RATE Experiments Notebook

Emily Winn

Introduction

This notebook is for visualizing the initial RATE experiments for the different interactions. These are the baby experiments and more complicated ones will follow.

In each experiment, we generate a data matrix X which is $n = 2000$ by $p = 25$. We set the mixing coefficient between marginal and interactive effects to be $\rho = 0.5$ and $h^2 = 0.6$. In every scenario, we have the marginal effects coming equally from SNPS 23, 24, and 25. We consider three basic scenarios:

- Same - In this situation, SNPs 23 and 24 each interact with 25. Thus all marginal and interactive effects come from these three SNPS.
- Diff - In this situation, SNPs 8 and 9 each interact with SNP 10. The interactive effects come from 8,9, and 10, while the marginal effects come from 23, 24, and 25
- Overlap - In this situation, SNPs 8 and 9 each interact with SNP 25. Thus SNP 25 contributes both interactive and marginal effects.

For each of the generated data sets in each scenario, we run each of the following functions:

- `RATE` - otherwise referred to as "OG RATE", this is the original RATE function from Crawford et al 2018, and captures and ranks linear effects.
- `RATE_combo` - Captures the quadratic effects only, and does this by subtracting out the linear effects.
- `RATE_combo2` - Captures the quadratic effects without the removal of the linear effects.
- `RATE_combo` - Calculates both the linear and quadratic coefficients and adds them together before calculating effect sizes
- `RATE_MC` - First calculates a g function for each SNP, then subtracts the predicted function f and then runs RATE as usual.

The data parameters, the effect sizes, delta, and ESS for each RATE function, and the time to calculation is saved and stored in a series of lists, which we will load below in our analyses. All calculations were conducted on a 32 cores with 256 GB of memory.

Load Libraries and Data

We now load the necessary the data.

```
load('Z:/RATEexpData/RATE_MA_same_2.Rdata')
load('Z:/RATEexpData/RATE_MA_diff_2.Rdata')
load('Z:/RATEexpData/RATE_MA_overlap_2.Rdata')
```

Data Plots

Effect Sizes

First code to put together data frames, which will be hidden in the output of the markdown file.

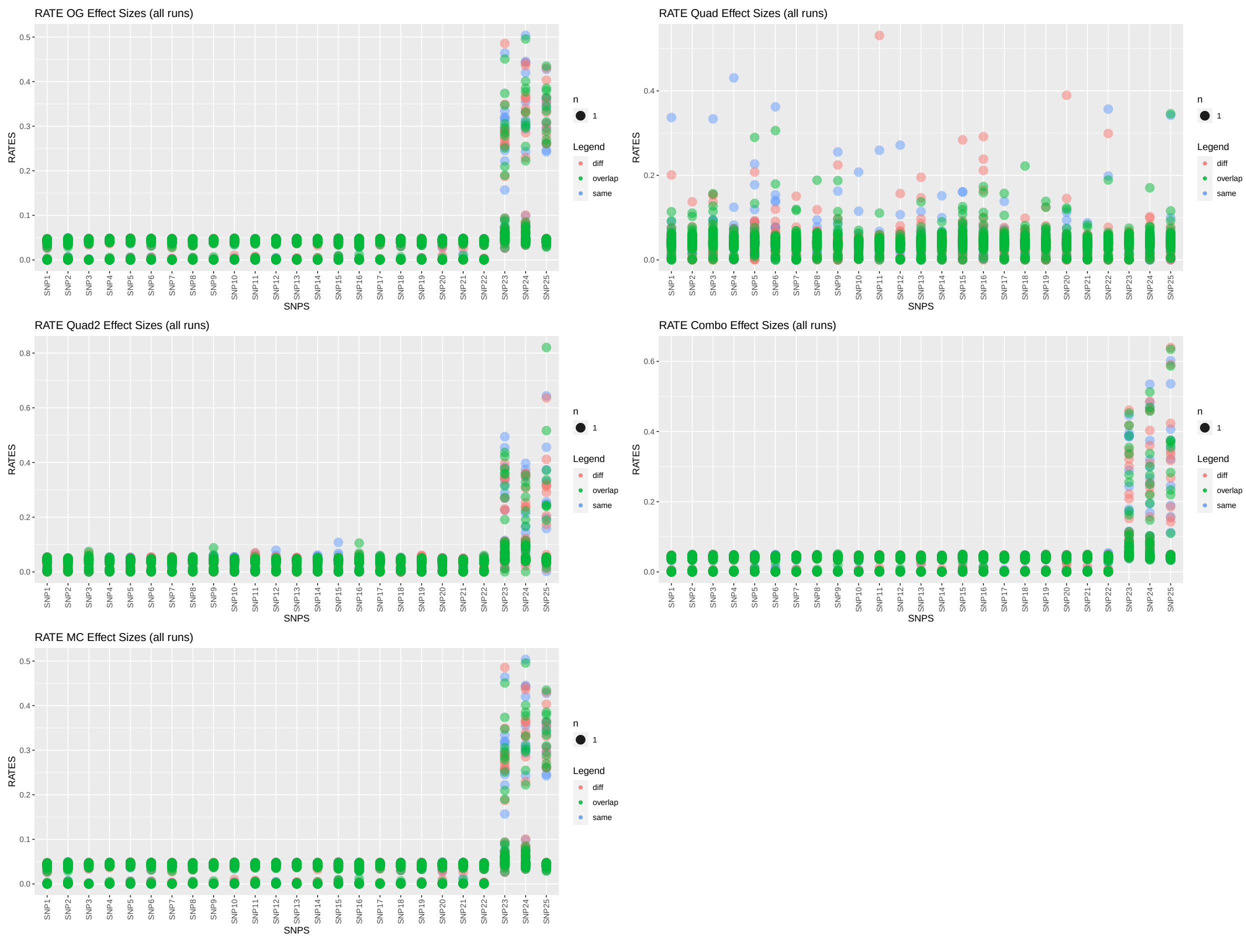
Now we want to plot stuff. There is a lot of code but you can skip all this to get to the graphs. The biggest thing to note is, regardless of scenario, it seems none of the functions were able to pick up on additive effects of 8, 9, or 10 when relevant. No idea why this is. `RATE_quad` and `RATE_quad2` also seem to be extra sensitive to noise.

```r
p_same <- SAME_OG_ESdf %>% select(snps) %>%
  pivot_longer(., cols = snps, names_to="SNPS", values_to = "RATES")
```

```
## Note: Using an external vector in selections is ambiguous.
## i Use ‘all_of(snps)‘ instead of ‘snps‘ to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

```r
p_diff <- DIFF_OG_ESdf %>% select(snps) %>%
  pivot_longer(., cols = snps, names_to="SNPS", values_to = "RATES")

p_overlap <- OVERLAP_OG_ESdf %>% select(snps) %>%
  pivot_longer(., cols = snps, names_to="SNPS", values_to = "RATES")

OG <- ggplot() +
  geom_count(data=p_same, aes(x=SNPS, y = RATES, color='same'), alpha=0.5) +
  geom_count(data=p_diff, aes(x=SNPS,y=RATES, color='diff'), alpha=0.5) +
  geom_count(data=p_overlap, aes(x=SNPS,y=RATES, color='overlap'),  alpha=0.5) +
  scale_x_discrete(limits=snps) +
  labs(color="Legend")+
  theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5), legend.position="right") +
  ggtitle("RATE OG Effect Sizes (all runs)")

#plot the quad

p_same <- SAME_quad_ESdf %>% select(snps) %>%
  pivot_longer(., cols = snps, names_to="SNPS", values_to = "RATES")

p_diff <- DIFF_quad_ESdf %>% select(snps) %>%
  pivot_longer(., cols = snps, names_to="SNPS", values_to = "RATES")

p_overlap <- OVERLAP_quad_ESdf %>% select(snps) %>%
  pivot_longer(., cols = snps, names_to="SNPS", values_to = "RATES")

Quad <- ggplot() +
  geom_count(data=p_same, aes(x=SNPS, y = RATES, color='same'), alpha=0.5) +
  geom_count(data=p_diff, aes(x=SNPS,y=RATES, color='diff'), alpha=0.5) +
  geom_count(data=p_overlap, aes(x=SNPS,y=RATES, color='overlap'), alpha=0.5) +
  scale_x_discrete(limits=snps) +
  labs(color="Legend")+
  theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5), legend.position="right") +
  ggtitle("RATE Quad Effect Sizes (all runs)")
```

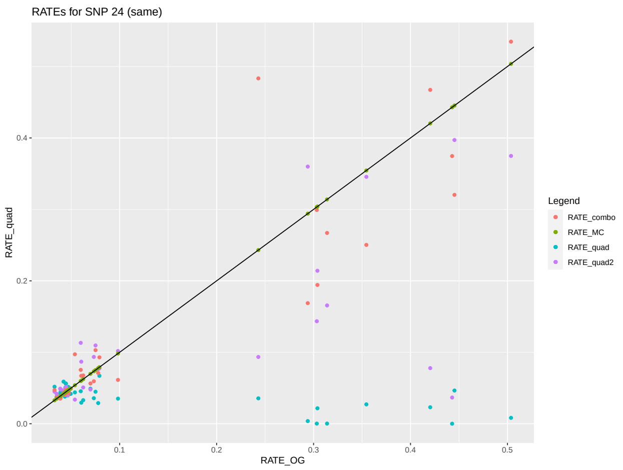
```
#plot the quad2

p_same <- SAME_quad2_ESdf %>% select(snps) %>%
  pivot_longer(., cols = snps, names_to="SNPS", values_to = "RATES")

p_diff <- DIFF_quad2_ESdf %>% select(snps) %>%
  pivot_longer(., cols = snps, names_to="SNPS", values_to = "RATES")

p_overlap <- OVERLAP_quad2_ESdf %>% select(snps) %>%
  pivot_longer(., cols = snps, names_to="SNPS", values_to = "RATES")

Quad2 <- ggplot() +
  geom_count(data=p_same, aes(x=SNPS, y = RATES, color='same'), alpha=0.5) +
  geom_count(data=p_diff, aes(x=SNPS,y=RATES, color='diff'), alpha=0.5) +
  geom_count(data=p_overlap, aes(x=SNPS,y=RATES, color='overlap'), alpha=0.5) +
  scale_x_discrete(limits=snps) +
  labs(color="Legend")+
  theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5), legend.position="right") +
  ggtitle("RATE Quad2 Effect Sizes (all runs)")

#plot the combo

p_same <- SAME_combo_ESdf %>% select(snps) %>%
  pivot_longer(., cols = snps, names_to="SNPS", values_to = "RATES")

p_diff <- DIFF_combo_ESdf %>% select(snps) %>%
  pivot_longer(., cols = snps, names_to="SNPS", values_to = "RATES")

p_overlap <- OVERLAP_combo_ESdf %>% select(snps) %>%
  pivot_longer(., cols = snps, names_to="SNPS", values_to = "RATES")

Combo <- ggplot() +
  geom_count(data=p_same, aes(x=SNPS, y = RATES, color='same'), alpha=0.5) +
  geom_count(data=p_diff, aes(x=SNPS,y=RATES, color='diff'), alpha=0.5) +
  geom_count(data=p_overlap, aes(x=SNPS,y=RATES, color='overlap'), alpha=0.5) +
  scale_x_discrete(limits=snps) +
  labs(color="Legend")+
  theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5), legend.position="right") +
  ggtitle("RATE Combo Effect Sizes (all runs)")

#plot the MC

p_same <- SAME_MC_ESdf %>% select(snps) %>%
  pivot_longer(., cols = snps, names_to="SNPS", values_to = "RATES")

p_diff <- DIFF_MC_ESdf %>% select(snps) %>%
  pivot_longer(., cols = snps, names_to="SNPS", values_to = "RATES")

p_overlap <- OVERLAP_MC_ESdf %>% select(snps) %>%
  pivot_longer(., cols = snps, names_to="SNPS", values_to = "RATES")

MC <- ggplot() +
  geom_count(data=p_same, aes(x=SNPS, y = RATES, color='same'), alpha=0.5) +
  geom_count(data=p_diff, aes(x=SNPS,y=RATES, color='diff'), alpha=0.5) +
  geom_count(data=p_overlap, aes(x=SNPS,y=RATES, color='overlap'), alpha=0.5) +
  scale_x_discrete(limits=snps) +
  labs(color="Legend")+
  theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5), legend.position="right") +
  ggtitle("RATE MC Effect Sizes (all runs)")


#Combine all plots below in 2 x 3 matrix
grid.arrange(OG, Quad, Quad2, Combo, MC, nrow=3)
```

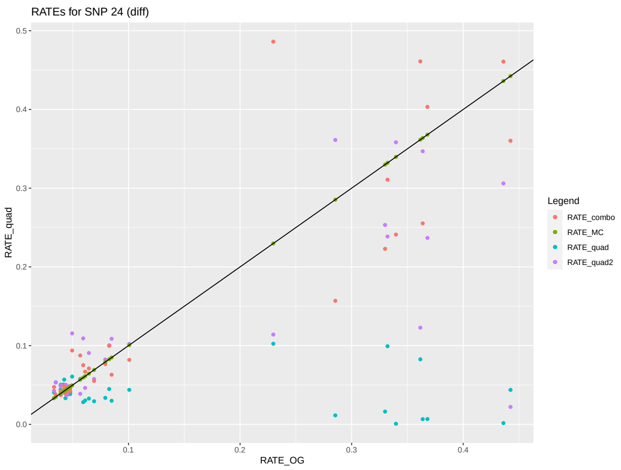


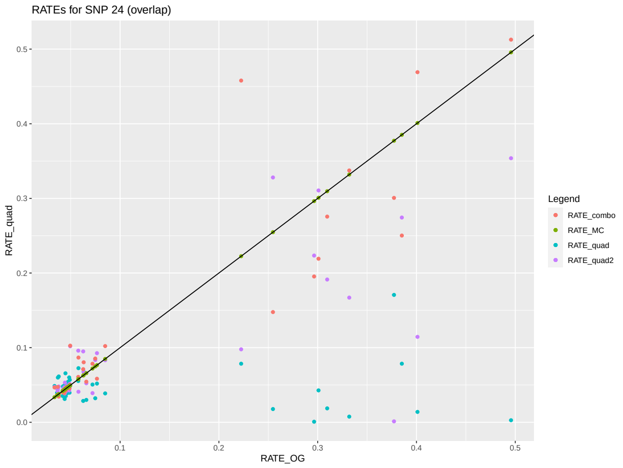
## Correlation of RATE Values (plotted)

Here I will plot various RATE values against each other in different scenarios. Sadly I can only do a snp at a time, but it's fun to play around with. See Statistical analysis section for actual numbers. Again, sorry for all the code. Please skip that and go to the pretty pictures.

```
same_compare = data.frame("RATE_OG"=SAME_OG_ESdf$SNP24, "RATE_quad"=SAME_quad_ESdf$SNP24, "RATE_quad2"

same <- ggplot(same_compare) +
  geom_point(aes(x=RATE_OG, y=RATE_quad, color = "RATE_quad")) +
  geom_point(aes(x=RATE_OG, y=RATE_quad2, color = "RATE_quad2")) +
  geom_point(aes(x=RATE_OG, y=RATE_combo, color = "RATE_combo")) +
  geom_point(aes(x=RATE_OG, y=RATE_MC, color = "RATE_MC")) +
  labs(color = "Legend") +
  geom_abline(slope=1, intercept=0) +
  ggtitle("RATEs for SNP 24 (same)")


diff_compare = data.frame("RATE_OG"=DIFF_OG_ESdf$SNP24, "RATE_quad"=DIFF_quad_ESdf$SNP24, "RATE_quad2" =

diff <- ggplot(diff_compare) +
  geom_point(aes(x=RATE_OG, y=RATE_quad, color = "RATE_quad")) +
  geom_point(aes(x=RATE_OG, y=RATE_quad2, color = "RATE_quad2")) +
  geom_point(aes(x=RATE_OG, y=RATE_combo, color = "RATE_combo")) +
  geom_point(aes(x=RATE_OG, y=RATE_MC, color = "RATE_MC")) +
  labs(color = "Legend") +
  geom_abline(slope=1, intercept=0) +
  ggtitle("RATEs for SNP 24 (diff)")

overlap_compare = data.frame("RATE_OG"=OVERLAP_OG_ESdf$SNP24, "RATE_quad"=OVERLAP_quad_ESdf$SNP24, "RATE

overlap <- ggplot(overlap_compare) +
  geom_point(aes(x=RATE_OG, y=RATE_quad, color = "RATE_quad")) +
  geom_point(aes(x=RATE_OG, y=RATE_quad2, color = "RATE_quad2")) +
  geom_point(aes(x=RATE_OG, y=RATE_combo, color = "RATE_combo")) +
  geom_point(aes(x=RATE_OG, y=RATE_MC, color = "RATE_MC")) +
  labs(color = "Legend") +
  geom_abline(slope=1, intercept=0) +
  ggtitle("RATEs for SNP 24 (overlap)")

grid.arrange(same, diff, overlap, nrow=2)
```







Times of Runs

Code hidden for assembling the data frame for times of each run.

Now we can plot the times comparatively. Again, skip the lengthy code. Analysis at the end.

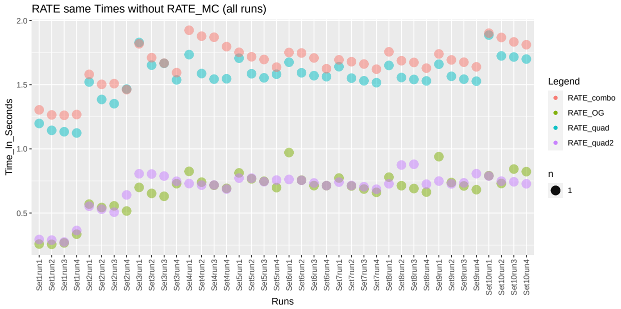
```
#Plot same times
p_sameOG <- Same_OG_df %>% select(time_names) %>%
  pivot_longer(., cols = time_names, names_to = "Runs", values_to = "Time_In_Seconds")
```

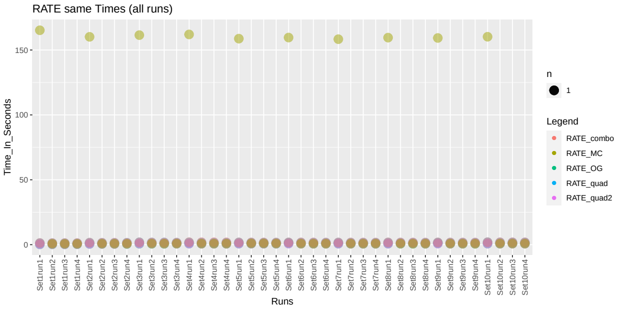
```
## Note: Using an external vector in selections is ambiguous.
## i Use ‘all_of(time_names)‘ instead of ‘time_names‘ to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
```

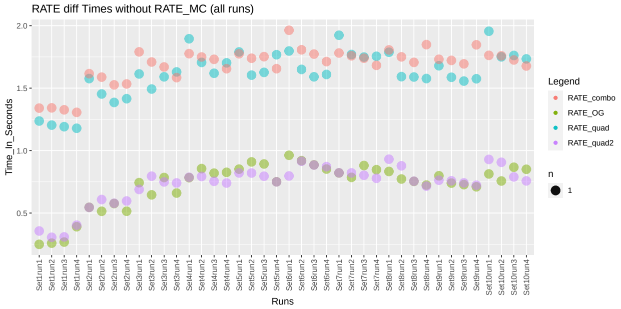
```
p_samequad <- Same_quad_df %>% select(time_names) %>%
  pivot_longer(., cols = time_names, names_to = "Runs", values_to = "Time_In_Seconds")

p_samequad2 <- Same_quad2_df %>% select(time_names) %>%
  pivot_longer(., cols = time_names, names_to = "Runs", values_to = "Time_In_Seconds")

p_samecombo <- Same_combo_df %>% select(time_names) %>%
  pivot_longer(., cols = time_names, names_to = "Runs", values_to = "Time_In_Seconds")

p_sameMC <- Same_mc_df %>% select(time_names) %>%
  pivot_longer(., cols = time_names, names_to = "Runs", values_to = "Time_In_Seconds")

same_nomc <- ggplot() +
  geom_count(data=p_sameOG, aes(x=Runs, y = Time_In_Seconds, color = 'RATE_OG'), alpha=0.5) +
  geom_count(data=p_samequad, aes(x=Runs, y = Time_In_Seconds, color = 'RATE_quad'), alpha=0.5) +
  geom_count(data=p_samequad2, aes(x=Runs, y = Time_In_Seconds, color='RATE_quad2'), alpha=0.5) +
  geom_count(data=p_samecombo, aes(x=Runs, y = Time_In_Seconds, color = 'RATE_combo'), alpha=0.5) +
  scale_x_discrete(limits=time_names) +
  labs(color="Legend")+
  theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5), legend.position="right") +
  ggtitle("RATE same Times without RATE_MC (all runs)")

same_mc <- same_nomc +
  geom_count(data=p_sameMC, aes(x=Runs, y = Time_In_Seconds, color = 'RATE_MC'), alpha=0.5) +
  ggtitle("RATE same Times (all runs)")

#Diff Rates

p_OG <- Diff_OG_df %>% select(time_names) %>%
  pivot_longer(., cols = time_names, names_to = "Runs", values_to = "Time_In_Seconds")

p_quad <- Diff_quad_df %>% select(time_names) %>%
  pivot_longer(., cols = time_names, names_to = "Runs", values_to = "Time_In_Seconds")

p_quad2 <- Diff_quad2_df %>% select(time_names) %>%
  pivot_longer(., cols = time_names, names_to = "Runs", values_to = "Time_In_Seconds")

p_combo <- Diff_combo_df %>% select(time_names) %>%
  pivot_longer(., cols = time_names, names_to = "Runs", values_to = "Time_In_Seconds")

p_MC <- Diff_mc_df %>% select(time_names) %>%
  pivot_longer(., cols = time_names, names_to = "Runs", values_to = "Time_In_Seconds")

diff_nomc <- ggplot() +
  geom_count(data=p_OG, aes(x=Runs, y = Time_In_Seconds, color = 'RATE_OG'), alpha=0.5) +
  geom_count(data=p_quad, aes(x=Runs, y = Time_In_Seconds, color = 'RATE_quad'), alpha=0.5) +
  geom_count(data=p_quad2, aes(x=Runs, y = Time_In_Seconds, color='RATE_quad2'), alpha=0.5) +
  geom_count(data=p_combo, aes(x=Runs, y = Time_In_Seconds, color = 'RATE_combo'), alpha=0.5) +
  scale_x_discrete(limits=time_names) +
  labs(color="Legend")+
  theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5), legend.position="right") +
  ggtitle("RATE diff Times without RATE_MC (all runs)")

diff_mc <- diff_nomc +
  geom_count(data=p_MC, aes(x=Runs, y = Time_In_Seconds, color = 'RATE_MC'), alpha=0.5) +
  ggtitle("RATE diff Times (all runs)")


#Overlap
p_OG <- Overlap_OG_df %>% select(time_names) %>%
  pivot_longer(., cols = time_names, names_to = "Runs", values_to = "Time_In_Seconds")

p_quad <- Overlap_quad_df %>% select(time_names) %>%
  pivot_longer(., cols = time_names, names_to = "Runs", values_to = "Time_In_Seconds")

p_quad2 <- Overlap_quad2_df %>% select(time_names) %>%
  pivot_longer(., cols = time_names, names_to = "Runs", values_to = "Time_In_Seconds")

p_combo <- Overlap_combo_df %>% select(time_names) %>%
  pivot_longer(., cols = time_names, names_to = "Runs", values_to = "Time_In_Seconds")

p_MC <- Overlap_mc_df %>% select(time_names) %>%
  pivot_longer(., cols = time_names, names_to = "Runs", values_to = "Time_In_Seconds")

overlap_nomc <- ggplot() +
  geom_count(data=p_OG, aes(x=Runs, y = Time_In_Seconds, color = 'RATE_OG'), alpha=0.5) +
  geom_count(data=p_quad, aes(x=Runs, y = Time_In_Seconds, color = 'RATE_quad'), alpha=0.5) +
  geom_count(data=p_quad2, aes(x=Runs, y = Time_In_Seconds, color='RATE_quad2'), alpha=0.5) +
  geom_count(data=p_combo, aes(x=Runs, y = Time_In_Seconds, color = 'RATE_combo'), alpha=0.5) +
  scale_x_discrete(limits=time_names) +
  labs(color="Legend")+
  theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5), legend.position="right") +
  ggtitle("RATE overlap Times without RATE_MC (all runs)")

overlap_mc <- overlap_nomc +
  geom_count(data=p_MC, aes(x=Runs, y = Time_In_Seconds, color = 'RATE_MC'), alpha=0.5) +
  ggtitle("RATE overlap Times (all runs)")

grid.arrange(same_nomc, same_mc, diff_nomc, diff_mc, overlap_nomc, overlap_mc, nrow=3)
```

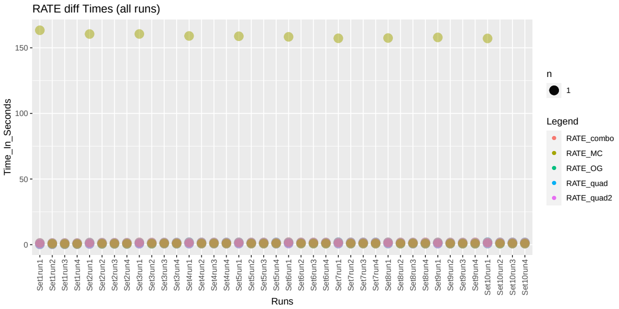
```
## Don’t know how to automatically pick scale for object of type difftime. Defaulting to continuous.
```

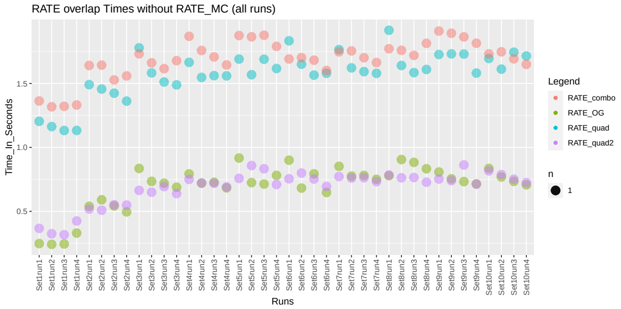
```
## Don’t know how to automatically pick scale for object of type difftime. Defaulting to continuous.
## Don’t know how to automatically pick scale for object of type difftime. Defaulting to continuous.
## Don’t know how to automatically pick scale for object of type difftime. Defaulting to continuous.
## Don’t know how to automatically pick scale for object of type difftime. Defaulting to continuous.
## Don’t know how to automatically pick scale for object of type difftime. Defaulting to continuous.
```

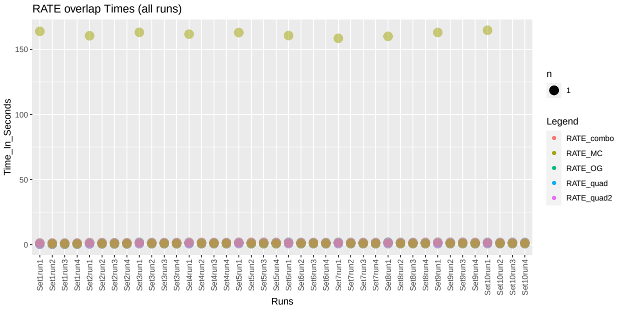












As you can see, regardless of the manner in which the data was simulated (which makes sense), the 'RATE_OG' and 'RATE_quad2' (which is the quad function that just calculates everything directly and does not take out the linear effect) work the fastest. 'RATE_combo' and 'RATE_quad' work about twice as slower. One the other hand, 'RATE_MC' takes a very long time to do that first calculation, because that's when we have to resample a function 'g.rep' to match the size of 'f.rep' for every SNP that we have. Once that part is done, the rest of the RATE calculations are just as long as 'RATE_OG'. It is worth noting that I could not figure out how to parallelize this no matter how much I tried without losing data (even in Linux it was throwing errors). For the sake of getting the data, I took it out. But this still means that, at best, we would have a time of around 190/32 ≈ 5.94 seconds. So still by far the longest one.

Statistical Analyses

Okay now we are going to run actual numbers. There are three scenarios and 5 rate functions. Gonna get the code first then will be easier to change stuff around.

Hopefully this sheds light on some things