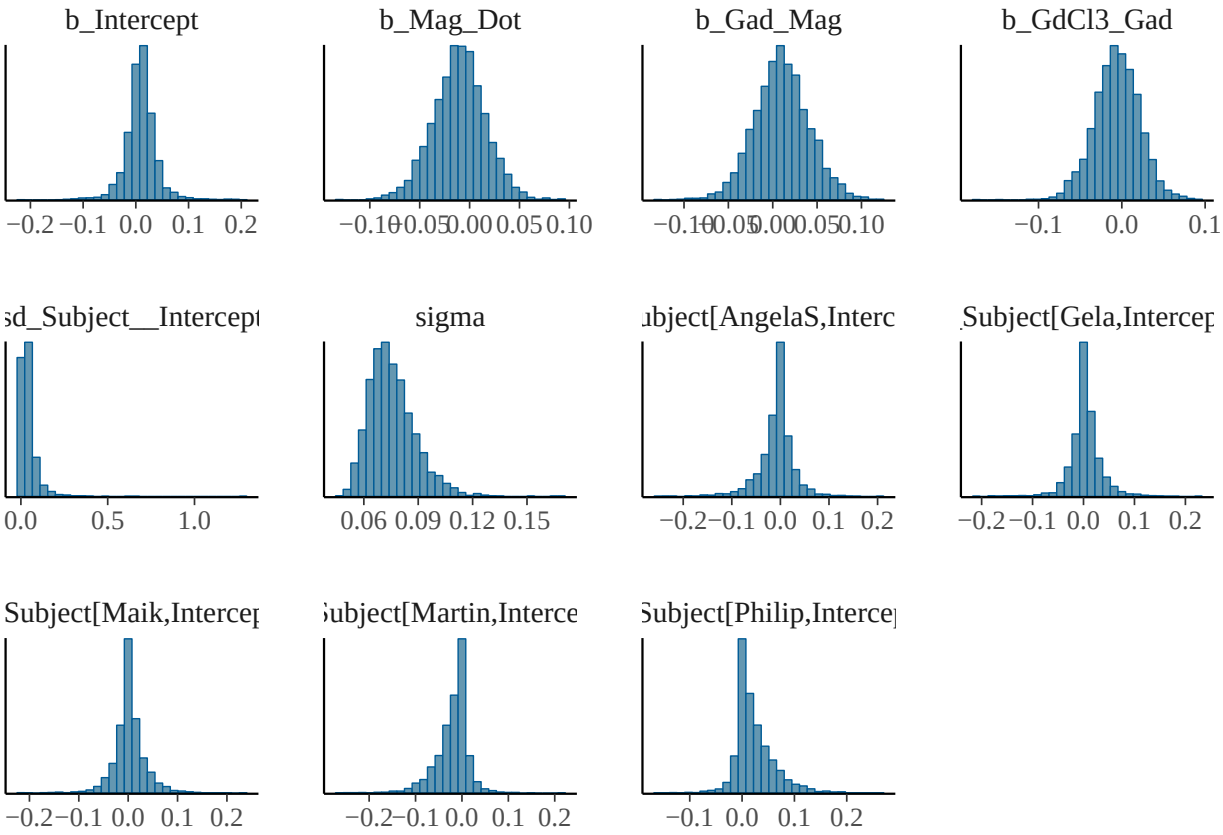
```
stanplot(mdiffNeutroContrAgt_E, type="trace",pars=c("Intercept","^b","^sd","sigma"))
```

```
## No divergences to plot.
```




```
stanplot(mdiffNeutroContrAgt_E, type="hist",pars=c("Intercept","^b","^sd","sigma"))
```

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




```
mNeutroContrAgtdiffEpost<-posterior_samples(mdiffNeutroContrAgt_E, "^b")
```

For Neutrophils, there isn't much evidence for increasing differences by Contrast Agent:

```
mean(mNeutroContrAgtdiffEpost$b_Mag_Dot>0)
```

```
## [1] 0.3415
```

```
mean(mNeutroContrAgtdiffEpost$b_Gad_Mag>0)
```

```
## [1] 0.635
```

```
mean(mNeutroContrAgtdiffEpost$b_GdCl3_Gad>0)
```

```
## [1] 0.4175
```