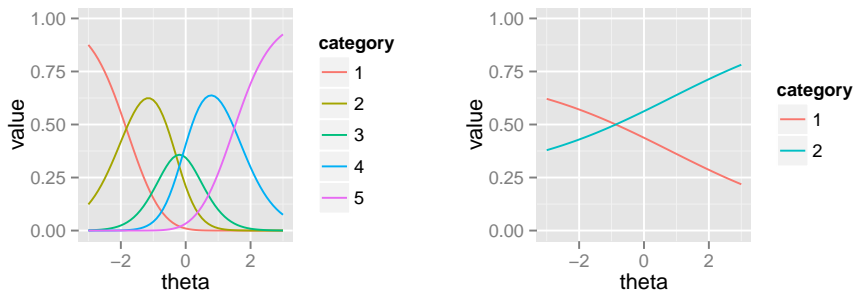


1 How to perform common IRT diagnostics

Here's how to create an item characteristic curve plot for any 1 dimensional item. These are random item parameters so the curves change every time the documentation is recreated.

```
plot.icc <- function(item, param, width = 3) {  
  pm <- rpf.probab(item, param, seq(-width, width, 0.1))  
  icc <- as.data.frame(melt(pm, varnames = c("theta", "category")))  
  icc$theta <- seq(-width, width, 0.1)  
  icc$category <- as.factor(icc$category)  
  ggplot(icc, aes(theta, value)) + geom_line(aes(color = category)) + ylim(0,  
    1) + xlim(-width, width)  
}  
  
i1 <- rpf.gpcm(5)  
i1.p <- rpf.rparam(i1)  
i2 <- rpf.drm()  
i2.p <- rpf.rparam(i2)  
grid.arrange(plot.icc(i1, i1.p), plot.icc(i2, i2.p), ncol = 2)
```



Now let us look at some real data.

```
data(ms.items)  
  
spec <- list()  
spec[1:10] <- rpf.gpcm(5)  
  
plot.info <- function(spec, param, i.name, width = 3) {  
  if (missing(i.name)) {  
    i.name <- paste0("i", 1:length(spec))  
  }  
  grid <- seq(-width, width, 0.1)  
  df <- list(score = grid)  
  total <- numeric(length(grid))  
}
```

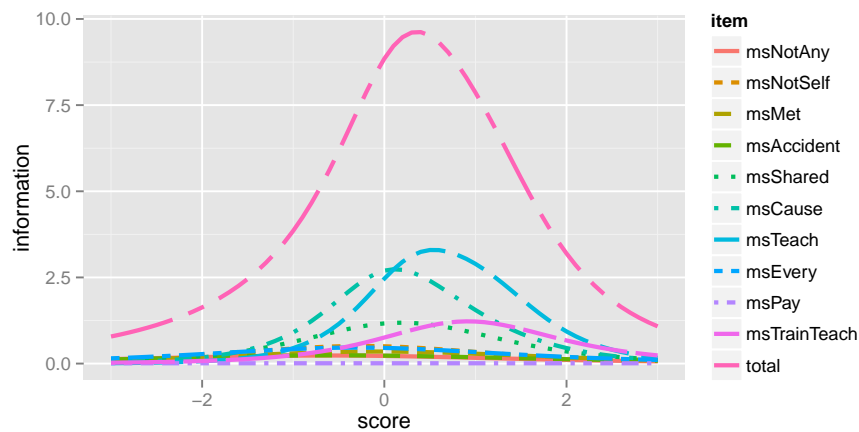
```

for (ix in 1:length(spec)) {
  id <- i.name[ix]
  s <- spec[[ix]]
  df[[id]] <- rpf.info(s, param[ix, 1:s@numParam], grid)
  total <- total + df[[id]]
}
df$total <- total
df <- as.data.frame(df)
long <- melt(df, id.vars = c("score"), variable.name = "item")
long$item <- factor(long$item)
ggplot(long, aes(score, value, group = item)) + geom_line(size = 1.1, aes(linetype = item,
  color = item)) + ylab("information")
}

param <- ms.items[, c("slope", paste0("b", 1:4))]

plot.info(spec, param, ms.items[, "name"])

```



```

data(ms.people)

data.vs.model <- function(spec1, param, espt, item.name, width = 3, data.bins = 10) {
  pm <- rpf.prob(spec1, param[1:spec1@numParam], seq(-width, width, 0.1))
  icc <- as.data.frame(melt(pm, varnames = c("theta", "category")))
  icc$theta <- seq(-width, width, 0.1)
  icc$category <- as.ordered(1 + max(icc$category) - icc$category) #parscale reverses stu
  icc$type <- "model"

  breaks <- seq(min(espt$score, na.rm = TRUE), max(espt$score, na.rm = TRUE),
    length.out = data.bins + 1)
  bin <- unclass(cut(espt$score, breaks, include.lowest = TRUE))
}

```

```

est <- tabulate(bin, length(levels(bin)))
if (any(est < 10)) {
  warning("Some bins have less than 10 samples; try fewer data.bins")
}

eout <- array(dim = c(data.bins, spec1@numOutcomes + 1))
for (px in 1:data.bins) {
  t <- table(espt[[item.name]][bin == px], useNA = "no")
  eout[px, 2:(spec1@numOutcomes + 1)] <- t/sum(t)
}
eout[, 1] <- ((c(breaks, 0) + c(0, breaks))/2)[2:(data.bins + 1)]

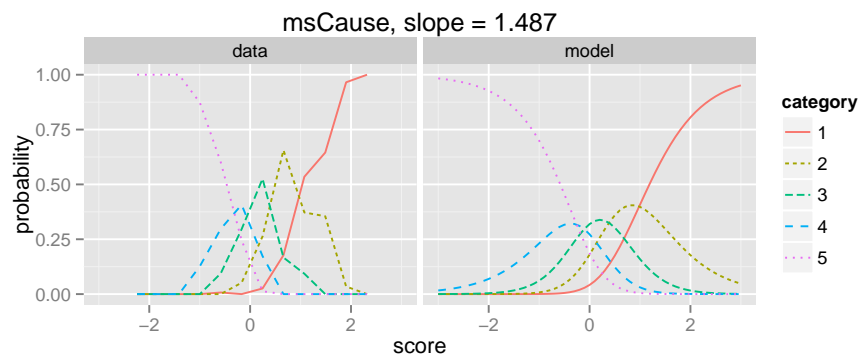
edf <- melt(as.data.frame(eout), id.vars = c("V1"), variable.name = "category")
edf$category <- ordered(unclass(edf$category))
edf$theta <- edf$V1
edf$V1 <- NULL
edf$type <- "data"

both <- rbind(edf, icc)
both$type <- factor(both$type)

ggplot(both, aes(theta, value)) + geom_line(aes(color = category, linetype = category)) +
  facet_wrap(~type) + ylim(0, 1) + xlim(-width, width) + labs(y = "probability",
    x = "score")
}

name <- "msCause"
item.x <- match(name, ms.items$name)
param <- ms.items[item.x, c("slope", paste0("b", 1:4))]
data.vs.model(spec[[item.x]], param, ms.people, name, data.bins = 12) + labs(title = paste0(
  ", slope = ", param[1]))

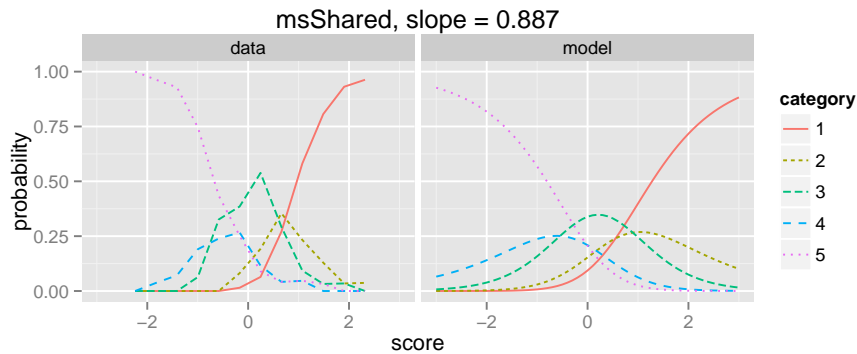
```



Let us plot one more. For msShared, categories 2 and 4 are never the

most likely choice. You can see this visually by confirming that the curves for categories 2 and 4 are already beneath the curves for other categories.

```
name <- "msShared"
item.x <- match(name, ms.items$name)
param <- ms.items[item.x, c("slope", paste0("b", 1:4))]
data.vs.model(spec[[item.x]], param, ms.people, name, data.bins = 12) + labs(title = paste0(
  ", slope = ", param[1]))
```



Item-wise covariance of residuals can be informative.

```
param <- ms.items[, c("slope", paste0("b", 1:4))]
res <- rpf.1dim.residual(spec, param, ms.people[, ms.items$name], ms.people$score)
res <- res[!apply(is.na(res), 1, any), ]
res.cov <- cov(res)
dimnames(res.cov) <- list(ms.items$id, ms.items$id)
round(cov(res.cov), 1)
```

##	NOTA	NOTS	MET	ACCI	SHAR	CAUS	TEAC	EVER	PAY	TT
## NOTA	0.7	0.9	0.6	0.5	0.9	1.0	0.8	0.7	-0.3	0.6
## NOTS	0.9	1.2	0.9	0.8	1.3	1.5	1.3	1.0	-0.4	1.0
## MET	0.6	0.9	1.3	0.8	1.5	1.7	1.5	0.9	-0.3	1.2
## ACCI	0.5	0.8	0.8	0.7	1.0	1.1	0.9	0.7	-0.3	0.7
## SHAR	0.9	1.3	1.5	1.0	2.1	2.3	2.1	1.2	-0.3	1.7
## CAUS	1.0	1.5	1.7	1.1	2.3	2.7	2.5	1.4	-0.3	1.9
## TEAC	0.8	1.3	1.5	0.9	2.1	2.5	2.4	1.3	-0.2	1.9
## EVER	0.7	1.0	0.9	0.7	1.2	1.4	1.3	0.9	-0.3	1.0
## PAY	-0.3	-0.4	-0.3	-0.3	-0.3	-0.3	-0.2	-0.3	0.3	-0.2
## TT	0.6	1.0	1.2	0.7	1.7	1.9	1.9	1.0	-0.2	1.6

Let's take another dataset and look at item fit. This will show the most overfitting and underfitting items.

```

data(science.items)
data(science.people)

scores <- science.people$trait
params <- cbind(1, science.items[, c(paste0("b", 1:2))])
rownames(params) <- as.character(science.items$name)
items <- list()
items[1:25] <- rpf.gpcm(3)
responses <- science.people[, as.character(science.items$name)]
rownames(responses) <- science.people$name
fit <- rpf.1dim.fit(items, params, responses, scores, 2)
head(fit[order(-fit$outfit), ])

##      infit infit.z outfit outfit.z      name
## 23 2.138  5.1802 3.9615  10.4071    WATCH A RAT
##  5 2.056  4.6570 3.3768   8.5301  FIND BOTTLES AND CANS
## 20 1.229  1.4549 1.5981   3.3955    WATCH BUGS
##  8 1.058  0.4359 1.1041   0.7370 LOOK IN SIDEWALK CRACKS
##  9 1.040  0.3358 1.0534   0.4319    LEARN WEED NAMES
## 16 1.008  0.1063 0.9636  -0.2062    MAKE A MAP

tail(fit[order(-fit$outfit), ])

##      infit infit.z outfit outfit.z      name
## 11 0.6973 -2.053 0.5638  -3.182 FIND WHERE ANIMAL LIVES
## 12 0.7915 -1.162 0.5637  -2.776    GO TO MUSEUM
## 17 0.6234 -3.034 0.5603  -3.664  WATCH WHAT ANIMALS EAT
##  2 0.7020 -1.927 0.5235  -3.403  READ BOOKS ON ANIMALS
## 10 0.7154 -1.452 0.5210  -2.745    LISTEN TO BIRD SING
## 21 0.6816 -1.995 0.5112  -3.372  WATCH BIRD MAKE NEST

```

And we can do the same with people.

```

fit <- rpf.1dim.fit(items, params, responses, scores, 1)
head(fit[order(-fit$outfit), ])

##      infit infit.z outfit outfit.z      name
## 72 1.800  2.3481 4.721   7.146 JACKSON, SOLOMON
## 71 2.945  4.8233 4.406   7.072  STOLLER, DAVE
## 73 2.624  4.4155 3.960   6.729  SANDBERG, RYNE
## 29 1.188  0.7115 3.767   5.881  LANDMAN, ALAN
## 12 1.566  1.9160 3.270   5.581 LIEBERMAN, DANIEL
## 49 1.293  1.1557 2.137   3.539  BAUDET, GERARD

tail(fit[order(-fit$outfit), ])

##      infit infit.z outfit outfit.z      name

```

##	16	0.61459	-1.14200	0.34774	-2.3793	BUFF, MARGE BABY
##	34	0.82123	-0.04295	0.33696	-1.0822	PASTER, RUTH
##	41	0.64827	-0.23107	0.28871	-0.9914	FATALE, NATASHA
##	22	0.28735	-3.55599	0.27813	-3.6325	HOGAN, KATHLEEN
##	21	0.25194	-4.11624	0.27123	-3.9369	EISEN, NORM L.
##	2	0.09884	-1.48753	0.05025	-1.7980	ROSSNER, LAWRENCE F.