Journal of Behavioral and Experimental Finance 23 (2019) 46–53

Contents lists available at ScienceDirect

Journal of Behavioral and Experimental Finance

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

A flexible z-Tree and oTree implementation of the Social Value Orientation Slider Measure

Paolo Crosetto [a], Ori Weisel [b], Fabian Winter [c,*]

[a] INRA-GAEL, Institut National pour la Recherche Agronomique, Grenoble Applied Economics Laboratory, 1241 Rue des Résidences Domaine Universitaire, 38400 Saint Martin d'Hères, France
[b] Coller School of Management, Organizational Behavior Department, Tel Aviv University, Tel Aviv, 6997801, Israel
[c] Max Planck Institute for Research on Collective Goods, MPRG "Mechanisms of Normative Change", Kurt Schumacher Strasse 10, 53113 Bonn, Germany

ARTICLE INFO

Article history:
Received 26 November 2018
Received in revised form 5 April 2019
Accepted 8 April 2019
Available online 7 May 2019

JEL classification:
C91
D03
D64

Keywords:
z-tree
oTree
Experimental software
Social value orientation
Slider measure

ABSTRACT

This manual describes z-Tree (fischbacher 2007) and oTree (chen et al. 2016) implementations of the paper-based Social Value Orientation Slider Measure (SVOSM) by murphy et al. (2011). The z-Tree implementation includes both discrete and continuous (slider-based) versions of the SVOSM; the oTree implementation uses a slider-based version of the SVOSM.

This manual describes implementations of the SVO Slider Measure (Murphy et al., 2011, SVOSM hereafter) in z-Tree (Fischbacher, 2007) and oTree (Chen et al., 2016), two of the most commonly used software packages for computerized experiments in the social sciences. The SVOSM is designed to measure social preferences on a continuous scale, and is frequently used throughout the social sciences literature (e.g., Ackermann et al., 2016; Fiedler et al., 2013; Pletzer et al., 2018; Soraperra et al., 2019; Weisel and Zultan, 2016; Winter, 2014). The SVOSM measures social value orientation via a series of either 6 (only the *primary items*) or 15 (*primary* and *secondary items*) generalized dictator games, which vary in the conversion rates between money allocated to the decision maker and the recipient. The decisions on the six primary items can be summarized in a single index by calculating the ratio of the average allocations to the other person to the average allocations to oneself

$$SVO^\circ = \arctan\left(\frac{\texttt{mean\_to\_other} - 50 * \texttt{scale}}{\texttt{mean\_to\_self} - 50 * \texttt{scale}}\right) \quad (1)$$

Fig. 1 shows a geometric representation of the SVOSM and plots the respective allocations on the six sliders together with the resulting average allocation and the SVO°. High values of SVO° correspond to a more prosocial or altruistic social value orientation, while low or negative values correspond to individualistic or competitive preferences. The resulting SVO° can be translated into a simple social preference utility function, which can be used in economic models. This utility function could take the following form

$$U(\pi_S, \pi_O) = \pi_S + \alpha_i \pi_O, \quad (2)$$

where $\alpha_i = \tan(SVO^\circ)$. The parameter α_i represents the weight the decision maker attaches to the payoff for another person, π_S is the payoff for the self (i.e. for the decision maker), and π_O is the payoff for the other person.

The nine secondary items can be used to further disentangle efficiency motives from equality motives (see Murphy et al., 2011, for a more detailed discussion of the secondary items).

1. General remarks

Murphy et al. (2011) present two different versions of the SVOSM: a quasi-continuous web-based measure using sliders,

* Corresponding author.
E-mail address: winter@coll.mpg.de (F. Winter).

https://doi.org/10.1016/j.jbef.2019.04.003
2214-6350/

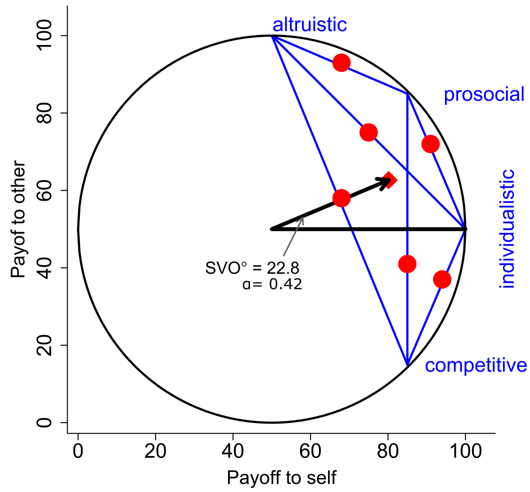

Fig. 1. Graphical representation of the six item SVO-Slider Measure. Each thin blue line shows the possible allocations from one ideal-typical decision end point to one other end point (e.g., between altruistic and competitive payoff allocations). Red circles show the six allocation decisions from one exemplary respondent. The red diamond shows the average of these six payoff allocations to the respondent herself and to the other. The angle between the thick black horizontal line and the thick black arrow corresponds to this respondent's *SVO*° of 22.8° and an α-value of 0.42 . (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

and a discrete paper-based version capturing the important features of the web-based version, which some researchers find easier to administer. Our z-Tree implementation provides both a discrete and a continuous version; the oTree implementation provides only the continuous version.

Our implementations are further customizable to allow the user to select the matching protocol, language, payoff scaling, the order of the items and whether to present only the primary items or the full measure.

1.1. Matching

A major addition to the original publication is that we add a `RANDOM_DICTATOR` matching protocol to the `RING` matching protocol used by Murphy et al. (2011). Assuming four decision makers, A, B, C, and D, in the `RING` protocol A gives to B, B gives to C, C gives to D and D gives to A (Fig. 2, left). Each subject is both a sender and a receiver, but in each role is matched with a different person. The `RING` matching works with any number of subjects *greater than 1*.

In the `RANDOM_DICTATOR` matching (Fig. 2, right), subjects A, B, C, D are matched in groups of 2 (say (A; B) and (C; D)). Each subject submits his/her choices in the SVOSM. Later, one member of each group (say A and C) is randomly selected to be the sender, and the other (B and D, respectively) is the receiver. In this case, B receives from A and D receives from C. The choices by B and D are recorded but not implemented. `RANDOM_DICTATOR` matching thus limits the interaction to two subjects, where only one choice is finally implemented. This may reduce chains of indirect reciprocity, where A gives to B because she expects to be compensated by D. The `RANDOM_DICTATOR` matching works with any *even* number of subjects.

The `RANDOM_DICTATOR` matching uses the strategy method; decisions are only implemented with probability 1/2. It has been argued that such decision making may be perceived as "colder", and thus may lead to more strategic (Brandts and Charness, 2000) or normative choices (Rauhut and Winter, 2010).

1.2. Main measures

The *SVO*° is recorded in the variable `svo_angle`. It is the core measure of the SVOSM. It is calculated from the six primary items as in Eq. (1) (see Murphy et al., 2011, for a detailed discussion).

The nine secondary items are used to calculate the `inequality_aversion_score`, which distinguishes, for subjects classified as prosocial according to the `svo_angle`, between efficiency maximization and material equality motives. The reason for this extra dimension is that both motives are maximized by the same choices in the primary items of the SVOSM (i.e. always giving the same share to both players.). For additional information, see Murphy et al. (2011). It is calculated as

$$\texttt{inequality\_aversion\_score} = \frac{\texttt{avg\_dist\_to\_equality}}{(\texttt{avg\_dist\_to\_equality} + \texttt{avg\_dist\_to\_joint})}.$$

The parameter `alpha` represents the weight the subject assigns to the outcome of the other participant and is calculated as

$$\texttt{alpha} = \tan(\texttt{svo\_angle}).$$

Finally, the applications also discretizes the resulting *SVO*° to a corresponding `svo_type`[1]:

$$\texttt{svo\_type} = \begin{cases} \text{Altruist} & \text{if} \quad \texttt{svo\_angle} > 57.15^\circ \\ \text{Prosocial} & \text{if} \quad 22.45^\circ > \texttt{svo\_angle} \geq 57.15^\circ \\ \text{Individualistic} & \text{if} \quad -12.04^\circ > \texttt{svo\_angle} \geq 22.45^\circ \\ \text{Competitive} & \text{if} \quad -12.04^\circ > \texttt{svo\_angle} \end{cases}$$

2. z-Tree implementation

The z-Tree implementation of the SVOSM is a stand-alone treatment which is easy to integrate in existing z-Tree treatments. There are two versions, created in z-Tree 3.6.7 and in z-Tree 4.1.7. Both can be used as any other z-Tree treatment.

The z-Tree version of the slider has an option to choose between a slider-based and a radio line-based implementation. The radio-line implementation of the SVOSM provides all the features of the paper-based version of the SVOSM, except for interior decisions (i.e., choices which are in between the discrete values on the scale). Note that while such interior decisions are possible in principle, they are very rarely used in practice.[2] The slider-based implementation is implemented using plot boxes and is close to the web-based version of the original slider measure. Fig. 3 displays screenshots of the two implementations. Previous implementations of the slider-based version had performance issues when too many participants moved the sliders at the same time. This was due to computations being carried out on the server side each time a subject moved a slider. To avoid these issues, we moved computations to the client side. The implementation has been tested with 16 participants without any noticeable delays. As a cautionary measure, we recommend testing the software in advance before running sessions with much larger groups.

2.1. Parameters

This section describes the parameters of the z-Tree treatment which can be changed by the experimenter to match specific needs. Standard parameters, e.g. number of subjects, are treated in the standard z-Tree way. It is not necessary nor recommended

1 The thresholds are derived by bisecting the respective adjacent ranges of the idealized types. See Murphy et al. (2011) for a detailed discussion.

2 Personal communication with R. Murphy.

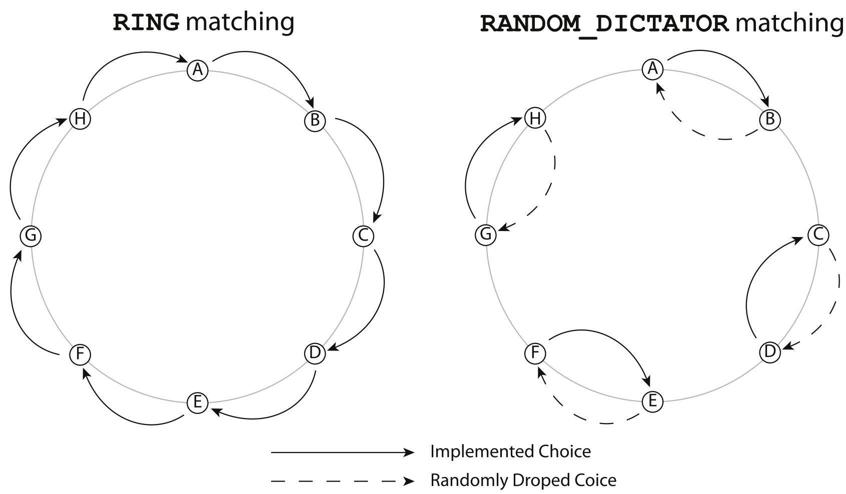

Fig. 2. Matching protocols.

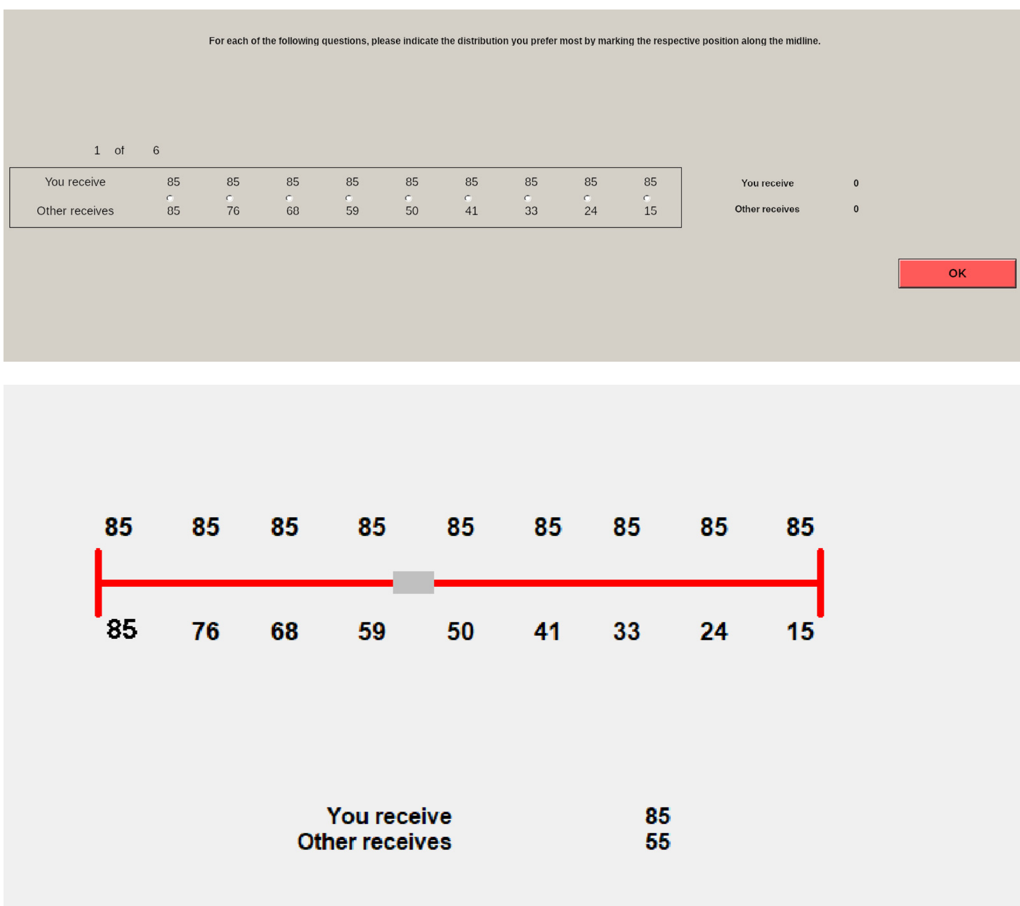

Fig. 3. Screenshot of the z-Tree decision screen using radio buttons (top) and a slider (bottom).

to change the number of rounds or groups. To change treatment specific parameters, open the globals program "////INSERT PARAMETERS HERE ////" (see Fig. 4) and change the respective values described below. Table A.1 in the Appendix gives an overview of the important parameters in the z-Tree implementation.

2.1.1. `slider_type`

Sets the elicitation mode to either continuous slider (`REAL_SLIDER`) or discrete radio line (`CHECK_BOXES`).

2.1.2. `language`

Sets the language in which the SVOSM is displayed to the decision maker. Implemented languages are English, French,

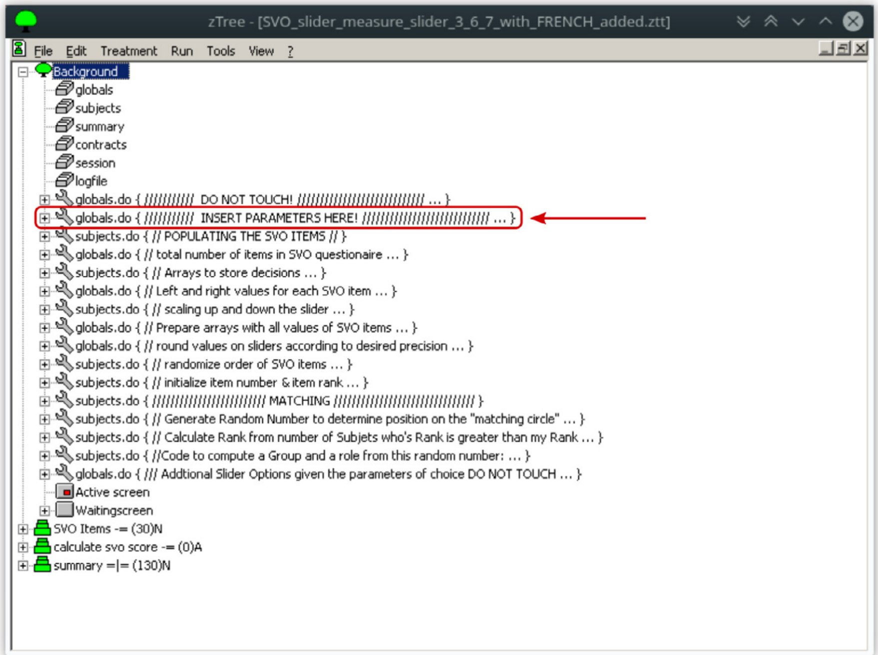

Fig. 4. Screenshot of the program indicating where to change setup parameters.

German, and Italian. To add additional languages, first declare the new language in the first program (DO NOT TOUCH), add the translations to all items, and finally choose the language in the program INSERT PARAMETERS HERE.

2.1.3. `select_items`

Allows you to choose whether only the six primary items (`PRIMARY`) or all fifteen items (six primary and nine secondary, `FULL`) are displayed. It is not possible to use only the secondary items.

2.1.4. `items_in_random_order`

Determines whether items are displayed in the order presented in the paper-based SVOSM (`ORDERED`) as in Murphy et al. (2011), or in random order (`RANDOM`). If the `RANDOM` option is chosen, the order is randomized separately for each subject.

2.1.5. `matching`

Determines the matching procedure: `RING` or `RANDOM_DICTATOR` (see Section 1.1 and Fig. 2). `RING` matching works with any number of subjects greater than 1. `RANDOM_DICTATOR` matching works with any even number of subjects.

2.1.6. `scale`

The scale of the circle underlying the SVOSM can be changed. The default is `scale` = 1, which results in a circle with a diameter of 100 centered at (50,50), as in the original publication. Inputs greater than 1 scale the circle up (e.g. 2 results in a circle with a diameter of 200 centered at (100,100)); inputs smaller than 1 scale the circle down (e.g. 0.5 results in a circle with a diameter of 50 centered at (25,25)). Scaling may be useful if you want to present the decisions in real monetary values and cannot afford to pay 100 €/$/CHF/. . .

2.1.7. `precision`

Determines the precision of the displayed values. This option is limited to two modes – integers and two decimal digits – because z-Tree does not allow for conditional layout of decimals. `INTEGERS` displays values as integers, which is sensible for larger numbers, e.g. a circle with a diameter of 100. For a circle with a small diameter, two decimals after the point may be more appropriate (choose option `TWO_DIGITS_AFTER_POINT`).

2.1.8. `debug`

Displays debugging information, including a kill-button. Useful for testing the treatment. Set to 0 to suppress all debugging information.

2.2. Output

The z-Tree treatment writes all the relevant output in the subjects table and calculates the important measures straight away, making it possible to readily use the results in the experiment (e.g. for matching purposes) and making data analysis more convenient, since all important measures are already in the data set. All the relevant variables in the subjects table of the z-Tree implementation are explained in Table A.2. Some of the output variables are only relevant under specific parameters, e.g. `avg_dist_to_equality` is calculated only if the secondary items are used. The rightmost column of Table A.2 indicates if and when the variables are used.

2.2.1. `Profit`

The `Profit`-variable is built-in in z-Tree, and is automatically written to the `TotalProfit`-variable in the session table. Thus, you can use the profits earned in the SVOSM in later treatments. To determine the profits, one of the sliders is randomly selected and the corresponding earnings are written to the `Profit`-variable. If `RING` matching is used, `Profit` contains the sum of the amount received as a receiver and the amount kept as a sender. When the `RANDOM_DICTATOR` option is chosen, `Profit` depends on the subject's role: for *senders*, `Profit` returns the amount kept, and for *receivers*, it returns the amount sent by the matched senders.

2.2.2. *svo_angle*

The `svo_angle` is the core measure of the SVOSM. The `svo_angle` is stored in the subjects table *and* in the session table. Storing values in the session table is useful if some of the SVOSM information is required in other treatments later in the session.

2.2.3. *svo_type*

`svo_type` assigns SVO types to specific value ranges of the `svo_angle`. The following values are used: 1 = Altruist, 2 = Prosocial, 3 = Individualist, 4 = Competitive. Thresholds for assigning labels are taken from Murphy et al. (2011). `svo_type` is stored in the subjects table *and* in the session table.

2.2.4. *alpha*

Calculates the α-level as used in Eq. (2) for each subject. It is stored in the subjects table.

2.2.5. *inequality_aversion_score*

The `inequality_aversion_score` is calculated from the secondary items and is computed only if they are used. It is calculated only if the subject is classified as being "prosocial" (see Section 1.2 above). In all other cases it is set to −99. `inequality_aversion_score` is stored in the subjects table *and* in the session table .

3. oTree implementation

The oTree implementation of the SVOSM is implemented as a normal oTree-app (Chen et al., 2016) and can be easily integrated in larger experiments.[3] It is based on the continuous version of the SVOSM and has a somewhat more up-to-date layout than the z-Tree version. The interface includes a bar chart to show the allocations to each player, but is fully customizable using HTML and JavaScript. When implementing the SVOSM, we tried to use as much of the standard oTree-dialect as possible, and only rely on common JavaScript libraries compatible with oTree such as `HighCharts` or `jquery` when necessary. These libraries are either already included or referenced in the source code, so there is no need to install them manually. Important parts of the code are implemented in JavaScript, so a fair amount of knowledge may be required to make fundamental changes. Debugging for both the oTree and JavaScript parts is possible with the internal oTree debugger (see the oTree documentation). The JavaScript parts use cookies to temporarily store information in the client's browser and issues a notification the first time the page is opened (see Fig. 5).

3.1. Setting the parameters

All relevant parameters can be set either in the file `settings.py` or, more conveniently, directly in the oTree web interface when creating a new session. Simply click on "Sessions" in the top panel of the admin page and change the parameters described below as required. If you want to set specific defaults, change them in `settings.py`. Table A.3 gives an overview of the important parameters in the oTree implementation.

[3] The current version of the oTree app was implemented in oTree 2.1.9. Please update the app according to the documentation if a new version of oTree is released.

In most cases the options available to the user in the oTree implementation are the same as in the z-Tree one. In these cases we refer to the relevant sections above. Other cases are discussed below.

3.1.1. *LANGUAGE_CODE*

Sets the language in which the SVOSM is displayed to the decision maker. Implemented languages are English, German, Italian, and French. Set `LANGUAGE_CODE` to `'en'`, `'de'`, `'it'`, or `'fr'`. Please refer to the oTree documentation on localization to learn how to add further languages. `LANGUAGE_CODE` also sets the language for the cookie warning.

3.1.2. *select_items*

See Section 2.1.3.

3.1.3. *items_in_random_order*

See Section 2.1.4.

3.1.4. *matching*

See Section 2.1.5.

3.1.5. *scale*

See Section 2.1.6.

3.1.6. *precision*

See Section 2.1.6.

3.2. Output

All relevant output, including the *SVO*° etc., can be downloaded in CSV or XLS format in the standard oTree way via the web browser. The data is stored internally, ready to be used within the experiment (e.g. for matching purposes). Other than these differences, all details regarding the output are the same as in the z-Tree implementation (see 2.2). All the relevant output variables in the oTree implementation are explained in Table A.4.

3.2.1. *alpha*

See Section 2.2.4

3.2.2. *payoff*

The `payoff`-variable is the standard oTree variable to calculate payoffs. It is highly recommended to use this variable name, because it makes e.g. the summing of payoffs over different apps easier. To determine the profits, one of the sliders is randomly selected and the corresponding earnings are written to the `payoff`-variable. If `RING` matching is used, `payoff` contains the sum of the amount received as a receiver and the amount kept as a sender. When the `RANDOM_DICTATOR` option is chosen, `payoff` depends on the subjects role: for *senders*, `payoff` returns the amount kept, for *receivers*, it returns the amount sent the matched senders.

3.2.3. *svo_angle*

See Section 2.2.2.

3.2.4. *svo_type*

See Section 2.2.3.

3.2.5. *inequality_aversion_score*

See Section 2.2.5.

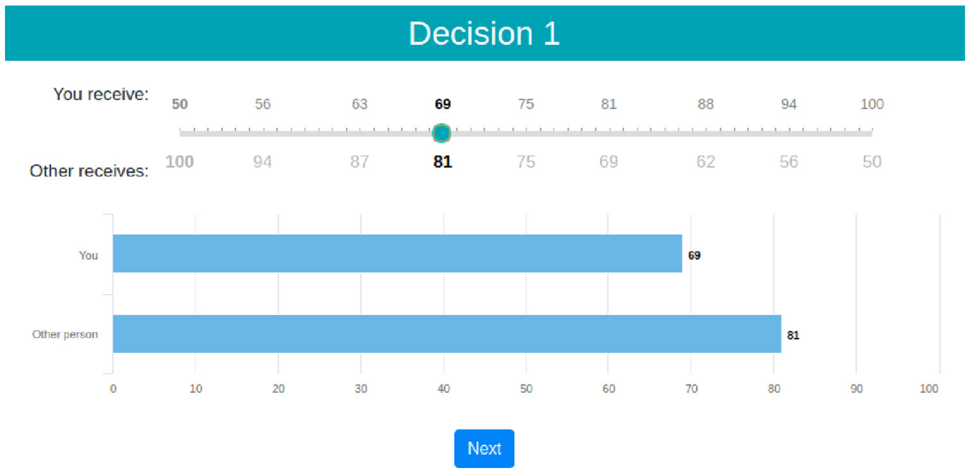

Fig. 5. Screenshot of the oTree decision screen.

4. Downloading the files

All implementations can be accessed via our github repository at https://github.com/drfwint/svo. Please feel free to submit potential bugs or pull-requests via github.

5. Disclaimer

The z-Tree treatment of the SVOSM is thoroughly tested and has been used in several labs and experiments over the years. The oTree implementation has also been tested thoroughly, but is much younger. As it has not been circulated widely so far, it should be used with somewhat greater care than the z-Tree version. Nevertheless, the authors disclaim all warranties, expressed or implied, regarding the Software, including any implied warranties of satisfactory quality, merchantability or fitness for a particular purpose. The authors shall have no liability whatsoever to the User of the Software for any direct, indirect, special or consequential loss and/or expense (including loss of profit) suffered by the User and arising out of a malfunctioning of the Software.

You can use, modify and distribute the corresponding treatments if you agree with the above points. If you use these implementations of the SVOSM, please cite the SVOSM as Murphy et al. (2011) and make sure to follow the license agreements associated with z-Tree or oTree (in particular to cite Fischbacher (2007), or Chen et al. (2016), respectively).

Acknowledgments

Programming assistance by Tina Hilbig, Abdul Al-Kattan, Emad Bahrami Rad and Tobias Werner is gratefully acknowledged. We thank Matthieu Pourieux the translation into French, and several users of the software for valuable feedback. Funding by the Deutsche Forschungsgemeinschaft (DFG) - Projekt number 395336584 is gratefully acknowledged.

Appendix

See Tables A.1–A.4.

Table A.1
Parameters in the z-Tree treatment.

Parameter	Values	Description
real_slider	REAL_SLIDER	SVOM is presented with sliders
	CHECK_BOXES	SVOM is presented with radio buttons
language	ENGLISH	Language is English
	GERMAN	Language is German
	ITALIAN	Language is Italian
	FRENCH	Language is French
select_items	PRIMARY	only the primary items (items 1–6) are elicited
	FULL	primary and secondary items (items 1–15) are elicited
items_in_random_order	ORDERED	items are presented according to the order in Murphy et al. (2011)
	RANDOM	items are presented in random order
matching	RING	Subject A, B, C, D are ordered on a ring-structure as in Murphy et al. (2011). In this case, A gives to B, B gives to C, C gives to D and D gives to A, which makes everyone a sender AND a receiver
	RANDOM_DICTATOR	Subjects A, B, C, D are matched in groups of 2 (say (A; B) and (C; D). One member of each group (say A and C) is selected to be the sender, the other one as receiver. In this case, B receives from A and D receives from C
precision	TWO_DIGITS_AFTER_POINT	values on sliders are rounded to two digits after decimal point
	INTEGERS	values on sliders are rounded to integers
scale	$(0, +\infty]$	Parameter to scale up (> 1) or down (< 1) all the numbers on a slider. Default is 1, resulting in a circle of diameter 100
debug	{1; 0}	set to 1 to display some debug info; set to 0 while running actual sessions

Table A.2
Relevant variables in the subjects table of the z-Tree implementation.

Relevant variables in the subjects table	Description	Relevant for parameters
Subject	Unique identifier for the Subject	Always
Group	Matching group of the Subject	matching = RANDOM_DICTATOR
Profit	Profit of the Subject	Always
input_self[i]	Allocation to self in item i	Always
input_other[i]	Allocation to other in item i	Always
random_order[i]	Order of item i in RANDOM order	items_in_random_order = RANDOM
chosen_option[i]	Chosen option on item i, counted from left to right (leftmost option = 1, rightmost option = 9)	Always
mean_to_self	Mean allocation to self in primary items	Always
mean_to_other	Mean allocation to other in primary items	Always
svo_angle	svo angle calculated as $\arctan\left(\frac{\texttt{mean\_to\_other}-50*\texttt{scale}}{\texttt{mean\_to\_self}-50*\texttt{scale}}\right)$	always
svo_type	svo type, 1 = Altruist ⇔ svo_angle > 57.15, 2 = Prosocial ⇔ 57.15 ≥ svo_angle > 22.45, 3 = Individualist ⇔ 22.45 ≥ svo_angle > −12.04, 4 = Competitive ⇔ svo_angle ≤ −12.04	Always
alpha	α-level of the Subject as calculated from the svo_angle and used in Eq. (2)	always
avg_dist_to_equality	Average standardized distance of the choice to the choice which would maximize equality	select_items = FULL
avg_dist_to_altruist	Average standardized distance of the choice to the choice which would maximize altruism	select_items = FULL
avg_dist_to_joint	Average standardized distance of the choice to the choice which would maximize joint earnings	select_items = FULL
avg_dist_to_indiv	Average standardized distance of the choice to the choice which would maximize individual gains	select_items = FULL
not_altru_indiv	Dummy taking the value 1 if avg_dist_equality, avg_dist_joint > avg_dist_altruist, avg_dist_indiv	select_items = FULL
inequality_aversion_score	Degree of inequality aversion calculated as $\frac{\texttt{avg\_dist\_to\_equality}}{(\texttt{avg\_dist\_to\_equality}+\texttt{avg\_dist\_to\_joint})}$ if not_altru_indiv == 1, else −99)	select_items = FULL
paid_slider	Slider selected for payment as sender	Always
slider_as_receiver	Slider selected for payment as receiver	matching = RING
kept_of_sender	Amount kept by the sender	always
received_from_sender	Amount received from the sender	always
kept_as_sender	Amount kept as sender	matching = RING
sent_as_sender	Amount sent as sender	matching = RING

Table A.3
Parameters in the oTree treatment.

Parameter	Values	Description
language	en	Language is English
	de	Language is German
	it	Language is Italian
	fr	Language is French
select_items	PRIMARY	only the primary items (items 1–6) are elicited
	FULL	primary and secondary items (items 1–15) are elicited
items_in_random_order	ORDERED	items are presented according to the order in Murphy et al. (2011)
	RANDOM	items are presented in random order
matching	RING	Subject A, B, C, D are ordered on a ring-structure as in Murphy et al. (2011). In this case, A gives to B, B gives to C, C gives to D and D gives to A, which makes everyone a sender AND a receiver
	RANDOM_DICTATOR	Subjects A, B, C, D are matched in groups of 2 (say (A; B) and (C; D). One member of each group (say A and C) is selected to be the sender, the other one as receiver. In this case, B receives from A and D receives from C
precision	TWO_DIGITS_AFTER_POINT	values on sliders are rounded to two digits after decimal point
	INTEGERS	values on sliders are rounded to integers
scale	$(0, +\infty]$	Parameter to scale up (> 1) or down (< 1) all the numbers on a slider. Default is 1, resulting in a circle of diameter 100
random_payoff	RAND	the payoff will be calculated as a random choice
	SUM	the payoff is the sum of all choices of the player
item_initialization	RAND	initialize the items in random manner
	AVG	initialize the items using average between min and max

Table A.4
Relevant variables in the subjects table of the oTree implementation.

Relevant variables in the subjects table	Description	Relevant for parameters
payoff	Profit of the Subject	Always
input_self_X	The amount of money that the user has chosen for himself for item X	Always
input_other_X	The amount of money that the user has chosen for others for item X	Always
random_orderX	order of item X in RANDOM order	items_in_random_order = RANDOM
mean_to_self	mean allocation to self in primary items	Always
mean_to_other	mean allocation to other in primary items	Always
svo_angle	svo angle calculated as $\arctan\left(\frac{\texttt{mean\_to\_other}-50*\texttt{scale}}{\texttt{mean\_to\_self}-50*\texttt{scale}}\right)$	always
svo_type	svo type, 1 = Altruist ⇔ svo_angle > 57.15, 2 = Prosocial ⇔ 57.15 ≥ svo_angle > 22.45, 3 = Individualist ⇔ 22.45 ≥ svo_angle > −12.04, 4 = Competitive ⇔ svo_angle ≤ −12.04	Always
alpha	α-level of the Subject as calculated from the svo_angle and used in Eq. (2)	Always
avg_dist_to_equality	Average standardized distance of the choice to the choice which would maximize equality	select_items = FULL
avg_dist_to_altruist	Average standardized distance of the choice to the choice which would maximize altruism	select_items = FULL
avg_dist_to_joint	Average standardized distance of the choice to the choice which would maximize joint earnings	select_items = FULL
avg_dist_to_indiv	Average standardized distance of the choice to the choice which would maximize individual gains	select_items = FULL
altru_indiv	Dummy taking the value true if avg_dist_equality, avg_dist_joint ≤ avg_dist_altruist, avg_dist_indiv	select_items = FULL
inequality_aversion_score	Degree of inequality aversion calculated as $\frac{\texttt{avg\_dist\_to\_equality}}{(\texttt{avg\_dist\_to\_equality}+\texttt{avg\_dist\_to\_joint})}$ if altru_indiv == true, else −99)	select_items = FULL
paid_slider	Slider selected for payment as sender	always
slider_as_receiver	Slider selected for payment as receiver	always
kept_of_sender	Amount kept by the sender	always
received_from_sender	Amount received from the sender	always

References

Ackermann, K.A., Fleiß, J., Murphy, R.O., 2016. Reciprocity as an individual difference. J. Conflict Resol. 60 (2), 340–367.

Brandts, J., Charness, G., 2000. Hot vs. cold: Sequential responses and preference stability in experimental games. Exp. Econ. 2 (3), 227–238.

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.

Fiedler, S., Glöckner, A., Nicklisch, A., Dickert, S., 2013. Social value orientation and information search in social dilemmas: An eye-tracking analysis. Organ. Behav. Human Dec. Process. 120 (2), 272–284.

Fischbacher, U., 2007. Z-tree: Zurich toolbox for ready-made economic experiments. Exp. Econom. 10 (2), 171–178.

Murphy, R.O., Ackermann, K.A., Handgraaf, M.J., 2011. Measuring social value orientation.. Judgment Dec. Mak. 6 (8).

Pletzer, J.L., Balliet, D., Joireman, J., Kuhlman, D.M., Voelpel, S.C., Van Lange, P.A., 2018. Social value orientation, expectations, and cooperation in social dilemmas: A meta-analysis. Eur. J. Personal. 32 (1), 62–83.

Rauhut, H., Winter, F., 2010. A sociological perspective on measuring social norms by means of strategy method experiments. Soc. Sci. Res. 39 (6), 1181–1194.

Soraperra, I., Weisel, O., Ploner, M., 2019. Is the victim max (planck) or moritz? how victim type and social value orientation affect dishonest behavior. J. Behav. Decis. Mak. 32 (2), 168–178.

Weisel, O., Zultan, R., 2016. Social motives in intergroup conflict: Group identity and perceived target of threat. Eur. Econ. Rev. 90, 122–133.

Winter, F., 2014. Fairness norms Can explain the emergence of specific cooperation norms in the battle of the prisoner's dilemma. J. Math. Sociol. 38 (4), 302–320.