Cellular Impact of Contrast Agents

Angela Ariza de Schellenberger, Eric Barnhill and Shravan Vasishth

2018.05.14

Overview

This study evaluated whether novel cell measurement technologies showed differences in cell behavior between three widely used MRI contrast agents: Gadovist, Magnevist and Dotarem. These three agents are considered to have differing levels of safety and toxicity and further insight into the relative biological impact of these agents could have immediate impact for MRI protocols worldwide. To evalaute the relative toxicitiy of these agents we investigated whether we could detect differences between their effects on varying kinds of human cells. We measured differences with two different measurement techniques: Time-Of-Flight Mass Cytometry (CyTOF) and Realtime Deformation of Cells (RT-DC).

While these methods produce many thousands of output points, many of the analyses recently published with these novel technologies take point estimates on central tendency measures (such as mean or median) of sparse data sets to estimate the statistical significance of effects. These estimates are likely to be underpowered and to overstate effects, (Vashishth, Gelman et al.), fail to accurately estimate uncertainty or correlations between parameters (when they can even be estimated), and take no advantage of the distribution of thousands of data points available with each measurement.

To optimally address these research questions, we developed an up-to-date, Bayesian statistical approach that can be applied to both of these new cell-measurement technologies as well as others. This approach incorporates quantile summary statistics to more fully estimate the distribution of individual measurements; applies orthogonal contrast codings; handles more fully specified models even with relatively sparse data, and produces a probability mass that enables posterior evaluation of the uncertainty of parameter estimates. Particularly with novel technologies and pilot studies, evaluation of uncertainty in parameter estimates is key to robust and reproducible research, and we provide a method of delivering both estimate and uncertainty that is straightforward to use, code and interpret, that may be useful for a wide range of biological measurement tools.

CyTOF

Time-Of-Flight Mass Cytometry (CyTOF) measures spectral response of different cell types at the cellular level. Here the research question was whether there were differences in performance between three clinically used Gadolinium compounds: Gadovist, Magnevist and Dotarem. As these three compounds have different safety levels, investigating whether the signal levels were also different could aid clinical policy in choosing which compound to use.

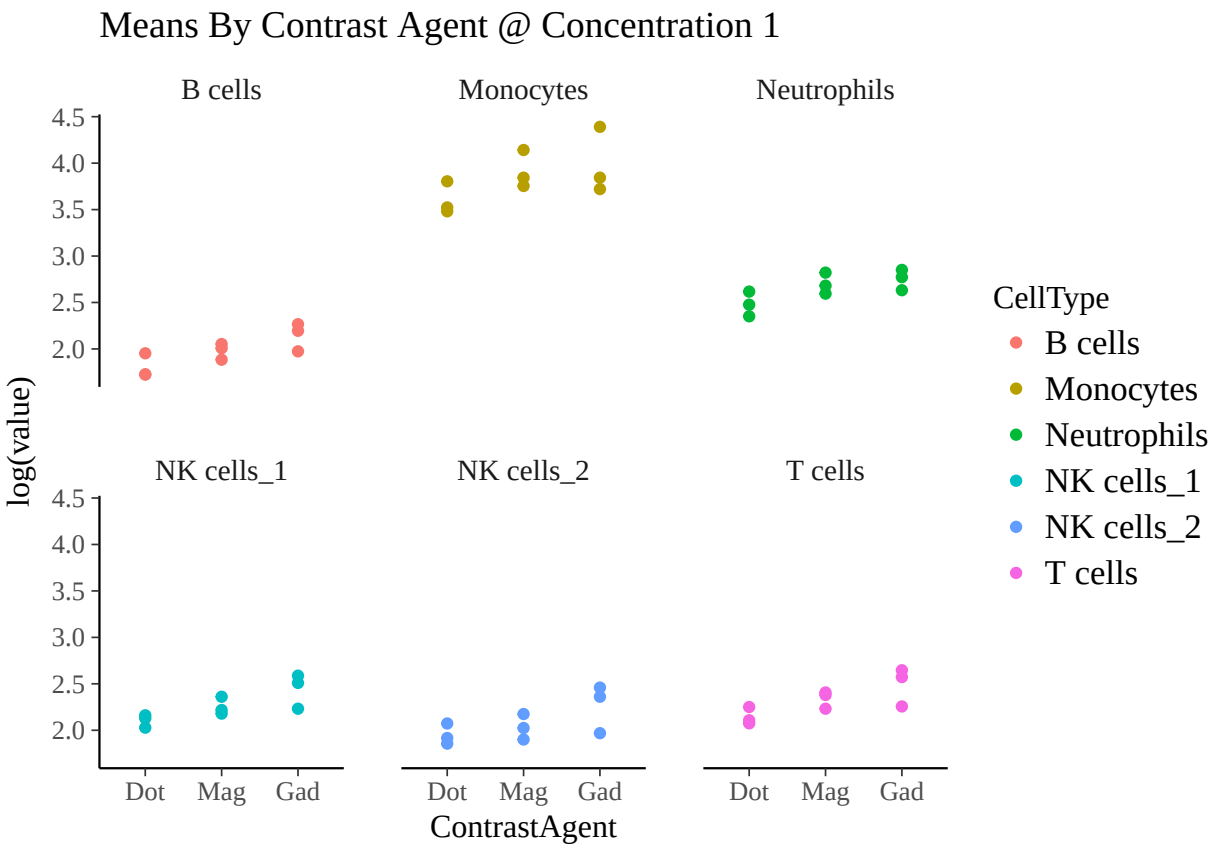
Samples were analyzed for each of six cell types. In this pilot study a single subject was analyzed three times at three different concentration levels.

Our specific statistical question is whether there is an effect of contrast agent on the signal, independent of trial and cell type. It is clear that results will covary with cell type, but we expect the distribution around the "experiment" parameter to be iid.

Visualizing the means

Here we plot means at concentration 1 for each contrast agent within each cell type. There seems good evidence of an effect where Dot $<$ Mag and Mag $<$ Gad. We use this sliding contrast coding.

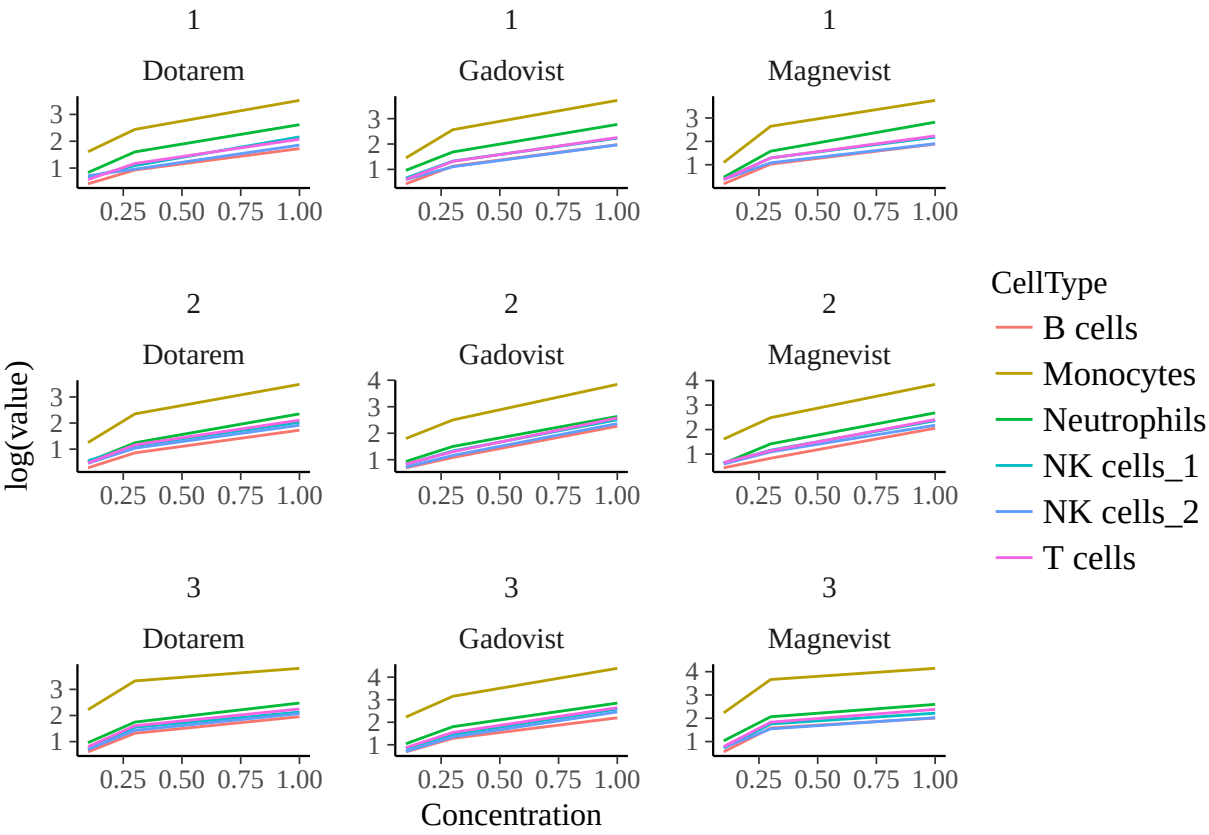
```
means_conc_1 <- subset(means, (Concentration == 1))
means_conc_1$ContrastAgent <- factor(means_conc_1$ContrastAgent, c("Dotarem", "Magnevist", "Gadovist"))
plt <- ggplot(means_conc_1) +
    geom_point(aes(x=ContrastAgent, y=log(value), color=CellType)) +
    facet_wrap(~ CellType, ncol=3) +
    ggtitle("Means By Contrast Agent @ Concentration 1") +
    scale_x_discrete(labels = c("Dot", "Mag", "Gad"))
print(plt)
```



Log-log means model

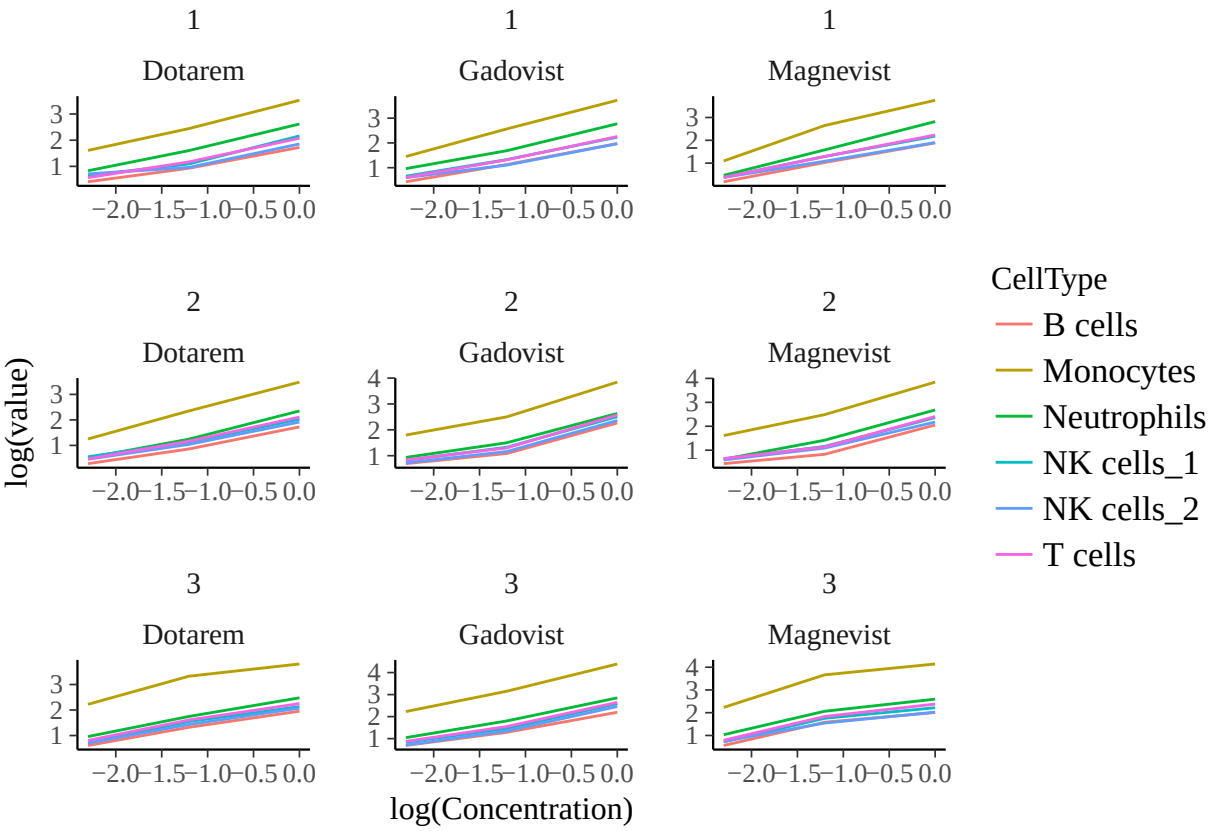
The distributions of the CyTOF data are well modelled by a lognormal distribution. However, the log means are then not linear in the concentration:

```
ggplot(subset(means, ContrastAgent != 'control')) +
    geom_line(aes(x=Concentration, y=log(value), group=CellType, color=CellType)) +
    facet_wrap(~ Experiment + ContrastAgent, scales='free')
```



Modelling log signal against log concentration appears to be roughly linear [to do – Shravan, can we justfiy this better?):

```
ggplot(subset(means, ContrastAgent != 'control')) +
    geom_line(aes(x=log(Concentration), y=log(value), group=CellType, color=CellType)) +
    facet_wrap(~ Experiment + ContrastAgent, scales='free')
```



In contrast to the previous normal model, which evaluated slopes, here we model the slope as uniform and the change as occurring in the *intercept* of the data. Consequently we will interpret change in *intercept* as an effect. The model is as follows:

```
## hand-coded sliding contrasts:
means$GvsM<-ifelse(means$ContrastAgent=="Gadovist",1,
           ifelse(means$ContrastAgent=="Magnevist",-1,0))
means$MvsD<-ifelse(means$ContrastAgent=="Magnevist",1,
           ifelse(means$ContrastAgent=="Dotarem",-1,0))

priors<-c(set_prior("cauchy(0,10)", class = "b"),
          set_prior("normal(0,1)", class = "sigma"))

means_log = means
means_log$value = log(means_log$value)
means_log$Concentration = log(means_log$Concentration)

cytof_brm_log<-brm(formula = value ~
            Concentration +
            GvsM+
            MvsD+
            Concentration:GvsM +
            Concentration:MvsD +
                (1 +
                Concentration +
                GvsM+
                MvsD+
                Concentration:GvsM +
```

```
                Concentration:MvsD
                | CellType),
            data = subset(means_log,ContrastAgent!="control"),
            family = gaussian(),
            prior = priors,
            warmup = 1000,
            iter = 2000,
            chains = 4,
            control = list(adapt_delta = 0.99,max_treedepth=15))
```

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

```
## Start sampling
```

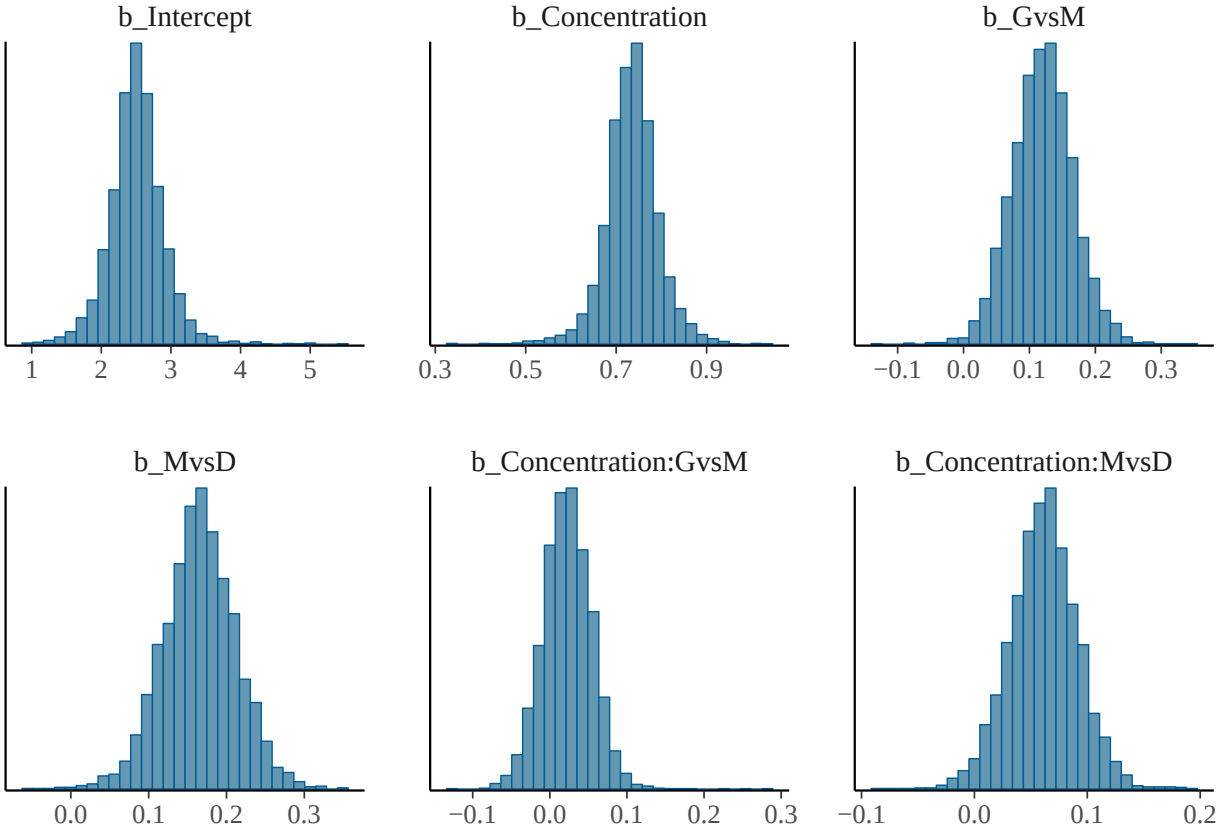
```
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 1).
##
## Gradient evaluation took 9.7e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.97 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  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: 17.0904 seconds (Warm-up)
##                14.9998 seconds (Sampling)
##                32.0901 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 2).
##
## Gradient evaluation took 5.6e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.56 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: 16.1738 seconds (Warm-up)
##                11.3705 seconds (Sampling)
##                27.5442 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 3).
##
## Gradient evaluation took 5.1e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.51 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: 15.2152 seconds (Warm-up)
##                11.7828 seconds (Sampling)
##                26.998 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 4).
##
## Gradient evaluation took 5.2e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.52 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: 16.7243 seconds (Warm-up)
##                12.8593 seconds (Sampling)
##                29.5835 seconds (Total)
```

```
stanplot(cytof_brm_log, type="hist",pars=c("^b"))
```

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



We then want to view the exponent of the posterior distributions, this time interested in the intercept:

```
## [1] "95% HDI, log(Gad) > log(Mag) :"
```

```
##    lower    upper
## 1.029186 1.232979
## attr(,"credMass")
## [1] 0.95
```

```
## [1] "95% HDI, log(Mag) > log(Dot) :"
```

```
##    lower    upper
## 1.073637 1.289291
## attr(,"credMass")
## [1] 0.95
```

In the log case the relationship is multiplicative, so if 95% of the probability mass is greater than 1, there is a 95% confident effect that the contrast agent increases the signal.

Predictive modelling

While the goal of the analysis was to find the impact of the contrast agent independent of cell type and experiment, it leaves the result somewhat abstract. We can use the model to predict, for example, what impact a change of contrast agent will have on each cell types. For example we can ask the question, if we use Gadovist instead of Magnevist, what is the expected increase in signal for each unit of concentration, as estimated at concentration 1:

[TO DO]

Latent error model

While many applications of CyTOF simply use the central tendency statistic, CyTOF machines typically produce a range of summary statistics including quantiles. Exploiting these additional statistics should create a more robust model in a way that is clinically easy to apply and interpret, and therefore of benefit to the CyTOF community.

```
## extract lower quantiles:
qlow<-subset(mass_cyto_tall,MeasurementType=="pct_05")
## extract upper quantiles:
qhigh<-subset(mass_cyto_tall,MeasurementType=="pct_95")

mean_se = means
mean_se$value <- log(mean_se$value)
# test
#mean_se$value <- (mean_se$value)
mean_se$qlow <- log(qlow$value)
# test
#mean_se$qlow <- (qlow$value)
mean_se$qdiff_lo <- mean_se$value - mean_se$qlow

mean_se$qhigh <- log(qhigh$value)
# test
#mean_se$qhigh <- (qhigh$value)
mean_se$qdiff_hi <- mean_se$qhigh - mean_se$value

mean_se$SE<-(mean_se$qdiff_hi) / 1.64
mean_se$Concentration <- log(mean_se$Concentration)
mean_se <- mean_se[c("ContrastAgent", "Concentration", "CellType", "GvsM", "MvsD", "qlow", "qdiff_lo",
```

We can see that the current approach to the SE of the model – subtracting log(hi) from log(mean) – does not appear to be a good choice. The difference between log(lo) and log(mean), and between log(hi) and log(mean), are often very different:

```
head(mean_se[,6:ncol(mean_se)])
```

```
##         qlow qdiff_lo      value  qdiff_hi     qhigh        SE
## 1 -1.5141277 1.290984 -0.2231436 0.8649974 0.6418539 0.5274375
## 2 -1.3470736 1.810808  0.4637340 1.0246656 1.4883996 0.6247961
## 3 -0.5978370 2.176816  1.5789787 1.0012381 2.5802168 0.6105111
## 4  1.0543120 1.767067  2.8213789 1.0180734 3.8394523 0.6207765
## 5 -1.2039728 2.036882  0.8329091 1.0280654 1.8609745 0.6268692
## 6 -0.4620355 2.065455  1.6034198 0.9615295 2.5649494 0.5862985
```

Nonetheless, we can now run the same fully-specified brms log-log model including the standard error term:

```
priors<-c(set_prior("cauchy(0,10)", class = "b"),
          set_prior("cauchy(0,5)", class = "b",coef="Concentration"),
          set_prior("normal(0,1)", class = "sd"),
          set_prior("lkj(2)", class = "cor"))

cytof_brm_se <-brm(formula = value | se(SE) ~
            Concentration +
            GvsM+
            MvsD+
            Concentration:GvsM +
            Concentration:MvsD +
                (1 +
                Concentration +
                GvsM+
                MvsD+
                Concentration:GvsM +
                Concentration:MvsD
                | CellType),
            data = subset(mean_se,ContrastAgent!="control"),
            family = gaussian(),
            prior = priors,
            warmup = 1000,
            iter = 2000,
            chains = 4,
            control = list(adapt_delta = 0.99,max_treedepth=15))
```

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

```
## Start sampling
```

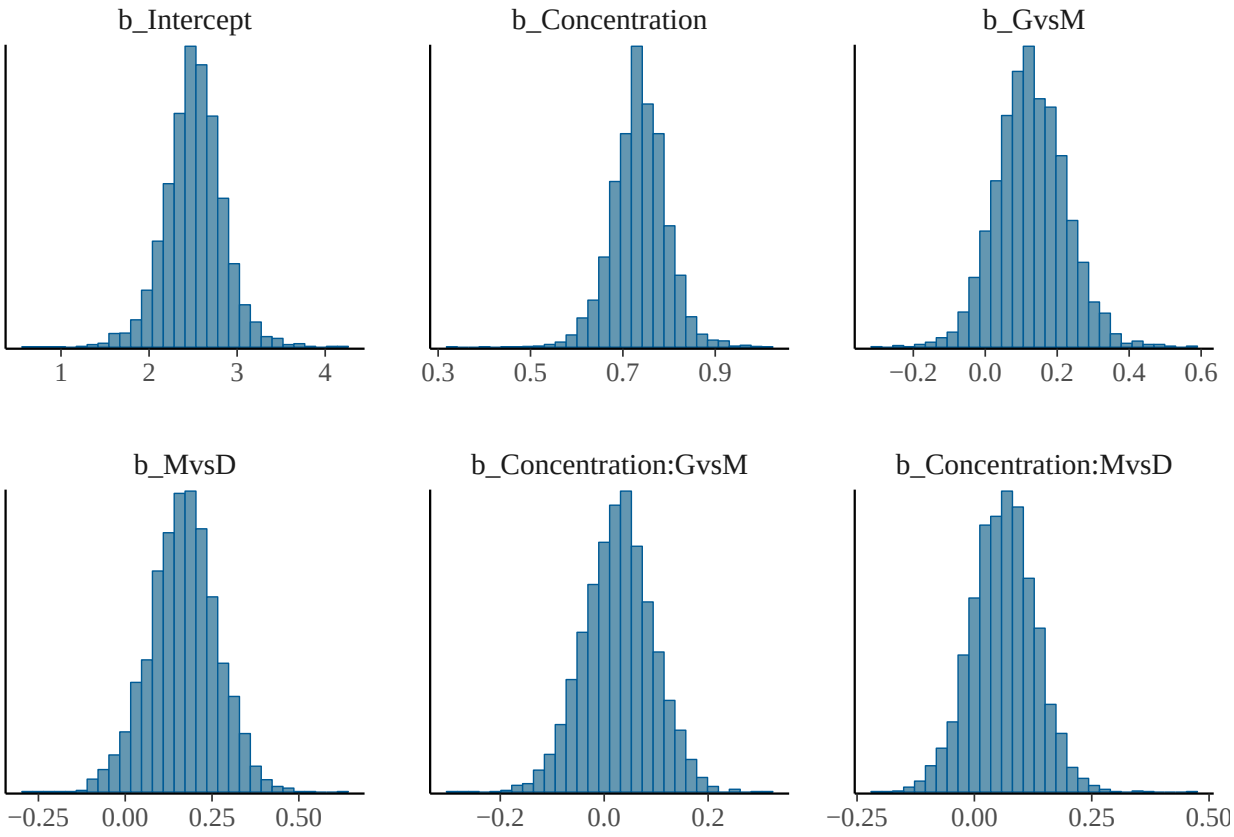
```
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 1).
##
## Gradient evaluation took 9.1e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.91 seconds.
## Adjust your expectations accordingly!
##
##
## Iteration:    1 / 2000 [  0%]  (Warmup)
## Iteration:  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: 8.50049 seconds (Warm-up)
##                4.23049 seconds (Sampling)
##                12.731 seconds (Total)
##
```

```
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 2).
##
## Gradient evaluation took 7.5e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.75 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: 7.27945 seconds (Warm-up)
##                3.59404 seconds (Sampling)
##                10.8735 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 3).
##
## Gradient evaluation took 5.4e-05 seconds
## 1000 transitions using 10 leapfrog steps per transition would take 0.54 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: 8.40579 seconds (Warm-up)
##                8.55141 seconds (Sampling)
##                16.9572 seconds (Total)
##
##
## SAMPLING FOR MODEL 'gaussian brms-model' NOW (CHAIN 4).
##
## Gradient evaluation took 5.2e-05 seconds
```

```
## 1000 transitions using 10 leapfrog steps per transition would take 0.52 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: 7.24825 seconds (Warm-up)
##                6.37841 seconds (Sampling)
##                13.6267 seconds (Total)
```

```
stanplot(cytof_brm_se, type="hist",pars=c("^b"))
```

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



Let's see if there is any difference in the confidence intervals:

```
## [1] "95% HDI, log(Gad) > log(Mag) :"
```

```
##     lower     upper
```

```
## 0.9280932 1.3596343
## attr(,"credMass")
## [1] 0.95
```

```
## [1] "95% HDI, log(Mag) > log(Dot) :"
```

```
##     lower     upper
## 0.9691777 1.4265403
## attr(,"credMass")
## [1] 0.95
```

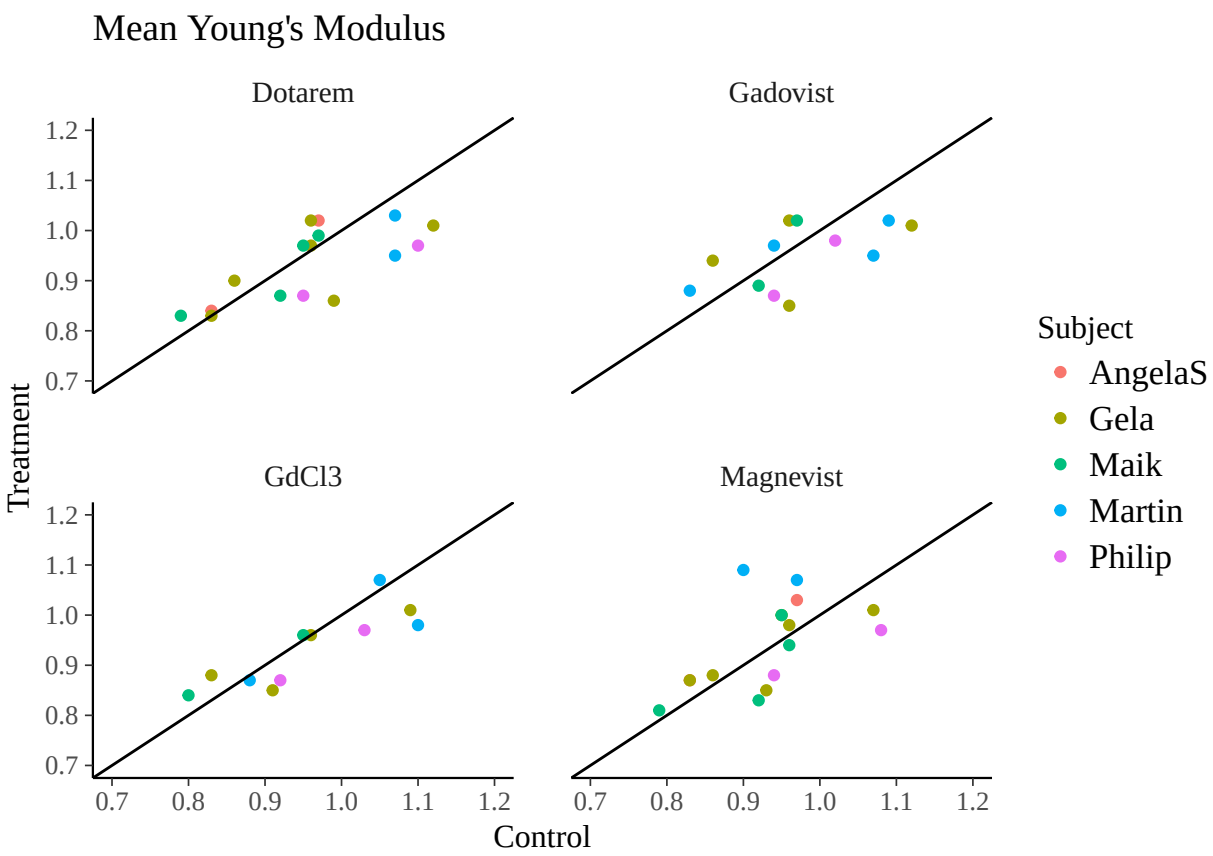
This approach as presently coded simply widens the variance of the estimated parameters. Central tendency is still similar (means of 1.1289004 and 1.1810095 for the log-log model without SE; and means of 1.1414745 and 1.1880338 for the log-log model with SE. Including the error creates less confidence in the estimate and takes us outside a 95% confidence interval for the effects. However, I do understand that small studies tend to be underpowered so if this is a superior model, this is what we should report.

RT-DC

RT-DC is a method of measuring cellular elastic deformations developed by TU-Dresden. We use this technology to evaluate whether cells containing common contrast agents show differences in the Young's modulus, which would indicate that their physiological properties are being altered by the contrast agent. [more details to come]

The RT-DC method is currently experimental, and control tests for an individual subject produce a wide range of values. Consequently each measurement with a contrast agent was paired with its own control from the same session. The Figure below shows pairwise control-treatment plots colored by subject:

```
## Warning: Removed 2 rows containing missing values (geom_point).
```



While the range of control values is quite wide (0.008131), the control-treatment pairs deviate an average of 0.0049137 from the unit line, suggesting the control-treatment pairings are a stabler value set.

Monocytes

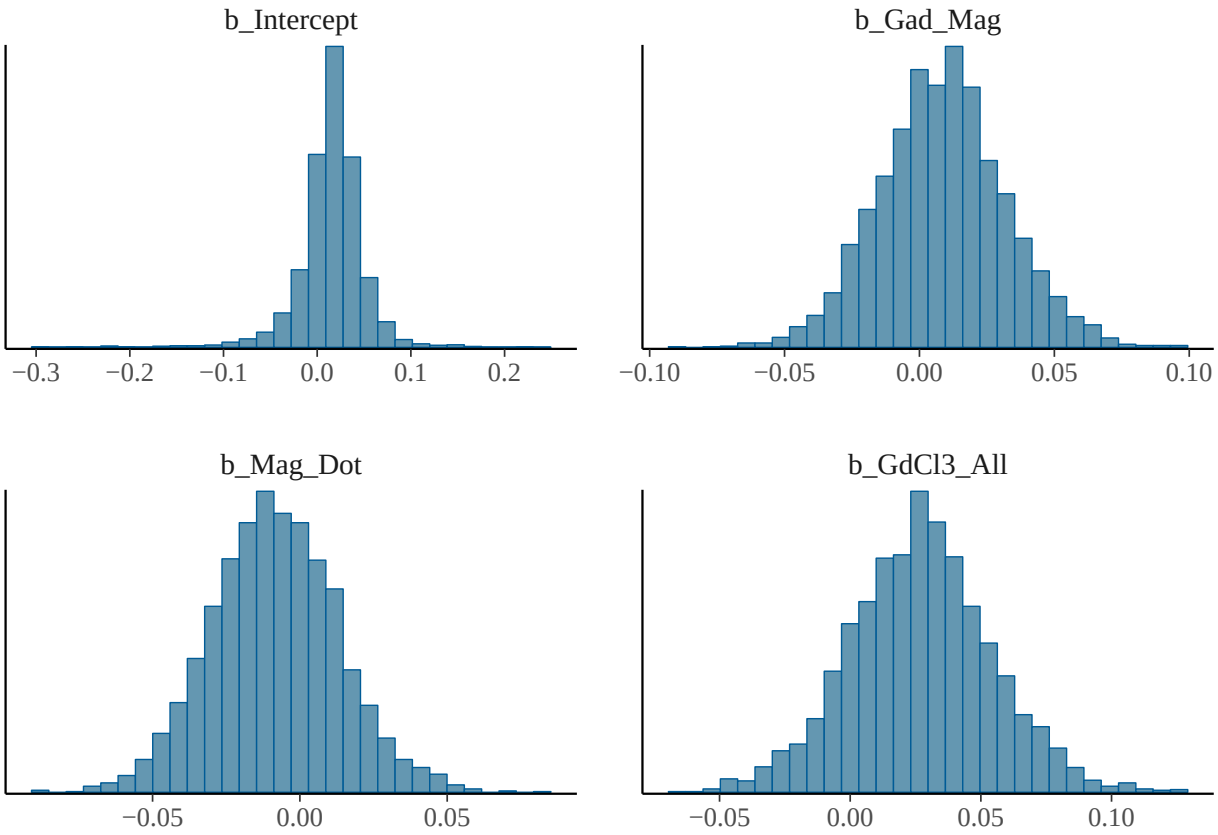
```
priors_cauchy_diff2 <- c(set_prior("cauchy(0, 10)", class = "Intercept"),
            set_prior("cauchy(0, 10)", class = "b"),
            set_prior("cauchy(0, 10)", class = "sd"),
            set_prior("cauchy(0, 10)", class = "sigma")
            )


mdiffMonoContrAgt_E <- brm(formula = diff ~ 1+Gad_Mag+Mag_Dot+GdCl3_All+
                (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
```

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

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




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

For Monocytes, there is weak evidence that GdCl3 causes a decrease in Young's Modulus against the other three contrast agents. The comparison of the individual contrast agents is not determinate at this time:

```
## [1] "Prob Gad > Mag"
```

```
## [1] 0.64275
```

```
## [1] "Prob Mag > Dot"
```

```
## [1] 0.3425
```

```
## [1] "Prob GdCl3 > All"
```

```
## [1] 0.84075
```

Neutrophils

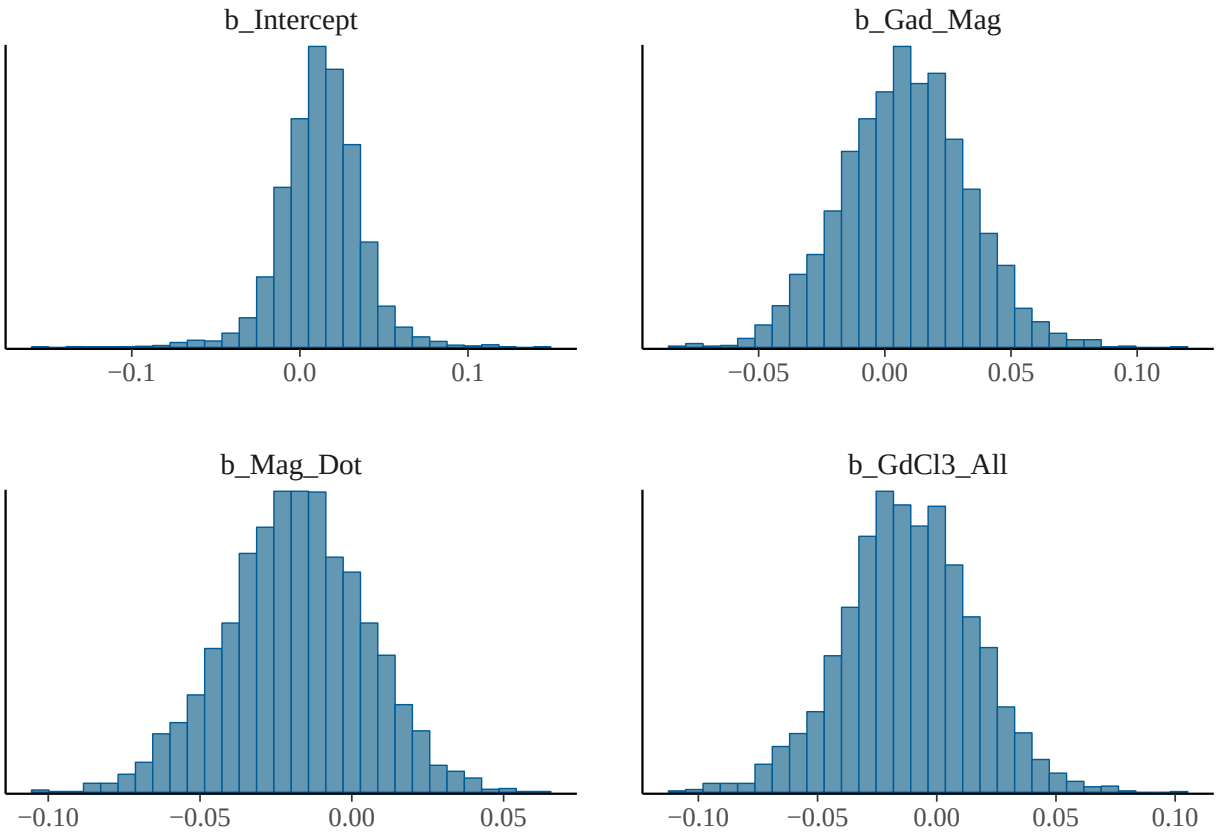
```
mdiffNeutroContrAgt_E <- brm(formula = diff ~ 1+Gad_Mag+Mag_Dot+GdCl3_All+
                (1| Subject),
              data = diffNeutro, family = gaussian(), prior = priors_cauchy_diff2,
              iter = 2000, chains = 4, control = list(adapt_delta = 0.999))
```

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

```
## Start sampling
```

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

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




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

For Neutrophils, there isn't much evidence for differences by Contrast Agent, though there is again maybe some weak evidence (80% likelihood) that Magnevist has higher Young's modulus than Dotarem:

```
## [1] "Prob Gad > Mag"
## [1] 0.636
## [1] "Prob Mag > Dot"
## [1] 0.212
## [1] "Prob GdCl3 > All"
## [1] 0.329
```

Latent Error Model

TODO: Use hacker stats to create distributions off the mean and quantiles, then subtract the two distributions to get a distribution of the differences??