Measurement checks — HPT (Czech data)

DIF & multi-group CFA by ideological attitude (ideological contamination tests)

HPT and Extremism project

2025-12-12

Contents

1 Purpose & plan

We assess whether *ideological attitude groups* respond differently to the HPT items **even at equal underlying HPT competence**. Concretely:

- Create **low/high ideology** groups from FR-LF-mini and KSA-3 (see codebook).

- Run **item-level DIF** (ordinal logistic) for each HPT item.
- Run **multi-group CFA** (configural → metric → scalar).
- **Interpretation:** If we find pervasive DIF and/or scalar non-invariance, this supports the PCI RR H1 that HPT scores can be ideologically contaminated.

2 Setup

```
options(width = 120)
library(dplyr)
library(tidyr)
library(ggplot2)
library(psych)
library(knitr)
library(stringr)
library(janitor)
library(difR)         # DIF for ordinal items
library(lavaan)       # CFA / invariance
library(semTools)     # helpers
library(car)          # recode
library(mirt)
```

3 Data

```
# Load the dataset created in 00_data-preparation
load("normalised_responses.RData")
stopifnot(exists("normalised_responses"))
dat_raw <- normalised_responses

# Cluster vars
dat_raw <- dat_raw %>%
  mutate(
```

```
    school_id   = as.factor(school_id),
    class_label = as.factor(class_label),
    class_id    = interaction(school_id, class_label, drop = TRUE)
  )

# Coerce HPT items to numeric early (1-4 expected in codebook)
hpt_items <- c(paste0("POP",1:3), paste0("ROA",1:3), paste0("CONT",1:3))
dat_raw <- dat_raw %>% mutate(across(all_of(hpt_items), ~ suppressWarnings(as.numeric(.))))

# Reverse POP item-wise (so higher = more contextualised)
POP_rev_items <- paste0("POP", 1:3)
dat_raw <- dat_raw %>%
  mutate(across(all_of(POP_rev_items), ~ 5 - ., .names = "{.col}_rev")) %>%
  mutate(
    HPT_POP_raw = rowMeans(across(POP1:POP3), na.rm = TRUE),           # presentism, higher = worse
    HPT_POP_rev = rowMeans(across(paste0(POP_rev_items, "_rev")), na.rm = TRUE),  # higher = better
    HPT_CONT    = rowMeans(across(CONT1:CONT3), na.rm = TRUE),
    HPT_ROA     = rowMeans(across(ROA1:ROA3),   na.rm = TRUE),

    # Canonical composites (CTX6 is our stable default)
    HPT_CTX6 = rowMeans(cbind(HPT_POP_rev, HPT_CONT), na.rm = TRUE),
    HPT_TOT9 = rowMeans(cbind(HPT_POP_rev, HPT_CONT, HPT_ROA), na.rm = TRUE)
  )
```

We use variables as defined in the **codebook** (metadata; KN1-KN6; POP1-POP3; ROA1-ROA3; CONT1-CONT3; FR-LF mini RD1-RD3 & NS1-NS3; KSA-3 A1-A3, U1-U3, K1-K3; SDR1-SDR5).

3.1 Scoring & grouping

- **HPT items** are 1-4. We reverse only *presentism* items (`POP1-POP3`: `5 - POP*`) so that a single higher-is-better direction is used for scale construction and MG-CFA. DIF uses original item codings (reversal is not required for DIF detection).
- **Ideology composite**: FR-LF-mini (RD1-3 + NS1-3) and KSA-3 (9 items). We z-score the two scale means and average them → **IDEO_Z**. *Low* = bottom 33%, *High* = top 33% (middle third excluded to sharpen contrasts).
- **Controls**: prior knowledge (sum KN1-KN6), social desirability (SDR1-SDR5; note SDR2–SDR4 are already reversed upstream per codebook).

```
# Select blocks
frlf_items <- c(paste0("RD",1:3), paste0("NS",1:3))
ksa_items  <- c(paste0("A",1:3), paste0("U",1:3), paste0("K",1:3))
kn_items   <- paste0("KN",1:6)
sdr_items  <- paste0("SDR",1:5)
```

```r
# Coerce predictors to numeric
num_blocks <- c(frlf_items, ksa_items, kn_items, sdr_items)

dat <- dat_raw %>%
  mutate(across(all_of(num_blocks), ~ suppressWarnings(as.numeric(.)))) %>%
  # Scale scores
  rowwise() %>%
  mutate(
    HPT_total = mean(c_across(c(paste0("POP",1:3,"_rev"), ROA1:ROA3, CONT1:CONT3)), na.rm = TRUE),
    HPT_CONT  = mean(c_across(CONT1:CONT3), na.rm = TRUE),
    HPT_ROA   = mean(c_across(ROA1:ROA3), na.rm = TRUE),
    HPT_POPR  = mean(c_across(paste0("POP",1:3,"_rev")), na.rm = TRUE),
    FRLF_mean = mean(c_across(all_of(frlf_items)), na.rm = TRUE),
    KSA_mean  = mean(c_across(all_of(ksa_items)),  na.rm = TRUE),
    KN_sum    = sum(c_across(all_of(kn_items)),    na.rm = TRUE),
    SDR_mean  = mean(c_across(all_of(sdr_items)),  na.rm = TRUE)
  ) %>%
  ungroup() %>%
  mutate(
    FRLF_z = as.numeric(scale(FRLF_mean)),
    KSA_z  = as.numeric(scale(KSA_mean)),
    IDEO_Z = (FRLF_z + KSA_z) / 2
  )

# Define tertile groups
qs <- quantile(dat$IDEO_Z, probs = c(.3334, .6666), na.rm = TRUE)
dat <- dat %>%
  mutate(
    ideology_group = case_when(
      IDEO_Z <= qs[1] ~ "Low",
      IDEO_Z >= qs[2] ~ "High",
      TRUE ~ "Mid"
    )
  )

kable(dat %>% count(ideology_group), caption = "Group sizes (Low/High ideology; Mid excluded from group-wise tests)")
```

Table 1: Group sizes (Low/High ideology; Mid excluded from group-wise tests)

ideology_group	n
High	77
Low	77
Mid	80

Note. We focus on *Low* vs *High* to maximise contrast for DIF/MG-CFA. *Mid* is retained for descriptives only.

4 Descriptives (checks)

```
desc_tbl <- dat %>%
  group_by(ideology_group) %>%
  summarise(n = n(),
            HPT_total = mean(HPT_total, na.rm = TRUE),
            KN_sum    = mean(KN_sum,    na.rm = TRUE),
            SDR_mean  = mean(SDR_mean,  na.rm = TRUE)) %>%
  arrange(match(ideology_group, c("Low","Mid","High")))
kable(desc_tbl, digits = 2, caption = "Descriptives by ideology group (means)")
```

Table 2: Descriptives by ideology group (means)

ideology_group	n	HPT_total	KN_sum	SDR_mean
Low	77	2.92	3.47	3.16
Mid	80	2.86	2.98	3.03
High	77	2.75	3.03	2.83

5 DIF analysis (ordinal, item-by-item)

Goal. At *equal HPT ability*, do Low/High ideology groups respond differently to specific items? We match on total HPT (item-rest) and test both uniform and non-uniform DIF per item (= .01, Bonferroni adjusted).

```
## Keep only Low/High groups
anal <- dat[dat$ideology_group %in% c("Low","High"), , drop = FALSE]
```

```r
## HPT item list in original coding
hpt_items <- c("POP1","POP2","POP3","ROA1","ROA2","ROA3","CONT1","CONT2","CONT3")
stopifnot(all(hpt_items %in% names(anal)))

## Build item matrix
hpt_mat <- anal[, hpt_items, drop = FALSE]
for (j in seq_along(hpt_items)) hpt_mat[[j]] <- suppressWarnings(as.numeric(hpt_mat[[j]]))

## Group factor
grp <- factor(anal$ideology_group, levels = c("Low","High"))

## Drop rows with <2 answered items
keep <- rowSums(!is.na(hpt_mat)) >= 2
hpt_mat <- hpt_mat[keep, , drop = FALSE]
grp     <- droplevels(grp[keep])

stopifnot(nrow(hpt_mat) == length(grp), nlevels(grp) == 2)
print(table(grp))
```

```
## grp
##  Low High
##   76   77
```

```r
# Constrained multi-group graded model, then DIF with scheme="drop"
mod_base <- multipleGroup(
  data       = hpt_mat,
  model      = 1,
  group      = grp,
  itemtype   = "graded",
  invariance = c("slopes", "intercepts", "free_means", "free_var")
)
```

```r
params_all <- mirt::mod2values(mod_base)$name
unique_pars <- unique(params_all)
pars_slope <- grep("^a", unique_pars, value = TRUE)
pars_thr   <- grep("^d\\d+$", unique_pars, value = TRUE)
stopifnot(length(pars_slope) > 0, length(pars_thr) > 0)

pars_to_test <- c(pars_slope, pars_thr)

dif_out <- DIF(
```

```r
  mod_base,
  which.par   = pars_to_test,
  scheme      = "drop",
  items2test  = colnames(hpt_mat),
  p.adjust    = "bonferroni",
  verbose     = FALSE
)

res_tbl <- as.data.frame(dif_out)
res_tbl$Item <- rownames(res_tbl)

get_min_p <- function(df, prefix_list) {
  cols <- unlist(lapply(prefix_list, function(p) grep(paste0("^p\\.", p, "$"), names(df), value=TRUE)))
  if (length(cols) == 0) return(rep(NA, nrow(df)))
  apply(df[, cols, drop=FALSE], 1, min, na.rm = TRUE)
}

alpha <- 0.01

res_tbl <- res_tbl %>%
  mutate(
    p_nonuniform = get_min_p(cur_data(), pars_slope),
    p_uniform    = get_min_p(cur_data(), pars_thr),
    Flag_nonuniform = ifelse(!is.na(p_nonuniform) & p_nonuniform < alpha, "YES", "no"),
    Flag_uniform    = ifelse(!is.na(p_uniform)    & p_uniform    < alpha, "YES", "no")
  ) %>%
  select(Item, p_nonuniform, p_uniform, Flag_nonuniform, Flag_uniform)

kable(res_tbl, digits = 4,
      caption = "DIF per item (mirt; graded). Non-uniform = Slope (a1); Uniform = Any Threshold (d1-dK). Bonferroni alpha = 0.01.")
```

Table 3: DIF per item (mirt; graded). Non-uniform = Slope (a1); Uniform = Any Threshold (d1-dK). Bonferroni alpha = 0.01.

	Item	p_nonuniform	p_uniform	Flag_nonuniform	Flag_uniform
POP1	POP1	NA	NA	no	no
POP2	POP2	NA	NA	no	no
POP3	POP3	NA	NA	no	no
ROA1	ROA1	NA	NA	no	no
ROA2	ROA2	NA	NA	no	no
ROA3	ROA3	NA	NA	no	no
CONT1	CONT1	NA	NA	no	no

	Item	p_nonuniform	p_uniform	Flag_nonuniform	Flag_uniform
CONT2	CONT2	NA	NA	no	no
CONT3	CONT3	NA	NA	no	no

```
flagged <- with(res_tbl, Item[Flag_uniform == "YES" | Flag_nonuniform == "YES"])
if (length(flagged) > 0) {
  which_item <- which(colnames(hpt_mat) == flagged[1])
  plot(mod_base, type = "trace", which.items = which_item,
       facet_items = FALSE, groups = levels(grp))
} else {
  plot.new(); text(0.5, 0.5, "No DIF-flagged item at alpha = 0.01.")
}
```

6 Multi-group CFA (configural → metric → scalar)

Model. We specify a **three-factor model** (POP_rev, ROA, CONT) with POP items reversed so that all loadings point to *more HPT-congruent* responses. We then test invariance across Low vs High ideology groups.

```
# Build analysis frame with reversed POP and intact ROA/CONT
cfad <- dat %>%
  filter(ideology_group %in% c("Low","High")) %>%
  transmute(
    ideology_group, class_id,         # keep cluster id for reference (not used by lavaan here)
    POP1 = 5 - POP1,
    POP2 = 5 - POP2,
    POP3 = 5 - POP3,
    ROA1, ROA2, ROA3,
    CONT1, CONT2, CONT3
  )

# Coerce all item columns to numeric and enforce ordinal 1:4 range; replace out-of-range with NA
ord_items <- setdiff(names(cfad), c("ideology_group", "class_id"))
cfad <- cfad %>% mutate(across(all_of(ord_items), ~ suppressWarnings(as.numeric(.))))
cfad <- cfad %>% mutate(across(all_of(ord_items), ~ ifelse(. %in% 1:4, ., NA_real_)))

# If any columns had non 1-4 values, warn instead of stopping
bad_cols <- vapply(cfad[ord_items], function(x) any(!is.na(x) & !(x %in% 1:4)), logical(1))
if (any(bad_cols)) {
  warning(sprintf("Non-1:4 values were set to NA in: %s", paste(names(bad_cols)[bad_cols], collapse=", ")))
}
```

```
model_3f <- '
  POP =~ POP1 + POP2 + POP3
  ROA =~ ROA1 + ROA2 + ROA3
  CONT =~ CONT1 + CONT2 + CONT3
'
```

```
# Run invariance ladder with WLSMV on ordered items; DO NOT pass cluster (not supported with ordered)
fit_conf <- cfa(model_3f, data = cfad, group = "ideology_group",
                ordered = ord_items, estimator = "WLSMV")

fit_metr <- cfa(model_3f, data = cfad, group = "ideology_group",
                ordered = ord_items, estimator = "WLSMV",
                group.equal = c("loadings"))
```

```r
fit_scal <- cfa(model_3f, data = cfad, group = "ideology_group",
                ordered = ord_items, estimator = "WLSMV",
                group.equal = c("loadings", "thresholds"))

get_fit <- function(fit) {
  fitMeasures(fit, c("chisq.scaled", "df.scaled", "pvalue.scaled",
                     "cfi.scaled", "rmsea.scaled", "srmr"))
}

fits <- bind_rows(
  Configural = get_fit(fit_conf),
  Metric     = get_fit(fit_metr),
  Scalar     = get_fit(fit_scal)
) %>%
  mutate(Model = c("Configural", "Metric", "Scalar")) %>%
  select(Model, everything())

fits %>% mutate(across(where(is.numeric), round, 3)) %>%
  kable(caption = "MG-CFA fit indices by invariance level (WLSMV).")
```

Table 4: MG-CFA fit indices by invariance level (WLSMV).

Model	chisq.scaled	df.scaled	pvalue.scaled	cfi.scaled	rmsea.scaled	srmr
Configural	64.608	48	0.055	0.950	0.070	0.089
Metric	79.964	54	0.012	0.921	0.082	0.103
Scalar	78.986	63	0.084	0.952	0.060	0.092

```r
deltas <- tibble(
  step  = c("Configural -> Metric", "Metric -> Scalar"),
  dCFI  = c(fits$cfi.scaled[2] - fits$cfi.scaled[1], fits$cfi.scaled[3] - fits$cfi.scaled[2]),
  dRMSEA= c(fits$rmsea.scaled[2] - fits$rmsea.scaled[1], fits$rmsea.scaled[3] - fits$rmsea.scaled[2])
)

deltas %>% mutate(across(where(is.numeric), round, 3)) %>%
  kable(caption = "Delta fit (CFI, RMSEA) across steps.")
```

Table 5: Delta fit (CFI, RMSEA) across steps.

step	dCFI	dRMSEA
Configural -> Metric	-0.028	0.012
Metric -> Scalar	0.030	-0.022

How to read this. If **metric holds** (small ΔCFI/ΔRMSEA), loadings are equivalent. If **scalar fails**, thresholds differ → **biased group mean comparisons**, supporting H1.

7 (Optional) Two-factor robustness check

```
model_2f <- '
  CTX =~ POP1 + POP2 + POP3 + CONT1 + CONT2 + CONT3
  ROA =~ ROA1 + ROA2 + ROA3
'
measurementInvariance(model_2f, data = cfad, group = "ideology_group",
                      estimator = "WLSMV",
                      ordered = ord_items)
```

8 Interpretation & reporting

8.1 DIF summary

```
kable(res_tbl, caption = "DIF results (for reference in text).")
```

Table 6: DIF results (for reference in text).

	Item	p_nonuniform	p_uniform	Flag_nonuniform	Flag_uniform
POP1	POP1	NA	NA	no	no
POP2	POP2	NA	NA	no	no
POP3	POP3	NA	NA	no	no
ROA1	ROA1	NA	NA	no	no
ROA2	ROA2	NA	NA	no	no
ROA3	ROA3	NA	NA	no	no

	Item	p_nonuniform	p_uniform	Flag_nonuniform	Flag_uniform
CONT1	CONT1	NA	NA	no	no
CONT2	CONT2	NA	NA	no	no
CONT3	CONT3	NA	NA	no	no

8.2 MG-CFA summary

```
kable(fits %>% mutate(across(where(is.numeric), round, 3)),
      caption = "MG-CFA fit to reference in text.")
```

Table 7: MG-CFA fit to reference in text.

Model	chisq.scaled	df.scaled	pvalue.scaled	cfi.scaled	rmsea.scaled	srmr
Configural	64.608	48	0.055	0.950	0.070	0.089
Metric	79.964	54	0.012	0.921	0.082	0.103
Scalar	78.986	63	0.084	0.952	0.060	0.092

```
kable(deltas %>% mutate(across(where(is.numeric), round, 3)),
      caption = "Delta fit (CFI, RMSEA) thresholds.")
```

Table 8: Delta fit (CFI, RMSEA) thresholds.

step	dCFI	dRMSEA
Configural -> Metric	-0.028	0.012
Metric -> Scalar	0.030	-0.022

9 Reproducibility appendix

```
sessionInfo()
```

```
## R version 4.4.2 (2024-10-31)
## Platform: x86_64-pc-linux-gnu
## Running under: Ubuntu 24.04.3 LTS
```

```
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.12.0
## LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.12.0
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C               LC_TIME=cs_CZ.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=cs_CZ.UTF-8    LC_MESSAGES=en_US.UTF-8    LC_PAPER=cs_CZ.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C             LC_MEASUREMENT=cs_CZ.UTF-8 LC_IDENTIFICATION=C
##
## time zone: Europe/Prague
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats4    stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] mirt_1.45.1    lattice_0.22-5 car_3.1-3      carData_3.0-5  semTools_0.5-7 lavaan_0.6-20  difR_6.1.0
##  [8] janitor_2.2.1  stringr_1.5.1  knitr_1.50     psych_2.4.12   ggplot2_4.0.1  tidyr_1.3.1    dplyr_1.1.4
##
## loaded via a namespace (and not attached):
##   [1] RColorBrewer_1.1-3   rstudioapi_0.17.1    audio_0.1-11         shape_1.4.6.1        magrittr_2.0.3
##   [6] TH.data_1.1-4        estimability_1.5.1   farver_2.1.2         nloptr_2.2.1         rmarkdown_2.29
##  [11] fs_1.6.5             vctrs_0.6.5          minqa_1.2.8          htmltools_0.5.8.1    forcats_1.0.0
##  [16] haven_2.5.4          cellranger_1.1.0     Formula_1.2-5        dcurver_0.9.3        parallelly_1.45.1
##  [21] testthat_3.3.1       sandwich_3.1-1       emmeans_1.10.6       rootSolve_1.8.2.4    zoo_1.8-14
##  [26] lubridate_1.9.4      admisc_0.39          lifecycle_1.0.4      iterators_1.0.14     pkgconfig_2.0.3
##  [31] Matrix_1.7-1         R6_2.6.1             fastmap_1.2.0        rbibutils_2.3        future_1.68.0
##  [36] snakecase_0.11.1     digest_0.6.37        Exact_3.3            vegan_2.7-2          progressr_0.18.0
##  [41] timechange_0.3.0     abind_1.4-8          httr_1.4.7           mgcv_1.9-1           compiler_4.4.2
##  [46] proxy_0.4-27         withr_3.0.2          S7_0.2.1             R.utils_2.13.0       MASS_7.3-61
##  [51] sessioninfo_1.2.3    GPArotation_2024.3-1 permute_0.9-8        gld_2.6.7            tools_4.4.2
##  [56] pbivnorm_0.6.0       future.apply_1.20.0  clipr_0.8.0          R.oo_1.27.1          glue_1.8.0
##  [61] quadprog_1.5-8       nlme_3.1-166         grid_4.4.2           cluster_2.1.8        generics_0.1.3
##  [66] gtable_0.3.6         tzdb_0.5.0           R.methodsS3_1.8.2    class_7.3-22         data.table_1.17.8
##  [71] lmom_3.2             hms_1.1.3            Deriv_4.1.6          foreach_1.5.2        pillar_1.10.0
##  [76] splines_4.4.2        survival_3.7-0       tidyselect_1.2.1     pbapply_1.7-4        reformulas_0.4.1
##  [81] gridExtra_2.3        deltaPlotR_1.6       xfun_0.54            expm_1.0-0           brio_1.1.5
##  [86] stringi_1.8.4        VGAM_1.1-14          yaml_2.3.10          boot_1.3-31          evaluate_1.0.5
##  [91] codetools_0.2-20     beepr_2.0            msm_1.8.2            tibble_3.2.1         cli_3.6.5
##  [96] xtable_1.8-4         DescTools_0.99.60    Rdpack_2.6.4         Rcpp_1.0.13-1        readxl_1.4.3
## [101] globals_0.18.0       polycor_0.8-1        coda_0.19-4.1        parallel_4.4.2       readr_2.1.5
```

```
## [106] lme4_1.1-38          listenv_0.10.0       glmnet_4.1-10        mvtnorm_1.3-2        SimDesign_2.21
## [111] scales_1.4.0         e1071_1.7-16         purrr_1.1.0          rlang_1.1.6          multcomp_1.4-28
## [116] mnormt_2.1.1         ltm_1.2-0
```