Journal of Behavioral and Experimental Finance 16 (2017) 33–38

Contents lists available at ScienceDirect

Journal of Behavioral and Experimental Finance

journal homepage: www.elsevier.com/locate/jbef

Full length article

oTree: Ready-made apps for risk preference elicitation methods☆

Felix Holzmeister

Leopold-Franzens Universität, Department of Banking and Finance, Universitaetsstrasse 15, Innsbruck, Austria

ARTICLE INFO

Article history:
Received 4 August 2017
Accepted 31 August 2017
Available online 8 September 2017

Keywords:
oTree
Risk preference elicitation
Experimental economics

ABSTRACT

Since the emergence of experimental methods in economics, a variety of methodologies to elicit and classify individual-level risk preferences have evolved. This paper presents ready-to-use software applications for widely used choice list methodologies for use with *oTree*, including (i) multiple price lists, (ii) certainty equivalent tests, (iii) staircase methods, and (iv) single choice lists. The parameterization and configuration of different features of a task are administered by specifying predefined variables in a single, thoroughly documented file. All apps feature responsive graphical designs, are prepared for multilingual use, and include predefined "bots" for automated testing. By this means, each task can be implemented in the laboratory, the field, or online within only a few minutes.

1. Introduction

Risk is an integral part of many economic decisions. Thus, it does not come at a surprise that the elicitation of individual-level risk preferences appears on the agenda of many economic experiments in different fields of research. Ever since the early work on risk aversion (Pratt, 1964; Arrow, 1965), scholars have been intrigued by the question how to properly measure preferences towards risks, or, more precisely, how to determine the curvature of an individual's utility function.

Since the emergence of experimental methods in economics and finance, numerous methodologies to elicit and classify participants' risk attitudes in the laboratory have evolved. Price list formats probably constitute the most widely used method.[1] On the one hand, the prevalent use of the choice list format might be due to the fact that a preference relation is defined as binary relation over alternatives; the observation of a binary choice, thus, serves as a self-evident tool for the elicitation of revealed preferences (Freeman et al., 2017). On the other hand, the popularity of the choice list method might stem from its transparency and its simplicity in terms of incentives for truthful revelation of preferences (Andersen et al., 2006).[2]

In recent decades, various different choice list methods have been introduced. What they typically have in common is that they ask subjects to reveal their preference for different alternatives to approximate indifference within a set of systematically modified lotteries. In the following discussion of different methods, $(x, p; y)$ denotes a two-outcome lottery that assigns probability p to outcome x and probability $1 - p$ to outcome y.

This paper presents four ready-to-use software modules for different choice list methods to elicit individual-level risk preferences under experimental conditions for use with *oTree* (Chen et al., 2016). In particular, ready-made apps for implementing (i) a multiple price list between pairs of lotteries with fixed lottery outcomes and varying probabilities (e.g Holt and Laury, 2002), (ii) a multiple choice list between a lottery and a sure payoff, either varying the prospect or the safe choice (e.g. Abdellaoui et al., 2011), (iii) an iterative choice list or 'staircase' elicitation procedure with a fixed lottery and varying certain payoffs (e.g. Falk et al., 2016a), and (iv) a single choice list, requiring subjects to choose the preferred among a set of given lotteries (e.g. Eckel and Grossman, 2002).

☆ All software application referred to in this article are free of charge and licensed under an adapted MIT open source license with a citation requirement. Any use of the software – whether as a whole or in parts – implies the acceptance of the license agreement provided in the download packages. The applications are available for download at the website of the author (www.holzmeister.biz) and the journal's GitHub repository (https://github.com/JBEF); demo versions are available at https://cl-demo.herokuapp.com.

1 *E-mail address:* felix.holzmeister@uibk.ac.at.

1 Other methodologies to measure individual-level risk preferences include elicitation methods for reservation prices (Becker et al., 1964; Kachelmeier and Shehata, 1992), allocation choices between a safe and a risky asset (Gneezy and Potters, 1997; Charness and Gneezy, 2012), laboratory versions of the game show *Deal or No Deal* (Post et al., 2008; Deck et al., 2014), graphical methods like the Balloon Analogue Risk Task (Lejuez et al., 2002) or the Bomb Risk Elicitation Task (Crosetto and Filippin, 2013), and non-incentivized questionnaires (Weber et al., 2002; Dohmen et al., 2011), without claiming to be complete.

2 The advantages and disadvantages of different methodologies to elicit and assess individual risk attitudes have been discussed repeatedly, as for instance by Harrison and Ruström (2008) or Charness et al. (2013). Several papers also examined the consistency of revealed preferences among different elicitation methods, both between subjects (e.g. Crosetto and Filippin, 2015) and within-subjects (e.g Csermely and Rabas, 2016). It is not within the scope of this paper to discuss virtues and drawbacks of different methods nor to argue which of the methodologies is the most suitable approach to elicit risk preferences.

http://dx.doi.org/10.1016/j.jbef.2017.08.003
2214-6350/

As an online, open-source, and object-oriented web framework, *oTree* offers a platform-independent environment utilizable on any device including desktop computers, tablets, and smartphones. Similar to the app introduced by Holzmeister and Pfurtscheller (2016), the apps presented below scoop the advantages of *oTree*, implying state-of-the-art graphical display, seamless integration into existing projects, and high levels of flexibility. All apps are easily set up and configured by specifying several predefined variables in a single file. By this means, relevant parameters (such as lottery payoffs and their corresponding probabilities) as well as variations in the experimental protocol or graphical display can be configured with ease in only a few minutes. Thereby, the apps offer a time-saving and efficient way to implement each of the particular risk elicitation methods with any arbitrary parameterization in different experimental environments.

2. Setup and usage

Each of the four different methodological procedures to elicit individual-level risk preferences is programmed as a stand-alone *oTree* app. Therefore, the apps can be utilized and seamlessly integrated in any experiment conducted with the *oTree* framework by copying the app's folder into *oTree*'s project directory and adapting the session configurations in *oTree*'s `settings.py` file.

The four *oTree* apps discussed below facilitate the implementation of the respective elicitation method as well as several variations which have been applied in previous research. In a user-friendly and straightforward manner, thoroughly documented variables are specified in a single file (`config.py`) at the root of an app's directory. The file `config.py` consists of *oTree*'s `Constants` class and comprises several predefined variables to be specified in order to determine different features of a task. As all apps are programmed as standard *oTree* applications, pre-implemented configurations can easily be altered or extended by custom-designed features.

To facilitate usage of the apps in a broad range of applications, the graphical display of all apps is optimized for different devices including tablets and smartphones (in landscape mode) with a minimum screen resolution of 560 pixels (see Fig. 1) and has been extensively tested in the most commonly used web browsers by the author. Utilizing responsive graphical designs, the apps support *oTree*'s functionality to run experiments in the laboratory, online, and in the field.

All apps are prepared for multilingual use based on Django's `i18n` internationalization routines, offering a convenient toolset for translating the app into different languages. That is, all texts displayed to subjects are tagged in the Python scripts and HTML templates such that translations are disentangled from the source code. Preparing translations in auto-generated message files allows for running the same app in any arbitrary language without modifying the Python scripts or HTML templates. Instructions on how to utilize the translation features are included in the download packages. Similarly, all numbers referring to monetary units and being displayed to subjects are flagged with currency field tags such that real world or experimental currency units can be globally defined in `settings.py`. Similar to translation routines, the segregation of numbers and units facilitates usage of the same piece of software in any arbitrary currency denomination without modifying the source code. Moreover, each app features "bots" (via `tests.py`) allowing for automated testing of the module by simulating participant behavior. To test the apps in a more full and realistic way, tests can be launched using *oTree*'s "browser bots", simulating the app in an actual web browser. Further information on how to run tests using command line and browser bots are included in the download folders.

All relevant information for utilizing the apps are summarized in documentation files in PDF-format, included in the download packages. The documentations include instructions on how to install the apps and outline the predefined variables in `config.py`. Instructions on how to utilize the `i18n` internationalization routines for multilingual use of the apps and how to run automated tests using command line and browser 'bots' are provided in separate `*.pdf`-files.

2.1. Multiple price list (MPL)

The MPL module allows conducting the multiple price list methodology proposed by Holt and Laury (2002). This kind of multiple choice list is characterized by a set of dichotomous choices between lotteries with fixed payoffs, with probabilities of high and low outcomes varying for each choice. The probabilities of the high lottery outcome in choice i, p_i, are determined by the number of choices, n, i.e. $p_i = i/n$.[3] Thus, subjects face a menu of n binary choices between lottery $L^A = (a_h, i/n;\ a_l)$ and lottery $L^B = (b_h, i/n;\ b_l)$ for $i = 1, 2, \ldots, n$, where $b_h > a_h > a_l > b_l$.[4] The only parameters to be specified in `config.py`, hence, are the number of choices and the high and low payoffs of lottery L^A and lottery L^B, respectively, which will dynamically create the multiple price list.

A persistent phenomenon in multiple price list experiments is the observation of choices contradicting the axioms of Expected Utility Theory. In particular, it has been reported frequently that a considerable share of subjects alternate between "Option A" and "Option B" more than once (e.g. 13.2% in Holt and Laury (2002), 12.9% in Eckel and Wilson (2004), or 22.0% in Deck et al. (2014)). Observing multiple switching behavior apparently violates monotonicity and transitivity of revealed preferences and, in turn, contravenes the paradigm of utility maximization. Bruner (2011) suggests that multiple switching behavior is, in large part, due to a lack of salience. A technical remedy to rule out multiple switch points, introduced by Andersen et al. (2006) and employed by Jacobson and Petrie (2009) and Tanaka et al. (2010), among others, is to enforce a single switching point in the menu of binary choices. If `enforce_consistency = True`, subjects are enforced to answer the choice list without preference reversals. In particular, this implies that all options "A" above a selected "Option A" and all options "B" below a selected "Option B" are automatically checked, imposing strict monotonicity and enforcing transitivity of revealed preferences.

Another recurring problem associated with multiple choice list methods is the tendency of subjects to anchor towards the middle of the list, referred to as "compromise effect" (see, e.g., Harrison and Ruström, 2008; Beauchamp et al., 2015). In order to mitigate potential framing and anchoring effects, the MPL app includes two remedies to be used separately or in combination. (i) While multiple price lists are commonly presented in tabular form on a single page, the menu of binary choices might be displayed one-by-one on separate screens (as in Hey and Orme (1994) or Drichoutis and Lusk (2016), for instance). The boolean variable `one_choice_per_page` determines whether the menu of binary choices is displayed as a list on a single screen or sequentially

[3] The Holt and Laury (2002) mechanism typically displays probabilities of lottery outcomes as fractions. For sake of generalizability, the file `config.py` comprises a boolean variable (`percentage`) to determine whether to display probabilities of lottery outcomes in terms of fractions or percentage numbers with two decimal places.

[4] The choice list introduced by Holt and Laury (2002) includes a choice for $i = n$, i.e. a comparison between lotteries $(a_h, 1;\ a_l)$ and $(b_h, 1;\ b_l)$; `config.py` includes a variable to determine whether or not to include this choice. If `certain_choice = False`, the choice between sure payoffs will not be rendered such that the list only includes $n - 1$ binary choices; the probabilities of the high and low outcomes, however, are still determined by n, not $n - 1$.

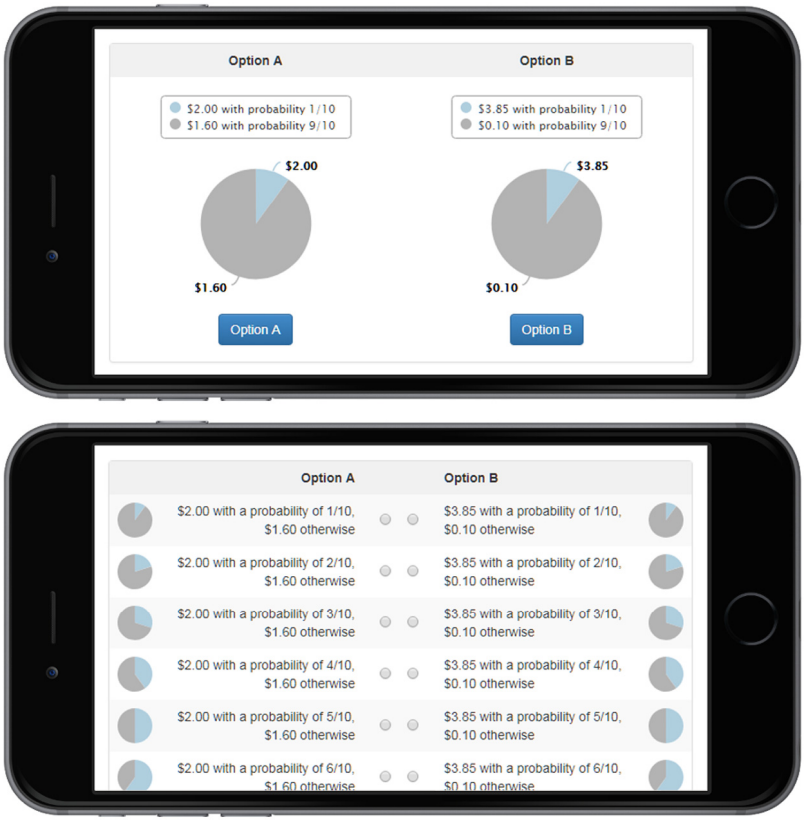

Fig. 1. MCL app simulated on an iPhone 6 (375 × 667px) in landscape mode. Multiple price list as proposed by Holt and Laury (2002) with choices being displayed sequentially on separate screens (top panel) and in tabular form (bottom panel) and pie charts emphasizing probabilities of high and low lottery outcomes.

on consecutive pages.[5] Whenever choices are displayed back-to-back, the boolean variable `progress_bar` allows for displaying a progress bar on the top of each screen, graphically highlighting the advance within the task. (ii) While the menu of choices is typically presented in ascending order, the MPL app allows for randomization of the choices, irrespective of whether choices are displayed sequentially or in tabular form. If `random_order = False`, binary choices are displayed in ascending order of the probability of the high lottery outcome; if `random_order = True`, lotteries are displayed randomly.

In order to graphically emphasize the probabilities of high and low lottery outcomes, the MPL app enables the integration of colored pie charts with slices corresponding to probabilities of lottery outcomes as implemented by Harbaugh et al. (2002), Habib et al. (2015), or Drichoutis and Lusk (2016), for instance. If `small_pies = True`, small pie charts (without payoff labels) are displayed to the left and to the right of the choice list table rows. If the menu of choices is shown sequentially (`one_choice_per_page = True`), the variable `large_pies` allows for rendering large pie charts with labels instead (see Fig. 1 for the two different implementations). If the variables `small_pies` and `large_pies` both take the value `False`, no graphical representation of probabilities will be displayed.

[5] While segregating choices might be an efficient way to attenuate compromise effects, several studies found that subjects make fewer inconsistent choices when lotteries are presented all at once rather then sequentially. It is argued that decisions made together are perceived as a "bundle" and, thus, are reconciled with each other while sequential choices are considered separately, implying independent chances of mistakes and errors. See, e.g., Langer and Weber (2001) and Chakravarti et al., (2002).

2.2. Certainty equivalence method (CEM)

Commonly referred to as certainty equivalent method, Cohen et al. (1987) introduced a multiple choice list, requiring subjects to compare a fixed lottery $(a_h, p;\ a_l)$ with varying degenerated lotteries yielding a payoff for sure, i.e. $(b_i, 1)$ for $i = 1, 2, \ldots, n$. In a similar vein, Dohmen et al. (2010) and Abdellaoui et al. (2011), among others, revitalized the methodology of eliciting the point of indifference between a risky lottery and certain payoffs. As an alternative, Bruner (2009) introduced two modified versions of the certainty equivalent method: rather than holding the lottery constant, the certain payoff is fixed while either (i) the high lottery outcome or (ii) the corresponding probabilities are varied. Formally, the methodologies suggested by Bruner (2009) imply binary choices between the constant, degenerated lottery $(b, 1)$ and (i) the lotteries $(a_{h,i}, p;\ a_l)$ or (ii) the lotteries $(a_h, p_i;\ a_l)$, for $i = 1, 2, \ldots, n$, respectively. Gächter et al. (2010) proposed a simplified version of the certainty equivalent method varying the low lottery outcome to measure and assess individual-level loss aversion; in particular, subjects are asked whether they prefer to accept a lottery $(a_h, p;\ a_{l,i})$ with $a_{l,i} < 0 < a_h$ or reject it and receive zero for all $i = 1, 2, \ldots, n$. The CEM app facilitates the implementation of any of the four possible variations. Depending on whether the variable `variation` in `config.py` is set to `'sure_payoff'`, `'probability'`, `'payoff_hi'`, or `'payoff_lo'` (further explained below), the choice list varies in the respective domain while the remaining parameters are held constant across all choices in the list.

Although the app is generalized to comprise different methodologies, the configuration of any of the four methods implies the specification of the same parameters: the variables `lottery_hi` and `lottery_lo` determine the high and low lottery payoffs,

respectively; `probability` defines the likelihood of the high lottery outcome; and `sure_payoff` specifies the certain amount offered as the alternative to the risky prospect. The variable `step_size` constitutes the increment in terms of the varying parameter while the three other ones are held constant over all binary choices. The variable corresponding to the value of the variable `variation` in labeling defines the value of the varying parameter in choice $i = 1$. For example, if `variation = 'sure_payoff'`, the values of the variables `lottery_hi`, `lottery_lo`, and `probability` are constant for all choices in the list; the sure amount offered in choice $i = 1$ is determined by the value of the variable `sure_payoff`; in choice $i = 2$, the sure amount equals `sure_payoff + step_size`; and so forth.

Basically, the certainty equivalent method resembles a multiple price list with one of the lotteries being replaced by degenerated lotteries yielding an outcome with a probability of one. Accordingly, most of the variables predefined in `config.py` of the CEM app are the same as in the CEM app. That is, choices can be rendered as a list in tabular form or sequentially on separate screens (`one_choice_per_page`), in ascending order of the varied parameter or randomized (`random_order`), and consistency of choices can be enforced by precluding multiple switching behavior (`enforce_consistency`). In addition to these configurations, the variable `accept_reject` allows for degenerating the decision problem to the question whether subjects prefer to accept or reject the lotteries. Thus, provided that the low payoffs of the lottery are set to be negative and `variation = 'payoff_lo'`, `accept_reject = True` implies that the app resembles the methodology proposed by Gächter et al. (2010) to elicit individual-level attitudes towards losses.[6]

2.3. Iterative choice list, "Staircase" procedure (ICL)

The iterative multiple price list, introduced by Andersen et al. (2006) and recently employed in large-scale experiments by Falk et al. (2016a, b), is a modified version of the certainty equivalent method with a constant lottery and varying sure payoffs (Cohen et al., 1987; Abdellaoui et al., 2011; Dohmen et al., 2011). Subjects are asked to reveal their preference in n successive binary choices, where each choice set is determined conditional on the decision in the previous choice set.[7]

While the lottery $L^A = (a_h, p;\ a_l)$ is identical for all dichotomous choices $i = 1, 2, \ldots, n$, the degenerated lottery yielding a certain outcome, $L_i^B = (b_i, 1)$, varies for each choice, with $a^h > b_i > a_l$ for all $i = 1, 2, \ldots, n$. Given an outcome b_1, the degenerated lottery in choice $i = 2$ will either be $L_2^B = (b_1 + \Delta_2, 1)$ or $L_2^B = (b_1 - \Delta_2, 1)$, depending on whether L^A or L_i^B has been revealed to be preferred ($\succ$) in choice $i = 1$. In choice $i = 3$, the sure payoff is given by $L_3^B = (b_2 + \Delta_3, 1)$ or $L_3^B = (b_2 - \Delta_3, 1)$, depending on whether $L^A \succ L_2^B$ or $L_2^B \succ L^A$, respectively. The same procedure is carried forward for all binary choices up to $i = n$. In each binary choice $i > 1$, Δ_i is divided in half, i.e. $\Delta_i = 1/2 \cdot \Delta_{i-1}$, such that the subject's certainty equivalent for the risky lottery L^A is approximated more and more precisely by each choice added to the staircase method. By this means, the staircase methodology implies 2^n possible, unique switching points between the lottery and varying sure payoffs, rendering a refined measurement of individual risk preferences with a relatively small number of choices to be made. Moreover, by construction of the method, preferences revealed in the n choices are transitive and monotone in payoffs. The underlying mechanism is depicted in Fig. 2.

The parameterization of the iterative choice list ICL app is akin to the one of the CEM app: the lottery is determined by a high (`lottery_hi`) and a low (`lottery_lo`) payoff, respectively, as well as the probability of the high payoff (`probability`). The varying certain payoffs are identified in an iterative manner based on the certain payoff in the first round (`sure_payoff`), the initial increase/decrease in the second choice (`delta`), and the number of choices (`num_choices`). For example, if `sure_payoff = 10` and `delta = 5`, "Option B" in the first choice will be a certain amount of \$10.00; if a subjects reveals to prefer the lottery over the sure payoff of \$10.0, "Option B" will be a sure payoff of \$15.00 in the second choice; if she reveals to prefer the sure payoff of \$15.00 over the lottery in the second choice, "Option B" will be a sure payoff of \$12.50 in the third choice; and so forth.

The default type of input for each decision are radio buttons. However, to facilitate users' inputs, the boolean variable `buttons` allows for rendering buttons instead of radio selects, implying that no "Next" button is needed and only half the number of clicks is required to complete the task. As for the MPL and CEM app, the variable `progress_bar` allows for displaying a graphical progress indicator at the top of each screen. In addition, the app allows for adding a choice to indicate indifference between "Option A" and "Option B", as, for instance, suggested by Andersen et al. (2006). If `indifference = False`, subjects can only reveal their preference for either "Option A" or "Option B"; if `indifference = True`, an additional option is available to indicate indifference between the lottery and the sure payoff. Whenever a subject chooses "Indifferent", the iteration procedure stops (i.e. all subsequent choices are automatically skipped) as indifference has been reached. If an "Indifferent" choice is drawn for payoff, it is randomly determined whether "Option A" or "Option B" constitutes the payment from the task.

2.4. Single choice list task (SCL)

The SCL app facilitates implementing the single choice list method as proposed by Binswanger (1980) and Eckel and Grossman (2002). This simple risk preference elicitation procedure offers subjects a menu of different lotteries, asking them to choose the one they prefer to be played.[8] The n lotteries in the list are defined as $L_i = (c + (i-1)\Delta^+, p; c - (i-1)\Delta^-)$ with $\Delta^+ > \Delta^- > 0$ for $i = 1, 2, \ldots, n$. Thus, the first lottery pays c for sure while the low and high outcomes in the subsequent lotteries increase and decrease by some constant increments Δ^+ and Δ^-, respectively. The probability that the high lottery outcome is drawn, p, is constant for all lotteries in the choice list.

Similar to the multiple choice list apps (MPL, CEM, and ICL), the lotteries in the single choice list are dynamically constructed based on the parameters of the first prospect and the increments Δ^+ and Δ^-. The variable `sure_payoff` defines the sure payoff in the first lottery and constitutes the "starting point" of all subsequent lotteries; `delta_hi` and `delta_lo` determine the increments Δ^+ and Δ^-, respectively. That is, the first lottery in the list will be a sure payment; the second one will either pay (`sure_payoff + delta_hi`) or (`sure_payoff - delta_lo`), the third one (`sure_payoff + 2 * delta_hi`) or (`sure_payoff - 2 * delta_lo`); and so forth.

[6] To prevent that subjects' payoffs turn negative in case of negative lottery outcomes being drawn for payment, the variable `endowment` allows for specifying an additional endowment to capture potential losses within the CEM task.

[7] Falk et al. (2016a, b) utilize the staircase procedure not only for eliciting attitudes towards risks but also for measuring individual-level time preferences. Similarly, the staircase method has been employed by Dimmock et al. (2016) to assess ambiguity aversion. By altering the model (`models.py`), the basic structure of the ICL app can easily be modified to elicit patience and ambiguity preferences too.

[8] As the methodology only requires subjects to make a single choice, preferences revealed in the task will always be consistent. Since a list of different prospects is presented all at once on a single page, however, lotteries can be randomized in display to countervail potential compromise effects with `random_order = True` (see, for instance, Harrison and Rutström, 2008; Beauchamp et al., 2015).

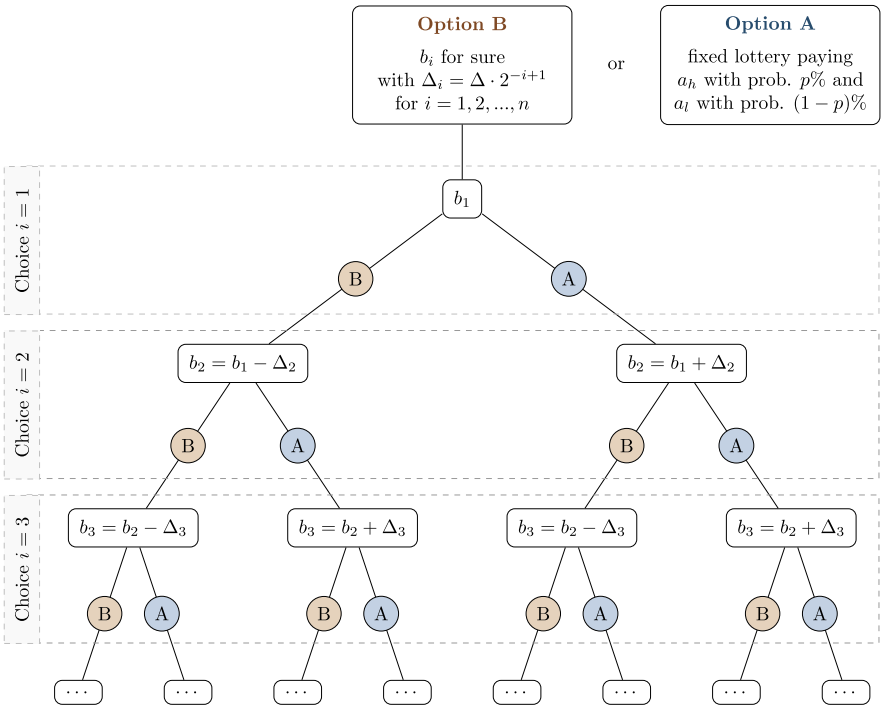

Fig. 2. Mechanism of the 'staircase' risk preference elicitation procedure. Subjects face n consecutive decisions between a fixed lottery ("Option A") and a varying sure payoff ("Option B"). The sure payoff for each choice $i = 1, 2, \ldots, n$ depends on the revealed preference in choice $i - 1$: whenever the lottery (sure payoff) has been revealed to be preferred over the sure payoff (lottery) in choice i, the sure payoff in choice $i + 1$ increases (decreases) by $\Delta_i = 1/2 \cdot \Delta_{i-1}$.

Note that the procedure outlined above implies that both the expected payoff and the standard deviation, i.e. the riskiness, of the lottery increases in n. This implies that the method proposed by Eckel and Grossman (2002) does not allow to discriminate between risk neutral and risk loving preferences as any subject with either of the two preference types will choose the last lottery in the list; a risk neutral subject will do so to maximize the expected payoff, a risk seeking subject to maximize the standard deviation. To distinguish risk neutral from risk seeking preferences, the last lottery in the list is modified in such a way that its expected payoff equals the expected payoff of the next-to-last lottery but with a higher standard deviation if `risk_loving = True`. In this vein, a risk neutral subject will choose the second to last lottery while a risk loving subject will opt for the last one.

2.5. Variables and features common to all apps

Except for the single choice list method (SCL), in which the lottery that has been revealed to be preferred is payoff-relevant by construction of the task, the apps apply a random lottery incentive procedure to avoid potential "wealth or portfolio effects" (see Cubitt et al., 1998; Harrison and Rustrom, 2008, for instance). As theoretically proven by Azrieli et al. (2012), choosing one out of several decision problems at random is the only incentive compatible mechanism assuming statewise monotonicity of revealed preferences. Accordingly, one of the binary choices is randomly picked (with equal probability) at the end of the task and played out according to the subject's decision.

In addition to the task-specific configurations outlined for each app, two boolean variables (`instructions` and `results`) are available for all apps to determine whether or not to display a separate HTML-template for instructions and the results, respectively. Instructions included in the HTML-files in the apps' template directories should only serve as examples. They do only refer to the default settings in `config.py` and need to be adjusted to different configurations. For each of the apps, the result screen resembles the decision a subject made in the choice which has been randomly drawn for payment and explains how the final payoff is derived. By this means, the payment resulting from any of the tasks is as transparent as possible, considered to enhance salience of incentives.

3. Data output (variables stored)

Each app stores all relevant information, with respect to both subjects' decisions as well as payments in the database. That is, for each choice $i = 1, 2, \ldots, n$ in either of the multiple price lists, a model field `choice_i` takes the value `'A'` or `'B'`, corresponding to the labeling of lotteries or options in each particular task. In addition, all multiple choice list apps (MPL, CEM, and ICL) determine whether the choice list has been answered consistently. The variable `inconsistent` takes value `1` whenever a subject alternates between "Option A" and "Option B" more then once and `0` otherwise. Given that preferences revealed in the choice list are consistent, i.e. if `inconsistent = 0`, the switching row for each participant is determined. That is, the binary choice in which a subject opts for "Option B" the first time constitutes the value of the variable `switching_row` stored in the database. As there is just one choice associated with the single choice list method, only the index of the preferred choice is stored in the model field `choice`. With respect to payment information, each multiple choice list module stores the label of the choice that has been randomly picked for payoff (`choice_to_pay`), a subject's choice in the payoff-relevant decision (`option_to_pay`), as well as the random draw determining which lottery outcome constitutes the payment (`random_draw`). In addition to the variables stored in the database, all relevant information is stored as "globals" in *oTree*'s `participant.vars['label']` in order to allow for accessing any data at any time within a session after the task has been completed. That is, when eliciting risk preferences at the beginning of an experimental session, payoff information can easily be displayed after completing other tasks at the end of the experiment, in order not to distort induced preferences. For information on variable labels, refer to the `models.py` file of the respective app.

Acknowledgments

I would like to thank Christoph Huber, Armin Pfurtscheller, Michael Razen, Julia Rose, and Matthias Stefan as well as participants at the Experimental Finance Conference 2016 in Mannheim and the Research in Behavioral Finance Conference 2016 in Amsterdam for helpful comments on the software applications, the documentation, as well as the manuscript. Financial support from the Austrian Science Fund FWF (START-grant Y617-G11 and SFB F63) is gratefully acknowledged.

References

Abdellaoui, M., Driouchi, A., L'Haridon, O., 2011. Risk aversion elicitation: Reconciling tractability and bias minimization. Theory Decis. 71, 63–80.
Andersen, S., Harrison, G.W., Lau, M.I., Rutström, E.E., 2006. Elicitation using multiple price list formats. Exp. Econ. 9, 383–405.
Arrow, K.J., 1965. Aspects of the theory of risk bearing. Yrjö Jahnssonin Säätiö, Helsinki.
Azrieli, Y., Chambers, C.P., Healy, P.J., 2012. Incentives in Experiments: A Theoretical Analysis, Working Paper.
Beauchamp, J.P., Benjamin, D.J., Chabris, C.F., Laibson, D.I., 2015. Controlling for the Compromise Effect Debiases Estimates of Risk Preference Parameters, Working Paper. National Bureau of Economic Research, p. 21792.
Becker, G.M., DeGroot, M.H., Marshak, J., 1964. Measuring utility by a single-response sequential method. Behav. Sci. 9 (3), 226–232.
Binswanger, H.P., 1980. Attitudes toward risk: Experimental measurement in rural india. Am. J. Agric. Econ. 62 (3), 395–407.
Bruner, D.M., 2009. Changing the probability versus changing the reward. Exp. Econ. 12 (4), 367–385.
Bruner, D.M., 2011. Multiple switching behavior in multiple price lists. Appl. Econ. Lett. 18 (5), 417–420.
Chakravarti, D., Krish, R., Paul, P., Srivastava, J., 2002. Partitioned presentation of multicomponent bundle prices: Evaluation, choice and underlying processing effects. J. Consum. Psychol. 12 (3), 215–229.
Charness, G., Gneezy, U., 2012. Portfolio choice and risk attitudes: An exeriment, J. Econ. Behav. Organ. 83, 50–58.
Charness, G., Gneezy, U., Imas, A., 2013. Experimental methods: eliciting risk preferences. J. Econ. Behav. Organ. 87, 43–51.
Chen, D.L., Schonger, M., Wickens, C., 2016. oTree—An open-source platform for laboratory, online, and field experiments. J. Behav. Exp. Finance 9, 88–97.
Cohen, M., Jaffray, J.-Y., Said, A., 1987. Experimental comparison of individual behavior under risk and under uncertainty for gains and for losses. Organ. Behav. Hum. Decis. Process. 39, 1–22.
Crosetto, P., Filippin, A., 2013. The "bomb" risk elicitation task. J. Risk Uncertain. 47, 31–65.
Crosetto, P., Filippin, A., 2015. A theoretical and experimental appraisal of four risk elicitation methods. Exp. Econ. 18 (6), 1–29.
Csermely, T., Rabas, A., 2016. How to reveal people's preferences: Comparing time consistency and predictive power of multiple price list risk elicitation methods. J. Risk Uncertain. 53 (2), 107–136.
Cubitt, R.P., Starmer, C., Sugden, R., 1998. On the validity of the random lottery incentive system. Exp. Econ. 1, 115–131.
Deck, C., Jungmin, L., Reyes, J., Rosen, C., 2014. Investing versus gambling: experimental evidence of multi-domain risk attitudes. Appl. Econ. Lett. 21 (1), 19–23.
Dimmock, S.G., Kouwenberg, R., Mitchell, O.S., Peijnenburg, K., 2016. Ambiguity aversion and household portfolio choice puzzles: Empirical evidence. J. Financ. Econ. 119, 559–577.
Dohmen, T., Falk, A., Huffman, D., Sunde, U., 2010. Are risk aversion and impatience related to cognitive ability? Am. Econ. Rev. 100 (3), 1238–1260.
Dohmen, T., Falk, A., Huffman, D., Sunde, U., Schupp, J., Wagner, G.G., 2011. Individual risk attitudes: Measurement, determinants, and behavioral consequences. J. Eur. Econ. Assoc. 9 (3), 522–550.
Drichoutis, A.C., Lusk, J.L., 2016. What can multiple price lists really tell us about risk preferences?. J. Risk Uncertain. 53 (2), 89–106.
Eckel, C.C., Grossman, P.J., 2002. Sex differences and statistical stereotyping in attitudes toward financial risk. Evol. Hum. Behav. 23, 281–295.
Eckel, C.C., Wilson, R.K., 2004. Is trust a risky decision? J. Econ. Behav. Organ. 55 (4), 447–465.
Falk, A., Becker, A., Dohmen, T., Enke, T., Huffman, D., Sunde, U., 2016a. The Nature and Predictive Power of Preferences: Global Evidence, Human Capital and Economic Opportunity Global Working Group, Working Paper 2016-004.
Falk, A., Becker, A., Dohmen, T., Huffman, D., Sunde, U., 2016b. The Preference Survey Module: A Validated Instrument for Measuring Risk Time, and Social Preferences, Working Paper.
Freeman, D., Halevy, Y., Kneeland, T., 2017. Eliciting Risk Preferences Using Choice Lists, Working Paper.
Gächter, S., Johnson, E.J., Herrmann, A., 2010. Individual-Level Loss Aversion in Riskless and Risky Choices, Centre for Decision Research and Experimental Economics, Discussion Paper No. 2010-20.
Gneezy, U., Potters, J., 1997. An experiment on risk taking and evaluation periods. Q. J. Econ. 112 (2), 631–645.
Habib, S., Friedman, D., Crockett, S., James, D., 2015. Eliciting Risk Preferences: Text vs. Graphical Multiple Price Lists, WZB Discussion Paper, No. SP II 2015-501.
Harbaugh, W.T., Krause, K., Vesterlund, L., 2002. Risk attitudes of children and adults: Choices over small and large probability gains and losses. Exp. Econ. 5, 53–84.
Harrison, G.W., Ruström, E., 2008. Risk aversion in the laboratory. In: Research in Experimental Economics 12. Emerald, Bingley, UK, pp. 41–196.
Hey, J.D., Orme, C., 1994. Investigating generalizations of expected utility theory using experimental data. Econometrica 62 (6), 1291–1326.
Holt, C.A., Laury, S.K., 2002. Risk aversion and incentive effects. Am. Econ. Rev. 92 (5), 1644–1655.
Holzmeister, F., Pfurtscheller, A., 2016. oTree: The bomb risk elicitation task, J. Behav. Exp. Financ 10, 105–108.
Jacobson, S., Petrie, R., 2009. Learning from mistakes: What do inconsistent choices over risk tell us? J. Risk Uncertain. 38 (2), 143–158.
Kachelmeier, S.J., Shehata, M., 1992. Examining risk preferences under high monetary incentives: experimental evidence from the people's republic of china. Am. Econ. Rev. 82 (5), 1120–1141.
Langer, T., Weber, M., 2001. Prospect theory, mental accounting, and differences in aggregated and segregated evaluation of lottery portfolios. Manag. Sci. 47 (5), 716–733.
Lejuez, C.W., Read, J.P., Kahler, C.W., Richards, J.B., Ramsey, S.E., Stuart, G.L., Strong, D.R., Brown, R.A., 2002. Evaluation of a behavioral measure of risk taking: The balloon analogue risk task (bart). J. Exp. Psychol.: Appl. 8 (2), 75–84.
Post, T., van den Assem, M.J., Baltussen, G., Thaler, R.H., 2008. Deal or no deal? decision making under risk in a large-payoff game show. Am. Econ. Rev. 98 (1), 38–71.
Pratt, J.W., 1964. Risk aversion in the small and in the large. Econometrica 32 (1), 122–136.
Tanaka, T., Camerer, C.F., Nguyen, Q., 2010. Risk and time preferences: Linking experimental and household survey data from bietnam. Am. Econ. Rev. 100 (1), 557–571.
Weber, E.U., Blais, A.-R., Betz, N.E., 2002. A domain-specific risk-attitude scale: Measuring risk perceptions and risk behaviors. J. Behav. Decis. Mak. 15, 263–290.