Maia Analysis

Maia Czerwonka

2024-02-27

```r
# Load in .csv
data <- read.csv("MaiaData_CardSort.csv")
data
```

```
##     Subject Age Sex IncorRespCount SymRespCount TxtRespCount   BiasScore
## 1  0156ce12  51   2              0            9           39  0.62500000
## 2  054ba968  87   2              0           20           28  0.16666667
## 3  05546b7f  60   2              5           12           31  0.44186046
## 4  08f746fa  38   2              2           39            7 -0.69565217
## 5  0aef8687  63   2              8            8           32  0.60000000
## 6  0b2a2504  34   2              4           17           27  0.22727273
## 7  0b482cbe  36   2              1           44            3 -0.87234043
## 8  0bf6b839  54   2              0           45            3 -0.87500000
## 9  0ccd33c5  54   2              0           14           34  0.41666667
## 10 0cf98284  45   2              1           11           36  0.53191489
## 11 0ebb6dab  80   2              0            1           47  0.95833333
## 12 107f8d3e  29   2              0           45            3 -0.87500000
## 13 117d6806  22   2              3           41            4 -0.82222222
## 14 14165170  83   3              3            1           44  0.95555556
## 15 16f9e2af  51   2              0            1           47  0.95833333
## 16 17d323d9  46   2             15           17           16 -0.03030303
## 17 188a146f  34   2              3            3           42  0.86666667
## 18 1aa15bd9  25   2              1            0           47  1.00000000
## 19 1c4f7ffb  44   1              2           38            8 -0.65217391
## 20 1d937134  34   2              1           45            2 -0.91489362
## 21 1ea6c7bb  32   1              0            4           44  0.83333333
## 22 2078bec9  10   2              2           42            4 -0.82608696
## 23 214a186c  54   2              1           25           22 -0.06382979
## 24 2485948d  51   2              1           44            3 -0.87234043
## 25 24d99a95  39   2              1           40            7 -0.70212766
## 26 2589c6ec  36   2              2            1           45  0.95652174
## 27 2887f859  36   2              1           35           12 -0.48936170
## 28 28a198fb  15   2              0            0           48  1.00000000
## 29 2958c25f  36   2              0           18           30  0.25000000
## 30 297c34a5  44   2              2           41            5 -0.78260870
## 31 2a91a3cc  60   1              1           46            1 -0.95744681
## 32 2c32d3f4  62   1              1           17           30  0.27659575
## 33 2dc7132a  38   2              0           12           36  0.50000000
## 34 2e5e60b7  33   2              1            2           45  0.91489362
## 35 2ee97ea7  29   1              0           45            3 -0.87500000
## 36 30d40fe2  26   1              3           44            1 -0.95555556
## 37 32308591  35   2              0           34           14 -0.41666667
```

```
## 38  32813fd0  60   1              1           3          44  0.87234043
## 39  335d619c  25   2              2          20          26  0.13043478
## 40  3456c8ae  35   4              1           1          46  0.95744681
## 41  36345909  23   2              0           1          47  0.95833333
## 42  3661b157  40   1              1           2          45  0.91489362
## 43  3b118efc  42   2              1           0          47  1.00000000
## 44  3b79c957  37   2              2          38           8 -0.65217391
## 45  3e459806  40   2              0           0          48  1.00000000
## 46  3e83a772  30   2              3          44           1 -0.95555556
## 47  400eed91  21   2              0           3          45  0.87500000
## 48  42bd8614  50   1              1           4          43  0.82978723
## 49  431d20df  45   2              0          43           5 -0.79166667
## 50  450af0e8  27   2              0          37          11 -0.54166667
## 51  46db6023  22   1              3          41           4 -0.82222222
## 52  4845278d  41   1             16          17          15 -0.06250000
## 53  4db4c716  30   2              1           8          39  0.65957447
## 54  4ea7e33e  45   2              1          34          13 -0.44680851
## 55  513c6adf  27   1              1           2          45  0.91489362
## 56  53124fc7  31   2              0           6          42  0.75000000
## 57  53afc254  20   1              1          40           7 -0.70212766
## 58  549e2bf3  23   2              1           2          45  0.91489362
## 59  54d4291b  41   2              1          44           3 -0.87234043
## 60  5701e616  64   2             14          12          22  0.29411765
## 61  5816ece9  19   2              3          36           9 -0.60000000
## 62  5843f329  54   1              2           1          45  0.95652174
## 63  59348bca  47   2              2           3          43  0.86956522
## 64  599a551a  32   2              0           7          41  0.70833333
## 65  59d4e173  53   2              0           1          47  0.95833333
## 66  5c8d0b14  45   2              9          26          13 -0.33333333
## 67  5ee0a865  15   2              2          40           6 -0.73913043
## 68  5fc4fa1b  12   2              1          17          30  0.27659575
## 69  61f78fb4  42   2              3           1          44  0.95555556
## 70  62c71be3  25   2              2           9          37  0.60869565
## 71  6409a8b2  22   2             10          24          14 -0.26315789
## 72  65c75fc2  34   2              1           2          45  0.91489362
## 73  66cfde31  56   2              1           1          46  0.95744681
## 74  66eb7b6a  20   2              1           9          38  0.61702128
## 75  66fab246  22   2              1          45           2 -0.91489362
## 76  68a9d481  44   2              3           2          43  0.91111111
## 77  69863308  40   2              2          15          31  0.34782609
## 78  703265e2  43   1              3           0          45  1.00000000
## 79  714b2d11  37   2              0           1          47  0.95833333
## 80  73f3b4e7  39   2              1           3          44  0.87234043
## 81  7c320256  24   2              1           3          44  0.87234043
## 82  7c99f480  32   2              2          40           6 -0.73913043
## 83  7f2bf12c  NA   4              0           7          41  0.70833333
## 84  80884a14  33   2              3          21          24  0.06666667
## 85  82a2bd0c  42   2              2           4          42  0.82608696
## 86  86655f6c  29   2              3           4          41  0.82222222
## 87  88a049b1  46   2              1           1          46  0.95744681
## 88  88f5e293  49   2              2           1          45  0.95652174
## 89  89601069  22   2              0          23          25  0.04166667
## 90  8afceed7  47   2              1           1          46  0.95744681
## 91  8f5c3e5c  55   2              0          43           5 -0.79166667
```

```
## 92  901e335b  44   2                1              0              47  1.00000000
## 93  90caab28  32   1                1              0              47  1.00000000
## 94  90db8365  58   2                0              0              48  1.00000000
## 95  90fd8c3c  52   2               13             14              21  0.20000000
## 96  91468fb8  22   2                2             44               2 -0.91304348
## 97  9267045e  36   2                3             16              29  0.28888889
## 98  93eb1c7c  51   1                1             45               2 -0.91489362
## 99  93f8eaf0  NA  NA                2              1              45  0.95652174
## 100 9449a552  57   2               19             17              12 -0.17241379
## 101 96341682  31   1                1              0              47  1.00000000
## 102 98424747  32   2                1              6              41  0.74468085
## 103 99056ad5  66   2                1             20              27  0.14893617
## 104 9c2294e3  50   1                1             25              22 -0.06382979
## 105 9cabc628  62   2                0              9              39  0.62500000
## 106 9cae589f  37   1                2             42               4 -0.82608696
## 107 9d172f36  37   2                0             30              18 -0.25000000
## 108 9e16af1d  50   2                0              2              46  0.91666667
## 109 a20c2c7f  50   1                0              3              45  0.87500000
## 110 a316c167  41   1                1             40               7 -0.70212766
## 111 a4e92a6d  52   2                0              0              48  1.00000000
## 112 a76f91d1  33   2                1             40               7 -0.70212766
## 113 a8f7199a  34   2                1             39               8 -0.65957447
## 114 a9d90d7b  31   1                0             47               1 -0.95833333
## 115 ad20acda  43   1                0              2              46  0.91666667
## 116 ae6e621d  33   2                0              1              47  0.95833333
## 117 b0798a36  22   2                0             44               4 -0.83333333
## 118 b1466d2f  78   2                1              0              47  1.00000000
## 119 b34c5e2c  64   1                1              1              46  0.95744681
## 120 b54d7127  54   2                0              0              48  1.00000000
## 121 b5a17561  56   2                2             13              33  0.43478261
## 122 b68374cc  22   2                2              1              45  0.95652174
## 123 b6859fc4  23   2                3              4              41  0.82222222
## 124 b8f82017  39   2                0             47               1 -0.95833333
## 125 ba441428  34   2                0              2              46  0.91666667
## 126 bad0b9f0  34   2                0             30              18 -0.25000000
## 127 bad632c6  38   1                4             15              29  0.31818182
## 128 bb626e6d  28   2                1             27              20 -0.14893617
## 129 bc551487  28   1                2              2              44  0.91304348
## 130 bd06a398  44   2                2              0              46  1.00000000
## 131 bd5495a7  36   2                2              0              46  1.00000000
## 132 be60cc2c  30   2                0              1              47  0.95833333
## 133 beb01630  22   2                2             38               8 -0.65217391
## 134 c3224966  45   2                1              5              42  0.78723404
## 135 c5f1529a  45   2                2             38               8 -0.65217391
## 136 c6dec119  21   2                2             31              15 -0.34782609
## 137 c97ba508  35   1                0              0              48  1.00000000
## 138 cac5cbcc  38   2                1              1              46  0.95744681
## 139 cb8d96e2  63   2                0              2              46  0.91666667
## 140 ccabde00  38   2                1              3              44  0.87234043
## 141 cd55f156  30   1                0             45               3 -0.87500000
## 142 d0d04e2f  30   2                0              0              48  1.00000000
## 143 d208937c  74   2                0              1              47  0.95833333
## 144 d41bb601  43   2                1              1              46  0.95744681
## 145 d541e4c0  22   1                3             38               7 -0.68888889
```

```
## 146 d6dead27  22   2            0           45           3 -0.87500000
## 147 d792be63  65   2            0            0          48  1.00000000
## 148 d7f2ce5b  13   2            3           39           6 -0.73333333
## 149 d84d792b  58   2            2            2          44  0.91304348
## 150 d85d3220  25   2            0           46           2 -0.91666667
## 151 d9a70fac  63   2            3            2          43  0.91111111
## 152 da94862b  41   2            0           45           3 -0.87500000
## 153 dc8267e2  56   2            2           33          13 -0.43478261
## 154 dcee3b58  48   2            1           11          36  0.53191489
## 155 df52db19  21   2           14           21          13 -0.23529412
## 156 dfe0dfb3  40   1            2           43           3 -0.86956522
## 157 e0216ded  54   2            0            0          48  1.00000000
## 158 e1264a4f  26   2            0            2          46  0.91666667
## 159 e15dad59  46   1            2           19          27  0.17391304
## 160 e7ff23cc  39   2           13           15          20  0.14285714
## 161 e80f9781  44   2            0           15          33  0.37500000
## 162 e8b26ab1  47   1            0           13          35  0.45833333
## 163 e90ecf36  52   2            1            0          47  1.00000000
## 164 e928f889  35   1            0           47           1 -0.95833333
## 165 e9957f47  39   2            1            1          46  0.95744681
## 166 ea0fcf6f  51   2            0           46           2 -0.91666667
## 167 ea946a6b  30   2            0            2          46  0.91666667
## 168 ec01fb3e  35   1            3           45           0 -1.00000000
## 169 ec41c586  21   2            2            3          43  0.86956522
## 170 eca1bcfe  44   2            1            4          43  0.82978723
## 171 ecca85f8  38   1            2           43           3 -0.86956522
## 172 ed2aa13b  26   1            0            1          47  0.95833333
## 173 f0d24353  47   1            1            0          47  1.00000000
## 174 f1fb3ed7  48   1            1           21          26  0.10638298
## 175 f27c8cbe  58   1            0           48           0 -1.00000000
## 176 f33cd5ba  65   2            1            1          46  0.95744681
## 177 f6292664  31   2            0           42           6 -0.75000000
## 178 f85a3e1b  34   2            1           31          16 -0.31914894
## 179 f9af1669  42   2            0           45           3 -0.87500000
## 180 fa217a97  70   1            0            2          46  0.91666667
## 181 fade4881  59   2           13           16          19  0.08571429
## 182 fb705df2  18   2            3           37           8 -0.64444444
## 183 fdd4712a  41   2            3           32          13 -0.42222222
## 184 feabc45c  33   2            0           44           4 -0.83333333
## 185 ffdf4f28  30   2            1            0          47  1.00000000
##        Con_RT  InCon_RT  Incon.ConRT
## 1    878.3542 1014.1875  135.8333333
## 2   1192.7917 1273.8125   81.0208333
## 3   1122.9306 1194.4583   71.5277778
## 4   1013.8264 1029.7708   15.9444444
## 5    990.1875  988.1875   -2.0000000
## 6   1066.3611 1098.3750   32.0138889
## 7    829.4028  954.7917  125.3888889
## 8    647.6250  728.7500   81.1250000
## 9   1119.5208 1274.3750  154.8541667
## 10   847.7083  913.3542   65.6458333
## 11   860.7292  871.4375   10.7083333
## 12   621.0694  710.6042   89.5347222
## 13   579.7847  605.9583   26.1736111
```

```
## 14  1056.6111 1070.5000   13.8888889
## 15   702.0417  698.2292   -3.8125000
## 16   893.2222  872.2083  -21.0138889
## 17   744.6597  791.7083   47.0486111
## 18   705.4097  646.8542  -58.5555556
## 19   948.7778 1282.8750  334.0972222
## 20  1126.0000 1313.3750  187.3750000
## 21   732.3611  690.7708  -41.5902778
## 22   837.9792  849.5625   11.5833333
## 23   895.5972  927.4375   31.8402778
## 24  1088.8056 1229.0208  140.2152778
## 25  1040.5486 1117.3750   76.8263889
## 26   542.7569  537.4167   -5.3402778
## 27  1136.2639 1393.7083  257.4444444
## 28   738.5000  735.3333   -3.1666667
## 29   883.0625 1100.8333  217.7708333
## 30   977.7153 1139.2708  161.5555556
## 31  1106.5278 1143.9167   37.3888889
## 32  1091.7708 1185.0833   93.3125000
## 33  1068.6875 1257.8125  189.1250000
## 34   807.9236  817.2708    9.3472222
## 35   912.3333 1066.9167  154.5833333
## 36   829.2292  894.3125   65.0833333
## 37   794.5903  846.1875   51.5972222
## 38   904.7778  838.1667  -66.6111111
## 39   722.9028  732.1875    9.2847222
## 40   673.9722  665.7083   -8.2638889
## 41   623.0764  612.8125  -10.2638889
## 42   862.1389  853.4583   -8.6805556
## 43   765.3056  752.3958  -12.9097222
## 44   820.5556  884.9583   64.4027778
## 45   778.8819  800.9167   22.0347222
## 46   603.7778  629.3958   25.6180556
## 47   529.0764  528.2917   -0.7847222
## 48  1006.2431  944.1458  -62.0972222
## 49   741.0694  923.5625  182.4930556
## 50   745.6944  859.7500  114.0555556
## 51   742.4306  856.2083  113.7777778
## 52   937.1181 1015.7708   78.6527778
## 53   906.2292  939.9167   33.6875000
## 54   775.1319  920.0000  144.8680556
## 55   764.0694  831.8333   67.7638889
## 56  1019.2708 1119.8542  100.5833333
## 57   899.9583  897.1875   -2.7708333
## 58   658.1667  616.8542  -41.3125000
## 59  1036.2083 1138.9583  102.7500000
## 60  1309.0208 1297.0000  -12.0208333
## 61   691.3542  738.4792   47.1250000
## 62   823.1806  770.8750  -52.3055556
## 63   984.2778 1029.2500   44.9722222
## 64   972.3750  982.6042   10.2291667
## 65   884.4375  879.6458   -4.7916667
## 66  1089.6806 1430.5208  340.8402778
## 67   890.7153  996.9167  106.2013889
```

```
## 68  1023.7569 1119.7500   95.9930556
## 69   813.7917  799.5000  -14.2916667
## 70  1215.9306 1260.7917   44.8611111
## 71  1176.2986 1060.2083 -116.0902778
## 72   806.4722  825.4167   18.9444444
## 73   680.7014  737.3958   56.6944444
## 74   829.9583  952.0625  122.1041667
## 75   857.7014  848.7292   -8.9722222
## 76   672.4583  697.6042   25.1458333
## 77  1209.5208 1219.1667    9.6458333
## 78   893.2292  895.1458    1.9166667
## 79   995.5694 1071.9583   76.3888889
## 80   933.4167  982.5625   49.1458333
## 81   941.6528 1174.0833  232.4305556
## 82   738.0764  763.2917   25.2152778
## 83   771.2917  796.7708   25.4791667
## 84   740.1111  748.7917    8.6805556
## 85   991.9097 1180.7917  188.8819444
## 86  1004.3750  953.7083  -50.6666667
## 87  1043.6389 1130.7292   87.0902778
## 88   707.2153  686.2500  -20.9652778
## 89   730.3403  673.3542  -56.9861111
## 90  1133.5833 1066.0000  -67.5833333
## 91   817.0417  986.9792  169.9375000
## 92   580.1181  628.8750   48.7569444
## 93   815.8333  823.1458    7.3125000
## 94   963.2569  979.1875   15.9305556
## 95  1213.4722 2693.2708 1479.7986110
## 96   852.2708 1059.8125  207.5416667
## 97   942.7569 1351.2292  408.4722222
## 98   874.8264  968.2708   93.4444444
## 99   678.2708  795.8958  117.6250000
## 100 1579.7153 1494.5625  -85.1527778
## 101  899.0972  945.6667   46.5694444
## 102  714.3611  895.3750  181.0138889
## 103 1260.3750 1705.9792  445.6041667
## 104 1225.6597 1620.2292  394.5694444
## 105 1198.6806 1229.4792   30.7986111
## 106  758.1528  822.0000   63.8472222
## 107 1232.2014 1605.5208  373.3194444
## 108  942.5625  925.8125  -16.7500000
## 109  826.6319  800.1042  -26.5277778
## 110  901.7431 1024.5000  122.7569444
## 111  849.4931  807.2917  -42.2013889
## 112 1175.7917 1815.8542  640.0625000
## 113  998.9583 1133.0208  134.0625000
## 114  910.7778 1085.3125  174.5347222
## 115  934.1736  970.9583   36.7847222
## 116  780.2639  862.5833   82.3194444
## 117  886.9097  876.7917  -10.1180556
## 118 1204.0625 1090.8958 -113.1666667
## 119  895.0069  916.9167   21.9097222
## 120  823.1250  797.0625  -26.0625000
## 121  827.9028  975.0208  147.1180556
```

```
## 122   749.9861  767.9375   17.9513889
## 123   631.3472  673.6875   42.3402778
## 124   810.7639  871.3542   60.5902778
## 125   702.7083  764.2083   61.5000000
## 126   867.2222 1008.1458  140.9236111
## 127  1044.6875 1127.5833   82.8958333
## 128   829.2083 1059.2292  230.0208333
## 129   785.0486  846.5417   61.4930556
## 130   945.1875 1015.7083   70.5208333
## 131   671.3542  672.1667    0.8125000
## 132   637.9167  672.8542   34.9375000
## 133   648.0833  654.6042    6.5208333
## 134   794.2014  995.8125  201.6111111
## 135   943.7639 1133.8125  190.0486111
## 136   706.6389  839.5625  132.9236111
## 137   738.3819  767.6042   29.2222222
## 138   850.4167  854.6875    4.2708333
## 139   787.3750  802.5208   15.1458333
## 140   768.5139  914.3125  145.7986111
## 141   800.1458  803.9792    3.8333333
## 142   939.9792 1051.3125  111.3333333
## 143  1121.3472 1124.1667    2.8194444
## 144   875.6597  900.8125   25.1527778
## 145   832.9722  817.9375  -15.0347222
## 146   862.1389  895.6250   33.4861111
## 147   853.6806  868.6458   14.9652778
## 148   606.1667  624.2708   18.1041667
## 149   734.6458  737.2917    2.6458333
## 150   648.6944  687.0208   38.3263889
## 151   952.1458  978.7708   26.6250000
## 152   927.7569 1090.1042  162.3472222
## 153   920.1111 1251.1875  331.0763889
## 154   914.5833 1111.3750  196.7916667
## 155   866.4167  881.4792   15.0625000
## 156  1307.9375 1449.2083  141.2708333
## 157   615.0417  610.6042   -4.4375000
## 158  1235.2431 1160.2083  -75.0347222
## 159   859.8681  948.0625   88.1944444
## 160   999.5833 2790.1667 1790.5833330
## 161   815.6875  911.2917   95.6041667
## 162   702.1944  716.8542   14.6597222
## 163   864.0278  794.0625  -69.9652778
## 164   921.6806  931.1458    9.4652778
## 165   964.9653  850.6042 -114.3611111
## 166   853.8472 1119.3958  265.5486111
## 167   867.6944  933.9167   66.2222222
## 168   792.6528  884.8333   92.1805556
## 169   716.9306  768.6250   51.6944444
## 170   963.7153 1019.8958   56.1805556
## 171   784.5417 1048.6042  264.0625000
## 172   683.2986  658.4167  -24.8819444
## 173   619.9306  626.3958    6.4652778
## 174  1163.1181 1463.3750  300.2569444
## 175   888.2361  928.7708   40.5347222
```

```
## 176  902.5208  893.0833   -9.4375000
## 177  845.6250 1391.8750  546.2500000
## 178  939.3819 1038.5000   99.1180556
## 179  810.0833  888.2083   78.1250000
## 180  815.9514  822.8542    6.9027778
## 181  960.7986 2863.0833 1902.2847220
## 182 1192.3681 1345.3125  152.9444444
## 183  988.2431 1345.1042  356.8611111
## 184  935.5833 1082.7708  147.1875000
## 185 1024.2500 1055.7500   31.5000000
```

Descriptives of Dataset

```r
data %>%
  select(-c(Subject, Sex)) %>%
  psych::describe()
```

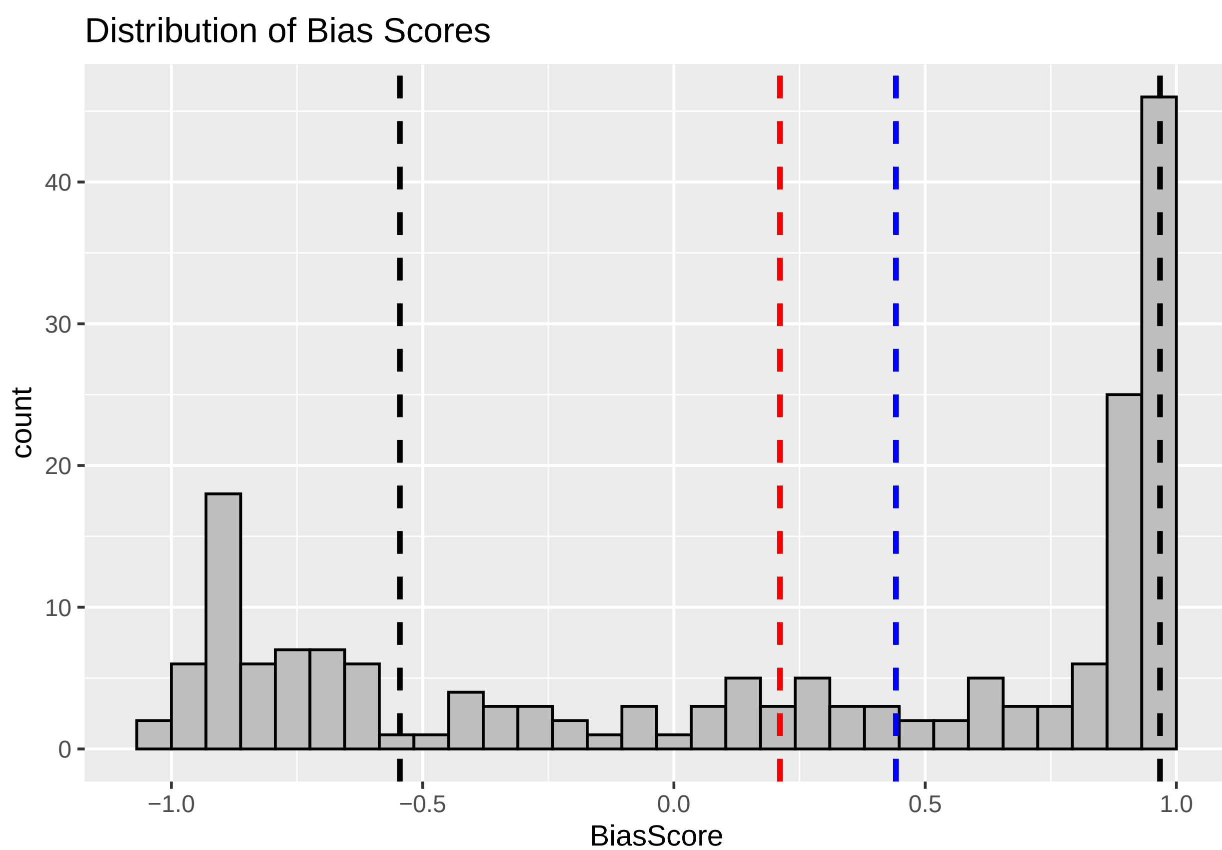
```
##                 vars   n   mean     sd median trimmed    mad     min     max
## Age                1 183  40.05  14.52  38.00   39.22  13.34   10.00   87.00
## IncorRespCount     2 185   1.83   3.12   1.00    1.13   1.48    0.00   19.00
## SymRespCount       3 185  18.10  17.66  12.00   16.96  16.31    0.00   48.00
## TxtRespCount       4 185  28.07  17.95  33.00   28.91  20.76    0.00   48.00
## BiasScore          5 185   0.21   0.76   0.44    0.25   0.77   -1.00    1.00
## Con_RT             6 185 885.79 174.50 866.42  875.88 160.51  529.08 1579.72
## InCon_RT           7 185 987.53 325.00 925.81  946.21 209.20  528.29 2863.08
## Incon.ConRT        8 185 101.74 239.42  46.57   62.58  74.70 -116.09 1902.28
##                   range  skew kurtosis    se
## Age               77.00  0.55     0.20  1.07
## IncorRespCount    19.00  3.35    11.70  0.23
## SymRespCount      48.00  0.45    -1.49  1.30
## TxtRespCount      48.00 -0.31    -1.61  1.32
## BiasScore          2.00 -0.38    -1.55  0.06
## Con_RT          1050.64  0.63     0.59 12.83
## InCon_RT        2334.79  2.91    13.12 23.89
## Incon.ConRT     2018.37  5.34    33.94 17.60
```

Distributions of Bias Scores

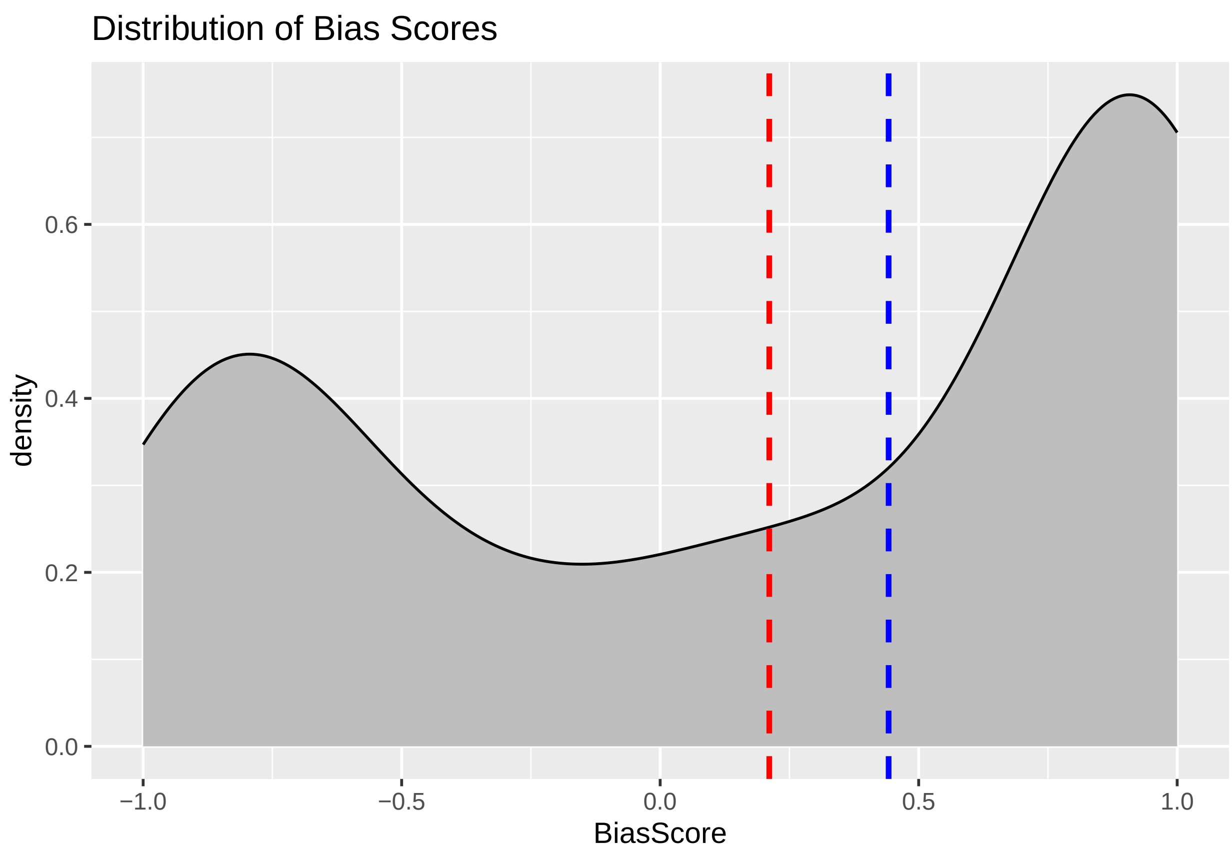
```r
biasMean <- mean(data$BiasScore, na.rm = T)
biasMedian <- median(data$BiasScore, na.rm = T)

bias1sd <- biasMean + sd(data$BiasScore)
biasneg1sd <- biasMean - sd(data$BiasScore)


data %>%
  ggplot(aes(x=BiasScore)) +
  geom_histogram(fill = "grey", color = "black") +
```

```
  geom_vline(mapping = aes(xintercept = biasMean), color = "red", linetype = "dashed", size = 1) +
  geom_vline(mapping = aes(xintercept = biasMedian), color = "blue", linetype = "dashed", size = 1) +
   geom_vline(mapping = aes(xintercept = bias1sd), color = "black", linetype = "dashed", size = 1) +
   geom_vline(mapping = aes(xintercept = biasneg1sd), color = "black", linetype = "dashed", size = 1) +
  ggtitle("Distribution of Bias Scores")
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

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




```
data %>%
  ggplot(aes(x=BiasScore)) +
  geom_density(fill = "grey", color = "black") +
   geom_vline(mapping = aes(xintercept = biasMean), color = "red", linetype = "dashed", size = 1) +
  geom_vline(mapping = aes(xintercept = biasMedian), color = "blue", linetype = "dashed", size = 1) +
  ggtitle("Distribution of Bias Scores")
```

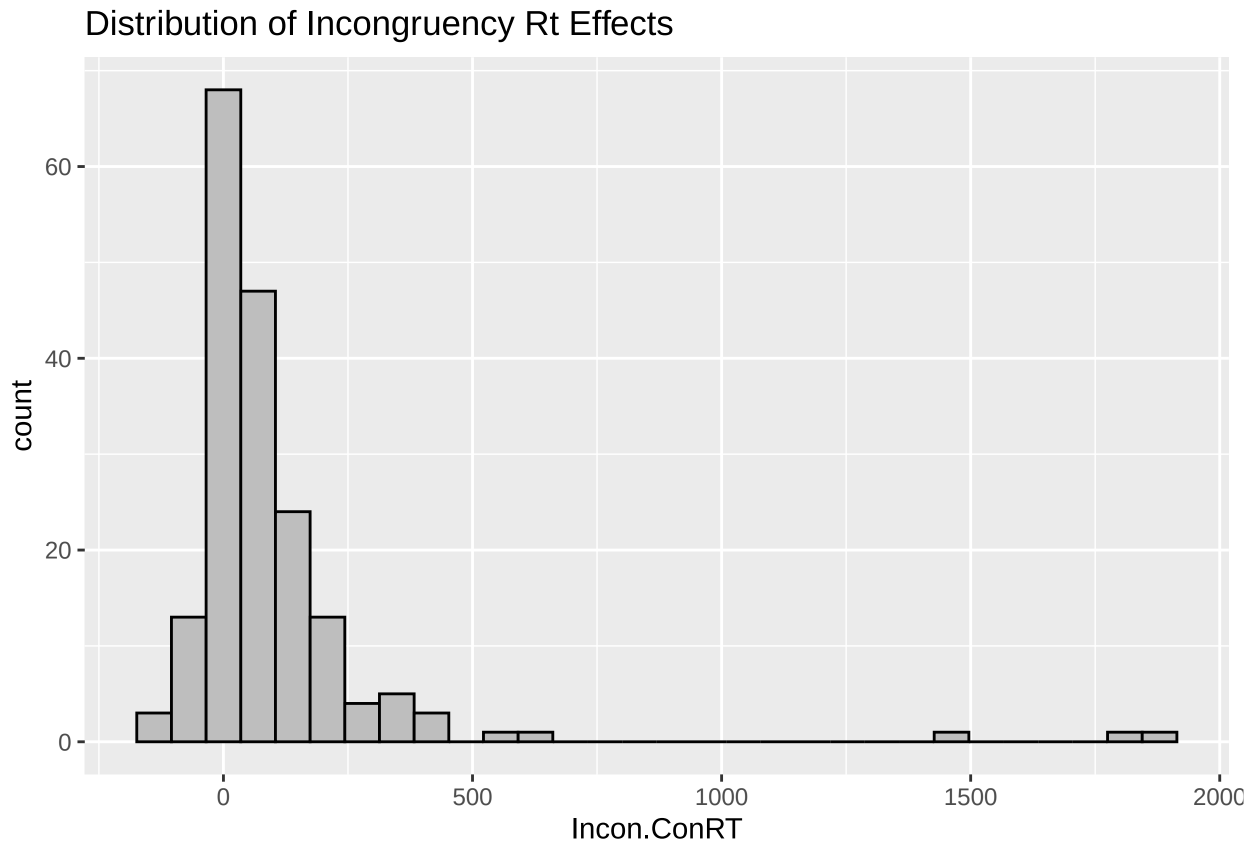


#A lot less neutral people than biased people

Distributions of Incongruency Effect - Whole Group & Bias Split

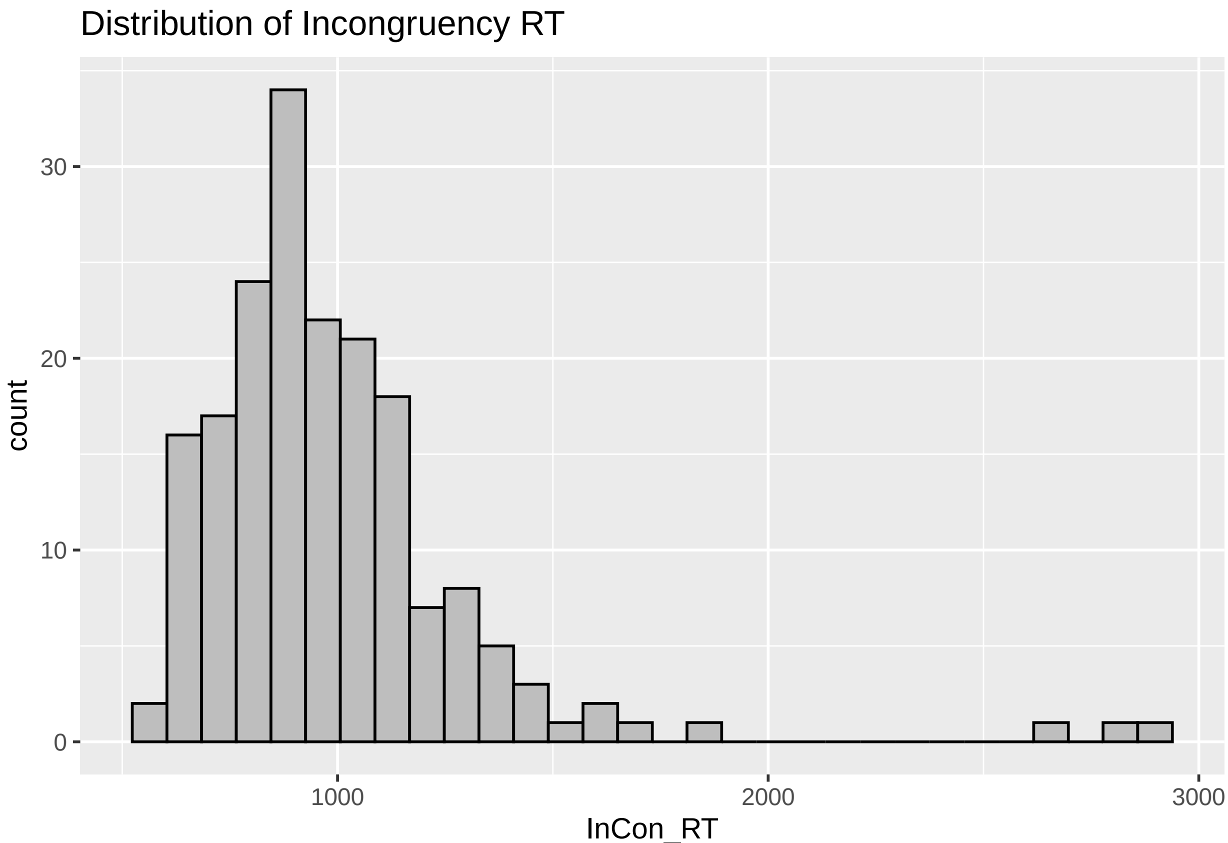
```
# Whole group distribution of incongruency effect

data %>%
  ggplot(aes(x=Incon.ConRT)) +
  geom_histogram(fill = "grey", color = "black") +
  ggtitle("Distribution of Incongruency Rt Effects")
```

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



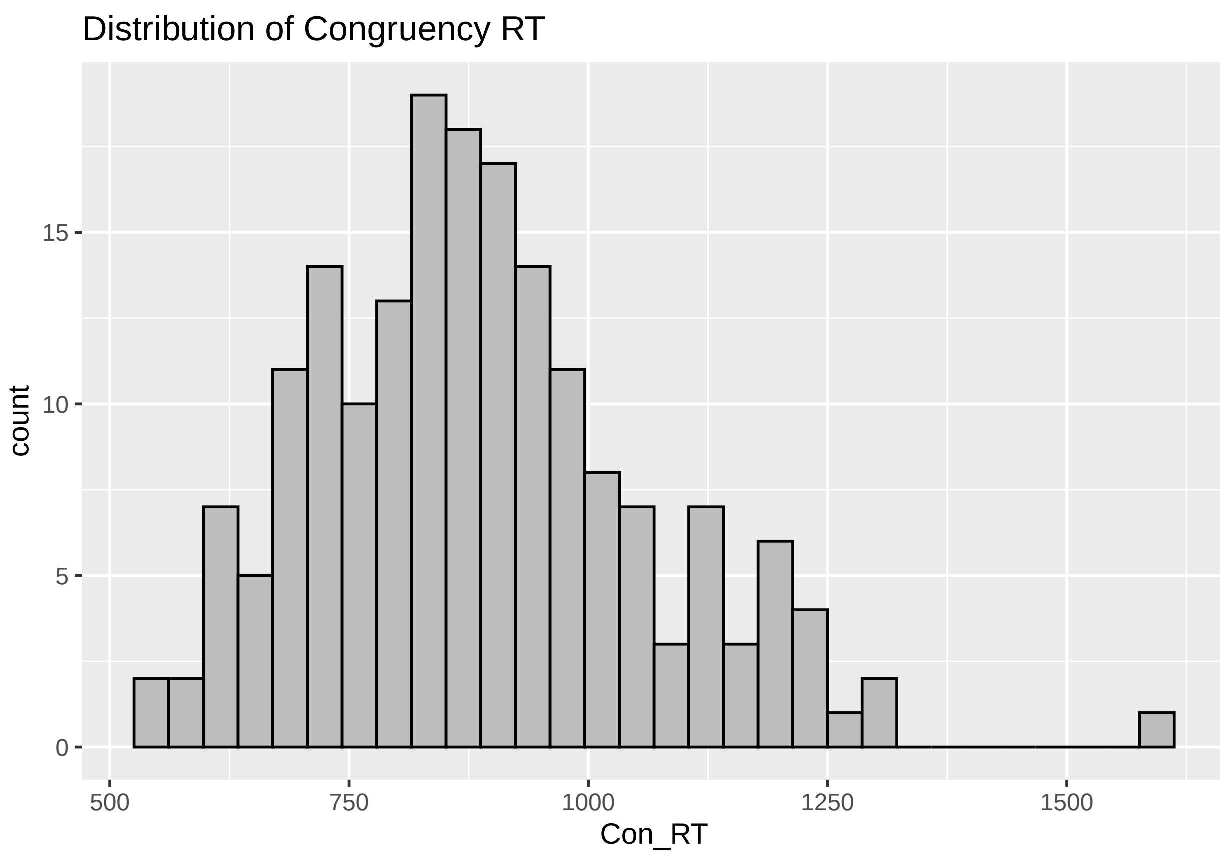

```
data %>%
  ggplot(aes(x=InCon_RT)) +
  geom_histogram(fill = "grey", color = "black") +
  ggtitle("Distribution of Incongruency RT")
```

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



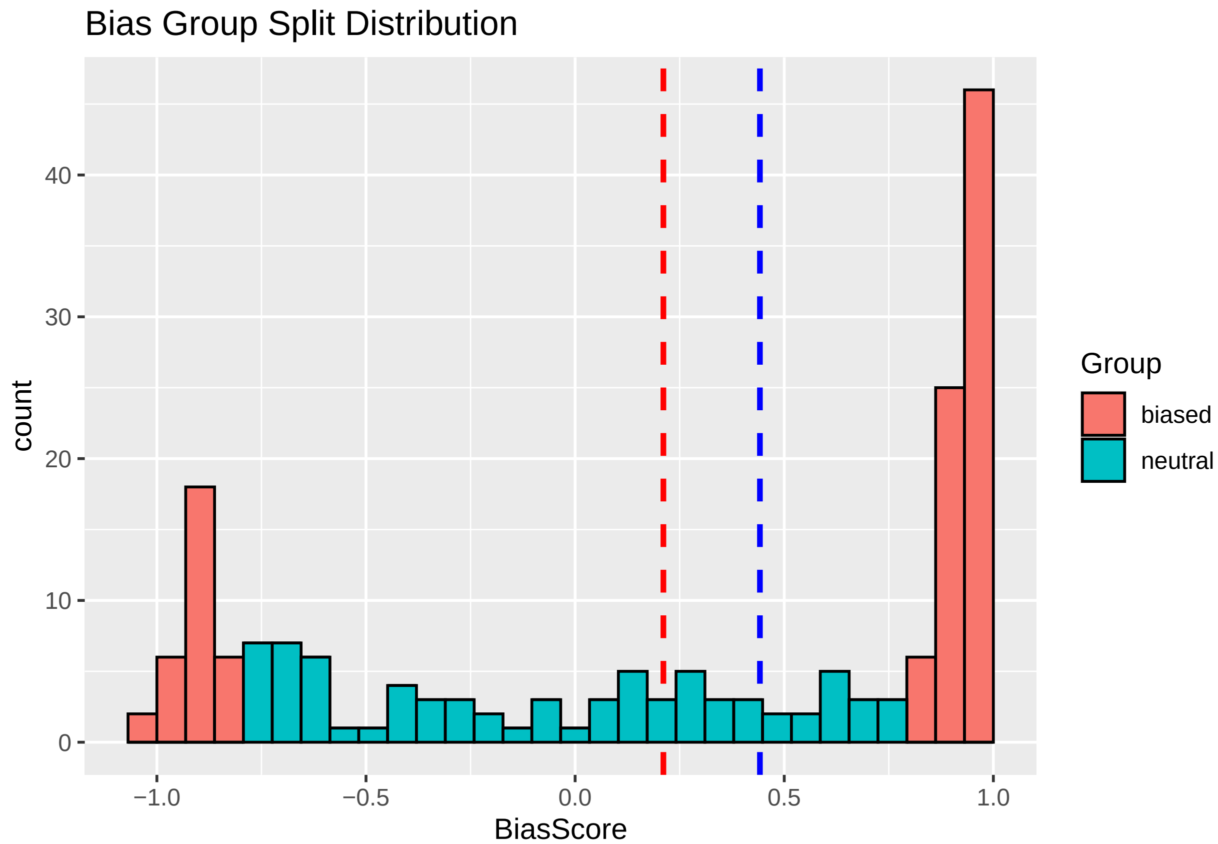

```
data %>%
  ggplot(aes(x=Con_RT)) +
  geom_histogram(fill = "grey", color = "black") +
  ggtitle("Distribution of Congruency RT")
```

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

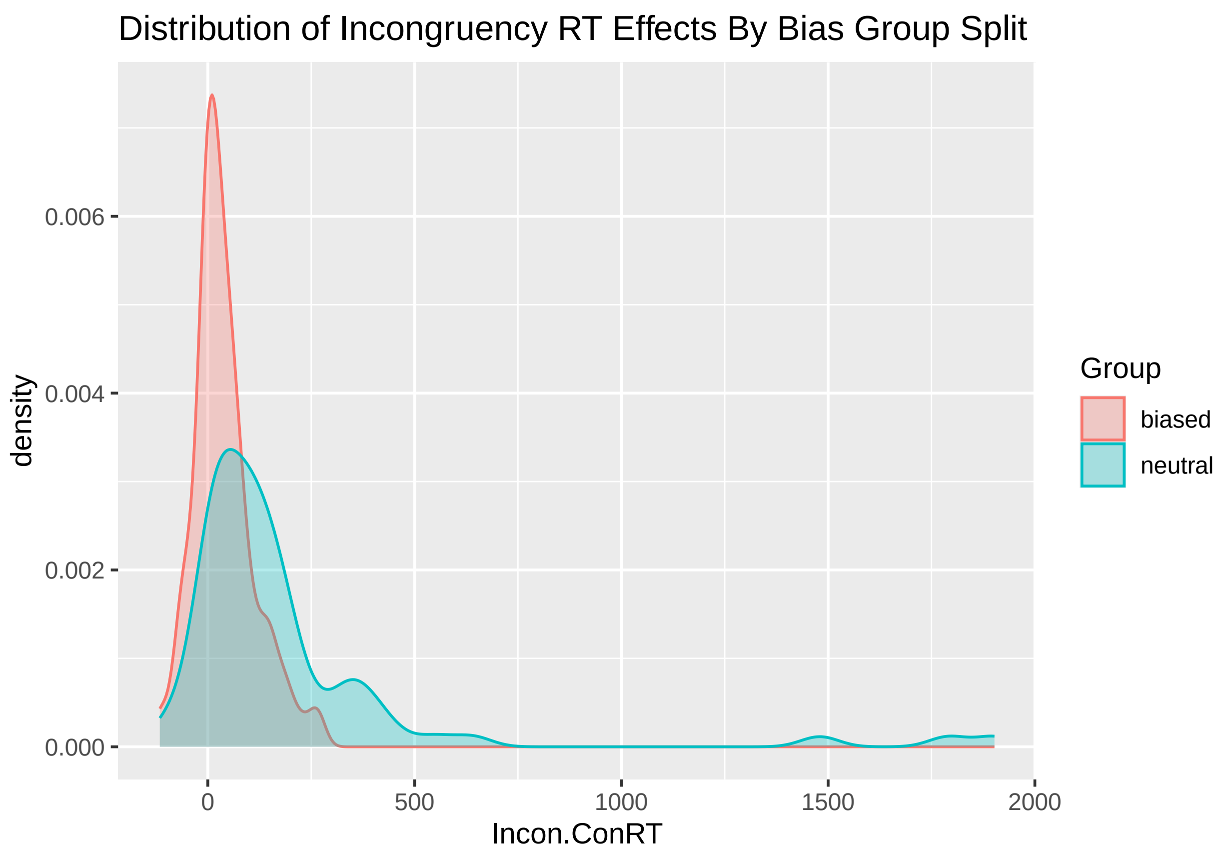



```
# Split groups based on bias score
biased <- subset(data, data$BiasScore > 0.8 | data$BiasScore < -0.8)
neutral <- subset(data, data$BiasScore <= 0.8 & data$BiasScore >= -0.8)

biased$Group = "biased"
neutral$Group = "neutral"

data_grouped <- rbind(biased, neutral)

data_grouped %>%
  ggplot(aes(x = BiasScore, fill = Group))+ geom_vline(mapping = aes(xintercept = biasMean), color = "r
  geom_vline(mapping = aes(xintercept = biasMedian), color = "blue", linetype = "dashed", size = 1) +
  geom_histogram(color="black") + ggtitle("Bias Group Split Distribution")
```

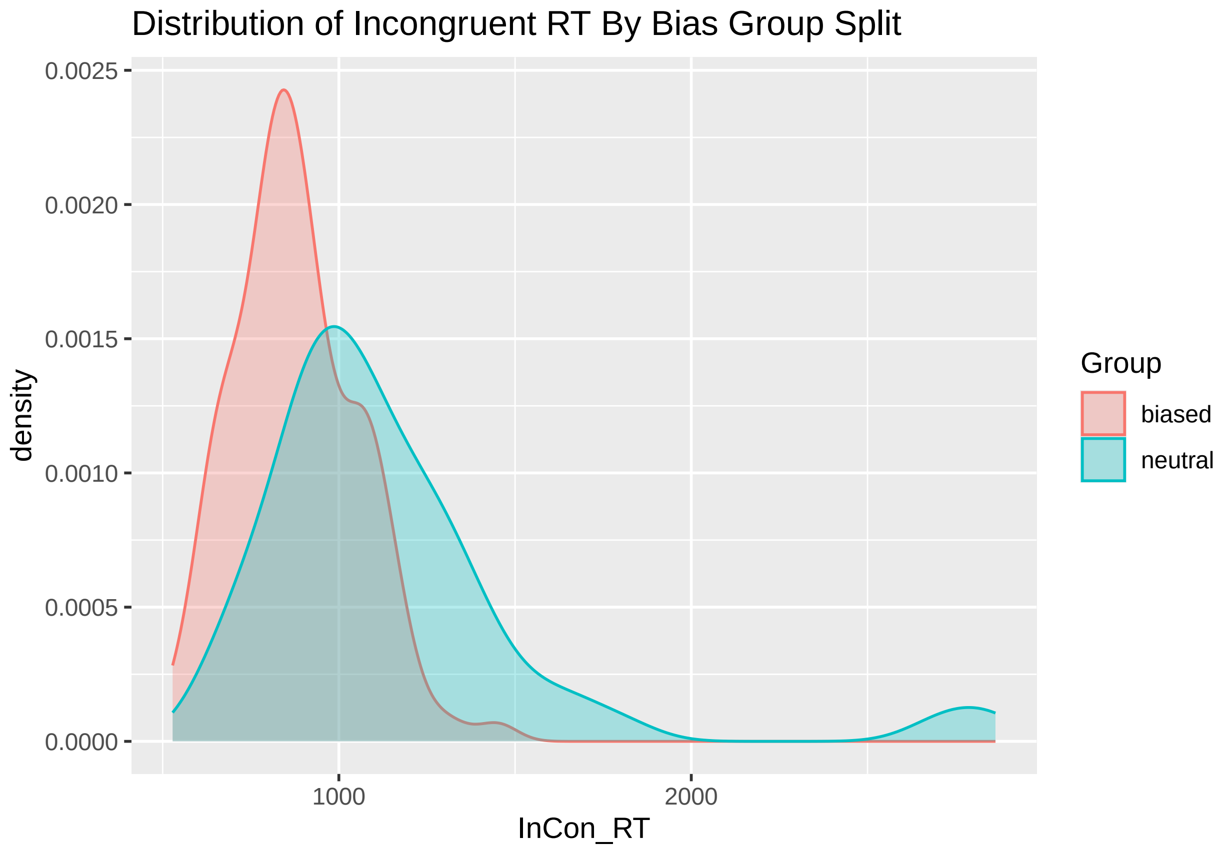
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



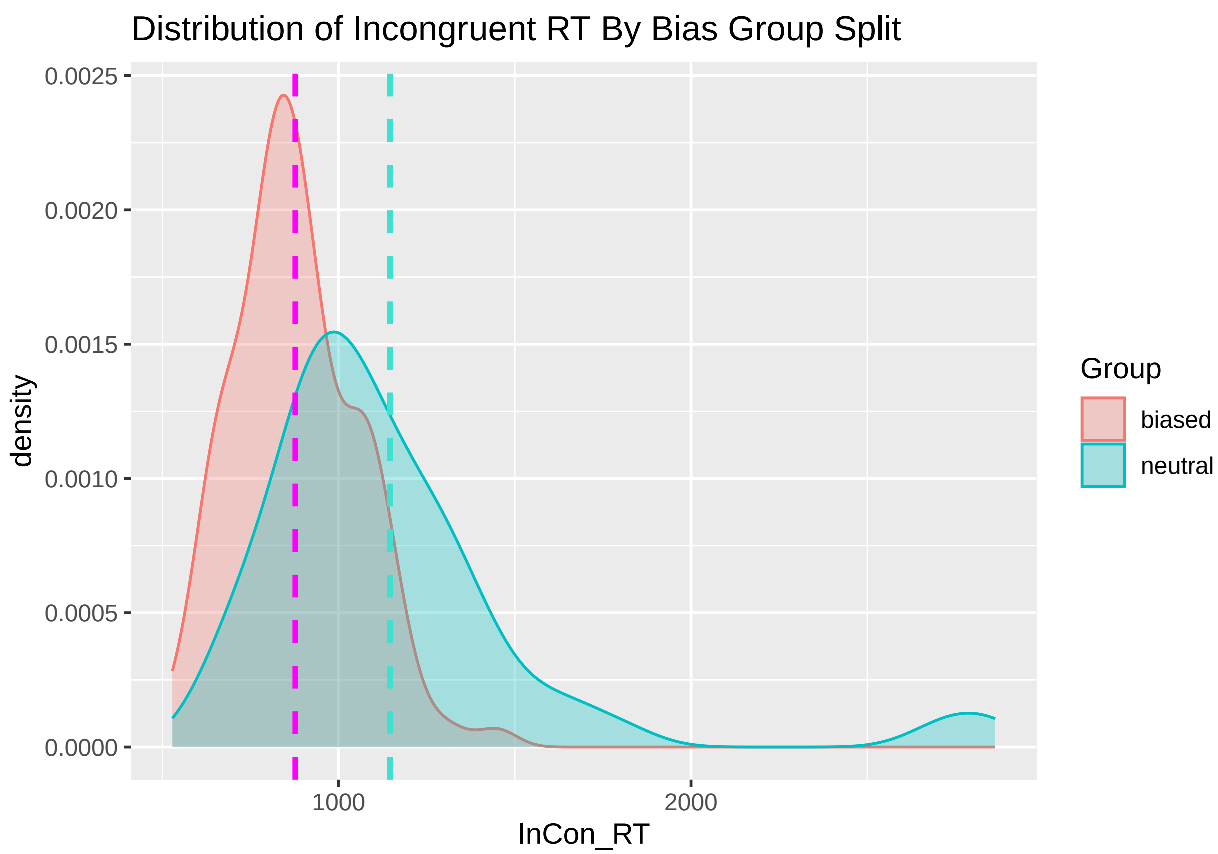

```r
data_grouped %>%
  ggplot(aes(x = Incon.ConRT, fill = Group, color = Group))+
  geom_density(alpha = 0.3)+
  ggtitle("Distribution of Incongruency RT Effects By Bias Group Split")
```



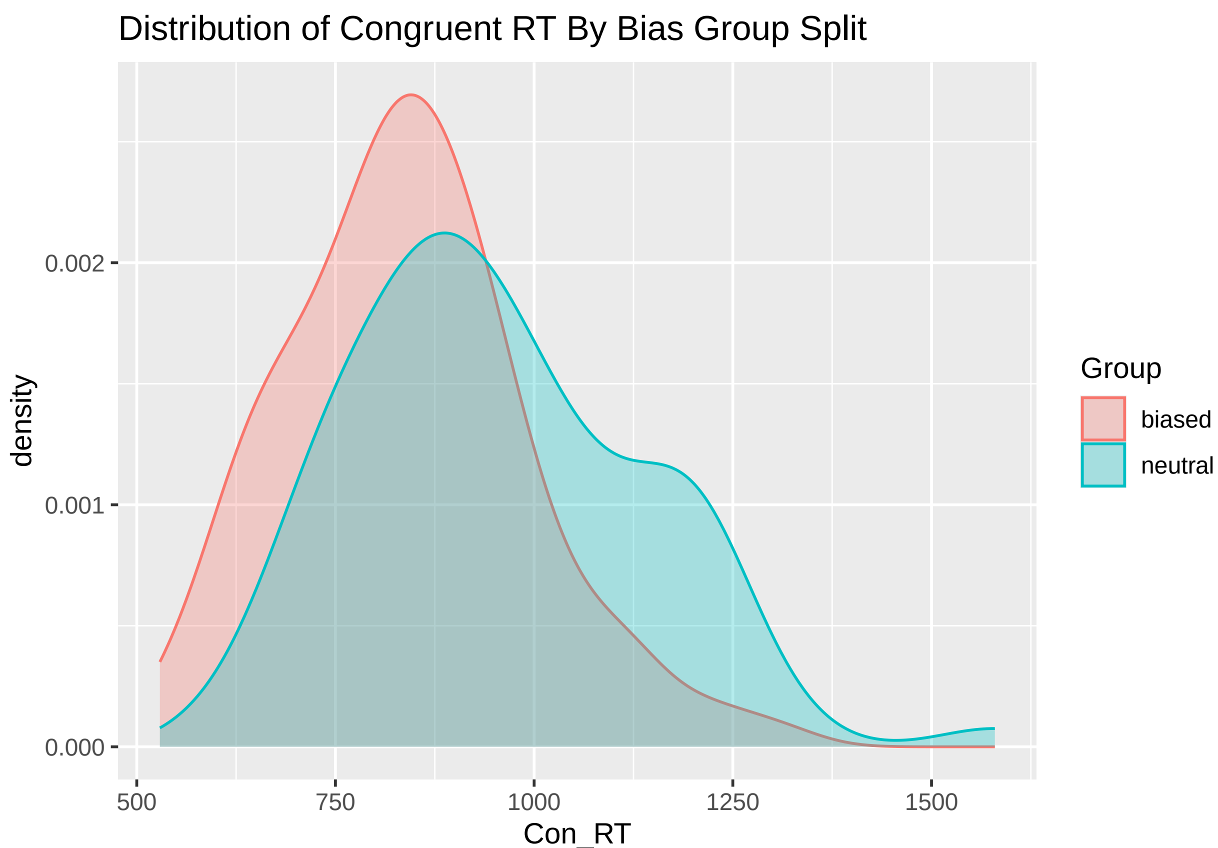

```r
data_grouped %>%
  ggplot(aes(x = InCon_RT, fill = Group, color = Group))+
  geom_density(alpha = 0.3)+
  ggtitle("Distribution of Incongruent RT By Bias Group Split")
```



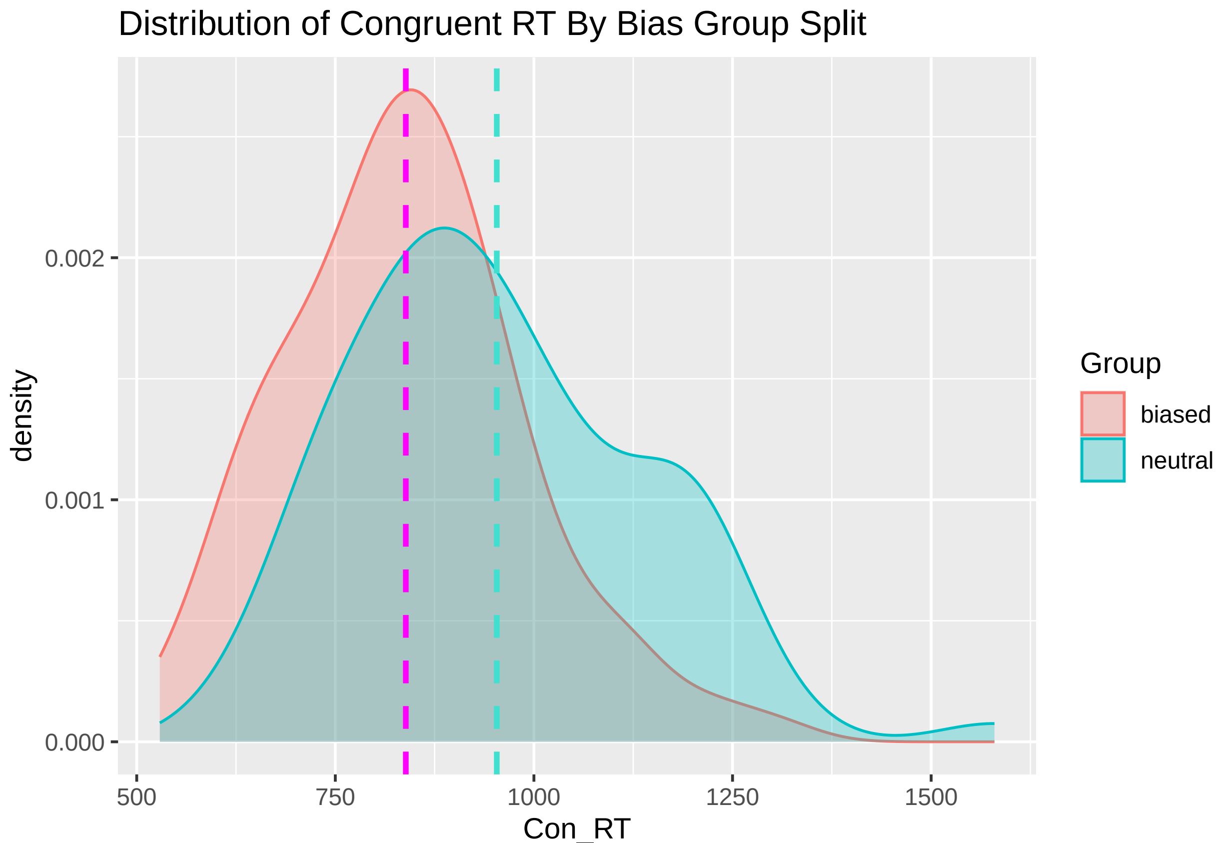

```
neutral_incon_mean = mean(neutral$InCon_RT)
neutral_con_mean = mean(neutral$Con_RT)
biased_incon_mean = mean(biased$InCon_RT)
biased_con_mean = mean(biased$Con_RT)


data_grouped %>%
  ggplot(aes(x = InCon_RT, fill = Group, color = Group))+
  geom_density(alpha = 0.3)+geom_vline(mapping = aes(xintercept = neutral_incon_mean), color = "turquoi
  ggtitle("Distribution of Incongruent RT By Bias Group Split")
```




```
data_grouped %>%
  ggplot(aes(x = Con_RT, fill = Group, color = Group))+
  geom_density(alpha = 0.3)+
  ggtitle("Distribution of Congruent RT By Bias Group Split")
```




```
data_grouped %>%
  ggplot(aes(x = Con_RT, fill = Group, color = Group))+
  geom_density(alpha = 0.3)+geom_vline(mapping = aes(xintercept = neutral_con_mean), color = "turquoise
  ggtitle("Distribution of Congruent RT By Bias Group Split")
```



#Incongruency RT Effect Dist: Most people have an incongruency effect of 0 or > so, as expected, incongruent trial reaction times are generally greater than congruent trial reaction times. Slight right skew that is made extreme by outliers- will reexamine with outliers removed. #Distribution of Incongruency Effect by Bias Group Split: There are more people who have a lower incongruency effect in the biased attender group than in the neutral attender group. Neutral attender group has a lot less people but its density is a lot more spread out while biased attender incongruency effect has a lot less range and is more concentrated near 0. #Congruent and Incongruent Distributions by Bias Group Split: Biased group seems to be taking less time than the neutral group in both trial types. Could be indication of biased attenders mainly paying attention to their preferred information processing style and ignoring the other stimulus so RTs are faster in both trial types.

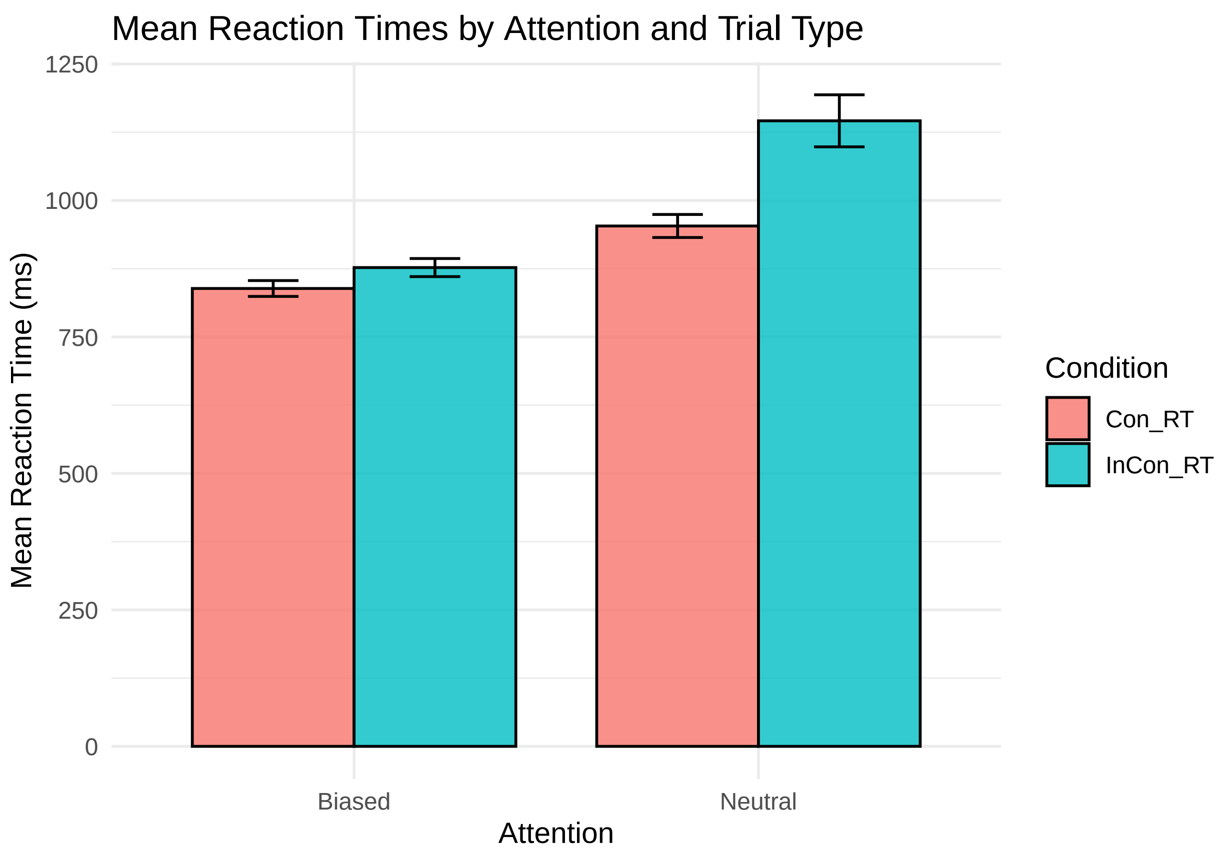
Reaction times by trial type and attention

```
# Load necessary libraries
library(tidyverse)
library(ggplot2)
library(reshape2)

data$Attention <- ifelse(data$BiasScore > 0.8 | data$BiasScore < -0.8, "Biased", "Neutral")
data$IPS <- ifelse(data$BiasScore > 0, "Verbal", "Visual")

#Reaction times
#Biased attender histograms and descriptive statistics
biased_data<- data[data$Attention == "Biased", ]
```

```r
neutral_data<- data[data$Attention == "Neutral", ]

# Combine biased and neutral data
data$Attention <- ifelse(data$BiasScore > 0.8 | data$BiasScore < -0.8, "Biased", "Neutral")
biased_data<- data[data$Attention == "Biased", ]
combined_data <- rbind(biased_data, neutral_data)

# Calculate means by attention and trial type
means <- combined_data %>%
  group_by(Attention) %>%
  summarise(Con_RT = mean(Con_RT), InCon_RT = mean(InCon_RT)) %>%
  pivot_longer(cols = c(Con_RT, InCon_RT), names_to = "Trial_Type", values_to = "RT")

# Order factor levels for better plotting
means$Attention <- factor(means$Attention, levels = unique(means$Attention))

combined_long <- combined_data %>%
  select(Attention, InCon_RT, Con_RT) %>%
  pivot_longer(cols = c(InCon_RT, Con_RT), values_to = "RT", names_to = "Condition")

# Creat error bars
se_sum <- combined_long %>%
  group_by(Attention, Condition) %>%
  summarise(
    sd = sd(RT),
    n = n(),
    mean = mean(RT)
  ) %>%
  mutate(se = sd/sqrt(n))
```

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

```r
# Create bar plot

ggplot(se_sum, aes(x = Attention, y = mean, fill = Condition)) +
  geom_bar(position = position_dodge(0.8), stat = "identity", color = "black", size = 0.5, width = 0.8,
  geom_errorbar(aes(ymin = mean - se, ymax = mean + se), position = position_dodge(0.8), width = 0.25,
  labs(title = "Mean Reaction Times by Attention and Trial Type",
       x = "Attention",
       y = "Mean Reaction Time (ms)") +
  theme_minimal()
```




```
combined_long_wid <- combined_data %>%
  select(Attention, InCon_RT, Con_RT, Subject) %>%
  pivot_longer(cols = c(InCon_RT, Con_RT), values_to = "RT", names_to = "Condition")
```

#incon rts are larger overall, but the difference between incon and con trial rts for the biased group is a lot smaller than the neutral group- perhaps because the neutral group is noticing both stimuli more? #figure out how to put error bars

Incongruency Effect Analysis

Incongruency Effect Descriptives

```
# Incongruency Effect Calculation
data$IncongruencyEffect <- data$InCon_RT - data$Con_RT
Incongruency_Effect_Data <- data$IncongruencyEffect

biased_data$Incongruency_Effect_Data <- biased_data$InCon_RT - biased_data$Con_RT
neutral_data$Incongruency_Effect_Data <- neutral_data$InCon_RT - neutral_data$Con_RT
```

Biased group descriptives

```r
psych::describe(biased_data$Incongruency_Effect_Data)
```

```
##    vars   n  mean   sd median trimmed   mad     min    max  range skew kurtosis
## X1    1 109 38.32 72.9  25.15   32.84 52.51 -114.36 265.55 379.91 0.85     0.97
##      se
## X1 6.98
```

Neutral group descriptives

```r
psych::describe(neutral_data$Incongruency_Effect_Data)
```

```
##    vars  n   mean    sd median trimmed    mad     min     max   range skew
## X1    1 76 192.68 344.6  99.85  124.84 121.86 -116.09 1902.28 2018.37  3.6
##    kurtosis    se
## X1    13.72 39.53
```

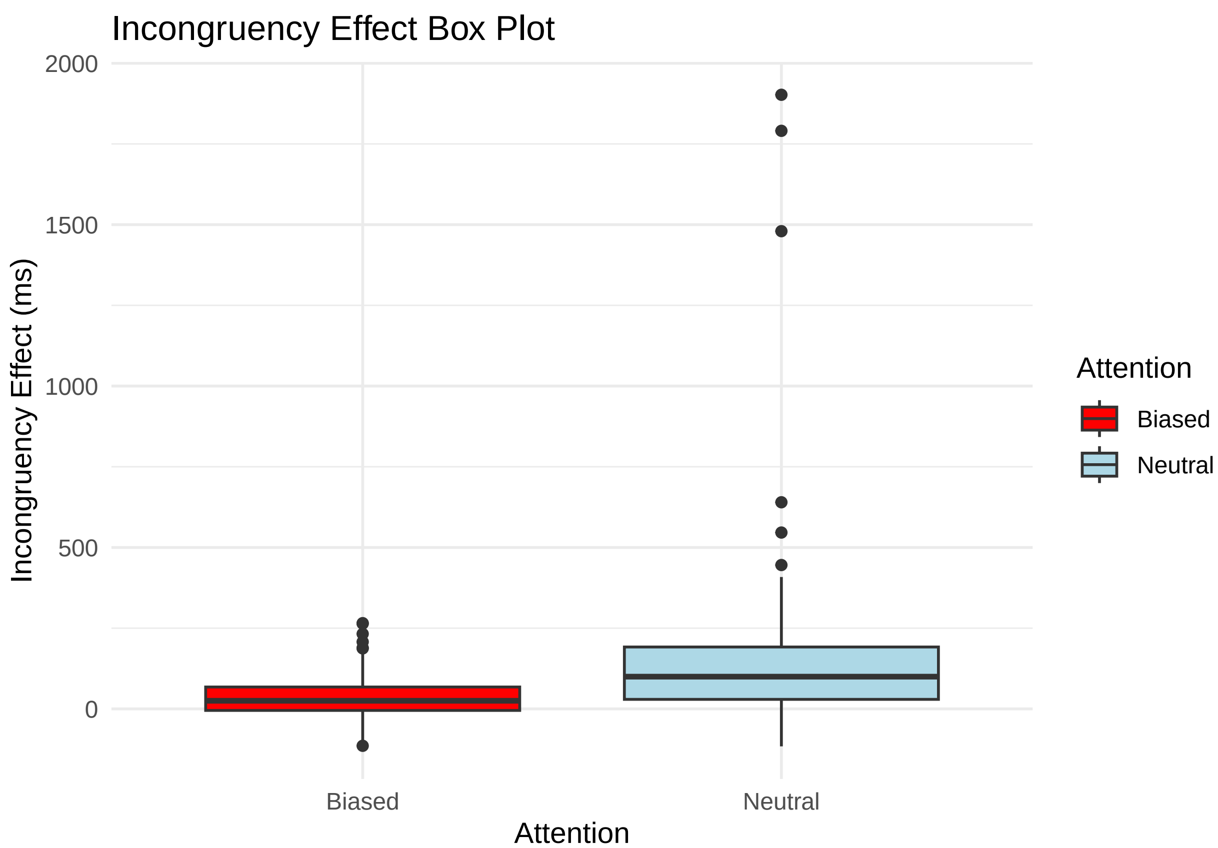
```r
# Biased Attender Incongruency Effect Descriptive Statistics
biased_IE_mean <- mean(biased_data$Incongruency_Effect_Data)
biased_IE_std <- sd(biased_data$Incongruency_Effect_Data)
biased_IE_min <- min(biased_data$Incongruency_Effect_Data)
biased_IE_max <- max(biased_data$Incongruency_Effect_Data)

# Data frame for Biased Attender Incongruency Effect Descriptive Statistics
biased_descriptive_IE <- data.frame(
  Attention = "Biased",
  Variable = "Incongruency Effect",
  Mean = biased_IE_mean,
  StdDev = biased_IE_std,
  Min = biased_IE_min,
  Max = biased_IE_max,
  stringsAsFactors = FALSE
)

# Neutral Attender Incongruency Effect Descriptive Statistics
neutral_IE_mean <- mean(neutral_data$Incongruency_Effect_Data)
neutral_IE_std <- sd(neutral_data$Incongruency_Effect_Data)
neutral_IE_min <- min(neutral_data$Incongruency_Effect_Data)
neutral_IE_max <- max(neutral_data$Incongruency_Effect_Data)

# Data frame for Neutral Attender Incongruency Effect Descriptive Statistics
neutral_descriptive_IE <- data.frame(
  Attention = "Neutral",
  Variable = "Incongruency Effect",
  Mean = neutral_IE_mean,
  StdDev = neutral_IE_std,
  Min = neutral_IE_min,
  Max = neutral_IE_max,
  stringsAsFactors = FALSE
```

```
)

descriptive_statistics_IE <- rbind(biased_descriptive_IE, neutral_descriptive_IE)
print(descriptive_statistics_IE)
```

```
##   Attention             Variable      Mean   StdDev       Min       Max
## 1    Biased Incongruency Effect  38.32225  72.8954 -114.3611  265.5486
## 2   Neutral Incongruency Effect 192.68412 344.5965 -116.0903 1902.2847
```

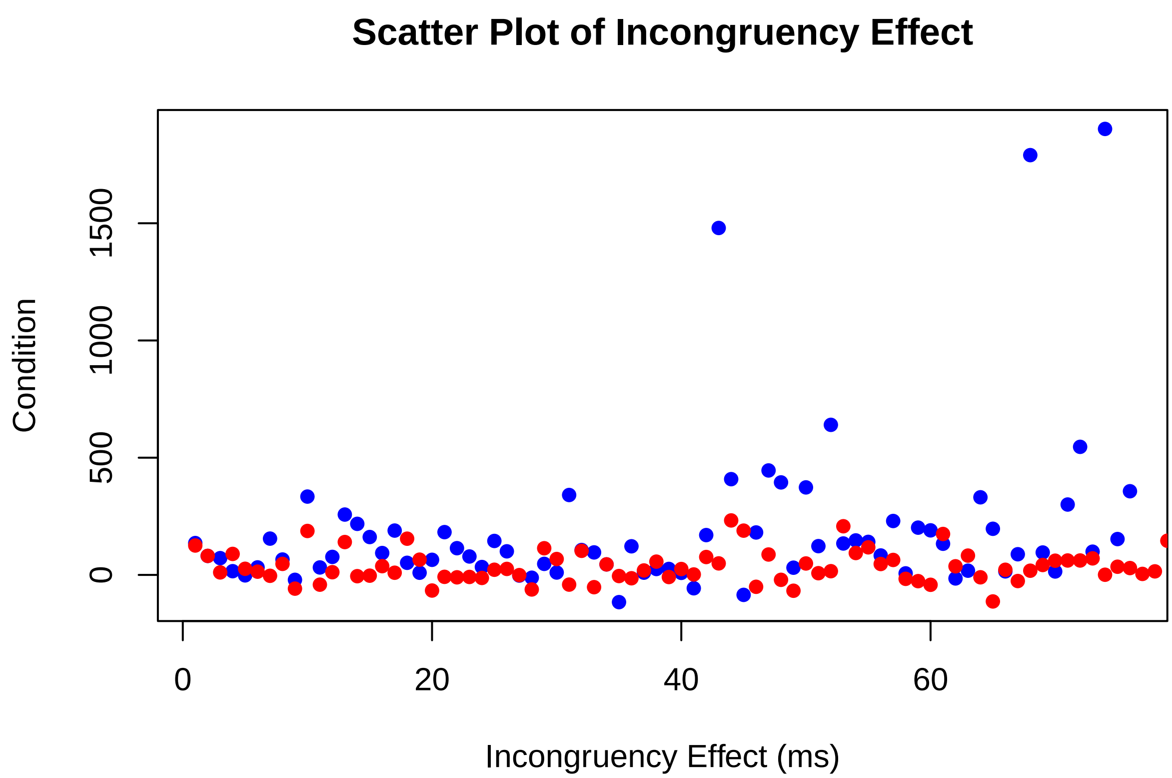
Incongruency Effect Box Plot

```
# Load necessary libraries
library(ggplot2)

# Combine biased and neutral data for the box plot
combined_data <- rbind(biased_data, neutral_data)

# Create box plot
ggplot(combined_data, aes(x = Attention, y = Incongruency_Effect_Data, fill = Attention)) +
  geom_boxplot() +
  labs(
    title = "Incongruency Effect Box Plot",
    x = "Attention",
    y = "Incongruency Effect (ms)"
  ) +
  scale_fill_manual(values = c("red", "lightblue")) +  # Color for biased and neutral data
  theme_minimal()
```



Incongruency Effect Scatter Plot

```
# Scatter Plot of Incongruency Effect
plot(neutral_data$Incongruency_Effect_Data,
     col = "blue",
     pch = 16,
     main = "Scatter Plot of Incongruency Effect",
     xlab = "Incongruency Effect (ms)",
     ylab = "Condition")

# Adding Biased Attender Data Points
points(biased_data$Incongruency_Effect_Data,
       col = "red",
       pch = 16)
```



Individual Differences Analysis

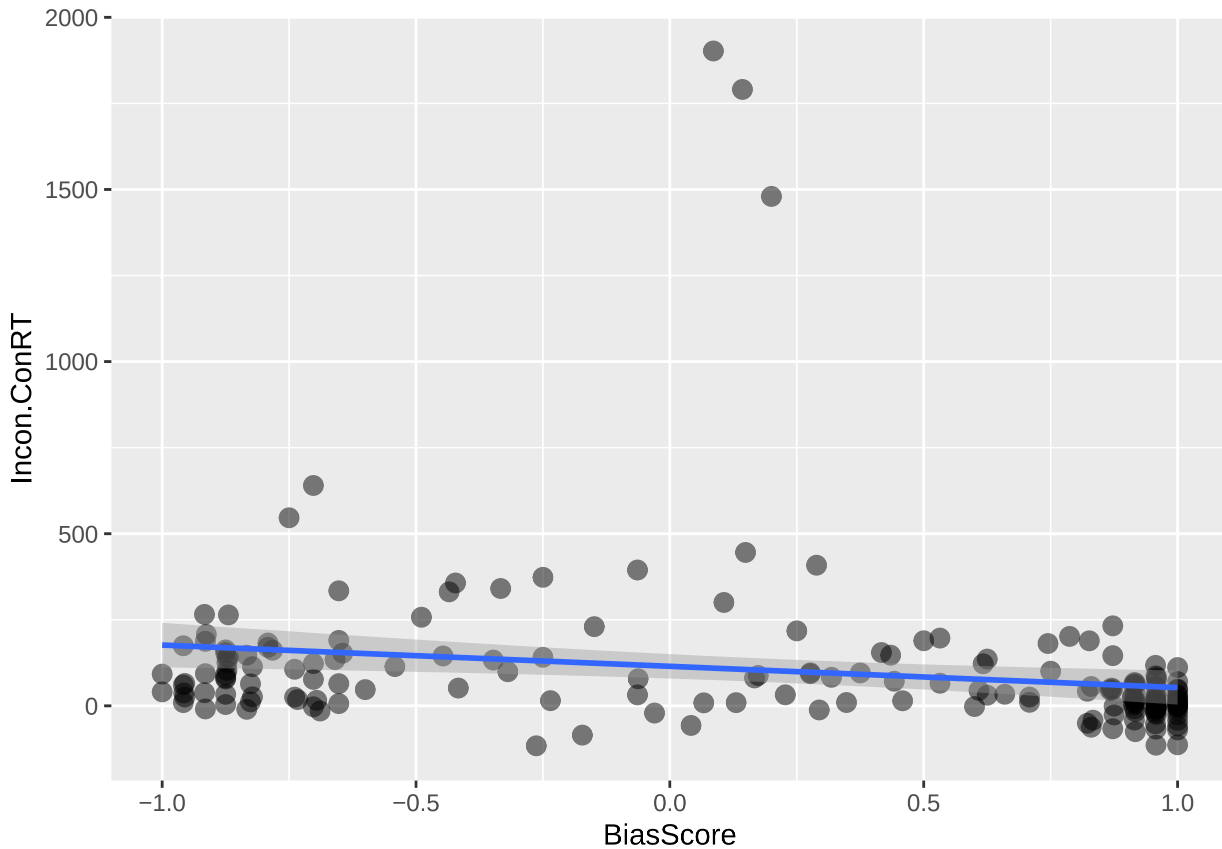
Correlate Bias score in whole group with incongruency RT effect

```
cor.test(data$BiasScore, data$Incon.ConRT)
```

```
##
##  Pearson's product-moment correlation
##
## data:  data$BiasScore and data$Incon.ConRT
## t = -2.6908, df = 183, p-value = 0.007789
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.33006526 -0.05228936
## sample estimates:
##        cor
## -0.1950863
```

```
data %>%
  ggplot(aes(x = BiasScore, y = Incon.ConRT)) +
  geom_point(size = 3, alpha = 0.5, fill = "grey") +
  geom_smooth(method = "lm")
```

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

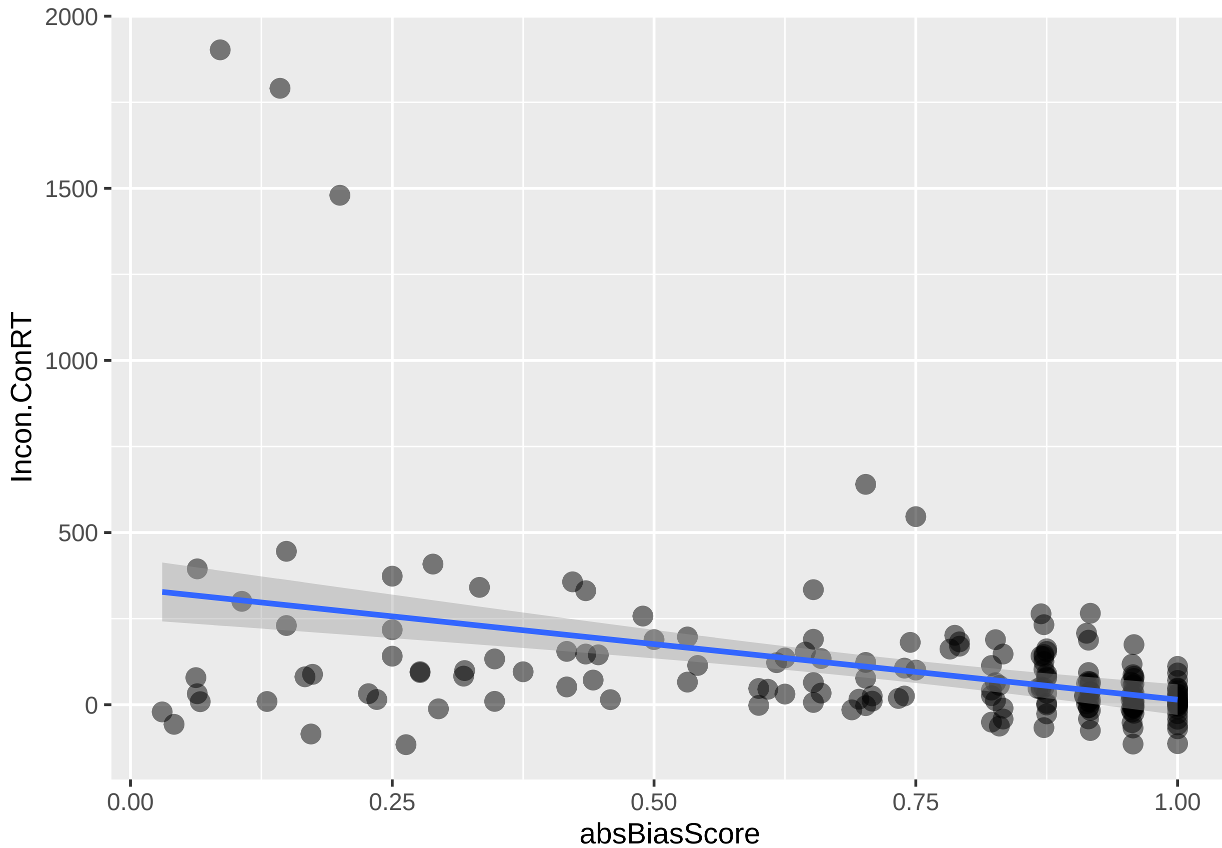



```r
data$absBiasScore <- abs(data$BiasScore)

cor.test(data$absBiasScore, data$Incon.ConRT)
```

```
##
##  Pearson's product-moment correlation
##
## data:  data$absBiasScore and data$Incon.ConRT
## t = -5.6186, df = 183, p-value = 7.071e-08
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.5001629 -0.2533216
## sample estimates:
##       cor
## -0.383572
```

```r
data %>%
  ggplot(aes(x = absBiasScore, y = Incon.ConRT)) +
  geom_point(size = 3, alpha = 0.5, fill = "grey") +
  geom_smooth(method = "lm")
```

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

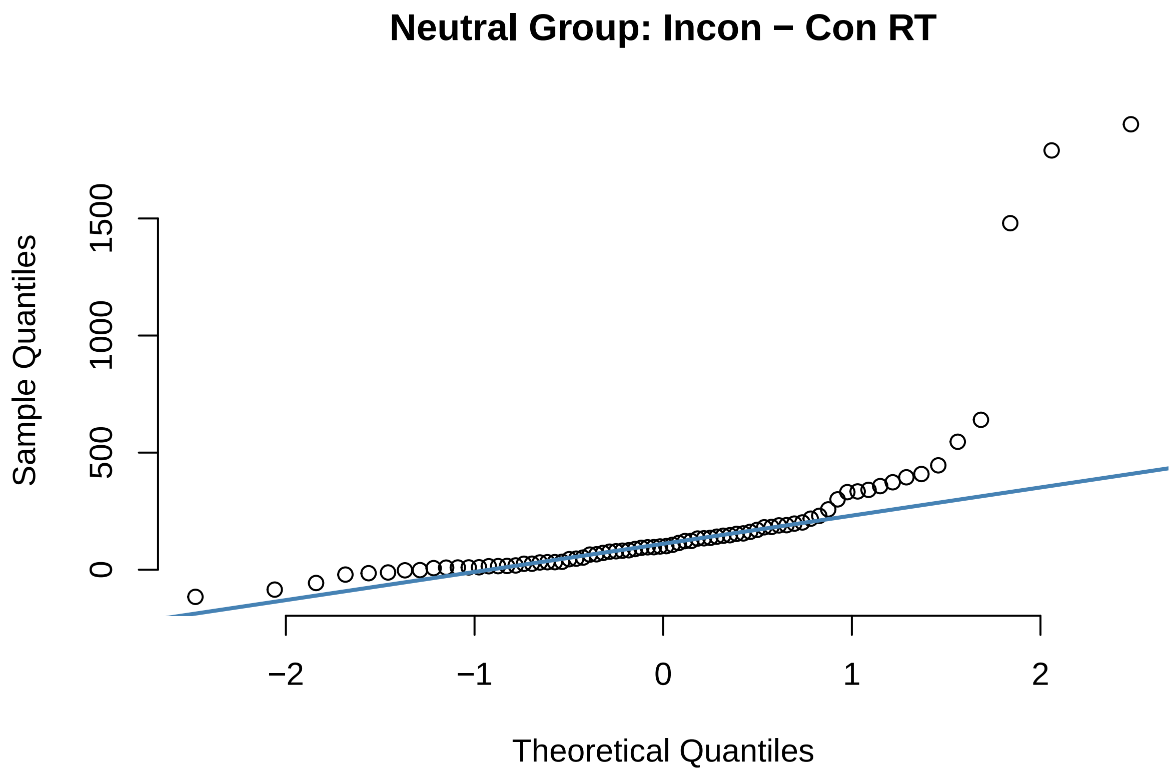


#doesn't look like much of a correlation, reexamine with outliers removed.

```
combined_long_wid <- combined_data %>%
  select(Attention, InCon_RT, Con_RT, Subject) %>%
  pivot_longer(cols = c(InCon_RT, Con_RT), values_to = "RT", names_to = "Condition")

#anova_test(data=combined_long_wid, dv=RT, wid=Subject, between=Attention, within = Condition)
```

T-test code - with full data (keeping outliers)

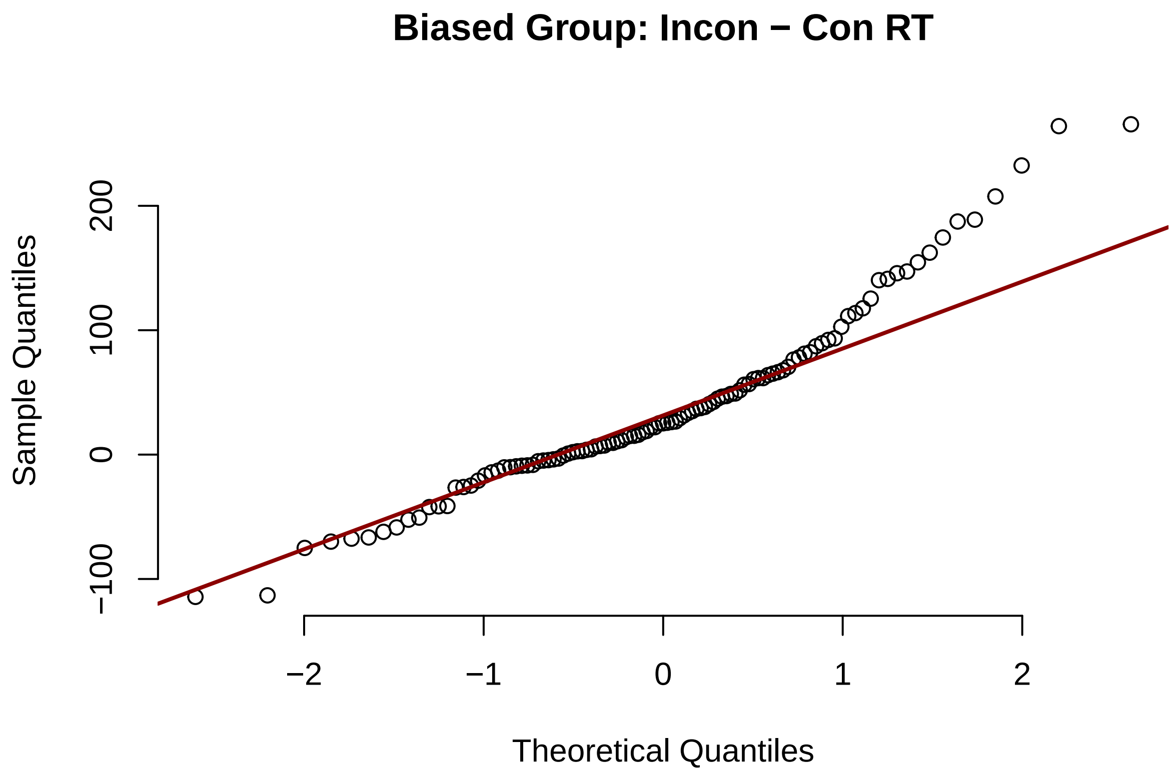
```
#   Difference in Incon-Con RT between Attention Groups

biased<- data[data$Attention == "Biased", ]
neutral<- data[data$Attention == "Neutral", ]

#First test normality assumption in Neutral Group
qqnorm(neutral$Incon.ConRT, pch = 1, frame = FALSE, main = "Neutral Group: Incon - Con RT")
qqline(neutral$Incon.ConRT, col = "steelblue", lwd = 2)
```




```r
shapiro.test(neutral$Incon.ConRT) # Assumption of normality is violated; probably due to outliers
```

```
##
##  Shapiro-Wilk normality test
##
## data:  neutral$Incon.ConRT
## W = 0.5497, p-value = 6.594e-14
```

```r
#Then test normality assumption in Biased Group
qqnorm(biased$Incon.ConRT, pch = 1, frame = FALSE, main = "Biased Group: Incon - Con RT")
qqline(biased$Incon.ConRT, col = "darkred", lwd = 2)
```




```r
shapiro.test(biased$Incon.ConRT) # Assumption of normality is marginally violated
```

```
##
##  Shapiro-Wilk normality test
##
## data:  biased$Incon.ConRT
## W = 0.94661, p-value = 0.0002631
```

```r
#Check that the variance does not differ between groups
# Perform Levene's AT_FormTest
print(leveneTest(Incon.ConRT ~ Attention, data = combined_data)) # Variances are marginally different;
```

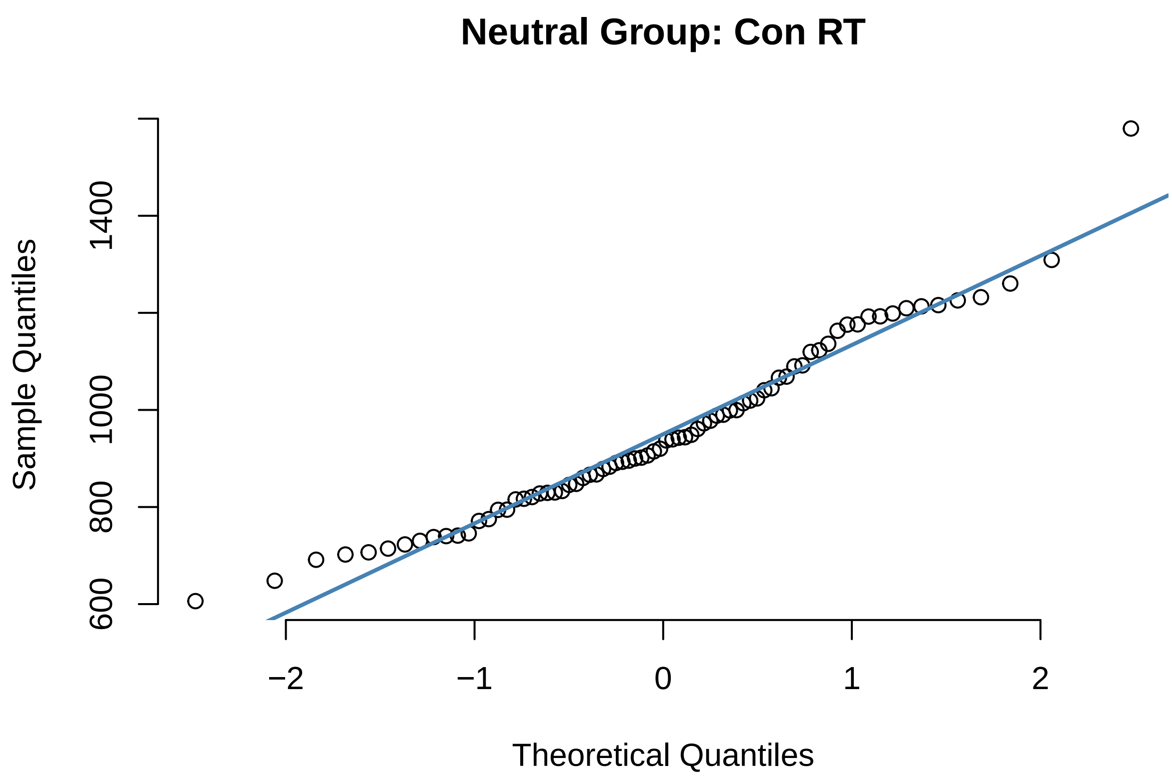
```
## Warning in leveneTest.default(y = y, group = group, ...): group coerced to
## factor.
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value    Pr(>F)
## group   1  12.698 0.0004668 ***
##       183
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Conduct t-test with equal variance assumption
print(t.test(neutral$Incon.ConRT, biased$Incon.ConRT, var.equal = F)) # T-Test is significant after cor
```

```
##
##  Welch Two Sample t-test
##
## data:  neutral$Incon.ConRT and biased$Incon.ConRT
## t = 3.8456, df = 79.699, p-value = 0.0002408
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##   74.47641 234.24733
## sample estimates:
## mean of x mean of y
## 192.68412  38.32225
```

```r
# test normality
qqnorm(neutral$Con_RT, pch = 1, frame = FALSE, main = "Neutral Group: Con RT")
qqline(neutral$Con_RT, col = "steelblue", lwd = 2)
```

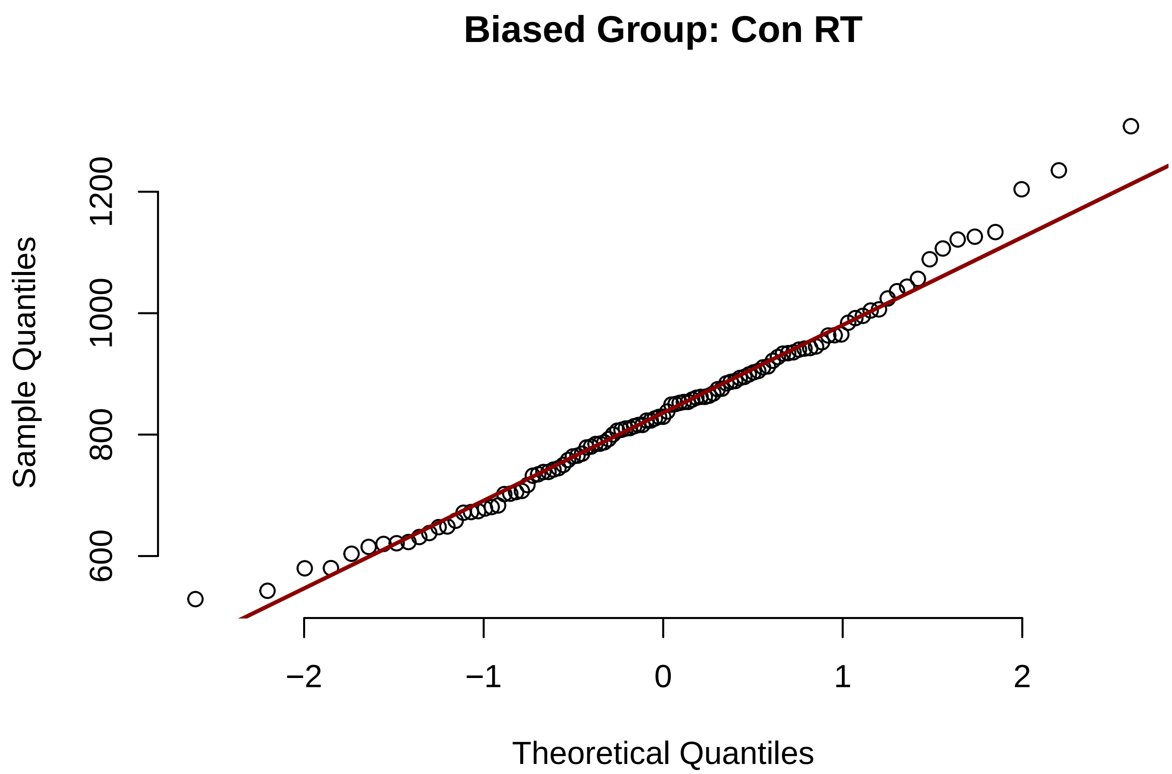



```r
shapiro.test(neutral$Con_RT) #assumption of normality violated
```

```
##
##  Shapiro-Wilk normality test
##
```

```
## data:  neutral$Con_RT
## W = 0.96673, p-value = 0.04339
```

```
qqnorm(biased$Con_RT, pch = 1, frame = FALSE, main = "Biased Group: Con RT")
qqline(biased$Con_RT, col = "darkred", lwd = 2)
```




```
shapiro.test(biased$Con_RT)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  biased$Con_RT
## W = 0.98426, p-value = 0.2281
```

```
# check if variance differs between groups
print(leveneTest(Con_RT ~ Attention, data = combined_data)) #assumption of homogeneity violated
```

```
## Warning in leveneTest.default(y = y, group = group, ...): group coerced to
## factor.
```

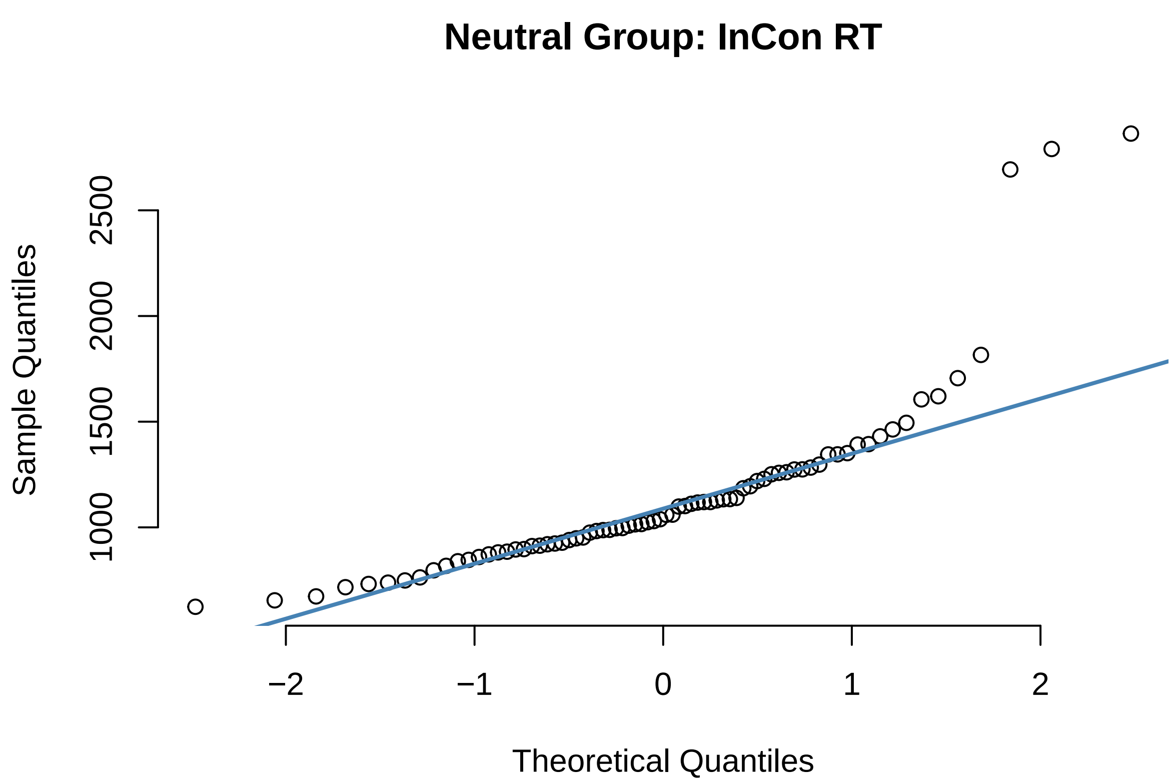
```
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value  Pr(>F)
## group   1  3.3796 0.06763 .
##       183
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#do t-test
print(t.test(neutral$Con_RT, biased$Con_RT, var.equal = F))
```

```
##
##  Welch Two Sample t-test
##
## data:  neutral$Con_RT and biased$Con_RT
## t = 4.4737, df = 141.06, p-value = 1.57e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##   63.90959 165.11498
## sample estimates:
## mean of x mean of y
##  953.2613  838.7490
```

```r
# test normality
qqnorm(neutral$InCon_RT, pch = 1, frame = FALSE, main = "Neutral Group: InCon RT")
qqline(neutral$InCon_RT, col = "steelblue", lwd = 2)
```

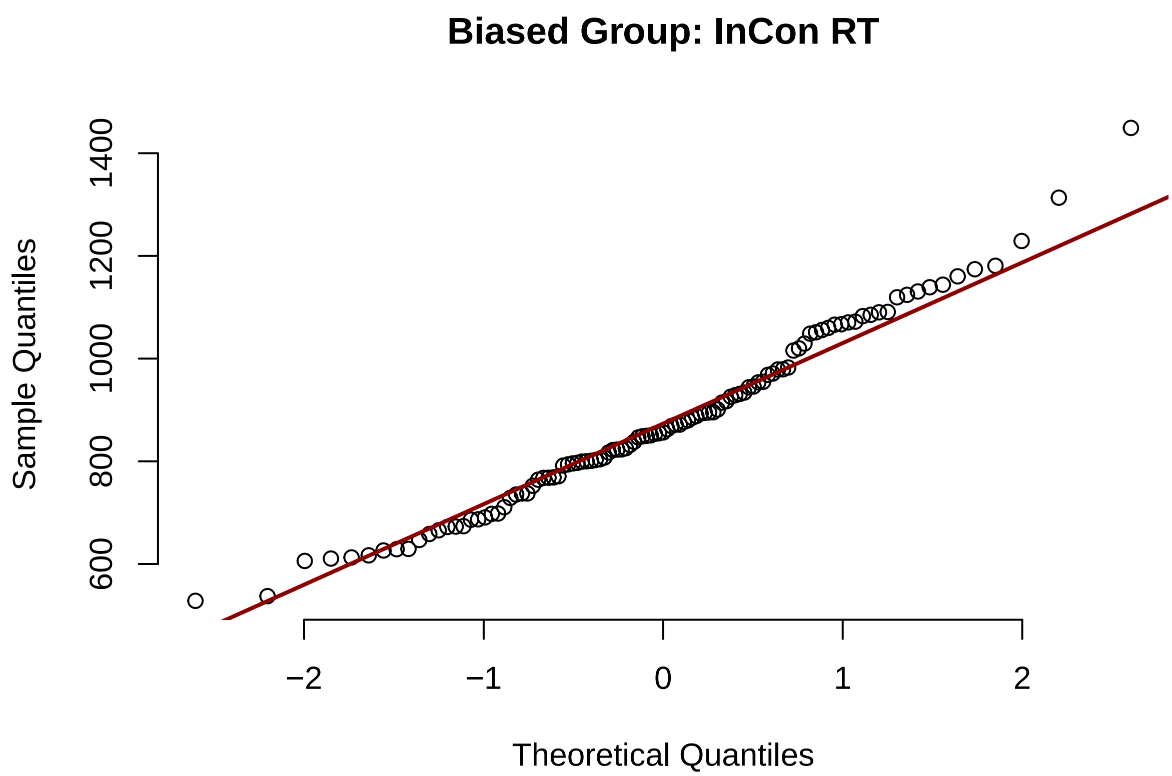



```r
shapiro.test(neutral$InCon_RT)
```

```
##
```

```
##  Shapiro-Wilk normality test
##
## data:  neutral$InCon_RT
## W = 0.75974, p-value = 8.108e-10
```

```
qqnorm(biased$InCon_RT, pch = 1, frame = FALSE, main = "Biased Group: InCon RT")
qqline(biased$InCon_RT, col = "darkred", lwd = 2)
```




```
shapiro.test(biased$InCon_RT) #assumption of normality violated
```

```
##
##  Shapiro-Wilk normality test
##
## data:  biased$InCon_RT
## W = 0.98125, p-value = 0.1282
```

```
# check if variance differs between groups
print(leveneTest(InCon_RT ~ Attention, data = combined_data))   #assumption of homogeneity not violated
```

```
## Warning in leveneTest.default(y = y, group = group, ...): group coerced to
## factor.
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value    Pr(>F)
```

```
## group   1  12.393 0.0005437 ***
##       183
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#do t-test
print(t.test(neutral$InCon_RT, biased$InCon_RT, var.equal = T))
```

```
##
##  Two Sample t-test
##
## data:  neutral$InCon_RT and biased$InCon_RT
## t = 6.0477, df = 183, p-value = 8.089e-09
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  181.1555 356.5928
## sample estimates:
## mean of x mean of y
## 1145.9454  877.0713
```

Removing Outliers InCon RT

```r
Incon_minus3SD <- mean(data$InCon_RT) - (3* sd(data$InCon_RT))
Incon_plus3SD <- mean(data$InCon_RT) + (3* sd(data$InCon_RT))

data <- data %>%
  mutate(InconOutlier = InCon_RT >= Incon_plus3SD)

outliers_subset = subset(data, data$InconOutlier == TRUE)


data_outliersremoved <- subset(data, data$InconOutlier == FALSE)
```

InCon Outliers Removed Descriptives

```r
data_outliersremoved %>%
  select(-c(Subject, Sex)) %>%
  psych::describe()
```

```
## Warning in FUN(newX[, i], ...): no non-missing arguments to min; returning Inf

## Warning in FUN(newX[, i], ...): no non-missing arguments to max; returning -Inf

##                    vars   n   mean     sd median trimmed    mad    min    max
## Age                   1 180  39.89  14.54  38.00   38.99  13.34  10.00  87.00
## IncorRespCount        2 182   1.64   2.79   1.00    1.10   1.48   0.00  19.00
## SymRespCount          3 182  18.15  17.80  12.00   17.00  16.31   0.00  48.00
```

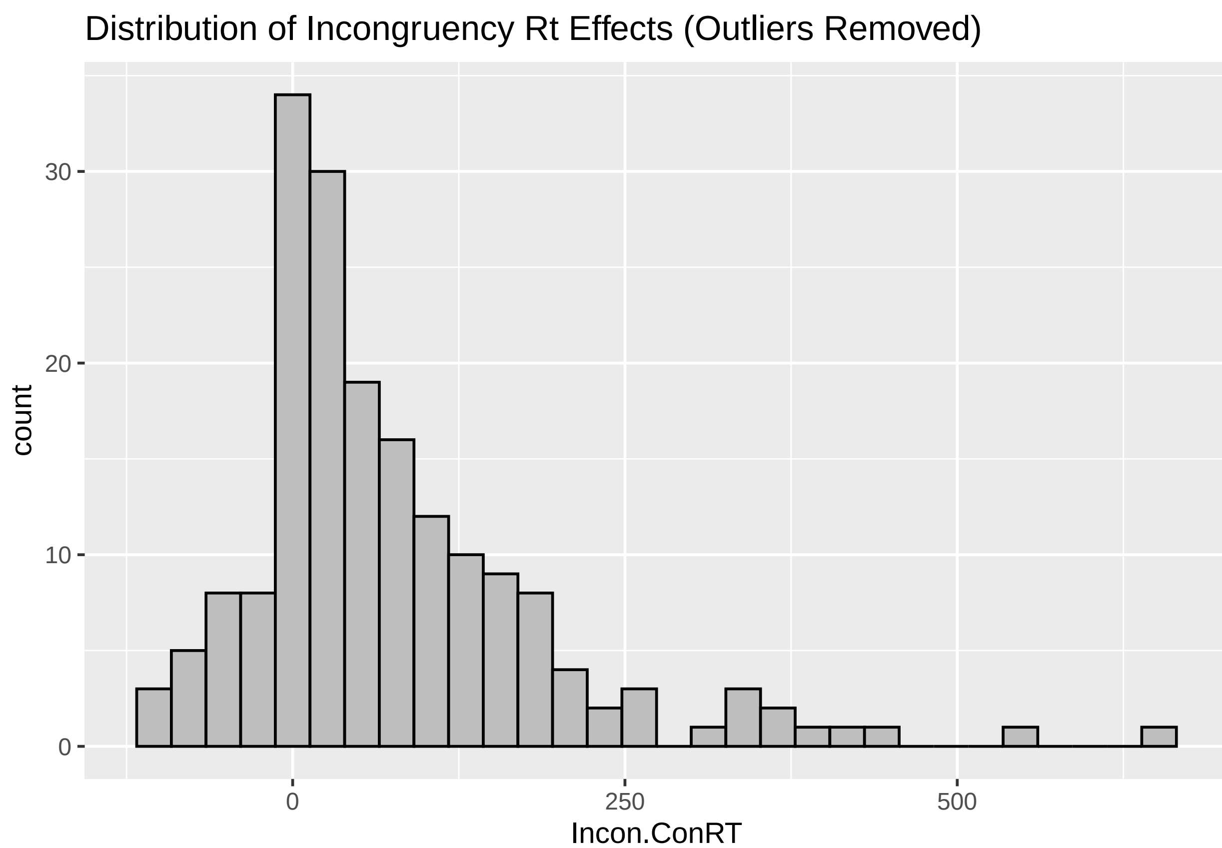
```
## TxtRespCount          4 182  28.20  18.07  34.50   29.10  18.53    0.00   48.00
## BiasScore             5 182   0.21   0.76   0.48    0.26   0.76   -1.00    1.00
## Con_RT                6 182 882.95 173.93 863.08  872.69 158.80  529.08 1579.72
## InCon_RT              7 182 957.95 230.28 921.78  939.41 209.53  528.29 1815.85
## Incon.ConRT           8 182  74.99 115.71  44.92   59.37  72.71 -116.09  640.06
## Attention*            9 182   1.40   0.49   1.00    1.38   0.00    1.00    2.00
## IPS*                 10 182   1.39   0.49   1.00    1.36   0.00    1.00    2.00
## IncongruencyEffect   11 182  74.99 115.71  44.92   59.37  72.71 -116.09  640.06
## absBiasScore         12 182   0.74   0.28   0.87    0.78   0.18    0.03    1.00
## InconOutlier         13 182    NaN     NA     NA     NaN     NA     Inf    -Inf
##                    range  skew kurtosis    se
## Age                77.00  0.58     0.24  1.08
## IncorRespCount     19.00  3.90    17.10  0.21
## SymRespCount       48.00  0.43    -1.51  1.32
## TxtRespCount       48.00 -0.33    -1.62  1.34
## BiasScore           2.00 -0.38    -1.57  0.06
## Con_RT           1050.64  0.65     0.67 12.89
## InCon_RT         1287.56  0.87     1.00 17.07
## Incon.ConRT       756.15  1.75     4.40  8.58
## Attention*          1.00  0.40    -1.85  0.04
## IPS*                1.00  0.45    -1.81  0.04
## IncongruencyEffect 756.15  1.75     4.40  8.58
## absBiasScore        0.97 -1.11    -0.04  0.02
## InconOutlier        -Inf    NA       NA    NA
```

#now that outliers are removed, incon effect mean went from 101.74 to 74.99.

Distributions of Incongruency Effect - Whole Group & Bias Split (InCon Outliers Removed)

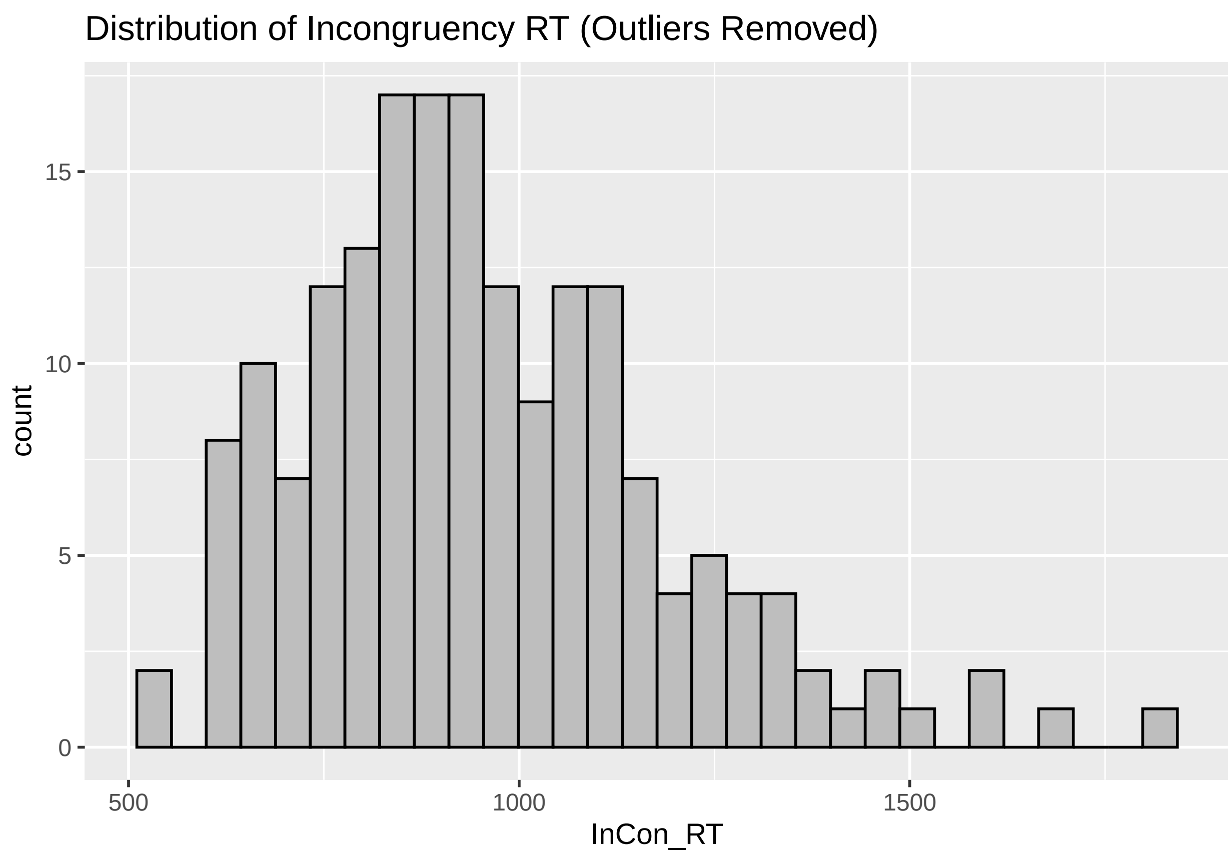
```
data_outliersremoved %>%
  ggplot(aes(x=Incon.ConRT)) +
  geom_histogram(fill = "grey", color = "black") +
  ggtitle("Distribution of Incongruency Rt Effects (Outliers Removed)")
```

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



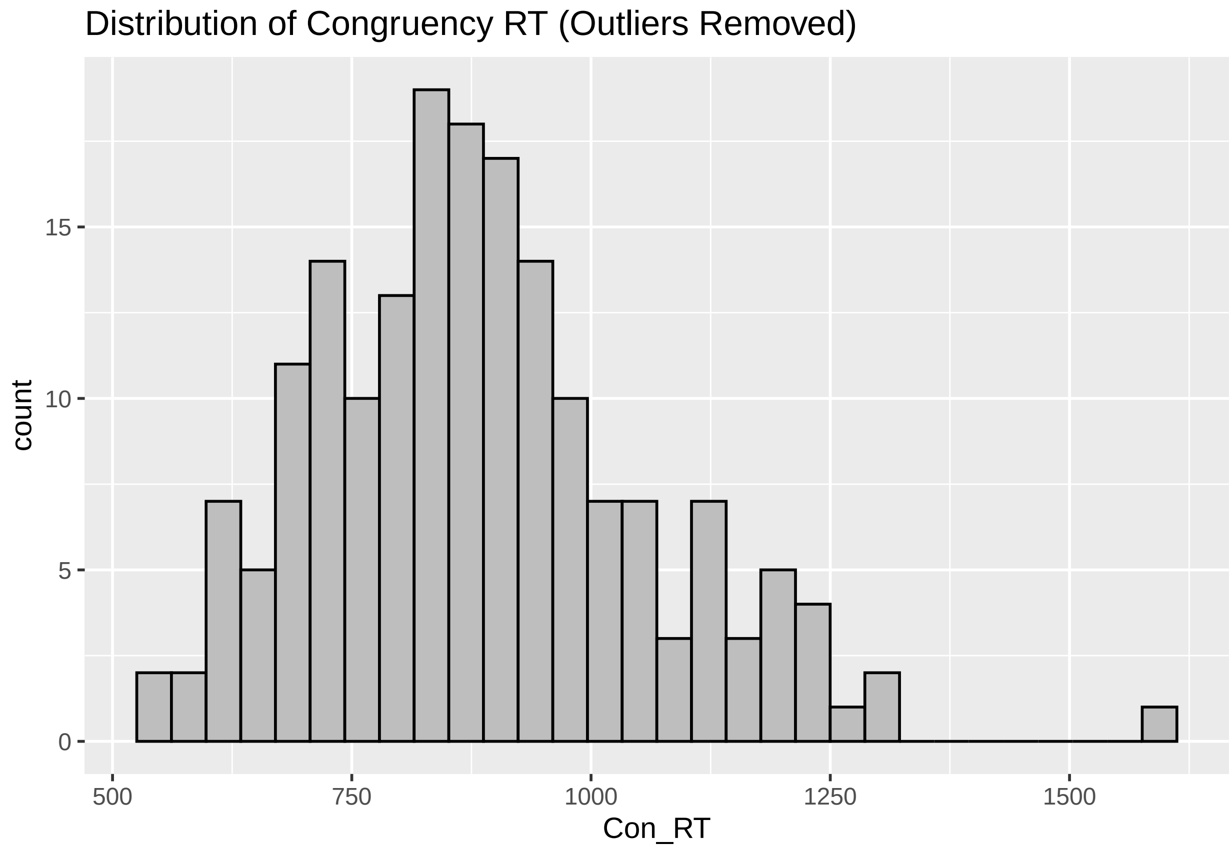

```r
data_outliersremoved %>%
  ggplot(aes(x=InCon_RT)) +
  geom_histogram(fill = "grey", color = "black") +
  ggtitle("Distribution of Incongruency RT (Outliers Removed)")
```

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

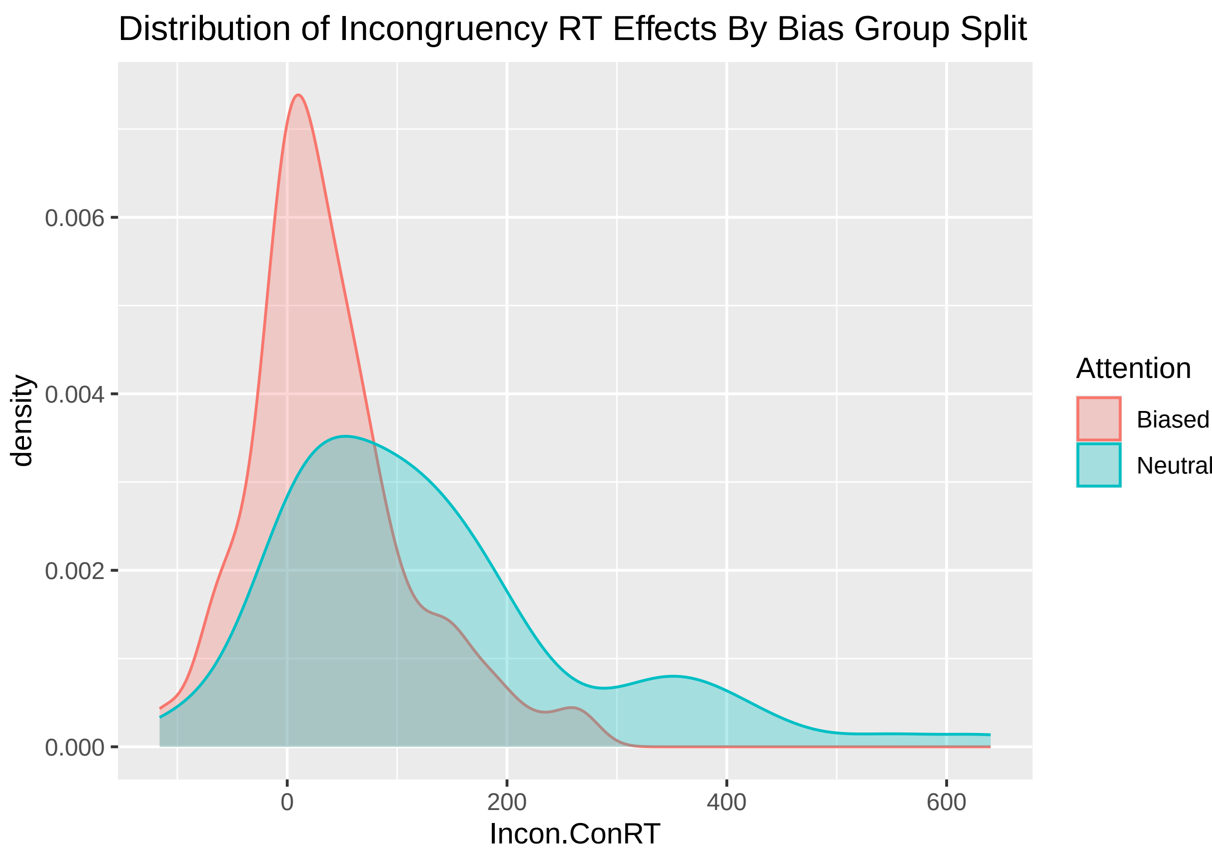



```
data_outliersremoved %>%
  ggplot(aes(x=Con_RT)) +
  geom_histogram(fill = "grey", color = "black") +
  ggtitle("Distribution of Congruency RT (Outliers Removed)")
```

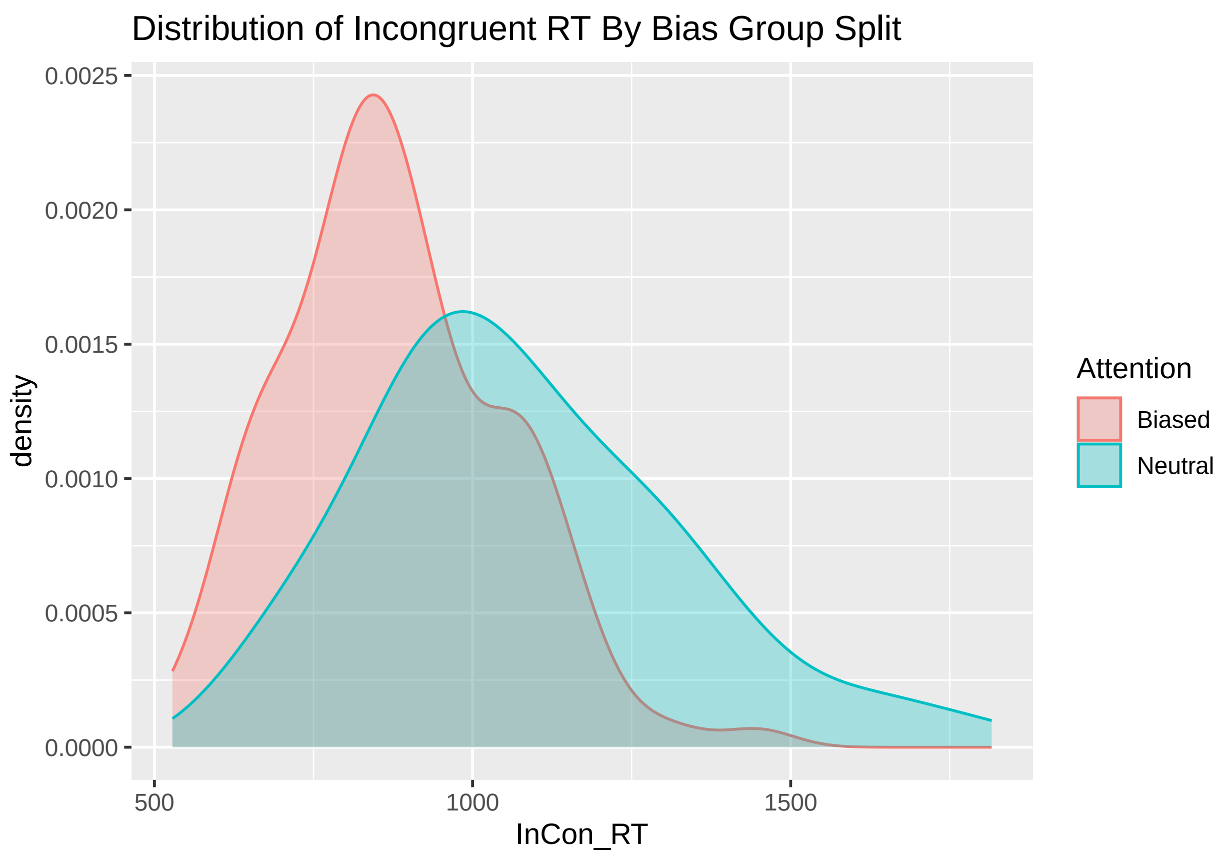
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



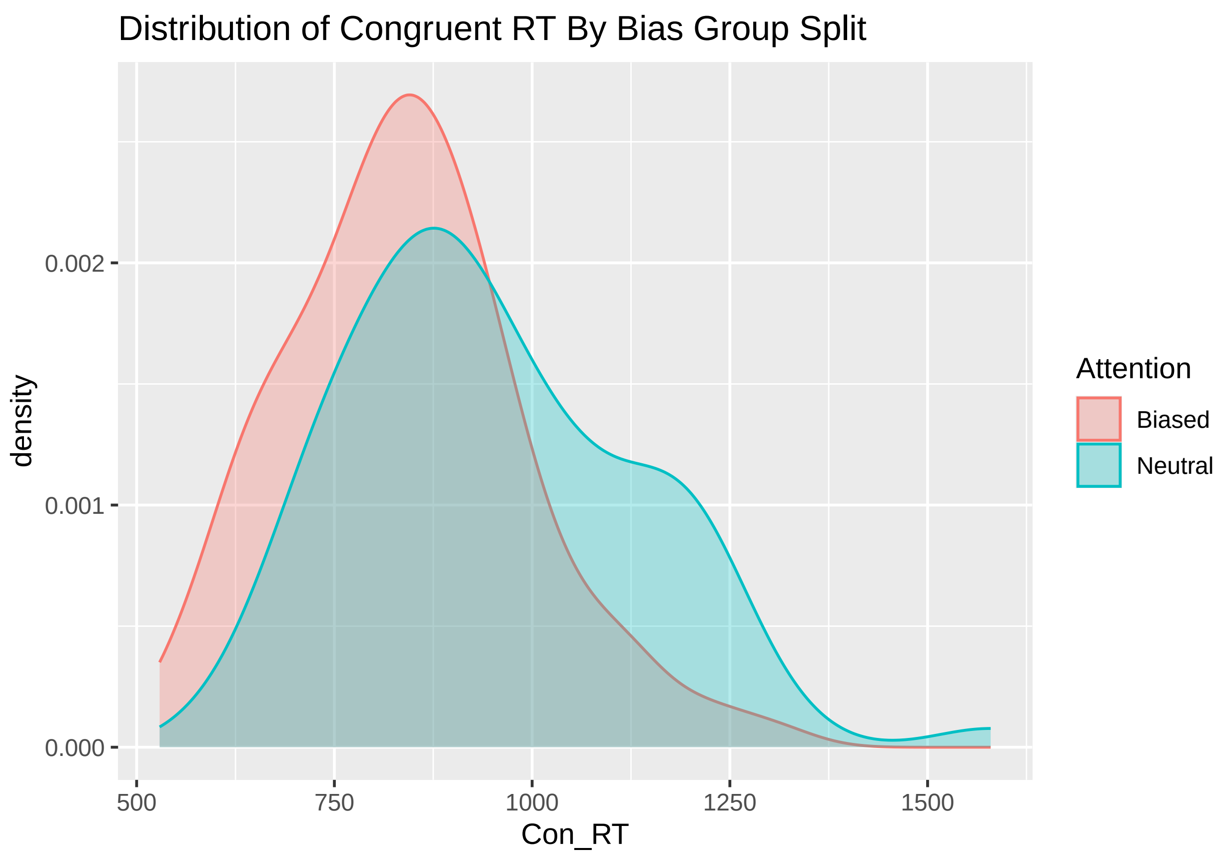

```
biased <- subset(data_outliersremoved, data_outliersremoved$BiasScore > 0.8 | data_outliersremoved$BiasS
neutral <- subset(data_outliersremoved, data_outliersremoved$BiasScore <= 0.8 & data_outliersremoved$Bi

data_grouped <- rbind(biased, neutral)

data_grouped %>%
  ggplot(aes(x = Incon.ConRT, fill = Attention, color = Attention))+
  geom_density(alpha = 0.3)+
  ggtitle("Distribution of Incongruency RT Effects By Bias Group Split")
```




```r
data_grouped %>%
  ggplot(aes(x = InCon_RT, fill = Attention, color = Attention))+
  geom_density(alpha = 0.3)+
  ggtitle("Distribution of Incongruent RT By Bias Group Split")
```




```r
data_grouped %>%
  ggplot(aes(x = Con_RT, fill = Attention, color = Attention))+
  geom_density(alpha = 0.3)+
  ggtitle("Distribution of Congruent RT By Bias Group Split")
```




```
# Incon effect dist: Most people seem to fall between 0 and 250 ms incongruency effect- incongruent tri
```

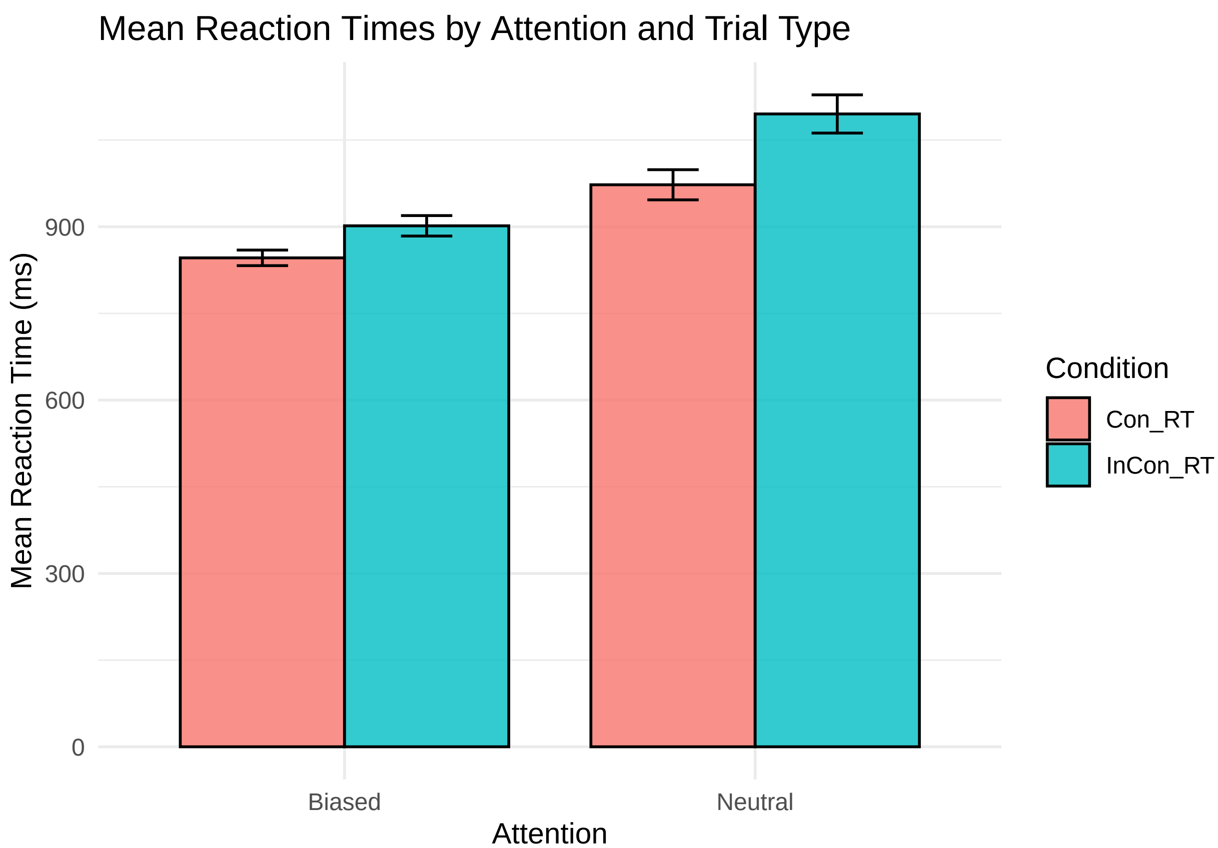
Reaction times by trial type and attention (InCon Outliers Removed)

```
data_outliersremoved$Attention <- ifelse(data_outliersremoved$BiasScore > 0.8 | data_outliersremoved$Bi
data_outliersremoved$IPS <- ifelse(data_outliersremoved$BiasScore > 0, "Verbal", "Visual")

# Reaction times
# Biased attender histograms and descriptive statistics
biased_data <- data_outliersremoved[data_outliersremoved$Attention == "Biased", ]
neutral_data <- data_outliersremoved[data_outliersremoved$Attention == "Neutral", ]

# Combine biased and neutral data
data_outliersremoved$Attention <- ifelse(data_outliersremoved$BiasScore > 0.8 | data_outliersremoved$Bi
biased_data <- data_outliersremoved[data_outliersremoved$Attention == "Biased", ]
combined_data <- rbind(biased_data, neutral_data)

# Calculate means by attention and trial type
library(dplyr)
library(tidyr)
library(ggplot2)

means <- combined_data %>%
  group_by(Attention) %>%
```

```r
  summarise(Con_RT = mean(Con_RT, na.rm = TRUE), InCon_RT = mean(InCon_RT, na.rm = TRUE)) %>%
  pivot_longer(cols = c(Con_RT, InCon_RT), names_to = "Trial_Type", values_to = "RT")

# Order factor levels for better plotting
means$Attention <- factor(means$Attention, levels = unique(means$Attention))

combined_long <- combined_data %>%
  select(Attention, InCon_RT, Con_RT) %>%
  pivot_longer(cols = c(InCon_RT, Con_RT), values_to = "RT", names_to = "Condition")

# Create error bars
se_sum <- combined_long %>%
  group_by(Attention, Condition) %>%
  summarise(
    sd = sd(RT, na.rm = TRUE),
    n = n(),
    mean = mean(RT, na.rm = TRUE)
  ) %>%
  mutate(se = sd/sqrt(n))
```

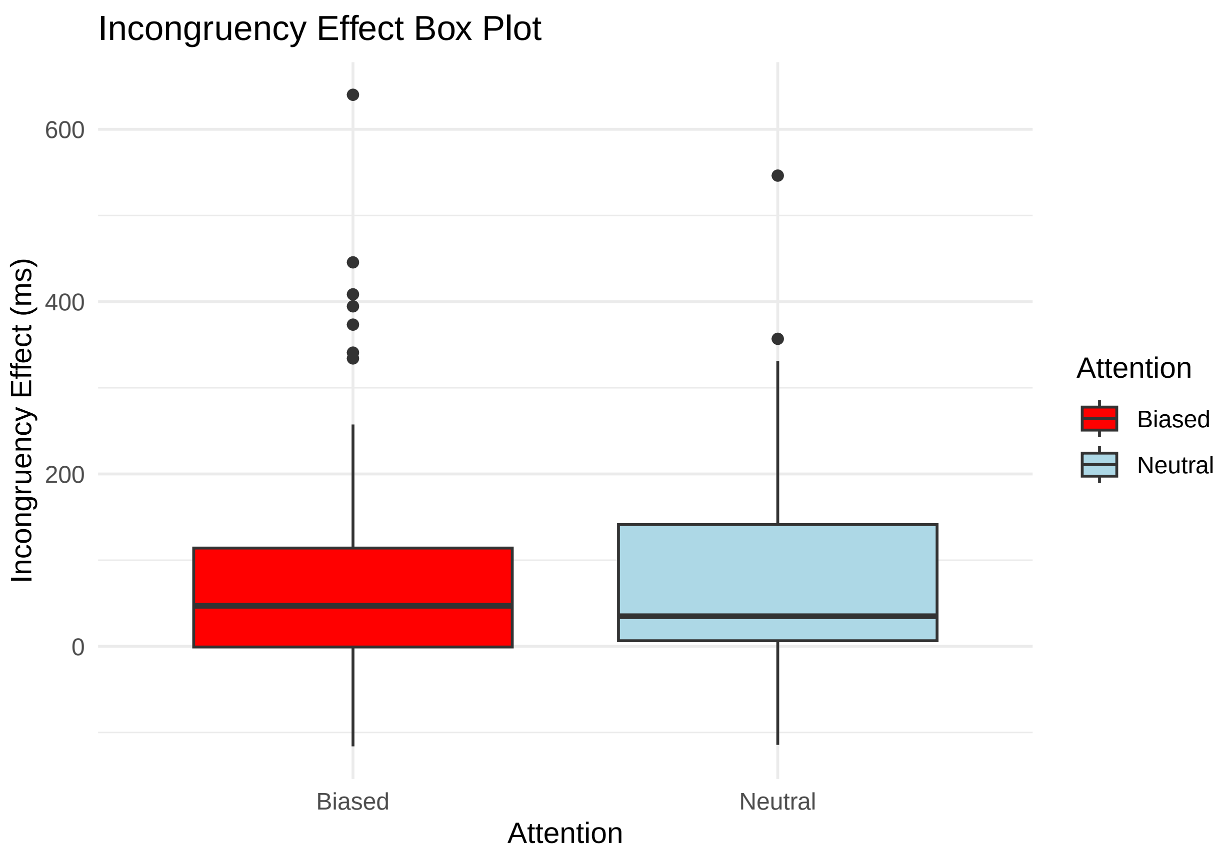
```
## `summarise()` has grouped output by 'Attention'. You can override using the
## `.groups` argument.
```

```r
# Create bar plot
ggplot(se_sum, aes(x = Attention, y = mean, fill = Condition)) +
  geom_bar(position = position_dodge(0.8), stat = "identity", color = "black", size = 0.5, width = 0.8,
  geom_errorbar(aes(ymin = mean - se, ymax = mean + se), position = position_dodge(0.8), width = 0.25,
  labs(title = "Mean Reaction Times by Attention and Trial Type",
       x = "Attention",
       y = "Mean Reaction Time (ms)") +
  theme_minimal()
```



Incongruency Effect Box Plot (InCon Outliers Removed)

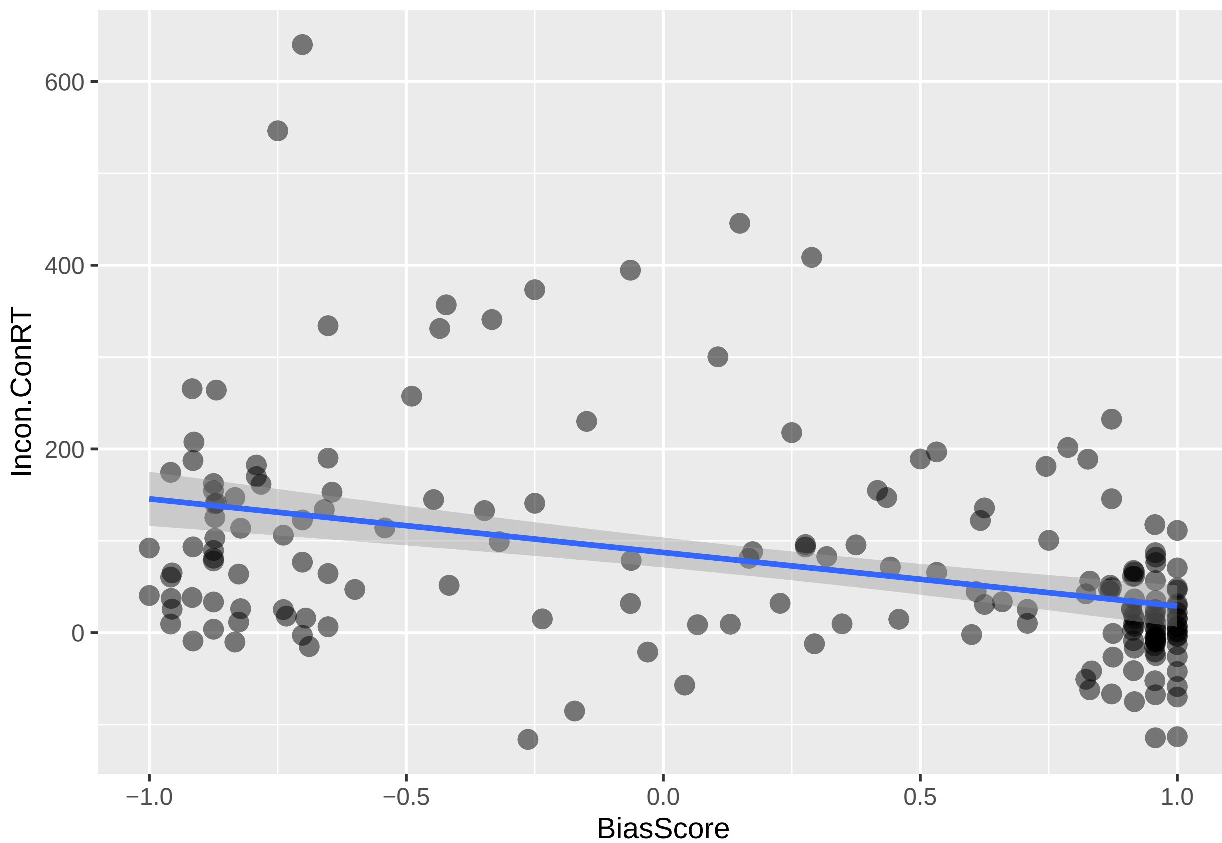
```
biased_data<- data_outliersremoved[data_outliersremoved$Attention == "Biased", ]
neutral_data<- data_outliersremoved[data_outliersremoved$Attention == "Neutral", ]

combined_data <- rbind(biased_data, neutral_data)
Incongruency_Effect_Data <- data_outliersremoved$IncongruencyEffect

# Create box plot
ggplot(combined_data, aes(x = Attention, y = Incongruency_Effect_Data, fill = Attention)) +
  geom_boxplot() +
  labs(
    title = "Incongruency Effect Box Plot",
    x = "Attention",
    y = "Incongruency Effect (ms)"
  ) +
  scale_fill_manual(values = c("red", "lightblue")) +  # Color for biased and neutral data
  theme_minimal()
```



Bias Score and Incongruency Effect Correlation (InCon Outliers Removed)

```
data_outliersremoved %>%
  ggplot(aes(x = BiasScore, y = Incon.ConRT)) +
  geom_point(size = 3, alpha = 0.5, fill = "grey") +
  geom_smooth(method = "lm")
```

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




```r
cor.test(data_outliersremoved$BiasScore, data_outliersremoved$Incon.ConRT)
```

```
##
##  Pearson's product-moment correlation
##
## data:  data_outliersremoved$BiasScore and data_outliersremoved$Incon.ConRT
## t = -5.5861, df = 180, p-value = 8.46e-08
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.5017781 -0.2530704
## sample estimates:
##        cor
## -0.3843768
```

t-test (InCon Outliers Removed)

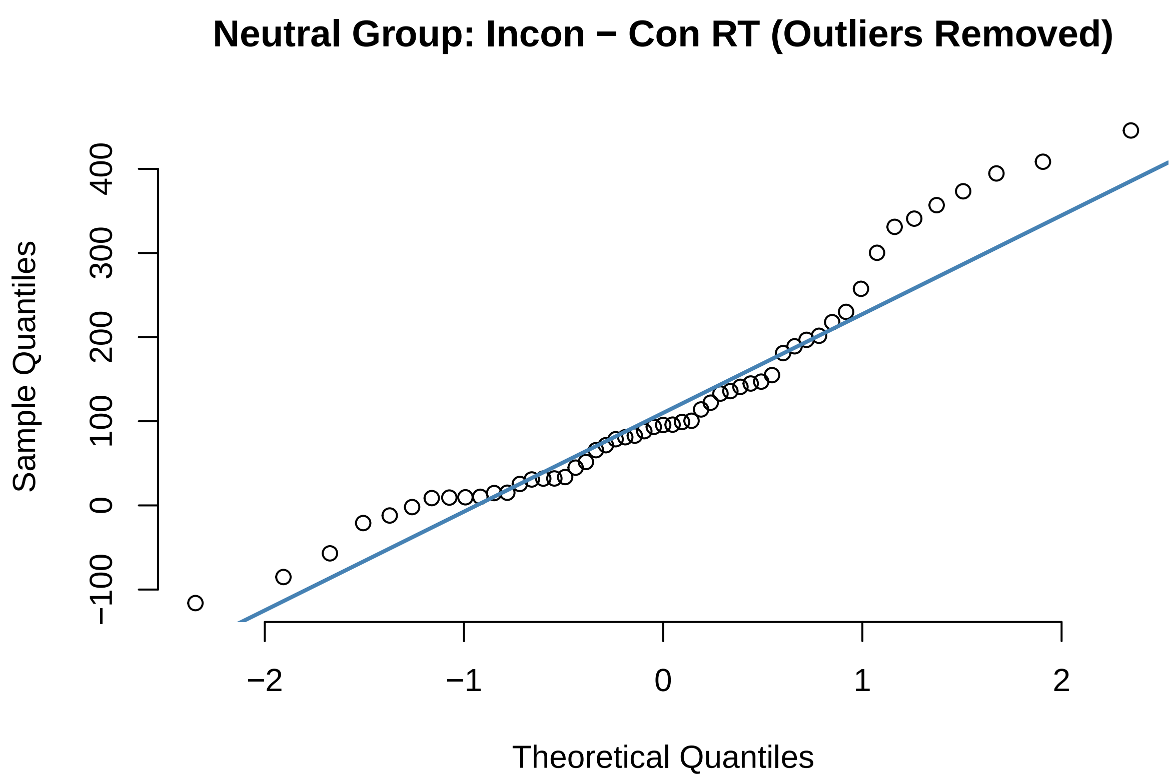
```r
t.test(Incon.ConRT~Attention, data=data_outliersremoved)
```

```
##
##  Welch Two Sample t-test
##
## data:  Incon.ConRT by Attention
## t = -3.3291, df = 79.807, p-value = 0.00132
```

```
## alternative hypothesis: true difference in means between group Biased and group Neutral is not equal
## 95 percent confidence interval:
##  -107.18506  -26.98088
## sample estimates:
##  mean in group Biased mean in group Neutral
##              55.45634             122.53931
```

T-test Code - Removing Incon Outliers

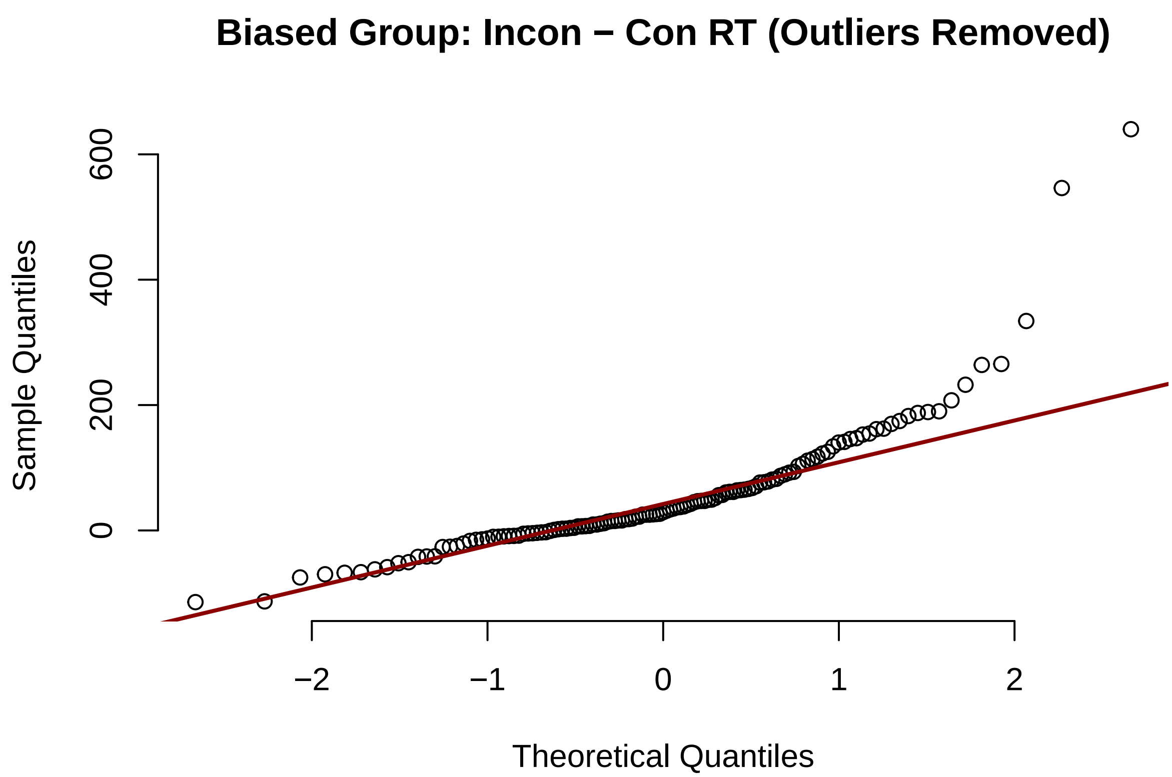
```
biased_data<- data_outliersremoved[data_outliersremoved$Attention == "Biased", ]
neutral_data<- data_outliersremoved[data_outliersremoved$Attention == "Neutral", ]

qqnorm(neutral_data$Incon.ConRT, pch = 1, frame = FALSE, main = "Neutral Group: Incon - Con RT (Outliers
qqline(neutral_data$Incon.ConRT, col = "steelblue", lwd = 2)
```




```
shapiro.test(neutral_data$Incon.ConRT)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  neutral_data$Incon.ConRT
## W = 0.9365, p-value = 0.007351
```

```r
# Then test normality assumption in Biased Group
qqnorm(biased_data$Incon.ConRT, pch = 1, frame = FALSE, main = "Biased Group: Incon - Con RT (Outliers R
qqline(biased_data$Incon.ConRT, col = "darkred", lwd = 2)
```




```r
shapiro.test(biased_data$Incon.ConRT)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  biased_data$Incon.ConRT
## W = 0.80037, p-value = 5.813e-12
```

```r
# Check that the variance does not differ between groups
# Perform Levene's Test
print(leveneTest(Incon.ConRT ~ Attention, data = data_outliersremoved))
```

```
## Warning in leveneTest.default(y = y, group = group, ...): group coerced to
## factor.

## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value  Pr(>F)
## group   1  5.5793 0.01924 *
##       180
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Conduct t-test with equal variance assumption
print(t.test(neutral_data$Incon.ConRT, biased_data$Incon.ConRT, var.equal = FALSE))
```

```
##
##  Welch Two Sample t-test
##
## data:  neutral_data$Incon.ConRT and biased_data$Incon.ConRT
## t = 3.3291, df = 79.807, p-value = 0.00132
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##   26.98088 107.18506
## sample estimates:
## mean of x mean of y
## 122.53931  55.45634
```

Removing outliers- Congruent RT

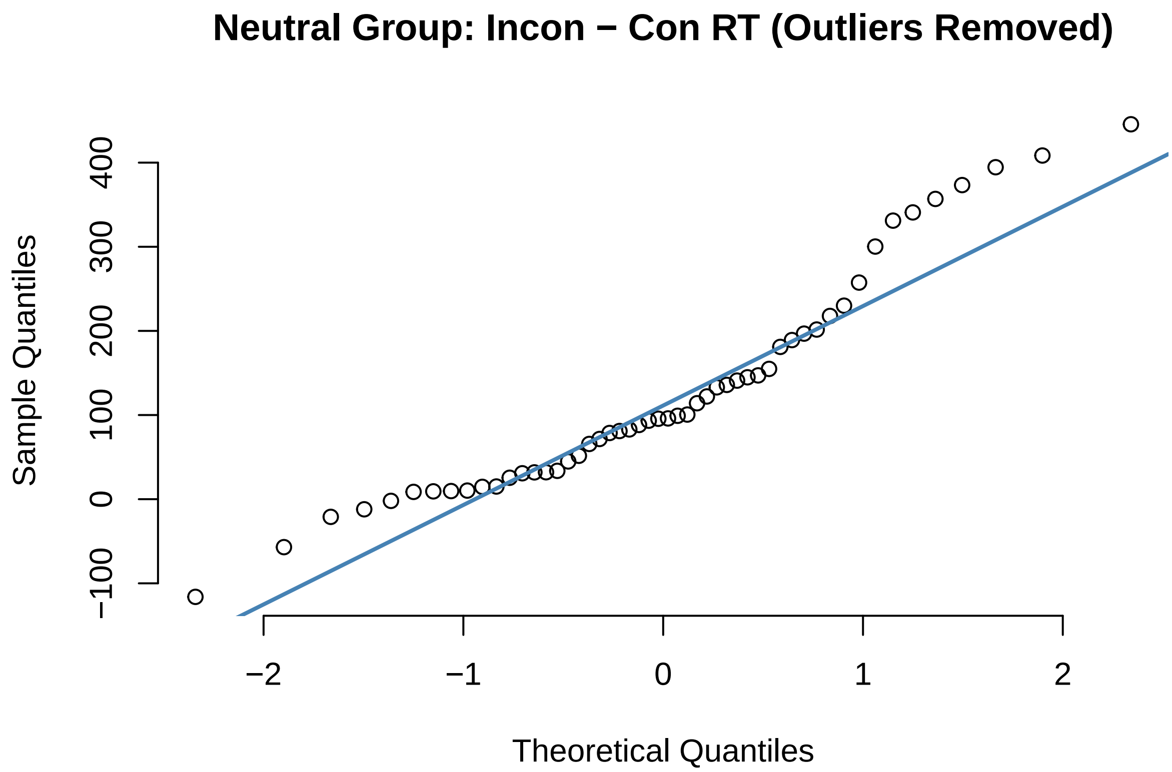
```r
Con_minus3SD <- mean(data_outliersremoved$Con_RT) - (3* sd(data_outliersremoved$Con_RT))
Con_plus3SD <- mean(data_outliersremoved$Con_RT) + (3* sd(data_outliersremoved$Con_RT))

data_outliersremoved <- data_outliersremoved %>%
  mutate(ConOutlier = Con_RT >= Con_plus3SD)

subset(data_outliersremoved, ConOutlier == TRUE)
```

```
##      Subject Age Sex IncorRespCount SymRespCount TxtRespCount  BiasScore
## 100 9449a552  57   2             19           17           12 -0.1724138
##       Con_RT InCon_RT Incon.ConRT Attention    IPS IncongruencyEffect
## 100 1579.715 1494.562   -85.15278   Neutral Visual          -85.15278
##     absBiasScore InconOutlier ConOutlier
## 100    0.1724138        FALSE       TRUE
```

```r
data_final <- subset(data_outliersremoved, ConOutlier == FALSE)
```

T-test with All Outliers Removed

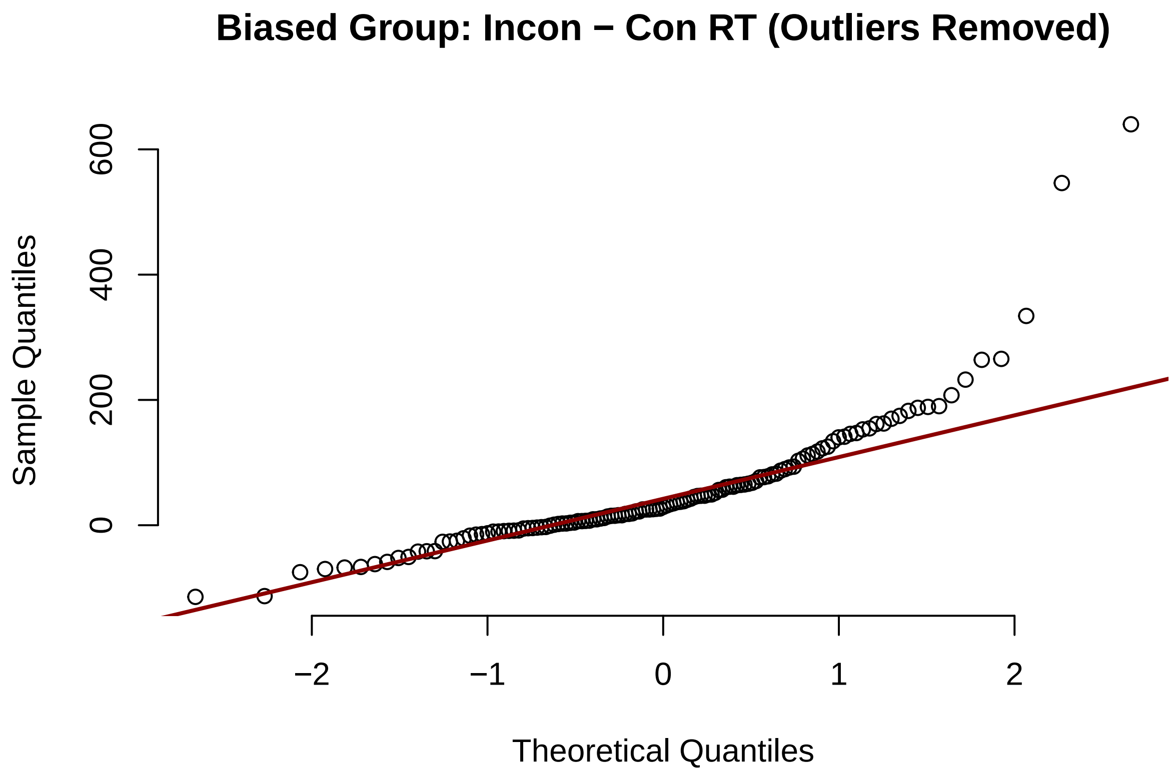
```r
biased_data <- data_final[data_final$Attention == "Biased", ]
neutral_data <- data_final[data_final$Attention == "Neutral", ]

qqnorm(neutral_data$Incon.ConRT, pch = 1, frame = FALSE, main = "Neutral Group: Incon - Con RT (Outliers
qqline(neutral_data$Incon.ConRT, col = "steelblue", lwd = 2)
```




```
shapiro.test(neutral_data$Incon.ConRT)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  neutral_data$Incon.ConRT
## W = 0.92908, p-value = 0.004134
```

```
# Then test normality assumption in Biased Group
qqnorm(biased_data$Incon.ConRT, pch = 1, frame = FALSE, main = "Biased Group: Incon - Con RT (Outliers R
qqline(biased_data$Incon.ConRT, col = "darkred", lwd = 2)
```




```r
shapiro.test(biased_data$Incon.ConRT)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  biased_data$Incon.ConRT
## W = 0.80037, p-value = 5.813e-12
```

```r
# Check that the variance does not differ between groups
# Perform Levene's Test
print(leveneTest(Incon.ConRT ~ Attention, data = data_final))
```

```
## Warning in leveneTest.default(y = y, group = group, ...): group coerced to
## factor.
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value  Pr(>F)
## group   1  4.9887 0.02675 *
##       179
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Conduct t-test with equal variance assumption
print(t.test(neutral_data$Incon.ConRT, biased_data$Incon.ConRT, var.equal = FALSE))
```

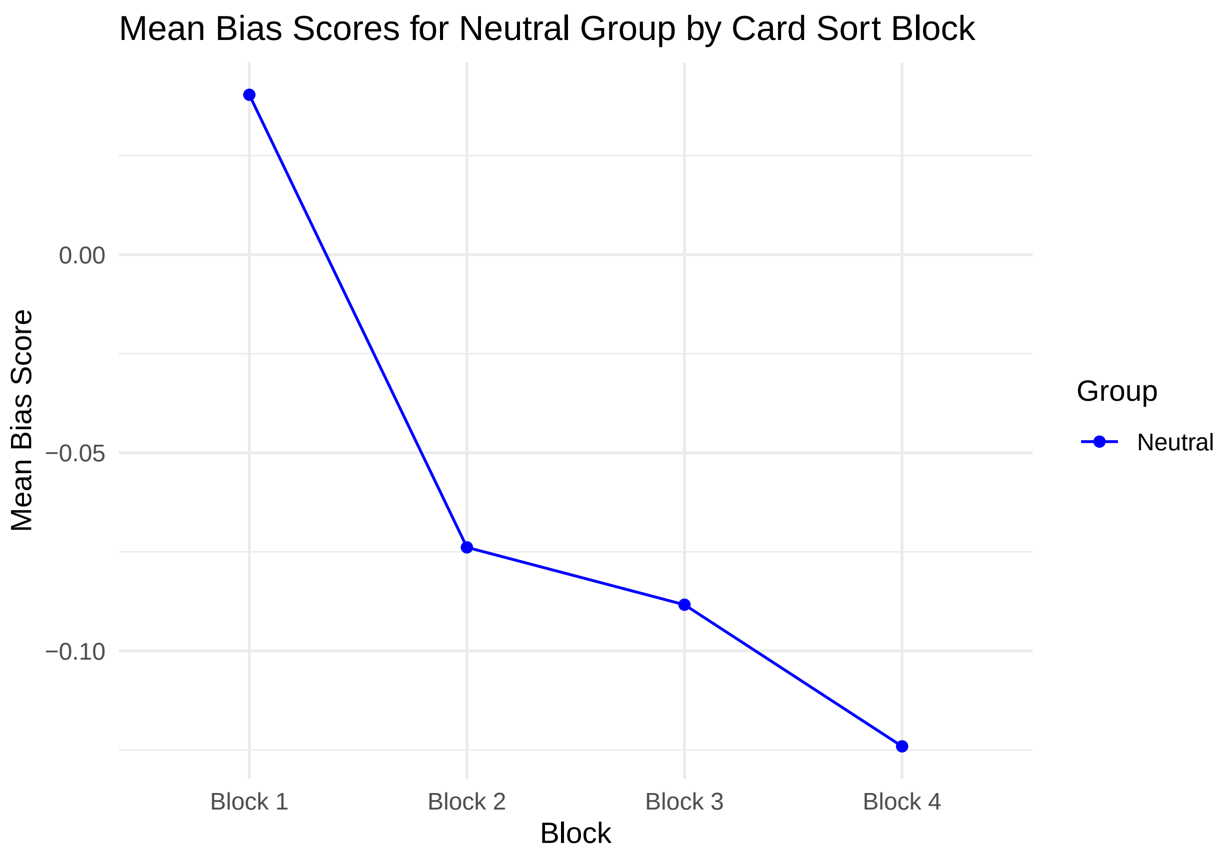
```
##
##  Welch Two Sample t-test
##
## data:  neutral_data$Incon.ConRT and biased_data$Incon.ConRT
## t = 3.5444, df = 78.651, p-value = 0.000666
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##   31.15857 110.99552
## sample estimates:
## mean of x mean of y
## 126.53339  55.45634
```

splitting blocks

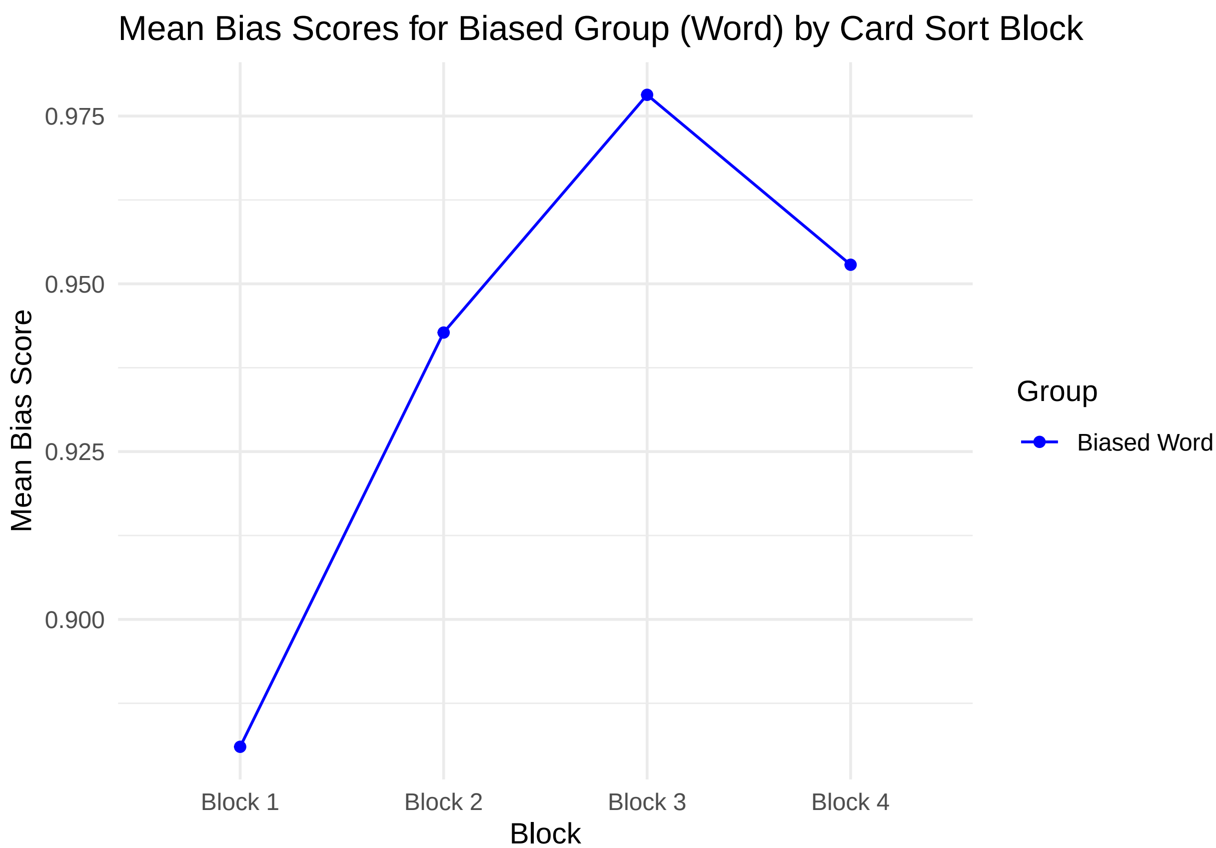
block data

```r
block_data <- read.csv("CardSort Data.csv")

for_calc = data.frame(Subject = character(), block1_mean = numeric(), block2_mean = numeric(), block3_me

for (subject in unique(data$Subject)){
  subject_cols = block_data[block_data$SubjectNumber == subject, ]
  subject_block1 <- subject_cols[subject_cols$"Block" == "CardSort_Block1",]
  subject_block2 <- subject_cols[subject_cols$"Block" == "CardSort_Block2",]
  subject_block3 <- subject_cols[subject_cols$"Block" == "CardSort_Block3",]
  subject_block4 <- subject_cols[subject_cols$"Block" == "CardSort_Block4",]

  block1_mean <- mean(subject_block1$RT)
  block2_mean <- mean(subject_block2$RT)
  block3_mean <- mean(subject_block3$RT)
  block4_mean <- mean(subject_block4$RT)

  block1_incon_rows <- subset(subject_block1, Status == 2)
  block1_word <- max(subject_block1$TxtRespCount)
  block1_pic <- max(subject_block1$SymRespCount)
  block1_correct <- length(block1_incon_rows) - max(subject_block1$IncorrRespCount)
  block1_bias <- (block1_word - block1_pic) / block1_correct

  block2_incon_rows <- subset(subject_block2, Status == 2)
  block2_word <- max(subject_block2$TxtRespCount)-max(subject_block1$TxtRespCount)
  block2_pic <- max(subject_block2$SymRespCount)- max(subject_block1$SymRespCount)
  block2_correct <- length(block2_incon_rows) - (max(subject_block2$IncorrRespCount)-max(subject_block1
  block2_bias <- (block2_word - block2_pic) / block2_correct

  block3_incon_rows <- subset(subject_block3, Status == 2)
  block3_word <- max(subject_block3$TxtRespCount)-max(subject_block2$TxtRespCount)
  block3_pic <- max(subject_block3$SymRespCount)-max(subject_block2$SymRespCount)
```

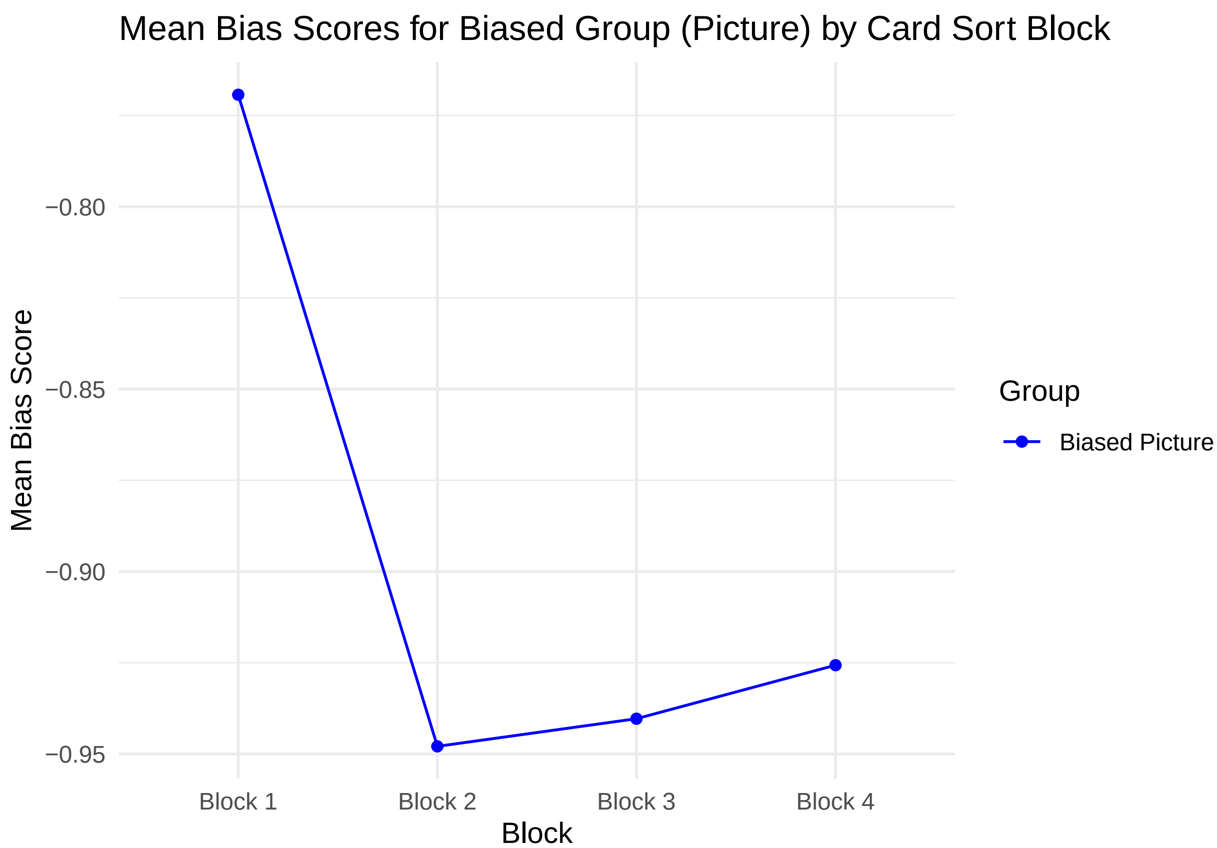
```
  block3_correct <- length(block3_incon_rows) - (max(subject_block3$IncorrRespCount)-max(subject_block2$
  block3_bias <- (block3_word - block3_pic) / block3_correct

  block4_incon_rows <- subset(subject_block4, Status == 2)
  block4_word <- max(subject_block4$TxtRespCount)-max(subject_block3$TxtRespCount)
  block4_pic <- max(subject_block4$SymRespCount)-max(subject_block3$SymRespCount)
  block4_correct <- length(block4_incon_rows) - (max(subject_block4$IncorrRespCount)-max(subject_block3$
  block4_bias <- (block4_word - block4_pic) / block4_correct

  new_row1 <- data.frame(Subject = subject, block1_mean = block1_mean, block2_mean =  block2_mean, block

  for_calc<- rbind(for_calc, new_row1)
}

data <- cbind(data, for_calc)


# mean rts for each block biased v neutral
biased_word <- subset(data, data$BiasScore > 0.8)
biased_picture <- subset(data, data$BiasScore < -0.8)
biased <- subset(data, data$BiasScore > 0.8 | data$BiasScore < -0.8)
neutral <- subset(data, data$BiasScore <= 0.8 & data$BiasScore >= -0.8)

biased_means <- colMeans(biased[, c("block1_bias", "block2_bias", "block3_bias", "block4_bias")], na.rm
# Calculate means for neutral group
neutral_means <- colMeans(neutral[, c("block1_bias", "block2_bias", "block3_bias", "block4_bias")], na.
biased_word_means <- colMeans(biased_word[, c("block1_bias", "block2_bias", "block3_bias", "block4_bias
biased_picture_means <- colMeans(biased_picture[, c("block1_bias", "block2_bias", "block3_bias", "block

means_df <- data.frame(
  Block = rep(c("Block 1", "Block 2", "Block 3", "Block 4"), 2),
  MeanBias = c(biased_means, neutral_means),
  Group = rep(c("Biased", "Neutral"), each = 4)
)

# subset of neutral
neutral_means_df <- subset(means_df, Group == "Neutral")

# Plotting neutral blocks from means_df
ggplot(neutral_means_df, aes(x = Block, y = MeanBias, group = Group, color = Group)) +
  geom_line() +
  geom_point() +
  labs(title = "Mean Bias Scores for Neutral Group by Card Sort Block", x = "Block", y = "Mean Bias Sco
  scale_color_manual(values = c("Neutral" = "blue")) +
  theme_minimal()
```



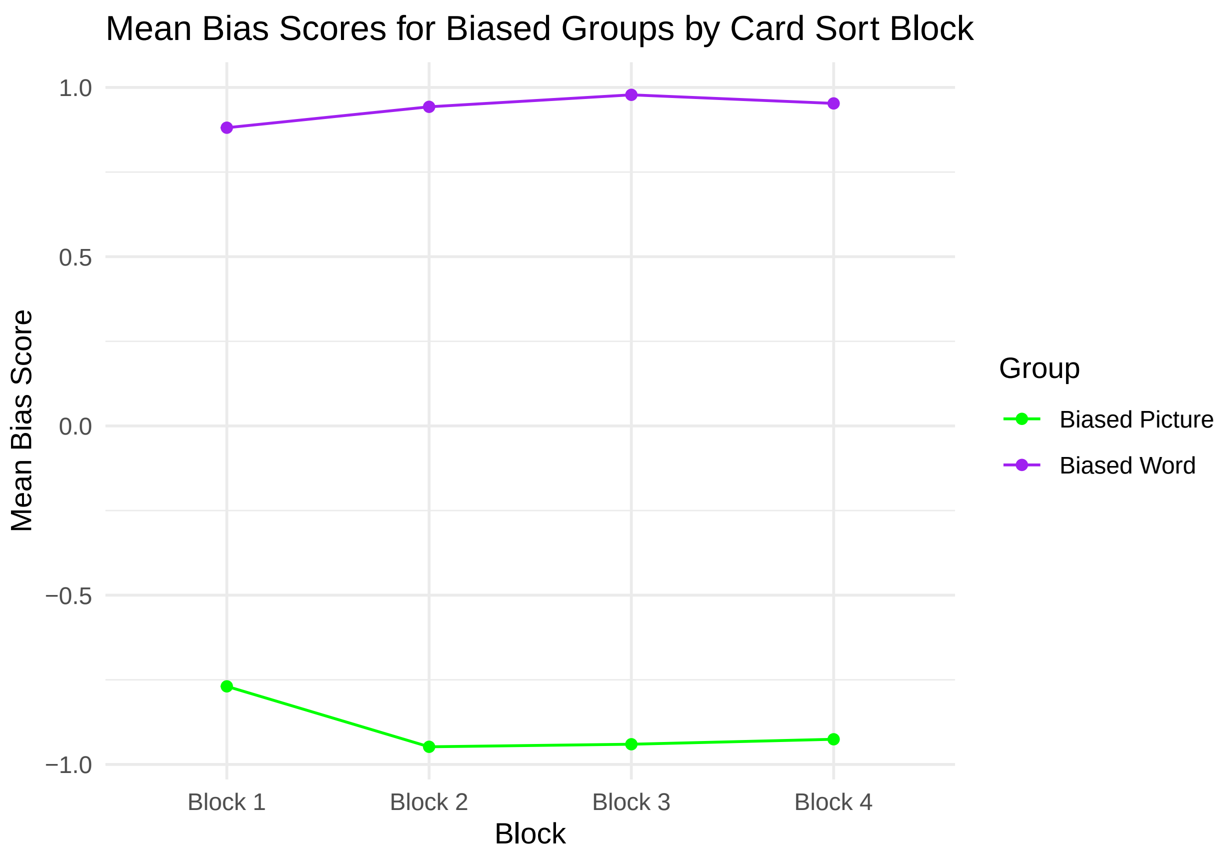

```r
# Creating means_df for two biased groups and one neutral group
biased_means_df <- data.frame(
  Block = rep(c("Block 1", "Block 2", "Block 3", "Block 4"), 3),
  MeanBias = c(biased_word_means, biased_picture_means, neutral_means),
  Group = rep(c("Biased Word", "Biased Picture", "Neutral"), each = 4)
)

#subset of word biased (positive)
biased_word_means_df <- subset(biased_means_df, Group == "Biased Word")
biased_picture_means_df <- subset(biased_means_df, Group == "Biased Picture")

ggplot(biased_word_means_df, aes(x = Block, y = MeanBias, group = Group, color = Group)) +
  geom_line() +
  geom_point() +
  labs(title = "Mean Bias Scores for Biased Group (Word) by Card Sort Block", x = "Block", y = "Mean Bia
  scale_color_manual(values = c("Biased Word" = "blue")) +
  theme_minimal()
```




```r
#subset of picture biased (negative)
ggplot(biased_picture_means_df, aes(x = Block, y = MeanBias, group = Group, color = Group)) +
  geom_line() +
  geom_point() +
  labs(title = "Mean Bias Scores for Biased Group (Picture) by Card Sort Block", x = "Block", y = "Mean
  scale_color_manual(values = c("Biased Picture" = "blue")) +
  theme_minimal()
```

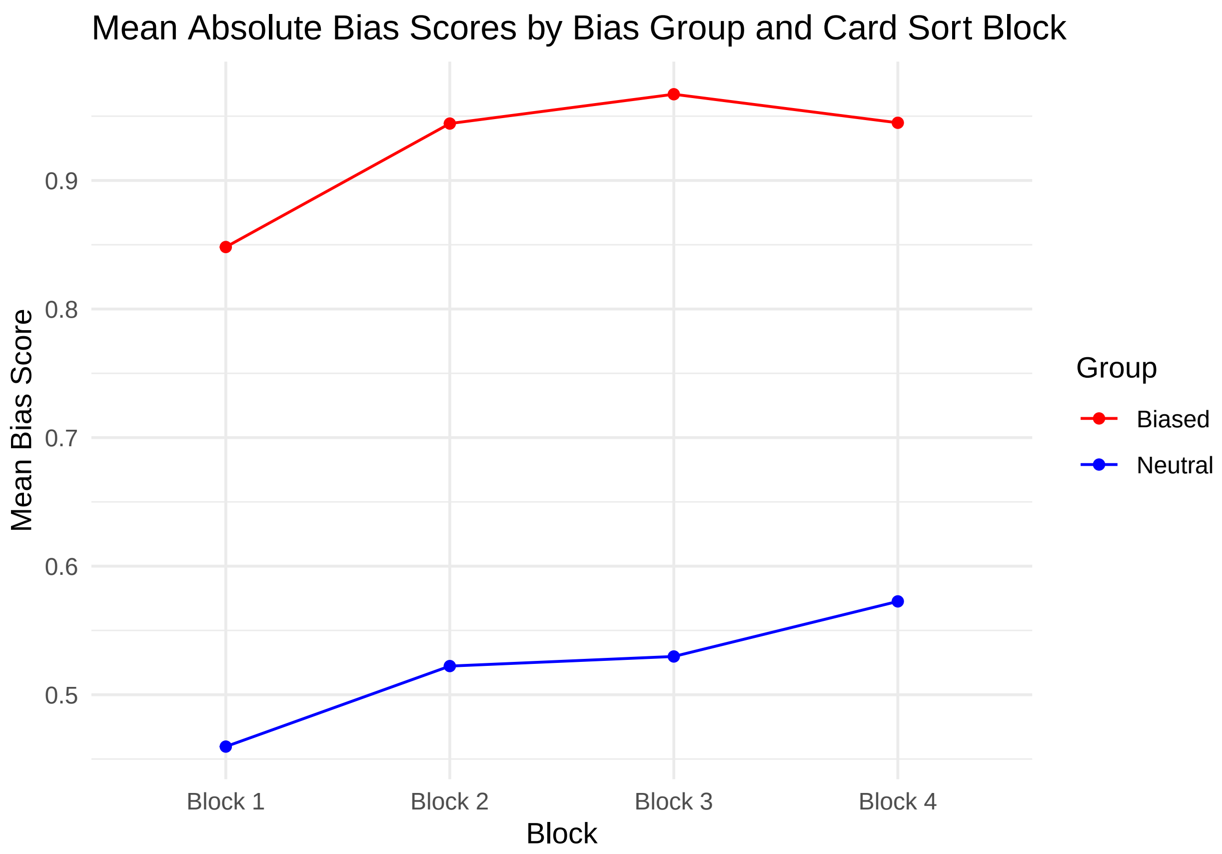



```r
ggplot() +
  geom_line(data = biased_word_means_df, aes(x = Block, y = MeanBias, group = Group, color = "Biased Wo
  geom_point(data = biased_word_means_df, aes(x = Block, y = MeanBias, color = "Biased Word")) +
  geom_line(data = biased_picture_means_df, aes(x = Block, y = MeanBias, group = Group, color = "Biased
  geom_point(data = biased_picture_means_df, aes(x = Block, y = MeanBias, color = "Biased Picture")) +
  labs(title = "Mean Bias Scores for Biased Groups by Card Sort Block", x = "Block", y = "Mean Bias Sco
  scale_color_manual(name = "Group", values = c("Biased Word" = "purple", "Biased Picture" = "green")) 
  theme_minimal()
```

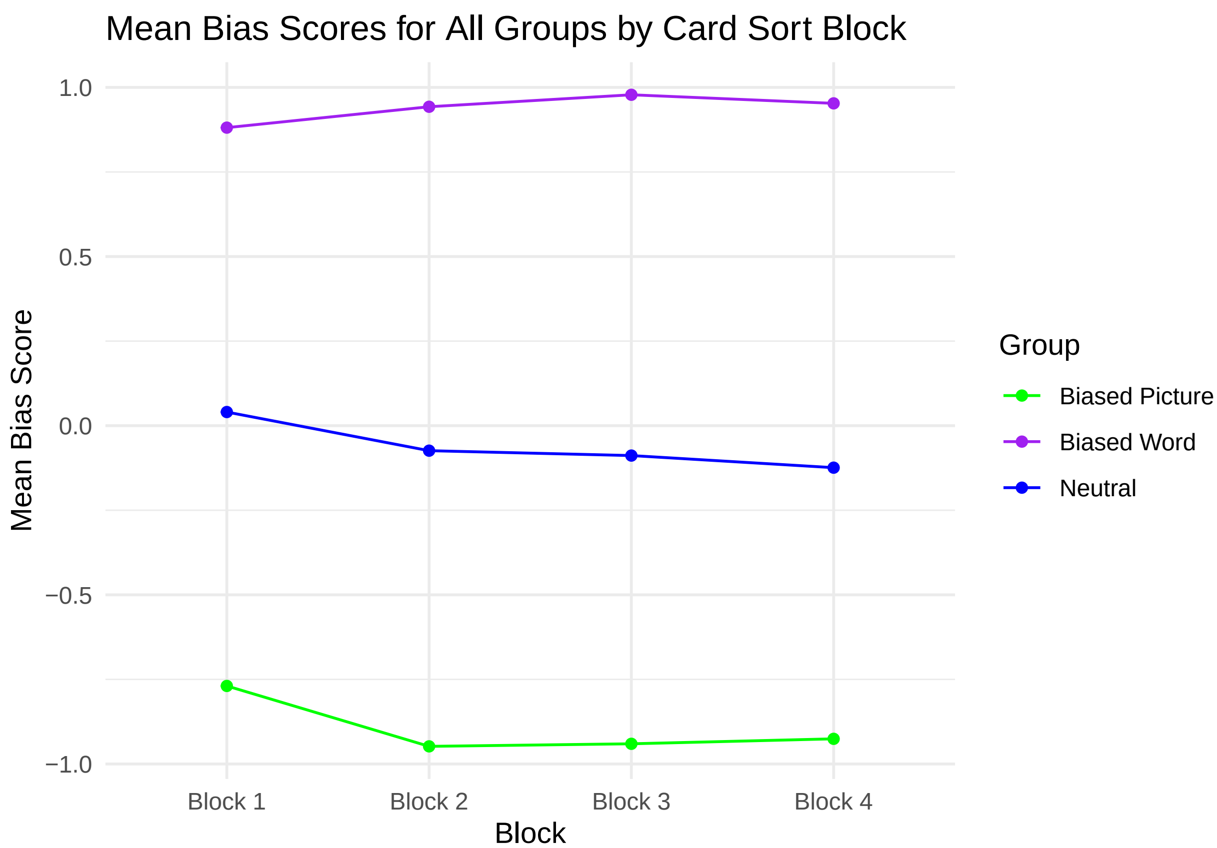



```
biased$block1_bias <- abs(biased$block1_bias)
biased$block2_bias <- abs(biased$block2_bias)
biased$block3_bias <- abs(biased$block3_bias)
biased$block4_bias <- abs(biased$block4_bias)

neutral$block1_bias <- abs(neutral$block1_bias)
neutral$block2_bias <- abs(neutral$block2_bias)
neutral$block3_bias <- abs(neutral$block3_bias)
neutral$block4_bias <- abs(neutral$block4_bias)

biased_means <- colMeans(biased[, c("block1_bias", "block2_bias", "block3_bias", "block4_bias")], na.rm

# Calculate means for neutral group
neutral_means <- colMeans(neutral[, c("block1_bias", "block2_bias", "block3_bias", "block4_bias")], na.

# Combine the means into a new data frame for plotting
means_df <- data.frame(
  Block = rep(c("Block 1", "Block 2", "Block 3", "Block 4"), 2),
  MeanBias = c(biased_means, neutral_means),
  Group = rep(c("Biased", "Neutral"), each = 4)
)


ggplot(means_df, aes(x = Block, y = MeanBias, color = Group, group = Group)) +
  geom_line() +  # Add lines
  geom_point() + # Add points
```

```r
  labs(title = "Mean Absolute Bias Scores by Bias Group and Card Sort Block", x = "Block", y = "Mean Bi
  scale_color_manual(values = c("Biased" = "red", "Neutral" = "blue")) +
  theme_minimal()
```



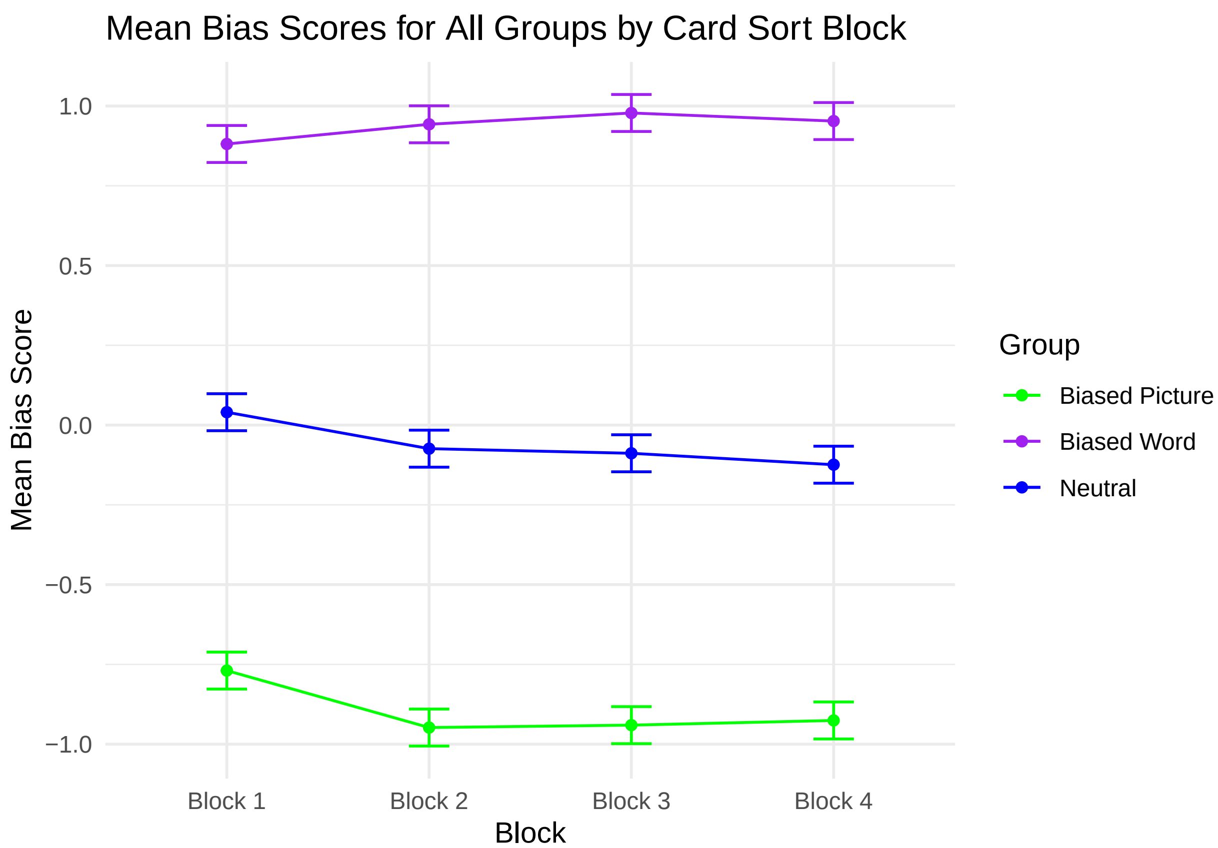

```r
# Create the combined plot
combined_plot <- ggplot() +
  geom_line(data = neutral_means_df, aes(x = Block, y = MeanBias, group = Group, color = "Neutral")) +
  geom_point(data = neutral_means_df, aes(x = Block, y = MeanBias, color = "Neutral")) +
  geom_line(data = biased_word_means_df, aes(x = Block, y = MeanBias, group = Group, color = "Biased Wo
  geom_point(data = biased_word_means_df, aes(x = Block, y = MeanBias, color = "Biased Word")) +
  geom_line(data = biased_picture_means_df, aes(x = Block, y = MeanBias, group = Group, color = "Biased
  geom_point(data = biased_picture_means_df, aes(x = Block, y = MeanBias, color = "Biased Picture")) +
  labs(title = "Mean Bias Scores for All Groups by Card Sort Block", x = "Block", y = "Mean Bias Score")
  scale_color_manual(name = "Group", values = c("Neutral" = "blue", "Biased Word" = "purple", "Biased Pi
  theme_minimal()

# Display the combined plot
print(combined_plot)
```




```
# trying to make it with error bars
# Step 1: Calculate descriptive statistics and store in a variable
biased_word <- subset(data, data$BiasScore > 0.8)
biased_picture <- subset(data, data$BiasScore < -0.8)
neutral <- subset(data, data$BiasScore <= 0.8 & data$BiasScore >= -0.8)

#biased word cleaning
cleaned_biased_word <- biased_word %>%
  select(-c(Subject.1))

descriptives_biased_word <- cleaned_biased_word %>%
  select(-c(Subject, Sex, InconOutlier,Attention)) %>%
  psych::describe()

# Convert the 'desc_df' to a regular dataframe
bw_descdf <- as.data.frame(descriptives_biased_word)

#biased picture cleaning

cleaned_biased_pic <- biased_picture %>%
  select(-c(Subject.1))

descriptives_biased_pic <- cleaned_biased_pic %>%
  select(-c(Subject, Sex, InconOutlier,Attention)) %>%
  psych::describe()
```

```r
# Convert the 'desc_df' to a regular dataframe
bp_descdf <- as.data.frame(descriptives_biased_pic)

#cleaning neutral
cleaned_neutral <- neutral %>%
  select(-c(block1_mean, block2_mean, block3_mean, block4_mean,
            block1_bias, block2_bias, block3_bias, block4_bias, Subject))

descriptives_neutral <- cleaned_neutral %>%
  select(-c (Sex, InconOutlier,Attention)) %>%
  psych::describe()

# Convert the 'desc_df' to a regular dataframe
n_descdf <- as.data.frame(descriptives_neutral)

# Calculate SE for each block for biased word data
bw_descdf$se <- bw_descdf$sd / sqrt(bw_descdf$n)

# Calculate SE for each block for biased picture data
bp_descdf$se <- bp_descdf$sd / sqrt(bp_descdf$n)

# Calculate SE for each block for neutral data
n_descdf$se <- n_descdf$sd / sqrt(n_descdf$n)

neutral_means_df$se <- NA  # Create a new column for SE

# Assuming order of SEs in 'n_descdf' corresponds to the blocks in 'neutral_means_df'
neutral_means_df$se[neutral_means_df$Block == "Block 1"] <- n_descdf$se[5]  # BiasScore SE for Block 1
neutral_means_df$se[neutral_means_df$Block == "Block 2"] <- n_descdf$se[5]  # BiasScore SE for Block 2
neutral_means_df$se[neutral_means_df$Block == "Block 3"] <- n_descdf$se[5]  # BiasScore SE for Block 3
neutral_means_df$se[neutral_means_df$Block == "Block 4"] <- n_descdf$se[5]  # BiasScore SE for Block 4


# Adding error bars to the plot
se<- n_descdf$se[5]
combined_plot <- ggplot() +
  geom_line(data = neutral_means_df, aes(x = Block, y = MeanBias, group = Group, color = "Neutral")) +
  geom_point(data = neutral_means_df, aes(x = Block, y = MeanBias, color = "Neutral")) +
  geom_errorbar(data = neutral_means_df, aes(x = Block, ymin = MeanBias - se, ymax = MeanBias + se, colo

  geom_line(data = biased_word_means_df, aes(x = Block, y = MeanBias, group = Group, color = "Biased Wor
  geom_point(data = biased_word_means_df, aes(x = Block, y = MeanBias, color = "Biased Word")) +
  geom_errorbar(data = biased_word_means_df, aes(x = Block, ymin = MeanBias - se, ymax = MeanBias + se,

  geom_line(data = biased_picture_means_df, aes(x = Block, y = MeanBias, group = Group, color = "Biased
  geom_point(data = biased_picture_means_df, aes(x = Block, y = MeanBias, color = "Biased Picture")) +
  geom_errorbar(data = biased_picture_means_df, aes(x = Block, ymin = MeanBias - se, ymax = MeanBias + s

  labs(title = "Mean Bias Scores for All Groups by Card Sort Block", x = "Block", y = "Mean Bias Score")
  scale_color_manual(name = "Group", values = c("Neutral" = "blue", "Biased Word" = "purple", "Biased Pi
  theme_minimal()

# Display the combined plot
```

```r
print(combined_plot)
```




```r
# Calculating changes between blocks for both groups
change1_bias = biased_means["block2_bias"] - biased_means["block1_bias"]
change2_bias = biased_means["block4_bias"] - biased_means["block3_bias"]

change1_neutral = neutral_means["block2_bias"] - neutral_means["block1_bias"]
change2_neutral = neutral_means["block4_bias"] - neutral_means["block3_bias"]

changes_bias = c(change1_bias, change2_bias)
changes_neutral = c(change1_neutral, change2_neutral)

t_test_all_changes = t.test(changes_bias, changes_neutral, alternative = "two.sided", var.equal = TRUE)

# Print the results
print(t_test_all_changes)
```

```
##
##  Two Sample t-test
##
## data:  changes_bias and changes_neutral
## t = -0.26396, df = 2, p-value = 0.8165
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.2736995  0.2420583
```

```
## sample estimates:
##  mean of x  mean of y
## 0.03691966 0.05274028
```

```
biased$block_diff <- abs(biased$block4_bias) - abs(biased$block1_bias)
neutral$block_diff <- abs(neutral$block4_bias) - abs(neutral$block1_bias)

# Perform t-test
t_test_result <- t.test(biased$block_diff, neutral$block_diff, var.equal = FALSE)

# Display the result
t_test_result
```

```
##
##  Welch Two Sample t-test
##
## data:  biased$block_diff and neutral$block_diff
## t = -0.33269, df = 102.15, p-value = 0.7401
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.1135304  0.0809159
## sample estimates:
##  mean of x  mean of y
## 0.09663609 0.11294334
```

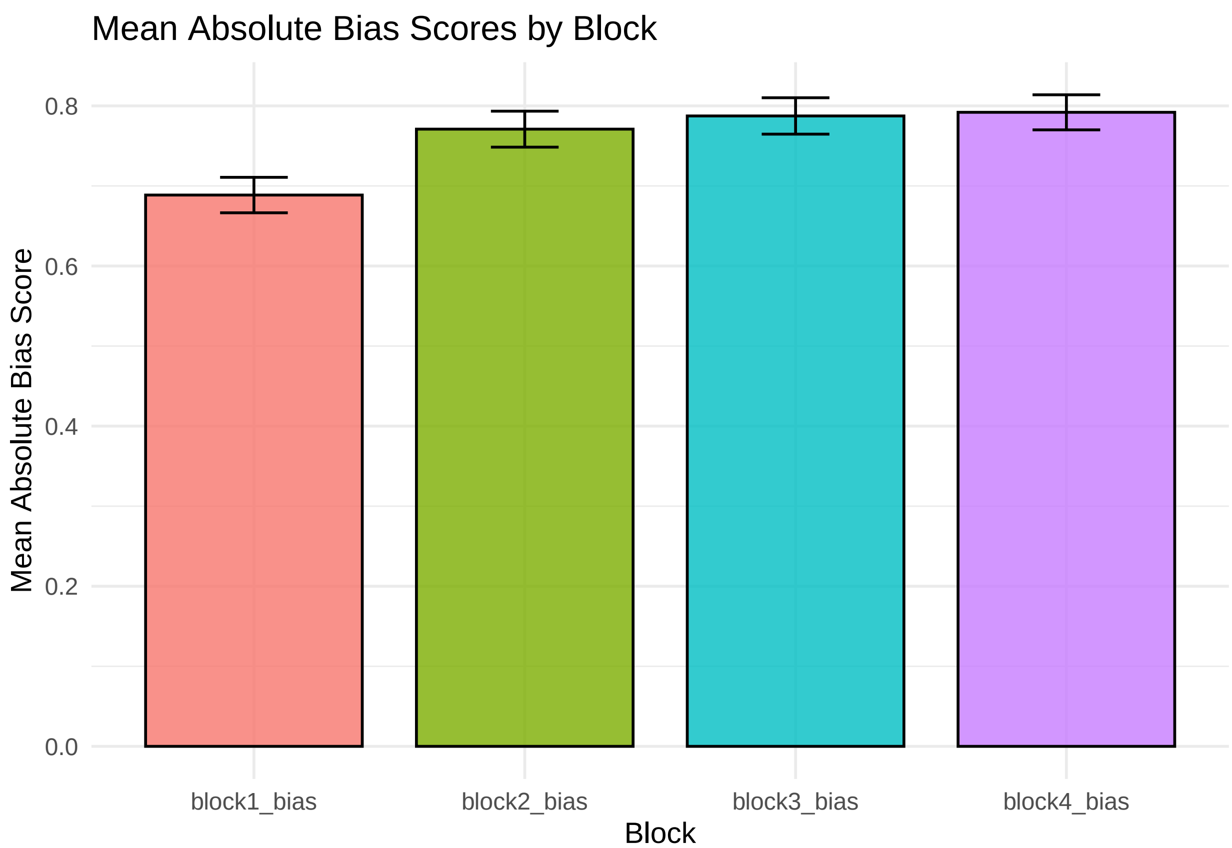
```
biased_ttest_b1 <- abs(biased$block1_bias)
neutral_ttest_b2 <- abs(neutral$block1_bias)

# Perform t-test
t_test_result <- t.test(biased_ttest_b1, neutral_ttest_b2, var.equal = FALSE)

# Display the result
t_test_result
```

```
##
##  Welch Two Sample t-test
##
## data:  biased_ttest_b1 and neutral_ttest_b2
## t = 10.214, df = 110.47, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.3131712 0.4639433
## sample estimates:
## mean of x mean of y
## 0.8482254 0.4596681
```

```
#convert data frame to long format
data <- data[, !duplicated(colnames(data))]
data <- data %>%
  mutate(Attention = ifelse(BiasScore > 0.8 | BiasScore < -0.8, "biased", "neutral"))

   long_block_data <- data %>%
```

```r
    gather(key = "Condition", value = "BiasScore", block1_bias:block4_bias) %>%
    mutate(Condition = gsub("_bias", "", Condition)) %>%
    mutate(BiasScore = abs(BiasScore)) %>%
    select(Attention, Subject, Condition, BiasScore)


#block_aov = anova_test(data=long_block_data, dv=BiasScore, wid=Subject, between=Attention, within = Co
#print(block_aov)

data <- data[, !duplicated(colnames(data))]
data <- data %>%
  mutate(Attention = ifelse(BiasScore > 0.8 | BiasScore < -0.8, "biased", "neutral"))

   long_block_data <- data %>%
    gather(key = "Condition", value = "BiasScore", block1_bias:block2_bias) %>%
    mutate(Condition = gsub("_bias", "", Condition)) %>%
    mutate(BiasScore = abs(BiasScore)) %>%
    select(Attention, Subject, Condition, BiasScore)



#block_aov = anova_test(data=long_block_data, dv=BiasScore, wid=Subject, between=Attention, within = Co
#print(block_aov)
```

```r
df_unique<-data
data_long <- df_unique %>%
  select(Subject, block1_bias, block2_bias, block3_bias, block4_bias) %>%
  pivot_longer(cols = starts_with("block"), names_to = "Block", values_to = "BiasScore") %>%
  mutate(absBiasScore = abs(BiasScore))

# Calculate mean and standard error for each block
se_sum <- data_long %>%
  group_by(Block) %>%
  summarise(
    mean = mean(absBiasScore, na.rm = TRUE),
    sd = sd(absBiasScore, na.rm = TRUE),
    n = n()
  ) %>%
  mutate(se = sd/sqrt(n))

# Plotting
ggplot(se_sum, aes(x = Block, y = mean, fill = Block)) +
  geom_bar(position = position_dodge(0.8), stat = "identity", color = "black", size = 0.5, width = 0.8,
  geom_errorbar(aes(ymin = mean - se, ymax = mean + se), position = position_dodge(0.8), width = 0.25, 
  labs(title = "Mean Absolute Bias Scores by Block",
       x = "Block",
       y = "Mean Absolute Bias Score") +
  theme_minimal() +
  theme(legend.position = "none")
```

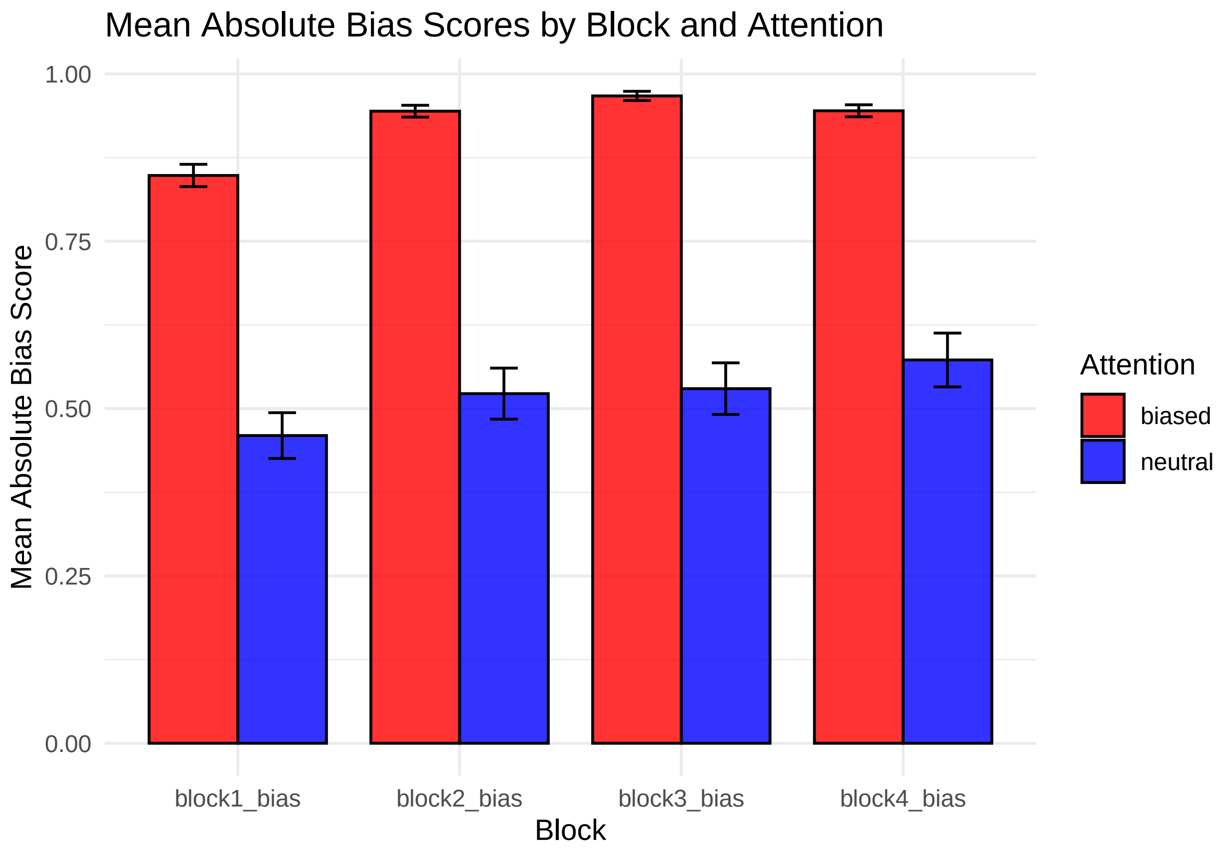



```
data_long <- df_unique %>%
  select(Subject, Attention, block1_bias, block2_bias, block3_bias, block4_bias) %>%
  pivot_longer(cols = starts_with("block"), names_to = "Block", values_to = "BiasScore") %>%
  mutate(absBiasScore = abs(BiasScore))

# Calculate mean and standard error for each block and attention group
se_sum <- data_long %>%
  group_by(Attention, Block) %>%
  summarise(
    mean = mean(absBiasScore, na.rm = TRUE),
    sd = sd(absBiasScore, na.rm = TRUE),
    n = n()
  ) %>%
  mutate(se = sd/sqrt(n))
```

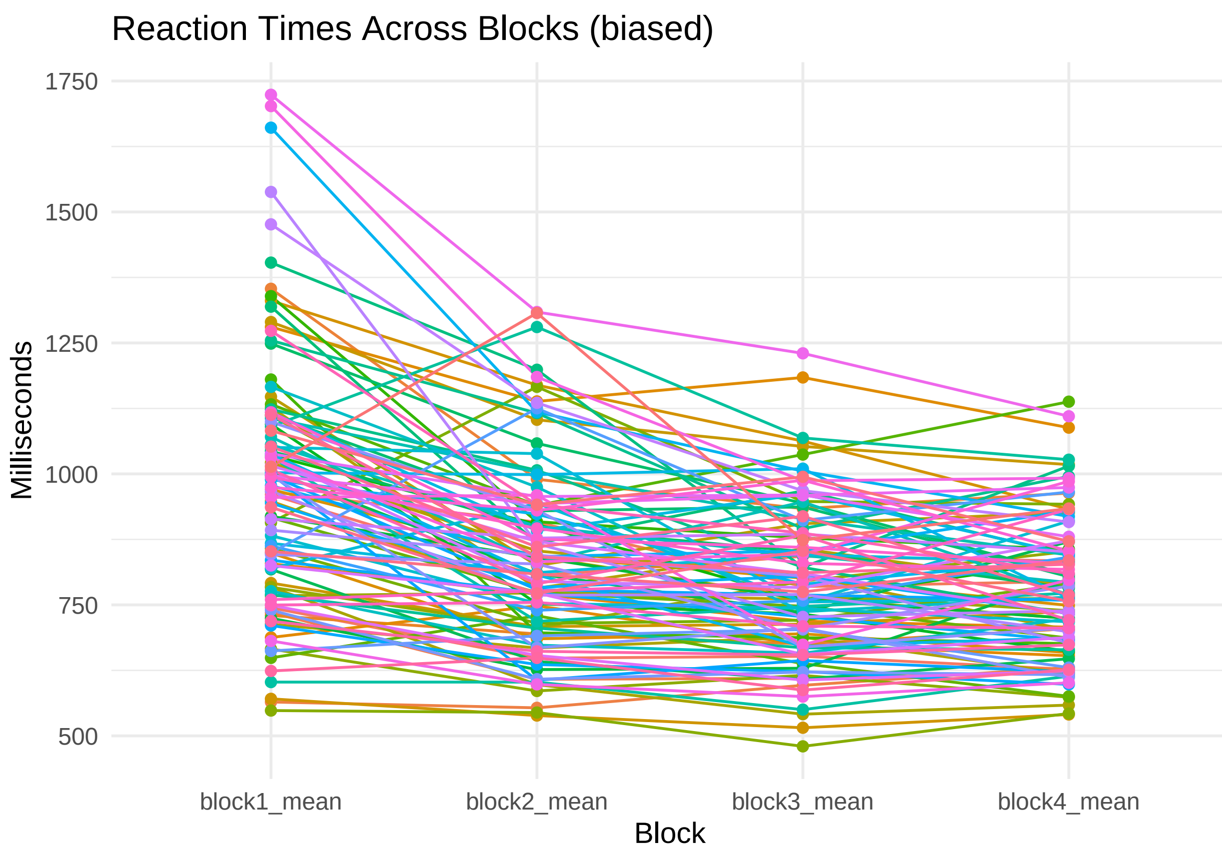
```
## `summarise()` has grouped output by 'Attention'. You can override using the
## `.groups` argument.
```

```
# Plotting
ggplot(se_sum, aes(x = Block, y = mean, fill = Attention)) +
  geom_bar(position = position_dodge(0.8), stat = "identity", color = "black", size = 0.5, width = 0.8,
  geom_errorbar(aes(ymin = mean - se, ymax = mean + se), position = position_dodge(0.8), width = 0.25,
  scale_fill_manual(values = c("neutral" = "blue", "biased" = "red")) +
  labs(title = "Mean Absolute Bias Scores by Block and Attention",
       x = "Block",
```

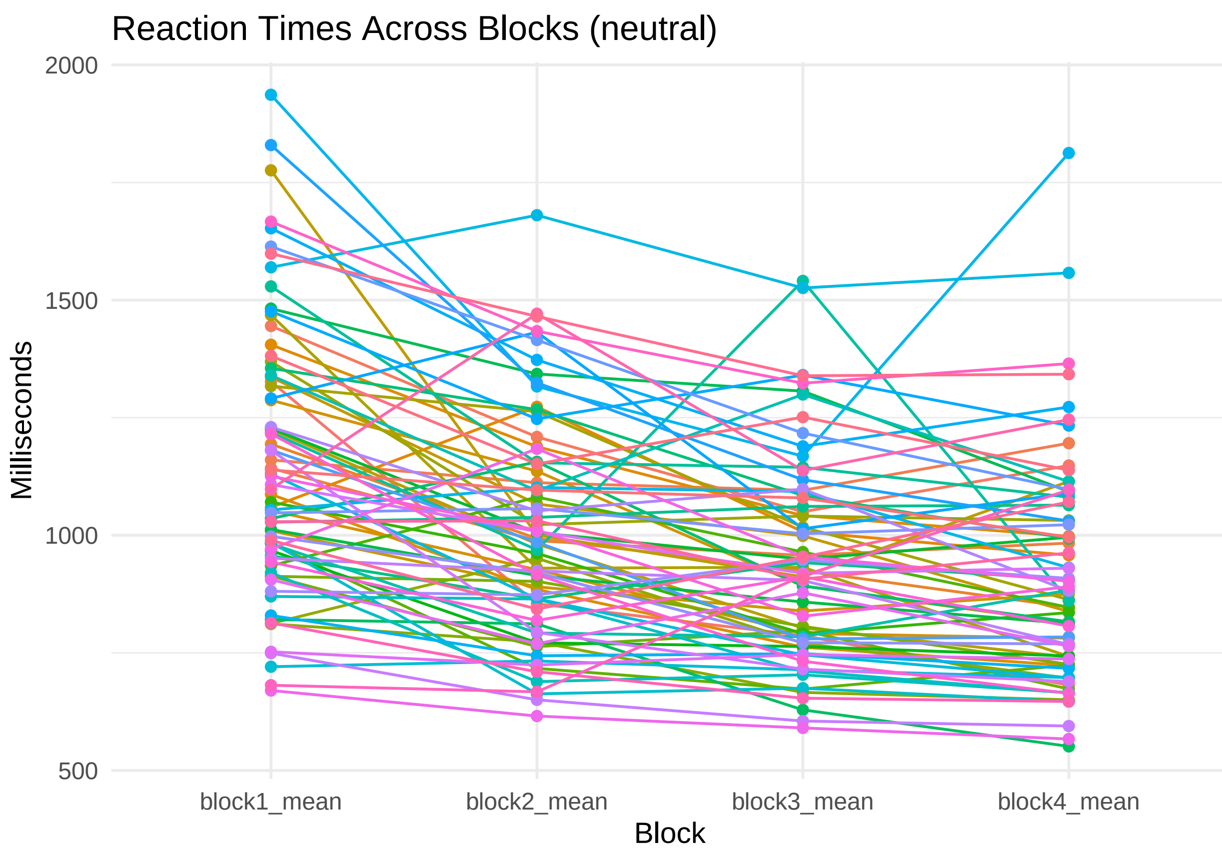
```
      y = "Mean Absolute Bias Score") +
  theme_minimal()
```



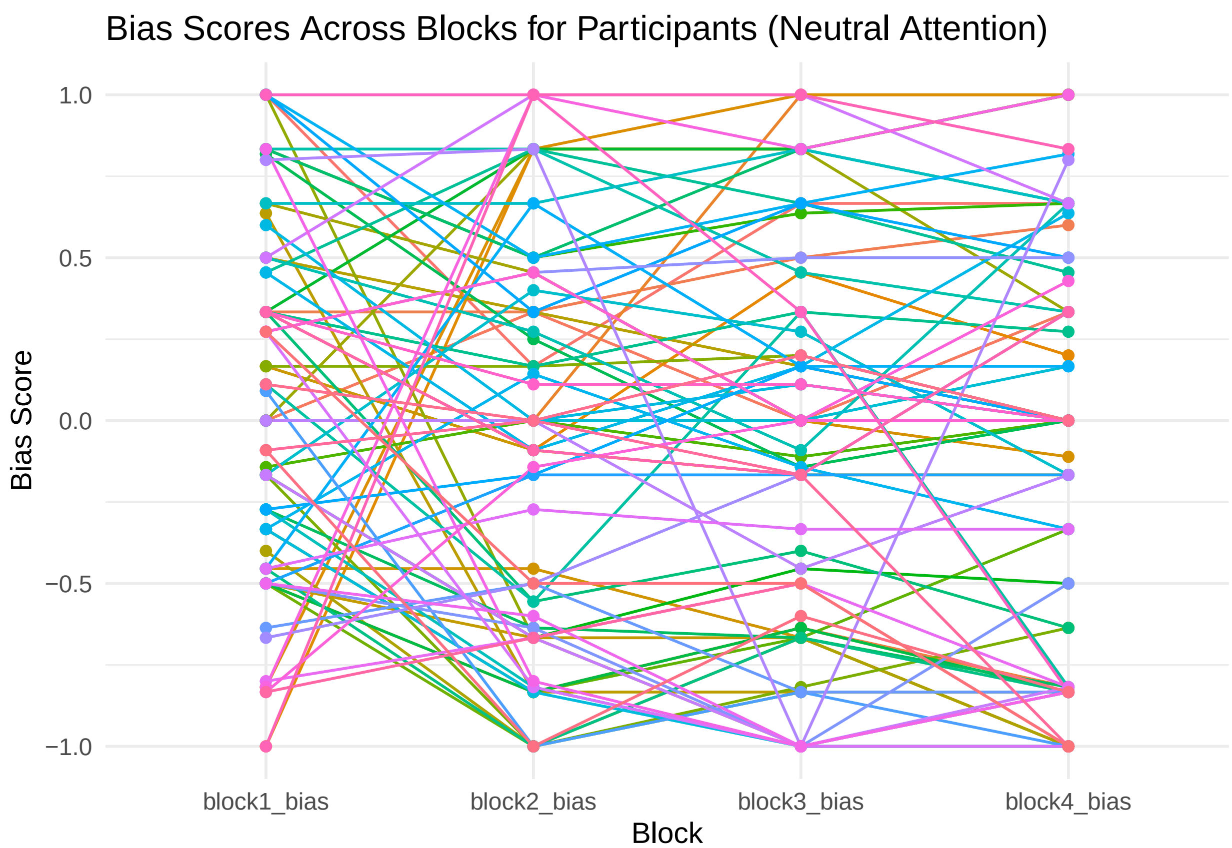

```
long_df <- data %>%
  gather(key = "Block", value = "RT", block1_mean:block4_mean)

# Convert block names to a factor to ensure proper ordering
long_df$Block <- factor(long_df$Block, levels = c("block1_mean", "block2_mean", "block3_mean", "block4_

biased <- subset(long_df, long_df$BiasScore > 0.8 | long_df$BiasScore < -0.8)
neutral <- subset(long_df, long_df$BiasScore <= 0.8 & long_df$BiasScore >= -0.8)


# Plot using ggplot2 with a subset of subjects
ggplot(biased, aes(x = Block, y = RT, group = Subject, color = Subject)) +
  geom_line() +
  geom_point() +
  labs(
    title = "Reaction Times Across Blocks (biased)",
    x = "Block",
    y = "Milliseconds"
  ) +
  theme_minimal()+
  theme(legend.position = "none")
```



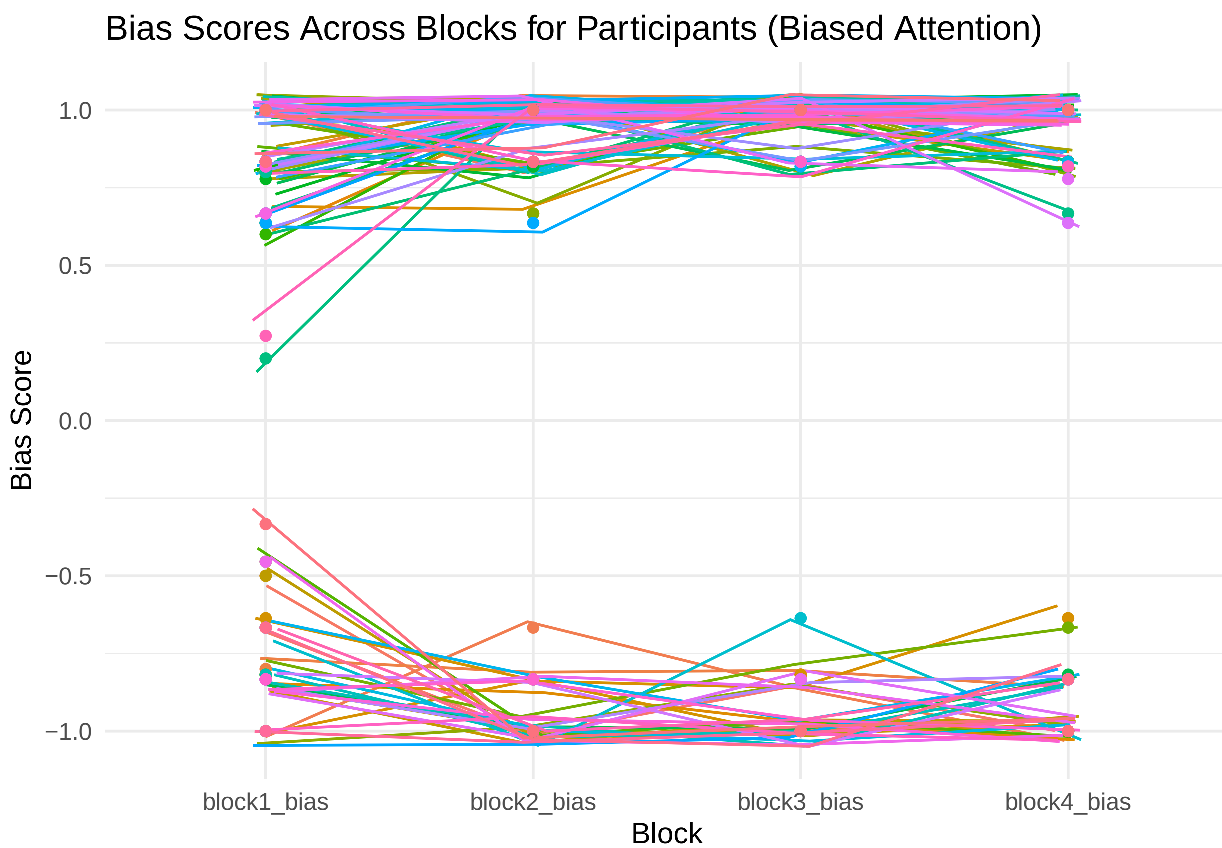

```r
ggplot(neutral, aes(x = Block, y = RT, group = Subject, color = Subject)) +
  geom_line() +
  geom_point() +
  labs(
    title = "Reaction Times Across Blocks (neutral)",
    x = "Block",
    y = "Milliseconds"
  ) +
  theme_minimal()+
  theme(legend.position = "none")
```



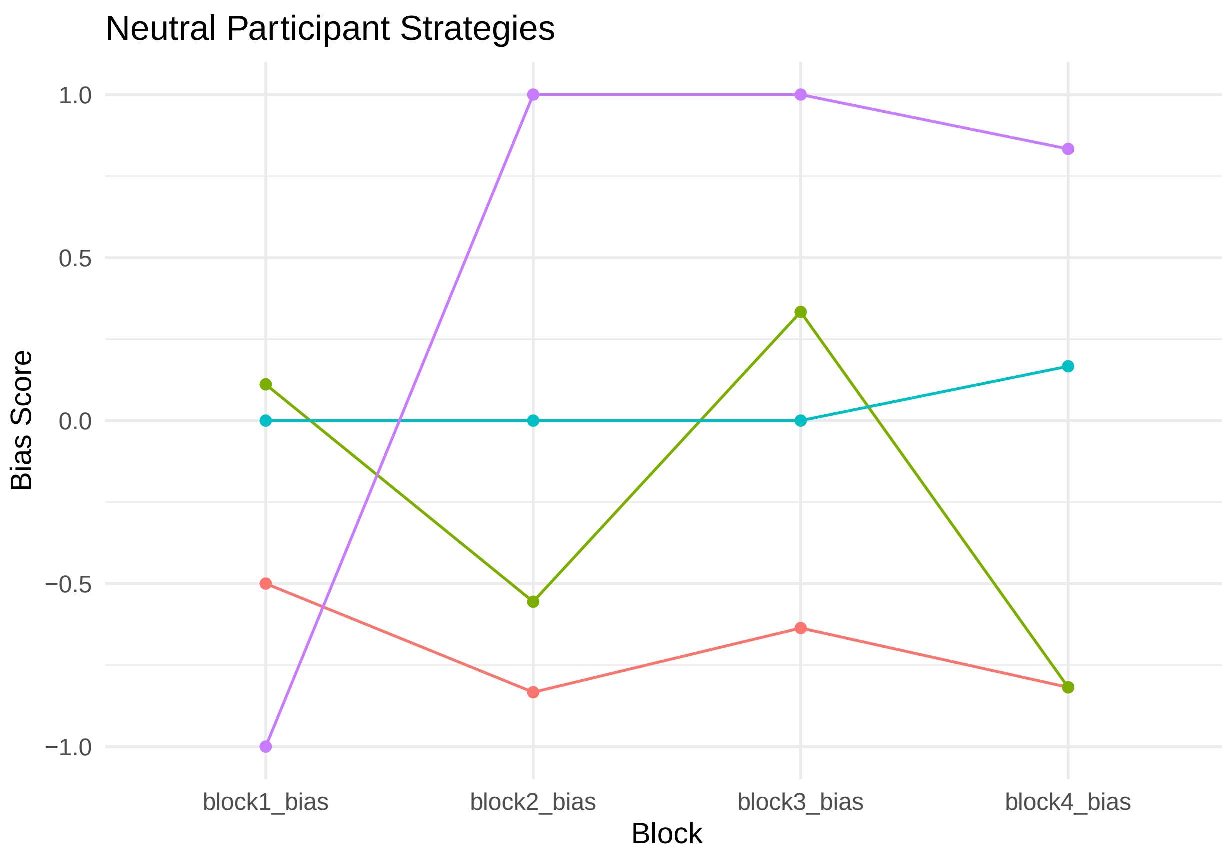

```r
long_df2 <- data %>%
  gather(key = "Block", value = "BiasScore", block1_bias:block4_bias)

#Bias Score across blocks

df_unique <- data[, !duplicated(as.list(data))]

data_neutral <- df_unique %>%
  filter(Attention == 'neutral')

# Reshape the data for ggplot2
data_neutral_long <- data_neutral %>%
  select(Subject, block1_bias, block2_bias, block3_bias, block4_bias) %>%
  pivot_longer(cols = starts_with("block"), names_to = "Block", values_to = "BiasScore")

# Plotting
ggplot(data_neutral_long, aes(x = Block, y = BiasScore, group = Subject, color = Subject)) +
  geom_line() +
  geom_point() +
  labs(title = "Bias Scores Across Blocks for Participants (Neutral Attention)",
       x = "Block",
       y = "Bias Score") +
  theme_minimal() +
  theme(legend.position = "none")
```




```r
#biased
data_biased <- df_unique %>%
  filter(Attention == 'biased')

# Reshape the data for ggplot2
data_biased_long <- data_biased %>%
  select(Subject, block1_bias, block2_bias, block3_bias, block4_bias) %>%
  pivot_longer(cols = starts_with("block"), names_to = "Block", values_to = "BiasScore")

# Plotting
ggplot(data_biased_long, aes(x = Block, y = BiasScore, group = Subject, color = Subject)) +
  geom_line(position=position_jitter(w=0.05,h=0.05)) +
  geom_point() +
  labs(title = "Bias Scores Across Blocks for Participants (Biased Attention)",
       x = "Block",
       y = "Bias Score") +
  theme_minimal() +
  theme(legend.position = "none")
```




```
data_neutral_selected <- data_neutral_long %>%
  filter(Subject %in% c("e8b26ab1", "89601069", "08f746fa", "6409a8b2"))  # Replace 1, 2, 3 with the su

# Plot the selected participants
ggplot(data_neutral_selected, aes(x = Block, y = BiasScore, group = Subject, color = Subject)) +
  geom_line() +
  geom_point() +
  labs(
    title = "Neutral Participant Strategies",
    x = "Block",
    y = "Bias Score"
  ) +
  theme_minimal() +
  theme(legend.position = "none")
```




```r
print(data_neutral_long)
```

```
## # A tibble: 304 x 3
##    Subject  Block       BiasScore
##    <chr>    <chr>           <dbl>
##  1 0156ce12 block1_bias     1
##  2 0156ce12 block2_bias     0.167
##  3 0156ce12 block3_bias     0.667
##  4 0156ce12 block4_bias     0.667
##  5 054ba968 block1_bias     0
##  6 054ba968 block2_bias     0.333
##  7 054ba968 block3_bias     0
##  8 054ba968 block4_bias     0.333
##  9 05546b7f block1_bias     0.333
## 10 05546b7f block2_bias     0.333
## # i 294 more rows
```

```r
demographics_data <-read.csv("Copy of CardSortingTask_Analysis_09.28.2023.csv")
names(demographics_data)[names(demographics_data) == "SubjectNumber"] <- "Subject"
merged_df <- merge(demographics_data, df_unique, by = "Subject")
cols_toremove <- c("SymRespCount.x", "TxtRespCount.x", "IncorrRespCount")
merged_df <- merged_df %>% select(-one_of(cols_toremove))

merged_df$Block <- as.factor(sub("CardSort_Block", "", merged_df$Block))
```

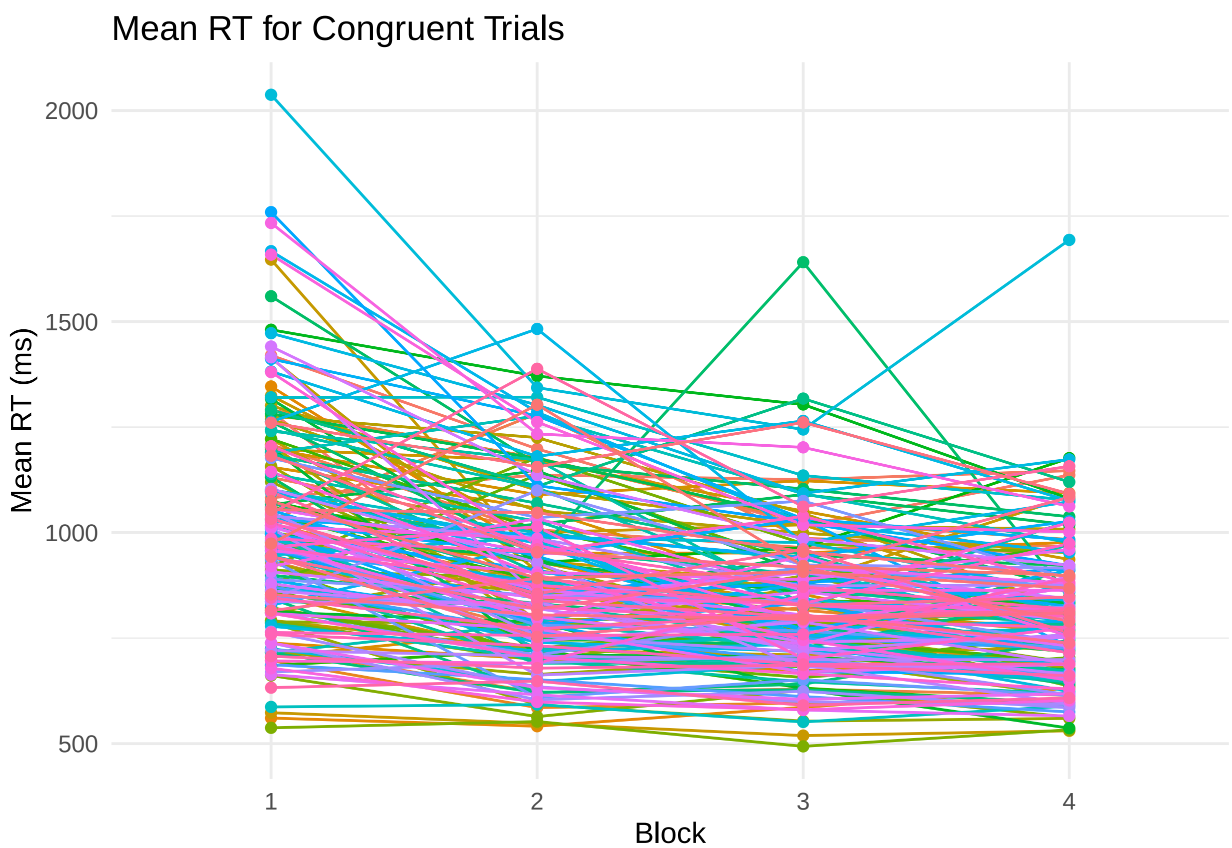
```r
# Split the data into congruent and incongruent trials based on some condition criteria
# Note: You need to adjust 'Condition' based on what defines congruent (1) and incongruent (2) trials i
congruent_data <- merged_df %>%
  filter(Condition == 1) %>%
  group_by(Subject, Block) %>%
  summarise(Mean_RT = mean(RT, na.rm = TRUE))
```

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

```r
incongruent_data <- merged_df %>%
  filter(Condition == 2) %>%
  group_by(Subject, Block) %>%
  summarise(Mean_RT = mean(RT, na.rm = TRUE))
```

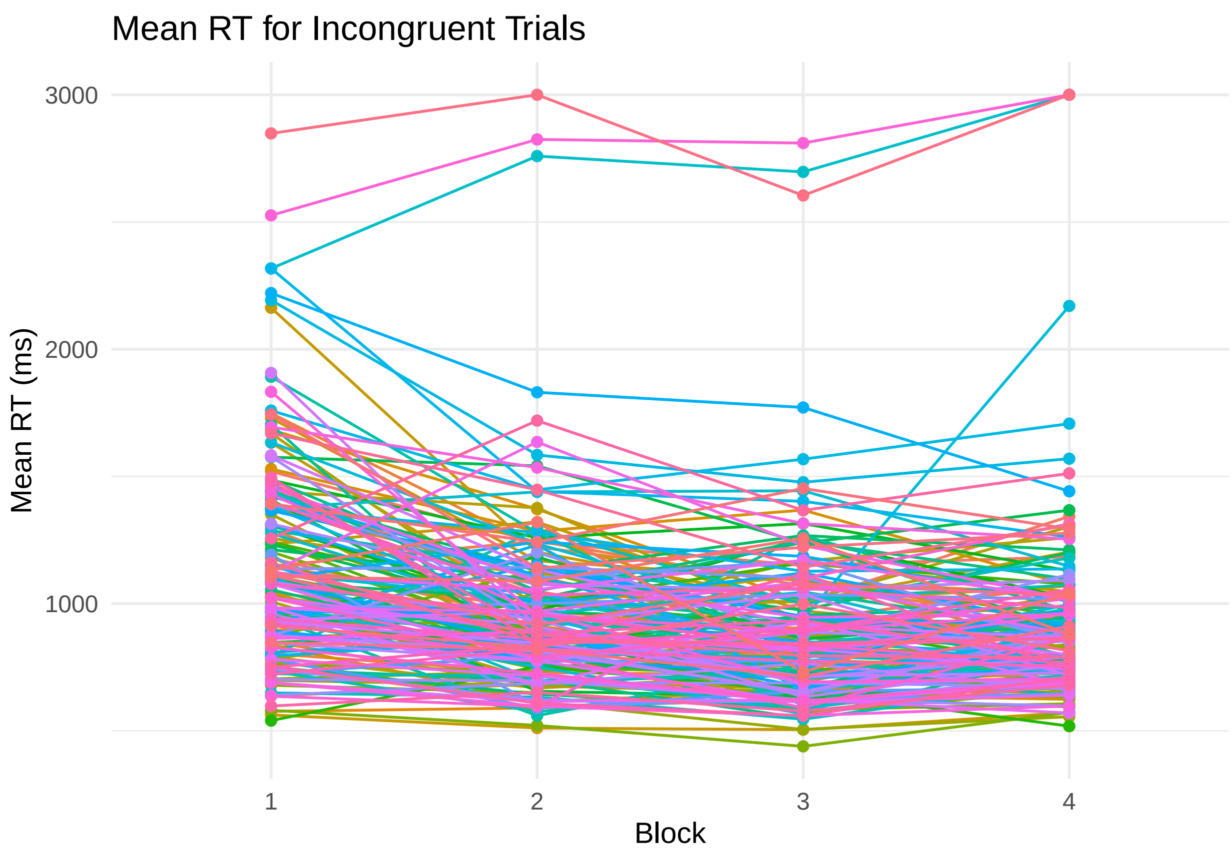
```
## `summarise()` has grouped output by 'Subject'. You can override using the
## `.groups` argument.
```

```r
# Create a plot for Congruent Trials
p1 <- ggplot(congruent_data, aes(x = Block, y = Mean_RT, group = Subject, color = Subject)) +
  geom_line() +
  geom_point() +
  labs(title = "Mean RT for Congruent Trials", x = "Block", y = "Mean RT (ms)") +
  theme_minimal()+
  theme(legend.position = "none")

# Create a plot for Incongruent Trials
p2 <- ggplot(incongruent_data, aes(x = Block, y = Mean_RT, group = Subject, color = Subject)) +
  geom_line() +
  geom_point() +
  labs(title = "Mean RT for Incongruent Trials", x = "Block", y = "Mean RT (ms)") +
  theme_minimal()+
  theme(legend.position = "none")

# Print the plots
print(p1)
```



Mean RT for Congruent Trials
Mean RT (ms)
2000
1500
1000
500
1
2
3
4
Block

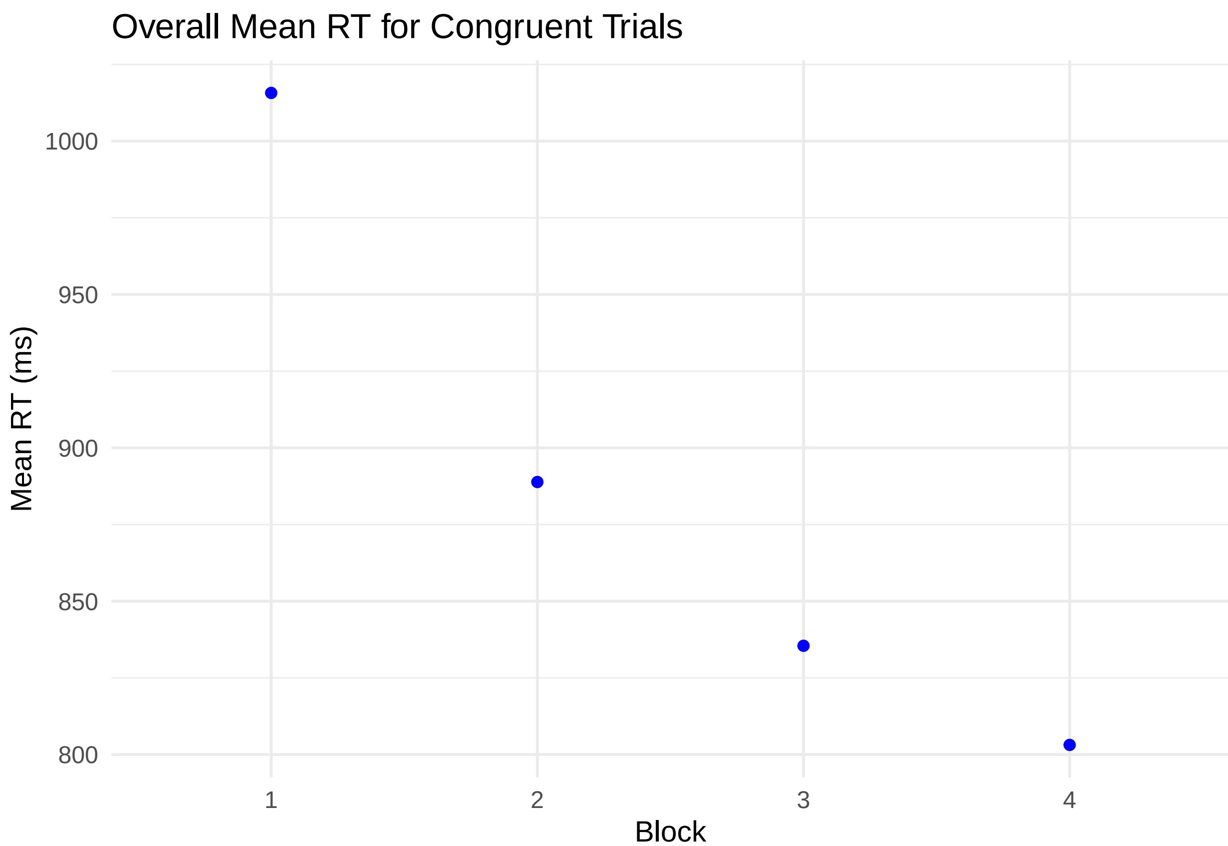

```r
print(p2)
```




```r
overall_congruent <- merged_df %>%
  filter(Condition == 1) %>%
  group_by(Block) %>%
  summarise(Mean_RT = mean(RT, na.rm = TRUE))

overall_incongruent <- merged_df %>%
  filter(Condition == 2) %>%
  group_by(Block) %>%
  summarise(Mean_RT = mean(RT, na.rm = TRUE))

# Plotting overall mean RT for Congruent Trials
p1 <- ggplot(overall_congruent, aes(x = Block, y = Mean_RT)) +
  geom_line(color = "blue") +
  geom_point(color = "blue") +
  labs(title = "Overall Mean RT for Congruent Trials", x = "Block", y = "Mean RT (ms)") +
  theme_minimal() +
  theme(legend.position = "none")

# Plotting overall mean RT for Incongruent Trials
p2 <- ggplot(overall_incongruent, aes(x = Block, y = Mean_RT)) +
  geom_line(color = "red") +
  geom_point(color = "red") +
  labs(title = "Overall Mean RT for Incongruent Trials", x = "Block", y = "Mean RT (ms)") +
  theme_minimal() +
  theme(legend.position = "none")
```

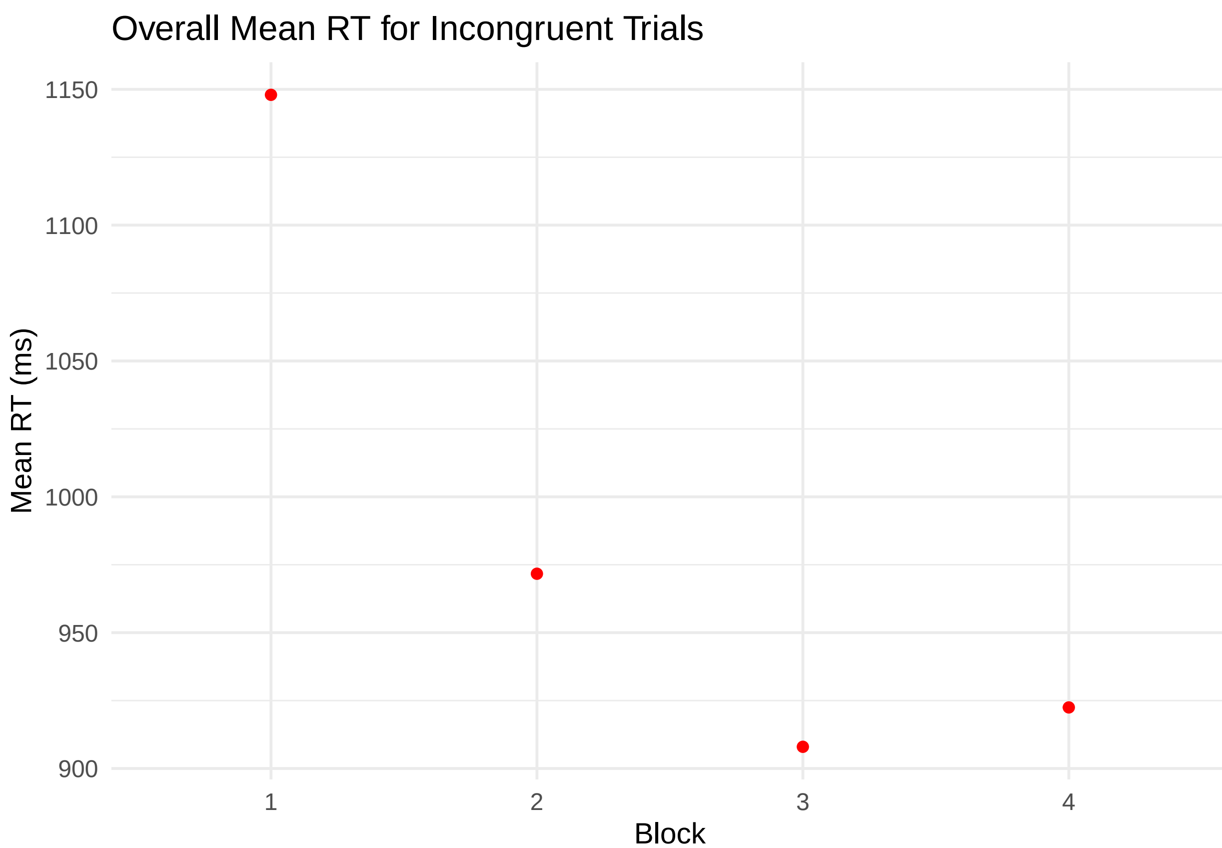
```r
# Print the plots
print(p1)
```

```
## `geom_line()`: Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?
```



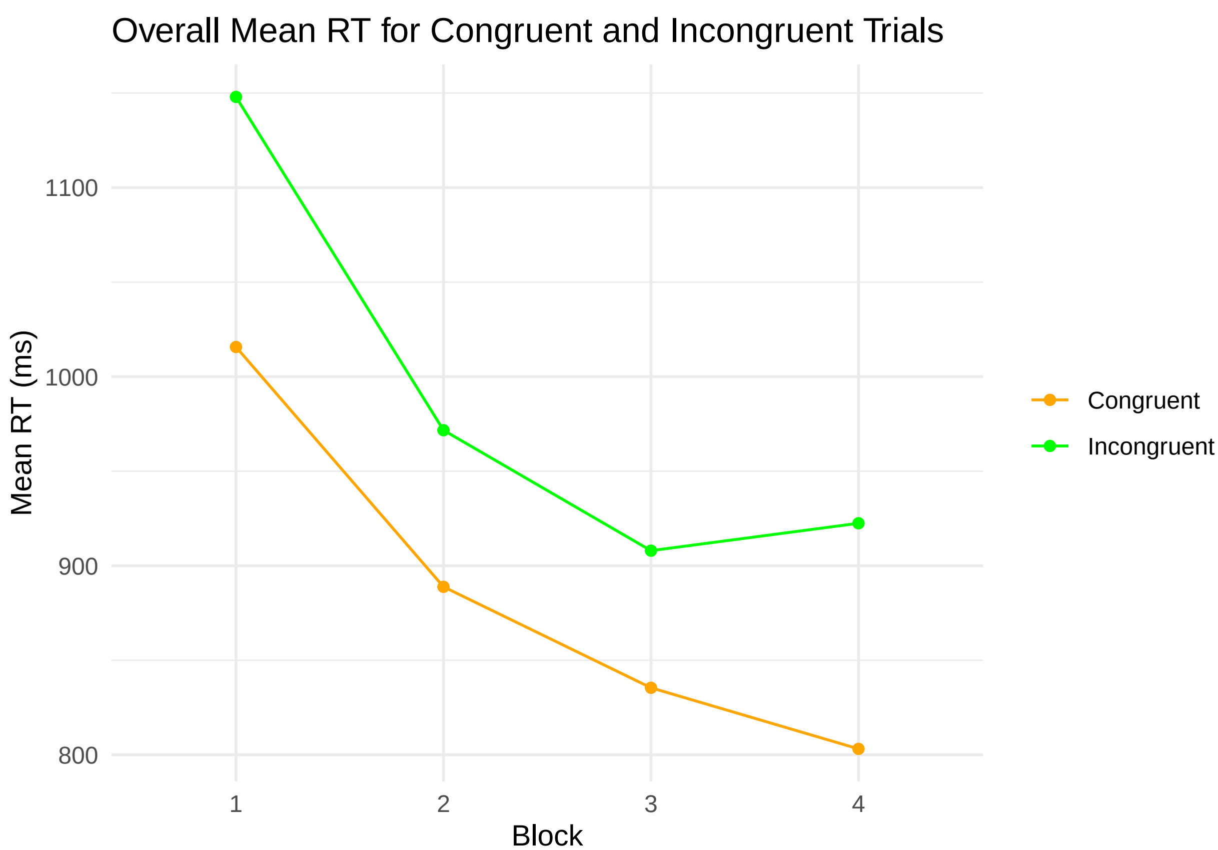

```r
print(p2)
```

```
## `geom_line()`: Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?
```




```
merged_df$Block <- as.factor(sub("CardSort_Block", "", merged_df$Block))

# Calculate overall mean RT for Congruent and Incongruent Trials for each block and create a new column
congruent_data <- merged_df %>%
  filter(Condition == 1) %>%
  group_by(Block) %>%
  summarise(Mean_RT = mean(RT, na.rm = TRUE)) %>%
  mutate(Type = "Congruent")

incongruent_data <- merged_df %>%
  filter(Condition == 2) %>%
  group_by(Block) %>%
  summarise(Mean_RT = mean(RT, na.rm = TRUE)) %>%
  mutate(Type = "Incongruent")

# Combine the datasets
combined_data <- rbind(congruent_data, incongruent_data)

# Plotting overall mean RT for both Congruent and Incongruent Trials
combined_plot <- ggplot(combined_data, aes(x = Block, y = Mean_RT, color = Type, group = Type)) +
  geom_line() +
  geom_point() +
  labs(title = "Overall Mean RT for Congruent and Incongruent Trials", x = "Block", y = "Mean RT (ms)")
  scale_color_manual(values = c("orange", "green")) +
  theme_minimal() +
  theme(legend.title = element_blank())  # Optionally remove legend title
```

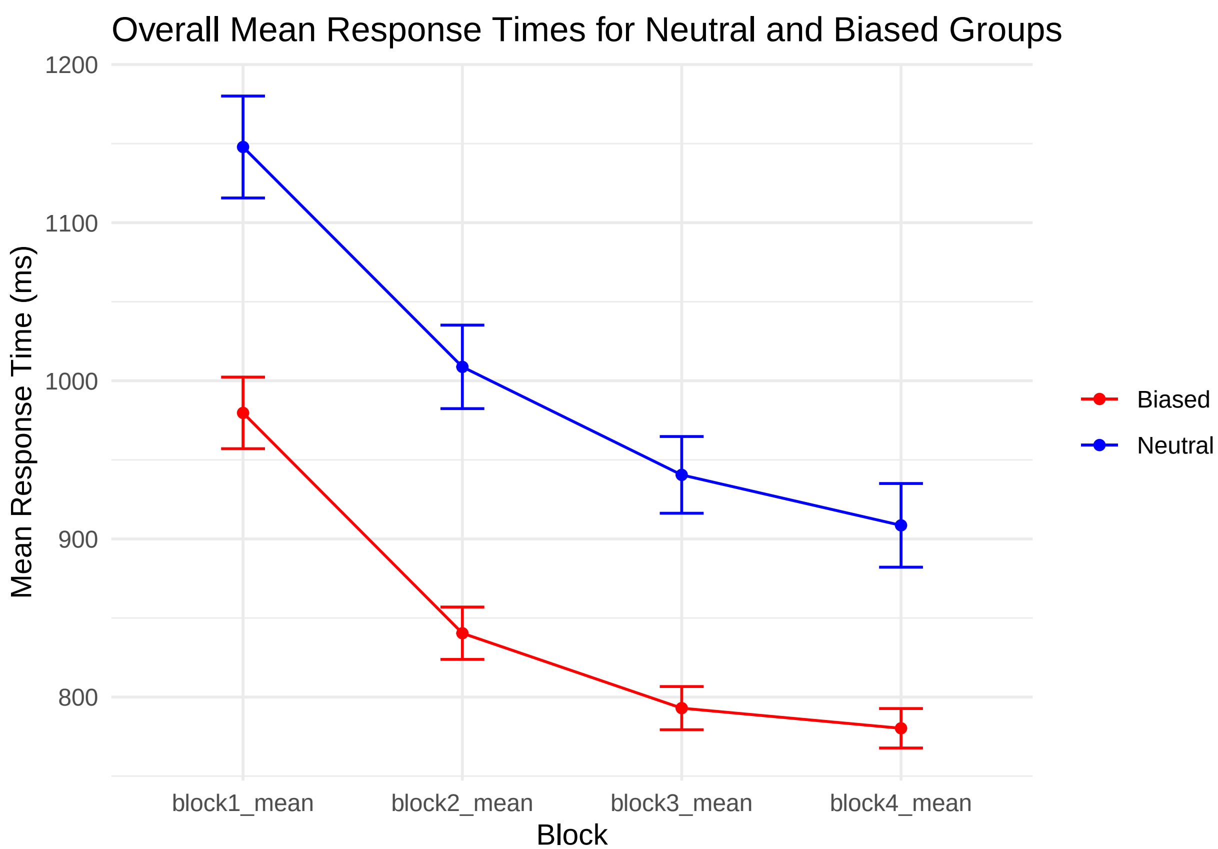
```r
# Print the combined plot
print(combined_plot)
```




```r
biased$Group <- "Biased"
neutral$Group <- "Neutral"

# Combine the two datasets into one
combined_data <- rbind(biased, neutral)

# Function to calculate standard error of the mean (SEM)
sem <- function(x) {
  return(sd(x, na.rm = TRUE) / sqrt(length(x)))
}

# Aggregate the data to calculate mean RT and SEM for each Block within each Group
mean_rt_scores <- combined_data %>%
  group_by(Block, Group) %>%
  summarise(
    Mean_RT = mean(RT, na.rm = TRUE),
    SEM = sem(RT)
  )
```

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

```r
# Plotting overall mean response time with error bars for Neutral and Biased Groups
combined_plot <- ggplot(mean_rt_scores, aes(x = Block, y = Mean_RT, color = Group, group = Group)) +
  geom_line() +
  geom_point() +
  geom_errorbar(aes(ymin = Mean_RT - SEM, ymax = Mean_RT + SEM), width = 0.2) +
  labs(title = "Overall Mean Response Times for Neutral and Biased Groups", x = "Block", y = "Mean Respo
  scale_color_manual(values = c("red", "blue")) +
  theme_minimal() +
  theme(legend.title = element_blank())  # Optionally remove legend title

# Print the combined plot
print(combined_plot)
```




```r
parc_cardsort <- read.csv("CardSort_Summary(in).csv")
parc_demographics <- read.csv("Demographics_Summary(in).csv")

parc_merged <- merge(parc_demographics, parc_cardsort, by = "Subject")
maia_data <- read.csv("CardSort Data.csv")
lang_data <- read.csv("CardSortLanguage.csv")
lang_data = lang_data[, c("SubjectNumber", "NativeEnglish", "SecondLang", "WhatSecLang")]
lang_data$L1 = NA

for (subject in lang_data$SubjectNumber){
  lang_data$L1[lang_data$NativeEnglish == 1] <- "English"
  lang_data$L1[lang_data$NativeEnglish == 2] <- lang_data$WhatSecLang[lang_data$NativeEnglish == 2]
```

```r
}

# Drop columns using base R
lang_data <- lang_data[, setdiff(names(lang_data), c("NativeEnglish", "SecondLang", "WhatSecLang"))]

maia_merged = merge(lang_data, maia_data, by= "SubjectNumber")

parc_merged = parc_merged[, c("Subject", "L1", "SymRespCount", "TxtRespCount","IncorrRespCount")]
maia_merged = maia_merged[, c("SubjectNumber", "L1", "SymRespCount", "TxtRespCount", "IncorrRespCount")]
names(maia_merged)[names(maia_merged) == "SubjectNumber"] <- "Subject"

merged <- rbind(maia_merged, parc_merged)

englishL1 <- merged[merged$L1 == "English", ]
unique_subjects_eng <- unique(englishL1$Subject)
num_sub_eng <- length(unique_subjects_eng)

not_englishL1 <- merged[merged$L1 != "English", ]
unique_subjects_not_eng <- unique(not_englishL1$Subject)
num_eng_not_L1 <- length(unique_subjects_not_eng)
```