Prediction with SEM: CERQ example on item level

true

November 08, 2021

Intro

I made an R-function for the SEM based prediction rule and in this note I will analyze the `CERQ` data with this rule. In this data set we have 4 items for each of 9 scales that measure emotion regulation. As a response variable we have the SCL score (actually we have item scores also) measured at a later time point. At the first timepoint we also have a SCL score. For more detail see https://scholarlypublications.universiteitleiden.nl/access/item%3A2871874/view

Data set

```
source('~/surfdrive/Predictive-Psychometrics/paper/SEM-Predictive Validity/versie2/Rcode/predict
load("~/surfdrive/Shared/pred_val_shared/Project_2/p2_application/CERQ Project/CERQdepr_12.Rdata
load("~/surfdrive/Shared/pred_val_shared/Project_2/p2_application/CERQ Project/wideCERQ_T1_T2_T3
# merge
CERQdepr_12 = merge(CERQdepr_12, wideCERQ_T1[ , c(1, 18)])

# depression scale at T = 1
SCLt1 = CERQdepr_12[, 76]

# response variable: scale score at T = 2
SCLt2 = CERQdepr_12[, 72]

#response variable: item scores at T = 2
SCLi = CERQdepr_12[, c(56:71)]
colnames(SCLi) = paste("d", c(1:16), sep = "")

# item data
CERQi = CERQdepr_12[ , c(2:5, 8:11, 14:17, 20:23, 26:29, 32:35, 38:41, 44:47, 50:53)]
colnames(CERQi) = c("sb1",
                    "sb2",
                    "sb3",
                    "sb4",
                    "ac1",
                    "ac2",
                    "ac3",
```

```
                   "ac4",
                   "ru1",
                   "ru2",
                   "ru3",
                   "ru4",
                   "rf1",
                   "rf2",
                   "rf3",
                   "rf4",
                   "rp1",
                   "rp2",
                   "rp3",
                   "rp4",
                   "ra1",
                   "ra2",
                   "ra3",
                   "ra4",
                   "pp1",
                   "pp2",
                   "pp3",
                   "pp4",
                   "ca1",
                   "ca2",
                   "ca3",
                   "ca4",
                   "bo1",
                   "bo2",
                   "bo3",
                   "bo4"
                   )
# scale data
CERQt = CERQdepr_12[ , c(6, 12, 18, 24, 30, 36, 42, 48, 54)]
colnames(CERQt) = c("sb",
                    "ac",
                    "ru",
                    "rf",
                    "rp",
                    "ra",
                    "pp",
                    "ca",
                    "bo"
)

# data set with item scores
```

Analysis Depression Scale

```
mydat1 = cbind(CERQi, SCLt2)
mydat1 = mydat1[complete.cases(mydat1), ]

model1 <- '
  # latent variable definitions
  sb =~ sb1 + sb2 + sb3 + sb4
  ac =~ ac1 + ac2 + ac3 + ac4
  ru =~ ru1 + ru2 + ru3 + ru4
  rf =~ rf1 + rf2 + rf3 + rf4
  rp =~ rp1 + rp2 + rp3 + rp4
  ra =~ ra1 + ra2 + ra3 + ra4
  pp =~ pp1 + pp2 + pp3 + pp4
  ca =~ ca1 + ca2 + ca3 + ca4
  bo =~ bo1 + bo2 + bo3 + bo4
  # regressions
  SCLt2 ~ sb + ac + ru + rf + rp + ra + pp + ca + bo
'

fit.sem <- sem(model1, data = mydat1, std.lv = TRUE, meanstructure = TRUE, warn = FALSE)
summary(fit.sem)
```

```
mydat2 = cbind(CERQt, SCLi)
mydat2 = mydat2[complete.cases(mydat2), ]
```

Analysis Depression Items

With the following code I define the SEM model. As methods of comparison I use a linear regression on the item scores estimated by various forms of elastic net (including lasso and ridge) as well as ordinary least squares. Also we use a linear regression on the scale scores, which is equivalent to a SEM model with an equality constraint on the factor loadings.

```
mydat3 = cbind(CERQi, SCLi)
mydat3 = mydat3[complete.cases(mydat3), ]
x = as.matrix(mydat3[, 1:36]);
y = as.matrix(mydat3[, 37:52])


model <- '
  # latent variable definitions
  sb =~ sb1 + sb2 + sb3 + sb4
  ac =~ ac1 + ac2 + ac3 + ac4
  ru =~ ru1 + ru2 + ru3 + ru4
  rf =~ rf1 + rf2 + rf3 + rf4
  rp =~ rp1 + rp2 + rp3 + rp4
  ra =~ ra1 + ra2 + ra3 + ra4
  pp =~ pp1 + pp2 + pp3 + pp4
  ca =~ ca1 + ca2 + ca3 + ca4
  bo =~ bo1 + bo2 + bo3 + bo4
  scl =~ d1 + d2 + d3 + d4 + d5 + d6 + d7 + d8 + d9 + d10 + d11 + d12 + d13 + d14 + d15  + d16
  # regressions
  scl ~ sb + ac + ru + rf + rp + ra + pp + ca + bo
'

xnames = colnames(CERQi)
ynames = colnames(SCLi)
```

Let us fit this model to the data and inspect the results:

```
fit <- sem(model, data = mydat3, std.lv = TRUE, meanstructure = TRUE, warn = FALSE)
summary(fit)
```

```
## lavaan 0.6-9 ended normally after 70 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                       201
##
##   Number of observations                           240
##
## Model Test User Model:
##
##   Test statistic                              2250.529
##   Degrees of freedom                              1229
```

```
##   P-value (Chi-square)                           0.000
##
## Parameter Estimates:
##
##   Standard errors                             Standard
##   Information                                 Expected
##   Information saturated (h1) model          Structured
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   sb =~
##     sb1               0.390    0.049    7.933    0.000
##     sb2               0.708    0.075    9.392    0.000
##     sb3               0.752    0.066   11.318    0.000
##     sb4               0.670    0.058   11.464    0.000
##   ac =~
##     ac1               0.836    0.071   11.856    0.000
##     ac2               0.860    0.071   12.056    0.000
##     ac3               0.470    0.074    6.377    0.000
##     ac4               0.920    0.067   13.833    0.000
##   ru =~
##     ru1               0.755    0.062   12.097    0.000
##     ru2               0.780    0.058   13.325    0.000
##     ru3               0.977    0.073   13.408    0.000
##     ru4               0.864    0.063   13.791    0.000
##   rf =~
##     rf1               0.858    0.063   13.528    0.000
##     rf2               0.834    0.054   15.524    0.000
##     rf3               0.773    0.049   15.657    0.000
##     rf4               0.725    0.064   11.348    0.000
##   rp =~
##     rp1               0.872    0.065   13.329    0.000
##     rp2               0.943    0.057   16.664    0.000
##     rp3               0.955    0.063   15.105    0.000
##     rp4               0.976    0.064   15.307    0.000
##   ra =~
##     ra1               0.945    0.067   14.082    0.000
##     ra2               0.865    0.077   11.160    0.000
##     ra3               0.959    0.065   14.855    0.000
##     ra4               0.970    0.063   15.301    0.000
##   pp =~
##     pp1               0.831    0.070   11.922    0.000
##     pp2               0.760    0.071   10.716    0.000
##     pp3               0.951    0.064   14.826    0.000
##     pp4               0.953    0.064   14.965    0.000
##   ca =~
##     ca1               0.214    0.051    4.236    0.000
##     ca2               0.532    0.039   13.655    0.000
```

```
##     ca3               0.463    0.049    9.455    0.000
##     ca4               0.688    0.053   12.910    0.000
##   bo =~
##     bo1               0.428    0.041   10.535    0.000
##     bo2               0.632    0.043   14.801    0.000
##     bo3               0.568    0.055   10.289    0.000
##     bo4               0.662    0.050   13.254    0.000
##   scl =~
##     d1                0.606    0.046   13.095    0.000
##     d2                0.548    0.056    9.855    0.000
##     d3                0.530    0.043   12.350    0.000
##     d4                0.254    0.040    6.280    0.000
##     d5                0.229    0.030    7.712    0.000
##     d6                0.510    0.040   12.599    0.000
##     d7                0.590    0.042   13.981    0.000
##     d8                0.550    0.042   13.111    0.000
##     d9                0.647    0.041   15.762    0.000
##     d10               0.718    0.052   13.782    0.000
##     d11               0.433    0.034   12.703    0.000
##     d12               0.594    0.041   14.369    0.000
##     d13               0.625    0.041   15.254    0.000
##     d14               0.358    0.040    9.029    0.000
##     d15               0.533    0.040   13.385    0.000
##     d16               0.450    0.049    9.177    0.000
##
## Regressions:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   scl ~
##     sb                0.348    0.128    2.720    0.007
##     ac               -0.051    0.126   -0.399    0.690
##     ru                0.050    0.170    0.296    0.767
##     rf                0.066    0.106    0.626    0.531
##     rp                0.034    0.236    0.143    0.886
##     ra               -0.316    0.279   -1.136    0.256
##     pp                0.010    0.184    0.053    0.958
##     ca                0.619    0.213    2.912    0.004
##     bo               -0.041    0.136   -0.298    0.765
##
## Covariances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   sb ~~
##     ac                0.467    0.069    6.775    0.000
##     ru                0.425    0.069    6.132    0.000
##     rf                0.153    0.077    1.975    0.048
##     rp                0.466    0.065    7.194    0.000
##     ra                0.455    0.067    6.821    0.000
##     pp                0.445    0.068    6.569    0.000
##     ca                0.282    0.079    3.597    0.000
```

```
##     bo                 0.101    0.080    1.257    0.209
##   ac ~~
##     ru                 0.471    0.064    7.367    0.000
##     rf                 0.356    0.068    5.216    0.000
##     rp                 0.493    0.060    8.169    0.000
##     ra                 0.492    0.062    7.968    0.000
##     pp                 0.513    0.061    8.426    0.000
##     ca                 0.355    0.073    4.873    0.000
##     bo                 0.256    0.074    3.463    0.001
##   ru ~~
##     rf                 0.109    0.074    1.474    0.141
##     rp                 0.652    0.047   13.832    0.000
##     ra                 0.418    0.064    6.499    0.000
##     pp                 0.142    0.074    1.912    0.056
##     ca                 0.531    0.062    8.638    0.000
##     bo                 0.327    0.070    4.688    0.000
##   rf ~~
##     rp                 0.324    0.066    4.916    0.000
##     ra                 0.416    0.063    6.619    0.000
##     pp                 0.564    0.054   10.414    0.000
##     ca                 0.022    0.077    0.289    0.772
##     bo                 0.063    0.075    0.847    0.397
##   rp ~~
##     ra                 0.811    0.032   25.460    0.000
##     pp                 0.430    0.062    6.948    0.000
##     ca                 0.105    0.076    1.382    0.167
##     bo                 0.114    0.073    1.558    0.119
##   ra ~~
##     pp                 0.685    0.045   15.076    0.000
##     ca                -0.145    0.077   -1.884    0.060
##     bo                -0.070    0.075   -0.932    0.351
##   pp ~~
##     ca                -0.070    0.078   -0.893    0.372
##     bo                 0.060    0.076    0.791    0.429
##   ca ~~
##     bo                 0.664    0.052   12.693    0.000
##
## Intercepts:
##                     Estimate  Std.Err  z-value  P(>|z|)
##    .sb1                1.733    0.047   36.636    0.000
##    .sb2                2.304    0.074   30.972    0.000
##    .sb3                2.304    0.068   33.984    0.000
##    .sb4                1.746    0.060   29.229    0.000
##    .ac1                2.658    0.075   35.445    0.000
##    .ac2                2.967    0.076   38.974    0.000
##    .ac3                2.446    0.071   34.512    0.000
##    .ac4                2.742    0.073   37.373    0.000
##    .ru1                2.492    0.068   36.530    0.000
```

```
##    .ru2              2.417    0.066   36.805    0.000
##    .ru3              2.646    0.082   32.283    0.000
##    .ru4              2.733    0.071   38.460    0.000
##    .rf1              2.500    0.072   34.757    0.000
##    .rf2              2.325    0.064   36.583    0.000
##    .rf3              2.183    0.059   37.245    0.000
##    .rf4              2.858    0.069   41.354    0.000
##    .rp1              3.487    0.075   46.647    0.000
##    .rp2              3.279    0.070   47.007    0.000
##    .rp3              3.000    0.075   39.877    0.000
##    .rp4              3.075    0.076   40.342    0.000
##    .ra1              3.138    0.078   40.338    0.000
##    .ra2              3.129    0.084   37.135    0.000
##    .ra3              2.888    0.076   37.921    0.000
##    .ra4              3.125    0.075   41.396    0.000
##    .pp1              2.596    0.076   34.099    0.000
##    .pp2              2.862    0.076   37.906    0.000
##    .pp3              2.783    0.075   37.300    0.000
##    .pp4              3.125    0.074   42.041    0.000
##    .ca1              1.433    0.048   30.063    0.000
##    .ca2              1.367    0.043   31.601    0.000
##    .ca3              1.350    0.050   26.944    0.000
##    .ca4              1.775    0.058   30.440    0.000
##    .bo1              1.408    0.042   33.146    0.000
##    .bo2              1.479    0.049   30.453    0.000
##    .bo3              1.771    0.057   30.801    0.000
##    .bo4              1.554    0.055   28.172    0.000
##    .d1               1.762    0.065   27.222    0.000
##    .d2               2.150    0.075   28.818    0.000
##    .d3               1.454    0.059   24.465    0.000
##    .d4               1.158    0.052   22.118    0.000
##    .d5               1.254    0.039   32.179    0.000
##    .d6               1.413    0.056   25.117    0.000
##    .d7               1.529    0.060   25.583    0.000
##    .d8               1.571    0.059   26.739    0.000
##    .d9               1.504    0.060   25.152    0.000
##    .d10              2.083    0.074   28.292    0.000
##    .d11              1.333    0.047   28.138    0.000
##    .d12              1.512    0.059   25.670    0.000
##    .d13              1.467    0.059   24.783    0.000
##    .d14              1.471    0.053   27.925    0.000
##    .d15              1.375    0.056   24.580    0.000
##    .d16              1.613    0.065   24.726    0.000
##     sb               0.000
##     ac               0.000
##     ru               0.000
##     rf               0.000
##     rp               0.000
```

```
##     ra                  0.000
##     pp                  0.000
##     ca                  0.000
##     bo                  0.000
##    .scl                 0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .sb1                 0.385    0.039    9.822    0.000
##    .sb2                 0.827    0.090    9.202    0.000
##    .sb3                 0.537    0.068    7.861    0.000
##    .sb4                 0.408    0.053    7.724    0.000
##    .ac1                 0.651    0.076    8.558    0.000
##    .ac2                 0.651    0.077    8.416    0.000
##    .ac3                 0.985    0.094   10.475    0.000
##    .ac4                 0.445    0.066    6.718    0.000
##    .ru1                 0.547    0.059    9.280    0.000
##    .ru2                 0.427    0.049    8.655    0.000
##    .ru3                 0.657    0.076    8.604    0.000
##    .ru4                 0.465    0.056    8.352    0.000
##    .rf1                 0.505    0.057    8.856    0.000
##    .rf2                 0.274    0.038    7.213    0.000
##    .rf3                 0.226    0.032    7.065    0.000
##    .rf4                 0.621    0.064    9.758    0.000
##    .rp1                 0.582    0.060    9.643    0.000
##    .rp2                 0.279    0.036    7.699    0.000
##    .rp3                 0.446    0.050    8.880    0.000
##    .rp4                 0.442    0.050    8.762    0.000
##    .ra1                 0.559    0.061    9.145    0.000
##    .ra2                 0.957    0.095   10.086    0.000
##    .ra3                 0.471    0.054    8.728    0.000
##    .ra4                 0.427    0.051    8.430    0.000
##    .pp1                 0.701    0.074    9.475    0.000
##    .pp2                 0.791    0.080    9.866    0.000
##    .pp3                 0.432    0.056    7.743    0.000
##    .pp4                 0.417    0.055    7.613    0.000
##    .ca1                 0.500    0.046   10.777    0.000
##    .ca2                 0.166    0.022    7.424    0.000
##    .ca3                 0.388    0.039    9.863    0.000
##    .ca4                 0.343    0.042    8.139    0.000
##    .bo1                 0.250    0.026    9.554    0.000
##    .bo2                 0.167    0.027    6.295    0.000
##    .bo3                 0.471    0.049    9.644    0.000
##    .bo4                 0.292    0.037    7.953    0.000
##    .d1                  0.399    0.039   10.282    0.000
##    .d2                  0.839    0.078   10.693    0.000
##    .d3                  0.383    0.037   10.419    0.000
##    .d4                  0.551    0.051   10.869    0.000
```

```
##    .d5                 0.278    0.026    10.817    0.000
##    .d6                 0.329    0.032    10.378    0.000
##    .d7                 0.282    0.028    10.055    0.000
##    .d8                 0.327    0.032    10.279    0.000
##    .d9                 0.165    0.018     9.100    0.000
##    .d10                0.448    0.044    10.114    0.000
##    .d11                0.229    0.022    10.359    0.000
##    .d12                0.250    0.025     9.922    0.000
##    .d13                0.195    0.021     9.489    0.000
##    .d14                0.454    0.042    10.749    0.000
##    .d15                0.281    0.028    10.217    0.000
##    .d16                0.686    0.064    10.740    0.000
##     sb                 1.000
##     ac                 1.000
##     ru                 1.000
##     rf                 1.000
##     rp                 1.000
##     ra                 1.000
##     pp                 1.000
##     ca                 1.000
##     bo                 1.000
##    .scl                1.000
```

Repeated Cross Validation

```
set.seed(1234)
repeats = 100
PE = data.frame(repetition = rep(1:repeats, each = 14),
                model = rep(1:14, repeats),
                pe = rep(0, 14 * repeats),
                pe1 = rep(0, 14 * repeats),
                pe2 = rep(0, 14 * repeats),
                pe3 = rep(0, 14 * repeats),
                pe4 = rep(0, 14 * repeats),
                pe5 = rep(0, 14 * repeats),
                pe6 = rep(0, 14 * repeats),
                pe7 = rep(0, 14 * repeats),
                pe8 = rep(0, 14 * repeats),
                pe9 = rep(0, 14 * repeats),
                pe10 = rep(0, 14 * repeats),
                pe11 = rep(0, 14 * repeats),
                pe12 = rep(0, 14 * repeats),
                pe13 = rep(0, 14 * repeats),
                pe14 = rep(0, 14 * repeats),
                pe15 = rep(0, 14 * repeats),
                pe16 = rep(0, 14 * repeats))
```

```
folds = rep(1:10, length.out = 240)
for (r in 1:repeats){
  yhat1 = yhat2 = yhat3 = yhat4 = yhat5 = yhat6 = yhat7 = yhat8 = yhat9 = yhat10 = yhat11 = yhat
  folds = sample(folds)
  for(k in 1:10){
    idx = which(folds == k)
    # sem model
    fit.sem <- sem(model, data = mydat3[-idx, ], std.lv = TRUE, meanstructure = TRUE, warn = FAL
    yhat1[idx, ] = predicty.lavaan(fit.sem, newdata = mydat3[idx, ], xnames = xnames, ynames = y

    # linear regession scale scores
    fit.lms = lm(cbind(d1, d2, d3, d4, d5, d6, d7, d8, d9, d10, d11, d12, d13, d14, d15, d16) ~
    yhat2[idx,] = predict(fit.lms, newdata = mydat2[idx, ])

    # elastic net on items
    cv.out = cv.glmnet(x[-idx, ],y[-idx,], family = "mgaussian", alpha = 1.0)
    out = glmnet(x[-idx, ],y[-idx,], family = "mgaussian", alpha = 1.0)
    yhat3[idx, ] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx,], family = "mgaussian", alpha = 0.9)
    out = glmnet(x[-idx, ],y[-idx,], family = "mgaussian", alpha = 0.9)
    yhat4[idx, ] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx,], family = "mgaussian", alpha = 0.8)
    out = glmnet(x[-idx, ],y[-idx,], family = "mgaussian", alpha = 0.8)
    yhat5[idx, ] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx,], family = "mgaussian", alpha = 0.7)
    out = glmnet(x[-idx, ],y[-idx,], family = "mgaussian", alpha = 0.7)
    yhat6[idx, ] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx,], family = "mgaussian", alpha = 0.6)
    out = glmnet(x[-idx, ],y[-idx,], family = "mgaussian", alpha = 0.6)
    yhat7[idx, ] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx, ], family = "mgaussian", alpha = 0.5)
    out = glmnet(x[-idx, ],y[-idx, ], family = "mgaussian", alpha = 0.5)
    yhat8[idx, ] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx, ], family = "mgaussian", alpha = 0.4)
    out = glmnet(x[-idx, ],y[-idx, ], family = "mgaussian", alpha = 0.4)
    yhat9[idx, ] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

    cv.out = cv.glmnet(x[-idx, ],y[-idx, ], family = "mgaussian", alpha = 0.3)
    out = glmnet(x[-idx, ],y[-idx, ], family = "mgaussian", alpha = 0.3)
    yhat10[idx, ] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)
```

```
  cv.out = cv.glmnet(x[-idx, ],y[-idx, ], family = "mgaussian", alpha = 0.2)
  out = glmnet(x[-idx, ],y[-idx, ], family = "mgaussian", alpha = 0.2)
  yhat11[idx, ] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

  cv.out = cv.glmnet(x[-idx, ],y[-idx, ], family = "mgaussian", alpha = 0.1)
  out = glmnet(x[-idx, ],y[-idx, ], family = "mgaussian", alpha = 0.1)
  yhat12[idx, ] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

  cv.out = cv.glmnet(x[-idx, ],y[-idx, ], family = "mgaussian", alpha = 0.0)
  out = glmnet(x[-idx, ],y[-idx, ], family = "mgaussian", alpha = 0.0)
  yhat13[idx, ] = predict(out, newx = x[idx, ], s = cv.out$lambda.1se)

  out = lm(cbind(d1, d2, d3, d4, d5, d6, d7, d8, d9, d10, d11, d12, d13, d14, d15, d16) ~ ., d
  yhat14[idx, ] = predict(out, newdata = mydat3[idx, ])

}# end folds

pe1 = sqrt(mean((y - yhat1)^2))
pe2 = sqrt(mean((y - yhat2)^2))
pe3 = sqrt(mean((y - yhat3)^2))
pe4 = sqrt(mean((y - yhat4)^2))
pe5 = sqrt(mean((y - yhat5)^2))
pe6 = sqrt(mean((y - yhat6)^2))
pe7 = sqrt(mean((y - yhat7)^2))
pe8 = sqrt(mean((y - yhat8)^2))
pe9 = sqrt(mean((y - yhat9)^2))
pe10 = sqrt(mean((y - yhat10)^2))
pe11 = sqrt(mean((y - yhat11)^2))
pe12 = sqrt(mean((y - yhat12)^2))
pe13 = sqrt(mean((y - yhat13)^2))
pe14 = sqrt(mean((y - yhat14)^2))

PE$pe[((r-1)*14 + 1): (r*14)] = c(pe1, pe2, pe3, pe4, pe5, pe6, pe7, pe8, pe9, pe10, pe11, pe1

for(j in 1:16){

  pe1 = sqrt(mean((y[, j] - yhat1[,j])^2))
  pe2 = sqrt(mean((y[, j] - yhat2[,j])^2))
  pe3 = sqrt(mean((y[, j] - yhat3[,j])^2))
  pe4 = sqrt(mean((y[, j] - yhat4[,j])^2))
  pe5 = sqrt(mean((y[, j] - yhat5[,j])^2))
  pe6 = sqrt(mean((y[, j] - yhat6[,j])^2))
  pe7 = sqrt(mean((y[, j] - yhat7[,j])^2))
  pe8 = sqrt(mean((y[, j] - yhat8[,j])^2))
  pe9 = sqrt(mean((y[, j] - yhat9[,j])^2))
  pe10 = sqrt(mean((y[, j] - yhat10[,j])^2))
  pe11 = sqrt(mean((y[, j] - yhat11[,j])^2))
```

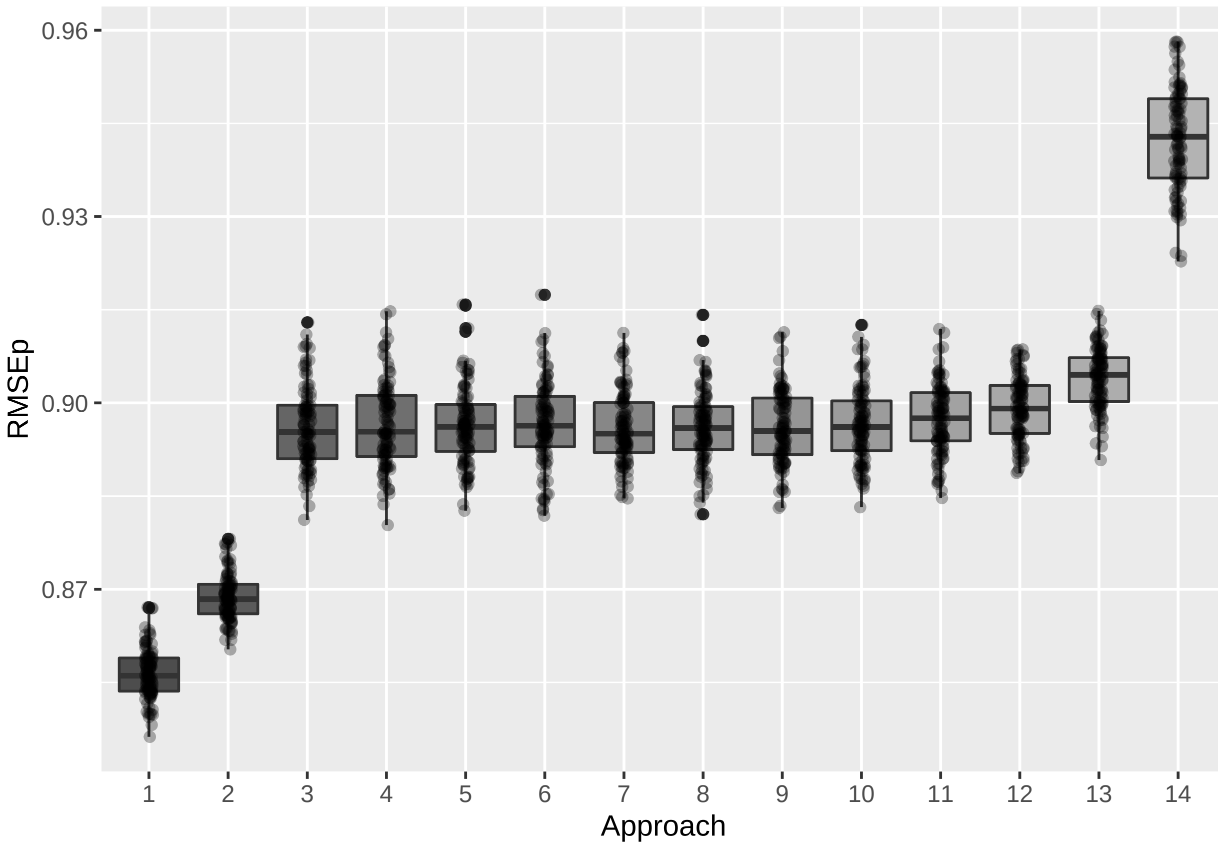
```
    pe12 = sqrt(mean((y[, j] - yhat12[,j])^2))
    pe13 = sqrt(mean((y[, j] - yhat13[,j])^2))
    pe14 = sqrt(mean((y[, j] - yhat14[,j])^2))

    PE[((r-1)*14 + 1): (r*14), (j+3)] = c(pe1, pe2, pe3, pe4, pe5, pe6, pe7, pe8, pe9, pe10, pe1
  }


} # end repetitions
save(PE, file = "xvalcerqitem.Rdata")
```

```
pe = cbind(PE[PE$model == 1, 3],
           PE[PE$model == 2, 3],
           PE[PE$model == 3, 3],
           PE[PE$model == 4, 3],
           PE[PE$model == 5, 3],
           PE[PE$model == 6, 3],
           PE[PE$model == 7, 3],
           PE[PE$model == 8, 3],
           PE[PE$model == 9, 3],
           PE[PE$model == 10, 3],
           PE[PE$model == 11, 3],
           PE[PE$model == 12, 3],
           PE[PE$model == 13, 3],
           PE[PE$model == 14, 3])
table(apply(pe, 1, which.min))
```

```
##
##   1
## 100
```

Overall prediction error

```
library(ggplot2)
PE$model = as.factor(PE$model)

p <- ggplot(PE, aes(x=model, y=pe, fill=factor(model)))
    p <- p + geom_boxplot(aes(group = factor(model))) +
      geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
      xlab("Approach") + ylab("RMSEp") + theme(legend.position="none") +
      scale_fill_grey(start=.3,end=.7)

p
```

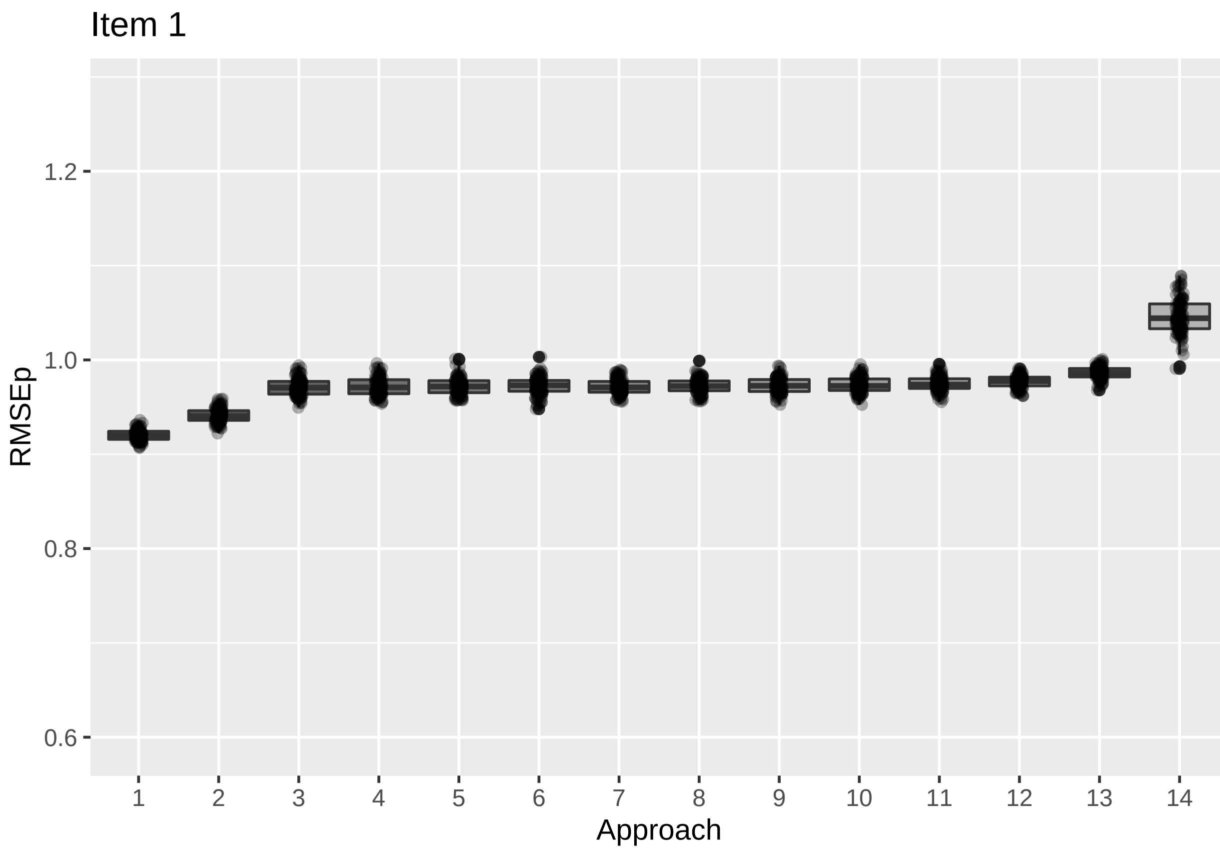



```
ggsave('~/surfdrive/Predictive-Psychometrics/paper/SEM-Predictive Validity/versie2/Figures/cerqi
```

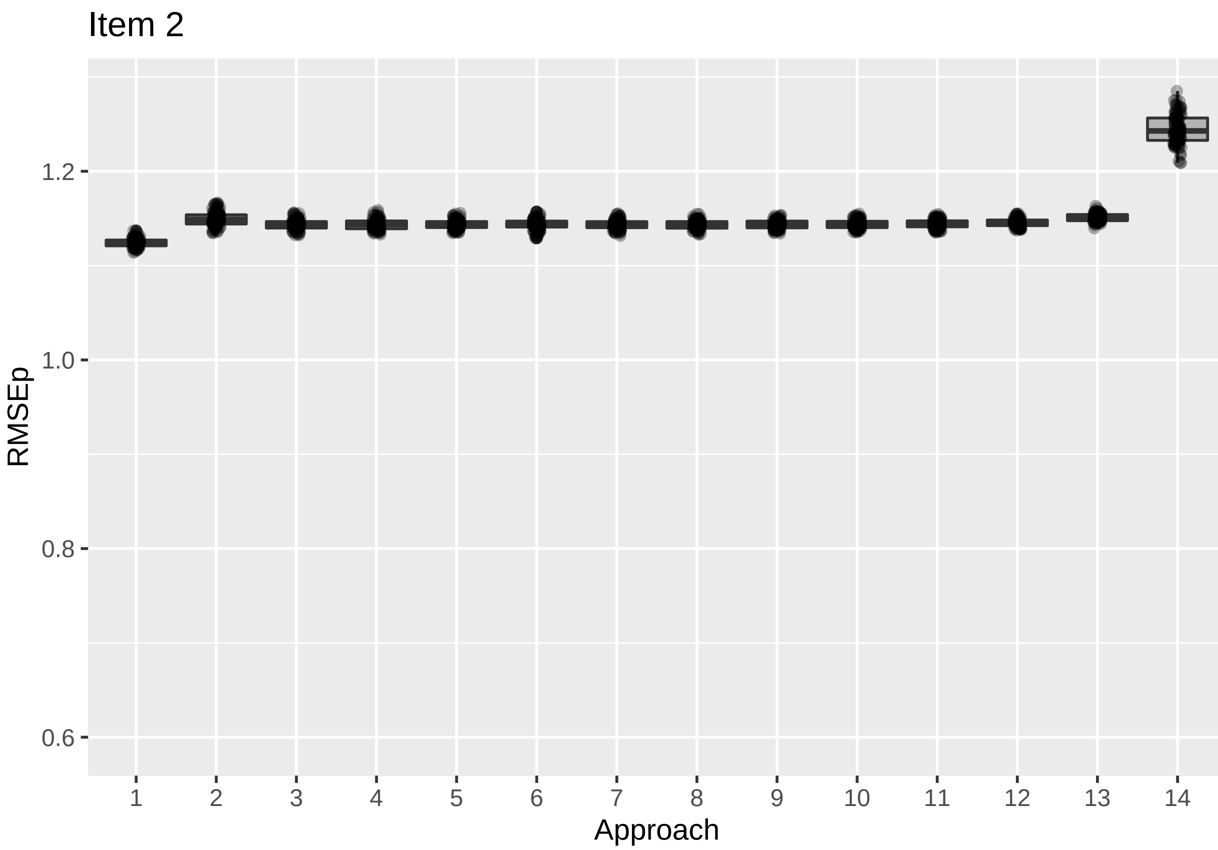
```
## Saving 6.5 x 4.5 in image
```

Prediction error per item

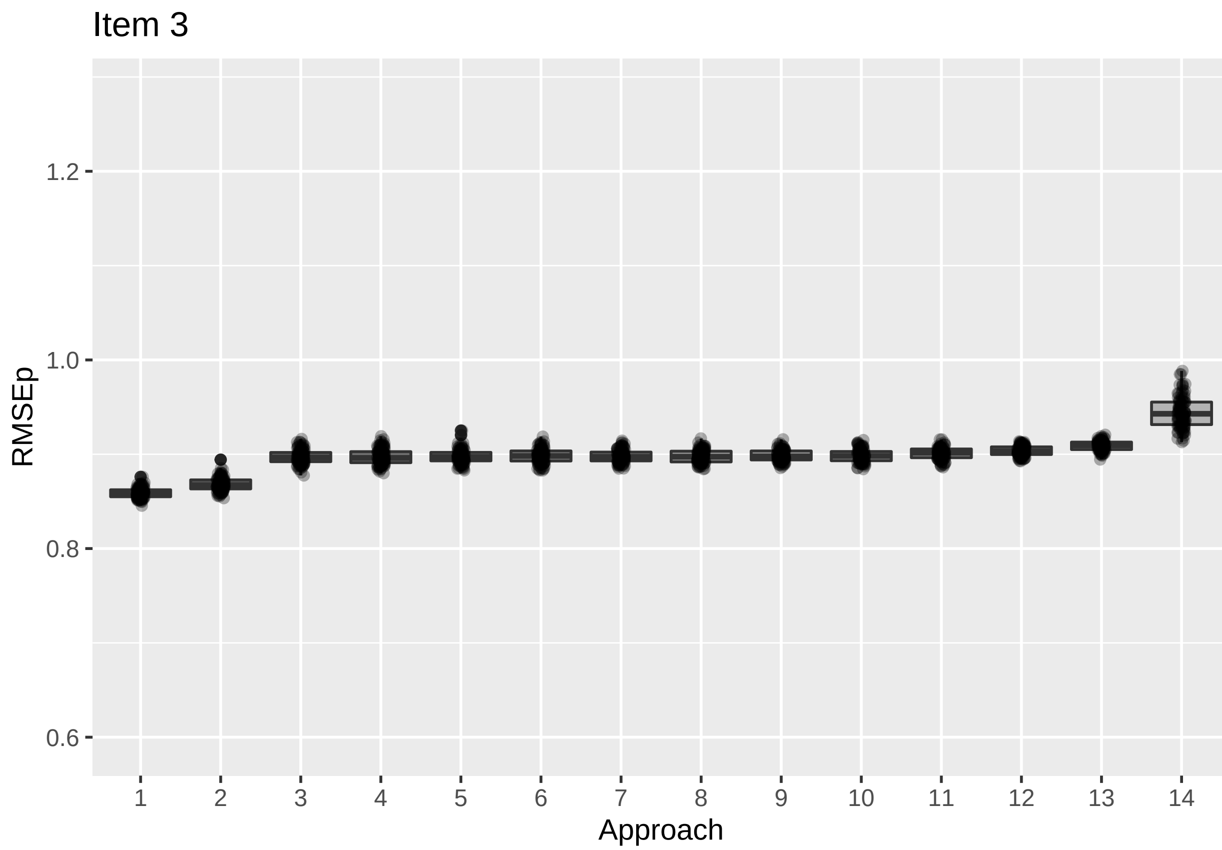
```
#load("~/surfdrive/Predictive-Psychometrics/paper/SEM-Predictive Validity/versie2/Rcode/xvalcerq
rpe = range(PE[, 4:19])

p <- ggplot(PE, aes(x=model, y=pe1, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 1") + ylim(rpe[1], rpe[2]) + theme(legen
  scale_fill_grey(start=.3,end=.7)
p
```



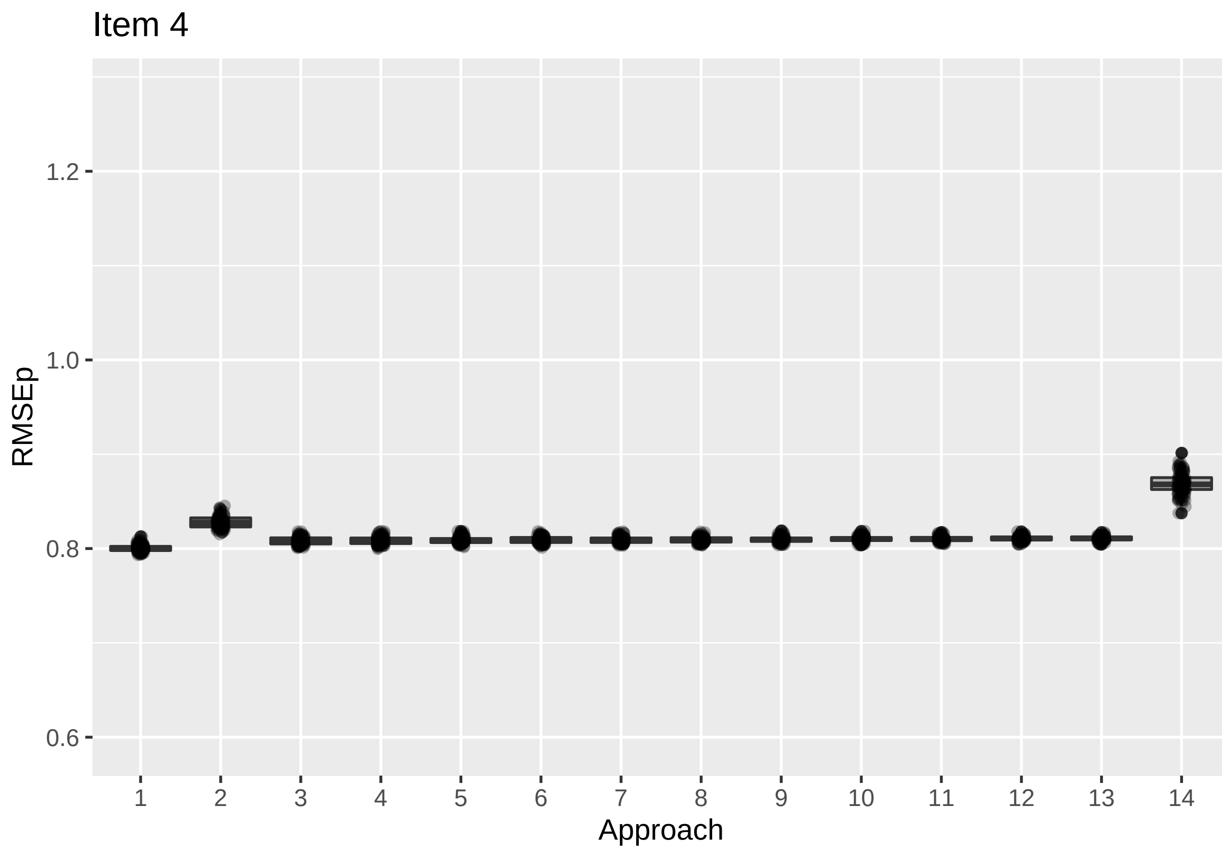

```
p <- ggplot(PE, aes(x=model, y=pe2, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 2") + ylim(rpe[1], rpe[2])+ theme(legend
  scale_fill_grey(start=.3,end=.7)
p
```



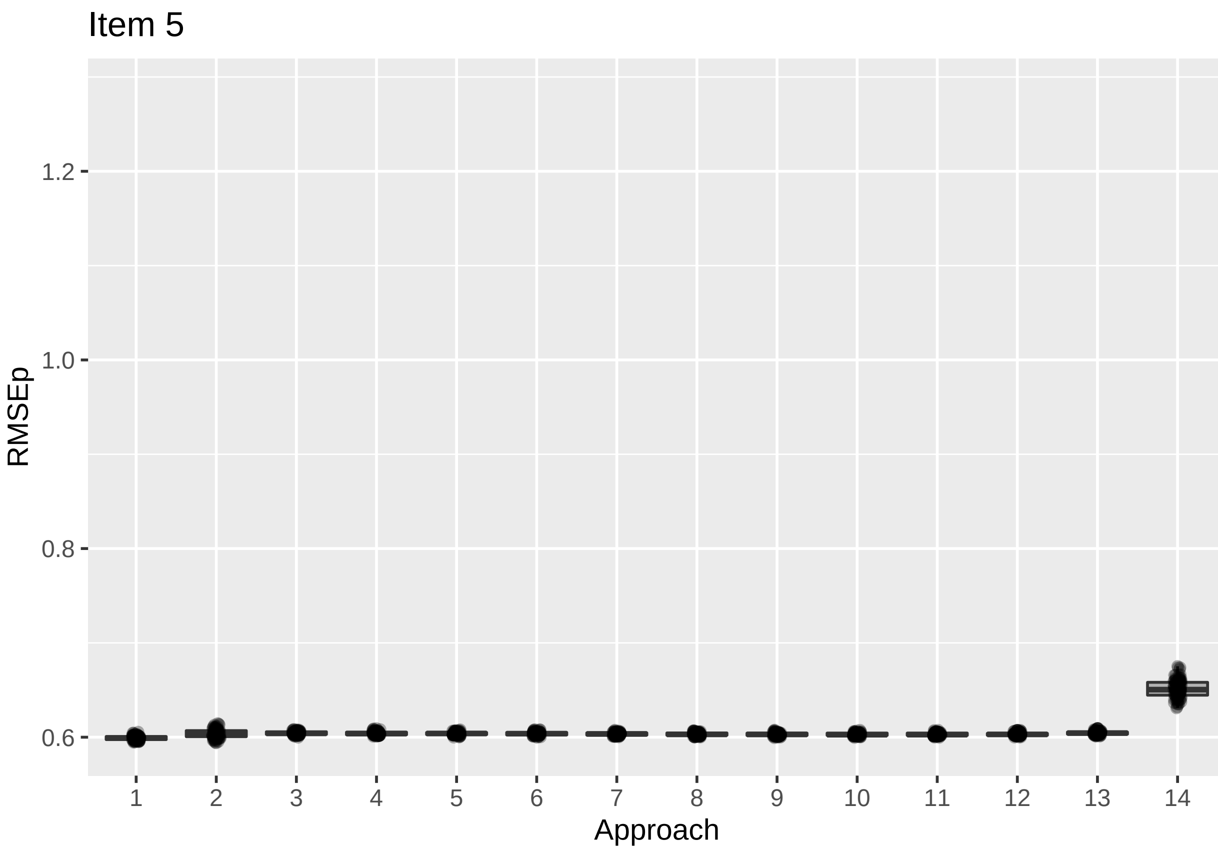

```
p <- ggplot(PE, aes(x=model, y=pe3, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 3") + ylim(rpe[1], rpe[2])+ theme(legend
  scale_fill_grey(start=.3,end=.7)
p
```



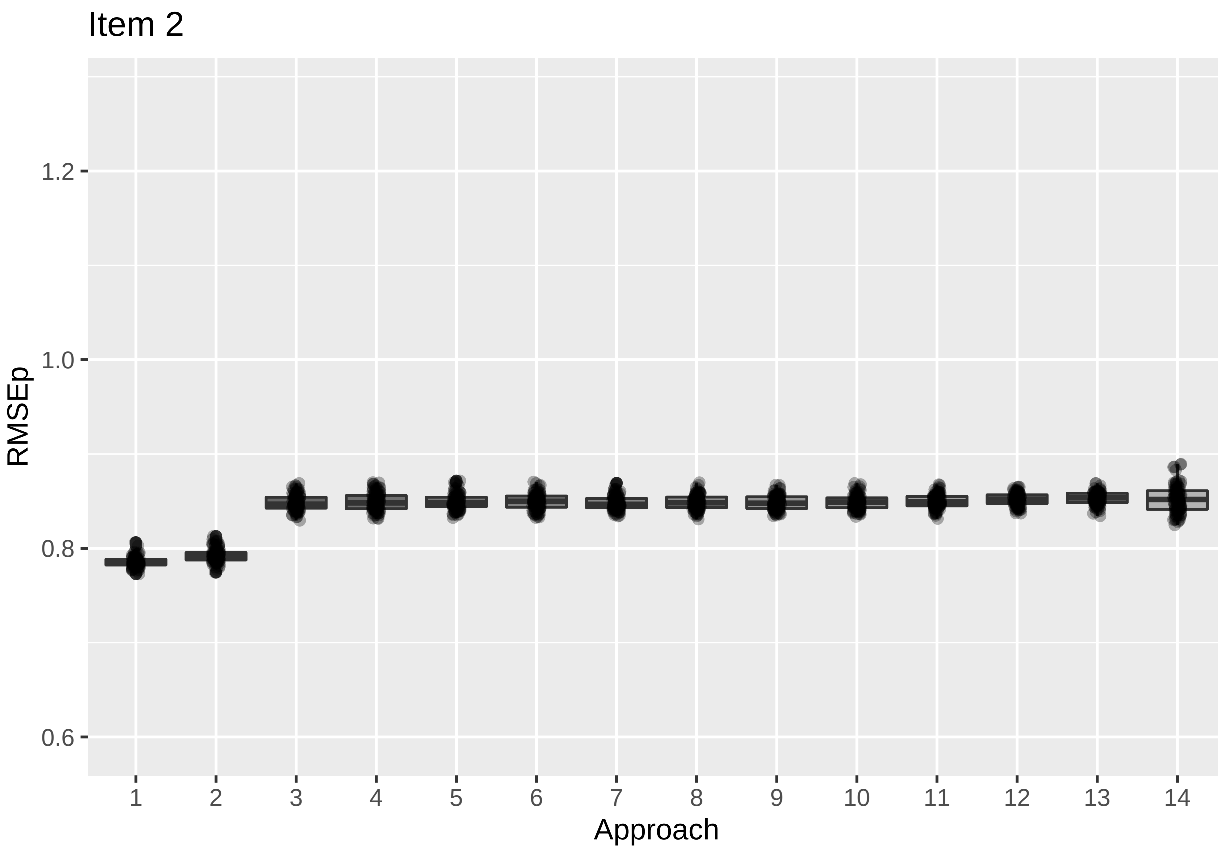

```
p <- ggplot(PE, aes(x=model, y=pe4, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 4") + ylim(rpe[1], rpe[2])+ theme(legend
  scale_fill_grey(start=.3,end=.7)
p
```



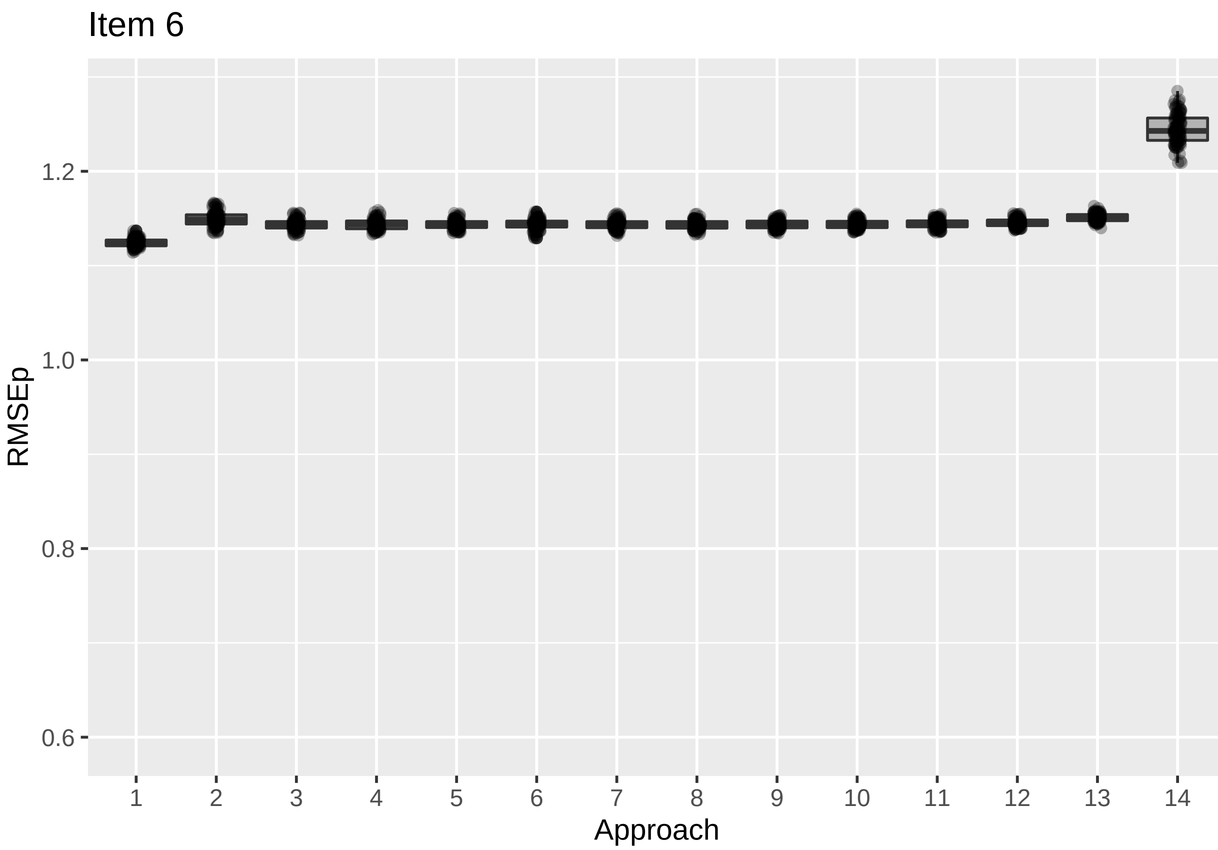

```r
p <- ggplot(PE, aes(x=model, y=pe5, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 5") + ylim(rpe[1], rpe[2])+ theme(legend
  scale_fill_grey(start=.3,end=.7)
p
```



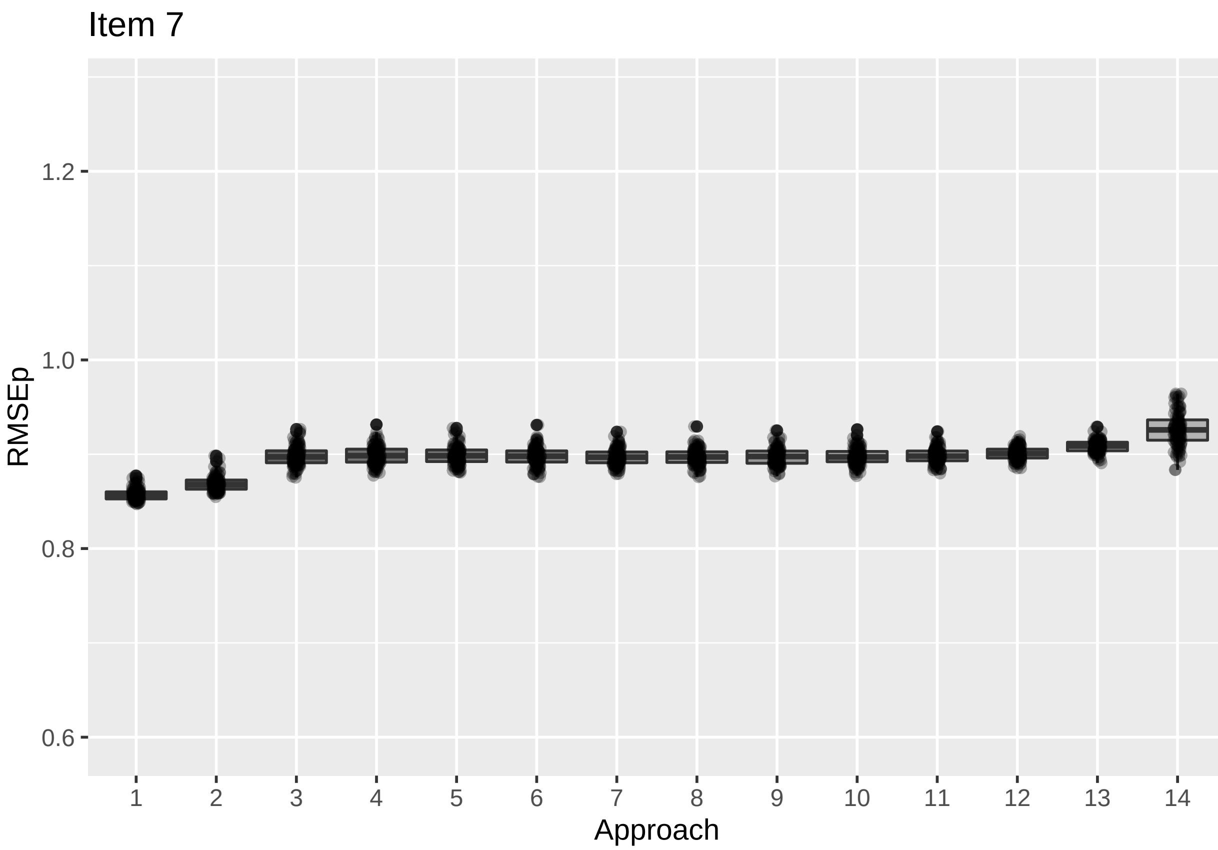

```
p <- ggplot(PE, aes(x=model, y=pe6, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 2") + ylim(rpe[1], rpe[2])+ theme(legend
  scale_fill_grey(start=.3,end=.7)
p
```



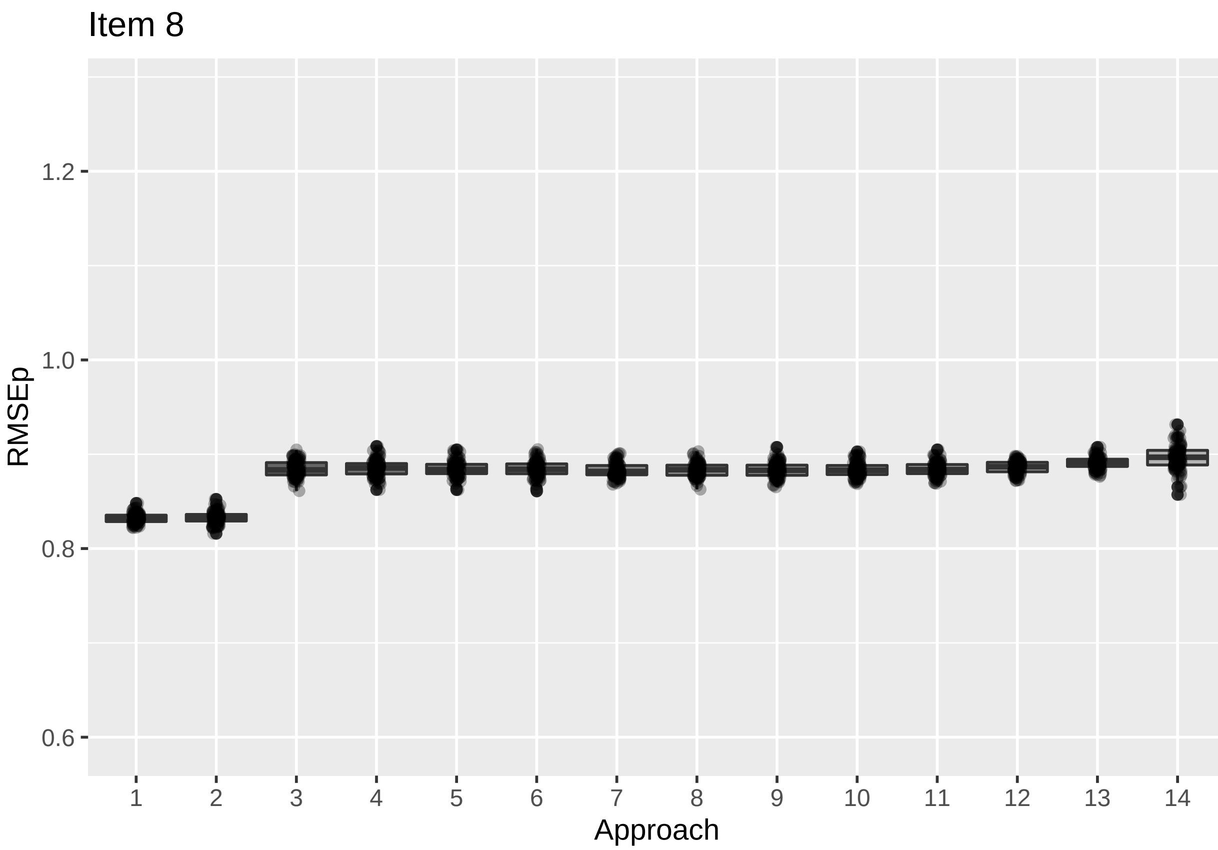

```
p <- ggplot(PE, aes(x=model, y=pe2, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 6") + ylim(rpe[1], rpe[2])+ theme(legend
  scale_fill_grey(start=.3,end=.7)
p
```



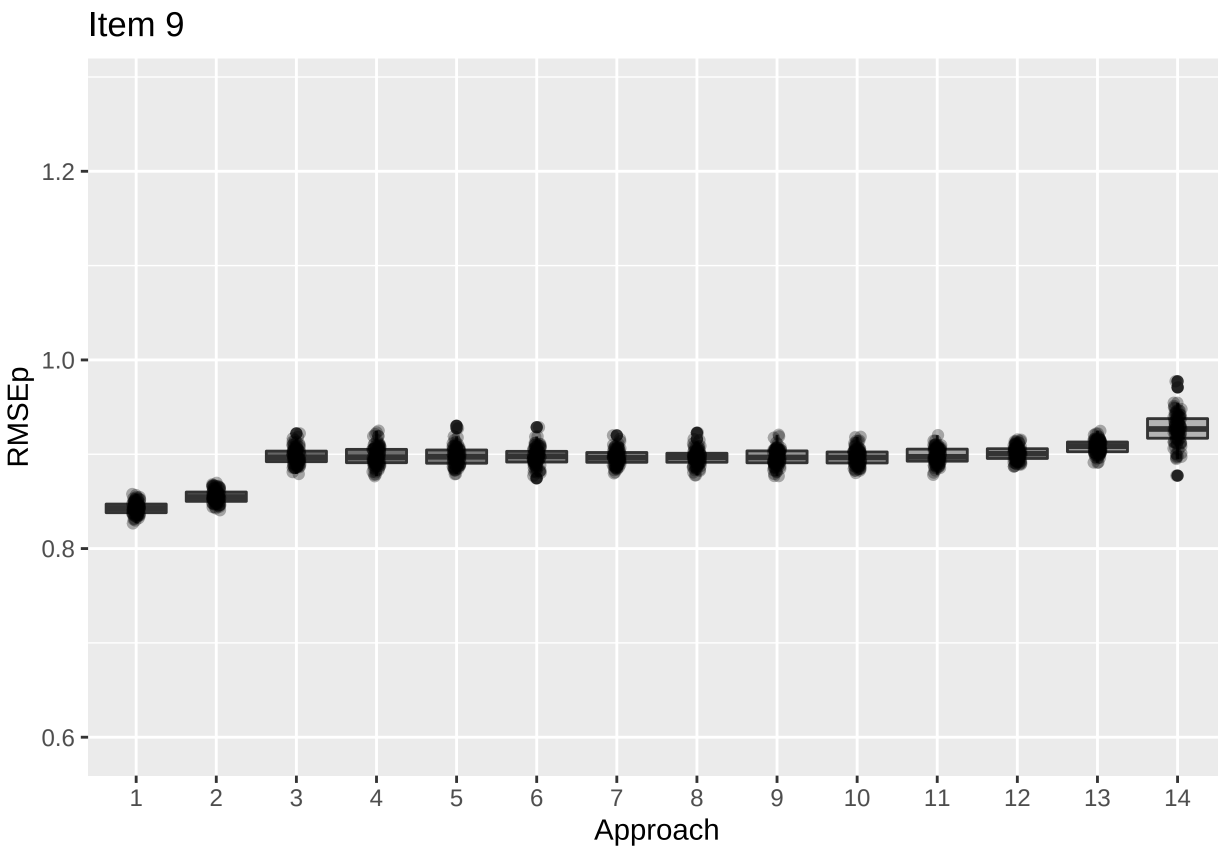

```
p <- ggplot(PE, aes(x=model, y=pe7, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 7") + ylim(rpe[1], rpe[2])+ theme(legend
  scale_fill_grey(start=.3,end=.7)
p
```



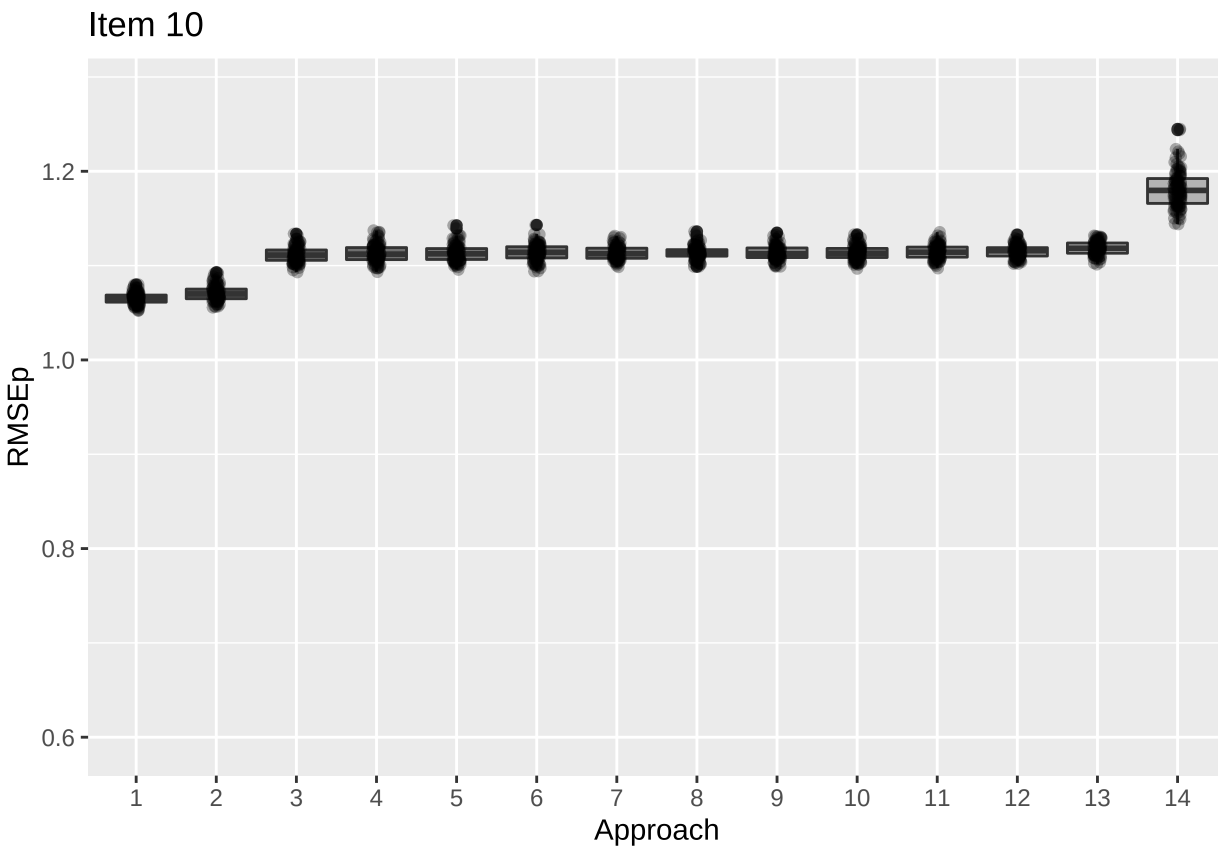

```
p <- ggplot(PE, aes(x=model, y=pe8, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 8") + ylim(rpe[1], rpe[2])+ theme(legend
  scale_fill_grey(start=.3,end=.7)
p
```



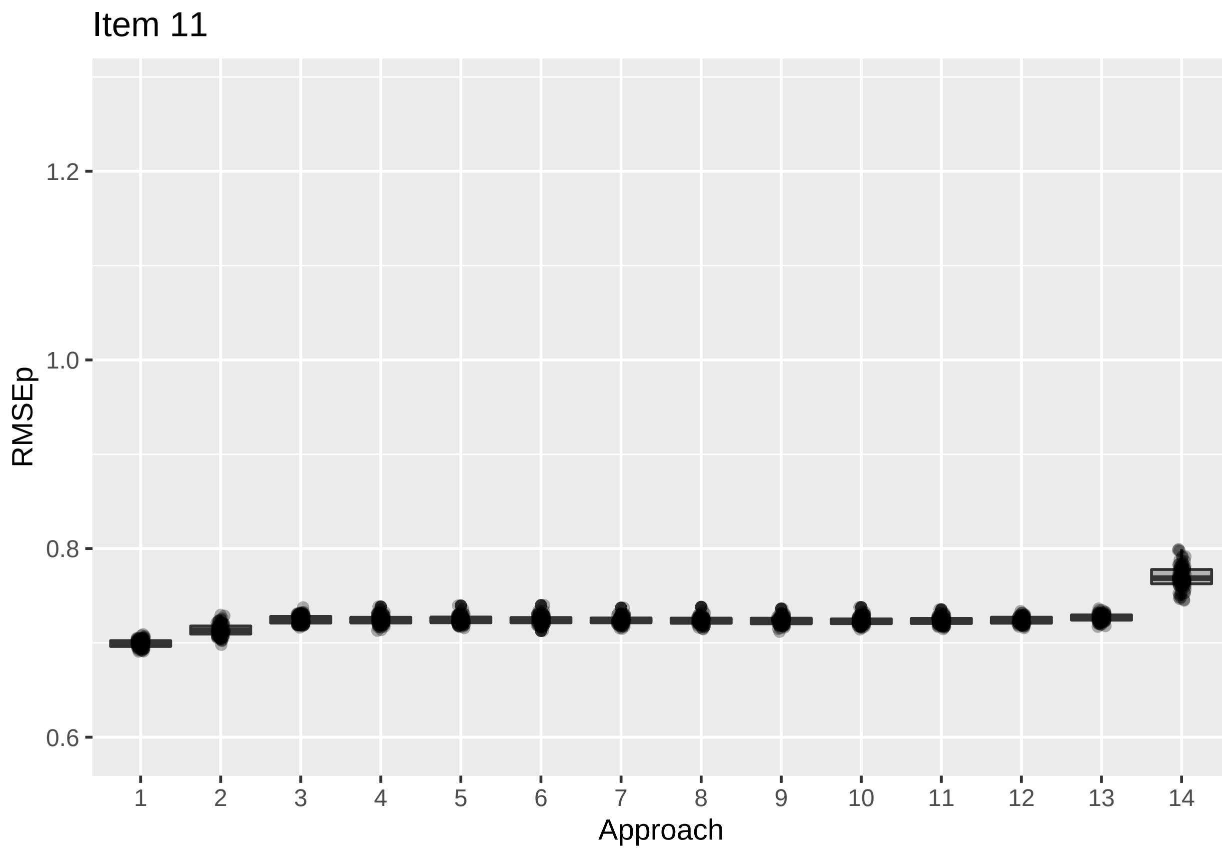

```
p <- ggplot(PE, aes(x=model, y=pe9, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 9") + ylim(rpe[1], rpe[2])+ theme(legend
  scale_fill_grey(start=.3,end=.7)
p
```



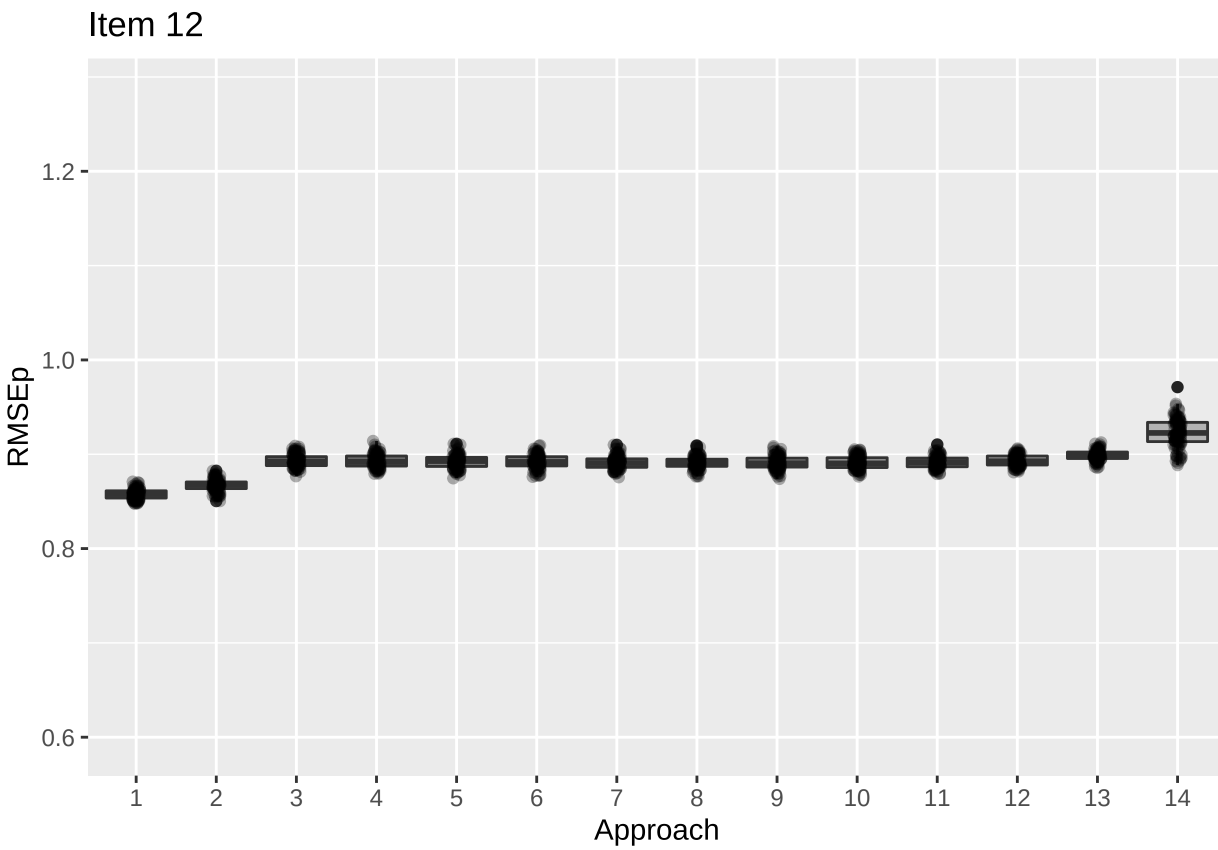

```
p <- ggplot(PE, aes(x=model, y=pe10, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 10") + ylim(rpe[1], rpe[2])+ theme(legen
  scale_fill_grey(start=.3,end=.7)
p
```



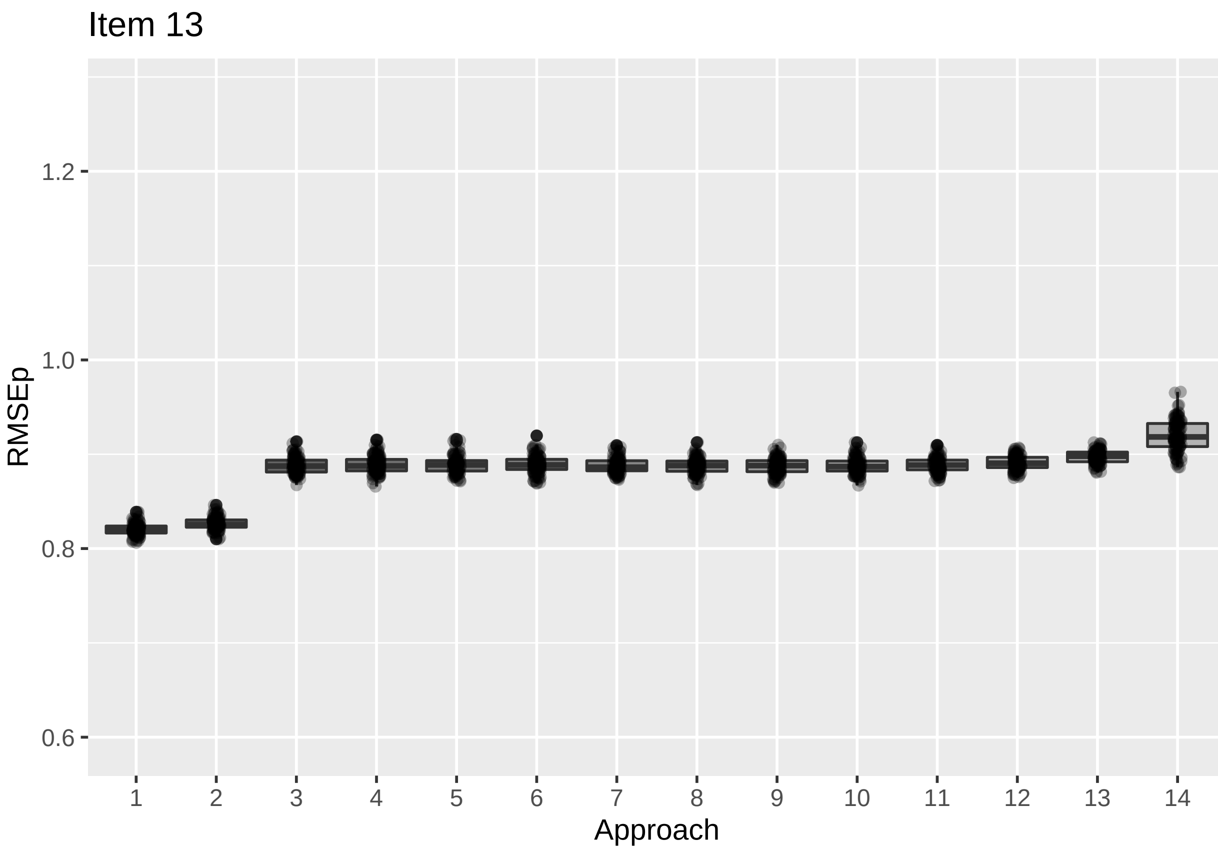

```
p <- ggplot(PE, aes(x=model, y=pe11, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 11") + ylim(rpe[1], rpe[2])+ theme(legen
  scale_fill_grey(start=.3,end=.7)
p
```



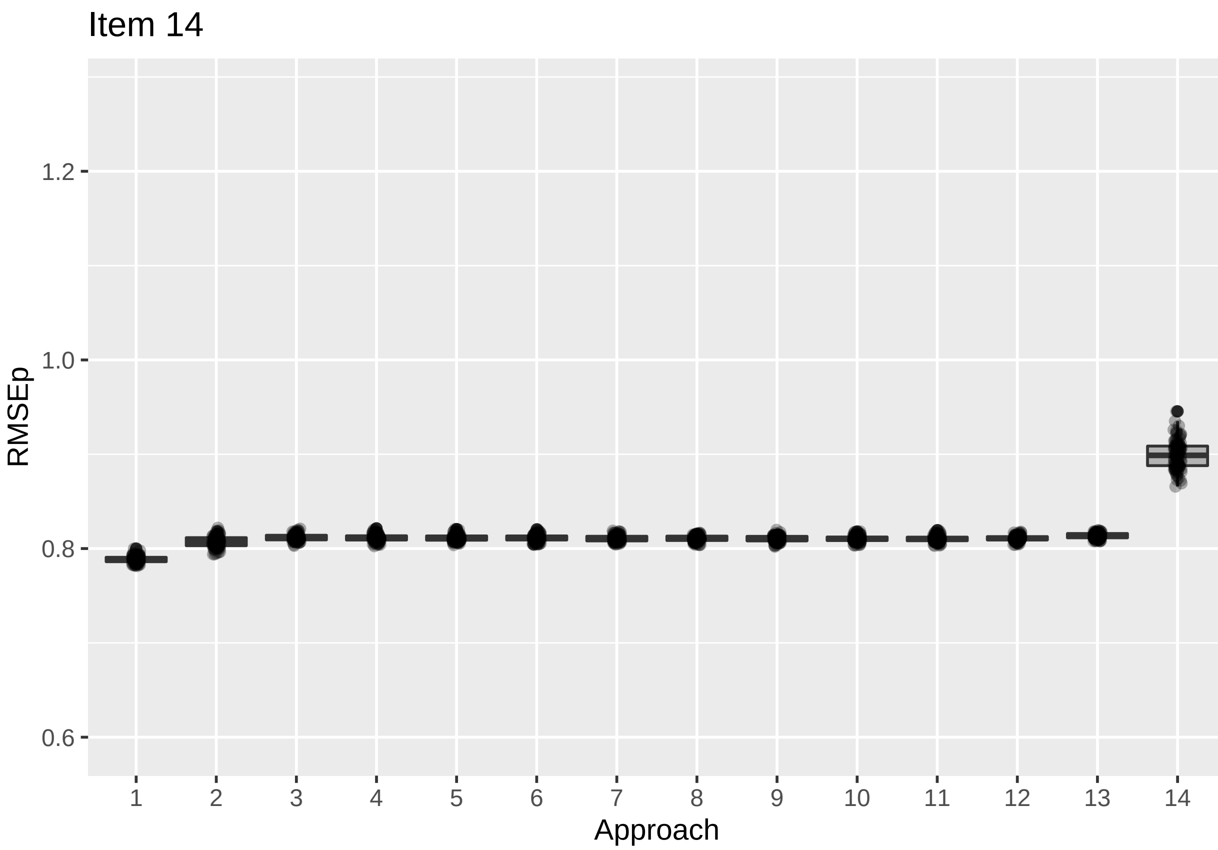

```
p <- ggplot(PE, aes(x=model, y=pe12, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 12") + ylim(rpe[1], rpe[2])+ theme(legen
  scale_fill_grey(start=.3,end=.7)
p
```



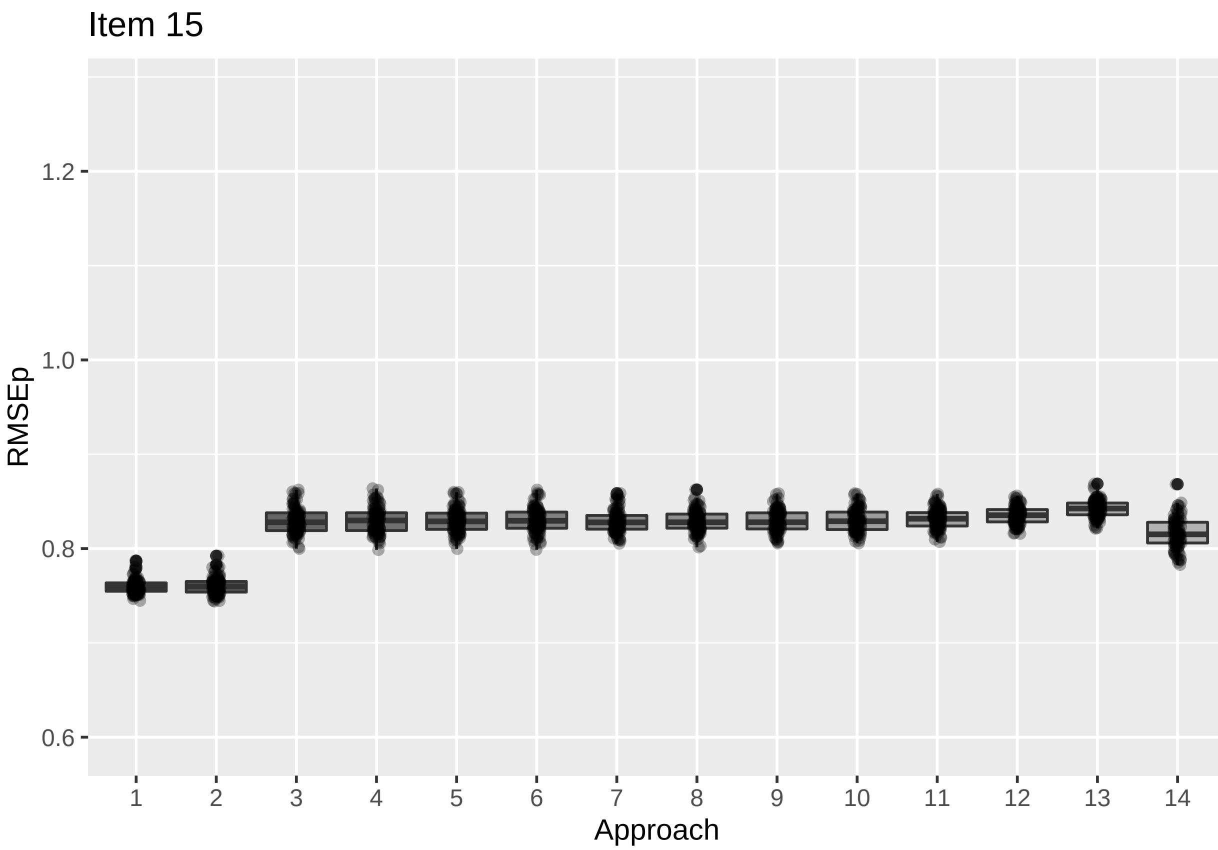

```
p <- ggplot(PE, aes(x=model, y=pe13, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 13") + ylim(rpe[1], rpe[2])+ theme(legen
  scale_fill_grey(start=.3,end=.7)
p
```



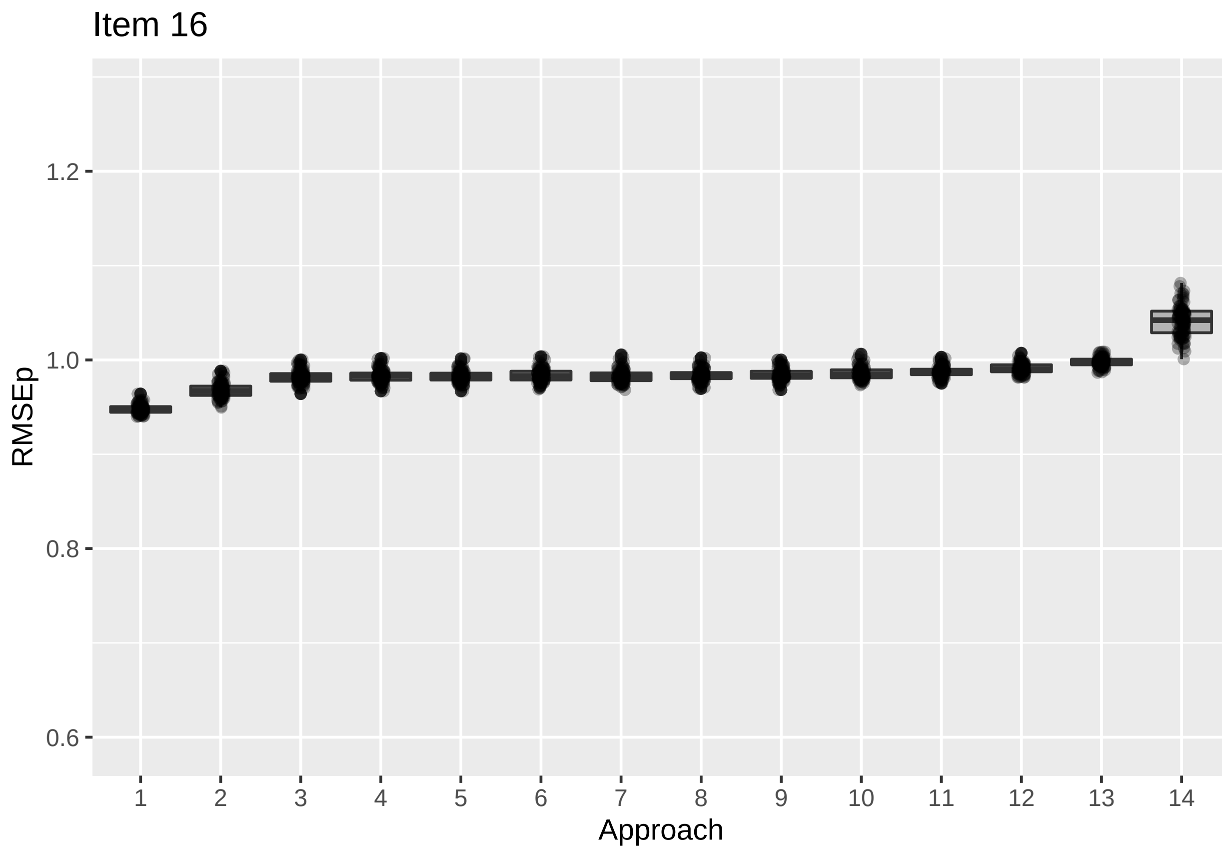

```
p <- ggplot(PE, aes(x=model, y=pe14, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 14") + ylim(rpe[1], rpe[2])+ theme(legen
  scale_fill_grey(start=.3,end=.7)
p
```




```
p <- ggplot(PE, aes(x=model, y=pe15, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 15") + ylim(rpe[1], rpe[2])+ theme(legen
  scale_fill_grey(start=.3,end=.7)
p
```




```
p <- ggplot(PE, aes(x=model, y=pe16, fill=factor(model)))
p <- p + geom_boxplot(aes(group = factor(model))) +
  geom_jitter(width = 0.05, height = 0, colour = rgb(0,0,0,.3)) +
  xlab("Approach") + ylab("RMSEp") + labs(title = "Item 16") + ylim(rpe[1], rpe[2])+ theme(legen
  scale_fill_grey(start=.3,end=.7)
p
```


Item 16
1.2
1.0
0.8
0.6
RMSEp
1 2 3 4 5 6 7 8 9 10 11 12 13 14
Approach