mirtverse ile madde tepki kuramı :: ÖZET KILAVUZ

mirtverse, **mirt** paketinin üzerine kuruludur. Modelleri mirt kestirir; mirtverse ise model seçmenize, sonuçları okumanıza, grafik üretmenize ve sonuçları paylaşmanıza yardım eder. Her fonksiyon sıradan bir mirt modeli kabul eder; iki paketi birlikte kullanabilirsiniz.

Kurulum

```
install.packages("mirtverse", repos = c(
  "https://okanbulut.r-universe.dev",
  "https://cloud.r-project.org"))
library(mirtverse)  mirt de yüklenir
```

Model seçin — buradan başlayın

```
data <- expand.table(LSAT7)
sel  <- select_model(data)
mod  <- sel$best    # sıradan bir mirt modeli
```

`select_model(data, candidates, dims = 1, criterion = "BIC")`
Modelleri kestirir ve birini önerir: 1-0 maddeler için Rasch, 2PL veya 3PL; diğerleri için PCM, GPCM veya aşamalı tepki.

`autoplot(sel)` · `interpret(sel)` · `tidy(sel)`
Adayları grafikle, sözle veya tabloyla karşılaştırın.

`mirt(data, 1, itemtype = "2PL")`
Ya da modeli doğrudan mirt ile kestirin.

Sonuçları sade bir dille okuyun

`interpret(mod, alpha = 0.01)`
Uyum, güvenirlik, zayıf ya da uyumsuz maddeler ve yerel bağımlılık, cümlelerle anlatılır. `select_model()`, `dif()` ve `fit_rt()` sonuçlarında da çalışır.

`interpret_prompt(mod, audience)`
Sonuçları bir yapay zekâ asistanına yapıştırılacak bir isteme dönüştürür. Hiçbir veri gönderilmez.

Düzenli (tidy) çıktılar — dplyr, gt ... için veri çerçeveleri

`tidy(mod, IRTpars = TRUE, conf.int = FALSE)`
Her madde parametresi için bir satır, MTK (a, b, c) metriğinde.

`glance(mod)`
Tek satırda model uyumu: logLik, AIC, BIC, M2, RMSEA, CFI.

`augment(mod, method = "EAP", person_fit = FALSE)`
Her bireyin puanı ve standart hatası eklenmiş veri.

`item_table(mod, fit_stats = "S_X2")`
Madde başına bir satır: Parametreler, uyum, durum uyarısı.

tidy(); DMF, tepki süresi, banka ve süreç nesnelerinde de çalışır.

Değişen madde fonksiyonu (DMF)

```
d <- dif(data, group = gender, model = "2PL")
tidy(d); interpret(d); gg_dif(d)
```

`dif(x, group, model, items, which.par, p_adjust = "BH")`
Olabilirlik oranı testleri, etki büyüklükleri, düzeltilmiş p değerleri.

Grafikler — her grafik bir ggplot


`gg_trace(mod)` madde eğrileri; `type = "expected"`


`gg_info(mod)` `type = "test"`, `"items"`, `"se"`


`gg_wright(mod)` bireyler ve maddeler aynı ölçekte


`gg_itemfit(mod)` kesme değeriyle madde uyumu


`gg_q3(mod, cutoff = 0.2)` yerel bağımlılık


`gg_scores(mod)` birey puanlarının dağılımı


`gg_empirical(mod, item)` model eğrisi ve gözlenen


`gg_dif(d)` `type = "effect"` veya `"curves"`

`autoplot(mod, type)`
Tüm grafikler tek fonksiyondan: `"trace"`, `"expected"`, `"info"`, `"iteminfo"`, `"se"`, `"wright"`, `"itemfit"`, `"q3"`, `"scores"`.

`gg_trace(mod) + theme_mirtverse(base_size = 11)`
Etiketleri, renkleri ve temaları her ggplot'ta olduğu gibi değiştirin.

Ortak argümanlar: `which.items` maddeleri seçer, `theta_lim = c(-4, 4)` yetenek aralığını belirler.

Tepki süreleri

```
sim <- sim_rt(N = 1500, J = 20, rapid = 0.08)
mod <- mirt(sim$responses, 1, "2PL")
fit <- fit_rt(mod, sim$rt)
```

`fit_rt(x, rt, flag_method = "residual")`
Tepkiler ve tepki süreleri için ortak model (van der Linden, 2007); birey hızı ve hızlı tahmin uyarılarıyla. Ayrıca `"nt10"`, `"nt15"`, `"nt20"`.

`tidy(fit)` · `glance(fit)` · `augment(fit)` · `autoplot(fit)` · `interpret(fit)`
Madde süre parametreleri, uyum, birey hızı ve uyarılar, grafikler ve sade dille bulgular.

`sim_rt(N, J, rho = -0.3, sd_speed, rapid, seed)`
Bir kısmı hızlı tahminde bulunan bireylerle tepki ve süre verisi üretir.

Başka bir MTK paketinden mi geliyorsunuz?

`mirtify(code, run = FALSE, notes = TRUE)`
ltm, **eRm**, **TAM** veya **irtQ** kodunu, farkları açıklayan notlarla mirt koduna çevirir. Bir metin ya da .R dosyası verin.

`mirtify_crosswalk(package)`
Karşılık gelen fonksiyonlar tablosu.

Sonuçları paylaşın

`item_card(mod, item)` · `item_cards(mod, file)`
İnceleme toplantıları için her maddeye trafik ışığı durumlu tek sayfalık bir madde durum kartı.

`mv_report(sel, file = "report.html", dif, cards, code)`
Tek bir HTML rapor: Model karşılaştırması, yorum, tablolar, grafikler, toplam puan tablosu, madde kartları ve raporu yeniden üretecek R kodu.

`mv_dashboard(mod, title, panels, dif, rt)`
R kullanmayanlar için salt okunur bir Shiny gösterge paneli.

`export_dashboard(mod, "results.html", format = "html")`
E-postayla gönderilebilen çevrim dışı tek bir HTML dosyası; shinyapps.io veya Posit Connect için `format = "shiny"`.

Tıklayın ya da sözle anlatın

`launch_workbench(data)`
Her adımın R kodunu sizin için yazan, tıkla-çalıştır bir uygulama.

```
mv_ask_setup("claude")        # bir kez
scr <- mv_ask("Fit a 2PL, check DIF by gender
  and make a report", data = mydata)
run_script(scr)
save_script(scr, "analysis.R")
```

`mv_ask(request, data, anonymize = FALSE, dry_run = FALSE)`
Bir isteği test edilmiş bir betiğe dönüştürür. `dry_run = TRUE` neyin gönderileceğini gösterir; `anonymize = TRUE` sütun adlarını gizler. Anahtar sözcük ayrıştırıcısıyla çevrim dışı da çalışır.

`profile_data(data)` · `mv_plan()` · `update_plan(plan, ...)`
Veriyi inceleyin ya da analiz planını kendiniz yazıp düzenleyin.

Öğretim laboratuvarları

`run_lab("model-choice")` · `run_lab("decisions")`
Etkileşimli dersler: Öğrenciler canlı modeller kurar ve geri bildirimle ölçme kararları verir.

`list_labs()` · `lab_truth()`
Mevcut laboratuvarlar; öğretim elemanları için gizli cevaplar.

`certexam` · `certexam_key`
İngilizce ve Fransızca, içine sorunlar yerleştirilmiş 32 maddelik benzetim bir sertifika sınavı.

Ölçme süreci (pipeline) :: ÖZET KILAVUZ

Tek bir nesne, bir testi belirtke tablosundan madde bankası izlemeye kadar taşır. Her aşama işini yapar, ardından bir **kontrol noktası** çalıştırır: Sonraki aşamadan önce değerlendirmeniz için **uyarılar** kaydeden kontroller. Her karar kayda geçer ve `p$code` tüm sürecin R kodunu saklar.

```
p <- mv_pipeline("Certification exam")
p <- add_responses(p, certexam, items = 3:34,
       key = certexam_key, group = "language")
p <- calibrate(p, model = "2PL")
flags(p)
pipeline_report(p, "report.html")
```

0	1	2	3	4	5	6	7	8	9	10	11	12
Belirtke tablosu	Madde üretimi	Madde inceleme	Deneme uygulaması	Veri girişi	Kalibrasyon	Bağlama ve banka	Formlar	Kesme puanları	Puanlama	Test güvenliği	Raporlama	Banka izleme

Kontrol noktaları ve uyarılar

karar verin, sonra ilerleyin

- **info** bilgi için
- **review** devam etmeden önce inceleyin
- **block** düzeltin, çıkarın ya da gerekçeyle geçin

`flags(p, status = "open", level, stage)`
Uyarıları listeler; `status = "all"` karara bağlananları da içerir.

`acknowledge(p, check, target, note)`
Bir uyarıyı, isteğe bağlı bir notla kabul eder.

`override(p, check, target, note)`
Bir uyarıyı geçersiz kılar; yazılı bir gerekçe gerekir.

`drop_items(p, items, note)`
Maddeleri çıkarır ve uyarılarını düzeltildi olarak kaydeder.

Kurallar ve katı mod

`mv_pipeline(title, rules = mv_rules(), strict = FALSE)`
Kontrol noktaları varsayılan olarak yalnızca uyarır. `strict = TRUE` ile açık review ve block uyarıları, bir karar kaydedilene kadar sonraki aşamayı durdurur.

`mv_rules(..., preset = "standards")`
Her eşik adıyla değiştirilebilir: `mv_rules(q3 = 0.25, reliability = 0.85)`. `preset = "lenient"` pilot çalışmalar için eşikleri gevşetir.

print(p) aşamaların durumunu, tidy(p) tüm uyarıları gösterir.

Testi oluşturun

aşama 0–3

0 set_blueprint(p, cells, construct, use, reliability**)**
İçerik hücreleri ve madde sayıları, örneğin `c(Etik = 4, Hukuk = 3)`.

1 generate_items(p, n, type, provider**)** LLM
Çoktan seçmeli veya açık uçlu taslaklar; `verify_items()` anahtarları kontrol eder; `add_items(p, items)` kendi maddelerinizi ekler.

2 review_packet(p, file, reviewers**)**
İnceleme formu yazar; `read_reviews(p, reviews)` onu geri okur.

3 design_field_test(p, n_forms, n_per_form)
Formları ve ortak maddeleri planlar, kestirimi benzetimle sınar.

Kalibre edin ve form oluşturun

aşama 4–7

4 add_responses(p, data, items, key, group, id, rt**)**
Ham cevapları anahtarla puanlar. Kayıp veri, hatalı anahtar, hep aynı seçeneği işaretleme, hızlı tahmin ve seyrek kategorileri kontrol eder.

5 calibrate(p, model = "auto", dif, reference**)**
Modeli kestirir. Yakınsama, model ve madde uyumu, yerel bağımlılık, parametre aralıkları, DMF, güvenirlik ve birey uyumunu kontrol eder.

6 link_to_bank(p, bank, method**)**
Yeni maddeleri bankanın ölçeğine taşır: `"stocking-lord"`, `"haebara"`, `"mean-sigma"`, `"mean-mean"` veya `"fixed"`. `bank_items(p, name)` onları kaydeder; `mv_bank(mod)` bir modelden banka başlatır.

7 assemble_forms(p, length, n_forms, theta, content, blueprint, enemies**)**
Belirtke tablosuna uyan paralel formlar kurar ve birbirini ele veren maddeleri ayrı tutar.

`simulate_cat(p, n, min_sem, max_items, criteria = "MI")`
Bankadan bilgisayarlı bireyselleştirilmiş test (BBT) benzetimi yapar.

`assemble_mst(p, router_length, module_length, n_modules)`
Çok aşamalı bir test kurar ve benzetimini yapar.

certexam örneği

```
p <- drop_items(p, "Q14", note = "Miskey")
p <- calibrate(p, model = "2PL")
p <- override(p, check = "dif_large",
       note = "French version revised")
p <- bank_items(p, name = "Certification bank")
p <- set_cuts(p, cuts = 18, scale = "raw",
       labels = c("Fail", "Pass"))
p <- score_persons(p)
```

Adım adım anlatımın tamamı web sitesindeki *Assessment pipeline* makalesindedir (İngilizce).

Karar verin, puanlayın ve raporlayın

aşama 8–12

8 set_cuts(p, cuts, scale = "theta", labels, panel**)**
Theta ya da ham puan ölçeğinde kesme puanları belirler; sınıflama doğruluğu ve tutarlılığıyla birlikte. `item_map(p, rp = 0.67)` standart belirleme için maddeleri ölçeğe yerleştirir.

9 score_persons(p, method = "EAP"**)**
Standart hatalarıyla birey puanları; güvenirlik ve uç puanlar açısından kontrol edilir. `score_open(p, responses)` açık uçlu cevapları bir rubrik ve bir büyük dil modeli (LLM) ya da kendi eğittiğiniz modelle puanlar. LLM

10 check_security(p, leaked, window, site**)**
Maddeleri önceden görme, sızdırılmış maddeler, parametre kayması, kopya ve olağan dışı süreler.

11 pipeline_report(p, file**)**
Her aşama, uyarı ve kararı içeren HTML teknik rapor ve R kodu.

12 monitor_bank(p, window, date, content**)**
Her uygulamadan sonra: Madde kullanım oranı, içerik hücresi başına banka derinliği, güvenirlik eğilimi ve kullanılmayan maddeler. `retire_items(p, items, status, note)` maddeleri kullanımdan kaldırır.

Her aşamanın gereksinimleri

`ellmer` LLM
Madde üretimi, yapay zekâ ile inceleme taraması ve açık uçlu puanlama; kendi API anahtarınızla (Claude, OpenAI veya yerel bir Ollama modeli).

`eatATA`
Form oluşturma.

`mirtCAT`
BBT ve çok aşamalı test benzetimi.

Bunlar önerilen paketlerdir: Yalnızca kullandığınız aşamalar için kurun. Bağlama ve kesme puanı yöntemleri pakette yerleşiktir.

Tıklamayı mı tercih edersiniz?

Workbench uygulamasındaki **Pipeline** sekmesi aynı aşamaları çalıştırır, uyarıları gösterir ve kodu yazar: `launch_workbench()`.