Title

A Stan tutorial on Bayesian IRTree models: Conventional models and explanatory extension

Authors

Xue, Mingfeng
Chen, Yi

Publication Date

2023-04-24

DOI

10.3758/s13428-023-02121-5

Behavior Research Methods
https://doi.org/10.3758/s13428-023-02121-5

A Stan tutorial on Bayesian IRTree models: Conventional models and explanatory extension

Mingfeng Xue[1] · **Yi Chen**[2]

Accepted: 30 March 2023

Abstract

IRTree models have been receiving increasing attention. However, to date, there are limited sources that provide a systematic introduction to Bayesian modeling techniques using modern probabilistic programming frameworks for the implementation of IRTree models. To facilitate the research and application of IRTree models, this paper introduces how to perform two families of Bayesian IRTree models (i.e., response tree models and latent tree models) in Stan and how to extend them in an explanatory way. Some suggestions on executing Stan codes and checking convergence are also provided. An empirical study based on the Oxford Achieving Resilience during COVID-19 data was conducted as an example to further illustrate how to apply Bayesian IRTree models to address research questions. Finally, strengths and future directions are discussed.

Keywords Bayesian estimation · IRTree models · Response styles · Explanatory IRT · Stan

Item response tree (IRTree) models have been emerging as a promising tool for addressing responses consisting of multiple processes due to their easy implementation, high interpretability, and flexibility (Ames & Leventhal, 2023; Debeer et al., 2014; Plieninger, 2021). As their names suggest, they take the form of a tree structure with each node representing a different process, and typically assume unidimensionality within each process. In the field of psychological assessments, IRTree models are typically applied in detecting and controlling for response styles – a kind of tendency to choose specific options regardless of the content of the item, such as extreme responses and middle responses (Ames & Myers, 2021; Jeon & De Boeck, 2019; LaHuis et al., 2019; Park, 2021). In the field of educational assessment, researchers have employed IRTree models to investigate test-taking behaviors, like skipping, not reaching (Debeer et al., 2017), and missing (Debeer et al., 2014). Extensions of IRTree models have been proposed as well. Khorramdel et al. (2019), Huang (2020), and Kim and Bolt (2021) incorporate mixture distribution into IRTree models. Ames and Leventhal (2021) extend IRTree models in a longitudinal manner. Jeon and De Boeck (2016) proposed a general IRTree model, which can accommodate multidimensionality and explanatory covariates, etc. Similarly, person- and item-side covariates have also been considered in the new IRTree models developed by Ames and Myers (2021). Moreover, the binary IRT function within some nodes can be replaced by a polytomous and multidimensional IRT function (Meiser et al., 2019; Park & Wu, 2019).

Despite the proliferation of the development and application of IRTree models, tutorials on how to perform IRTree models remain limited. As of today, the most cited and relevant tutorial paper is contributed by De Boeck and Partchev (2012), which is under the framework of generalized linear mixed models (GLMMs) and employs the *lme4* R package. However, *lme4* cannot handle generalized nonlinear models, which makes it impossible to incorporate two parameters IRT models into IRTree models. Moreover, *lme4* only adopts frequentist mechanisms in model design and parameter estimation.

On the other hand, Bayesian statistics provide a conceptually and practically coherent tool for addressing these broader and increasingly sophisticated problems because of its advantages in straightforward and easy implementation, support in quantifying uncertainty in statistical inferences with a full probability model, availability of various prior distributions, and flexibility of extending traditional psychometric models (Fox & Glas, 2001; Levy & Mislevy, 2017). Therefore, Bayesian IRTree models can be more appealing than

✉ Mingfeng Xue
mingfengxue@berkeley.edu

[1] Berkeley School of Education, University of California Berkeley, Berkeley, CA, USA

[2] Teachers College, Columbia University, New York, NY, USA

Published online: 24 April 2023

frequentist-based IRTree models in many real applications. With the popularity of probabilistic programming languages, it is easy to implement the Markov chain Monte Carlo (MCMC) technique for any customized Bayesian model design. Thus, an introduction to performing Bayesian IRTree models on MCMC software would be meaningful and practical.

There are several well-known and widely used MCMC software programs that free researchers from directly writing the sampling algorithms and enable them to focus more on the models. Popular MCMC software includes BUGS (Lunn et al., 2000), JAGS (Plummer, 2003), and Stan (Carpenter et al., 2017). Furthermore, a myriad of tutorials on Bayesian IRT provided by researchers have also made it flourish. Examples can be found in a series of Bayesian IRT publications on Stan (Luo & Jiao, 2018), JAGS, BUGS (Curtis, 2010), and SAS (Ames & Samonte, 2015), and this paper shares the same ambition.

This paper aims to facilitate the application and development of IRTree models and contribute to the field by introducing how to perform Bayesian IRTree models in Stan, where the two-parameter logistic model (2PL) and the graded response model (GRM) are used in response tree models and latent tree models, respectively. Thus, this paper goes beyond the Rasch family model used in previous tutorials on IRTree, where discrimination parameters will be obtained, and items have different weights in estimating latent traits, which can result in better model and item fits. The second contribution of this paper is to extend IRTree models in an explanatory way on both person and item sides in Stan, so more explanatory information can be extracted from IRTree analyses. The article is organized as follows: first, two families of IRTree models and Bayesian estimation are introduced; then, a general description of Stan is presented as the basis of the Stan codes for Bayesian IRTree models; third, a detailed introduction of how to perform Bayesian IRTree models and how to extend them to explanatory IRTree models is given; fourth, one empirical study using the Bayesian IRTree models are presented; and finally, we will discuss the strengths and future research direction of Bayesian IRTree models.

IRTree models

The rationale of IRTree models lies in the assumption that multiple processes exist in a single response, which can be represented as a tree structure. Roughly speaking, there are two families of IRTree models (De Boeck & Partchev, 2012), and the classification rests on where the multiple processes occur – either in responses or in latent traits, and they hereafter are referred to as the response tree model and the latent tree model, respectively.

Response tree models

Response tree models decompose observed ordered responses (or responses with missingness) into various subresponses (aka pseudo-item responses) through prespecified nodes that represent different traits of interest.

Figure 1 demonstrates an example of a response tree model of a four-point Likert scale item with options from 'strongly disagree' to 'strongly 'agree'. The observed response Y_{pi} of respondent p to item i is decomposed into three pseudo-item responses through three nodes. The top node is related to the targeted trait, say personality (LaHuis et al., 2019), while the bottom two nodes concern the tendency to extreme responses. Specifically, the bottom left node represents the respondents' extreme response style conditional on disagreement on the

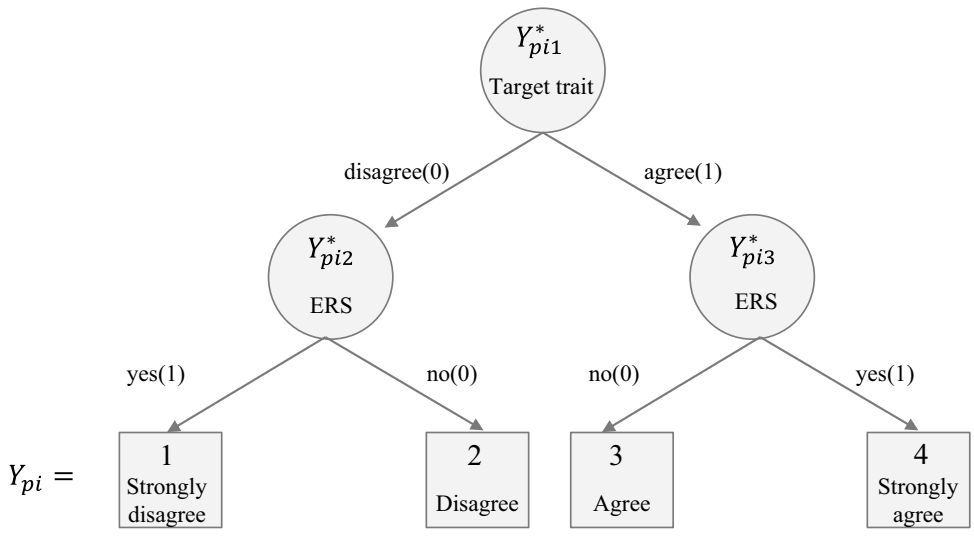

Fig. 1 An example of response tree models. *Note.* ERS= extreme response style; the number within parentheses indicates the pseudo-item response

targeted trait, while the bottom right node is respondents' extreme response style conditional on agreement on the targeted trait. Consequently, there are two processes underlying one observed response – that is, determining whether he/she has a high level of the targeted trait and choosing between extreme and non-extreme responses. Take one point as an instance – it is a composition of disagreement of the targeted trait and preference for extreme responses. Table 1 displays the mapping schemes corresponding to Fig. 1, which bridges theoretical expectation and mathematical expression. In Table 1, NA stands for not applicable where the observed response does not go through that node. For example, the observed score 1 doesn't go through node 3 and should be recoded into a pseudo-item response vector (0, 1, NA). Generally speaking, the scheme of mapping pseudo-items responses reflects researchers' assumptions about the processes underlying the observed responses.

After obtaining pseudo-item responses, a unidimensional IRT model for binary response is used within each node to formulate the probability of $Y^*_{pik} = 1$. As of today, most research relies on the one-parameter logistic model within the framework of GLMMs (e.g., Ames & Leventhal, 2021; Debeer et al., 2014; Plieninger, 2021). Alternatively, more general IRT designs, such as a two-parameter logistic model (e.g., Jeon & De Boeck, 2016; Kim & Bolt, 2021), could also be used, which follows the equation below:

$$P\left(Y^*_{pik}=1\right)=\frac{\exp\left(\alpha_{ik}\theta_{pk}+\beta_{ik}\right)}{1+\exp\left(\alpha_{ik}\theta_{pk}+\beta_{ik}\right)} \tag{1}$$

where α, β, and θ denote discrimination parameters, difficulty parameters, and latent traits, respectively, while p, i, and k index person, item, and node, respectively. Because each observed response is specified as a combination of pseudo-item response vector from different processes, multiplication is used to calculate its joint probability. Thus, the likelihood of response IRTree model takes the form:

$$P\left(Y_{pi}=m\right)=\prod_{k=1}^{K}\left[\frac{\exp\left(\alpha_{ik}\theta_{pk}+\beta_{ik}\right)^{c_{mk}}}{1+\exp\left(\alpha_{ik}\theta_{pk}+\beta_{ik}\right)}\right]^{d_{mk}} \tag{2}$$

where c_{mr} and d_{mr} are indicators for endorsement and applicableness. When a pseudo-item involving score m is NA, the probability is multiplied by 1, which means this pseudo-item has no influence on the probability of getting m. In Stan, we don't have to directly define Eq. (2), rather than we can use the conditional statement and Eq. (1) to achieve the same function because for a given observed score, the branch and the pseudo-item responses in the IRTree model are known (e.g., see Table 1), and mapping observed responses to pseudo-item responses within Stan makes IRTree analysis more fluid and future researchers using our codes can avoid transforming data outside Stan. The details of the conditional statement are presented in the codes section. Note that we use the 2PL form of $\alpha_{ik}\theta_{pk}+\beta_{ik}$ here, instead of $\alpha_{ik}(\theta_{pk}+\beta_{ik})$ in that it has been used in some of the previous studies on IRTree models and explanatory IRT (e.g., Ames & Myers, 2021; Jeon & De Boeck, 2016) and make the interpretation of explanatory IRT models on item sides more straightforward as interpreting the effect of predictors on response probability does not involve α_{ik}.

Table 1 A mapping scheme for a response tree model

Y_{pi}	$Y^*_{pi}1$	$Y^*_{pi}2$	$Y^*_{pi}3$
1	0	1	NA
2	0	0	NA
3	1	NA	0
4	1	NA	1

NA represents not applicable

Latent tree models

On the other hand, latent tree models focus on describing the complex cognitive structure of latent traits, which can also be viewed as a special kind of multidimensional IRT model.

Figure 2 displays a latent tree model, in which circle nodes represent latent traits of interest, diamonds represent the summation of all preceding traits leading to this node (e.g., $\theta_1 = \theta'_1$, $\theta_2 = \theta'_1 + \theta'_2$, and $\theta_3 = \theta'_1 + \theta'_2 + \theta'_3$ for unweighted summation), and rectangles represent observed responses. In De Boeck

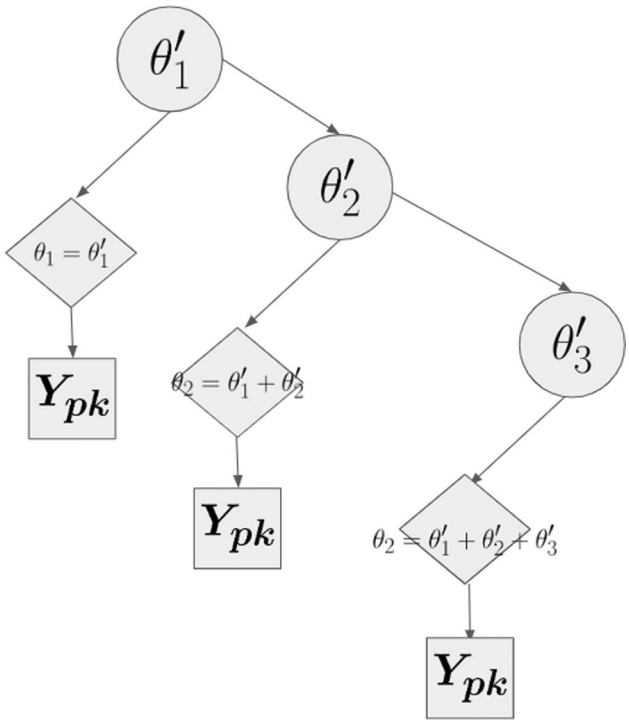

Fig. 2 An example of latent tree models

Table 2 A mapping scheme for a latent tree model

	θ'_1	θ'_2	θ'_3
θ_1	1	0	0
θ_2	1	1	0
θ_3	1	1	1

and Partchev's example (2012), the weighted summation is dropped due to the limitation of GLMM. A mapping scheme is also needed for the summation (see Table 2). For weighted summation, a weight vector is also necessary for each θ'. The latent IRTree model in the framework of Fig. 2 might be useful for modeling the change in latent trait, where θ'_2 and θ'_3 are the provisional growth at corresponding time points. In addition, latent tree models can formulate a wide range of multidimensional IRT models, such as bifactor models and compensatory multidimensional IRT models (De Boeck & Partchev, 2012).

Taking a latent tree model with dichotomous responses using 2PL as an example:

$$P\left(Y_{pik}=1\right)=\frac{\exp\left[\sum_{k=1}^{K}\left(q_{ik}\alpha_{ik}\theta'_{pk}-\beta_i\right)\right]}{1+\exp\left[\sum_{k=1}^{K}\left(q_{ik}\alpha_{ik}\theta'_{pk}-\beta_i\right)\right]} \tag{3}$$

Here p, k, and i denote index of respondents, nodes, and items, respectively; q_{ik} comes from the mapping matrix and α_{ik} stands for corresponding weights; θ'_{pk} denotes respondent p's latent trait at node k; and β_i is the difficulty parameters of item i. For polytomous responses, we can exploit partial credit models, graded response models, and so on.

Explanatory IRTree models

The flexibility of Bayesian IRT enables numerous extensions. This section introduces how to incorporate explanatory variables into IRTree models in the framework proposed by Wilson and De Boeck (2004). As illustrated in Table 3, explanatory variables can be included on the person and item sides. For example, if researchers are interested in how negatively worded items differ from positively worded items in difficulty parameters, a binary covariate can be included as item covariate; if group differences in latent traits are of main importance, say gender, a covariate about gender can be incorporated as person covariate.

Moreover, in doubly explanatory response tree models, we need to take into account that the effects of the same explanatory variable might vary across the pseudo-items, that is, the coefficients for the same explanatory variable might not be the same when it comes to different pseudo-items. Therefore, the doubly explanatory variables are incorporated to both sides, latent traits and difficulty parameters of Eq. (1) are formulated as:

$$\theta_{pk}=\Sigma_j\lambda_{jk}Z_{pj}+\eta_{pk} \tag{4}$$

$$\beta_{ik}=\Sigma_v\gamma_{vk}X_{iv}+\delta_{ik} \tag{5}$$

where k still denotes the kth pseudo-item, and Z and X are covariates for person and items, respectively, and their corresponding coefficients are λ and γ. η_{pk} and δ_{ik} are normally distributed random effects. The same formulation also holds true for explanatory latent tree models except there are not pseudo-items (see Stan codes in appendices).

Bayesian estimation

The key to Bayesian estimation is Bayes' theorem:

$$P(a|D)=\frac{P(D|a)\bullet P(a)}{P(D)} \tag{6}$$

where on the left side, $p(a|D)$ is the posterior distribution of the parameters of interest (denoted as a) given data observed (denoted as D); on the right side, $p(a|D)$ is the likelihood of obtaining data D given parameter a, $p(a)$ is the prior distribution for parameter a, representing our belief of a, and $p(D)$ is the marginal distribution, which can be considered as the normalizing constant of the numerator and is of least importance. Therefore, in Eq. (4), we only set the likelihood function and prior distribution.

In the case of Bayesian IRTree models, the likelihood function can be obtained from Eqs. (2) and (3), and prior distributions can be set according to the prior information and belief about the parameters to be estimated. For example, they can be set as below:

Table 3 Explanatory IRT models adapted from Wilson and De Boeck (2004)

		Person side	
		Absence of properties (person indicators)	Inclusion of properties (person properties)
Item side	Absence of properties (item indicators)	Doubly descriptive	Person explanatory
	Inclusion of properties (item properties)	Item explanatory	Doubly explanatory

For person trait vector $\boldsymbol{\theta}$:

$$\boldsymbol{\theta} \sim N(\boldsymbol{\mu}, \Sigma_{\theta}),$$

where the mean vector $\boldsymbol{\mu}$ is set as 0s and the covariance matrix as an identity matrix, a common way to determine the identification of scaling in IRT.

For pseudo-item discrimination parameters α, we can use lognormal prior distribution because log multivariate normal distribution is not available in Stan.:

$$a \sim lognormal(\mu_a, \sigma_a)$$

with hyper-prior distributions:

$$\begin{aligned} \mu_a &\sim N(0, 1) \\ \sigma_a &\sim Cauchy(0, 5) \end{aligned}$$

For pseudo-item difficulty parameters β, following the same vein of $\boldsymbol{\theta}$, we can use multivariate normal distribution:

$$\beta \sim N(\boldsymbol{\mu_\beta}, \boldsymbol{\Sigma}_\beta)$$

with each element of $\boldsymbol{\mu_\beta}$ sampling from a normal distribution:

$$\mu_\beta \sim N(0, 1)$$

and the diagonal elements of $\boldsymbol{\Sigma}_\beta$ sampling from

$$\sigma_\beta \sim Cauchy(0, 5),$$

and the elements are set to 0.

The trickiest step in Bayesian estimation lies in calculating the evidence in posterior distributions. To solve this problem, MCMC methods are generally applied to calculating numer ical approximations of multi-dimensional integrals. In running MCMC estimation, we need to determine the number of chains because poor starting values might cause slow convergence, and one chain limits our possibility of diagnosing the convergence, e.g., Rhats compare the between- and within-chain variance of parameters estimates (Vehtari et al., 2021). Then, we need to decide the number of iterations of each MCMC chain, and how many iterations to throw away as warm-up iterations. After running MCMC, it is important to check its convergence and precision before extracting and summarizing parameters. There are many techniques to examine the convergence of MCMC. For example, as rules of thumb, we can check whether the trace plot of each parameter shows a caterpillar shape (e.g., see Fig. 3); whether Rhats are smaller than 1.1 (Gelman et al., 2014) or strictly speaking, 1.05 (Vehtari et al., 2021); For precision, we can check whether the bulk and tail effective sample sizes of estimates in each chain are larger than 100, or more strictly than 1000 (Vehtari et al., 2021; Zitzmann & Hecht, 2019). In addition to convergence and effective sample sizes, another big challenge of MCMC is time intensive. There are at least four ways to alleviate it: (1) adopting appropriate MCMC software, e.g., Stan and JAGS; (2) parallel computing; (3) avoiding unnecessary looping (see codes below); (4) improving hardware conditions, e.g., computers with powerful CPU.

A general description of Stan

Stan is a free, open-source probabilistic programming language for specifying statistical models and performing MCMC (Carpenter et al., 2017). Several sampling algorithms are available on Stan, such as the No-U-Turn sampler (NUTS; Hoffman & Gelman, 2014) and Hamiltonian Monte Carlo sampling. It can be called from Stata, R, Python, Matlab, or Julia. It is also worth noting that debates on the comparison of computation efficiency among different MCMC programs have not been settled. For example, Hecht et al. (2021) found that JAGS has better performance in estimating multilevel models under certain simulation conditions while Gelman et al. (2015) claimed that Stan is expected to be more efficient for complex models and scales better than its counterparts in MCMC (e.g., JAGS and WinBUGS) and in a comparison study on structural equation modeling, Merkle et al. (2021) provided evidence in favor of Stan. Despite the ongoing debates, Stan has been widely used in many fields and numerous settings due to its flexibility, ease of use, and free access (e.g., Korner-Nievergelt et al., 2015; Luo & Jiao, 2018; Monnahan et al., 2017).

In running a model on Stan, three blocks are required, that is, *data*, *parameters*, and *model*. In a *data* block, users specify the relevant data information. A *parameters* block states the parameters needed to estimate and required in the *model* block. The *model* block is the key to Stan codes and describes probabilistic models to connect data and parameters, which includes the prior distribution of parameters and the likelihood of the model. In addition to required blocks, three optional blocks are also worth mentioning – i.e., *transformed data*, *transformed parameters*, and *generated quantities*. *Transformed data* and *transformed parameters* blocks allow an elegant way of data manipulation and parameter transformation. *Generated quantities* block can generate many useful quantities, such as log-likelihoods, deviances, posterior prediction, etc. Thus, users who want to calculate model fits and compare model fitness can specify the *generated quantities* block.

Stan codes for IRTree models

Stan codes for response tree models

Every block of Stan starts with a block type statement, and the content is wrapped within a pair of curly brackets {}, where each line ends with a semicolon. Statements within

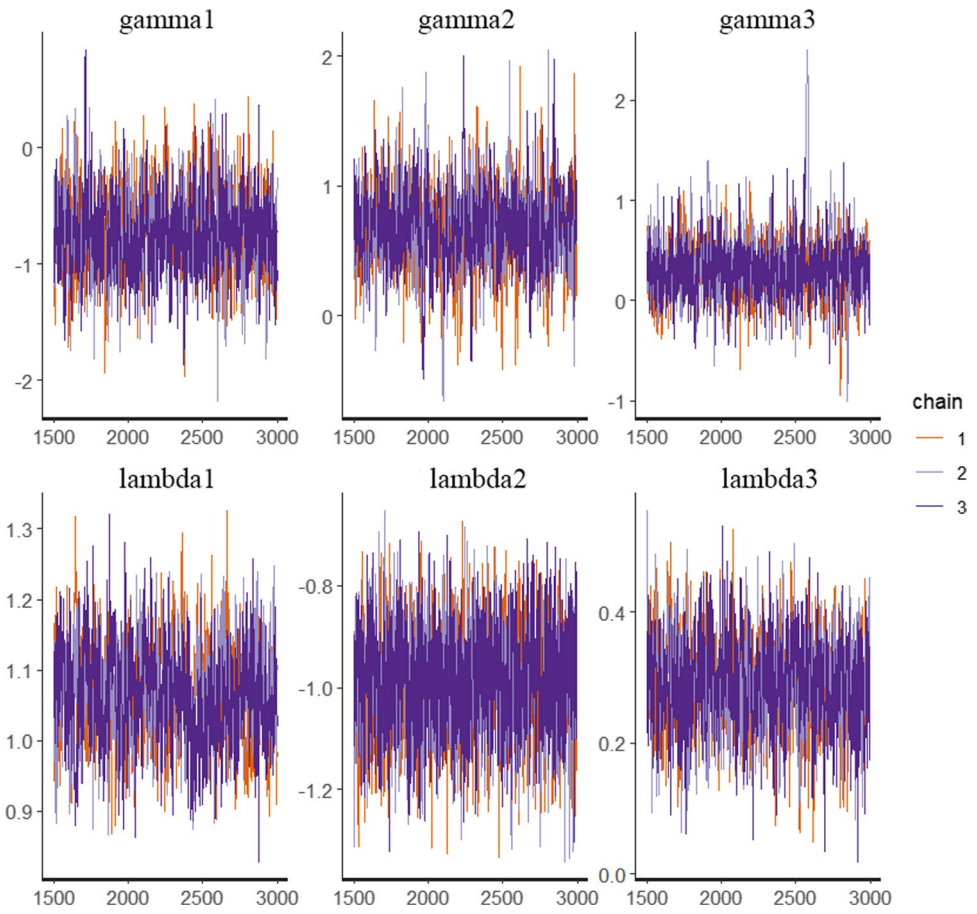

Fig. 3 Trace plots of the six coefficients of interest. *Note. lambda*s denote the coefficients for adolescents–parents differences in self-esteem and two conditional extreme response styles; *gamma*s denote the coefficients for the effects of worded direction

< >, following data and parameters declaration, specify the constraints imposed on the data and parameters, such as the data type and ranges. It's also good practice to write comments (//) on some lines of code to explain their purpose, especially when you want to share them with others.

As shown in Box 1, the first required block is the *data* block. This tutorial uses long format data, where each row contains one response of a respondent to an item. In the *data* block, we declare the number of items, persons, total responses, and data structure of each item, their indices, and responses. As stated in the previous section, the key to IRTree models lies in the construction of pseudo-items. That's also what needs to be specified in the *data* block. Two points are worth mentioning here – 1) we only need to specify the pseudo items responses for each possible observed category rather than all responses, which results in a C×K mapping matrix, where C and K are the numbers of categories and pseudo-items, respectively; 2) Stan doesn't allow missingness, so a new placeholder is in need, and this tutorial chooses -1. Therefore, in the example of Fig. 1 and Table 1, considering a target trait and an extreme response tendency for a scale with four-point items, the pseudo-items matrix is defined in Table 4.

Box 1 Stan codes: the data block of a response tree model

```
data {
  int<lower=1> I;              // # of items
  int<lower=1> P;              // # of persons
  int<lower=1> N;              // # of total responses
  int<lower=1> C;              // # of categories
  int<lower=1> K;              // # of pseudo categories
  int<lower=1, upper=I> ii[N];  // item for n
  int<lower=1, upper=P> pp[N];  // person for n
  int y[N];                    // endorsement or correctness for n
  int mapping[C,K];            // mapping pseudo-items
}
```

Box 2 demonstrates the *parameters* block. We use vectors to specify a person's latent traits, with P vectors in the size of K. In other words, for each respondent, there is a vector that consists of K latent traits. There are good reasons to use vectors rather than matrices in Stan, that is, vectorization will help speed up the sampling procedures and avoid some loops. For item parameters, we first declare the mean and variance of the prior distribution for pseudo-item difficulty parameters. Then pseudo-item difficulty parameters are specified as 'vector[K] beta[I]', where 'beta[i]' is the vector for the *i*th item, which encompasses K pseudo-item difficulties. So are settings for pseudo-items discrimination parameters.

Box 2 Stan codes: the parameters block of the response tree model

```
parameters {
  vector[K] theta[P];          // latent traits matrix

  vector[K] mu_beta;           // mean vector of difficulty parameters
  vector<lower=0>[K] sigma_beta;// variance vector of difficulty parameters
  vector[K] beta[I];           // difficulty parameters of pseudo items

  real mu_alpha;               // mean of discrimination parameters
  real<lower=0>sigma_alpha;    // variance of discrimination parameters
  vector<lower=0>[K] alpha[I];  // discrimination parameters of pseudo items
}
```

Table 4 An example of pseudo-items used in stan that includes a targeted trait and extreme response style

Y	Y1*	Y2*	Y3*
1	0	1	-1
2	0	0	-1
3	1	-1	0
4	1	-1	1

-1 represents not applicable

Box 3 displays the *model* block for the IRTree Stan code, where all the parameters specified in the *parameters* block will be sampled and probabilistic models are made. As stated in the Bayesian estimation section, the mean vector and variance vector of the prior distribution for difficulty are sampled from hyper normal and Cauchy distributions, respectively, and difficulties are drawn from a multivariate normal distribution with the sampled mean and variances. Note that we only enumerate items instead

of their pseudo-items because we have declared vectorization in the *parameter* block and the sampling procedure in the model block will automatically sample a vector for each item. Likewise, the sampling procedures occur to discrimination parameters and latent traits.

On the probabilistic model for the IRTree models, a for-loop is used to enumerate all the observed responses, and within each response, another for-loop is used to enumerate the corresponding pseudo-items responses as specified by the mapping matrix. The *if* statement excludes *not applicable* pseudo-item responses (denoted as -1 in the Stan codes). Then *bernoulli_logit* is the probabilistic model connecting the parameters and dichotomous pseudo-items responses.

Box 3 Stan codes: the model block of the response tree model

```
model {

  mu_beta ~ normal(0,1);       //hyper-prior distribution for difficulty: mean
  sigma_beta ~ cauchy(0,5);    //hyper-prior distribution for difficulty: variance
  beta ~ multi_normal(mu_beta, diag_matrix(sigma_beta)); //prior for difficulty

  mu_alpha ~ normal(0,1);      //hyper-prior distribution for discrimination: mean
  sigma_alpha ~ cauchy(0,5);   //hyper-prior distribution for discrimination: variance
  for (i in 1:I){
     alpha[i] ~ lognormal(mu_alpha,sigma_alpha);       //prior for discrimination
  }

  theta ~ multi_normal(rep_vector(0,K), diag_matrix(rep_vector(1,K))); //prior for latent traits

// enumerate every response
  for (n in 1:N){
    for (k in 1:K){
       if(mapping[y[n],k] != -1){ //exclude NA nodes
         mapping[y[n],k]~bernoulli_logit(alpha[ii[n],k]*(theta[pp[n],k]-beta[ii[n],k]));
       }
    }
  }

}
```

Box 4 displays the optional *generated quantities* block. For the purpose of assessing model fits, we generate log-likelihood in this block. Typical fit indices in Bayesian statistics include Watanabe–Akaike information criterion (WAIC), deviance information criterion (DIC), and leave-one-out cross-validation (LOO-CV).

Box 4 Stan codes: the generated quantities block of the response tree model

```
generated quantities {
  // a matrix to store log-likelihood
  matrix[N,P] log_lik;

  // generate log-likelihood for each response
  for (n in 1: N){
     for (p in 1:P){
       if(mapping[y[n],p] != -1){
         log_lik[n,p]=bernoulli_logit_lpmf(mapping[y[n],p]|alpha[ii[n],p]*(theta[jj[n],p]-
beta[ii[n],p]));
       }else{
         log_lik[n,p]=0;
       }
    }
     }
}
```

Stan codes for latent tree models

Given the limited space and similarities to response tree models, this section will not go through all the blocks of the Stan codes for latent tree models. Instead, this part will only introduce the *model* block (see Box 5), and the complete Stan codes are provided in appendices.

In Box 5, I, L, and N stand for the number of items, the number of latent traits, and the number of total observed responses, respectively. In the last for-loop, we use the ordered_logistic function in Stan, which will be turned into the 2PL model for dichotomous observed responses and GRMs for observed polytomous responses. Within the ordered_logistic function, we use element-wise multiplication (denoted as .*) and summation to calculate respondents' traits as indicated in equation (3).

Box 5 Stan codes: the model block of the latent tree model

```
model {

  mu_beta ~ normal(0,1);       //hyper-prior distribution for the mean of difficulties
  sigma_beta ~ cauchy(0,5);    //hyper-prior distribution for the variance of difficulty

  mu_alpha ~ normal(0,1);      //hyper-prior distribution for the mean of discrimination
  sigma_alpha ~ cauchy(0,5);   //hyper-prior distribution for the variance of discrimination

  for (i in 1:I){
    beta[i] ~ normal(mu_alpha,sigma_alpha);
    alpha[i] ~ lognormal(mu_beta,sigma_beta);
  }

  theta ~ multi_normal(rep_vector(0,L), diag_matrix(rep_vector(1,L)));

  for (n in 1:N){
  y[n] ~ ordered_logistic(sum(alpha[ii[n]].*theta[pp[n]].*mapping[dd[n]]),beta[ii[n]]);
  }

}
```

Stan codes for explanatory IRTree models

Box 6 shows the Stan codes for the doubly explanatory response tree models. Compared with the codes for response tree models, covariates are specified in the *data* block, we declare η, δ, λ, and γ in the *parameter* block and Eqs. (5) and (6) in the *transformed parameters* block. For the *model* block, η, δ, λ, and γ are sampled.

Box 6 Stan codes: doubly explanatory response tree model

```
model {

 for (j in 1:J){
  lambda[j] ~ normal(0,3);    //coeffient on person side
 }
 eta ~ multi_normal(rep_vector(0,K), diag_matrix(rep_vector(1,K)));


 for (v in 1:V){
  gamma[v] ~ normal(0,3);     //coeffient on item side
 }

 sigma_delta ~ cauchy(0,5);
 mu_delta ~ normal(0,1);
 delta ~ multi_normal(mu_delta, diag_matrix(sigma_delta));

 sigma_alpha ~ cauchy(0,5);
 mu_alpha ~ normal(0,1);
 for (i in 1:I){
    alpha[i] ~ lognormal(mu_alpha,sigma_alpha);
 }

 for (n in 1:N){
  for (k in 1:K){
     if(mapping[y[n],k] != -1){
       mapping[y[n],k]~bernoulli_logit(alpha[ii[n],k]*(theta[pp[n],k]-beta[ii[n],k]));
     }
  }
 }

}
```

Executing Stan model

After setting up the Stan models above, the next step is to execute the model, which can be done in R, Python, Cmd, etc. Despite the differences in these platforms, some settings are common, such as the number and the length of the MCMC chain and warms-up iteration. In this tutorial, I take Rstan as an example.

In Box 7, taking *stan* function as an example, five parameters are set. First, the *file* option specifies the location of your Stan code. Second, the *data* option specifies the data that we have prepared for the Stan model with all the elements mentioned in the *data* block of the Stan code. Third, the *chain* option specified the number of Markov chains. Though its default number is four, three chains are also common. Fourth, the *iter* option specifies the number of iterations of each chain, which should "adequately" approximate the posterior distribution of interest. There is no general rule of thumb. But Raftery and Lewis (1992) investigated several posterior distributions and found that 5000 iterations typically lead to reasonable accuracy and 1000 iterations are acceptable when the posterior distribution is known to have a small tail. In our IRTree examples, 3,000 iterations can produce satisfactory results. Furthermore, researchers can also use the effective sample size and Rhats to determine to increase the number of iterations. Fifth, the *warmup* option specifies the number of iterations we want to throw away as warm-up iterations, and the default setting is half of the number of iterations. Finally, the *cores* options specify how many cores you want to use in executing the Markov chains in parallel. We first have to consider the number of cores in our computer. Then a typical setting is the same number of Markov chains. We strongly recommend specifying this option as parallel computation can substantially decrease the running time.

After executing the Stan model, we can obtain posterior point estimates, credible intervals, Rhats, and effective sample size by using the summary method. The latter two values are useful in evaluating the convergence and effectiveness of the MCMC estimation. Rhats generally indicate convergence when they are smaller than 1.05 or 1.1. Note that for effective sample sizes, Stan will pose a warning when the bulk effective sample size (useful for point estimates) or tail effective sample size (useful for credible intervals) per Markov chain is smaller than 100. In addition, we can also call traceplot function in the rstan package to draw the trace of MCMC sampling to examine convergence, and the pars option specify what parameters we want to check.

Upon finishing the above steps and obtaining convergent results, we can move on to visualizing and making inferences on the estimated parameters.

Box 7 Executing Stan model in R

```
fit_rse_ex <- stan(file='Documents/explanatory_irtree_2pl.stan', data=data_rse_ex, chain=3,
cores=3, iter = 3000, warmup=1500)

fit_rse_summary <- summary(fit_rse)$summary

traceplot(fit_rse, pars = c('alpha','beta'))
```

Empirical study

To illustrate the application of Bayesian IRTree models, we conducted an empirical study on the data collected in the Oxford Achieving Resilience during COVID-19 (ARC) study (Parsons et al., 2022) using an explanatory response tree model (for an empirical study on latent tree models, please refer to supplementary material). It is a longitudinal study, which took place between March 2020 and August 2021. It covered several mental health assessments and the targeted population was adolescents (aged 13 to 18) and their parents. For our empirical studies, we concentrate on the Rosenberg self-esteem scale, which consists of four negatively worded items and six positively worded items on a four-point scale. All negatively worded items were recoded before analysis.

In the study, we used doubly explanatory response tree models to address two research questions: 1) whether pseudo-item difficulties differentiate in terms of the worded direction (negative = 0 vs. positive = 1); 2) whether there is a significant difference in self-esteem and extreme response style between adolescents and parents (adolescents = 0 vs. parents = 1). To assess the targeted trait and extreme response styles, we employed the pseudo-items mapping schemes displayed in Table 2 and 2PL for each pseudo-item. Therefore, there were three coefficients related to research question one, representing the effect of worded direction on difficulties of the targeted trait pseudo-item (γ_1), extreme response style pseudo-item at lower targeted trait (γ_2), and extreme response style pseudo-item at higher targeted trait (γ_3). Likewise, there are also three coefficients accounting for the adolescent-parent difference in self-esteem (λ_1) and extreme response style (λ_2 at lower self-esteem, λ_3 at higher self-esteem). In terms of the amount of data, we only kept the first wave data and used listwise deletion to avoid over complications in dealing with repeated measurement and missingness, which ended up with 1499 respondents. In executing the Stan code, we used three chains, each of which contained 3000 iterations, with the first half as warm up. Then we checked the convergence of the Bayesian IRTree models using trace plots and Rhats, extracted the posterior distribution of parameters of interest, and made inference from them.

Results

Figure 3 illustrates the trace plots of the six coefficients related to research question one and two. The trace plot indicated the convergence of the estimated parameters. Besides, the Rhat values of the six coefficients were less than 1.05, and the smallest effective size is 705. According to the rule of thumb, they showed a sight of convergence and effectiveness.

Figure 4 displays the 95% credible intervals of the posterior distribution for the parameters of interest on both person and item sides. Results show that parents have significantly higher self-esteem (λ_1) and extreme response style conditional on agreement on self-esteem (λ_3) than adolescents. For the extreme response style conditional on disagreement on self-esteem (λ_2), parents show significantly lower scores than adolescents. In other words, parents tend to choose extreme high categories while adolescents prefer extreme low categories.

On the item side, positively worded items have significantly lower difficulties in endorsement than negatively worded items (γ_1). Moreover, negatively worded direction tends to elicit more extreme responses than positively-worded items as evidenced by the significantly lower difficulties of the pseudo-items concerning extreme response difficulties at (γ_2).

Discussion and conclusion

IRTree models help reveal the multiple processes underlying observed responses through the combination of IRT and expert-defined mapping schemes, whereby item quality can be investigated in more detail, interpretability can be enhanced, and some of respondents' biases can be controlled. For example, response styles are a serious bias in surveys (van Vaerenbergh & Thomas, 2013), and IRTree models show some promising features in controlling for such biases. There is no wonder the last decade has seen

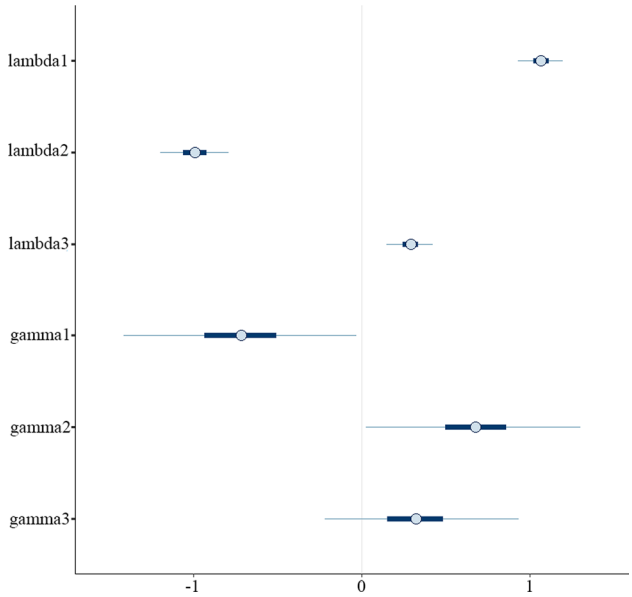

Fig. 4 95% Credible intervals of posterior distributions for adolescents–parents differences and the effects of worded direction on pseudo-items. *Note. Dark blue shades* indicate 50% credible intervals; *light blue lines* indicate 95% credible intervals; *circles* indicate mean estimates; *lambdas* denote the coefficients for adolescents–parents differences in self-esteem and two conditional extreme response styles; *gammas* denote the coefficients for the effects of worded direction

the proliferation of IRTree models. Nevertheless, relevant tutorials remain limited, which hinders their applications. This paper introduces the Stan codes for IRTree models and their extension to the explanatory IRT framework with an aim to facilitate a wider application and research of IRTree models. The Bayesian approach can employ various prior distributions on the parameters of interest based on the available information and produce posterior distributions, where credible intervals are easy to obtain. The flexibility of Bayesian IRT not only makes running IRTree models possible but also enables a variety of extensions, such as in an explanatory way. Bayesian IRTree models show promising features in controlling biases, increasing interpretability, and opening to numerous extensions, such as hierarchical structure, exploratory IRTree models, etc. Researchers and practitioners can alter the codes to accommodate their own research questions but need to be cautious that information captured by nodes might not be mutually independent, for example, the extreme response style nodes might contain some information that should be captured the node of the target trait. In this paper, we demonstrate how to incorporate both item and respondents' properties into IRTree models to address two research questions in the empirical study and how to make inference of the Stan results.

On the other hand, although MCMC has advantages in calculating high-dimensional posterior distributions, we acknowledge that challenges exist in terms of convergence, effective sample size, acceptance rates, and time-consuming procedure, and thus this paper also presents ways to assess convergence and effectiveness (e.g., Rhats, bulk and tail effective sample sizes) and provides some suggestions to tackle with some of these challenges, such as using vectorization.

Moreover, there are many opportunities for future research. First, the Bayesian IRTree models used in this paper don't address the long-standing question that the sequential order of the IRTree processes is not fixed, that is, different orders of pseudo-items result in the same mathematical formulation. For example, in the response tree model, we assume that respondents go through the first node concerning targeted traits and then through the second node about extreme response tendency. However, mathematically speaking, this restriction has not been imposed yet. Respondents can first deal with extreme response tendencies and then consider their agreement on targeted traits, which will result in the same mathematical formulation as shown in Eq. (2). Future research can go beyond the conventional statistical models and explore how to incorporate the sequence of nodes into Bayesian IRTree models. Second, to date, the pseudo-items are prespecified by experts and researchers. Though it guarantees high interpretability, we lose some power of exploration. For example, there might be more than one response style in the surveys and complicated mechanisms. An exploratory Bayesian IRTree model can help shed light on such questions. Third, since this is a Stan tutorial, JAGS and BUGS codes for IRTree models are out of scope, but they have some similarities in coding, and thus future researchers can also consider writing the JAGS and BUGS codes by adopting the ideas mentioned in this paper.

Appendices

A. Stan codes

Box A Full Stan code for response tree models

```
data {
  int<lower=1> I;              // # of items
  int<lower=1> P;              // # of persons
  int<lower=1> N;              // # of total responses
  int<lower=1> C;              // # of categories
  int<lower=1> K;              // # of pseudo categories
  int<lower=1, upper=I> ii[N];  // item for n
  int<lower=1, upper=P> pp[N];  // person for n
  int y[N];                    // endorsement or correctness for n
  int mapping[C,K];            // mapping pseudo items
}

parameters {
  vector[K] theta[P];          // latent traits matrix

  vector[K] mu_beta;           // mean vector of difficulty parameters
  vector<lower=0>[K] sigma_beta;// variance vector of difficulty parameters
  vector[K] beta[I];           // difficulty parameters of pseudo items

  real mu_alpha;               // mean of discrimination parameters
  real<lower=0>sigma_alpha;    // variance of discrimination parameters
  vector<lower=0>[K] alpha[I];  // discrimination parameters of pseudo items
}

model {

  mu_beta ~ normal(0,1);       //hyper-prior distribution for difficulty: mean
  sigma_beta ~ cauchy(0,5);    //hyper-prior distribution for difficulty: variance
  beta ~ multi_normal(mu_beta, diag_matrix(sigma_beta)); //prior for difficulty

  mu_alpha ~ normal(0,1);      //hyper-prior distribution for discrimination: mean
  sigma_alpha ~ cauchy(0,5);   //hyper-prior distribution for discrimination: variance
  for (i in 1:I){
     alpha[i] ~ lognormal(mu_alpha,sigma_alpha);       //prior for discrimination
  }

  theta ~ multi_normal(rep_vector(0,K), diag_matrix(rep_vector(1,K))); //prior for latent traits

// enumerate every response
  for (n in 1:N){
    for (k in 1:K){
       if(mapping[y[n],k] != -1){ //exclude NA node
         mapping[y[n],k]~bernoulli_logit(alpha[ii[n],k]*(theta[pp[n],k]-beta[ii[n],k])); // Likelihood
       }
    }
  }
}
```

Box B Full Stan codes for doubly explanatory response tree models

```
data {
  int<lower=1> I;              // # of items
  int<lower=1> P;              // # of persons
  int<lower=1> N;              // # of total responses
  int<lower=1> C;              // # of categories
  int<lower=1> K;              // # of pseudo categories
  int<lower=1, upper=I> ii[N];  // item for n
  int<lower=1, upper=P> pp[N];  // person for n
  int y[N];                    // endorsement or correctness for n
  int mapping[C,K];            // mapping pseudo items
  int<lower=1> J;              // # of person covariates
  int<lower=1> V;              // # of item covariates
  matrix[P,J] personcov;         // covariates for persons
  matrix[I,V] itemcov;           // covariates for items
}
parameters {
  vector[K] eta[P];
  real mu_alpha;
  real<lower=0>sigma_alpha;      // variance of discrimination parameters
  vector<lower=0>[K] alpha[I];   //  discrimination parameters
  vector[K] mu_delta;
  vector<lower=0>[K] sigma_delta;
  vector[K] delta[I];
  vector[J] lambda[K];          //coefficients for person properties
  vector[V] gamma[K];           //coefficients for item properties
}

transformed parameters {
  vector[K] theta[P];           // latent traits
  vector[K] beta[I];            // difficulties
  for (p in 1:P){
    for(k in 1:K){
      theta[p,k] = sum(lambda[k,]*personcov[p,])+eta[p,k];
    }
  }
  for (i in 1:I){
    for(k in 1:K){
      beta[i,k] = sum(gamma[k,]*itemcov[i,]) +delta[i,k];
    }
  }
}
model {

  for (j in 1:J){
    lambda[j] ~ normal(0,3);    //coeffient on person side
  }
  eta ~ multi_normal(rep_vector(0,K), diag_matrix(rep_vector(1,K)));

  for (v in 1:V){
    gamma[v] ~ normal(0,3);     //coeffient on item side
  }

  sigma_delta ~ cauchy(0,5);
  mu_delta ~ normal(0,1);
  delta ~ multi_normal(mu_delta, diag_matrix(sigma_delta));

  sigma_alpha ~ cauchy(0,5);
  mu_alpha ~ normal(0,1);
  for (i in 1:I){
     alpha[i] ~ lognormal(mu_alpha,sigma_alpha);
  }


  for (n in 1:N){
    for (k in 1:K){
       if(mapping[y[n],k] != -1){
         mapping[y[n],k]~bernoulli_logit(alpha[ii[n],k]*(theta[pp[n],k]-beta[ii[n],k]));
       }
    }
  }

}
```

Box C Full Stan codes for latent tree models

```
data {
  int<lower=1> I;              // # of items
  int<lower=1> P;              // # of persons
  int<lower=1> N;              // # total responses
  int<lower=1> C;              // # of categories
  int<lower=1> L;              // # of latent traits
  int<lower=1> K;              // # of nodes
  int<lower=1, upper=I> ii[N];  // item for n
  int<lower=1, upper=P> pp[N];  // person for n
  int<lower=1, upper=K> dd[N];  // nodes for n
  int y[N];                    // correctness for n
  vector[L] mapping[K];            // mapping pseudo items
}
parameters {
  vector[L] theta[P];             // abilities

  real mu_alpha;               // mean of discrimination parameters
  real<lower=0>sigma_alpha;     // variance of discrimination parameters
  vector<lower=0>[L] alpha[I];      //discrimination parameters of pseudo items

  real mu_beta;            // mean vector of difficulty parameters
  real<lower=0> sigma_beta;// variance vector of difficulty parameters
  ordered[C-1] beta[I];            //category difficulty
}

model {

  mu_beta ~ normal(0,1);       //hyper-prior distribution for difficulty: mean
  sigma_beta ~ cauchy(0,5);    //hyper-prior distribution for difficulty: variance

  mu_alpha ~ normal(0,1);      //hyper-prior distribution for discrimination: mean
  sigma_alpha ~ cauchy(0,5);   //hyper-prior distribution for discrimination: variance

  for (i in 1:I){
    beta[i] ~ normal(mu_beta,sigma_beta);
    alpha[i] ~ lognormal(mu_alpha,sigma_alpha);
  }
  theta ~ multi_normal(rep_vector(0,L), diag_matrix(rep_vector(1,L)));
  for (n in 1:N){
  y[n] ~ ordered_logistic(sum(alpha[ii[n]].*theta[pp[n]].*mapping[dd[n]]),beta[ii[n]]);
  }

}
```

Box C Full Stan code for explanatory latent tree models

```
data {
  int<lower=1> I;              // # of items
  int<lower=1> P;              // # of persons
  int<lower=1> N;              // # total responses
  int<lower=1> C;              // # of categories
  int<lower=1> L;              // # of latent traits
  int<lower=1> K;              // # of nodes
  int<lower=1, upper=I> ii[N];  // item for n
  int<lower=1, upper=P> pp[N];  // person for n
  int<lower=1, upper=K> dd[N];  // nodes for n
  int y[N];                    // correctness for n
  vector[L] mapping[K];            // mapping pseudo items

  int<lower=1> J;              // # of person covariates
  int<lower=1> V;              // # of item covariates
  matrix[P,J] personcov;          // covariates for persons
  matrix[I,V] itemcov;            // covariates for items
}
parameters {
  vector[K] eta[P];

  real mu_alpha;
  real<lower=0>sigma_alpha;      // variance of discrimination parameters
  vector<lower=0>[K] alpha[I];   //  discrimination parameters
  real mu_delta;
  real<lower=0> sigma_delta;
  ordered[C-1] delta[I];

  vector[J] lambda[K];          //coefficients for person properties
  vector[V] gamma;           //coefficients for item properties
}

transformed parameters {
  vector[K] theta[P];           // latent traits
  vector[K] beta[I];            // difficulties

  for (p in 1:P){
    for(k in 1:K){
      theta[p,k] = sum(lambda[k,]*personcov[p,])+eta[p,k];
    }
  }

  for (i in 1:I){
     beta[i] = sum(gamma .* to_vector(itemcov[i,])) +delta[i];
  }

}
model {
  for (j in 1:J){
    lambda[j] ~ normal(0,3);    //coeffient on person side
  }
  eta ~ multi_normal(rep_vector(0,K), diag_matrix(rep_vector(1,K)));
  for (v in 1:V){
    gamma[v] ~ normal(0,3);     //coeffient on item side
  }
  mu_delta ~ normal(0,1);       //hyper-prior distribution for difficulty: mean
  sigma_delta ~ cauchy(0,5);    //hyper-prior distribution for difficulty: variance
  mu_alpha ~ normal(0,1);       //hyper-prior distribution for discrimination: mean
  sigma_alpha ~ cauchy(0,5);   //hyper-prior distribution for discrimination: variance
  for (i in 1:I){
    delta[i] ~ normal(mu_delta,sigma_delta);
    alpha[i] ~ lognormal(mu_alpha,sigma_alpha);

  }

  for (n in 1:N){
  y[n] ~ ordered_logistic(sum(alpha[ii[n]].*theta[pp[n]].*mapping[dd[n]]),beta[ii[n]]);
  }
}
```

B. R Codes for the empirical study

```
########################################
####    explanatory response tree model    #####
########################################
### negatively-worded = 0; positively-worded = 1
itemcov <- rep(1,10)
itemcov[c(2,5,6,8,9)] <- 0

### child=0; parent/carer=1
personcov <- oxf[,c('record_id','group')]
personcov <- personcov[personcov$record_id %in% rse_u$record_id,]
personcov <- personcov[!duplicated(personcov$record_id),]-1
data_rse_ex <-
list(I=length(unique(dt_rse$item)),P=length(unique(dt_rse$person)),N=nrow(dt_rse),
                C=4,K=3,mapping=map_rse,
ii=as.integer(factor(x=dt_rse$item)),pp=as.integer(factor(x=dt_rse$person)),y=dt_rse$respons
e,
                V=1,J=1,itemcov=matrix(itemcov), personcov=matrix(personcov$group)
)
### run stan model
fit_rse_ex <- stan(file='explanatory_response_irtree_2pl.stan',data=data_rse_ex,
               chain=3,iter = 3000, warmup=1500,cores=3)
### summarize results
fit_rse_ex_summary <- summary(fit_rse_ex)$summary
traceplot(fit_rse_ex,c('gamma','lambda'))
### visualize results
mcmc_intervals(as.array(fit_rse_ex),pars=c('lambda[1,1]','lambda[2,1]','lambda[3,1]',
                                'gamma[1,1]','gamma[2,1]','gamma[3,1]'),prob_outer=0.95)
```

Empirical study II: Latent tree models

In empirical study II, the research question is how self-esteem changed over time. There were three waves of data, including baseline self-esteem and two growths in self-esteem, whose mapping scheme is given in Table 2. GRM was the probabilistic model that connected responses and latent traits. Nevertheless, not all respondents completed the three waves of the survey. In this study, we selected respondents who responded to at least two waves of the survey. In total, there were 224 respondents.

Results

Since there were hundreds of parameters but limited space, Appendix Fig. 5 only demonstrated six exemplar trace plots, which showed a mixed and caterpillar-like shape. Their Rhat values were less than 1.05, and the smallest effective sample size is larger than 438. Hence, the parameters of interest converged.

Appendix Fig. 6 displays the growth in self-esteem between the first and second waves. Adolescents had a distribution with a mean around zero, while parents' growth distribution had a negative mean. Nevertheless, they did not differ significantly. Appendix Fig. 7 shows the distribution of growth between the second and third waves. It became denser than in Appendix Fig. 6. Taken together, the change in self-esteem between first and second wave is larger than that between second and third wave and not much difference was observed between the adolescent and parent groups.

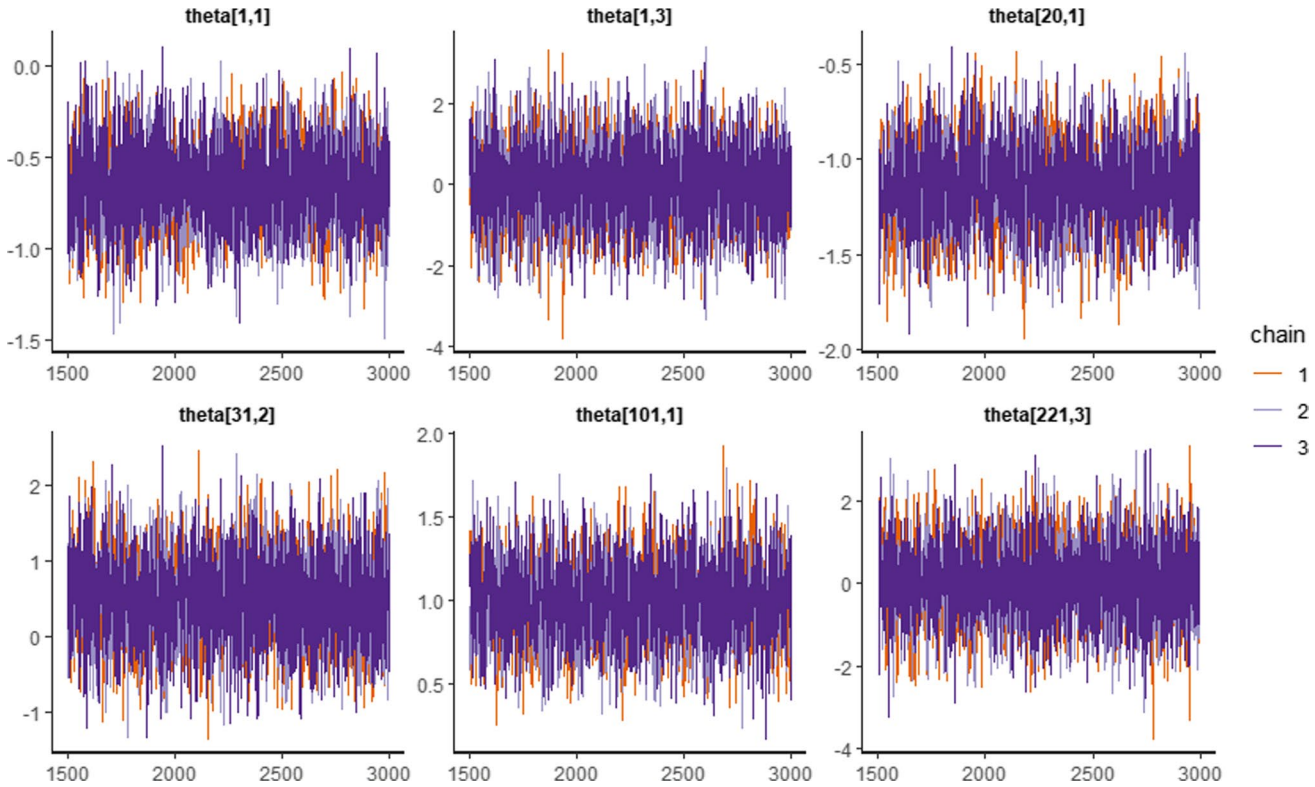

Fig. 5 Trace plot examples of the latent tree models

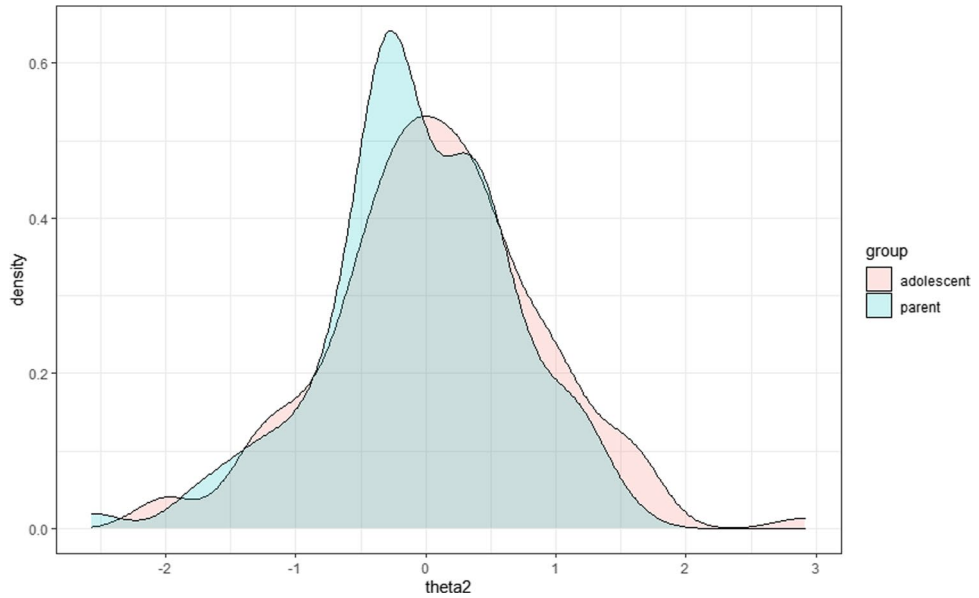

Fig. 6 The distribution of growth in self-esteem between the first and second wave

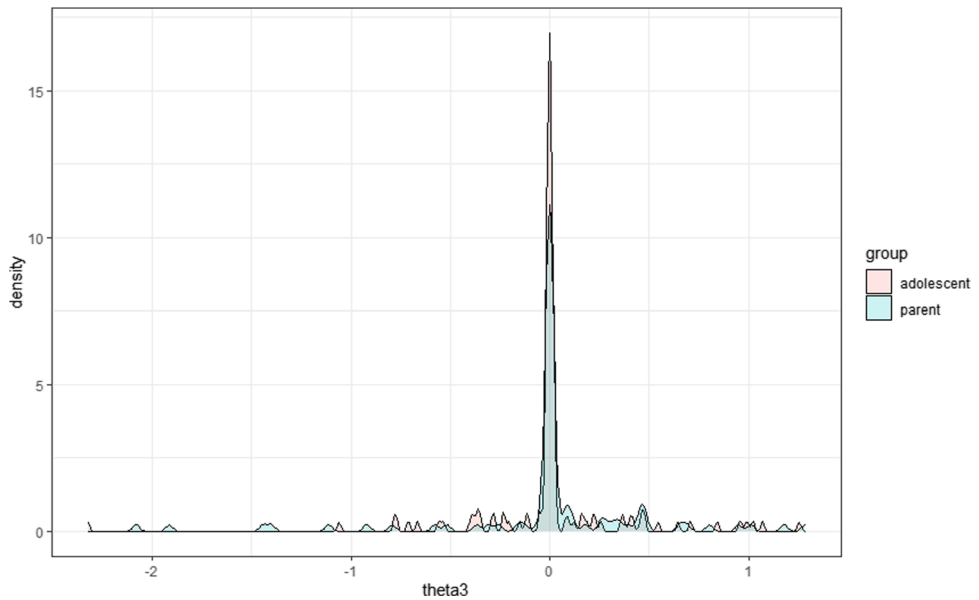

Fig. 7 The distribution of growth in self-esteem between the second and third wave

References

Ames, A. J., & Leventhal, B. C. (2023). Application of a longitudinal IRTree model: Response style changes over time. *Assessment*, 30(2), 332–347.

Ames, A. J., & Myers, A. J. (2021). Explaining variability in response style traits: A covariate-adjusted IRTree. *Educational and Psychological Measurement, 81*(4), 756–780.

Ames, A. J., & Samonte, K. (2015). Using SAS PROC MCMC for item response theory models. *Educational and Psychological Measurement, 75*(4), 585–609.

Carpenter, B., Gelman, A., Hoffman, M.D., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., & Riddell, A. (2017). Stan: A probabilistic programming language. *Journal of Statistical Software*, *76*, 1–37.

Curtis, S. M. (2010). BUGS code for item response theory. *Journal of Statistical Software, 36*, 1–34.

De Boeck, P., & Partchev, I. (2012). IRTrees: Tree-based item response models of the GLMM family. *Journal of Statistical Software, 48*, 1–28.

Debeer, D., Janssen, R., & De Boeck, P. (2014, April). *Modeling missing-data processes: An IRTree-based approach* [Paper presentation]. National Council on Measurement in Education (NCME) Annual Meeting 2014, Philadelphia (PA), USA. https://lirias.kuleuven.be/retrieve/294464. Assessed 26 Jan 2023.

Debeer, D., Janssen, R., & De Boeck, P. (2017). Modeling skipped and not-reached items using IRTrees. *Journal of Educational Measurement, 54*(3), 333–363.

Fox, J. P., & Glas, C. A. (2001). Bayesian estimation of a multilevel IRT model using Gibbs sampling. *Psychometrika, 66*(2), 271–288.

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., & Rubin, D. B. (2014). *Bayesian Data Analysis* (3rd ed.). CRC Press.

Gelman, A., Lee, D., & Guo, J. (2015). Stan: A probabilistic programming language for Bayesian inference and optimization. *Journal of Educational and Behavioral Statistics, 40*(5), 530–543.

Hecht, M., Weirich, S., & Zitzmann, S. (2021). Comparing the MCMC efficiency of JAGS and Stan for the multi-level intercept-only model in the covariance-and mean-based and classic parametrization. *Psych, 3*(4), 751–779.

Hoffman, M. D., & Gelman, A. (2014). The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research, 15*(1), 1593–1623.

Huang, H. Y. (2020). A mixture IRTree model for performance decline and nonignorable missing data. *Educational and Psychological Measurement, 80*(6), 1168–1195.

Jeon, M., & De Boeck, P. (2016). A generalized item response tree model for psychological assessments. *Behavior Research Methods, 48*(3), 1070–1085.

Jeon, M., & De Boeck, P. (2019). Evaluation on types of invariance in studying extreme response bias with an IRTree approach. *British Journal of Mathematical and Statistical Psychology, 72*(3), 517–537.

Khorramdel, L., von Davier, M., & Pokropek, A. (2019). Combining mixture distribution and multidimensional IRTree models for the measurement of extreme response styles. *British Journal of Mathematical and Statistical Psychology, 72*(3), 538–559.

Kim, N., & Bolt, D. M. (2021). A mixture IRTree model for extreme response style: Accounting for response process uncertainty. *Educational and Psychological Measurement, 81*(1), 131–154.

Korner-Nievergelt, F., Roth, T., Von Felten, S., Guélat, J., Almasi, B., & Korner-Nievergelt, P. (2015). *Bayesian Data Analysis in Ecology Using Linear Models with R, BUGS, and Stan*. Academic Press.

LaHuis, D. M., Blackmore, C. E., Bryant-Lees, K. B., & Delgado, K. (2019). Applying item response trees to personality data in the selection context. *Organizational Research Methods, 22*(4), 1007–1018.

Levy, R., & Mislevy, R. J. (2017). *Bayesian Psychometric Modeling*. CRC Press.

Lunn, D. J., Thomas, A., Best, N., & Spiegelhalter, D. (2000). WinBUGS-a Bayesian modelling framework: Concepts, structure, and extensibility. *Statistics and Computing, 10*(4), 325–337.

Luo, Y., & Jiao, H. (2018). Using the Stan program for Bayesian item response theory. *Educational and Psychological Measurement, 78*(3), 384–408.

Meiser, T., Plieninger, H., & Henninger, M. (2019). IRTree models with ordinal and multidimensional decision nodes for response styles and trait-based rating responses. *British Journal of Mathematical and Statistical Psychology, 72*(3), 501–516.

Merkle, E. C., Fitzsimmons, E., Uanhoro, J., & Goodrich, B. (2021). Efficient Bayesian structural equation modeling in Stan. *Journal of Statistical Software, 100*, 1–22. https://doi.org/10.18637/jss.v100.i06

Monnahan, C. C., Thorson, J. T., & Branch, T. A. (2017). Faster estimation of Bayesian models in ecology using Hamiltonian Monte Carlo. *Methods in Ecology and Evolution, 8*(3), 339–348.

Park, M. (2021). *Using Item Response Tree Models for Studying Response Behaviors* (T). University of British Columbia. Retrieved from https://open.library.ubc.ca/collections/ubctheses/24/items/1.0403712. Assessed 23 Jan 2023.

Park, M., & Wu, A. D. (2019). Item response tree models to investigate acquiescence and extreme response styles in Likert-type rating scales. *Educational and Psychological Measurement, 79*(5), 911–930.

Parsons, S., Todorovic, A., Lim, M.C., Songco, A., & Fox, E. (2022). Data and Protocol for the Oxford Achieving Resilience During COVID-19 (ARC) Study. *Journal of Open Psychology Data, 10*(1), 4. https://doi.org/10.5334/jopd.56

Plieninger, H. (2021). Developing and applying IR-tree models: Guidelines, caveats, and an extension to multiple groups. *Organizational Research Methods, 24*(3), 654–670.

Plummer, M. (2003, March). JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In: *Proceedings of the 3rd International Workshop on Distributed Statistical Computing* (Vol. 124, No. 125.10, pp. 1–10).

Raftery, A. E., & Lewis, S. (1992). How many iterations in the Gibbs sampler? *Bayesian Statistics, 4*, 763–773.

Van Vaerenbergh, Y., & Thomas, T. D. (2013). Response styles in survey research: A literature review of antecedents, consequences, and remedies. *International Journal of Public Opinion Research, 25*(2), 195–217.

Vehtari, A., Gelman, A., Simpson, D., Carpenter, B., & Bürkner, P. C. (2021). Rank-normalization, folding, and localization: An improved R for assessing convergence of MCMC (with discussion). *Bayesian Analysis, 16*(2), 667–718.

Wilson, M., & De Boeck, P. (2004). Descriptive and explanatory item response models. In P. De Boeck & M. Wilson (Eds.), *Explanatory Item Response Models*. Springer.

Zitzmann, S., & Hecht, M. (2019). Going beyond convergence in Bayesian estimation: Why precision matters too and how to assess it. *Structural Equation Modeling, 26*(4), 646–661.

Open practices statement Oxford Achieving Resilience during COVID-19 (ARC) study data can be obtained from OSF [https://osf.io/4b85w/] and codes are provided in appendices.