ReBAC
Revista Brasileira de Análise do Comportamento
Brazilian Journal of Behavior Analysis

ISSN 1807-8338 - Versão Impressa | ISSN 2526-6551 - Versão Eletrônica
periodicos.ufpa.br/index.php/rebac

Pymts: A Python Matching-to-Sample Software

Pymts: Um Software De Matching-to-Sample Baseado em Python

Filipe C. Carvalho[1]
Alceu Regaco [1]
Julio C. de Rose [1]

[1]Universidade Federal de São Carlos

Resumo

Matching-to-Sample (MTS) é um procedimento comumente utilizado em treinos e testes de discriminação condicional em pesquisas sobre equivalência de estímulos. Considerando que o desenvolvimento de softwares intuitivos e acessíveis desempenha um papel crucial na promoção e avanço de pesquisas, e que há uma lacuna na disponibilidade de softwares gratuitos e atualizados de MTS, desenvolvemos o PyMTS, um software de código aberto desenvolvido na linguagem Python. O PyMTS consiste em um arquivo executável e pastas contendo estímulos e configurações de blocos, e oferece uma interface de programação simples para planejar e implementar pesquisas que utilizam o procedimento de MTS. Além disso, a disponibilidade do código do software permite a modificação de variáveis experimentais por pesquisadores com conhecimento em programação em linguagem Python. Esperamos que este software incentive a realização de novas pesquisas, facilitando o trabalho de futuros pesquisadores.

Palavras-chave: matching-to-sample, equivalência de estímulos, software, Python.

Abstract

Matching-to-Sample (MTS) is commonly used in conditional discrimination training and testing. Considering that the development of intuitive and accessible software tools plays a crucial role in promoting and advancing research and that there is a gap in the availability of free and updated MTS software, we have developed PyMTS, an open-source software developed in the Python language. PyMTS consists of an executable file and folders containing stimuli and block configurations, and it offers a simple programming interface for planning and implementing research that utilizes the MTS procedure. Additionally, the availability of the software's code allows for the modification of experimental variables by individuals with knowledge of Python programming. We hope that this software will encourage the conduct of new research, thereby facilitating the work of researchers.

Keywords: matching-to-sample, Python, software, stimulus equivalence.

Notas. The two first authors were supported by the Brazilian Coordination for the Improvement of Higher Education Personnel (CAPES; Finance Code 88887.486112/2020-00, and Finance Code 88887.663283/2022-00), and the second author was supported by the São Paulo Research Foundation (FAPESP; Grant #2021/09616-0) This project is part of the scientific program of Instituto Nacional de Ciência e Tecnologia sobre Comportamento Cognição e Ensino (INCT- ECCE), supported by São Paulo Research Foundation (FAPESP: Grant #2014/50909-8), by the Brazilian Council for Scientific and Technological Development (CNPq: grant #465686/2014-1), and by Coordination for the Improvement of Higher Education Personnel (CAPES: grant #88887.136407/2017-00). We thank the editor Mirela Cengher for the careful reading and review of the manuscript.

✉ filipehora@gmail.com

DOI: http://dx.doi.org/10.18542/rebac.v20i1.16401

Matching-to-sample (MTS) is commonly used in behavior analysis and cognitive experiments (Anderson & Colombo, 2022). The task typically involves presenting a sample stimulus. An observing response to the same stimulus, such as clicking on it, is typically followed by the presentation of at least two comparison stimuli. Correct and incorrect comparison stimulus selection results in the presentation of differential consequences. This sequence is called a “trial” and is repeated until a predetermined criterion is met. MTS has been used for a long time in cognitive research (Hively, 1962); however, in behavior analysis, MTS became of fundamental importance for stimulus equivalence research (Sidman, 1994).

Sidman and Tailby (1982) argued that stimulus equivalence could be described based on the properties of mathematical set theory. According to the authors, after teaching relations between stimuli (e.g., A → B, and B → C), equivalence would be demonstrated if the learners responded in accordance with reflexivity (e.g., A → A, B → B, and C → C), symmetry (e.g., B → A, and C → B), and transitivity (e.g., A → C). Stimulus equivalence became one of the major experimental accounts of language within behavior analysis (Clayton & Hayes, 1999), has generated a broader understanding of symbolic meaning (see Sidman, 1994, for a full book treatment on stimulus equivalence), has been extensively applied on the development of reading abilities (e.g., de Albuquerque & de Melo, 2023; Pilgrim, 2020), and in education more generally (e.g., Brodsky & Fienup, 2018; Fienup & Brodsky, 2020).

In the first experiment on stimulus equivalence, Sidman and his colleagues used a MTS machine to teach reading with comprehension to a teenage boy with developmental disabilities (Sidman, 1971).The machine was composed of nine translucent windows arranged in a 3x3 matrix and disposed in a panel (Stoddard & Sidman, 1966). Other researchers, without access to mechanical equipment, printed the stimuli used in the MTS procedure and glued them onto sheets of paper (de Rose et al., 1996). Each page presented one trial and was placed in a plastic bag, inside a binder. Two experimenters had to conduct the procedures using this equipment: one presented the trials and provided feedback, and the other recorded responses. In this initial experiment, the apparatus, stimuli, and data registration were analog, requiring several efforts to conduct the experiment and the data analysis.

With the advance of computing and the availability of personal computers, several software programs were developed to automate the MTS procedure, such as Mestre (Goyos & Almeida, 1994), Matching to Sample Program (Dube & Hiris, 1997), Match to Sample Program III (Wallace, 2003), and Contingência Programada (Hanna et al., 2014). The creation of these programs was an important advance that facilitated planning and conducting the experiments: the procedure and data registration became entirely automatic. Without the need for human interference during the experiment and the data recording, research and application of the MTS could be done faster, easier, and without the risk of human error. The creation of computerized software allowed, for example, the development of stimulus-equivalence programs that have helped more than 3,000 children learn reading with comprehension (de Albuquerque & de Melo, 2023; Pilgrim, 2020).

However, none of the cited programs were updated to recent programming languages and computer development. A program that stands as an alternative and is broadly used in experimental psychology is PsychoPy (Peirce et al., 2019; Peirce, 2007). This program has frequently been updated in the past years, has an open-source programming code, and is free to use. However, it is a software designed for general psychological experiments and its usability to MTS procedures depends on the experimenter's programming skills. Additionally, due to the frequent updates and its purpose for general psychology experiments, PsychoPy requires more computer performance to start and run experimental programs and can encounter issues with different software versions (a PsychoPy file created in a different version may not work depending on the program's version).

Therefore, although there are some available options, there is a lack of modern, flexible, and intuitive software to facilitate the implementation of MTS procedures. Aiming to promote research using MTS procedures, we developed the software PyMTS. It is an executable file (which means it does not need to be installed on the computer) and it is programmed using the Python language, a common programming language that is easy to learn and use (Sanner, 1999). In this paper, we describe some aspects and functionalities of PyMTS[1] and encourage researchers and students to use it in experiments and applied contexts (e.g., clinical applications).

PyMTS Software Configurations

PyMTS is a MTS software for Windows computers, developed based on PsychoPy (Peirce et al., 2019; Peirce, 2007) and on other MTS software programs (Dube & Hiris, 1997; Wallace, 2003). The manipulation of the program procedure can be divided into general configurations and trial configurations. *General configurations* consist of variables such as screen color, intertrial interval, trial-block names, trial-block instructions, comp display (Delayed or

[1] The instructions for using PyMTS software are available as appendix in both English and Portuguese. If you want to download a recent version of both the software and the instructions, visit PyMTS page on GitHub: https://github.com/AlceuRegaco/PyMTS, or on ResearchGate: https://www.researchgate.net/publication/373277794_PyMTS_vs204.

Simultaneous MTS), number of repetitions, start block, and mastery criterion (see “Configuring the experiment” in the manual). The *screen color* variable represents the background color that will be presented throughout the experiment. The *intertrial interval* refers to the time elapsed between the end of a trial and the start of a new trial. During this interval, the screen will be filled with the background color. The *trial-block names* can be written as preferred, but it is advisable to name the blocks based on what will be presented. For example, if the block involves training the relation of stimuli A1 and B1, and the relation of A2 and B2, a good name would be “AB training”. *Trial-block instructions* are the presentation of any type of instructions before a block of training or testing (an instruction might be given before the first test block stating that differential consequences will not be presented, as an example). The *comp display* refers to the time elapsed between clicking on the sample stimulus and the presentation of the comparison stimuli. There are two options for displaying the comparison stimuli: simultaneous and delayed. In *Simultaneous MTS (SMTS)*, the sample stimulus remains on the screen when the comparison stimuli are presented, and all stimuli are withdrawn after a selection response. In *Delayed MTS (DMTS)*, the sample stimulus is withdrawn (usually after an observing response, such as clicking on the sample with the mouse) and comparison stimuli are displayed after a delay, during which the screen is filled with the background color. The protocol and delay duration (for DMTS) can be specified for each block. The *number of repetitions* is related to the maximum number of times each block will be presented if participant’s responding does not reach the mastery criterion, which is specified for each block. If the maximum number of repetitions is reached, a specified message to end the session is presented, and the program is completed. In this case, after reviewing the data, the experimenter can decide whether the participant can continue the experiment. If the experimenter decides that the participant can continue, the researcher can select the block in which the participant must start again using the variable “start block.” This variable controls the start of the experiment, and changing it will determine the first block presented to the participant.

The *trial configurations* are the variables related to the specificities of each block. PyMTS allows for the manipulation of the following variables: sample stimuli (both visual or auditory), comparison stimuli, correct comparison stimuli, consequences for correct comparison selection, consequences for incorrect comparison selection, and duration of consequence presentation (see “Configuring the trial blocks” in the manual). Therefore, for each block, all these variables can be specified for each trial. For example, it is possible to program a training block in which half of the trials will train A1B1 and A2B2 relations, and the other half will train A1B2 and A2B1 relations[2]. Alternatively, you can program specific consequences for each class (A1B1 → Consequence 1 and A2B2 → Consequence 2).

The software saves the following information: experimenter’s name, participant’s name, date, total number of trials, block name, total number of trials for each block, accuracy of each selection, cumulative number of correct comparison stimulus selections (for each block presentation), sample stimulus presented, comparison stimuli presented (in the order they were presented), the selected comparison stimulus, the time elapsed between seeing the sample stimulus and clicking on it, the time between presentation of the comparison stimuli and selection of one of them, and the total trial time (the times are presented in seconds). The file is saved in .CSV (comma-separated values) format[3].

Advances and Limitations

Considering previous versions of MTS software programs, PyMTS's main advances are its simplicity, the use of a current broadly used programming language, and being an open-source software. PyMTS also offers great flexibility in the number of variables that can be manipulated when planning stimulus equivalence research. For example, it is possible to manipulate the training protocols, training structures, the number of classes, the number of stimuli, the order of presentation of training and testing blocks, the type of stimuli used, and the mastery criteria (Arntzen, 2012). However, its usefulness goes beyond stimulus equivalence basic research: the software can be used in applied identity matching, to fastly program teaching procedures of college-level courses (Fienup & Brodsky, 2020), to program a multiple exemplar training procedure, and it can be adapted to rule-governed behavior procedures with different contingencies for winning or losing points[4].

[2] This is just an example to show the flexibility of the program. Don’t do this unless you have a very specific experimental objective. For most purposes, it is important that to keep the trained relations constant.

[3] The data saved from one test of the software are available in the annex. In this particular data set, the criteria to finish the training blocks was the consecutive number of correct responses (12 consecutive correct trials), as an example of a different application.

[4] A version of PyMTS with this feature was already developed and is available in PyMTS GitHub page.

PyMTS is an open-source software, which means the script can be downloaded and it is free to be used and changed, so that other scientists working with MTS procedures can contribute and help to further develop and modify the software. We invite researchers to read the manual, download the software, and test its functionality and the variables that can be manipulated. The development of good and stable software often relies on the use and feedback from the community. We hope that this software can serve as a starting point for improved communication regarding technological advancements and software development among behavior analytic researchers..

Declaração de conflito de interesses

Os autores declaram que não há conflito de interesses relativos à publicação deste artigo.

Contribuição de cada autor

A contribuição de cada autor pode ser atribuída como se segue: F. C. Carvalho e A. Regaço desenvolveram o software e o manuscrito e J. C. de Rose auxiliou no registro do software, na revisão do manuscrito e o desenvolvimento do projeto como um todo.

Direitos Autorais

Referências

Anderson, C., & Colombo, M. (2022). Matching-to-sample: Comparative overview. In *Encyclopedia of animal cognition and behavior* (pp. 4097–4102). Springer International Publishing. https://doi.org/10.1007/978-3-319-55065-7_1708

Arntzen, E. (2012). Training and testing parameters in formation of stimulus equivalence: Methodological issues. *European Journal of Behavior Analysis*, *13*(1), 123–135. https://doi.org/10.1080/15021149.2012.11434412

Brodsky, J., & Fienup, D. M. (2018). Sidman goes to college: A meta-analysis of equivalence-based instruction in higher education. *Perspectives on Behavior Science*, *41*(1), 95–119. https://doi.org/10.1007/s40614-018-0150-0

Clayton, M. C., & Hayes, L. J. (1999). Conceptual differences in the analysis of stimulus equivalence. *The Psychological Record*, *49*(1), 145–161. https://doi.org/10.1007/BF03395312

de Albuquerque, A. R., & de Melo, R. M. (2023). *Contributions of behavior analysis to reading and writing comprehension*. Springer Nature.

de Rose, J. C., de Souza, D. G., & Hanna, E. S. (1996). Teaching reading and spelling: Exclusion and stimulus equivalence. *Journal of Applied Behavior Analysis*, *29*(4), 451–469. https://doi.org/10.1901/jaba.1996.29-451

Dube, W., & Hiris, J. (1997). Matching to Sample Program (version 11.08) [Computer software]. EK Shriver Center for Mental Retardation.

Fienup, D. M., & Brodsky, J. (2020). Equivalence-based instruction: Designing instruction using stimulus equivalence. In M. Fryling, R. A. Rehfeldt, J. Tarbox, & L. J. Hayes (Eds.), *Applied behavior analysis of language and cognition: Core concepts and principles for practitioners* (1st ed., Vol. 1, pp. 157–173). Context Press.

Goyos, C., & Almeida, J. C. B. (1994). Mestre (Versão 1.0) [Computer software].

Hanna, E. S., Anísio, L., Batitucci, V., Da, J., Lacerda, S., Universidade, B., & Brasília, D. E. (2014). Software Contingência Programada: utilidade e funcionalidades. *Revista Brasileira de Análise Do Comportamento*, *10*(1), 97–104. https://doi.org/http://dx.doi.org/10.18542/rebac.v10i1.3949

Hively, W. (1962). Programming stimuli in matching to sample. *Journal of the Experimental Analysis of Behavior*, *5*(3), 279–298. https://doi.org/10.1901/jeab.1962.5-279

Peirce, J., Gray, J. R., Simpson, S., MacAskill, M., Höchenberger, R., Sogo, H., Kastman, E., & Lindeløv, J. K. (2019). PsychoPy2: Experiments in behavior made easy. *Behavior Research Methods*, *51*(1), 195–203. https://doi.org/10.3758/s13428-018-01193-y

Peirce, J. W. (2007). PsychoPy — psychophysics software in Python. *Journal of neuroscience methods*, *162*(1), 8–13. https://doi.org/10.1016/j.jneumeth.2006.11.017

Pilgrim, C. (2020). Equivalence-based instruction. In J. O. Cooper, T. E. Heron, & W. L. Heward (Eds.), *Applied behavior analysis* (3rd ed., pp. 442–496). Pearson.

Sanner, F. (1999). Python: a programming language for software integration and development Related papers. *J Mol Graph Model*.

Sidman, M. (1971). Reading and auditory-visual equivalence. *Journal of Speech and Hearing Research*, *14*(1), 5–13.

Sidman, M. (1994). *Equivalence relations and behavior: A research story.* Authors Cooperative. https://psycnet.apa.org/record/1994-98777-000

Sidman, M., & Tailby, W. (1982). Conditional discrimination vs. matching to sample: An expansion of the testing paradigm. *Journal of the Experimental Analysis of Behavior*, *37*(1), 5–22. https://doi.org/10.1901/jeab.1982.37-5

Stoddard, L. T., & Sidman, M. (1966). Programming perception and learning for retarded children. In *International review of research in mental retardation* (Vol. 2, pp. 151–208). Academic Press. https://doi.org/10.1016/S0074-7750(08)60206-2

Wallace, B. W. (2003). Match to Sample Program III [Computer software]. UMass/Eunice Kennedy Shriver Center's Behavioral Sciences Department.

Submetido em: 01/11/2023
Aceito em: 20/03/2024