Shape Discrimination Analysis

Ari Kahn

Imports

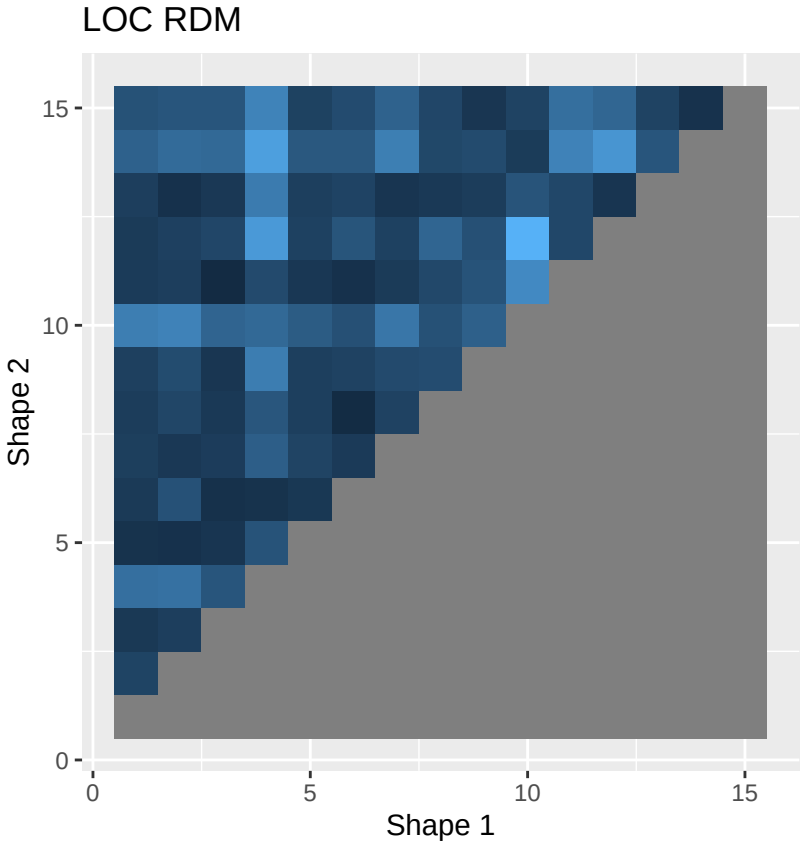
Load Data

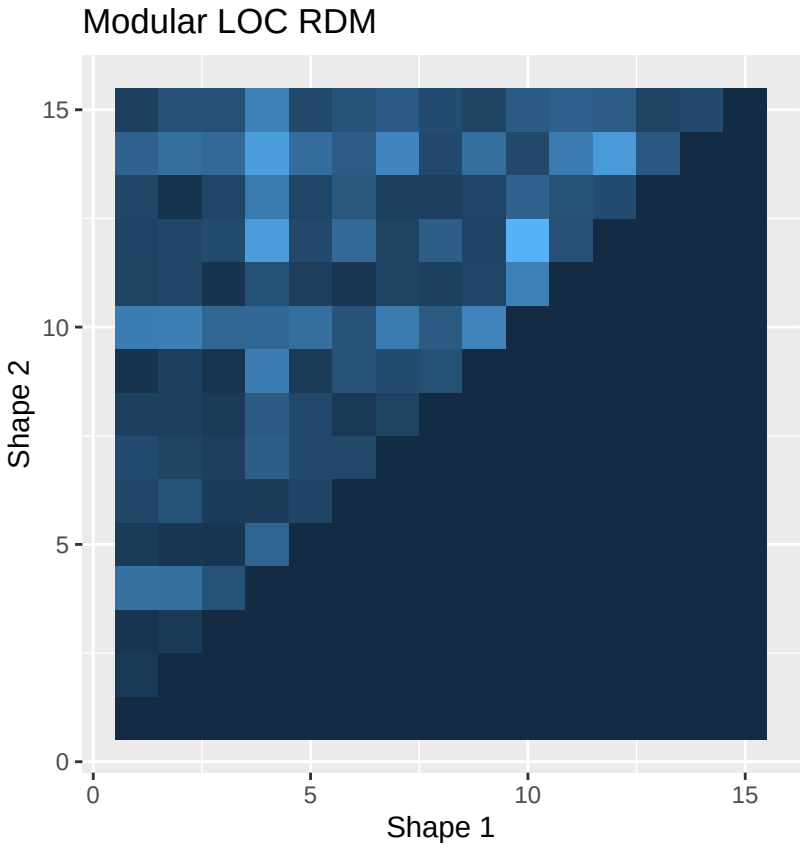
Data is all of the format `${subject}_Behavioral-2_Discrimination_${date}.csv`.

```
## [1] "Loaded 50 subjects"
```

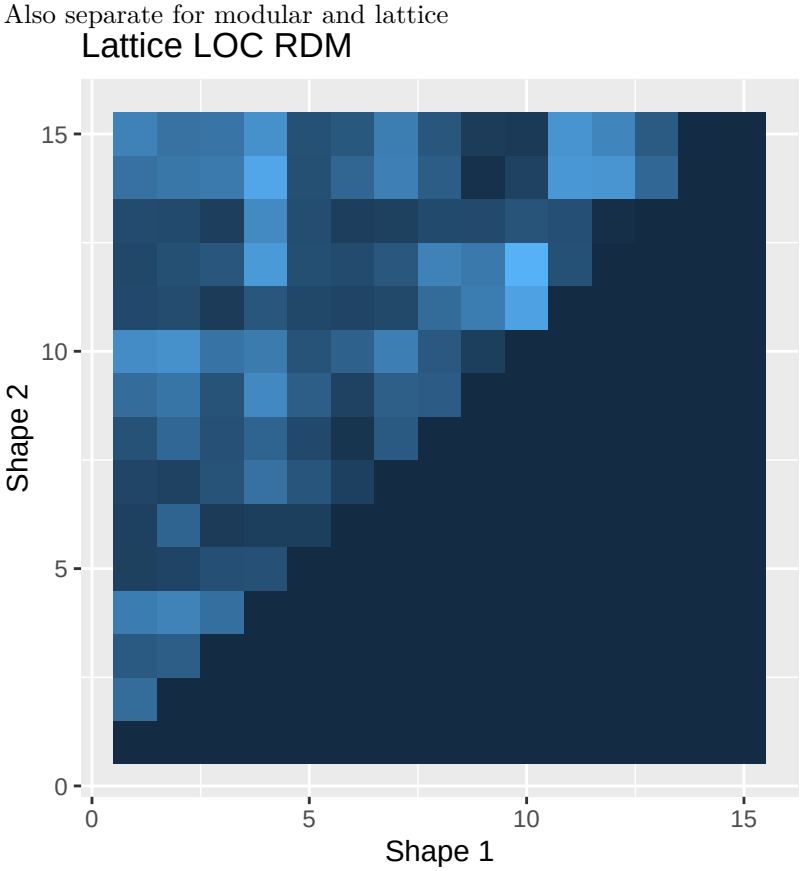
Read existing LOC triu data. This is the shape-by-shape dissimilarity metric, across both modular and lattice subjects.

```
## Warning: The `x` argument of `as_tibble.matrix()` must have unique column names if
## `.name_repair` is omitted as of tibble 2.0.0.
## i Using compatibility `.name_repair`.
```





Also separate for modular and lattice



And bind the LOC data to the result of the results.

Final dataframe:

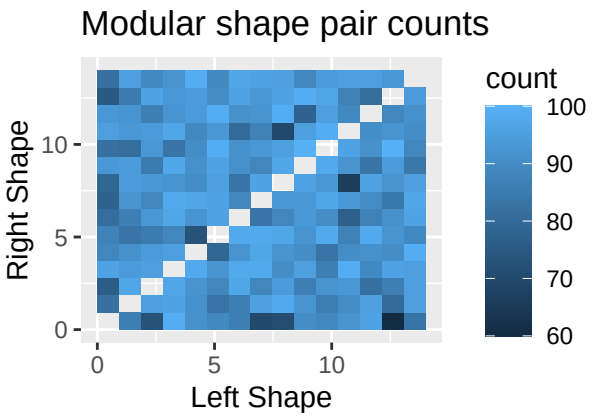
```
## # A tibble: 6 x 44
##   node_left node_~1 shape~2 shape~3 varia~4 varia~5 jitter trial discr~6 discr~7
##       <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>  <dbl> <dbl>   <dbl>   <dbl>
## 1         3      12       1       7       0       3  -0.15     3       0       2
## 2        11       9       9       8       3       0  -0.25     4       0       3
## 3         2      14       6       3       4       1  -0.15     6       0       5
## 4         8       2       5       6       4       1   0        7       0       6
## 5        10      12      14       7       1       4   0.1     11       0      10
## 6        10       2      14       6       3       0   0.2     15       0      14
## # ... with 34 more variables: discrimination.thisN <dbl>,
## #   discrimination.thisIndex <dbl>, experiment.time <dbl>, block.type <chr>,
## #   block.name <chr>, block.timings.start <dbl>, trial.correct <lgl>,
## #   trial.response <chr>, trial.timeout <lgl>, trial.timings.stimulus <dbl>,
## #   trial.timings.fixation_one <dbl>, trial.timings.fixation_two <dbl>,
## #   trial.timings.event <dbl>, trial.timings.rt <dbl>, Experiment <chr>,
## #   subject <fct>, graph <fct>, counterbalance <fct>, subset <fct>, ...
```

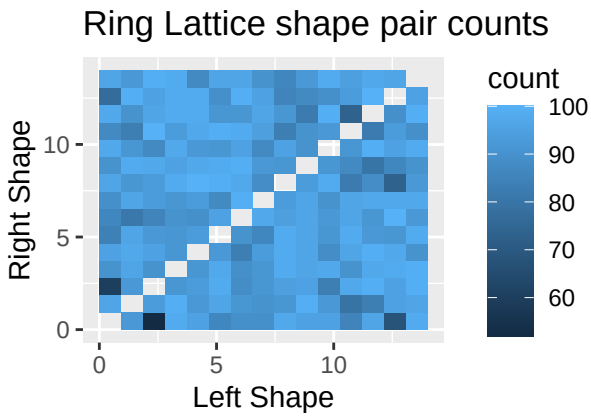
```
## # A tibble: 6 x 44
##   node_left node_~1 shape~2 shape~3 varia~4 varia~5 jitter trial discr~6 discr~7
##       <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>  <dbl> <dbl>   <dbl>   <dbl>
## 1         0       0      12      12       4       1   0.05     1       0       0
## 2        12      12       7       7       3       0  -0.05     2       0       1
## 3         3      12       1       7       0       3  -0.15     3       0       2
## 4        11       9       9       8       3       0  -0.25     4       0       3
## 5         0       0      12      12       3       0   0.15     5       0       4
## 6         2      14       6       3       4       1  -0.15     6       0       5
## # ... with 34 more variables: discrimination.thisN <dbl>,
## #   discrimination.thisIndex <dbl>, experiment.time <dbl>, block.type <chr>,
## #   block.name <chr>, block.timings.start <dbl>, trial.correct <lgl>,
## #   trial.response <chr>, trial.timeout <lgl>, trial.timings.stimulus <dbl>,
## #   trial.timings.fixation_one <dbl>, trial.timings.fixation_two <dbl>,
## #   trial.timings.event <dbl>, trial.timings.rt <dbl>, Experiment <chr>,
## #   subject <fct>, graph <fct>, counterbalance <fct>, subset <fct>, ...
```

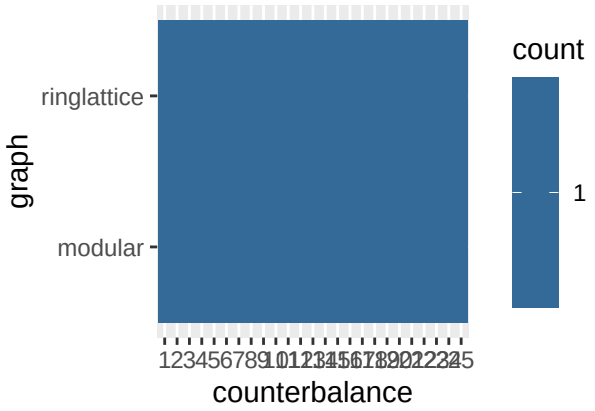
Data Verification

Are all shape pairs covered?

And make sure that each condition was covered once by condition







Results

RT and LOC Correlation

Is the average 1/RT correlated with the LOC dissimilarity across shape pairs?

First plot: - Each point is a shape pair - Plotting mean 1/R and LOC Dissimilarity values - We observe that more dissimilar shapes have lower reaction times (higher 1/RT)

```
cor.test(df.mean$rt.inv.mean, df.mean$z.loc_val.mean)
```

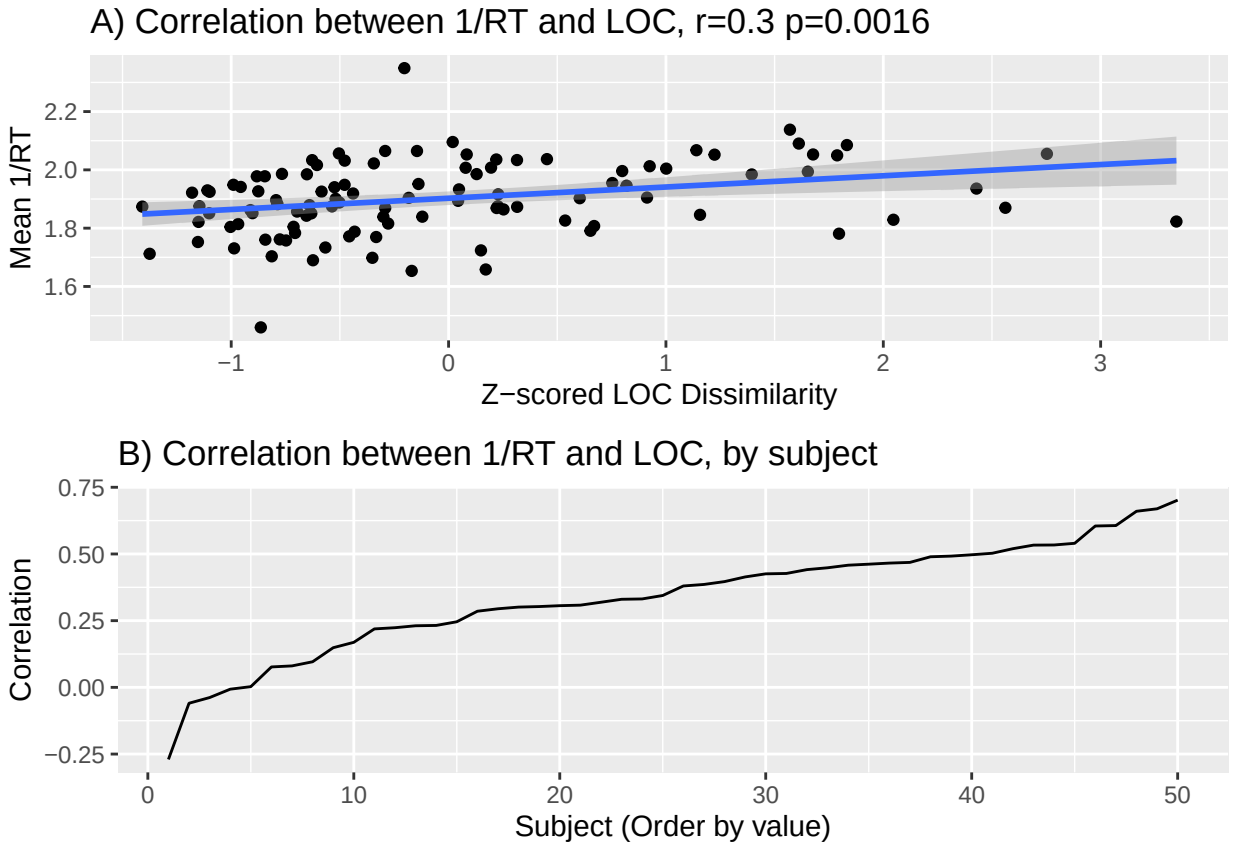
```
##
##  Pearson's product-moment correlation
##
## data:  df.mean$rt.inv.mean and df.mean$z.loc_val.mean
## t = 3.2476, df = 103, p-value = 0.001572
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.1201214 0.4690369
## sample estimates:
##       cor
## 0.3047699
```

If we compute the correlation on a per-subject basis, nearly all subjects display a positive effect, in most cases very strong.

We find a strong linear relationship between mean of 1/RT (across subjects), and LOC dissimilarity, across all shape pairs.

When this correlation is taken on a per-subject basis (for the behavioral data), against the canonical LOC RDM, almost all subjects show individually high correlations.

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
## Saving 6.5 x 4.5 in image
```

```
ggsave("images/rt_vs_loc.pdf", p1, height=5, width=5)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

Group LOC Correlation

We can also separate out the correlations by group.

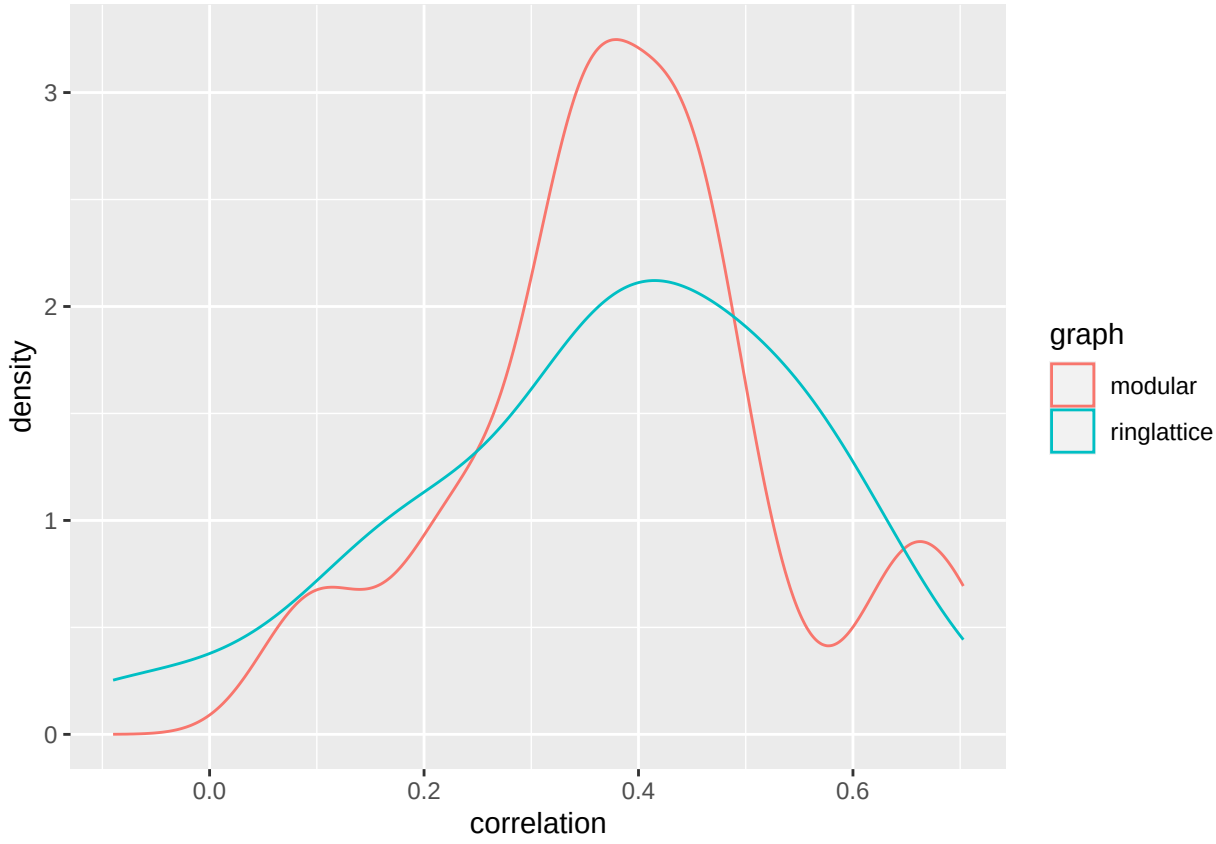
```
df %>%
  group_by(subject, shape_pair_ordered) %>%
  summarise(rt.inv.median = median(rt.inv), graph=first(graph), z.loc_val = first(z.loc_val), .groups =
  group_by(subject) %>%
  summarize(correlation = cor(rt.inv.median, z.loc_val), graph=first(graph), .groups = 'drop') %>%
  group_by(graph) %>%
  summarize(correlation.mean = mean(correlation))
```

```
## # A tibble: 2 x 2
##   graph       correlation.mean
##   <fct>                  <dbl>
## 1 modular                0.386
## 2 ringlattice            0.368
```

We find roughly similar distributions for correlations between subjects in the modular and lattice behavioral

groups

```
df %>%
  group_by(subject, shape_pair_ordered) %>%
  summarise(rt.inv.median = median(rt.inv), graph=first(graph), z.loc_val = first(z.loc_val), .groups =
  group_by(subject) %>%
  summarise(correlation = cor(rt.inv.median, z.loc_val), graph=first(graph), .groups = 'drop') %>%
  ggplot(aes(x=correlation, color=graph)) + geom_density()
```



Group Differences

```
df.median <- df %>%
  group_by(subject) %>%
  summarise(rt.inv.median = median(rt.inv), graph=first(graph), rt.inv.mean = mean(rt.inv))

df.median %>%
  group_by(graph) %>%
  summarize(rt.inv.median.mean = mean(rt.inv.median), rt.inv.median.sem = sd(rt.inv.median) / sqrt(leng
```

```
## # A tibble: 2 x 3
##   graph       rt.inv.median.mean rt.inv.median.sem
##   <fct>                    <dbl>             <dbl>
## 1 modular                   1.89            0.0583
## 2 ringlattice               1.94            0.0534
```

```
x <- df %>%
  group_by(graph) %>%
  summarise(rt.inv.median = median(rt.inv))
```

```
x
```

```
## # A tibble: 2 x 2
##   graph       rt.inv.median
##   <fct>               <dbl>
## 1 modular              1.89
## 2 ringlattice          1.97
```

```
rt.inv.median.modular <- (df.median %>% filter(graph == "modular"))$rt.inv.median
rt.inv.median.lattice <- (df.median %>% filter(graph == "ringlattice"))$rt.inv.median
t.test(rt.inv.median.modular, rt.inv.median.lattice)
```

```
##
##  Welch Two Sample t-test
##
## data:  rt.inv.median.modular and rt.inv.median.lattice
## t = -0.67042, df = 47.636, p-value = 0.5058
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.2120431  0.1060123
## sample estimates:
## mean of x mean of y
##  1.887699  1.940714
```

Basic Linear Model

First analysis: Purely linear model.

We're averaging across subjects for our behavioral RT, first.

Note that we can't include both `loc_val` and `shape_pair_ordered`, since here the effect for `shape_pair_ordered` already allows a separate fit for each shape, and we end up with collinearity. Since we don't care about `loc_val` here, don't think this is really a problem, though I've included both. `shape_pair_ordered` provides a slightly better fit, but the huge number of parameters result in a higher AIC, though not a 'sigificant' AIC difference (of <2)

We find graph type to be significant in either case, with decreased reaction times (higher 1/RT) for the ring lattice.

```
df.mean <- df %>%
  group_by(shape_pair_ordered, graph) %>%
  summarize(rt.inv.mean = mean(rt.inv), z.loc_val.mean = mean(z.loc_val))
```

```
## `summarise()` has grouped output by 'shape_pair_ordered'. You can override
## using the `.groups` argument.
```

```
m1 <- lm(rt.inv.mean ~ graph + z.loc_val.mean, df.mean)
summary(m1)
```

```
##
## Call:
## lm(formula = rt.inv.mean ~ graph + z.loc_val.mean, data = df.mean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.51199 -0.07704 -0.00119  0.10080  0.84129
##
## Coefficients:
```

```
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       1.86630    0.01475 126.553  < 2e-16 ***
## graphringlattice  0.07419    0.02085   3.558 0.000464 ***
## z.loc_val.mean    0.03835    0.01049   3.655 0.000326 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1511 on 207 degrees of freedom
## Multiple R-squared:  0.1116, Adjusted R-squared:  0.1031
## F-statistic: 13.01 on 2 and 207 DF,  p-value: 4.771e-06
```

```
AIC(m1)
```

```
## [1] -192.768
```

```
df.mean <- df %>%
  group_by(shape_pair_ordered, graph) %>%
  summarize(rt.inv.mean = mean(rt.inv), z.loc_val.mean = mean(z.loc_val))
```

```
## `summarise()` has grouped output by 'shape_pair_ordered'. You can override
## using the `.groups` argument.
```

```
m2 <- lm(rt.inv.mean ~ graph + shape_pair_ordered, df.mean)
summary(m2)
```

```
##
## Call:
## lm(formula = rt.inv.mean ~ graph + shape_pair_ordered, data = df.mean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36525 -0.05922  0.00000  0.05922  0.36525
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)               1.8780181  0.0929362  20.208  < 2e-16 ***
## graphringlattice          0.0741937  0.0180535   4.110 7.92e-05 ***
## shape_pair_ordered0_10    0.0702896  0.1308101   0.537  0.59218
## shape_pair_ordered0_11   -0.1535583  0.1308101  -1.174  0.24312
## shape_pair_ordered0_12    0.0700183  0.1308101   0.535  0.59361
## shape_pair_ordered0_13   -0.1209685  0.1308101  -0.925  0.35723
## shape_pair_ordered0_14    0.1272187  0.1308101   0.973  0.33304
## shape_pair_ordered0_2    -0.4326641  0.1308101  -3.308  0.00129 **
## shape_pair_ordered0_3    -0.0666757  0.1308101  -0.510  0.61133
## shape_pair_ordered0_4     0.0119040  0.1308101   0.091  0.92767
## shape_pair_ordered0_5    -0.2106864  0.1308101  -1.611  0.11029
## shape_pair_ordered0_6    -0.0645646  0.1308101  -0.494  0.62265
## shape_pair_ordered0_7    -0.0666440  0.1308101  -0.509  0.61150
## shape_pair_ordered0_8    -0.0001883  0.1308101  -0.001  0.99885
## shape_pair_ordered0_9     0.0790094  0.1308101   0.604  0.54716
## shape_pair_ordered1_10   -0.0357618  0.1308101  -0.273  0.78510
## shape_pair_ordered1_11   -0.1811704  0.1308101  -1.385  0.16902
## shape_pair_ordered1_12   -0.0941281  0.1308101  -0.720  0.47340
## shape_pair_ordered1_13    0.0848057  0.1308101   0.648  0.51821
## shape_pair_ordered1_14   -0.0402385  0.1308101  -0.308  0.75899
## shape_pair_ordered1_2    -0.0723407  0.1308101  -0.553  0.58144
```

```
## shape_pair_ordered1_3    0.1367738  0.1308101   1.046  0.29817
## shape_pair_ordered1_4   -0.0387931  0.1308101  -0.297  0.76739
## shape_pair_ordered1_5    0.0859303  0.1308101   0.657  0.51269
## shape_pair_ordered1_6    0.0114275  0.1308101   0.087  0.93055
## shape_pair_ordered1_7   -0.2163550  0.1308101  -1.654  0.10115
## shape_pair_ordered1_8    0.0453769  0.1308101   0.347  0.72937
## shape_pair_ordered1_9    0.1349791  0.1308101   1.032  0.30453
## shape_pair_ordered10_11 -0.0762775  0.1308101  -0.583  0.56108
## shape_pair_ordered10_12  0.1498584  0.1308101   1.146  0.25458
## shape_pair_ordered10_13 -0.1292234  0.1308101  -0.988  0.32551
## shape_pair_ordered10_14  0.1490714  0.1308101   1.140  0.25707
## shape_pair_ordered11_12 -0.1725568  0.1308101  -1.319  0.19002
## shape_pair_ordered11_13  0.0215212  0.1308101   0.165  0.86964
## shape_pair_ordered11_14  0.0305845  0.1308101   0.234  0.81559
## shape_pair_ordered12_13 -0.0040090  0.1308101  -0.031  0.97561
## shape_pair_ordered12_14  0.1159936  0.1308101   0.887  0.37727
## shape_pair_ordered13_14  0.0155912  0.1308101   0.119  0.90536
## shape_pair_ordered2_10  -0.0363804  0.1308101  -0.278  0.78148
## shape_pair_ordered2_11   0.1071230  0.1308101   0.819  0.41470
## shape_pair_ordered2_12   0.0549125  0.1308101   0.420  0.67551
## shape_pair_ordered2_13   0.0980395  0.1308101   0.749  0.45526
## shape_pair_ordered2_14   0.1161485  0.1308101   0.888  0.37663
## shape_pair_ordered2_3   -0.0459731  0.1308101  -0.351  0.72596
## shape_pair_ordered2_4   -0.1109154  0.1308101  -0.848  0.39843
## shape_pair_ordered2_5    0.0052144  0.1308101   0.040  0.96828
## shape_pair_ordered2_6   -0.1084834  0.1308101  -0.829  0.40882
## shape_pair_ordered2_7   -0.1536814  0.1308101  -1.175  0.24274
## shape_pair_ordered2_8    0.0263881  0.1308101   0.202  0.84052
## shape_pair_ordered2_9    0.0354589  0.1308101   0.271  0.78687
## shape_pair_ordered3_10   0.4565251  0.1308101   3.490  0.00071 ***
## shape_pair_ordered3_11  -0.0395316  0.1308101  -0.302  0.76310
## shape_pair_ordered3_12   0.2206175  0.1308101   1.687  0.09469 .
## shape_pair_ordered3_13   0.1410654  0.1308101   1.078  0.28335
## shape_pair_ordered3_14   0.1714035  0.1308101   1.310  0.19297
## shape_pair_ordered3_4   -0.1871759  0.1308101  -1.431  0.15546
## shape_pair_ordered3_5   -0.0633496  0.1308101  -0.484  0.62920
## shape_pair_ordered3_6   -0.0825370  0.1308101  -0.631  0.52945
## shape_pair_ordered3_7   -0.0509304  0.1308101  -0.389  0.69782
## shape_pair_ordered3_8    0.1745343  0.1308101   1.334  0.18503
## shape_pair_ordered3_9   -0.0100859  0.1308101  -0.077  0.93869
## shape_pair_ordered4_10  -0.0531587  0.1308101  -0.406  0.68530
## shape_pair_ordered4_11   0.0278755  0.1308101   0.213  0.83167
## shape_pair_ordered4_12   0.1197721  0.1308101   0.916  0.36199
## shape_pair_ordered4_13   0.1163233  0.1308101   0.889  0.37592
## shape_pair_ordered4_14  -0.0156680  0.1308101  -0.120  0.90489
## shape_pair_ordered4_5   -0.0706438  0.1308101  -0.540  0.59032
## shape_pair_ordered4_6   -0.1266822  0.1308101  -0.968  0.33507
## shape_pair_ordered4_7   -0.2222708  0.1308101  -1.699  0.09227 .
## shape_pair_ordered4_8    0.0984548  0.1308101   0.753  0.45336
## shape_pair_ordered4_9    0.1183908  0.1308101   0.905  0.36753
## shape_pair_ordered5_10  -0.1565470  0.1308101  -1.197  0.23413
## shape_pair_ordered5_11   0.0955970  0.1308101   0.731  0.46654
## shape_pair_ordered5_12  -0.1379655  0.1308101  -1.055  0.29401
## shape_pair_ordered5_13  -0.0419871  0.1308101  -0.321  0.74887
```

```
## shape_pair_ordered5_14   0.1509095  0.1308101   1.154  0.25129
## shape_pair_ordered5_6   -0.0111558  0.1308101  -0.085  0.93220
## shape_pair_ordered5_7   -0.1965797  0.1308101  -1.503  0.13592
## shape_pair_ordered5_8    0.1413327  0.1308101   1.080  0.28244
## shape_pair_ordered5_9    0.1807396  0.1308101   1.382  0.17003
## shape_pair_ordered6_10  -0.0317031  0.1308101  -0.242  0.80898
## shape_pair_ordered6_11  -0.0458448  0.1308101  -0.350  0.72670
## shape_pair_ordered6_12   0.0337831  0.1308101   0.258  0.79672
## shape_pair_ordered6_13   0.1380786  0.1308101   1.056  0.29361
## shape_pair_ordered6_14  -0.1060428  0.1308101  -0.811  0.41941
## shape_pair_ordered6_7   -0.0354671  0.1308101  -0.271  0.78683
## shape_pair_ordered6_8   -0.0155735  0.1308101  -0.119  0.90546
## shape_pair_ordered6_9    0.0702540  0.1308101   0.537  0.59237
## shape_pair_ordered7_10  -0.0990872  0.1308101  -0.757  0.45047
## shape_pair_ordered7_11   0.0808803  0.1308101   0.618  0.53773
## shape_pair_ordered7_12   0.0656543  0.1308101   0.502  0.61680
## shape_pair_ordered7_13  -0.0431268  0.1308101  -0.330  0.74230
## shape_pair_ordered7_14  -0.1433409  0.1308101  -1.096  0.27570
## shape_pair_ordered7_8   -0.0746202  0.1308101  -0.570  0.56961
## shape_pair_ordered7_9    0.0190746  0.1308101   0.146  0.88435
## shape_pair_ordered8_10   0.0711929  0.1308101   0.544  0.58744
## shape_pair_ordered8_11  -0.0335715  0.1308101  -0.257  0.79796
## shape_pair_ordered8_12  -0.1251884  0.1308101  -0.957  0.34077
## shape_pair_ordered8_13  -0.2517150  0.1308101  -1.924  0.05705 .
## shape_pair_ordered8_14  -0.1005920  0.1308101  -0.769  0.44364
## shape_pair_ordered8_9   -0.0065262  0.1308101  -0.050  0.96031
## shape_pair_ordered9_10  -0.0804191  0.1308101  -0.615  0.54004
## shape_pair_ordered9_11  -0.0927509  0.1308101  -0.709  0.47988
## shape_pair_ordered9_12  -0.2570267  0.1308101  -1.965  0.05210 .
## shape_pair_ordered9_13  -0.1505379  0.1308101  -1.151  0.25245
## shape_pair_ordered9_14   0.0336627  0.1308101   0.257  0.79742
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1308 on 104 degrees of freedom
## Multiple R-squared:  0.6655, Adjusted R-squared:  0.3278
## F-statistic: 1.971 on 105 and 104 DF,  p-value: 0.0003073
```

```
AIC(m2)
```

```
## [1] -191.8998
```

Mixed Effects Model

Here we're modeling 1/RT to be predicted by `graph`, LOC values, and random effects for each subject, and each shape pair.

I *think* this is the correct model to be using, but would appreciate a critical eye.

However, it doesn't appear that graph effects are significant here, with presumably a richer dataset allowing us to fit all repetitions of a shape pair per subject.

```
m3 <- lmer(rt.inv ~ graph + z.loc_val + (1 | subject) + (1 | shape_pair_ordered), df)
summary(m3)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
```

```
## Formula: rt.inv ~ graph + z.loc_val + (1 | subject) + (1 | shape_pair_ordered)
##    Data: df
##
## REML criterion at convergence: 111249.3
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
##  -1.721  -0.218   0.016   0.214 137.759
##
## Random effects:
##  Groups             Name        Variance Std.Dev.
##  shape_pair_ordered (Intercept) 0.003454 0.05877
##  subject            (Intercept) 0.071356 0.26713
##  Residual                       1.050584 1.02498
## Number of obs: 38431, groups:  shape_pair_ordered, 105; subject, 50
##
## Fixed effects:
##                   Estimate Std. Error        df t value Pr(>|t|)
## (Intercept)       1.871355   0.054249 48.808521  34.496  < 2e-16 ***
## graphringlattice  0.066073   0.076280 47.704012   0.866    0.391
## z.loc_val         0.033908   0.007827 93.178768   4.332 3.72e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) grphrn
## grphrnglttc -0.703
## z.loc_val    0.001  0.000
```

```
m4 <- lmer(rt.inv ~ graph * z.loc_val + (1 | subject) + (1 | shape_pair_ordered), df)
summary(m4)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: rt.inv ~ graph * z.loc_val + (1 | subject) + (1 | shape_pair_ordered)
##    Data: df
##
## REML criterion at convergence: 111256.5
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
##  -1.722  -0.217   0.016   0.214 137.758
##
## Random effects:
##  Groups             Name        Variance Std.Dev.
##  shape_pair_ordered (Intercept) 0.003455 0.05878
##  subject            (Intercept) 0.071358 0.26713
##  Residual                       1.050610 1.02499
## Number of obs: 38431, groups:  shape_pair_ordered, 105; subject, 50
##
## Fixed effects:
##                              Estimate Std. Error         df t value Pr(>|t|)
## (Intercept)                 1.871e+00  5.425e-02  4.881e+01  34.495  < 2e-16
## graphringlattice            6.607e-02  7.628e-02  4.770e+01   0.866 0.390714
## z.loc_val                   3.526e-02  9.434e-03  1.965e+02   3.738 0.000243
```

```
## graphringlattice:z.loc_val -2.704e-03  1.052e-02  3.833e+04  -0.257 0.797102
##
## (Intercept)                ***
## graphringlattice
## z.loc_val                  ***
## graphringlattice:z.loc_val
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) grphrn z.lc_v
## grphrngl:ttc -0.703
## z.loc_val    0.001  0.000
## grphrngl:._  0.000  0.000 -0.558
```

Figure 5 Exploration

Inter-pattern Distance versus Intra-pattern distance

```
df.full %>%
  filter(match) %>%
  filter(trial.correct) %>%
  group_by(subject, graph) %>%
  summarize(mean.rt.inv = mean(rt.inv)) %>%
  group_by(graph) %>%
  summarize(mean.rt.inv = mean(mean.rt.inv))
```

```
## `summarise()` has grouped output by 'subject'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 2 x 2
##   graph       mean.rt.inv
##   <fct>             <dbl>
## 1 modular            1.98
## 2 ringlattice        2.05
```

```
df.full %>%
  filter(match) %>%
  group_by(graph) %>%
  summarize(mean.correct= mean(trial.correct))
```

```
## # A tibble: 2 x 2
##   graph       mean.correct
##   <fct>              <dbl>
## 1 modular            0.898
## 2 ringlattice        0.911
```

```
df.full %>%
  filter(!match) %>%
  filter(trial.correct) %>%
  group_by(graph) %>%
  summarize(mean.rt.inv = mean(rt.inv))
```

```
## # A tibble: 2 x 2
##   graph       mean.rt.inv
##   <fct>             <dbl>
```

```
## 1 modular            1.86
## 2 ringlattice        1.94
```

```
m5 <- lmer(rt.inv ~ graph*z.modular_loc_val + graph:z.lattice_loc_val + (1 | subject) + (1 | shape_pair
summary(m5)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: rt.inv ~ graph * z.modular_loc_val + graph:z.lattice_loc_val +
##     (1 | subject) + (1 | shape_pair_ordered)
##    Data: df
##
## REML criterion at convergence: 111269.7
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
##  -1.732  -0.217   0.016   0.214 137.753
##
## Random effects:
##  Groups              Name        Variance Std.Dev.
##  shape_pair_ordered (Intercept) 0.003521 0.05934
##  subject            (Intercept) 0.071440 0.26728
##  Residual                       1.050617 1.02500
## Number of obs: 38431, groups:  shape_pair_ordered, 105; subject, 50
##
## Fixed effects:
##                                     Estimate Std. Error         df t value
## (Intercept)                        1.871e+00  5.429e-02  4.882e+01  34.472
## graphringlattice                   6.605e-02  7.632e-02  4.769e+01   0.865
## z.modular_loc_val                  2.886e-02  1.307e-02  1.994e+02   2.208
## graphringlattice:z.modular_loc_val -1.186e-02  1.457e-02  3.839e+04  -0.814
## graphmodular:z.lattice_loc_val      9.398e-03  1.301e-02  1.980e+02   0.723
## graphringlattice:z.lattice_loc_val  1.873e-02  1.300e-02  1.976e+02   1.441
##                                    Pr(>|t|)
## (Intercept)                          <2e-16 ***
## graphringlattice                     0.3911
## z.modular_loc_val                    0.0284 *
## graphringlattice:z.modular_loc_val   0.4158
## graphmodular:z.lattice_loc_val       0.4708
## graphringlattice:z.lattice_loc_val   0.1512
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##                   (Intr) grphrn z.md__ grphrnglttc:z.m__ grphm:.__
## grphrngltttc      -0.703
## z.mdlr_lc_v        0.001  0.000
## grphrngltttc:z.m__  0.000  0.000 -0.559
## grphmdl:.__         0.000  0.000 -0.684  0.383
## grphrngltttc:z.l__  0.000  0.000 -0.257 -0.382             0.378
```

```
df.full.match <-
  df.full %>%
  filter(match) %>%
  filter(trial.correct)
```

```
m6 <- lmer(rt.inv ~ graph + (1 | subject) + (1 | shape_pair_ordered), df.full.match)
summary(m6)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: rt.inv ~ graph + (1 | subject) + (1 | shape_pair_ordered)
##    Data: df.full.match
##
## REML criterion at convergence: 103607.1
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
##  -2.048  -0.254   0.004   0.250 127.795
##
## Random effects:
##  Groups             Name        Variance Std.Dev.
##  subject            (Intercept) 0.067288 0.25940
##  shape_pair_ordered (Intercept) 0.003222 0.05676
##  Residual                       0.888734 0.94273
## Number of obs: 38003, groups:  subject, 50; shape_pair_ordered, 15
##
## Fixed effects:
##                  Estimate Std. Error       df t value Pr(>|t|)
## (Intercept)       1.97500    0.05435 54.40298  36.339   <2e-16 ***
## graphringlattice  0.07689    0.07401 47.97186   1.039    0.304
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr)
## grphrngltc -0.681
```

```
df.full.correct <-
  df.full %>%
  filter(trial.correct)
m7 <- lmer(rt.inv ~ graph*match + (1 | subject) + (1 | shape_pair_ordered), df.full.correct)
summary(m7)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: rt.inv ~ graph * match + (1 | subject) + (1 | shape_pair_ordered)
##    Data: df.full.correct
##
## REML criterion at convergence: 215161.8
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
##  -1.830  -0.238   0.008   0.232 143.280
##
## Random effects:
##  Groups             Name        Variance Std.Dev.
##  shape_pair_ordered (Intercept) 0.004328 0.06579
##  subject            (Intercept) 0.067302 0.25943
##  Residual                       0.972456 0.98613
```

```
## Number of obs: 76434, groups:  shape_pair_ordered, 120; subject, 50
##
## Fixed effects:
##                              Estimate Std. Error        df t value Pr(>|t|)
## (Intercept)                 1.872e+00  5.277e-02 5.030e+01  35.483  < 2e-16 ***
## graphringlattice            6.656e-02  7.407e-02 4.882e+01   0.899    0.373
## matchTRUE                   1.035e-01  2.080e-02 8.200e+01   4.976 3.52e-06 ***
## graphringlattice:matchTRUE 9.329e-03  1.428e-02 7.634e+04   0.653    0.514
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) grphrn mtTRUE
## grphrnglttc -0.702
## matchTRUE   -0.084  0.033
## grphrn:TRUE  0.068 -0.096 -0.346
```